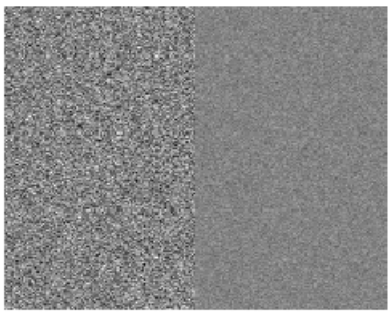
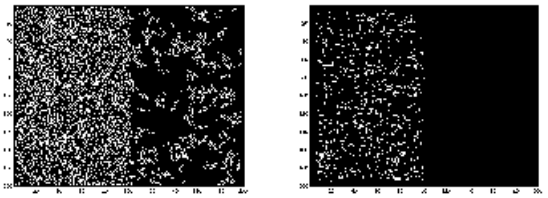
Announcements

- For future problems sets: email matlab code by 11am, due date (same as deadline to hand in hardcopy).
- Today's reading: Chapter 9, except 9.4.

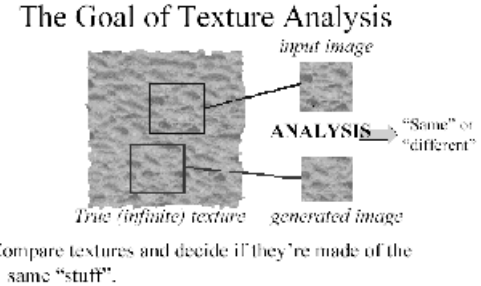
Texture

- Edge detectors find differences in overall intensity.
- Average intensity is only simplest difference.



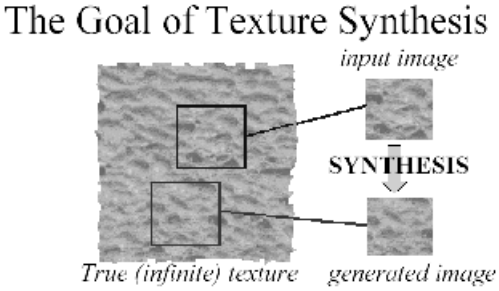


Issues: 1) Discrimination/Analysis



(Freeman)

2) Synthesis



Many more issues

3. Texture boundary detection.
4. Shape from texture.

We'll focus on 1, mention 2.

What is texture?

- Something that repeats with variation.
- Must separate what repeats and what stays the same.
- Model as repeated trials of a random process
  - The probability distribution stays the same.
  - But each trial is different.

Simplest Texture

- Each pixel independent, identically distributed (iid).
- Examples:
  - Region of constant intensity.
  - Gaussian noise pattern.
  - Speckled pattern

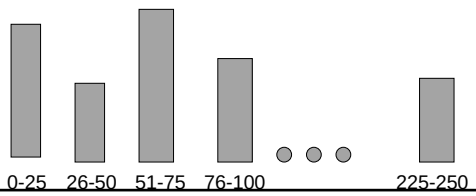
Matlab

Texture Discrimination is then Statistics

- Two sets of samples.
- Do they come from the same random process?

Simplest Texture Discrimination

- Compare histograms.
  - Divide intensities into discrete ranges.
  - Count how many pixels in each range.



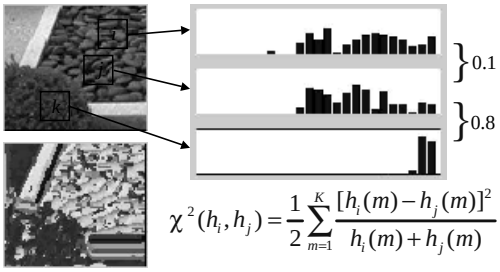
How/why to compare

- Simplest comparison is SSD, many others.
- Can view probabilistically.
  - Histogram is a set of samples from a probability distribution.
  - With many samples it approximates distribution.
  - Test probability samples drawn from same distribution. Ie., is difference greater than expected when two samples come from same distribution?

Matlab

Chi square distance between texton histograms

Chi-square



$$\chi^2(h_i,h_j)=\frac{1}{2}\sum_{m=1}^{K}\frac{[h_i(m)-h_j(m)]^2}{h_i(m)+h_j(m)}$$

(Malik)

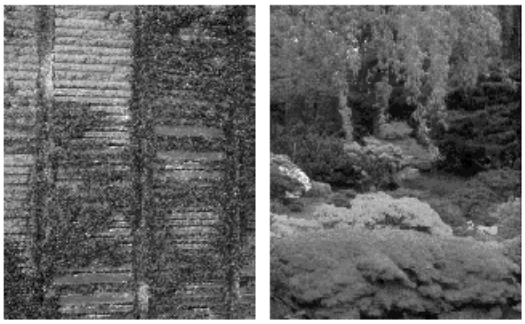
More Complex Discrimination

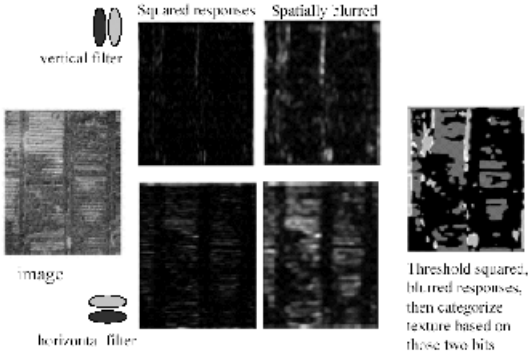
- Histogram comparison is very limiting
  - Every pixel is independent.
  - Everything happens at a tiny scale.

Matlab

- Use output of filters of different scales.

Example (Forsyth & Ponce)





What are Right Filters?

- Multi-scale is good, since we don't know right scale a priori.
- Easiest to compare with naïve Bayes:

Filter image one: *(F1, F2, …)*

Filter image two: *(G1, G2, …)*

S means image one and two have same texture.

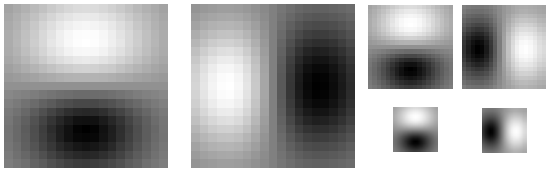
Approximate: *P(F1,G1,F2,G2, …| S)*

By P(*F1,G1|S)**P(*F2,G2|S)*…*

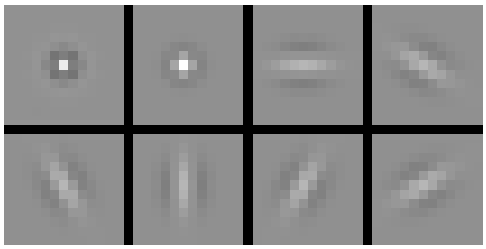
What are Right Filters?

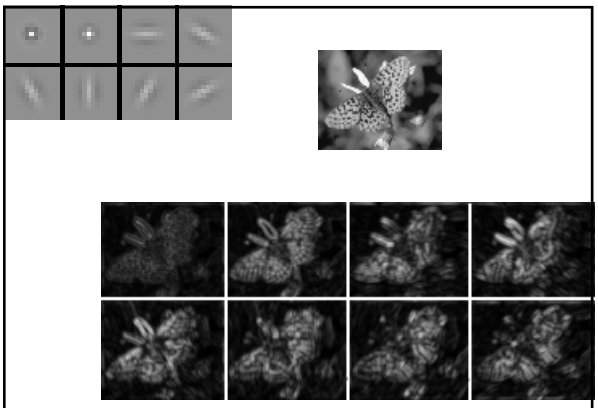
- The more independent the better.
  - In an image, output of one filter should be independent of others.
  - Because our comparison assumes independence.
  - Wavelets seem to be best.

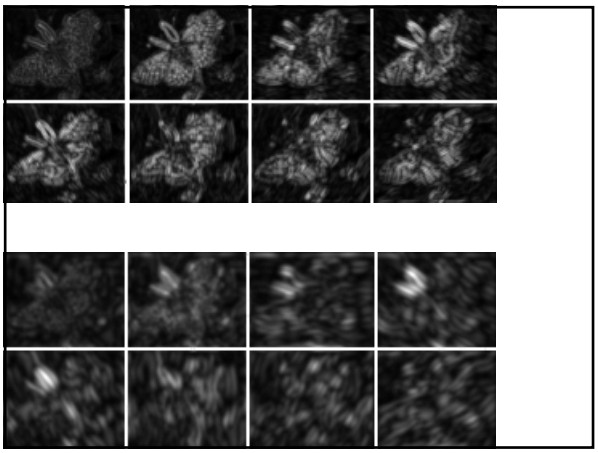
Difference of Gaussian Filters



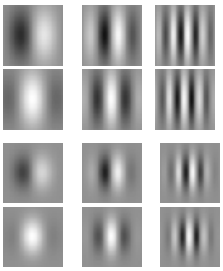
Spots and Oriented Bars (Malik and Perona)







Gabor Filters



Gabor filters at different scales and spatial frequencies

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

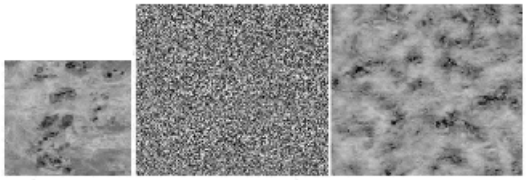
$$\cos(k_x x + k_y y)\exp-\left\{\frac{x^2+y^2}{2\sigma^2}\right\}$$

Matlab

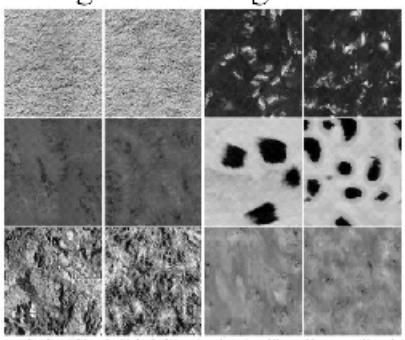
Gabor filters are examples of Wavelets

- We know two bases for images:
  - Pixels are localized in space.
  - Fourier are localized in frequency.
- Wavelets are a little of both.
- Good for measuring frequency locally.

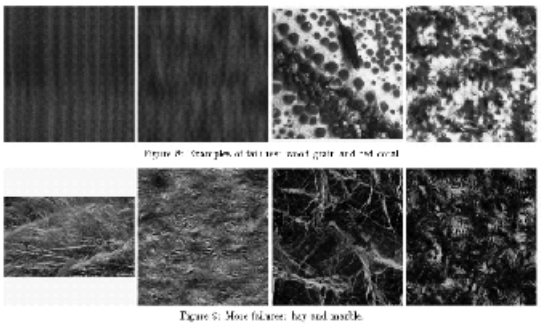
Synthesis with this Representation (Bergen and Heeger)



Bergen and Heeger results



Bergen and Heeger failures



Markov Model

- Captures local dependencies.
  - Each pixel depends on neighborhood.
- Example, 1D first order model

$P(p1, p2, \ldots pn) = P(p1)*P(p2|p1)*P(p3|p2,p1)*\ldots$

$= P(p1)*P(p2|p1)*P(p3|p2)*P(p4|p3)*\ldots$

Example 1st Order Markov Model

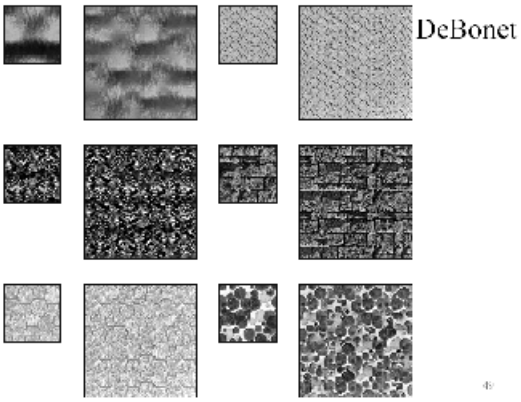
- Each pixel is like neighbor to left + noise with some probability.

Matlab

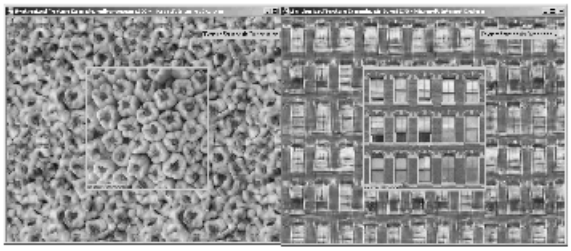
- These capture a much wider range of phenomena.

There are dependencies in Filter Outputs

- Edge
  - Filter responds at one scale, often does at other scales.
  - Filter responds at one orientation, often doesn't at orthogonal orientation.
- Synthesis using wavelets and Markov model for dependencies:
  - DeBonet and Viola
  - Portilla and Simoncelli



Portilla and Simoncelli

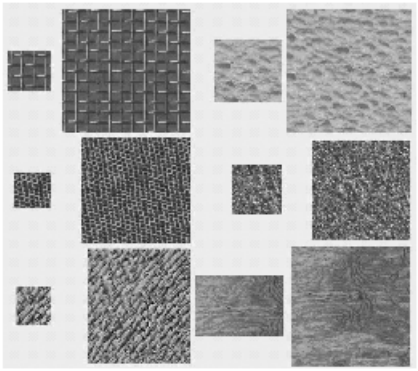


We can do this without filters

- Each pixel depends on neighbors.

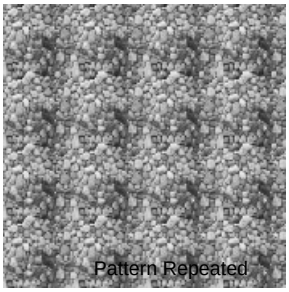
1. As you synthesize, look at neighbors.
2. Look for similar neighborhood in sample texture.
3. Copy pixel from that neighborhood.
4. Continue.

Efros and Leung

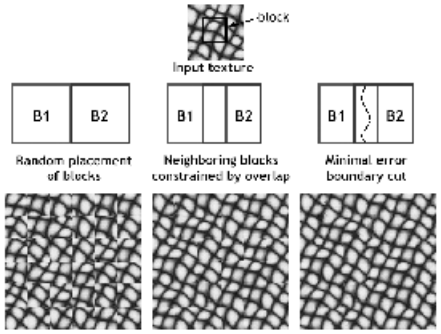


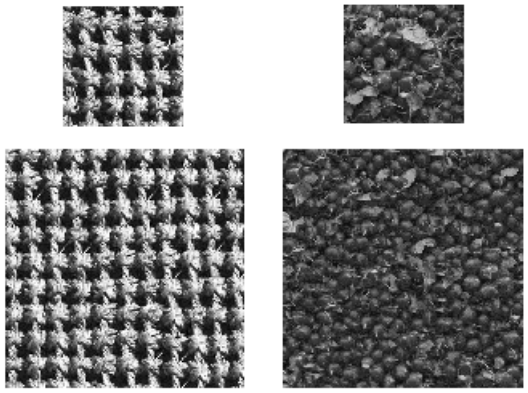
This is like copying, but not just repetition

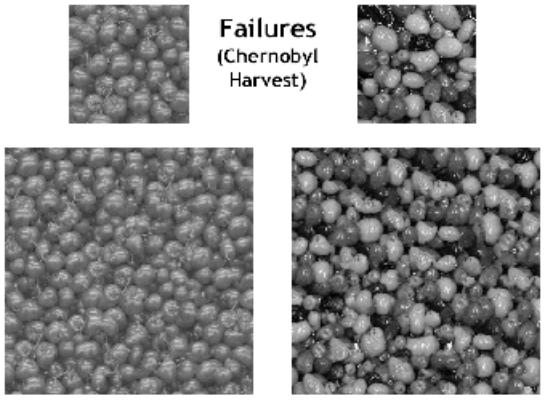
Photo



With Blocks







Conclusions

- Model texture as generated from random process.
- Discriminate by seeing whether statistics of two processes seem the same.
- Synthesize by generating image with same statistics.

To Think About

- 3D effects
  - Shape: Tiger's appearance depends on its shape.
  - Lighting: Bark looks different with light angle
- Given pictures of many chairs, can we generate a new chair?