# 6.801/6.866 Machine Vision

## Syllabus

| # | Date | Description | Readings | Assignments | Materials |
|---|---|---|---|---|---|
| 1 | 9/5 | Course Introduction | | Pset #0 (not collected) | Freeman Slides<br>Darrell Slides<br>Matlab Tutorial Diary |
| 2 | 9/10 | Cameras, Lenses, and Sensors | Req: FP 1<br>Opt: H 2.1, 2.3 | | Freeman Slides |
| 3 | 9/12 | Radiometry and Shading Models I | Req: FP 2, 5.4; H 10<br>Opt: FP 4 | Pset #1 Assigned | Freeman Slides |
| 4 | 9/17 | Radiometry and Shading Models II | " | | Freeman Slides |
| 5 | 9/19 | Multiview Geometry | Req: FP 10 | | Darrell Slides |
| 6 | 9/24 | Stereo | Req: FP 11; H 13 | Pset #1 Due | Darrell Slides |
| 7 | 9/26 | Color | Req: FP 6.1-6.4 | Pset #2 Assigned | Freeman Slides |
| 8 | 10/1 | Shape from Shading | Req: H 11.1, 11.5-11.9<br>Opt: H 11.2-11.4 | | Freeman Slides |
| 9 | 10/3 | Image Filtering | Req: FP 7<br>Opt: H 7,8 | | Freeman Slides |
| 10 | 10/8 | Image Representations | Handouts (2) | Pset #2 Due | |
| 11 | 10/10 | Texture | Req: FP 9 | Exam #1 Assigned | |
| | 10/15 | Columbus Day (NO LECTURE) | | | |
| 12 | 10/17 | Bayesian Analysis and Optic Flow | Req: H 12 | | |
| 13 | 10/22 | Direct SFM | Req: H 17 | Exam #1 Due | |
| 14 | 10/24 | Affine Reconstruction | Req: FP 12 | Pset #3 Assigned | |

# Today: non-linear filters, and uses for the filters and representations from last time

- Review pyramid representations
- Non-linear filtering
- Textures

# Reading

- Related to today's lecture:
  - Chapter 9, Forsyth&Ponce..
- For next Thursday's lecture:
  - Horn, Ch. 12
  - Bishop chapter 1 (handout from last lecture)

# Mid-term exam

Problem set 3 given out today

- Open book, open web.
- Work by yourself. This problem set is a mid-term exam, and you can't: talk about it, e-mail about it, give hints, etc, with others.
- Due Tuesday, Oct. 22 (in 12 days).

# Image representations

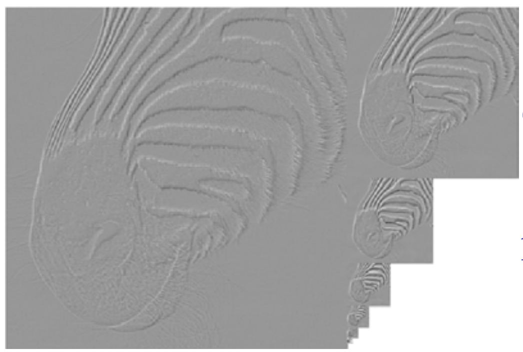
- Fourier basis
- Image pyramids

# Image pyramids

- Gaussian

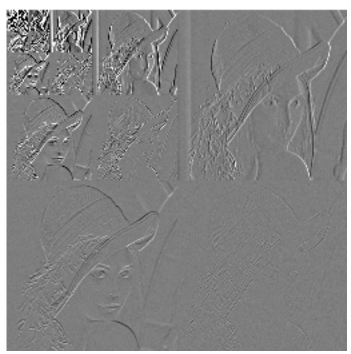
Progressively blurred and subsampled versions of the image. Adds scale invariance to fixed-size algorithms.

- Laplacian



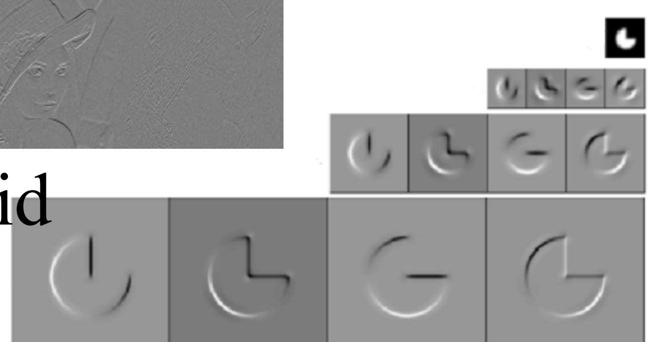
Shows the information added in Gaussian pyramid at each spatial scale. Useful for noise reduction & coding.

- Wavelet/QMF



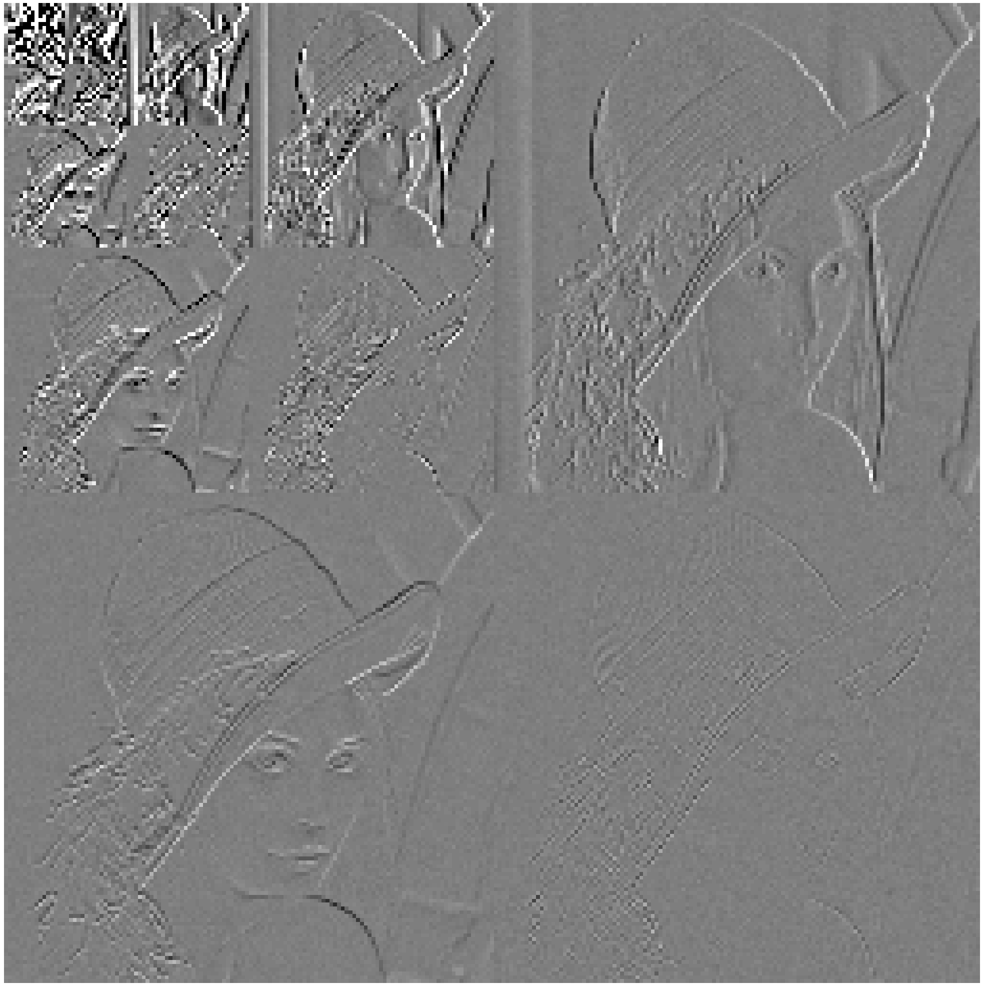
Bandpassed representation, complete, but with aliasing and some non-oriented subbands.

- Steerable pyramid



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

# Wavelet/QMF representation



# Linear image transformations

- In analyzing images, it's often useful to make a change of basis.

transformed image

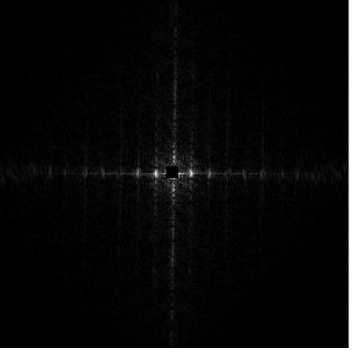
$$\vec{F} = U\vec{f}$$

Vectorized image

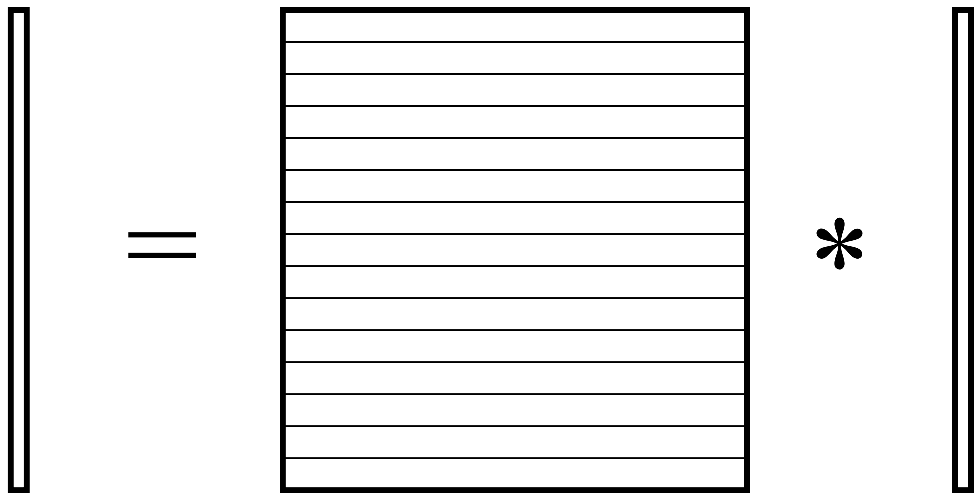
Fourier transform, or
Wavelet transform, or
Steerable pyramid transform

# 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.



# Fourier transform

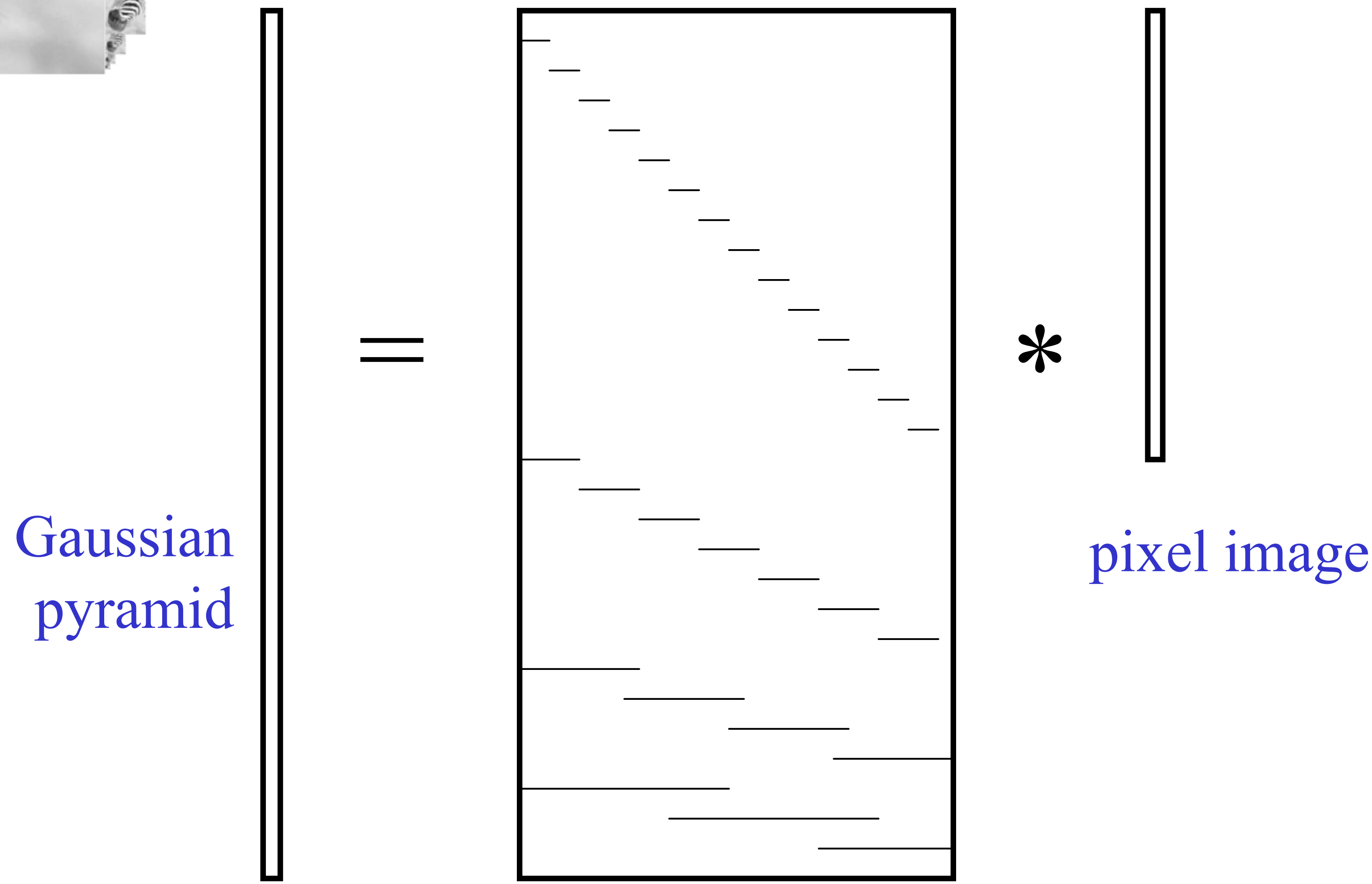


Fourier transform

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

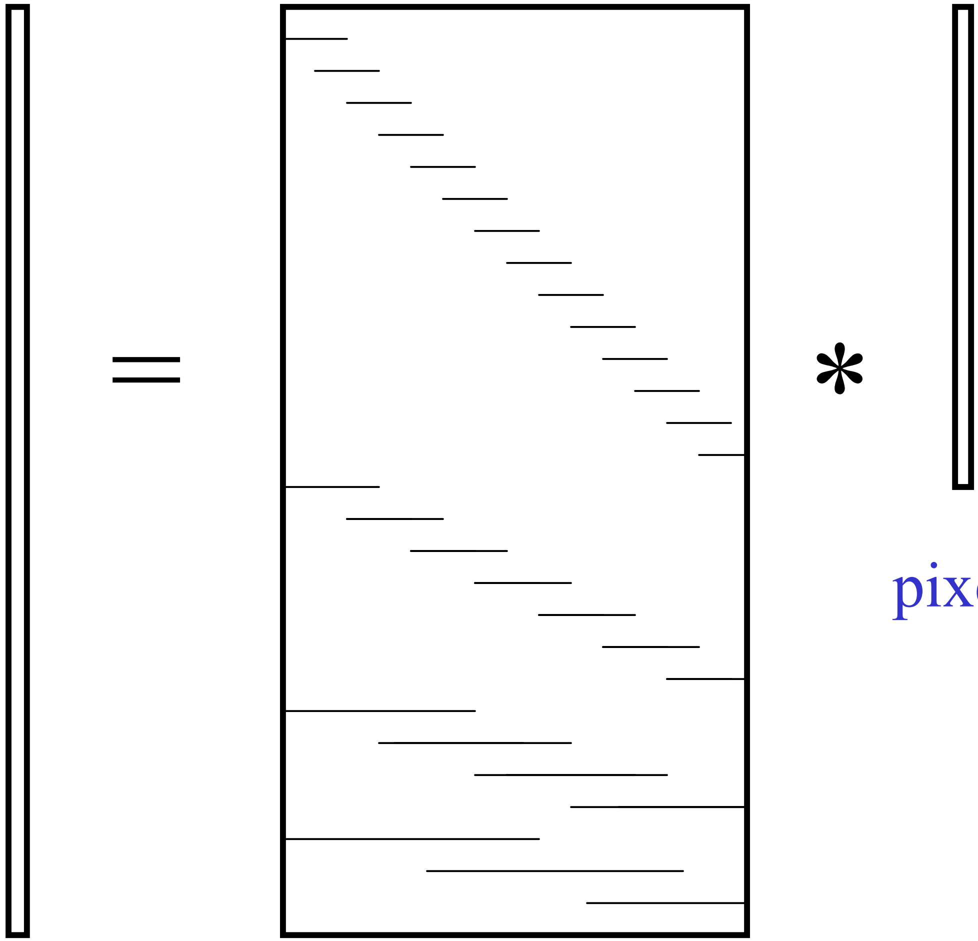
pixel domain image

# Gaussian pyramid




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

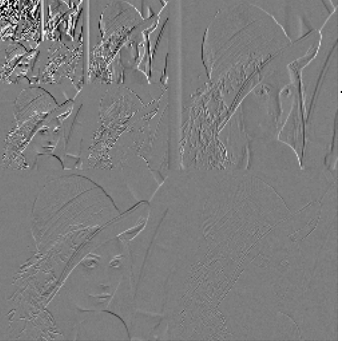
# Laplacian pyramid



Laplacian pyramid

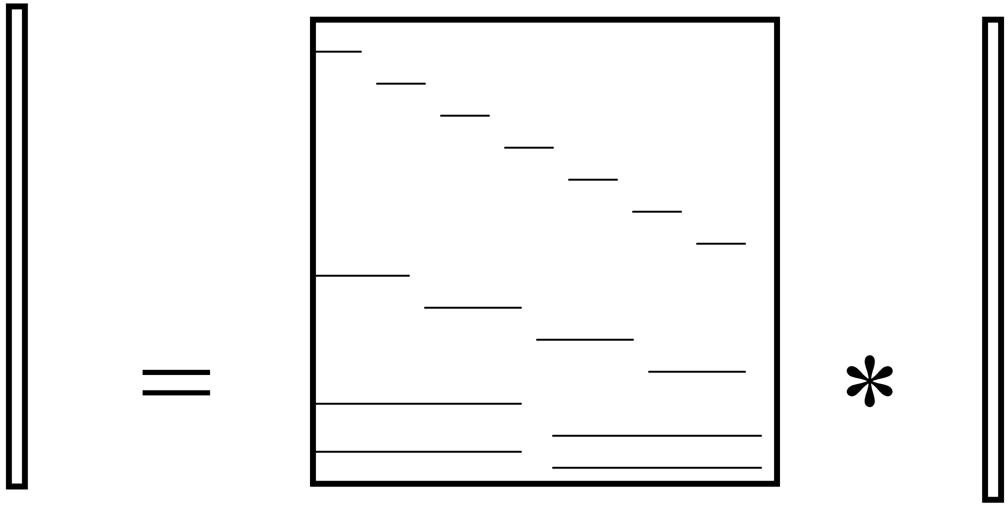
pixel image

Overcomplete representation.
Transformed pixels represent
bandpassed image information.



# Wavelet (QMF) transform

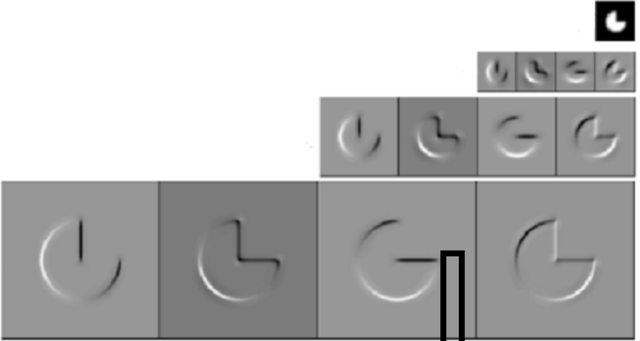
Wavelet pyramid

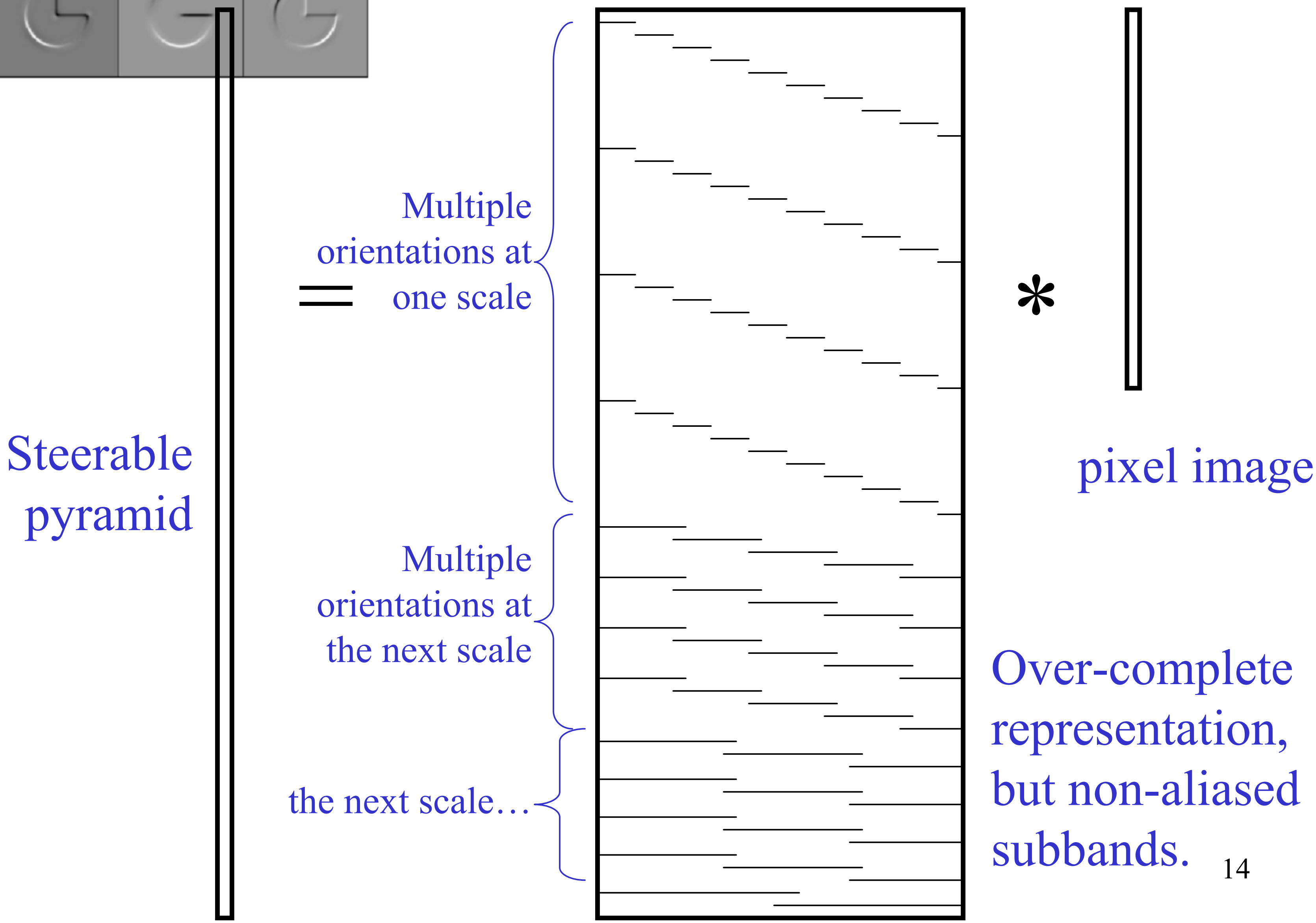


Ortho-normal transform (like Fourier transform), but with localized basis functions.

pixel image

# Steerable pyramid






Over-complete representation, but non-aliased subbands.

# Matlab resources for pyramids (with tutorial)

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

lcv

Laboratory for Computational Vision

Home | People | Research | Publications | Software

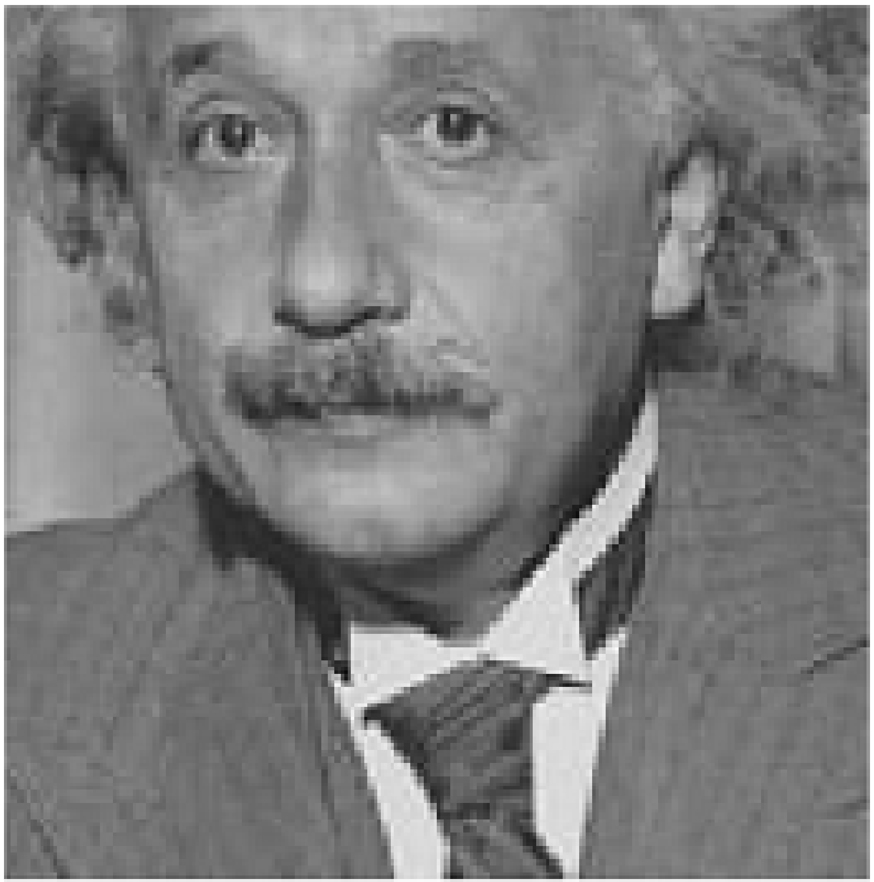
## Publicly Available Software Packages

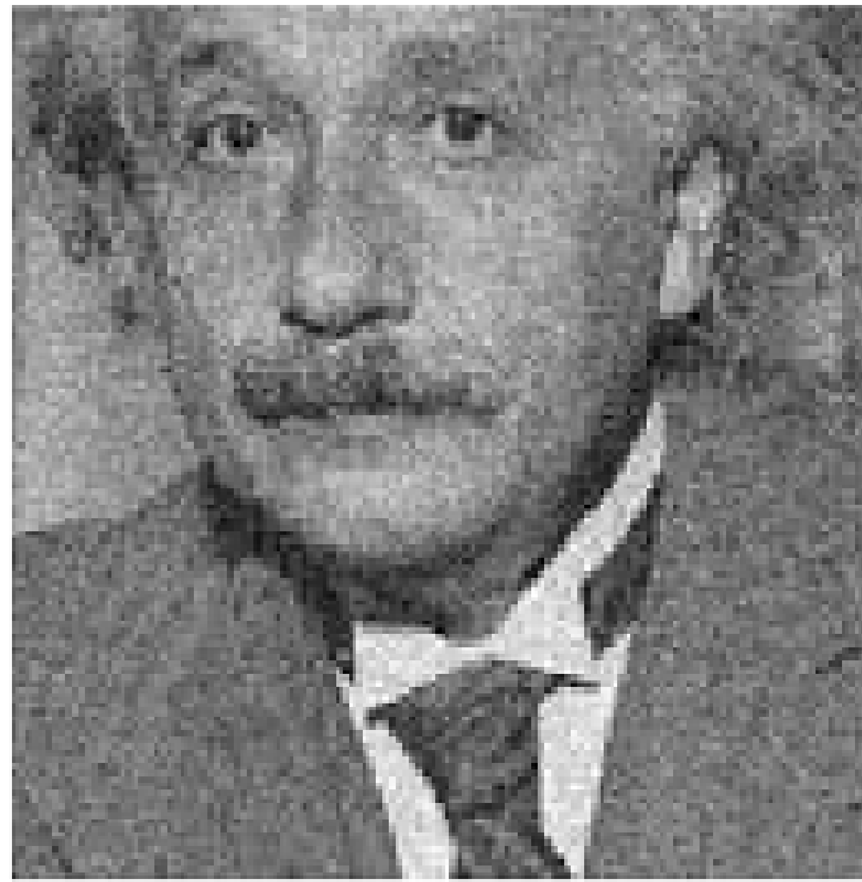
- Texture Analysis/Synthesis - Matlab code is available for analyzing and synthesizing visual textures. README | Contents | ChangeLog | Source code (UNIX/PC, gzip'ed tar file)
- EPWIC - Embedded Progressive Wavelet Image Coder. C source code available.
- **matlabPyrTools** - Matlab source code for multi-scale 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

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





# Bayesian MAP estimator for clean bandpass coefficient values

Let x = bandpassed image value before adding noise.

Let y = noise-corrupted observation.

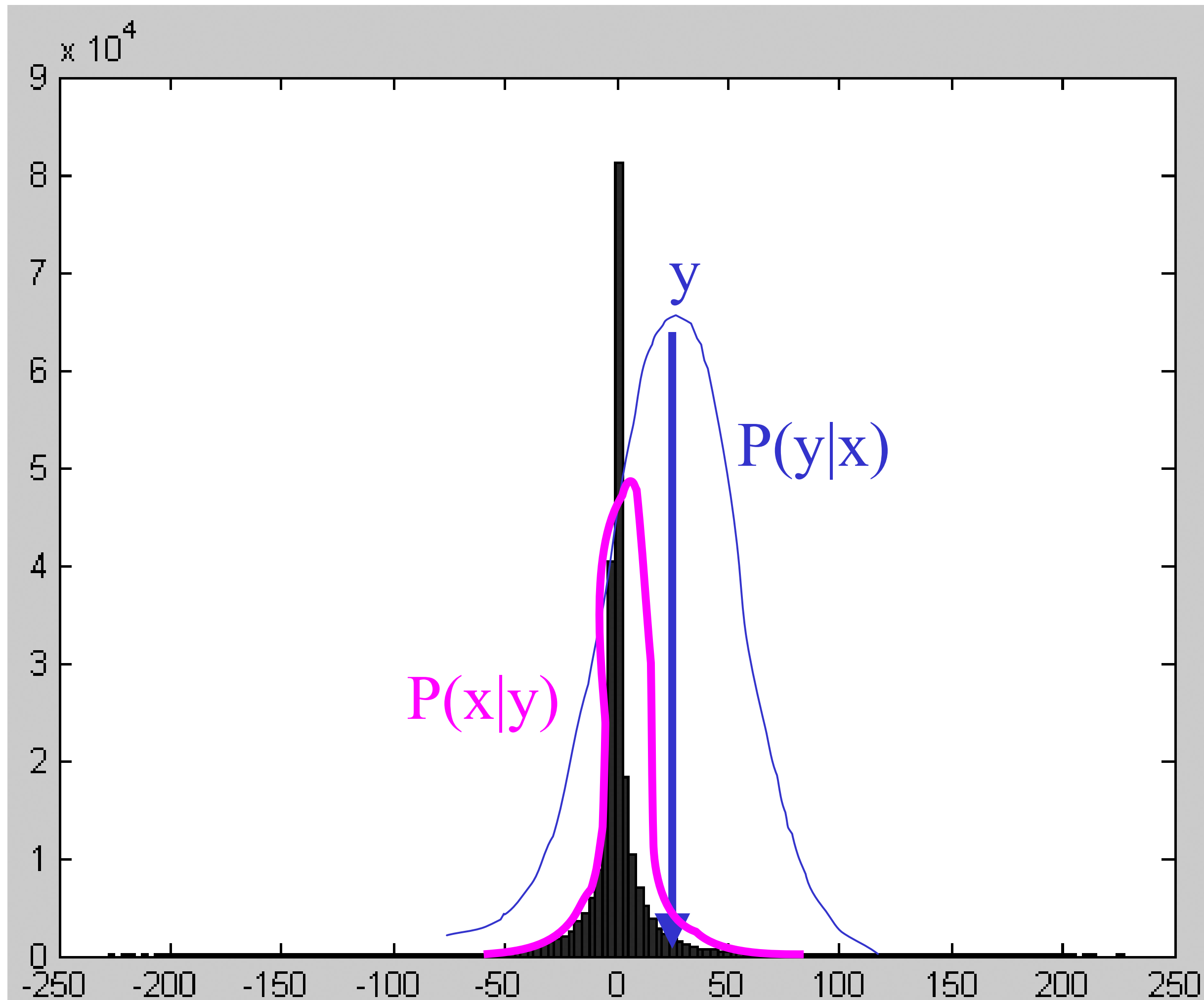
By Bayes theorem

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.

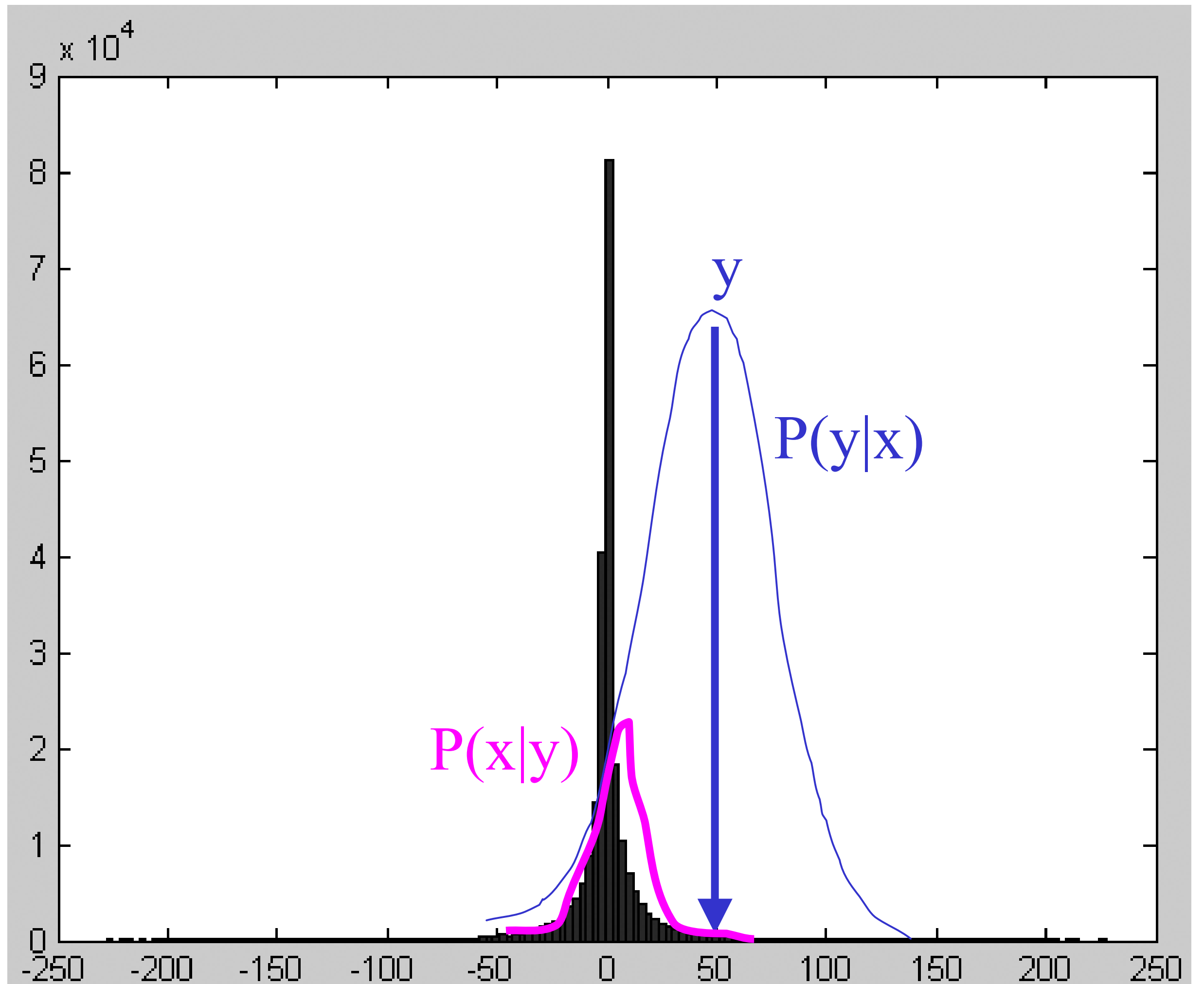
By Bayes theorem

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

P(x)

P(y|x)

P(x|y)



# Bayesian MAP estimator

Let x = bandpassed image value before adding noise.

Let y = noise-corrupted observation.

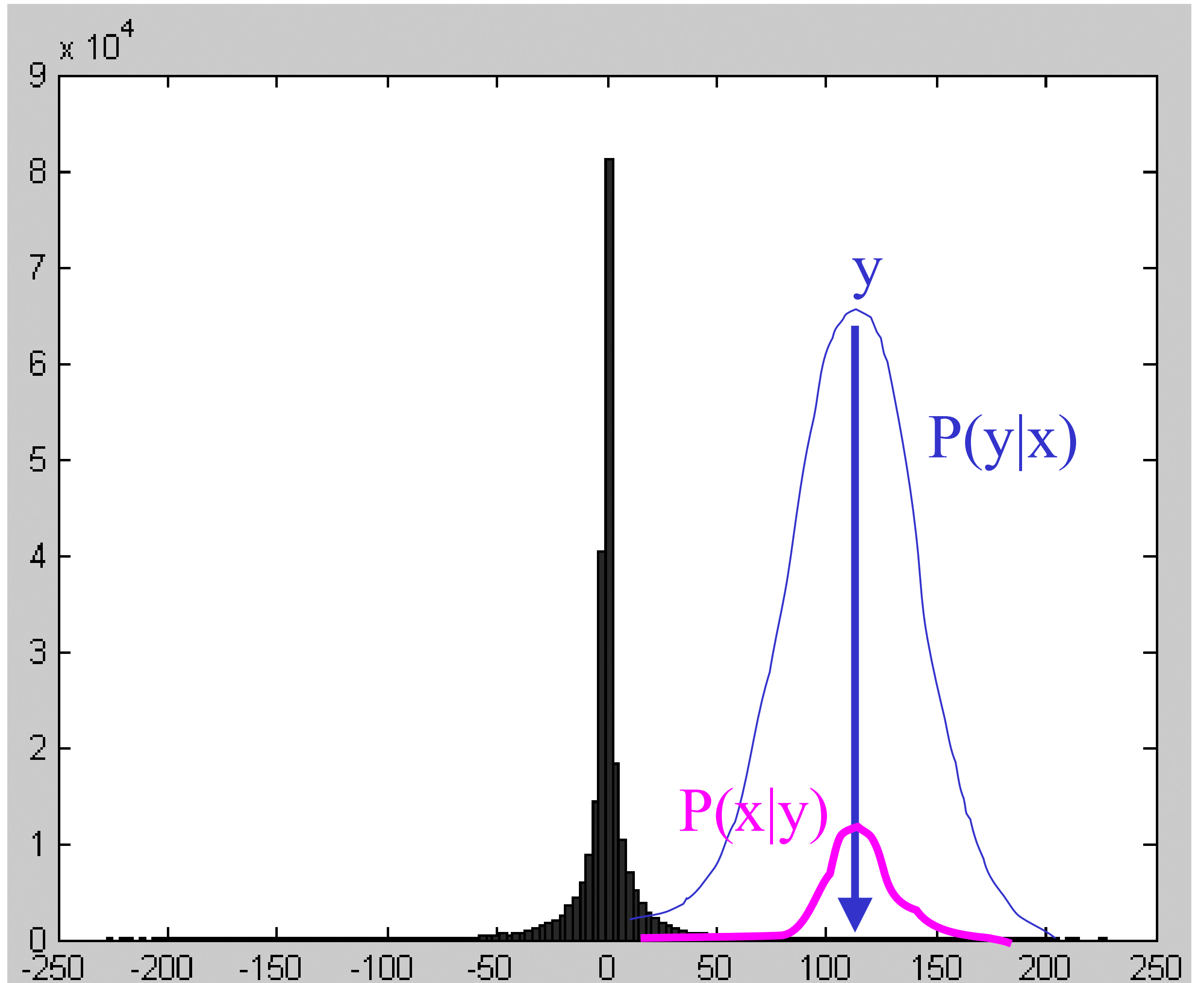
By Bayes theorem

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

P(x)

P(y|x)

P(x|y)



# Noise removal results

(a) (b) (c) (d)

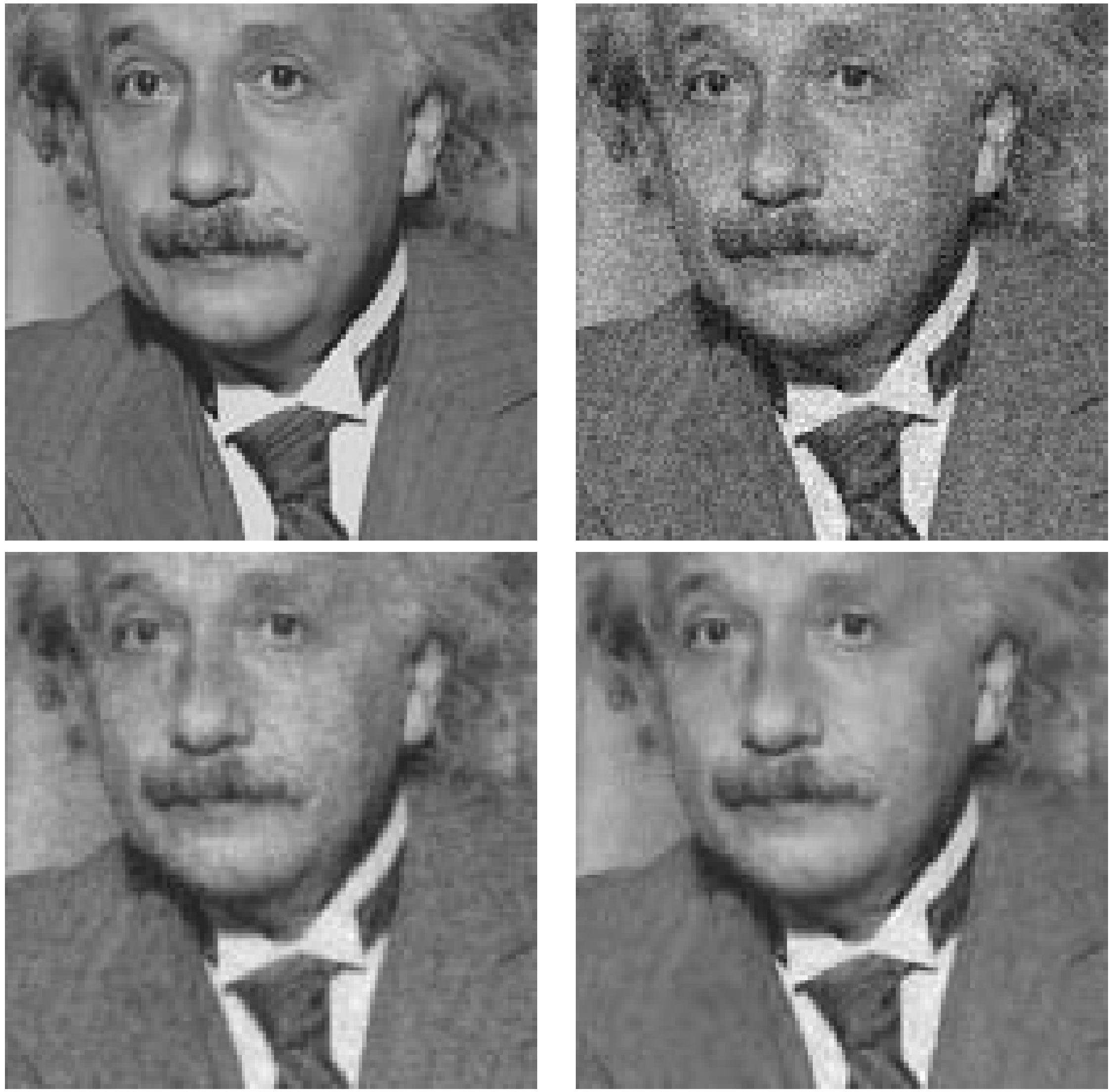


**Figure 4:** Noise reduction example. (a) Original image (cropped). (b) Image contaminated with additive Gaussian white noise (SNR = 9.00dB). (c) Image restored using (semi-blind) Wiener filter (SNR = 11.88dB). (d) Image restored using (semi-blind) Bayesian estimator (SNR = 13.82dB).

**Simoncelli and Adelson, Noise Removal via Bayesian Wavelet Coring**

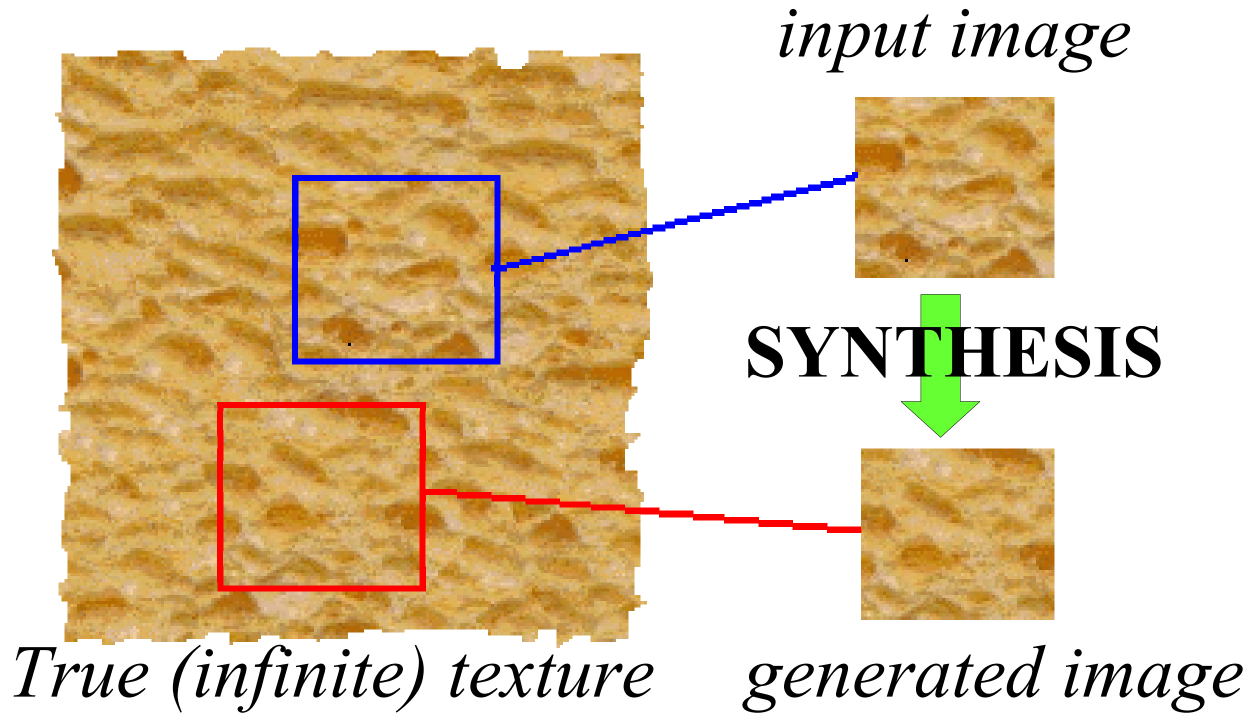
http://www-bcs.mit.edu/people/adelson/pub_pdfs/simoncelli_noise.pdf

# Image texture

# Texture

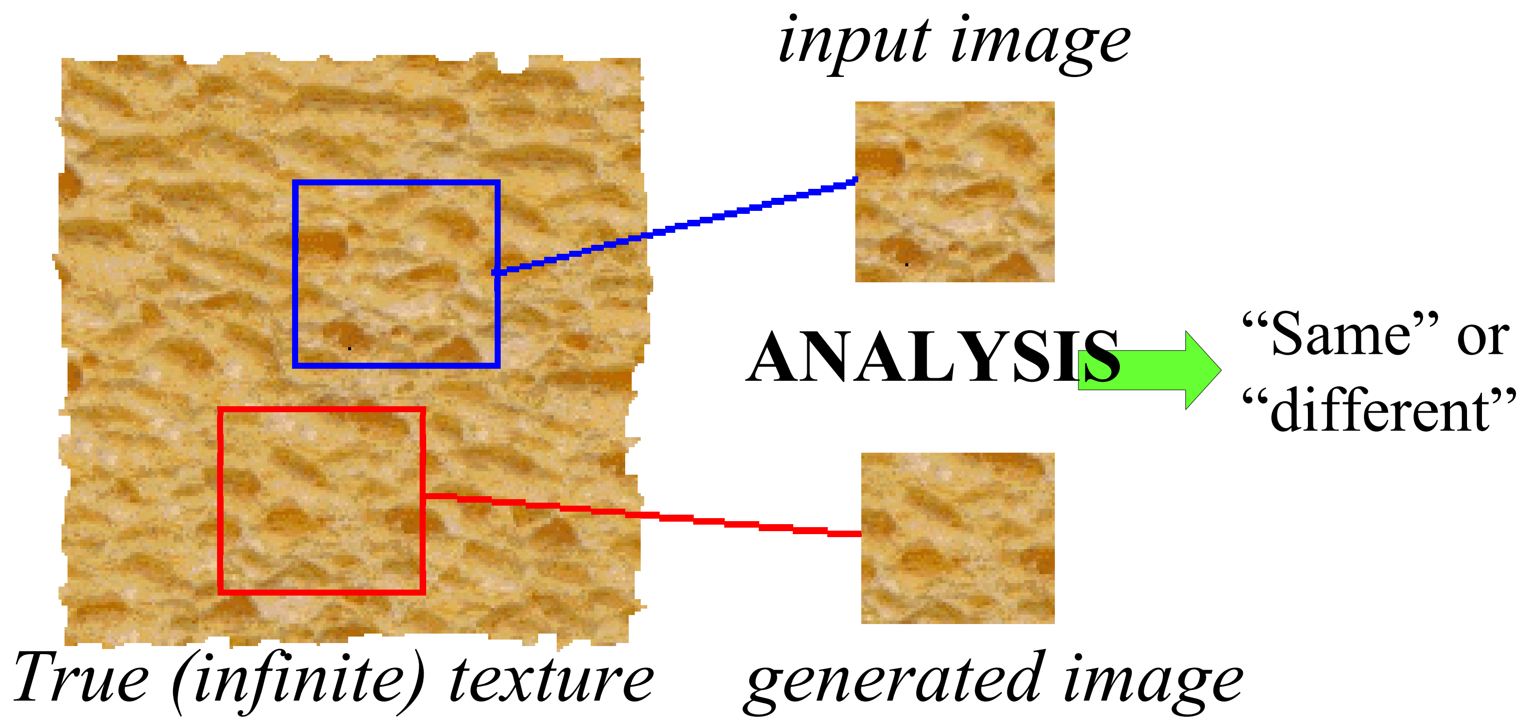
- Key issue: representing texture
  - Texture based matching
    - little is known
  - Texture segmentation
    - key issue: representing texture
  - Texture synthesis
    - useful; also gives some insight into quality of representation
  - Shape from texture
    - cover superficially

# 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“

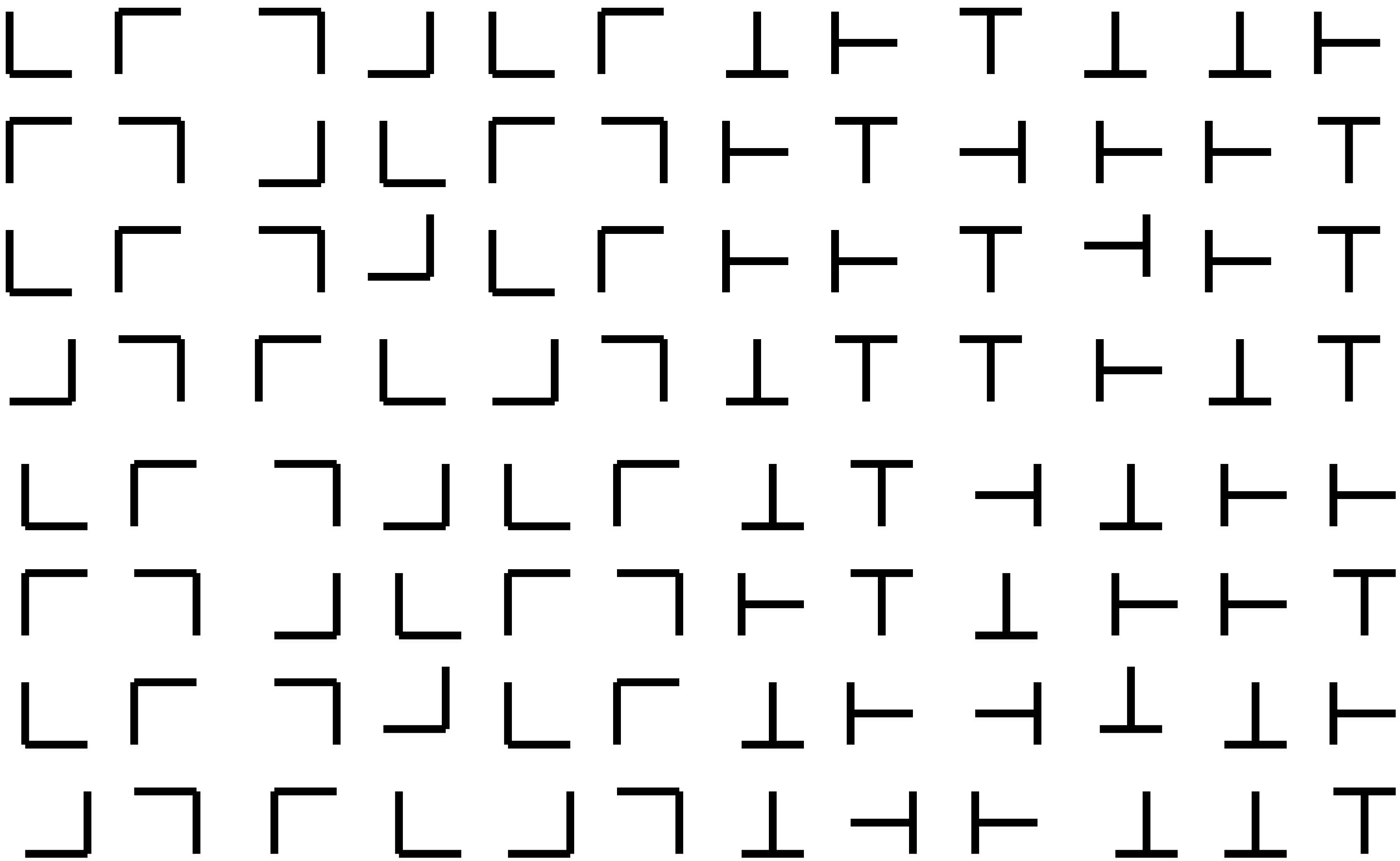
# The Goal of Texture Analysis




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

# Pre-attentive texture discrimination

# Pre-attentive texture discrimination

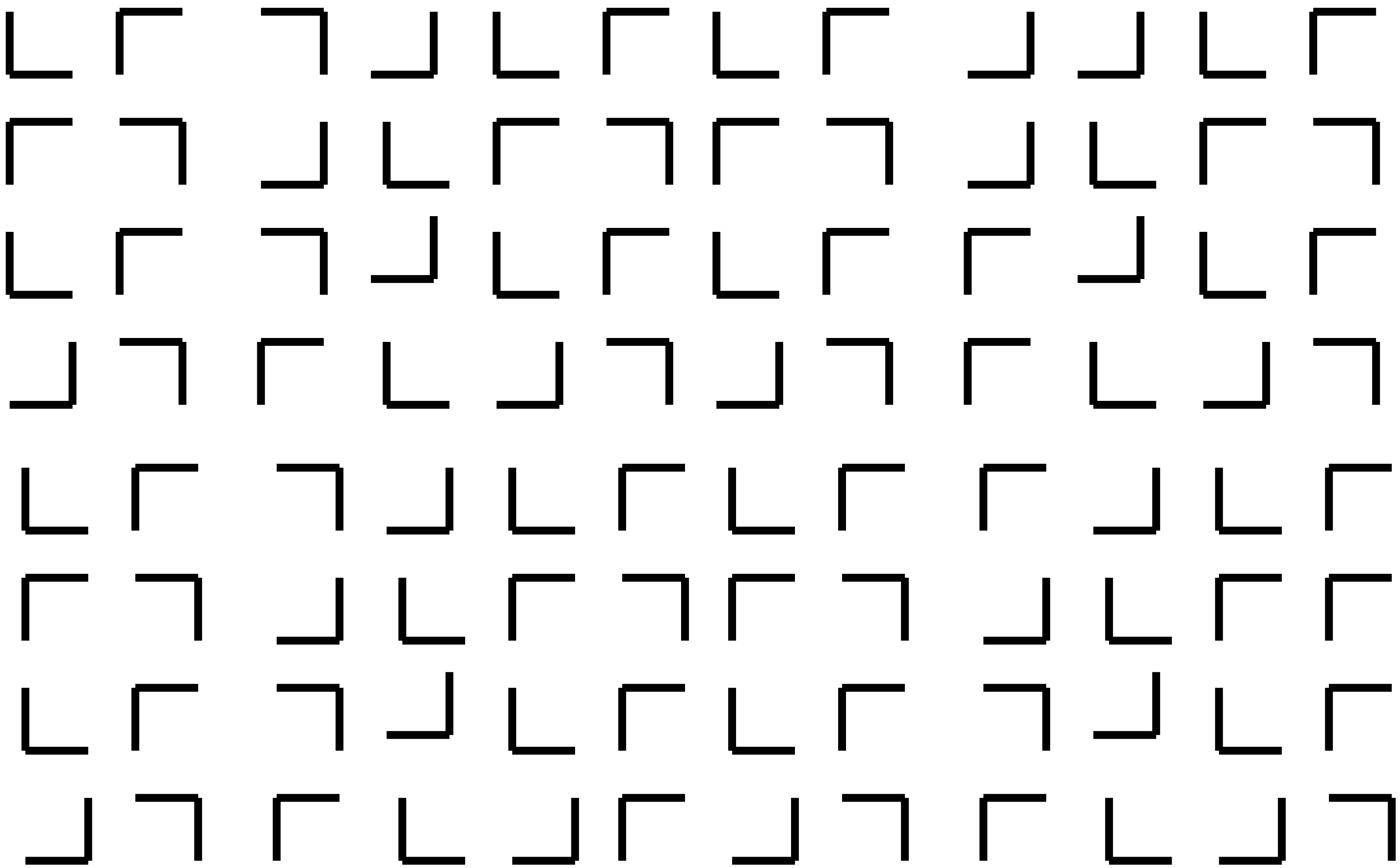


# Pre-attentive texture discrimination

Same or different textures?

# Pre-attentive texture discrimination

# Pre-attentive texture discrimination

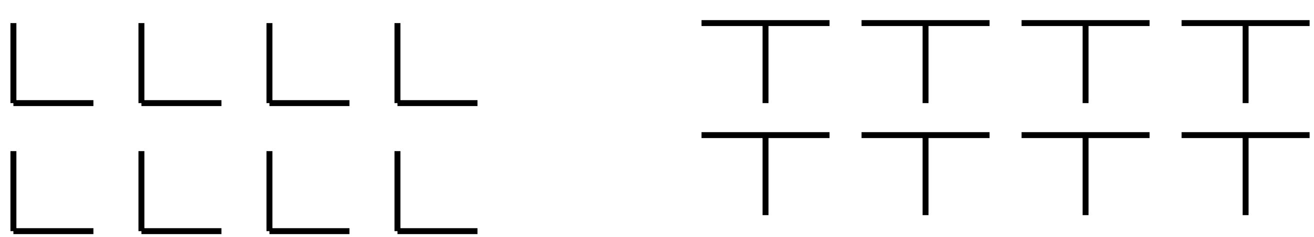


# Pre-attentive texture discrimination

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**

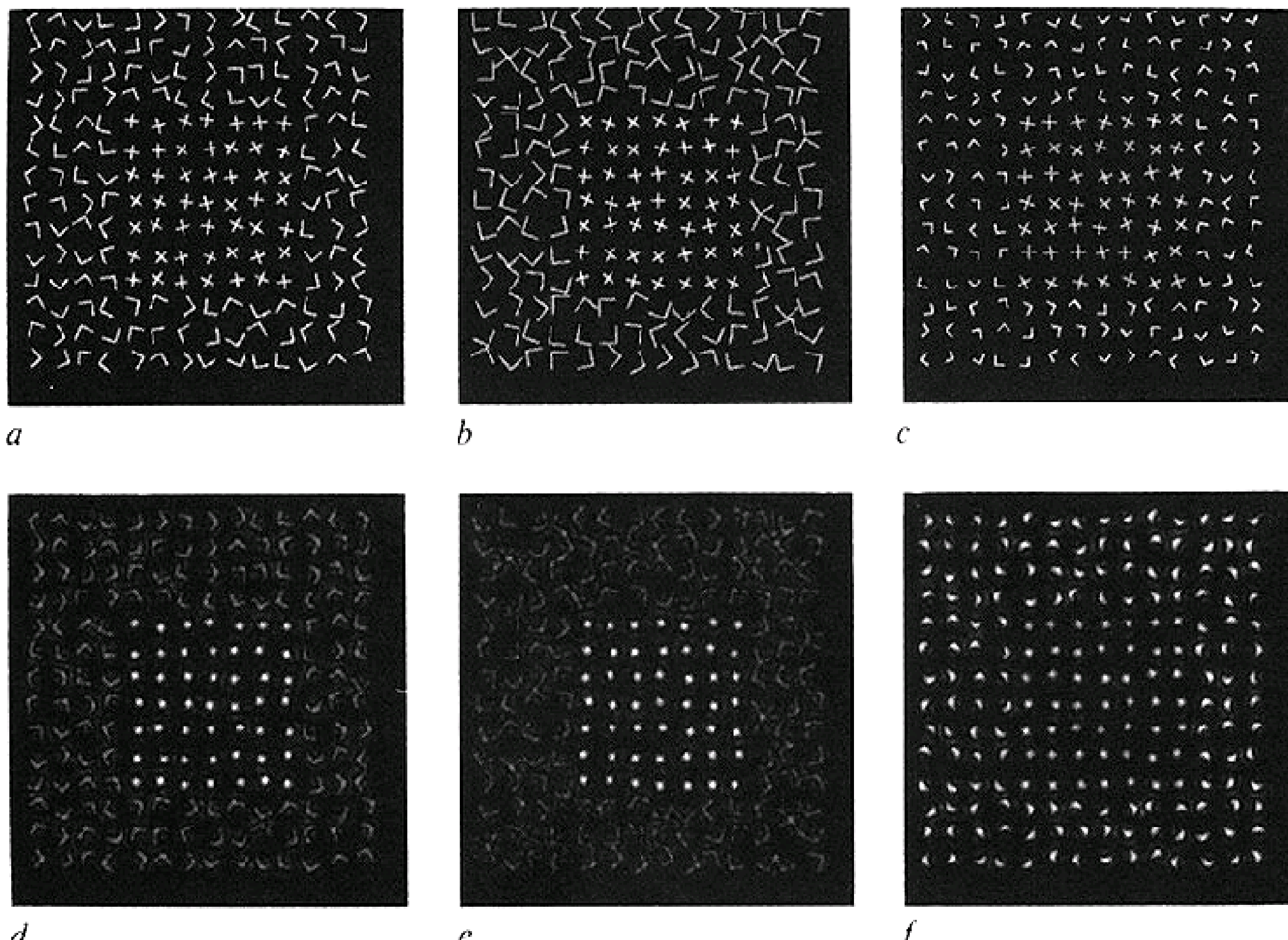
**James R. Bergen* & Edward H. Adelson****

* 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.

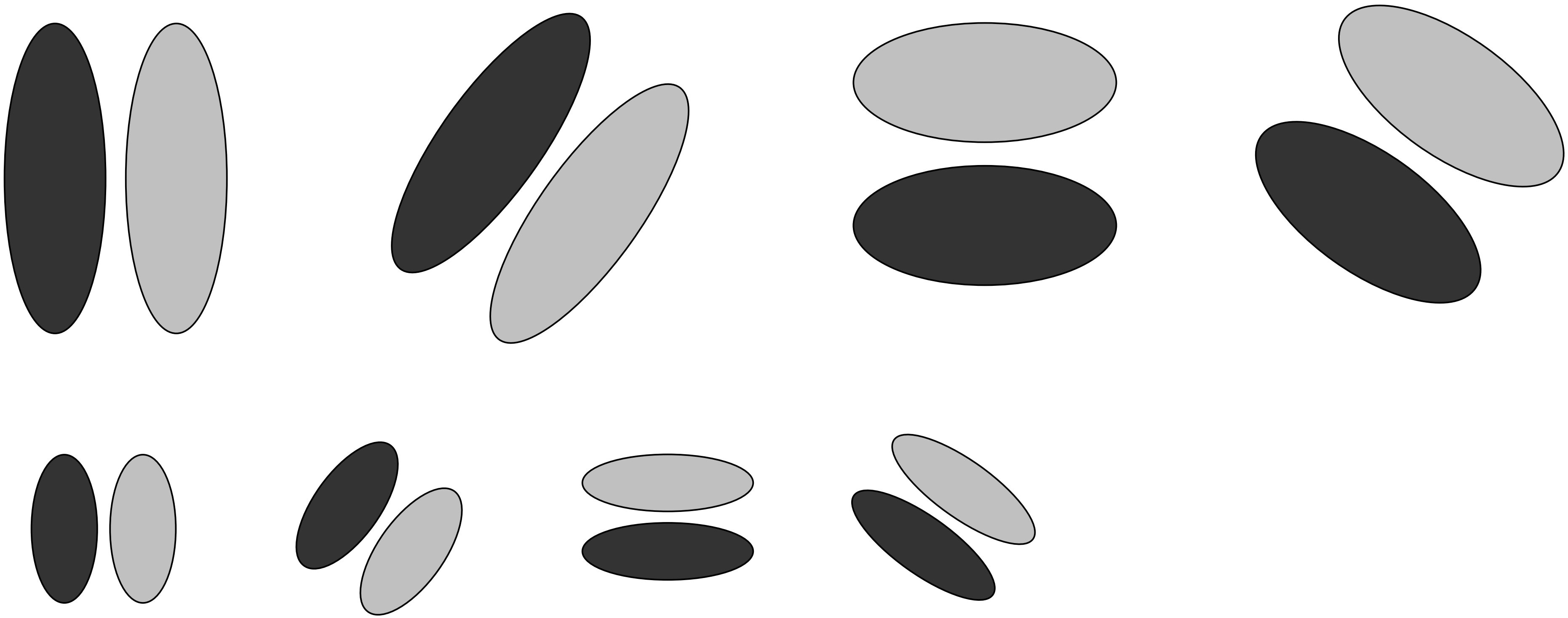
# Bergen and Adelson, Nature 1988

**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.

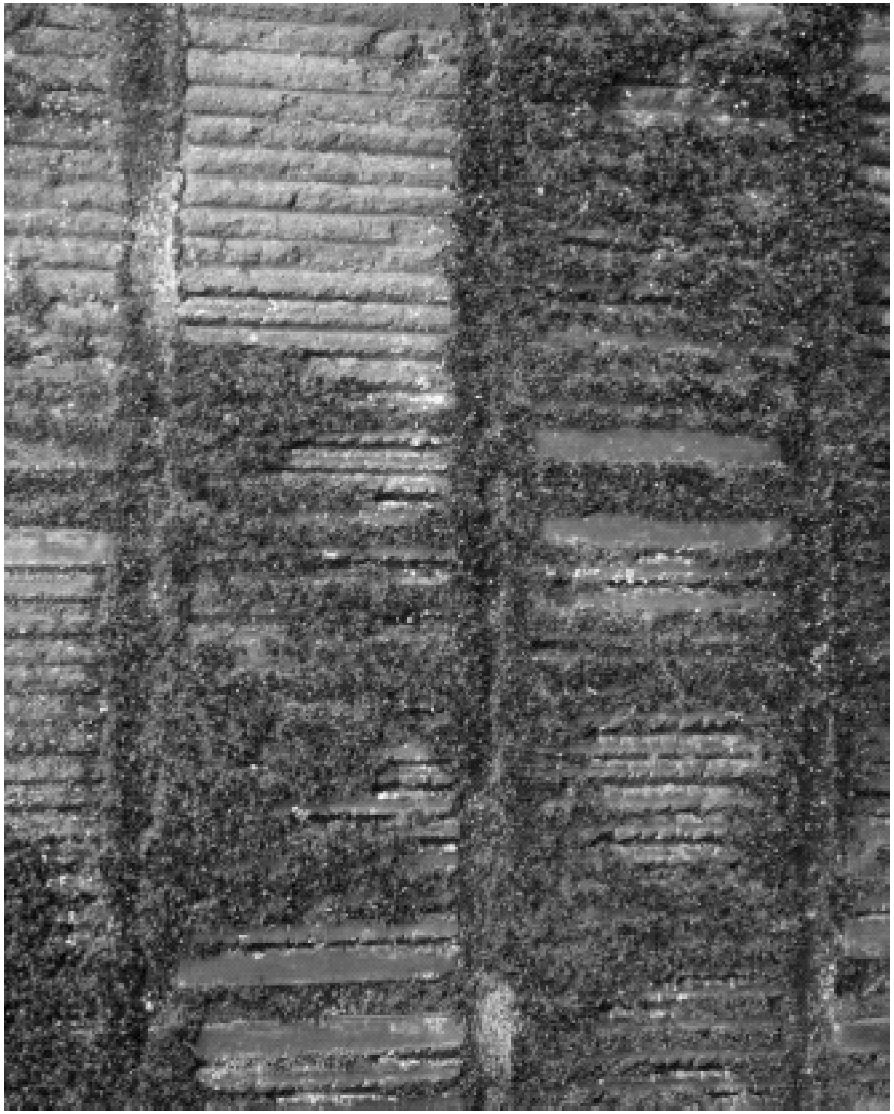
# Malik and Perona

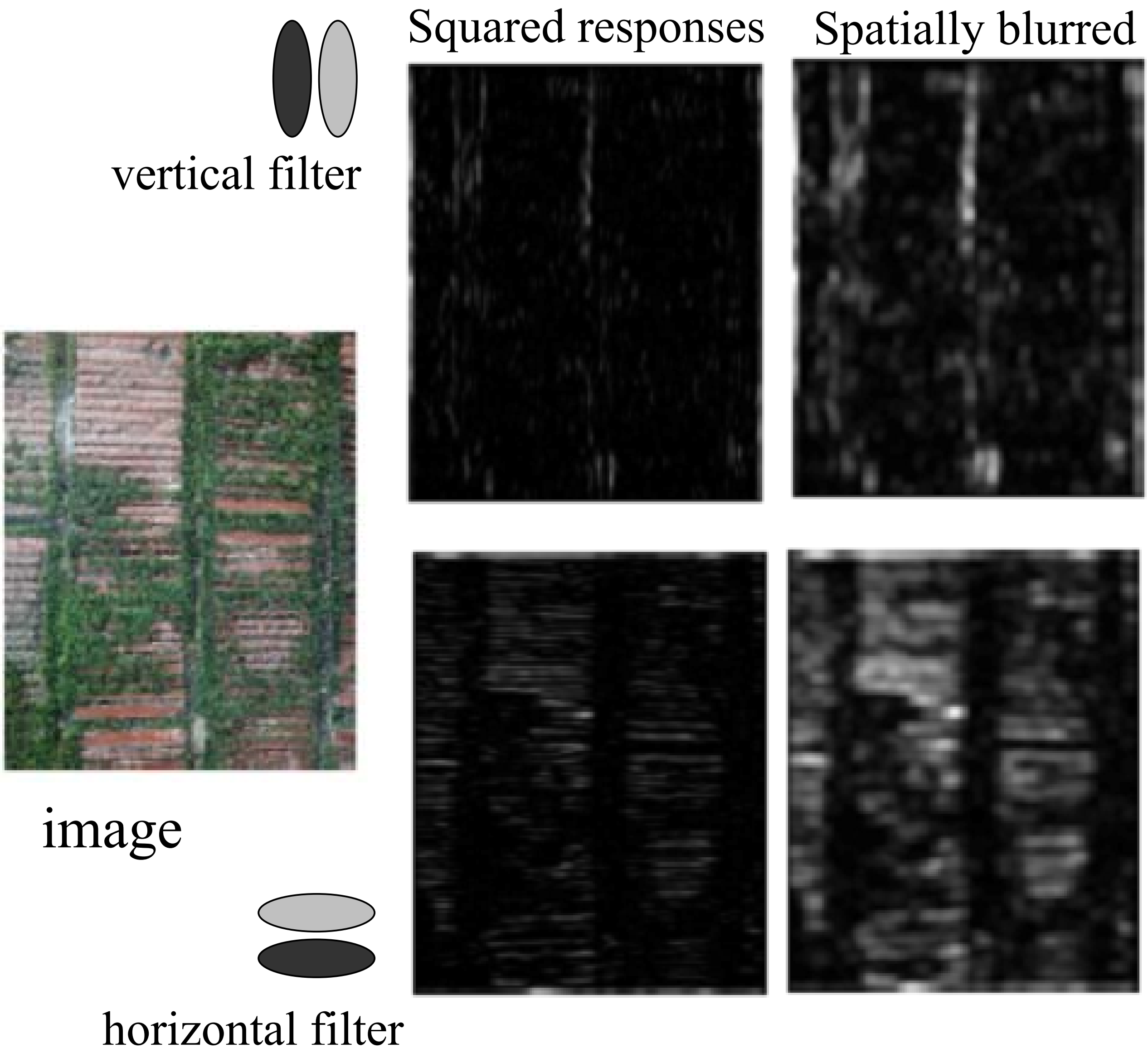


Malik J, Perona P. Preattentive texture discrimination with early vision mechanisms. J OPT SOC AM A 7: (5) 923-932 MAY 1990

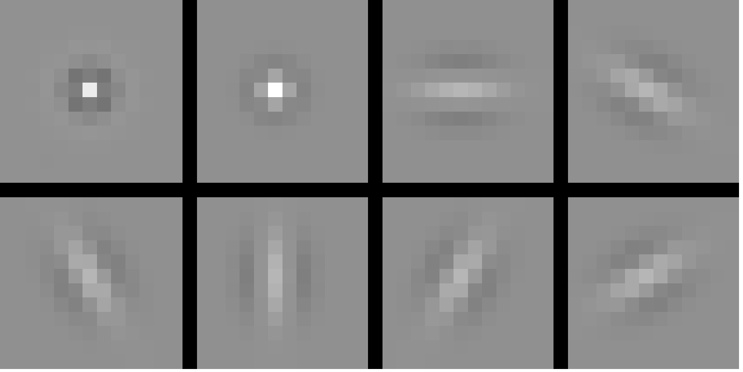
# Representing textures

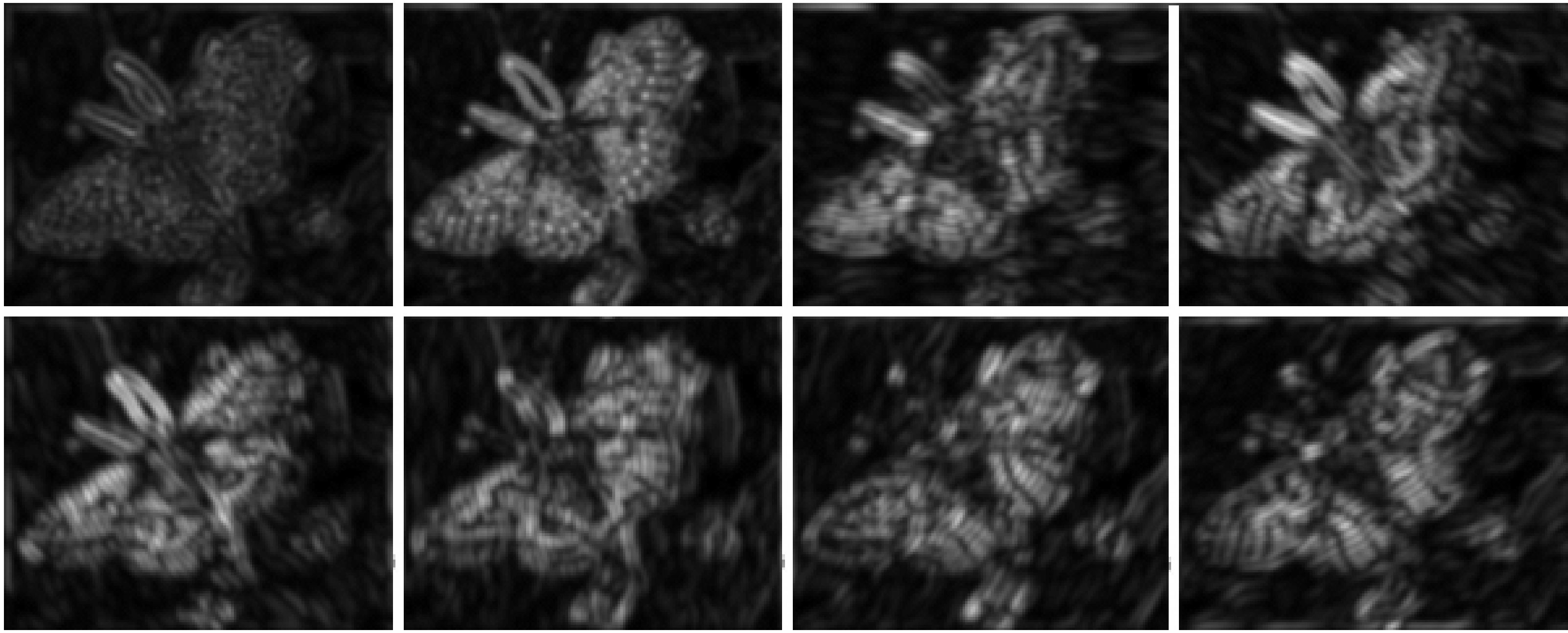
- Textures are made up of quite stylised subelements, repeated in meaningful ways
- Representation:
  - find the subelements, and represent their statistics
- But what are the subelements, and how do we find them?
  - recall normalized correlation
  - find subelements by applying filters, looking at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

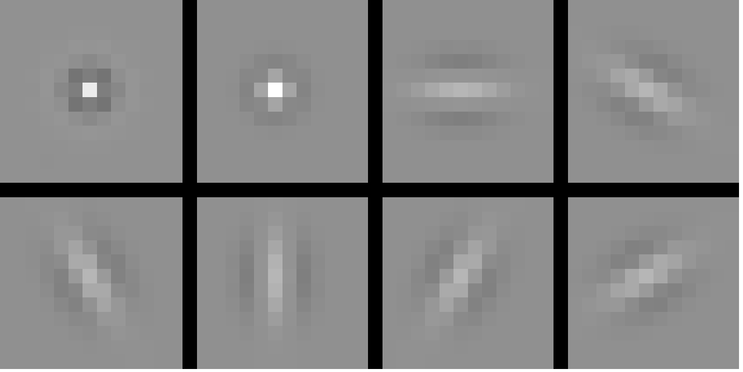


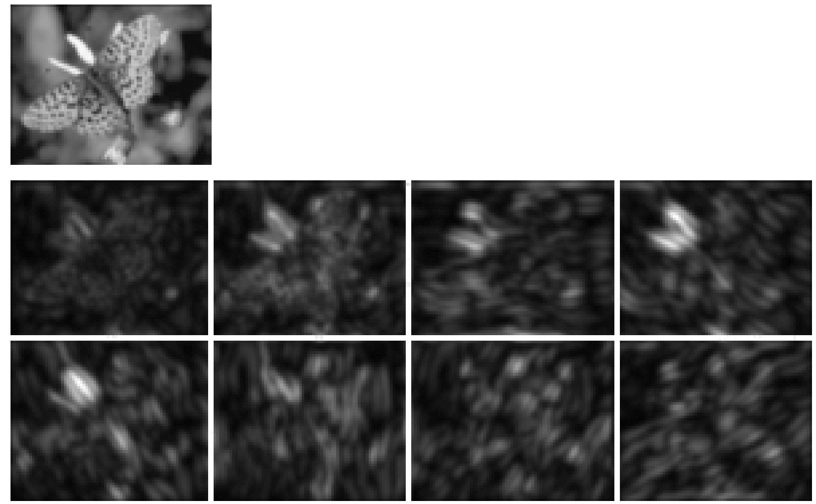


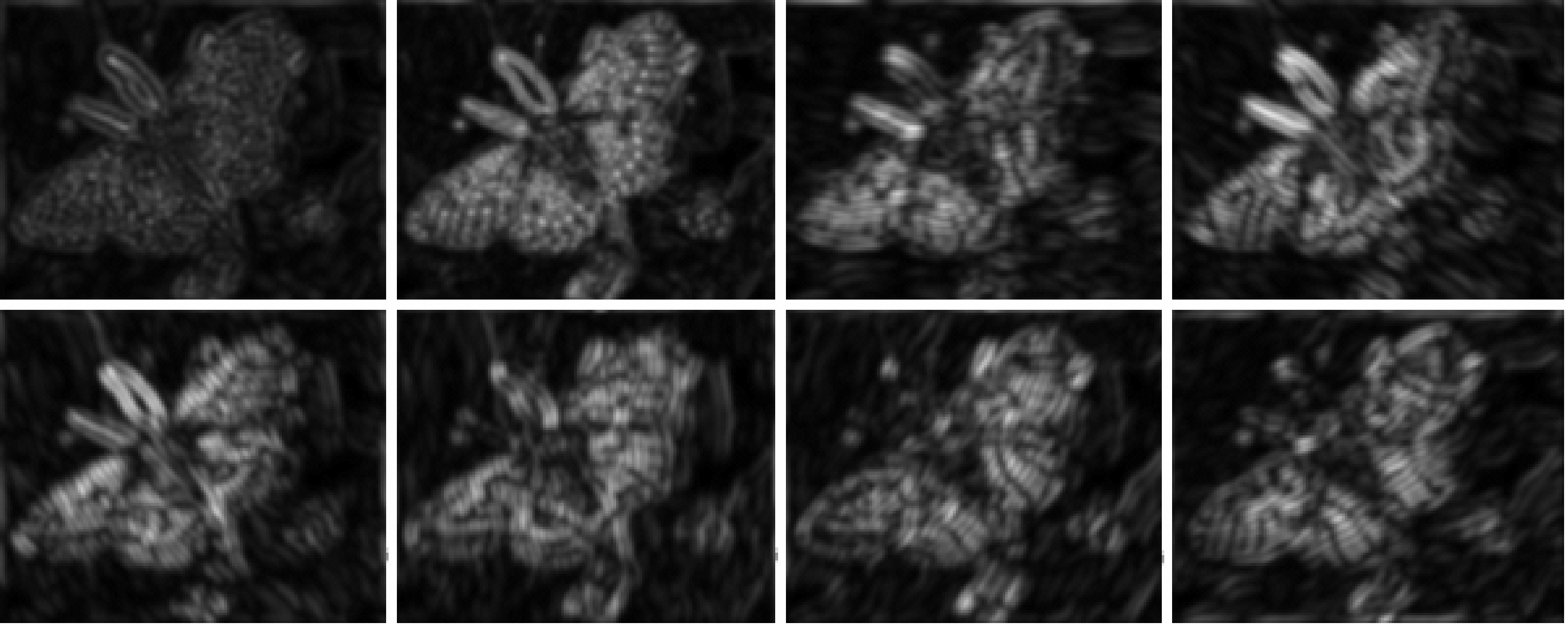

Threshold squared, blurred responses, then categorize texture based on those two bits

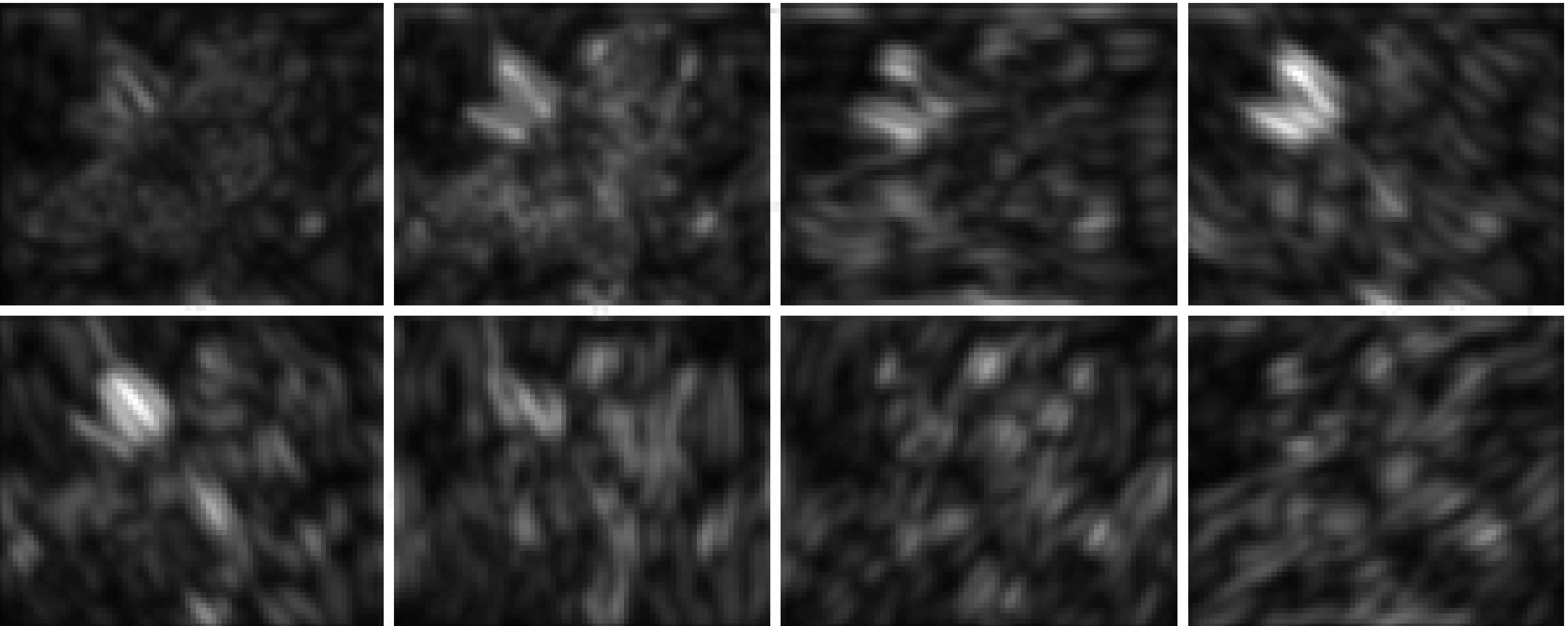










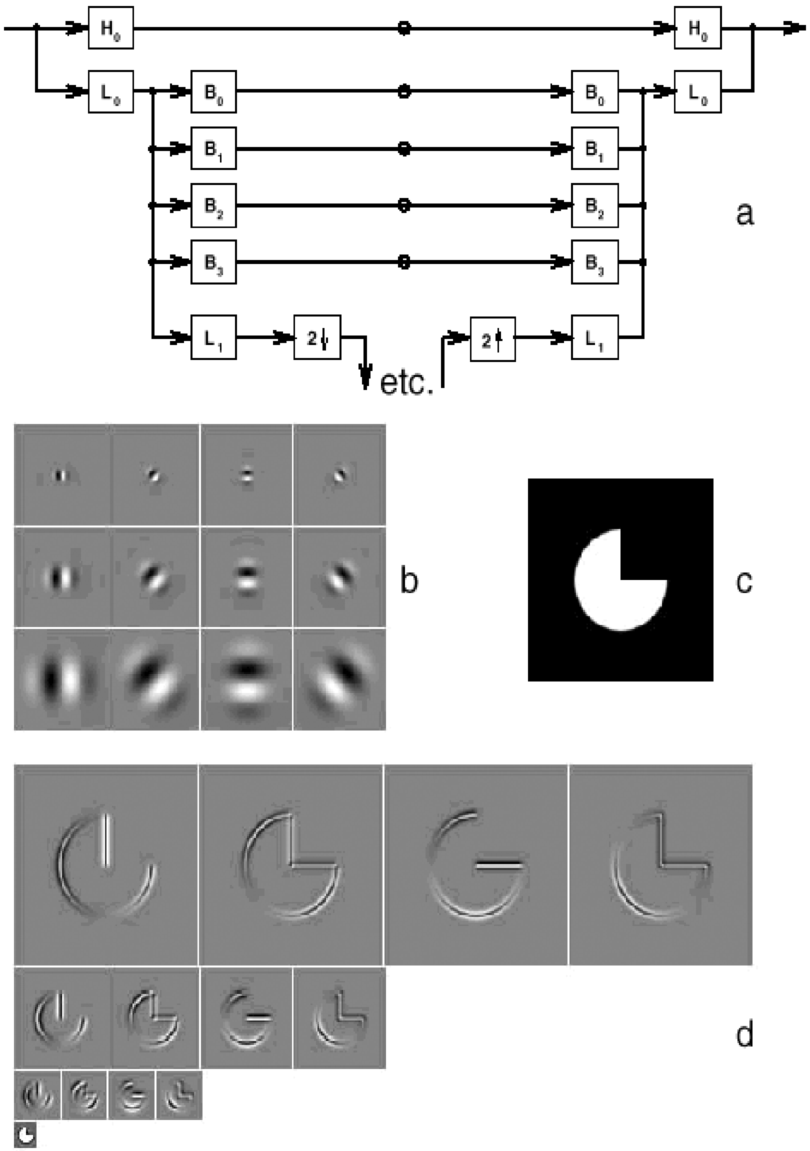


# Pyramid-Based Texture Analysis/Synthesis

David J. Heeger[‡]
Stanford University

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

SIGGRAPH 1994



Learn: use filter marginal statistics.

# Bergen and Heeger

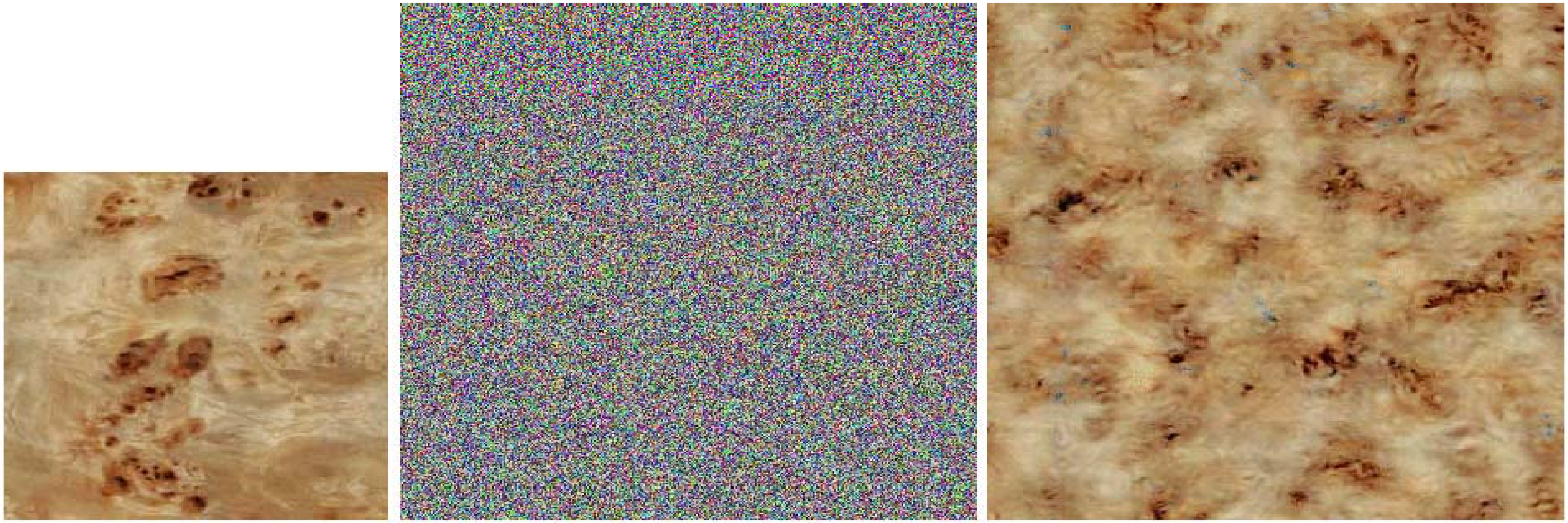


Figure 2: (Left) Input digitized sample texture: burled mappa wood. (Middle) Input noise. (Right) Output synthetic texture that matches the appearance of the digitized sample. Note that the synthesized texture is larger than the digitized sample; our approach allows generation of as much texture as desired.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

# 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.

# Bergen and Heeger failures

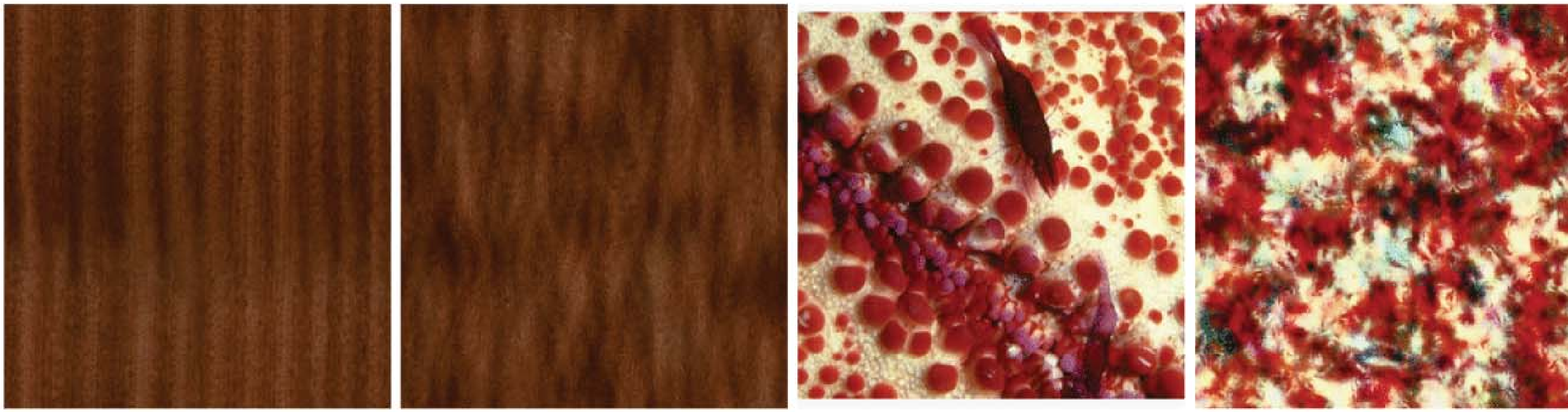


Figure 8: Examples of failures: wood grain and red coral.

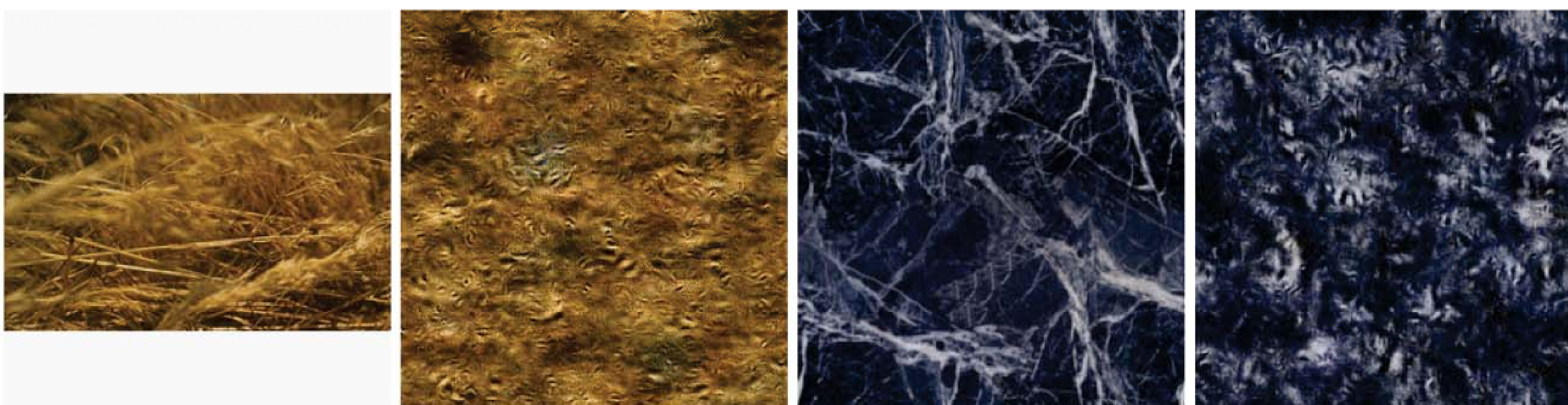


Figure 9: More failures: hay and marble.

# De Bonet (and Viola)

SIGGRAPH 1997

## Multiresolution Sampling Procedure for Analysis and Synthesis of Texture Images

Jeremy S. De Bonet
Learning & Vision Group
Artificial Intelligence Laboratory
Massachusetts Institute of Technology

EMAIL: jsd@ai.mit.edu
HOMEPAGE: http://www.ai.mit.edu/_jsd

Learn: use filter conditional statistics across scale.

# DeBonet

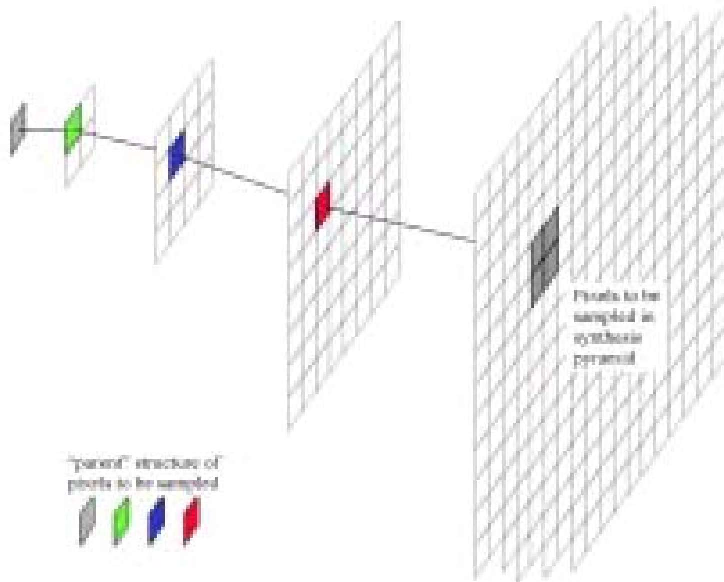



Figure 8: The distribution from which pixels in the synthesis pyramid are sampled is conditioned on the "parent" structure of those pixels. Each element of the parent structure contains a vector of the feature measurements at that location and scale.

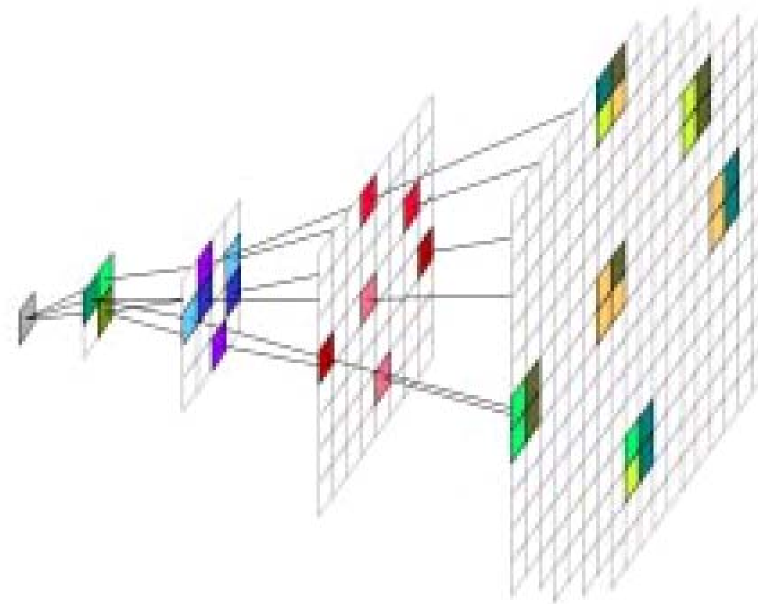
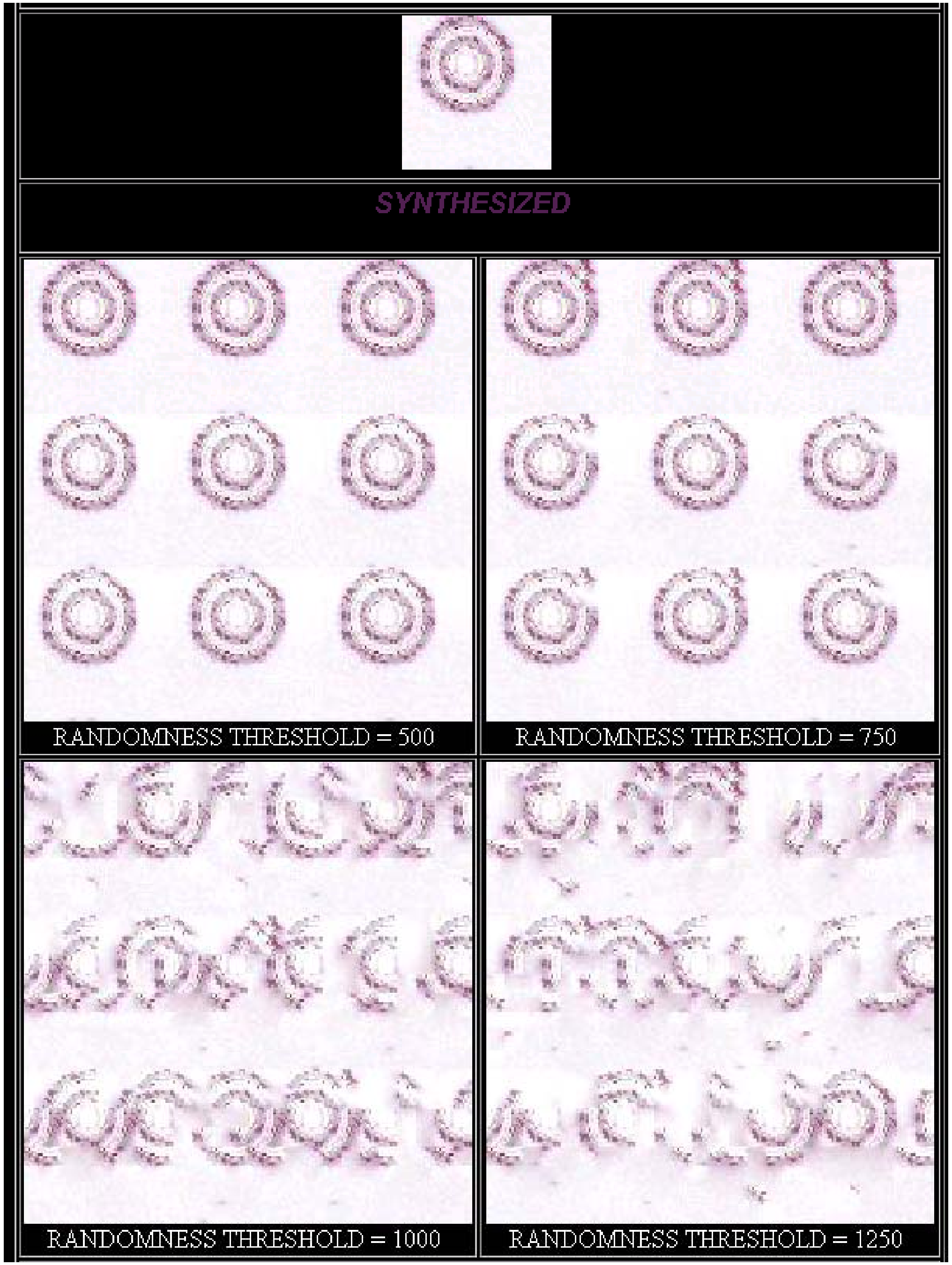


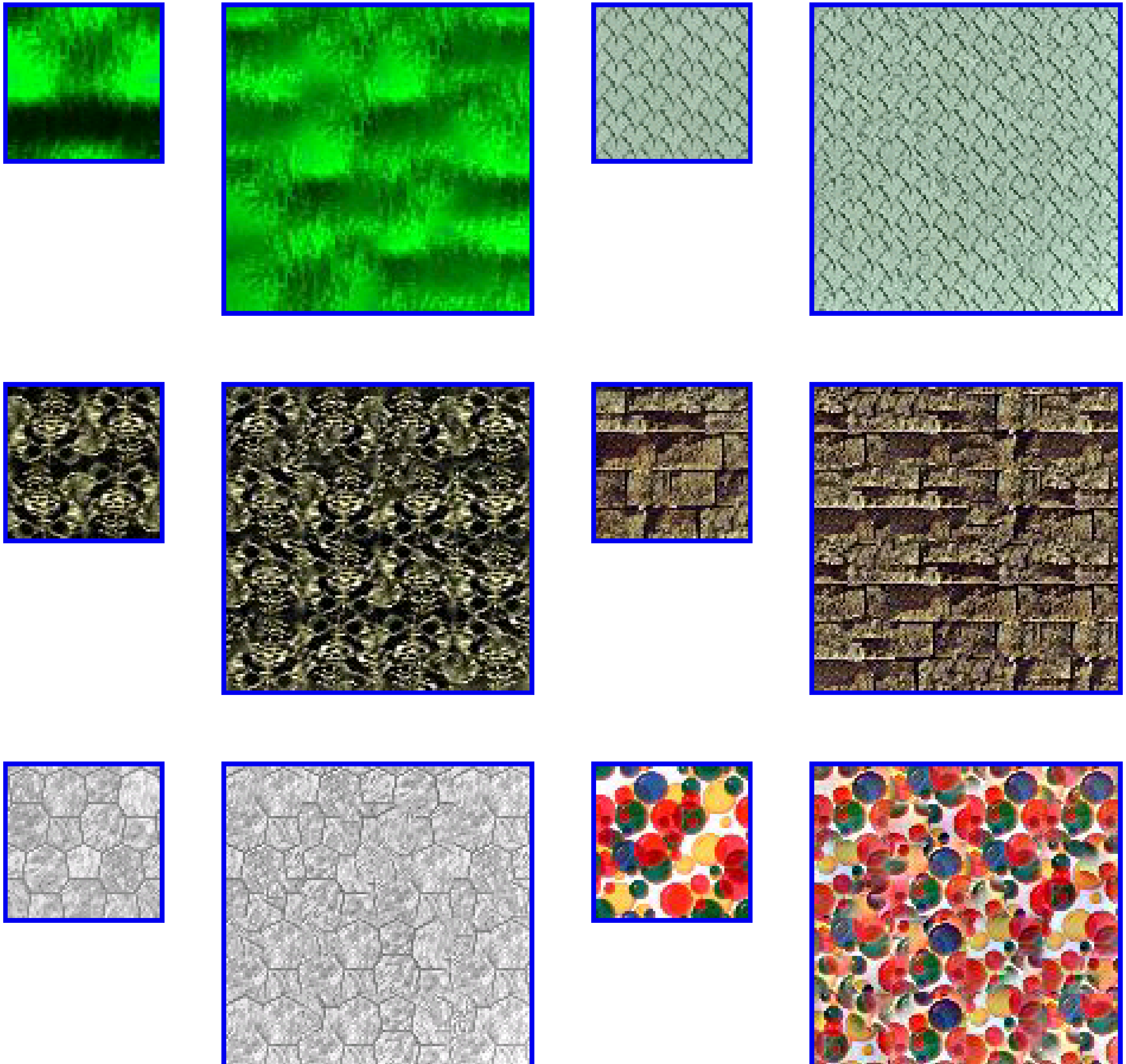
Figure 9: An input texture is decomposed to form an analysis pyramid, from which a new synthesis pyramid is sampled, conditioned on local features within the pyramids. A filter bank of local texture measures, based on psychophysical models, are used as features.

# DeBonet



SYNTHESIZED
RANDOMNESS THRESHOLD = 500
RANDOMNESS THRESHOLD = 750
RANDOMNESS THRESHOLD = 1000
RANDOMNESS THRESHOLD = 1250

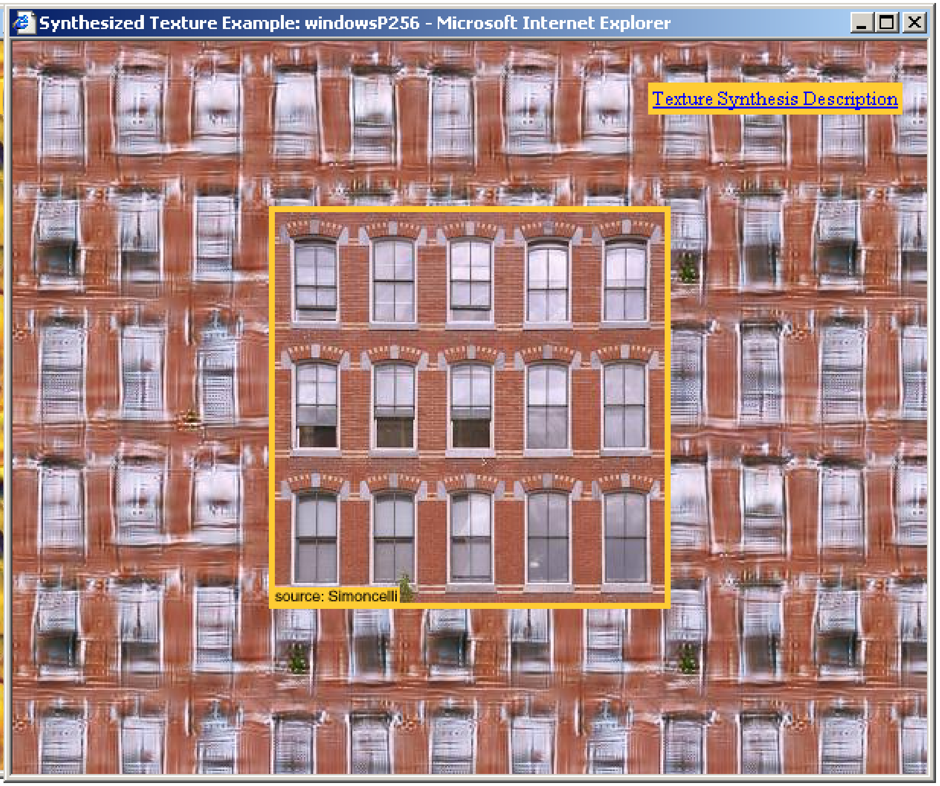
# DeBonet



# Portilla and Simoncelli

- Parametric representation.
- About 1000 numbers to describe a texture.
- Ok results;  maybe as good as DeBonet.

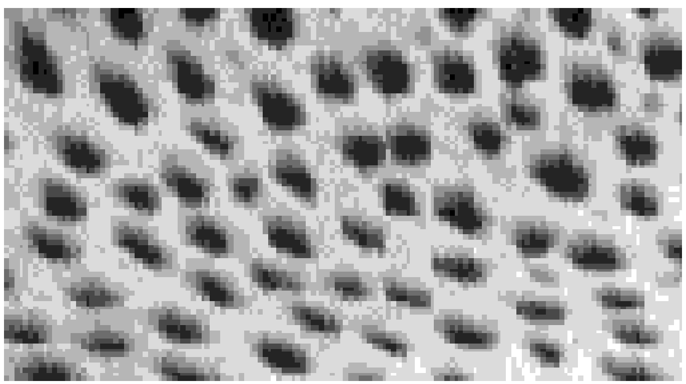
# Portilla and Simoncelli



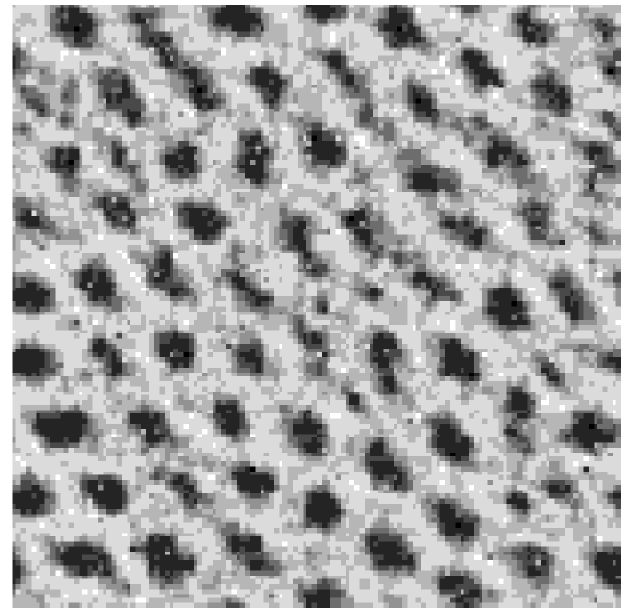
# Zhu, Wu, & Mumford, 1998

- Principled approach.
- Synthesis quality not great, but ok.

# Zhu, Wu, & Mumford



a



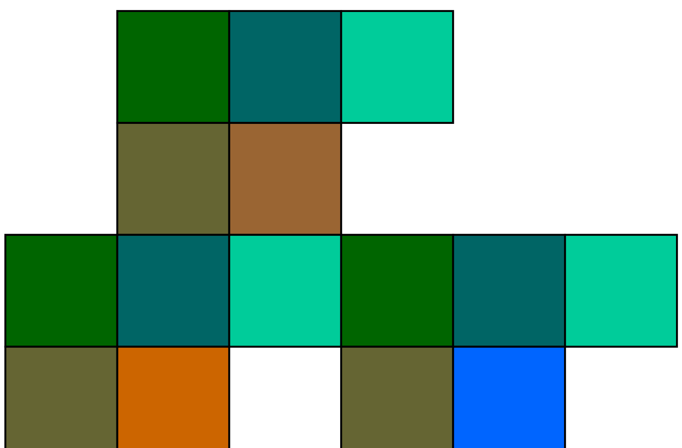
b

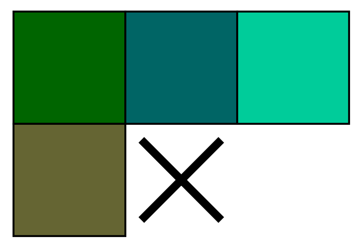
- Cheetah Synthetic

IEEE International Conference on Computer Vision, Corfu, Greece, September 1999

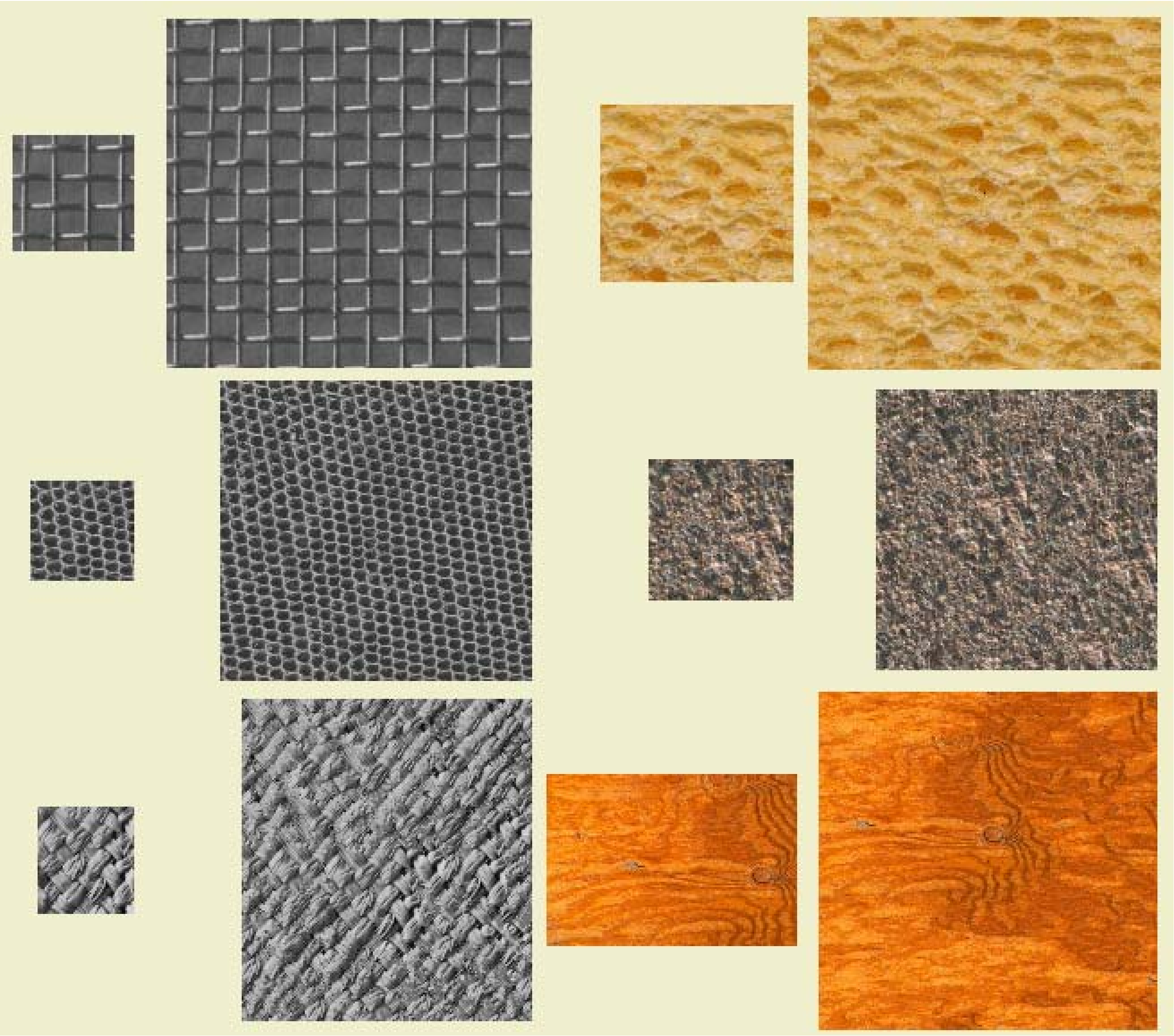
# Texture Synthesis by Non-parametric Sampling

Alexei A. Efros and Thomas K. Leung
Computer Science Division
University of California, Berkeley
Berkeley, CA 94720-1776, U.S.A.
{efros,leungt}@cs.berkeley.edu





# Efros and Leung



(a)

(b)

(c)

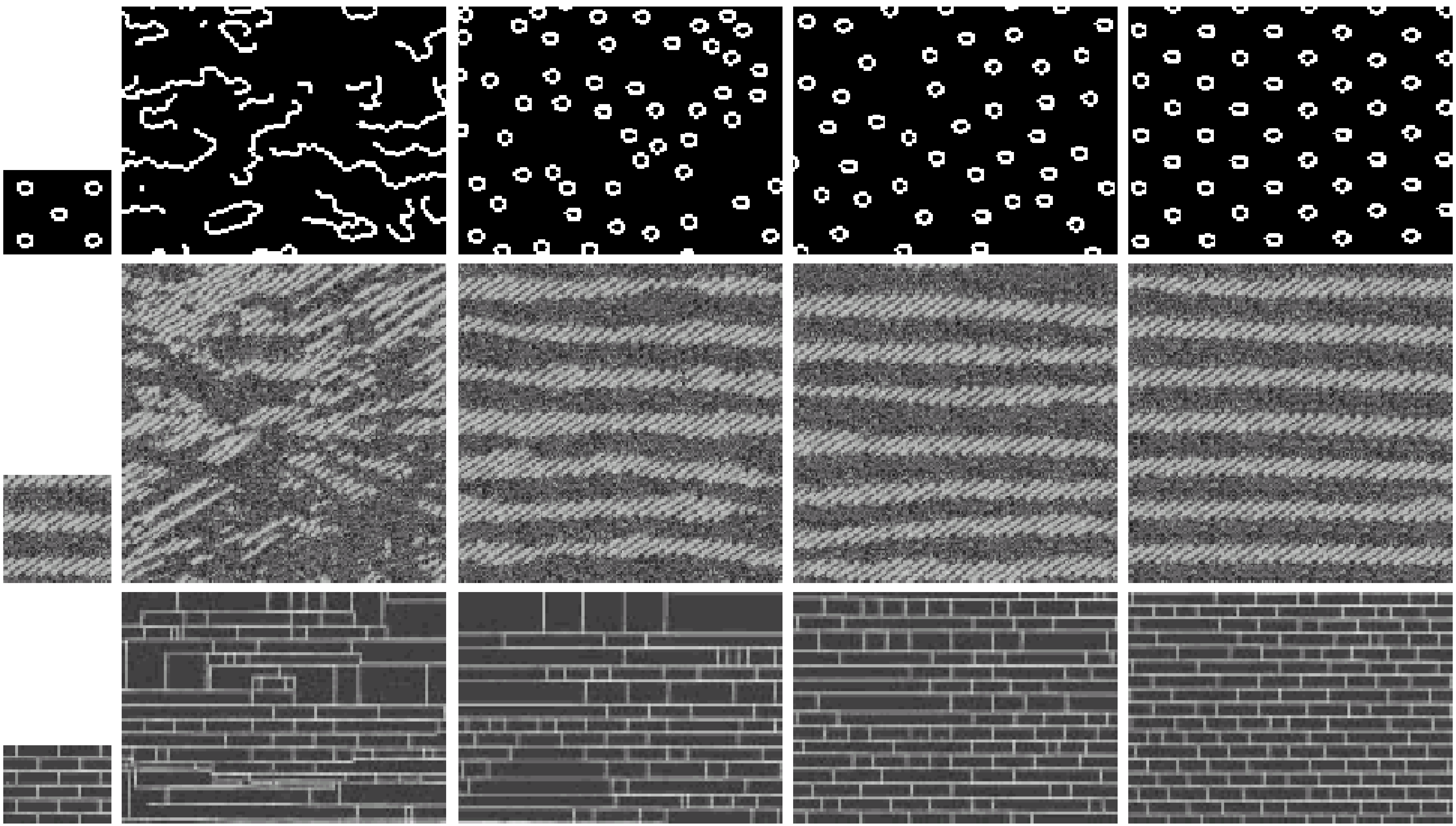


Figure 2. Results: given a sample image (left), the algorithm synthesized four new images with neighborhood windows of width 5, 11, 15, and 23 pixels respectively. Notice how perceptually intuitively the window size corresponds to the degree of randomness in the resulting textures. Input images are: (a) synthetic rings, (b) Brodatz texture D11, (c) brick wall.

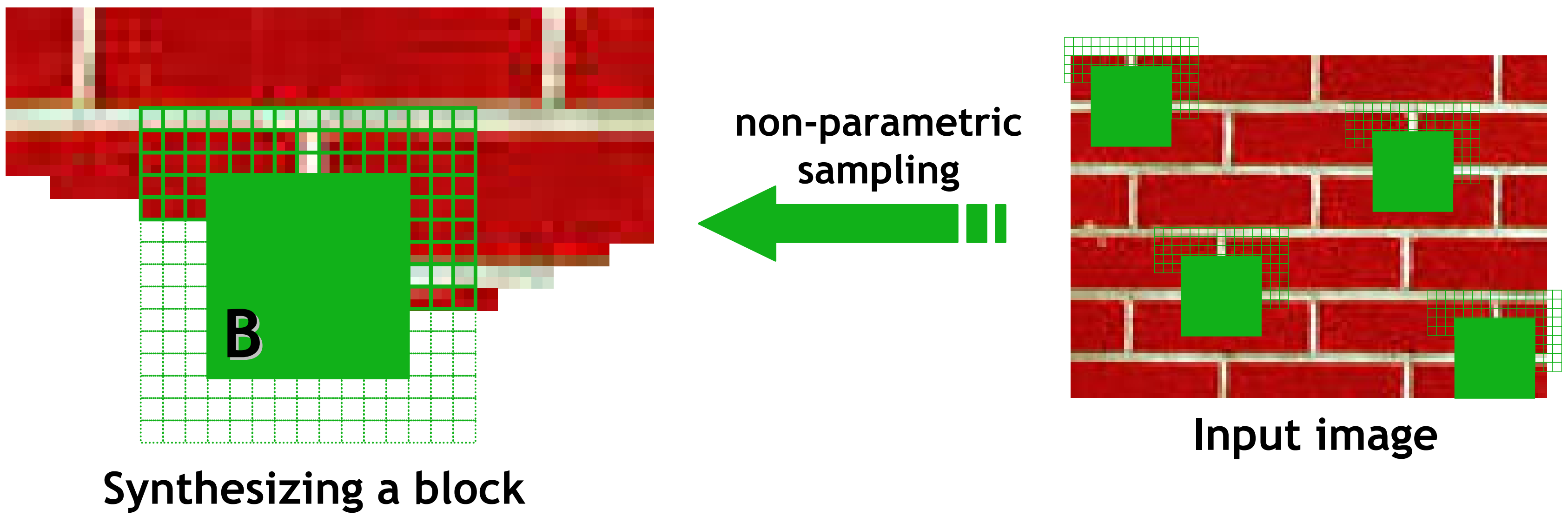
# What we learned from Efros and Leung regarding texture synthesis

- Don’t need conditional filter responses across scale
- Don’t need marginal statistics of filter responses.
- Don’t need multi-scale, multi-orientation filters.
- Don’t need filters.

# Efros & Leung '99

- The algorithm
  - Very simple
  - Surprisingly good results
  - Synthesis is easier than analysis!
  - …but very slow
- Optimizations and Improvements
  - [Wei & Levoy,'00] (based on [Popat & Picard,'93])
  - [Harrison,'01]
  - [Ashikhmin,'01]

# Efros & Leung ’99 extended



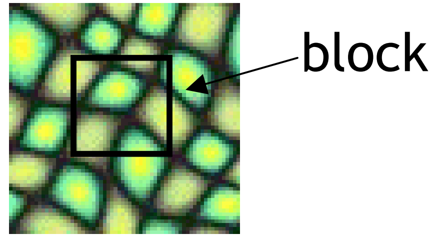

- <u>Observation:</u> neighbor pixels are highly correlated

**<u>Idea:</u> unit of synthesis = block**

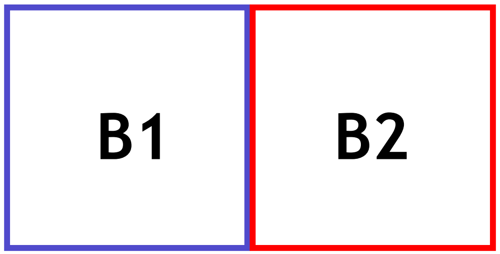
- Exactly the same but now we want P(**B**|N(**B**))
- Much faster: synthesize all pixels in a block at once
- Not the same as multi-scale!

# Image Quilting

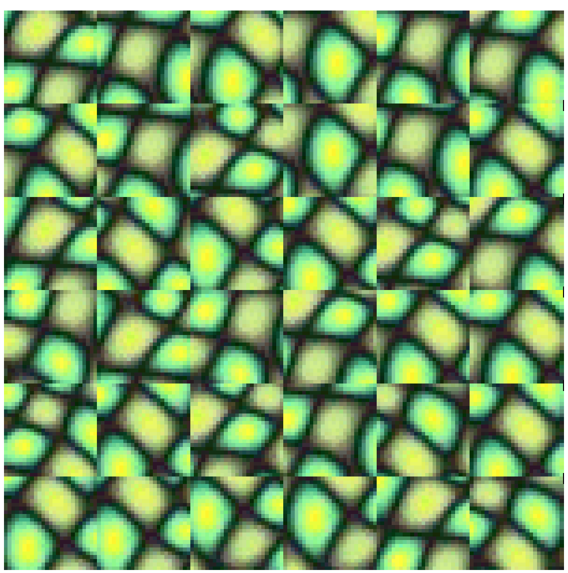
- Idea:
  - let's combine random block placement of Chaos Mosaic with spatial constraints of Efros & Leung
- Related Work (concurrent):
  - Real-time patch-based sampling [Liang et.al. '01]
  - Image Analogies [Hertzmann et.al. '01]

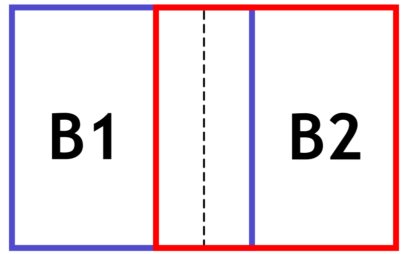



Input texture

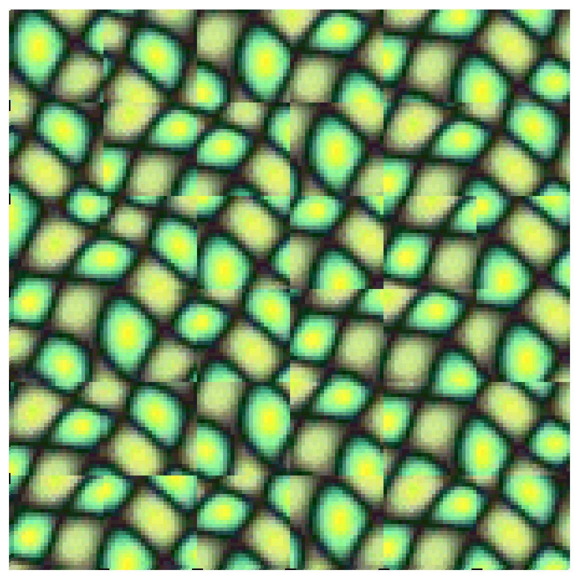


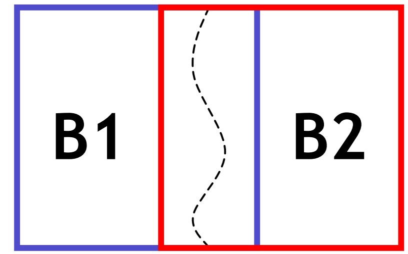

Random placement of blocks



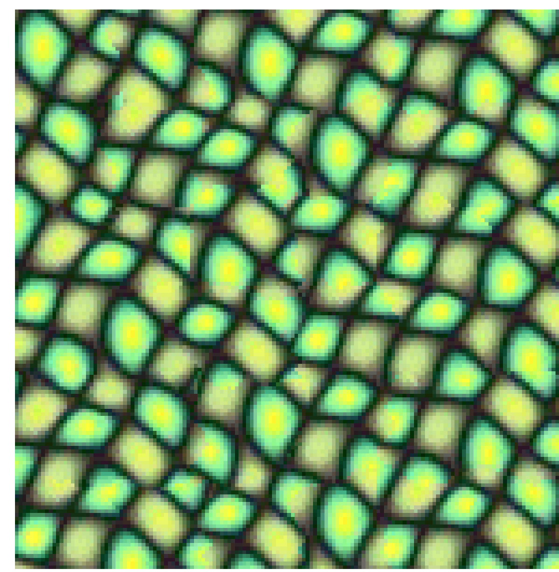



Neighboring blocks constrained by overlap

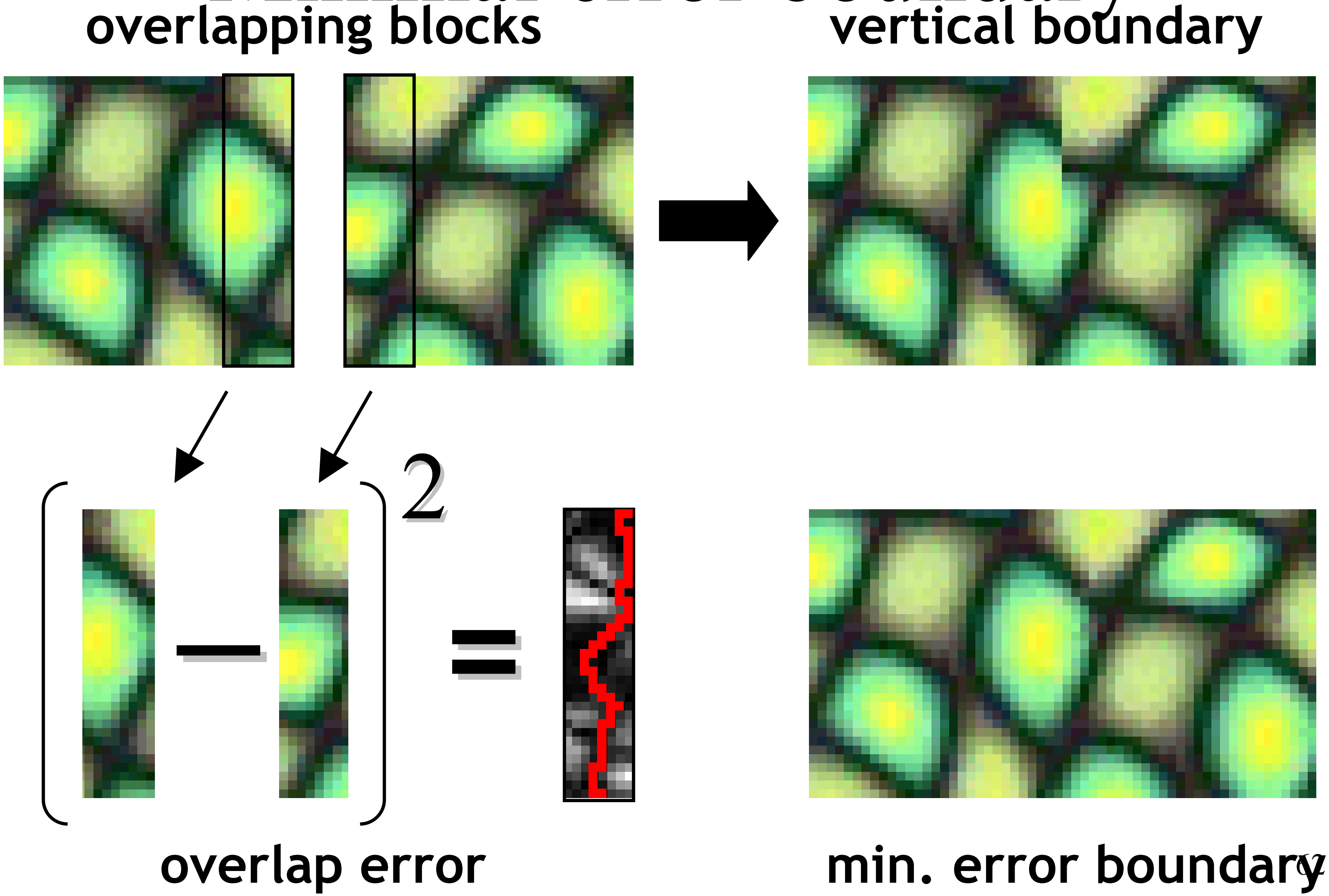





Minimal error boundary cut



# Minimal error boundary

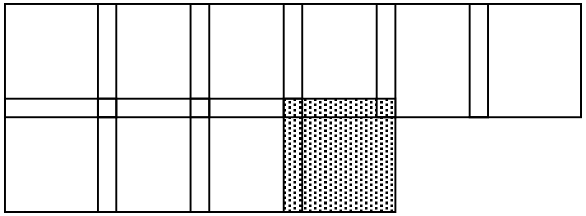


# Our Philosophy

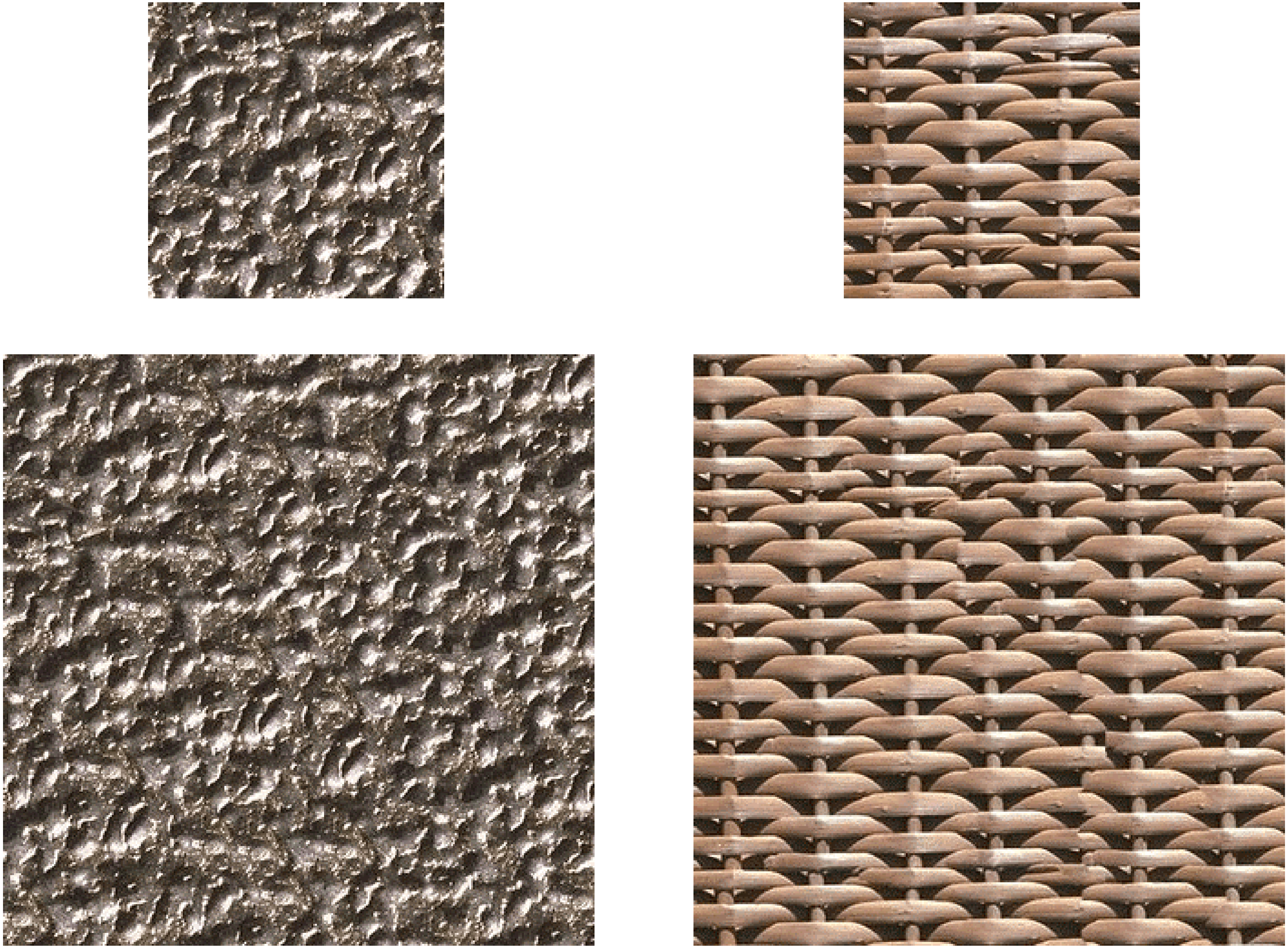
- The “Corrupt Professor’s Algorithm”:
  – Plagiarize as much of the source image as you can
  – Then try to cover up the evidence
- Rationale:
  – Texture blocks are by definition correct samples of texture so problem only connecting them together

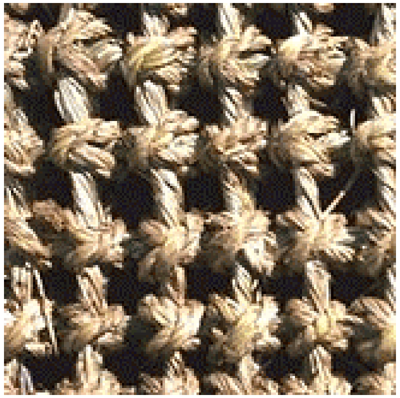
# Algorithm

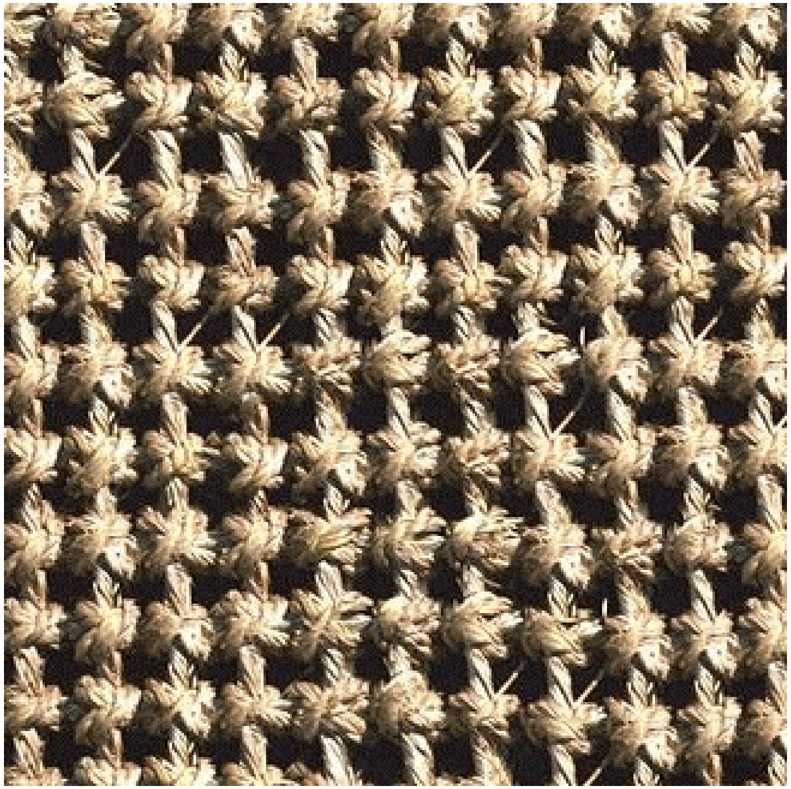
– Pick size of block and size of overlap
– Synthesize blocks in raster order



– Search input texture for block that satisfies overlap constraints (above and left)
  • Easy to optimize using NN search [Liang et.al., ’01]
– Paste new block into resulting texture
  • use dynamic programming to compute minimal error boundary cut

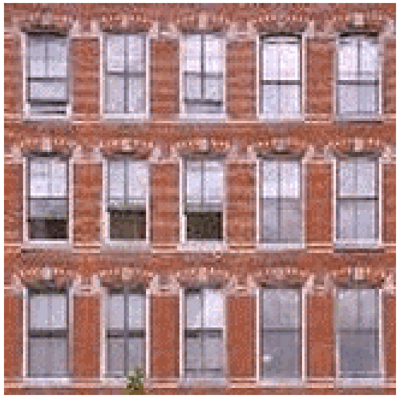




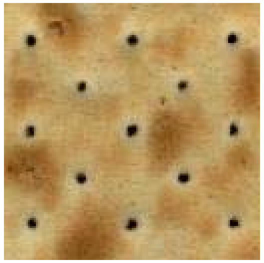
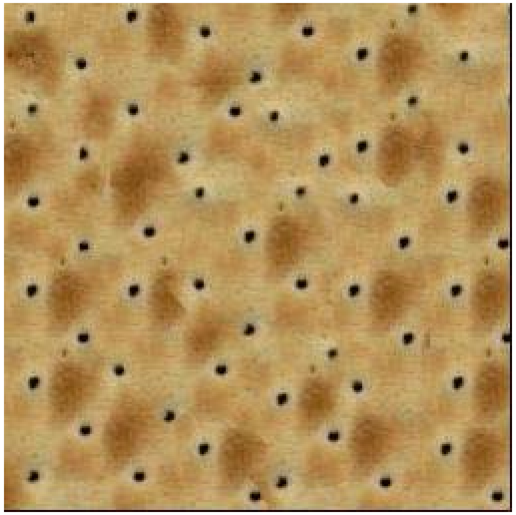




College Inn

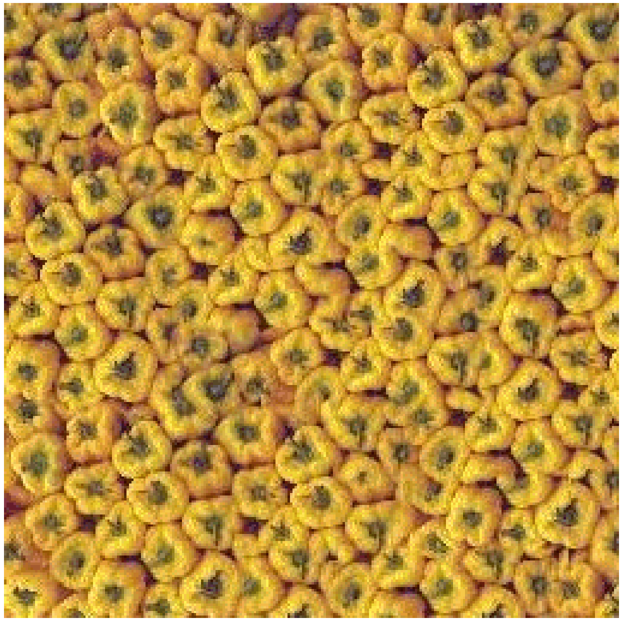


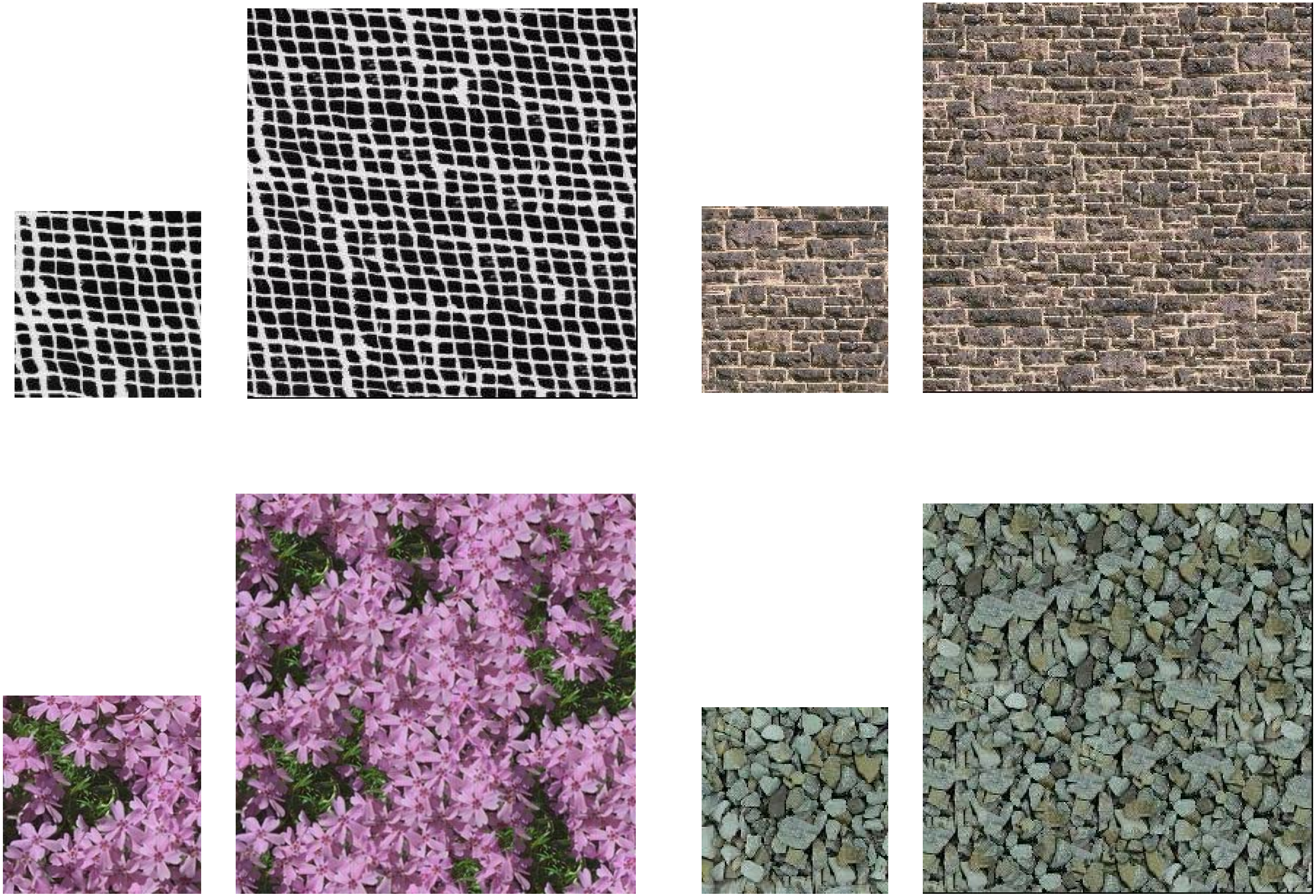

College Inn





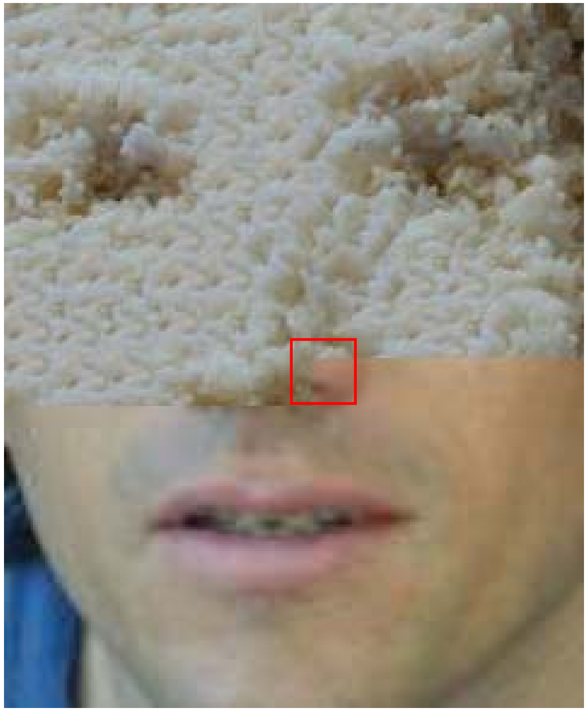




# Failures
## (Chernobyl Harvest)

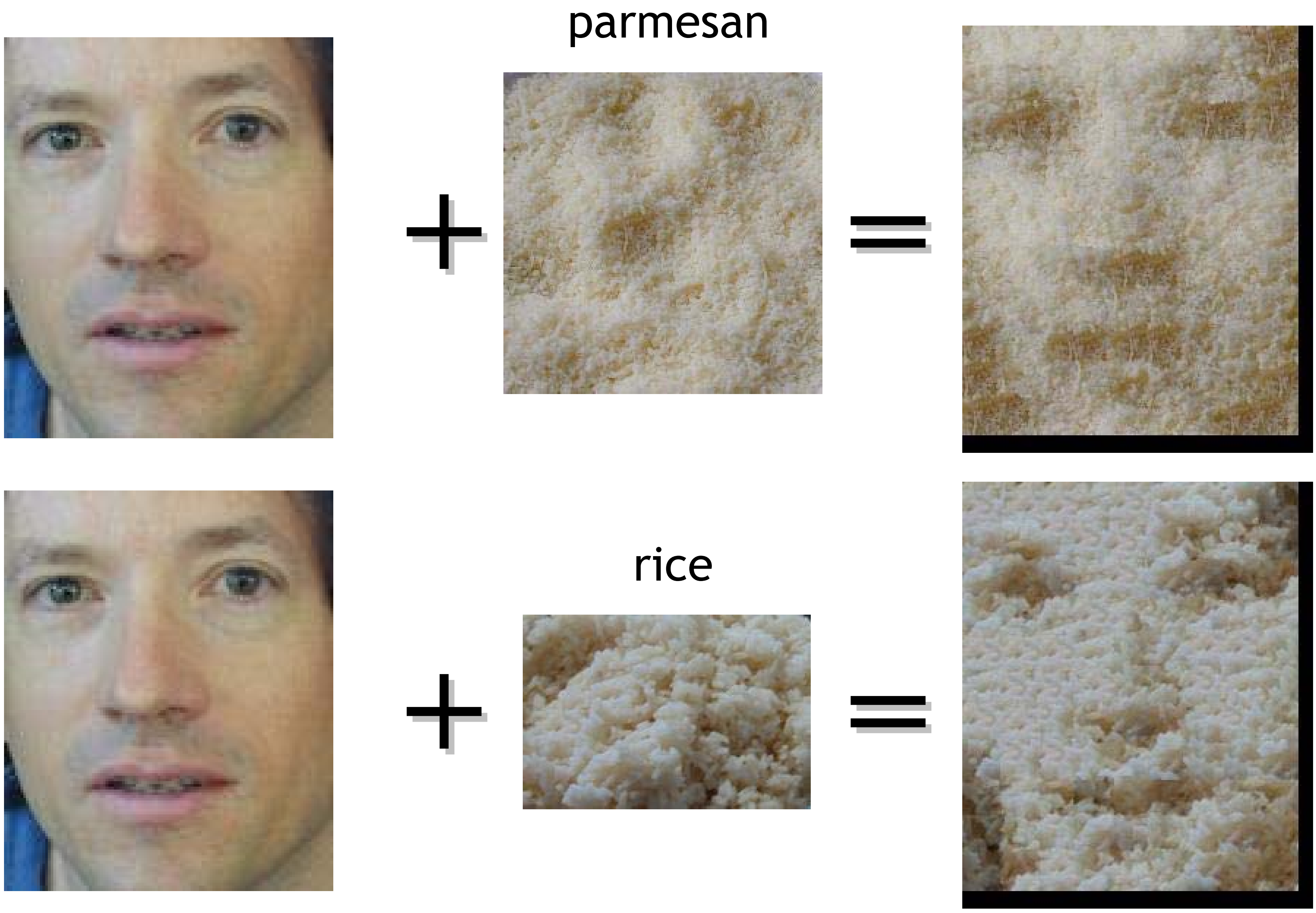
# Texture Transfer

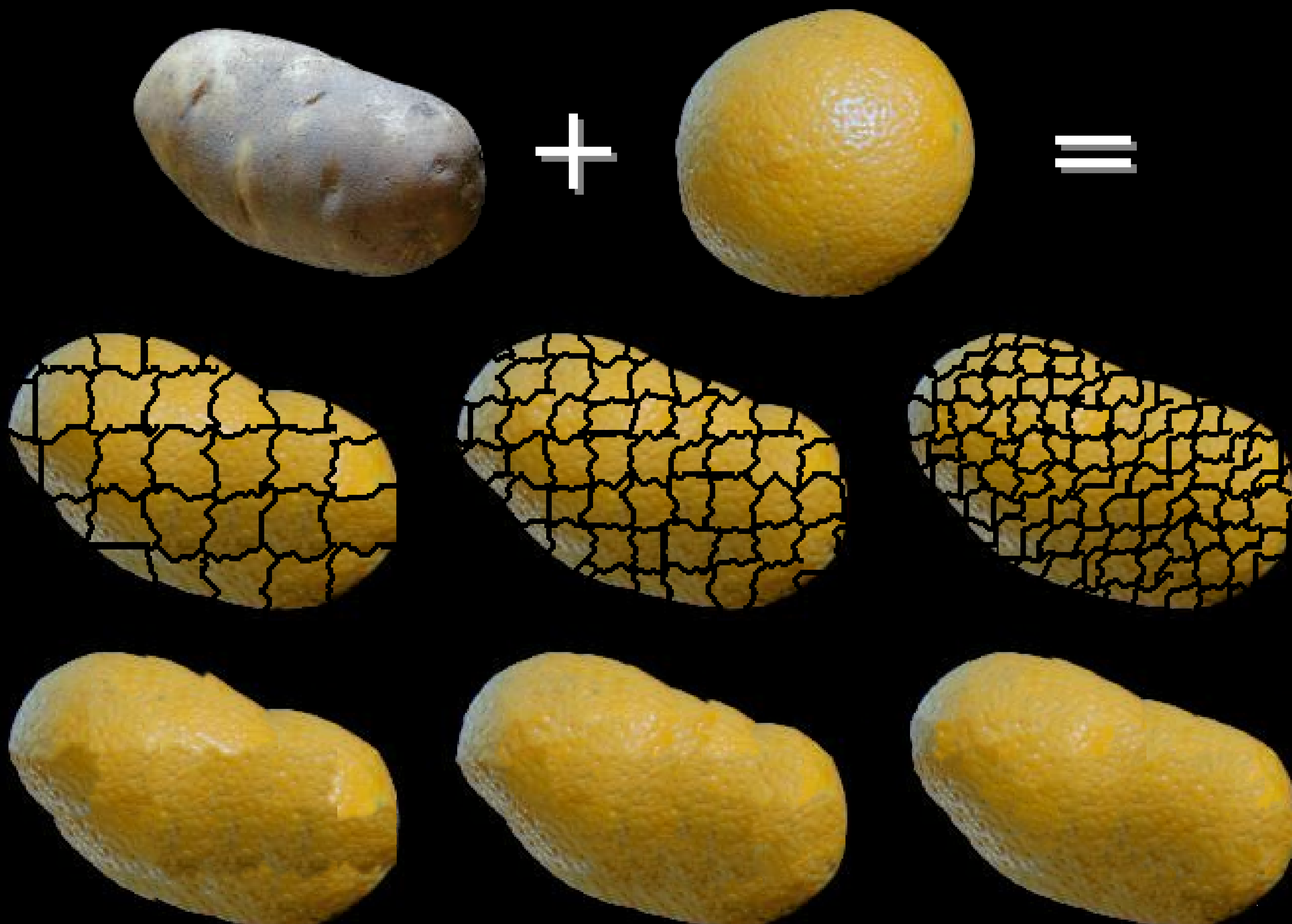
- Take the texture from one object and "paint" it onto another object
  - This requires separating texture and shape
  - That's HARD, but we can cheat
  - Assume we can capture shape by boundary and rough shading

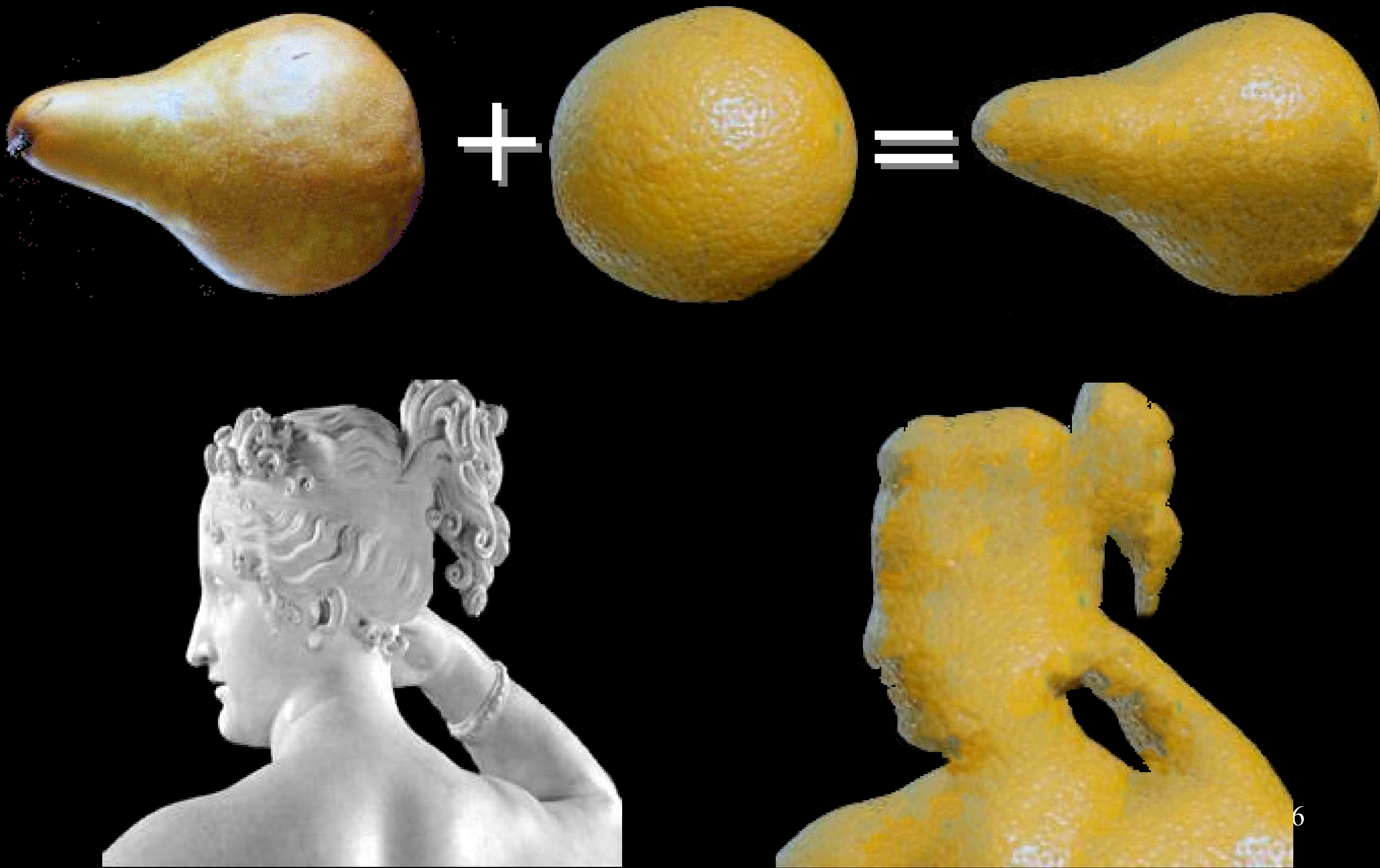


-

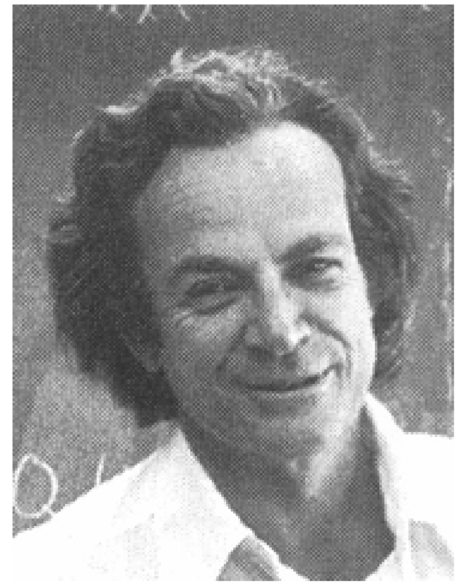
**Then, just add another constraint when sampling: similarity to underlying image at that spot**


parmesan
rice





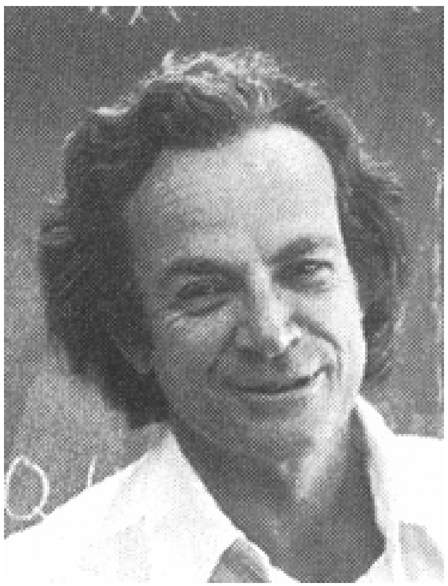
**Source texture**



**Target image**

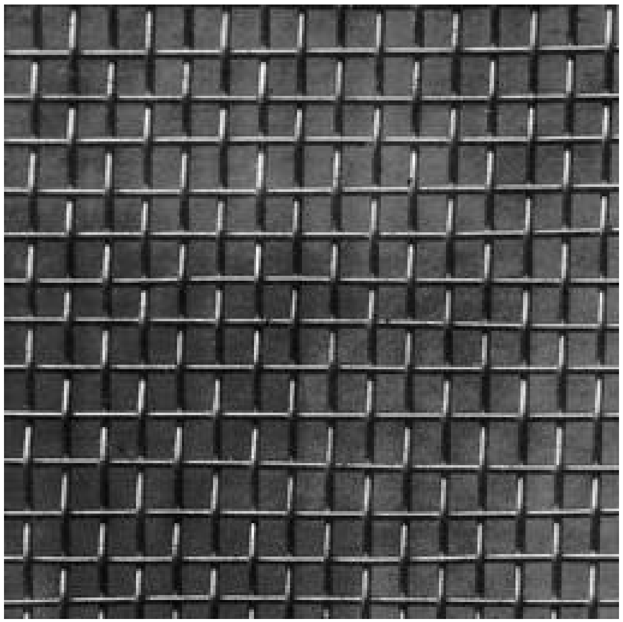
**Source correspondence image**

**Target correspondence image**

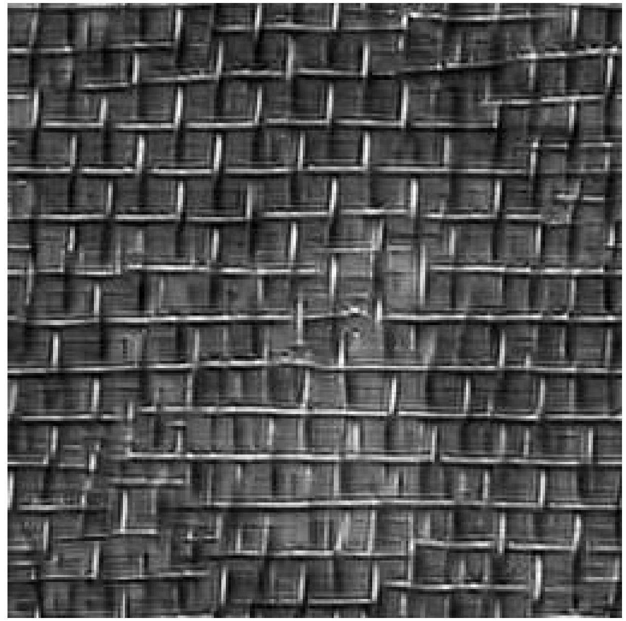


+

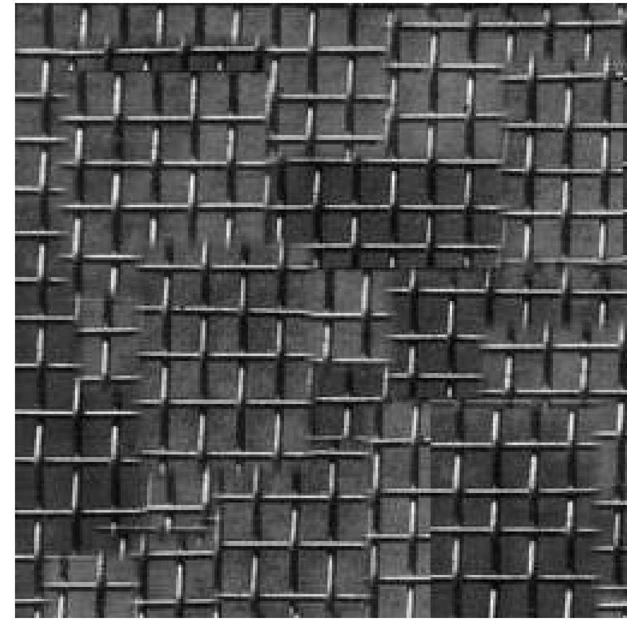
=



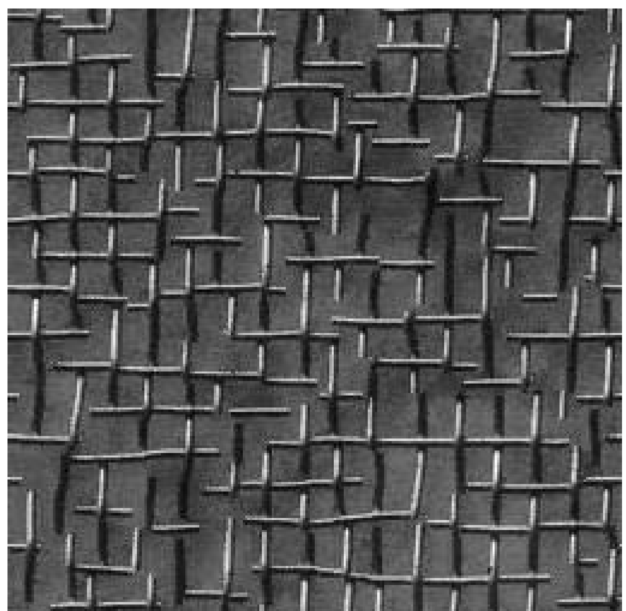
input image



Portilla & Simoncelli



Xu, Guo & Shum



Wei & Levoy

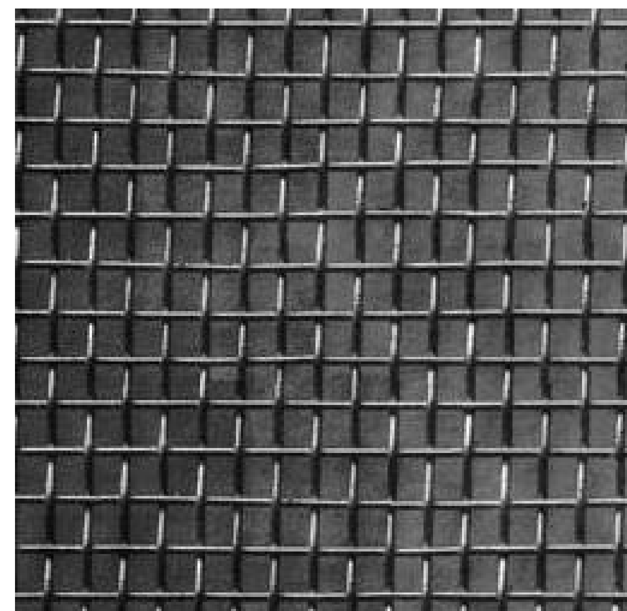
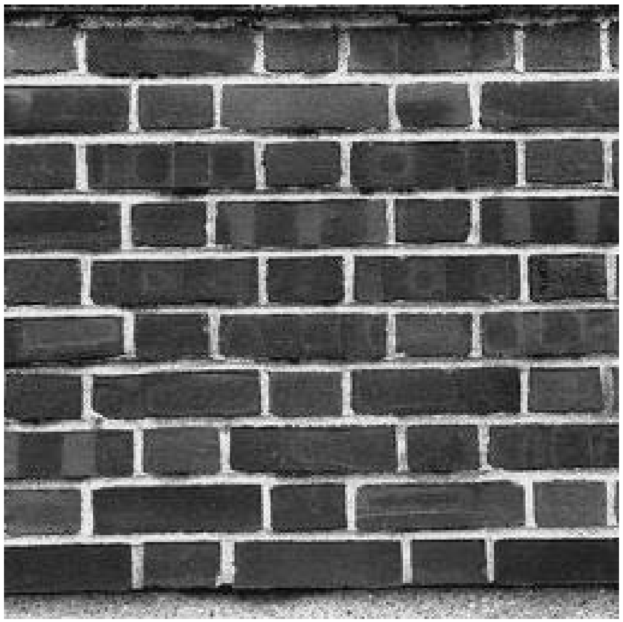
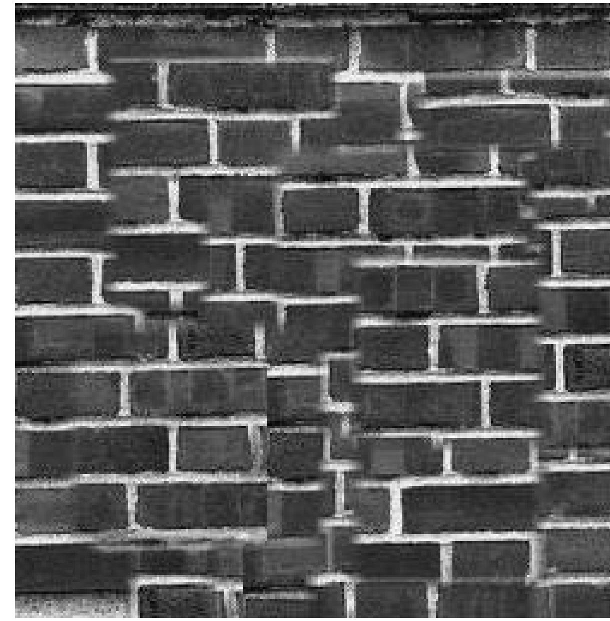


Image Quilting

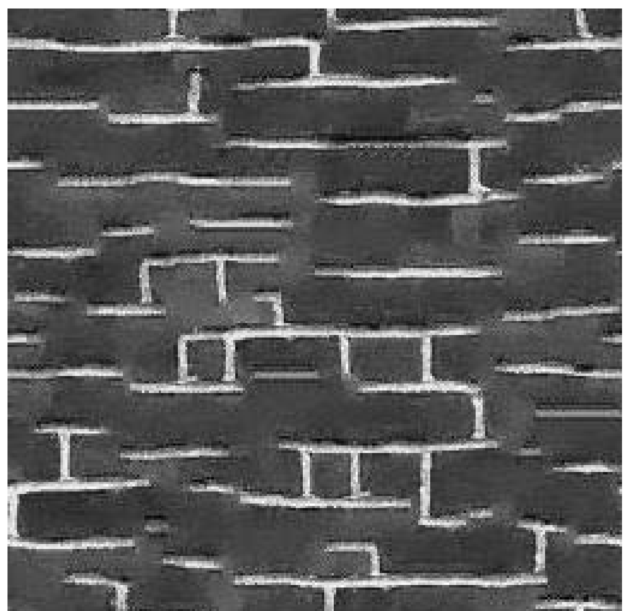


input image

Portilla & Simoncelli



Xu, Guo & Shum



Wei & Levoy

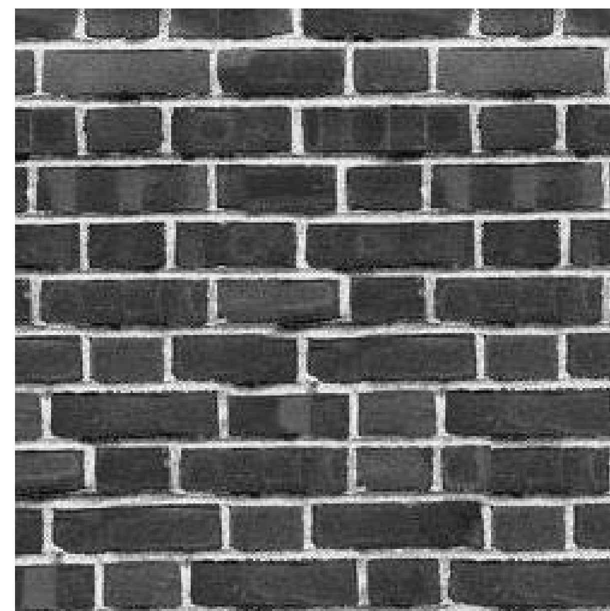
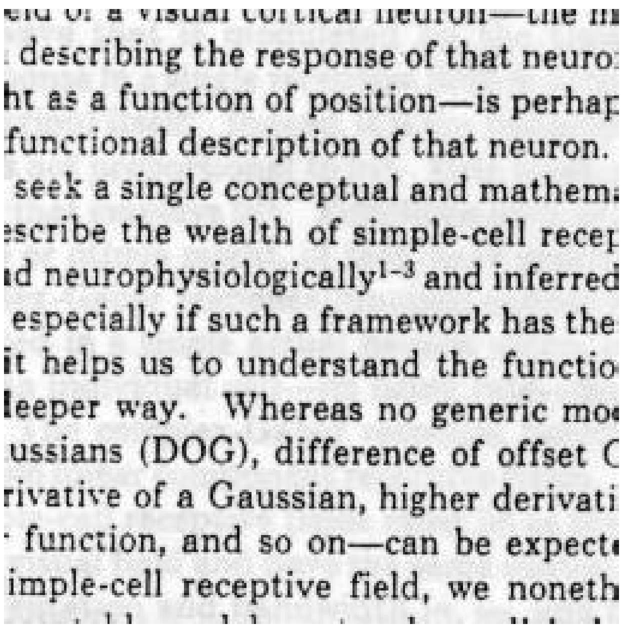
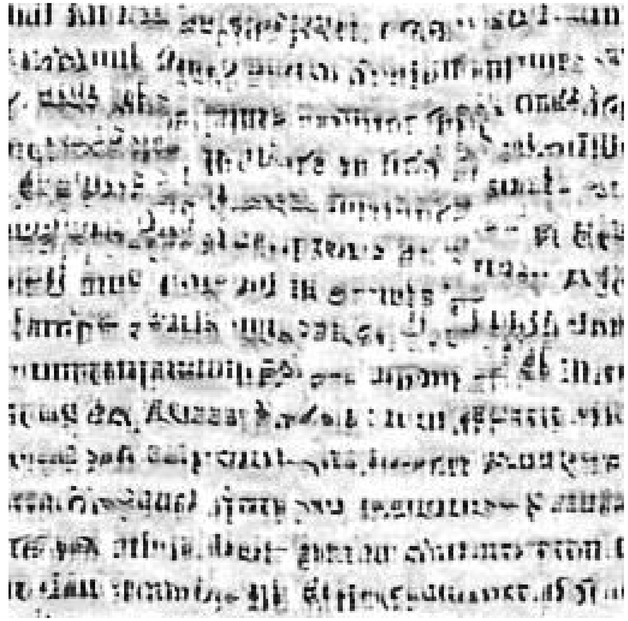


Image Quilting

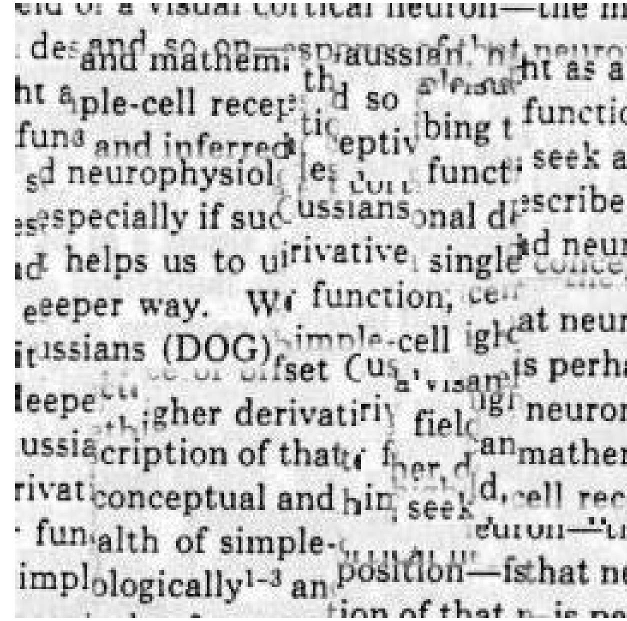
*Homage to Shannon!*



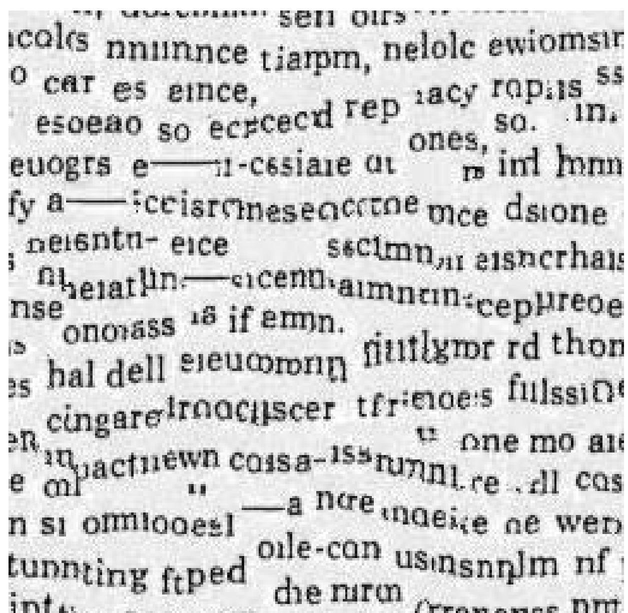
**input image**



**Portilla & Simoncelli**



**Xu, Guo & Shum**



**Wei & Levoy**

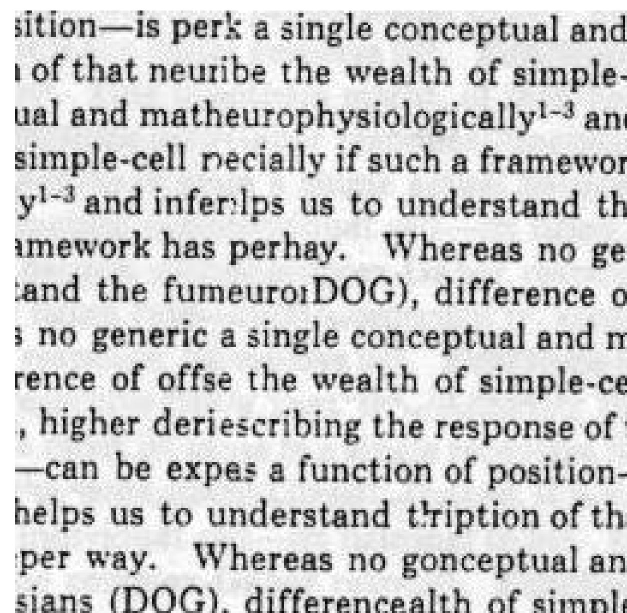


**Image Quilting**

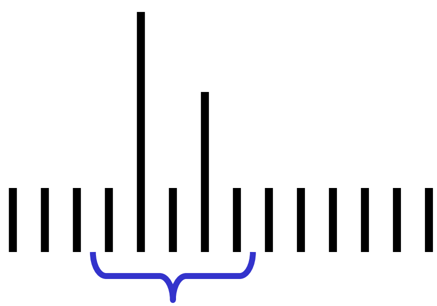
# Summary of image quilting

- Quilt together patches of input image
  - randomly (texture synthesis)
  - constrained (texture transfer)
- Image Quilting
  - No filters, no multi-scale, no one-pixel-at-a-time!
  - fast and very simple
  - Results are not bad

# Median filter

Replace each pixel by the median over N pixels (5 pixels, for these examples). Generalizes to “rank order” filters.

In:

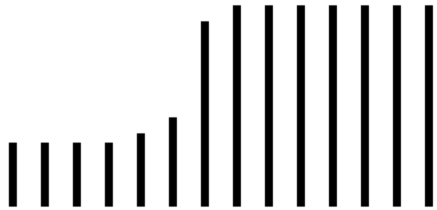


5-pixel neighborhood

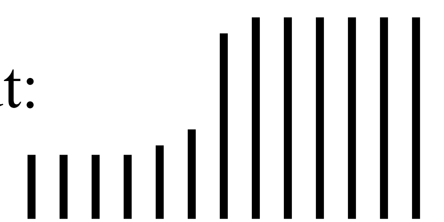
Out:

Spike noise is removed

In:

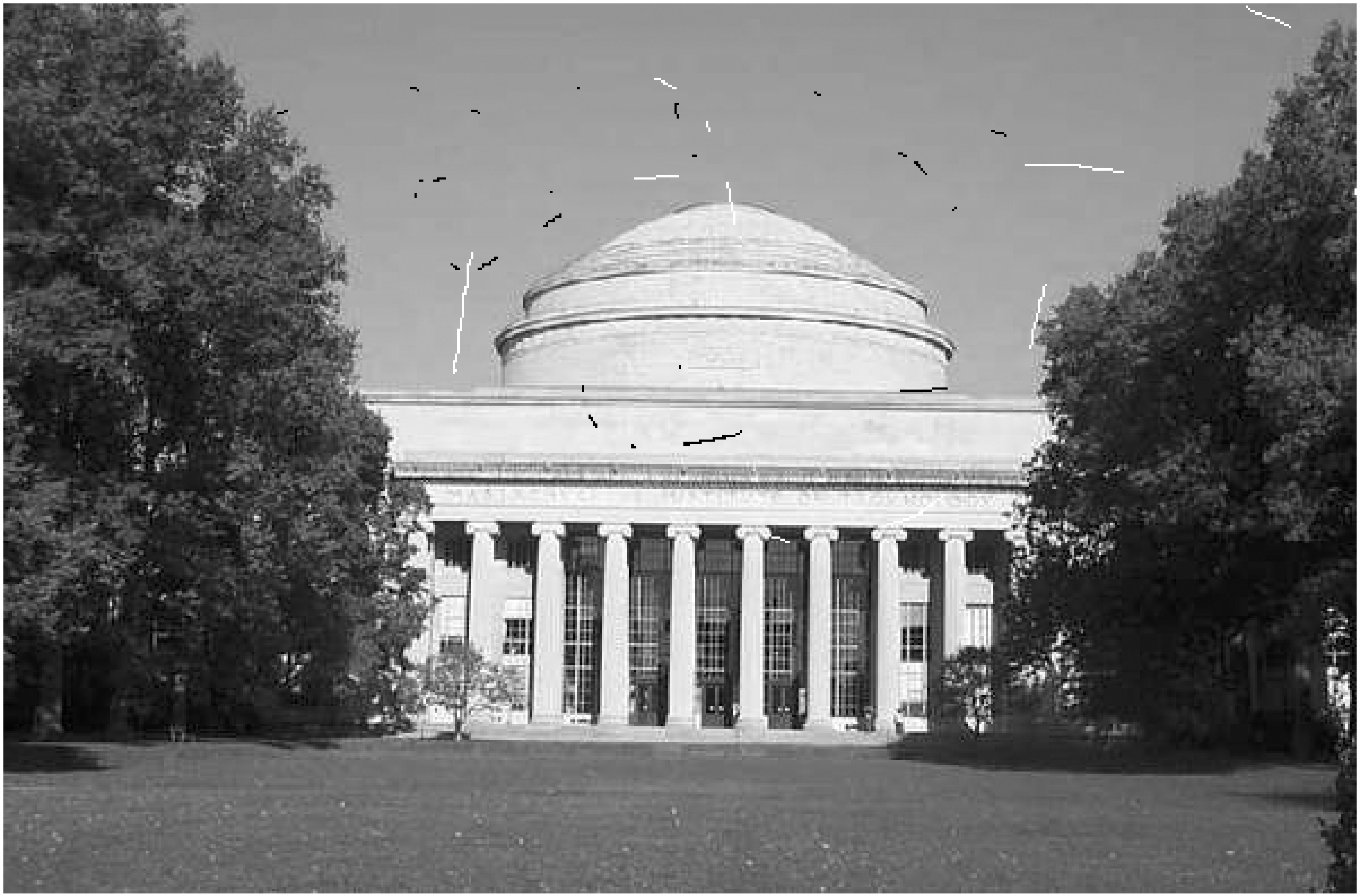


Out:



Monotonic edges remain unchanged

# Degraded image



# Radius 1 median filter

# Radius 2 median filter

# CCD color sampling

# Color sensing, 3 approaches

- Scan 3 times (temporal multiplexing)
- Use 3 detectors (3-ccd camera, and color film)
- Use offset color samples (spatial multiplexing)

# Typical errors in temporal multiplexing approach

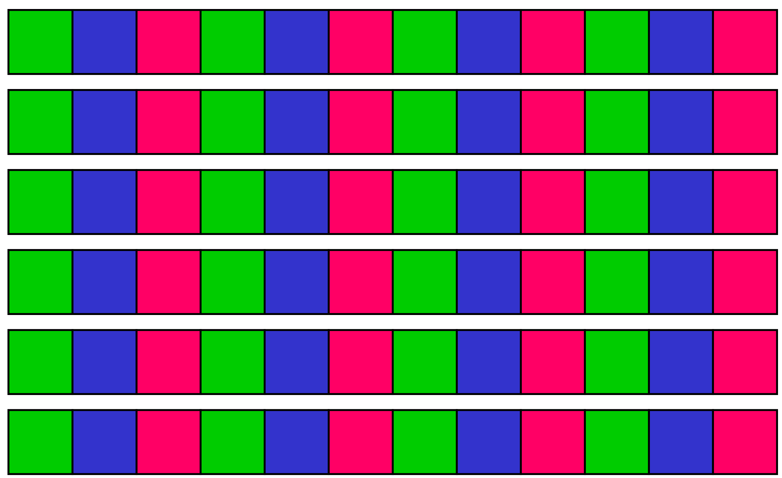
- Color offset fringes

# Typical errors in spatial multiplexing approach.

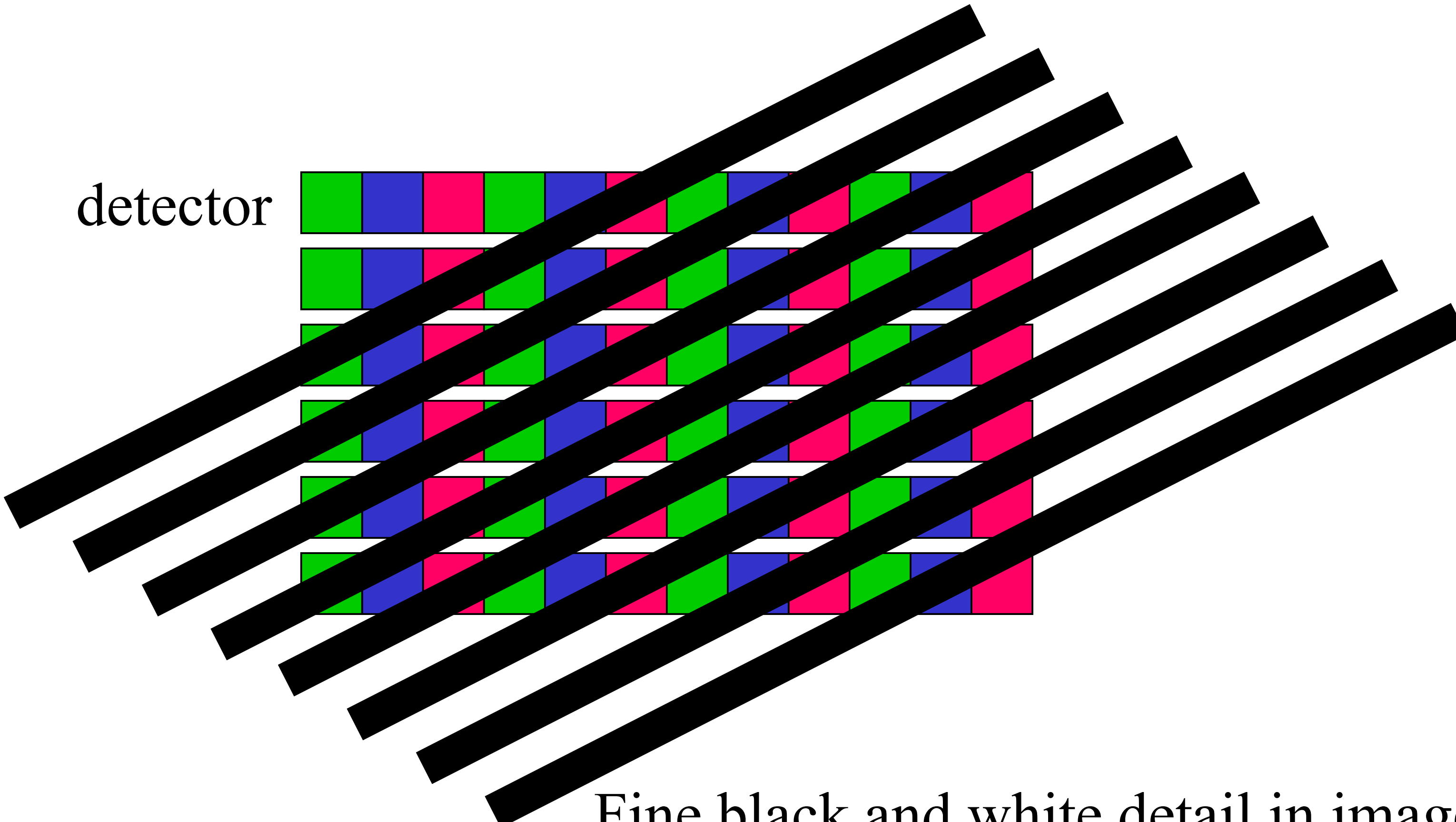
- Color fringes.

# CCD color filter pattern

detector



# The cause of color moire

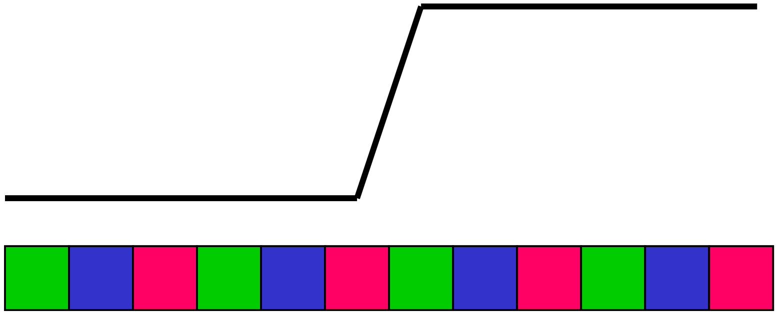


Fine black and white detail in image mis-interpreted as color information.

# Black and white edge falling on color CCD detector

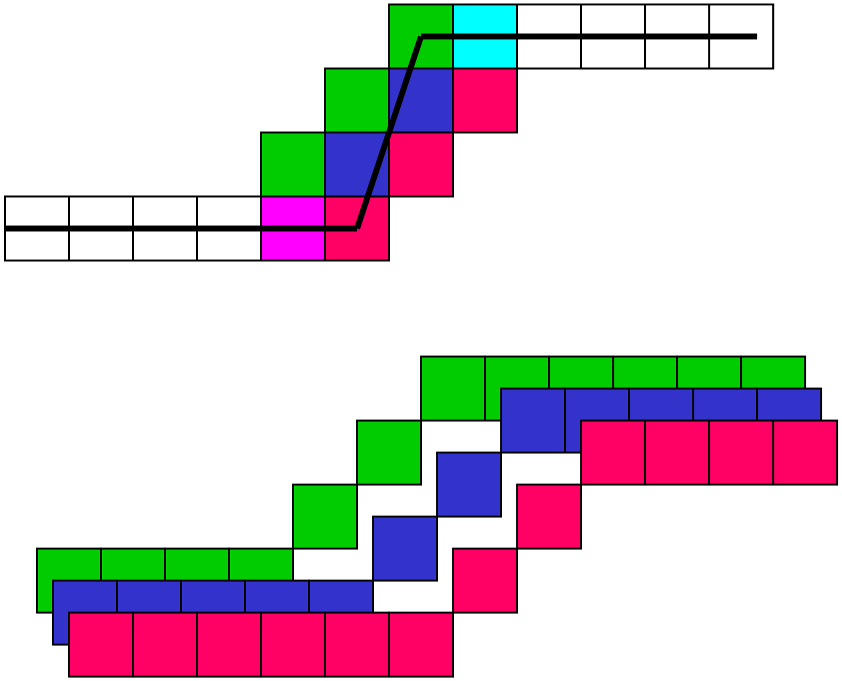
Black and white image (edge)

Detector pixel colors



# Color sampling artifact

Interpolated pixel colors,
for grey edge falling on colored
detectors (linear interpolation).

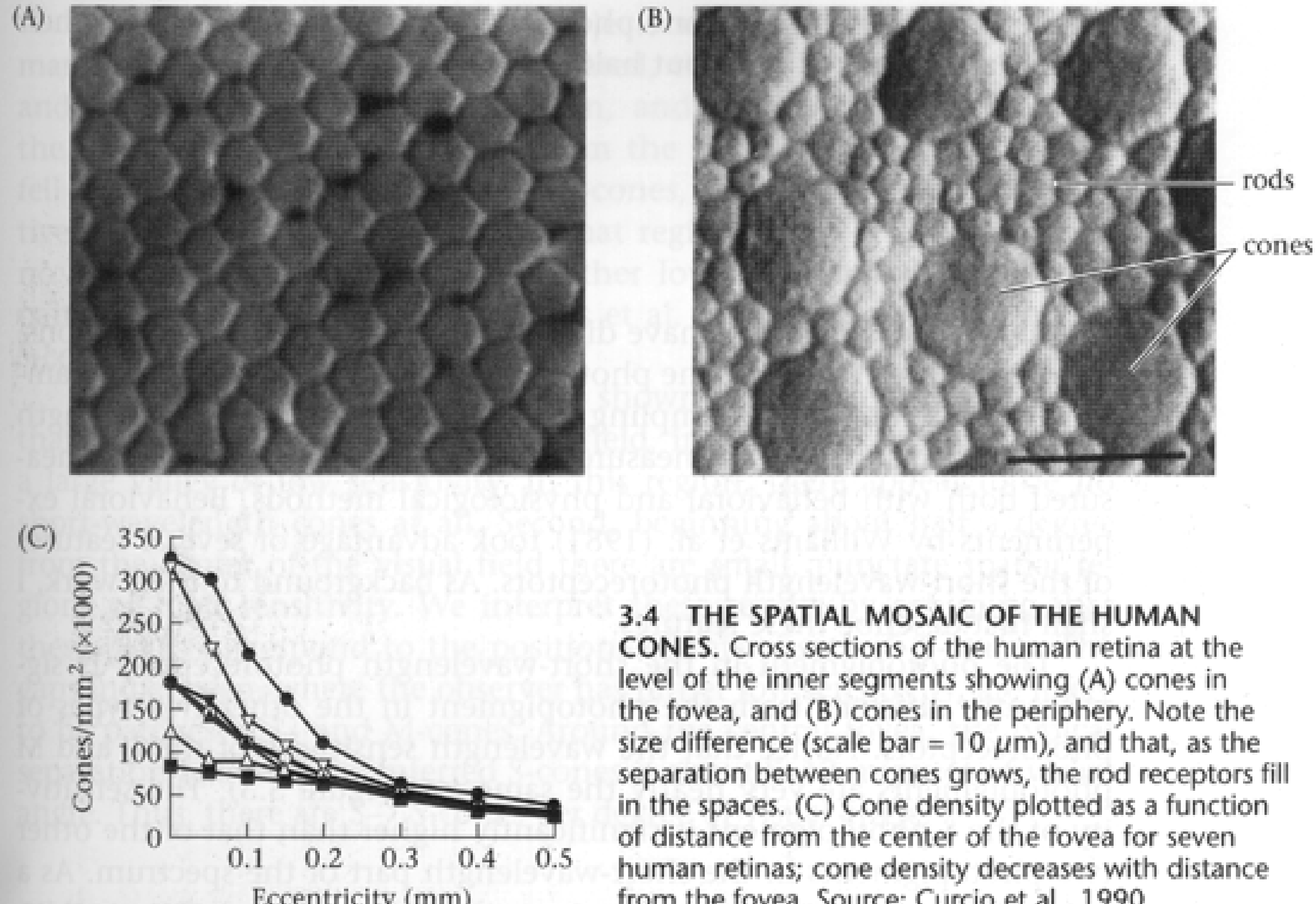


# Typical color moire patterns

Blow-up of electronic camera image. Notice spurious colors in the regions of fine detail in the plants.

# Color sampling artifacts

# Human Photoreceptors

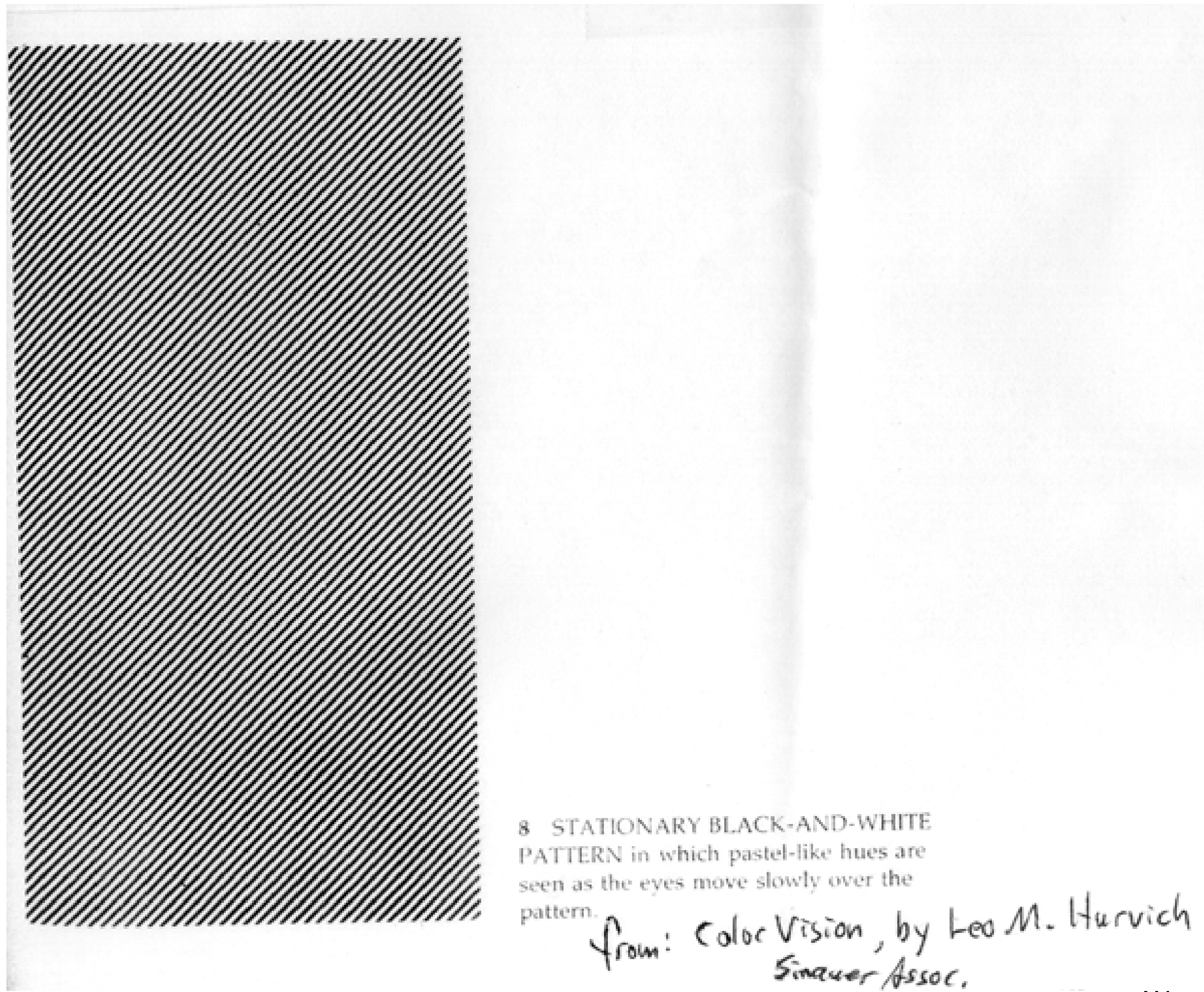



**3.4 THE SPATIAL MOSAIC OF THE HUMAN CONES.** Cross sections of the human retina at the level of the inner segments showing (A) cones in the fovea, and (B) cones in the periphery. Note the size difference (scale bar = 10 $\mu$m), and that, as the separation between cones grows, the rod receptors fill in the spaces. (C) Cone density plotted as a function of distance from the center of the fovea for seven human retinas; cone density decreases with distance from the fovea. Source: Curcio et al., 1990.

(From Foundations of Vision, by Brian Wandell, Sinauer Assoc.)

# Brewster's colors example (subtle).

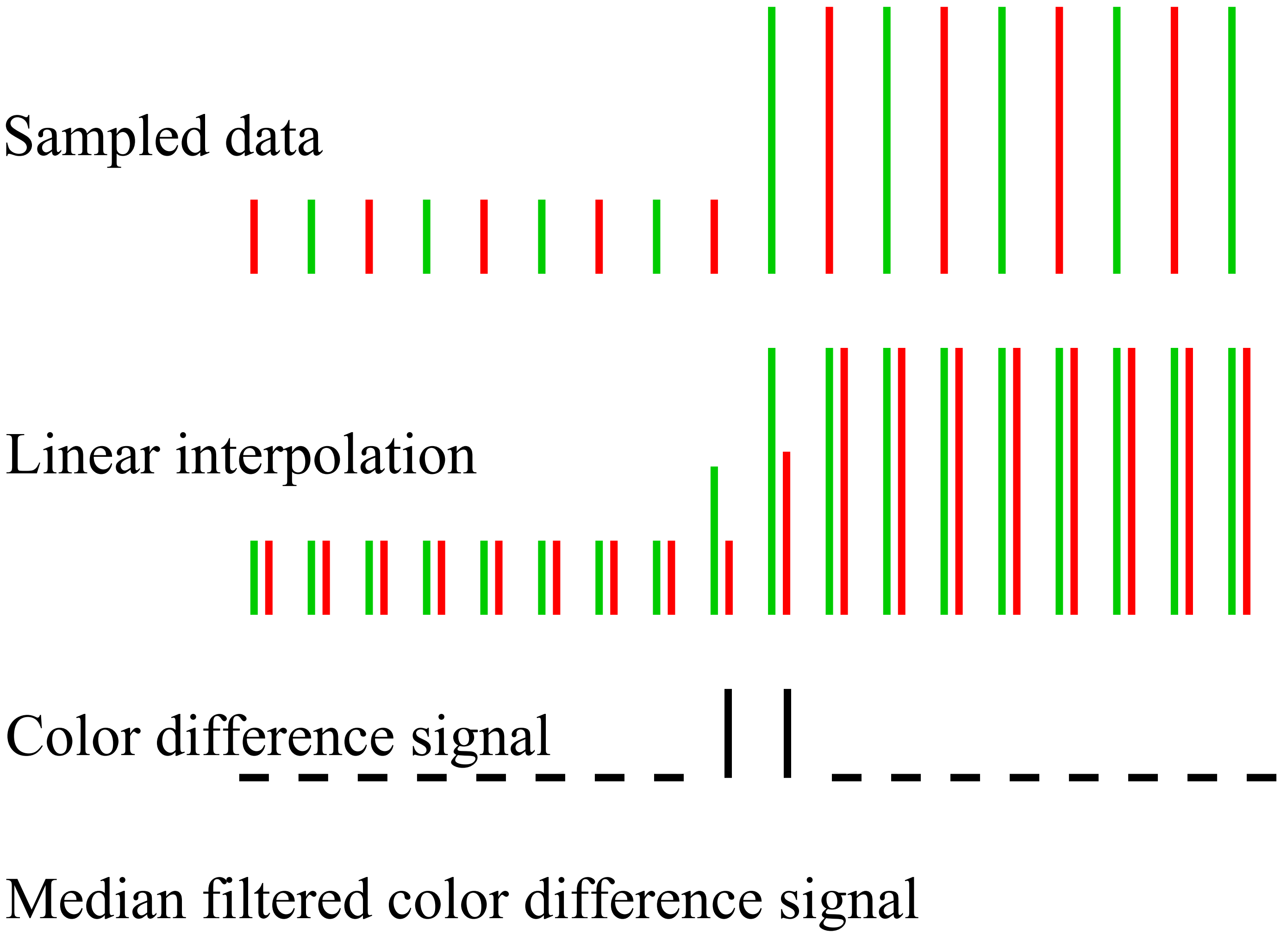
Scale relative to human photoreceptor size:  each line covers about 7 photoreceptors.



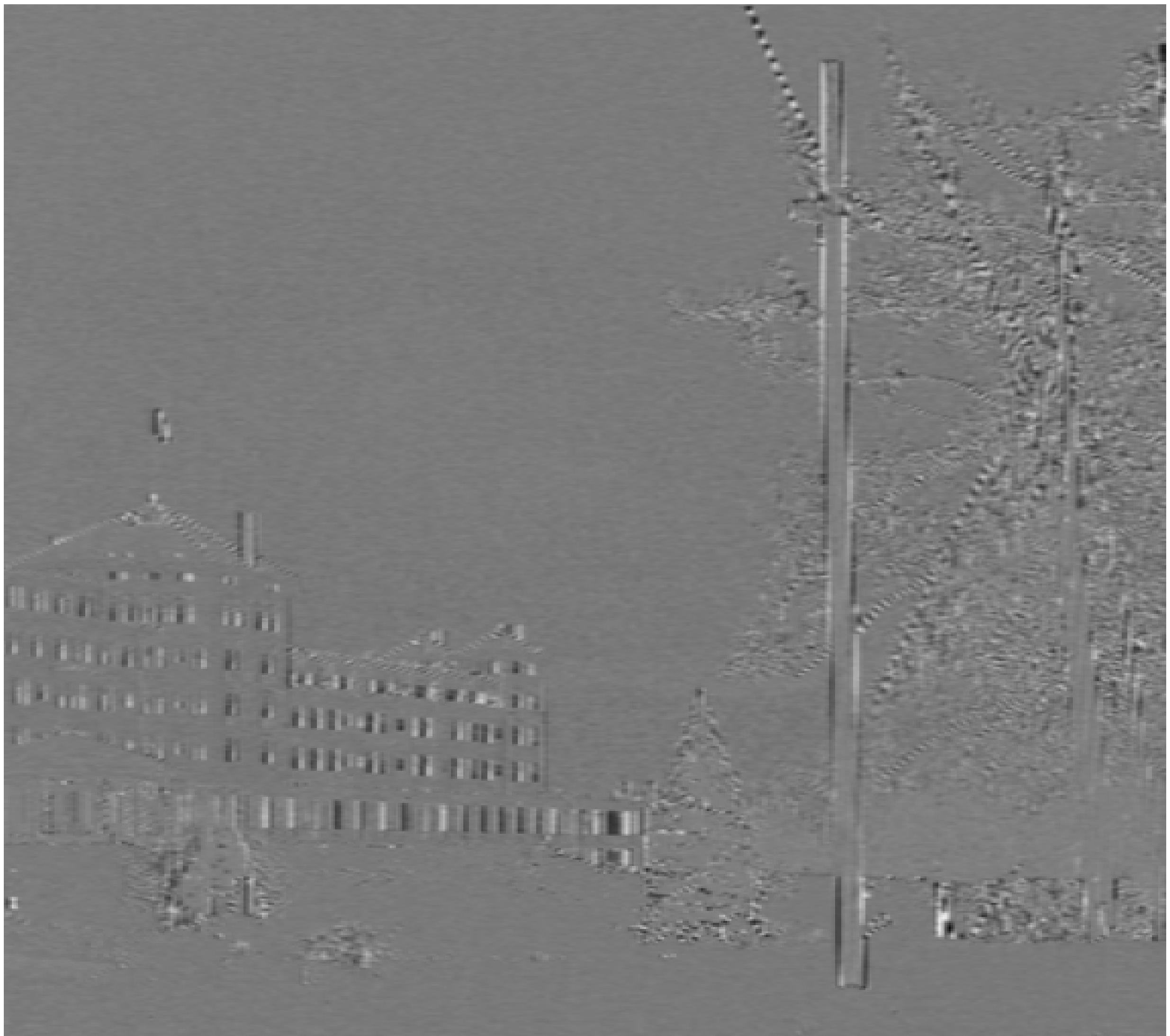
# Median Filter Interpolation

- Perform first interpolation on isolated color channels.
- Compute color difference signals.
- Median filter the color difference signal.
- Reconstruct the 3-color image.

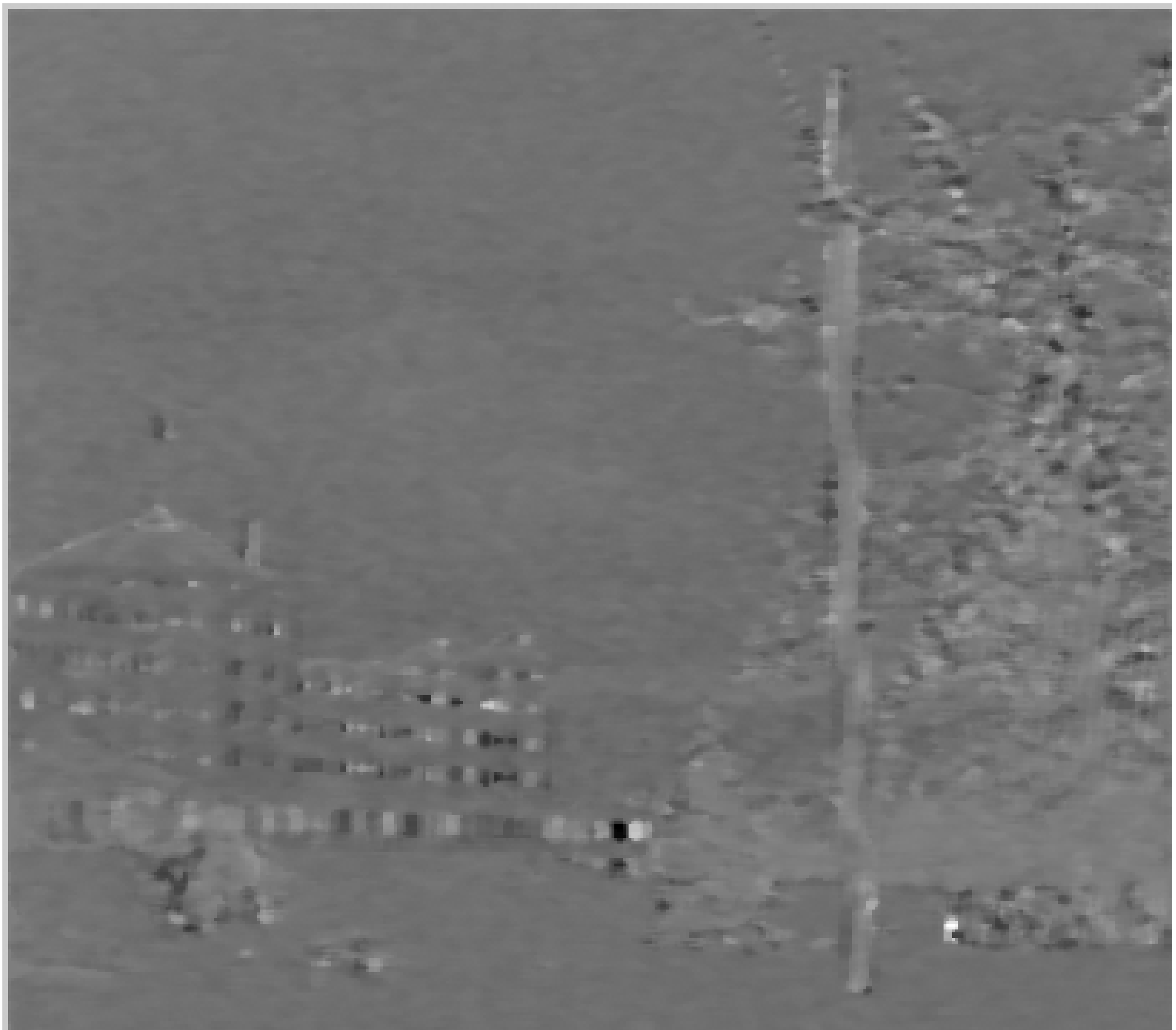
# Two-color sampling of BW edge



# R-G, after linear interpolation



# R – G, median filtered (5x5)



# Recombining the median filtered colors

Linear interpolation

Median filter interpolation

# Didn’t get a chance to show:

Local gain control.