Multiscale

Announcements

- Demo of PS4
- Problems compiling c on Macs?

Image Matching

- Matching whole images
  - For alignment, eg., mosaicing
- Matching small regions
  - Eg., for stereo
- Matching Objects

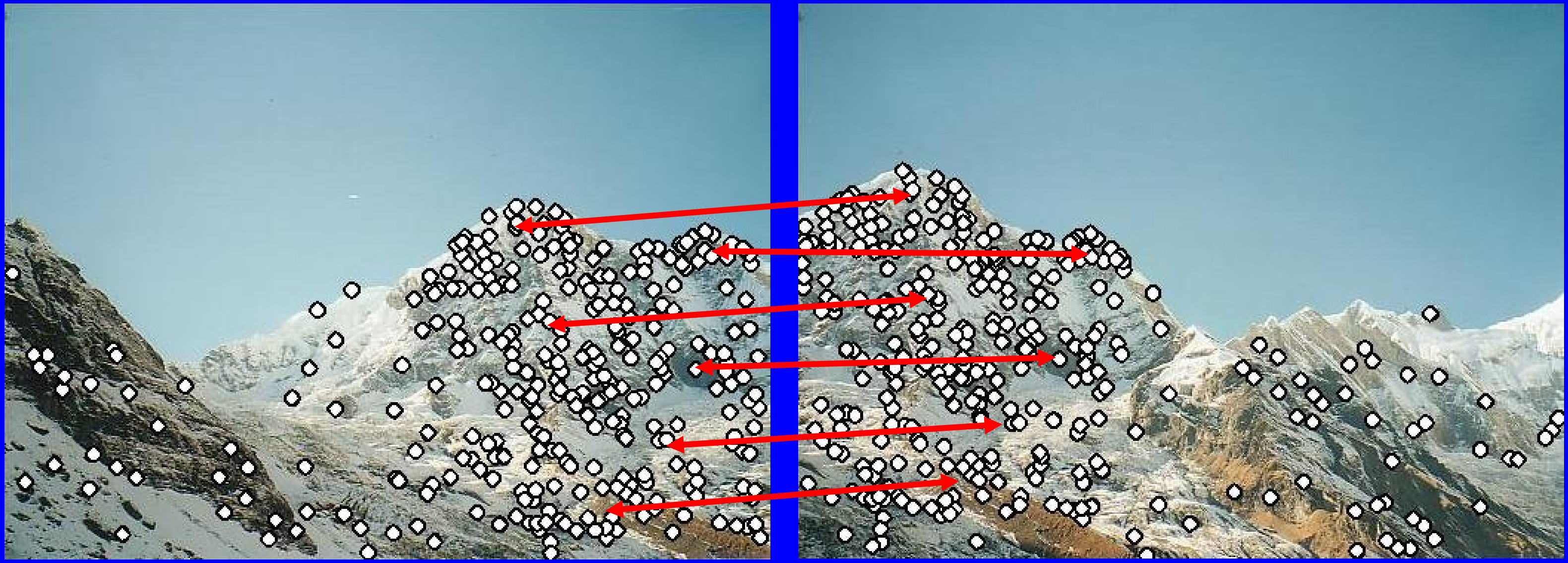
Feature-based Matching

1. Find distinctive features
    - Corners, blobs, MSER…
2. Describe region around feature
    - Intensities, SIFT, …
3. Compare features to find matches
    - Local matches: Histogram comparison, normalized correlation…
    - Global matches: RANSAC
4. Use these matches
    - Find rigid alignment of images, compute disparity from each match, compute similarity score.

Example: Mosaicing

(Slides from Lazebnik)

Why extract features?



Step 1: extract features

Step 2: match features

(Slides from Lazebnik)

Why extract features?

Step 1: extract features
Step 2: match features
Step 3: align images

(Slides from Lazebnik)

Plan for next 5 classes

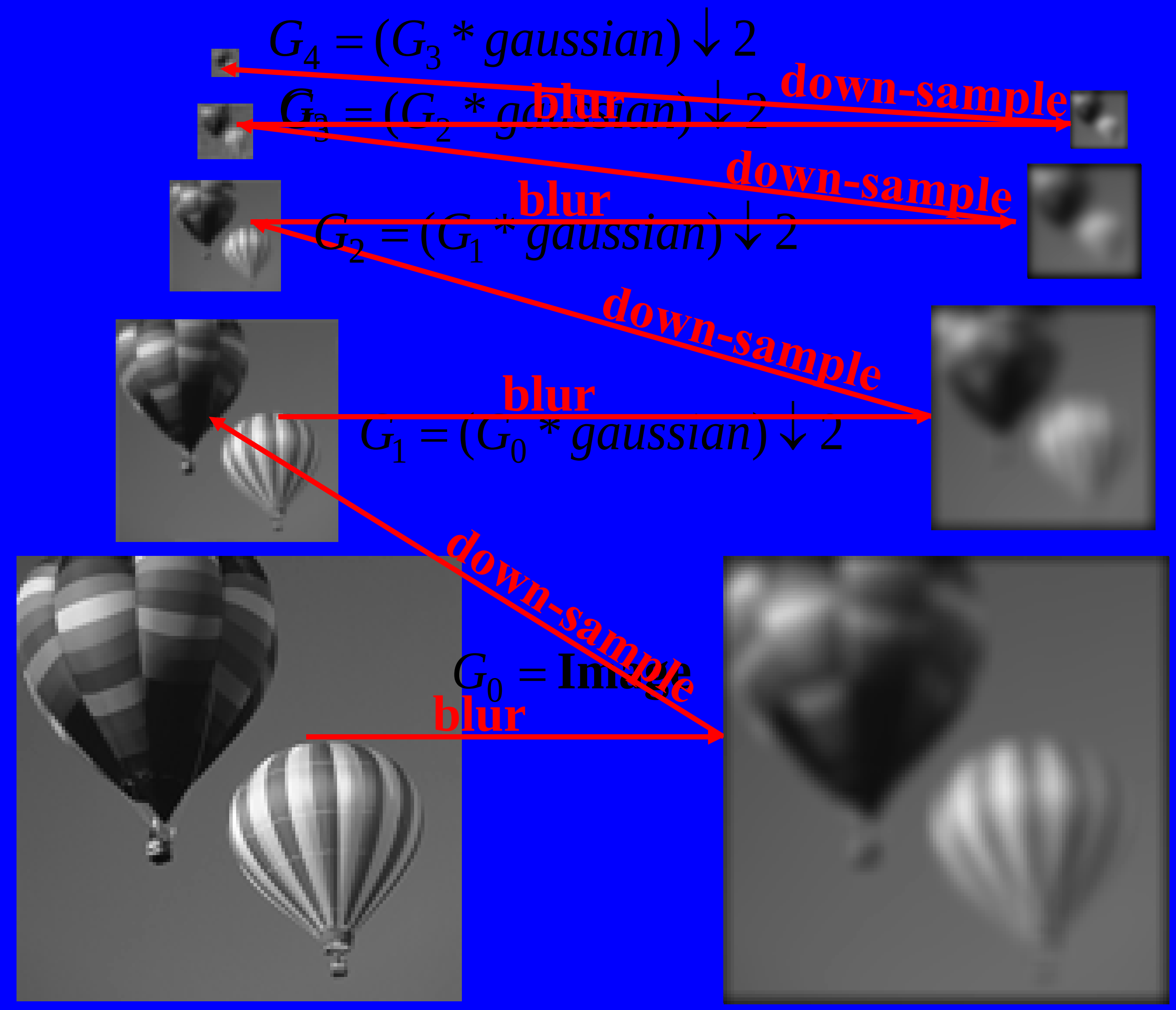
- Features
  - Blobs
    - Multiscale
    - While we’re talking about this, brief detour into texture.
  - Corners
- SIFT descriptors
  - Histogram of gradients
  - Compare using histogram comparison
- Alignment
- Matching – RANSAC
- Putting Mosaicing together

Why Multiscale

- To look at images at different sizes
- To analyze images at different scales.
  - Eg., key points (such as blobs) might exist at different scales.
  - Eg., video with large motion will have small motion at after we shrink images.
- Efficiency; fewer operations needed for small image.

The Gaussian Pyramid

Low resolution

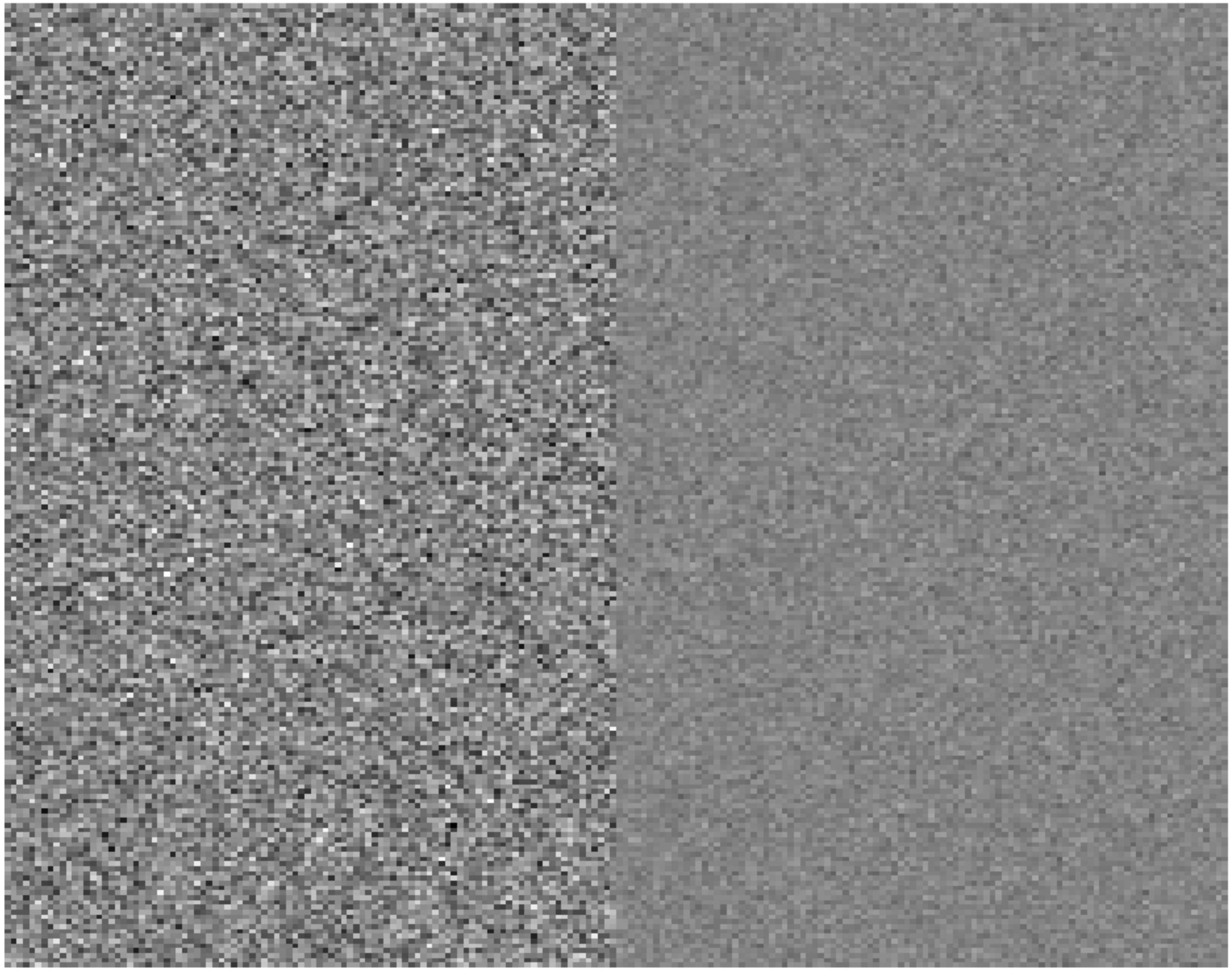


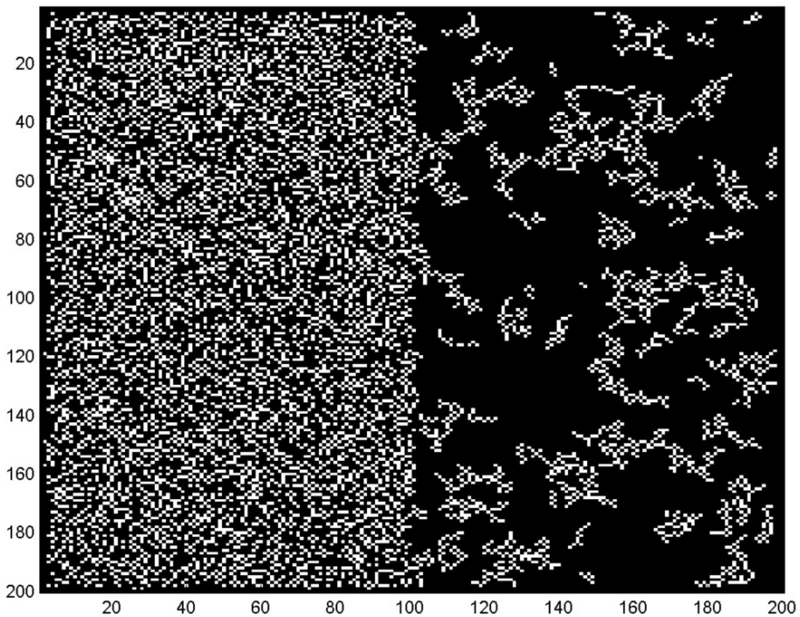
High resolution

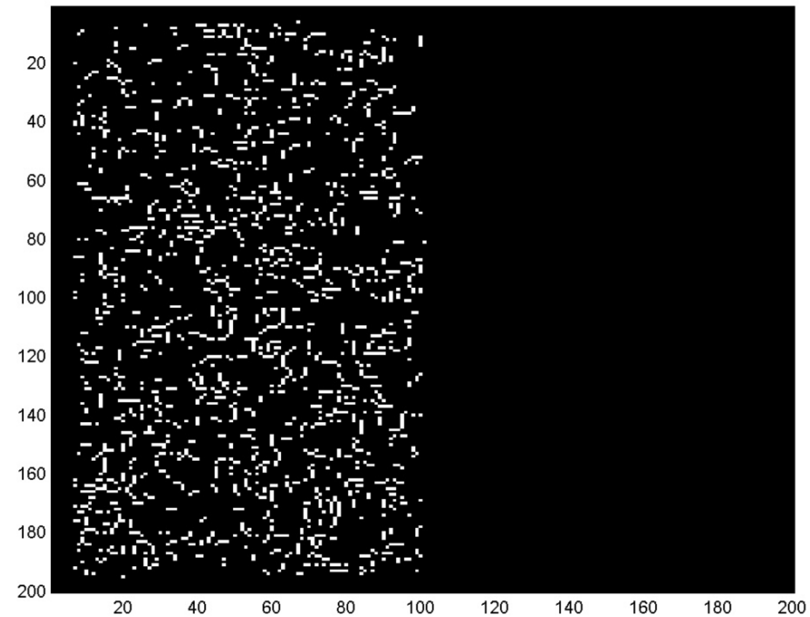
(Weizmann Institute Vision Class)

Texture

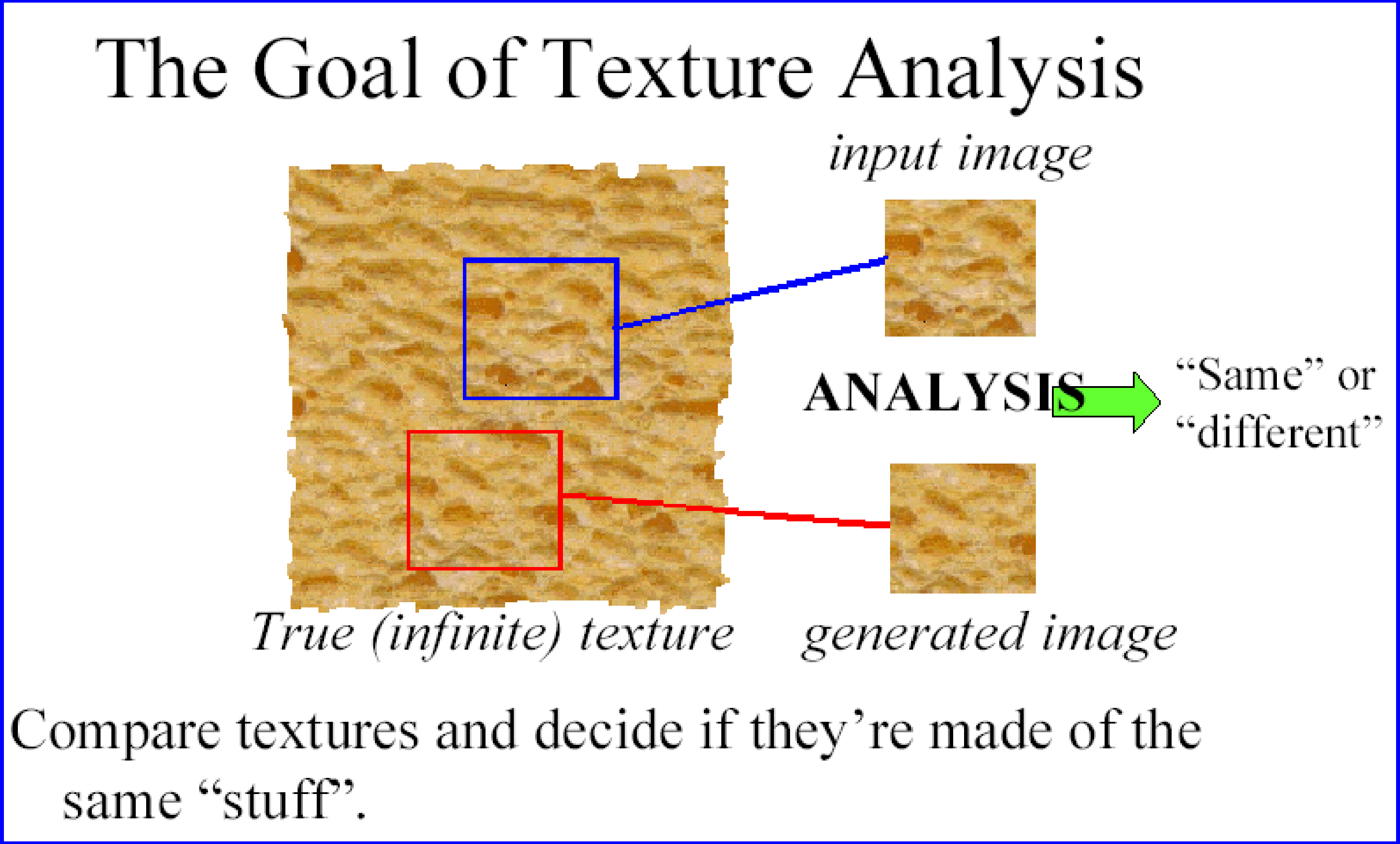
- Edge detectors find differences in overall intensity.
- Average intensity is only simplest difference.




20
40
60
80
100
120
140
160
180
200
20 40 60 80 100 120 140 160 180 200

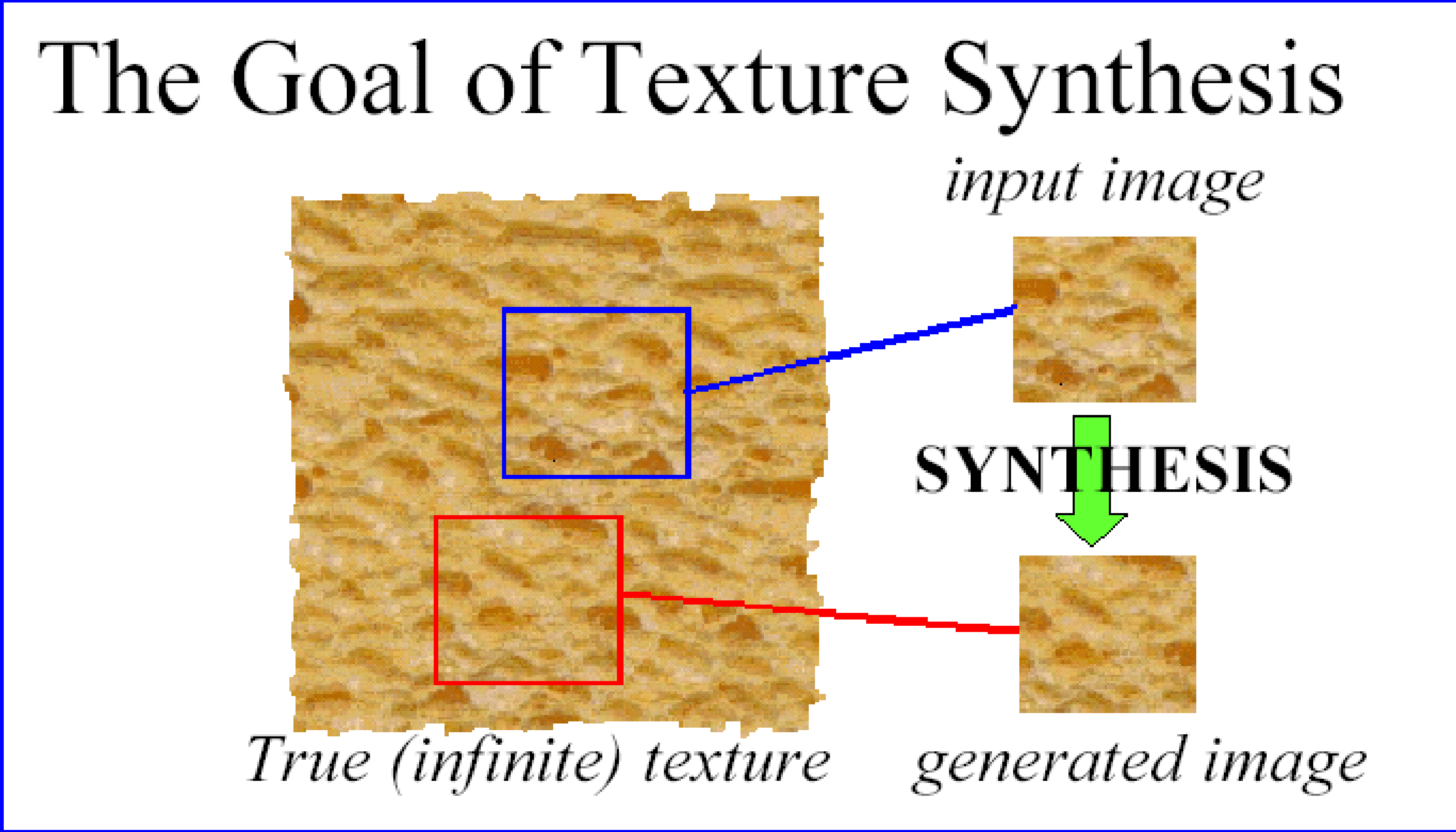

20
40
60
80
100
120
140
160
180
200
20 40 60 80 100 120 140 160 180 200

Issues: 1) Discrimination/Analysis



(Freeman)

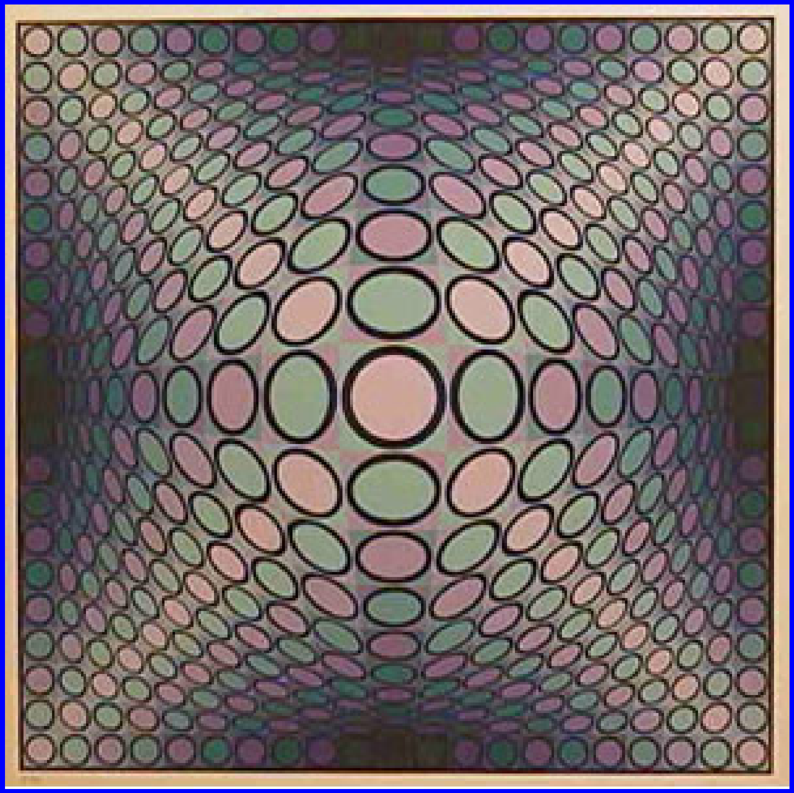
2) Synthesis


The Goal of Texture Synthesis
input image
SYNTHESIS
True (infinite) texture
generated image

Many more issues

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

.



(www.cmap.polytechnique.fr/~maureen/vasarely3.jpg)

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.
  - This may be true (eg., pile of objects)
  - Or not really (tile floor).

Simplest Texture

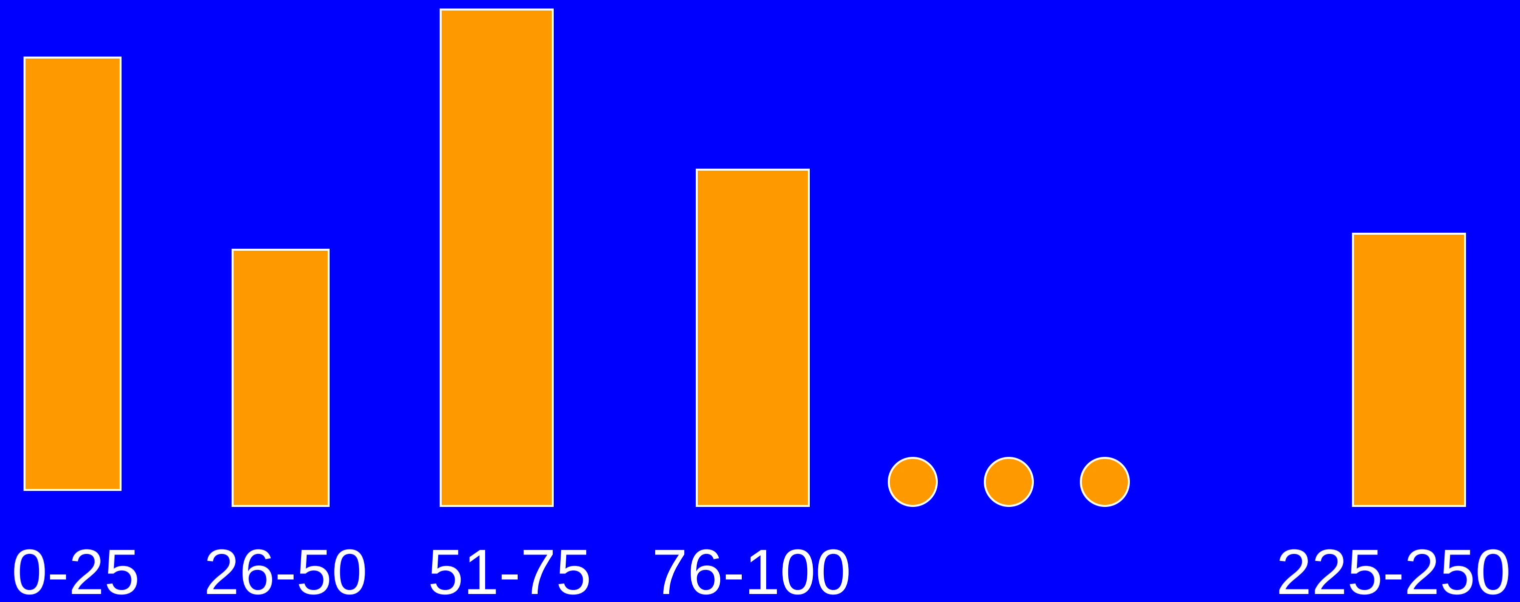
- Each pixel independent, identically distributed (iid).
- Examples:
  - Region of constant intensity.
  - Gaussian noise pattern.
  - Speckled pattern

Texture Discrimination is then Statistics

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

Simplest Texture Discrimination

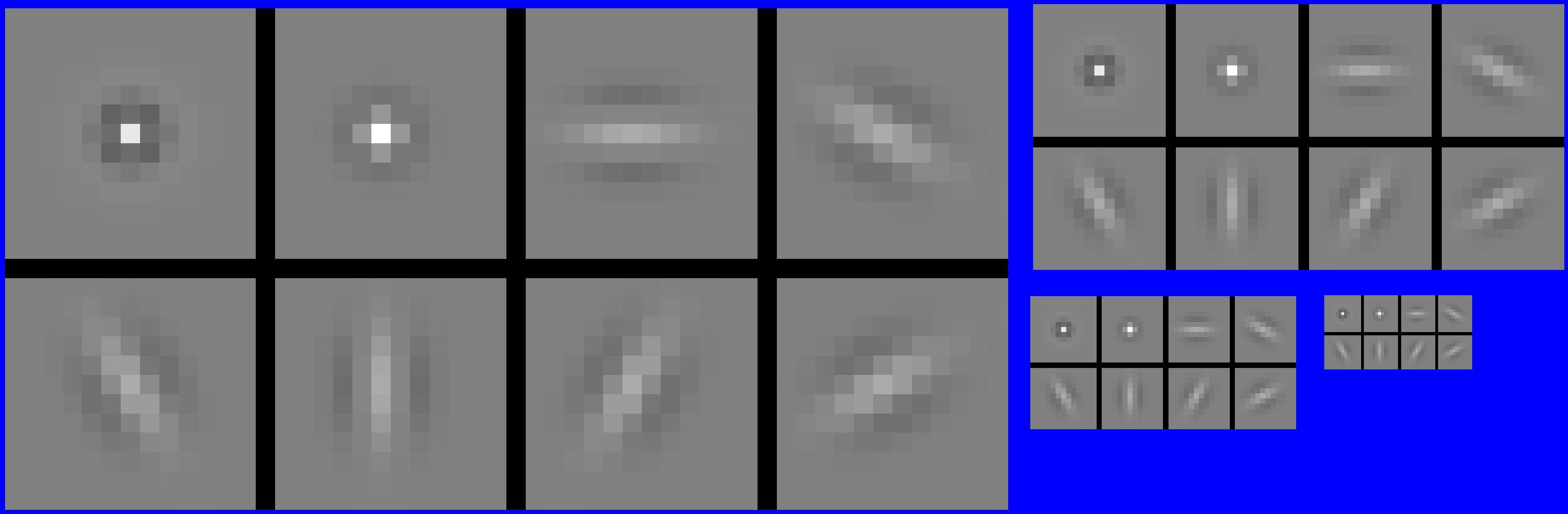
- Compare sample distributions (histograms).
  - Divide intensities into discrete ranges.
  - Count how many pixels in each range.



Histograms of Filters

- We saw two textures with different intensity histograms.
- Square vs. dots
  - Same intensity histogram
  - Different histograms of derivatives
- Uniform squares vs. clustered
  - Same histograms of derivatives
  - Different if you smooth first
- Bright Diamonds vs. squares.
  - Same horizontal and vertical derivatives
  - Different when considered jointly (or different orientations)

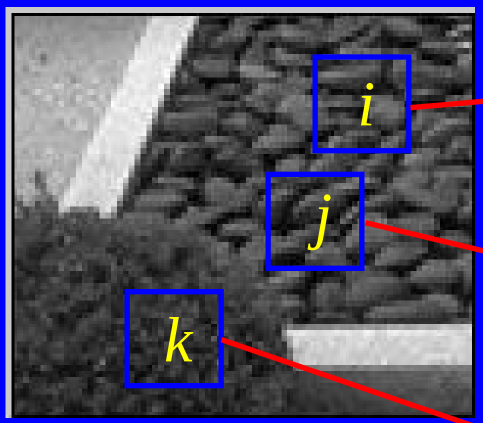
Multi-scale Spots and Oriented Bars
(Malik and Perona)

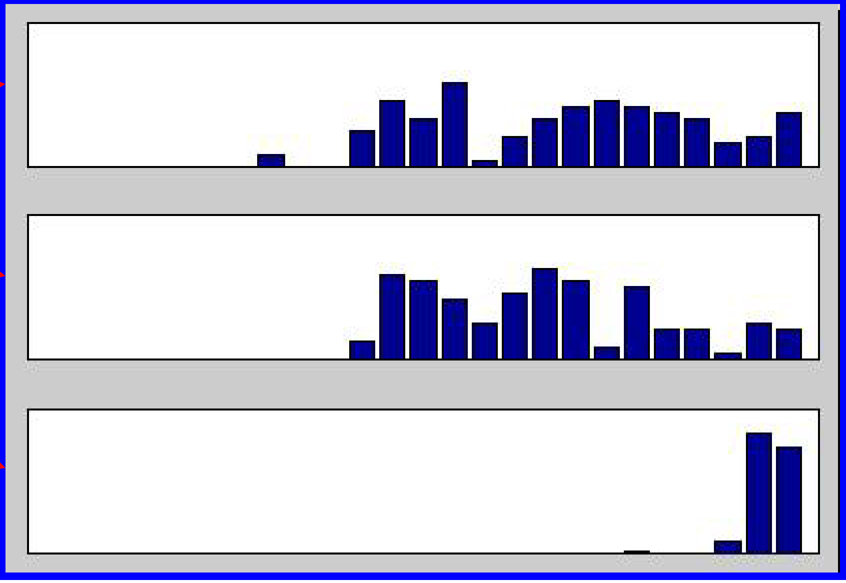


Textons

- Apply bank of filters.
- Represent each pixel with high-dimensional vector (one dim. for each filter).
- Cluster with K-means
- Build histogram using cluster center.

Chi square distance between texton histograms

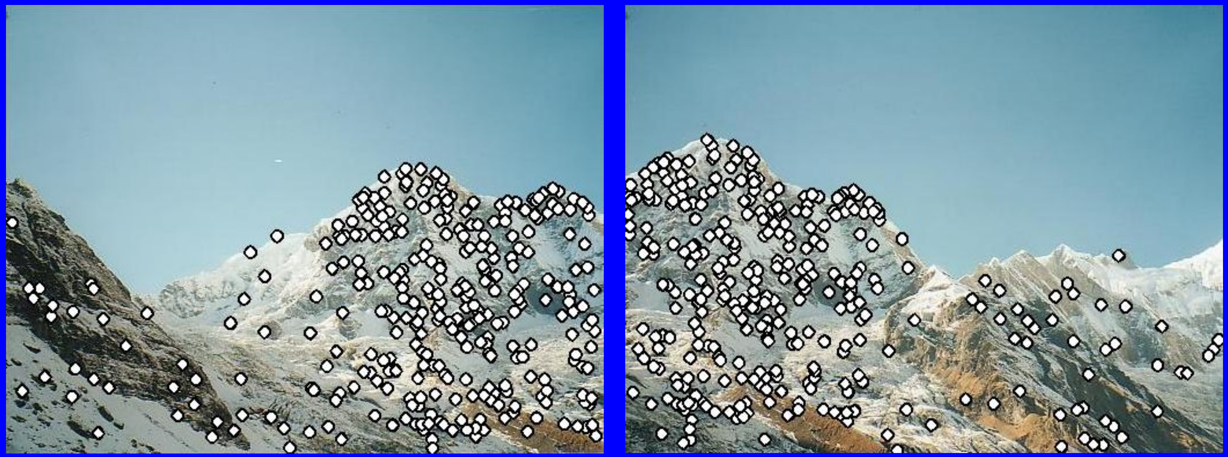




$$\chi^2(h_i, h_j) = \frac{1}{2}\sum_{m=1}^{K}\frac{[h_i(m) - h_j(m)]^2}{h_i(m) + h_j(m)}$$

(Malik)

Characteristics of good features



- Repeatability
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency
  - Each feature has a distinctive description
- Compactness and efficiency
  - Many fewer features than image pixels
- Locality
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion

(Slides from Lazebnik)

Distinctive Features

- A point is distinctive when it looks different from its neighbors.
- A blob also looks different from neighbors at different scales.

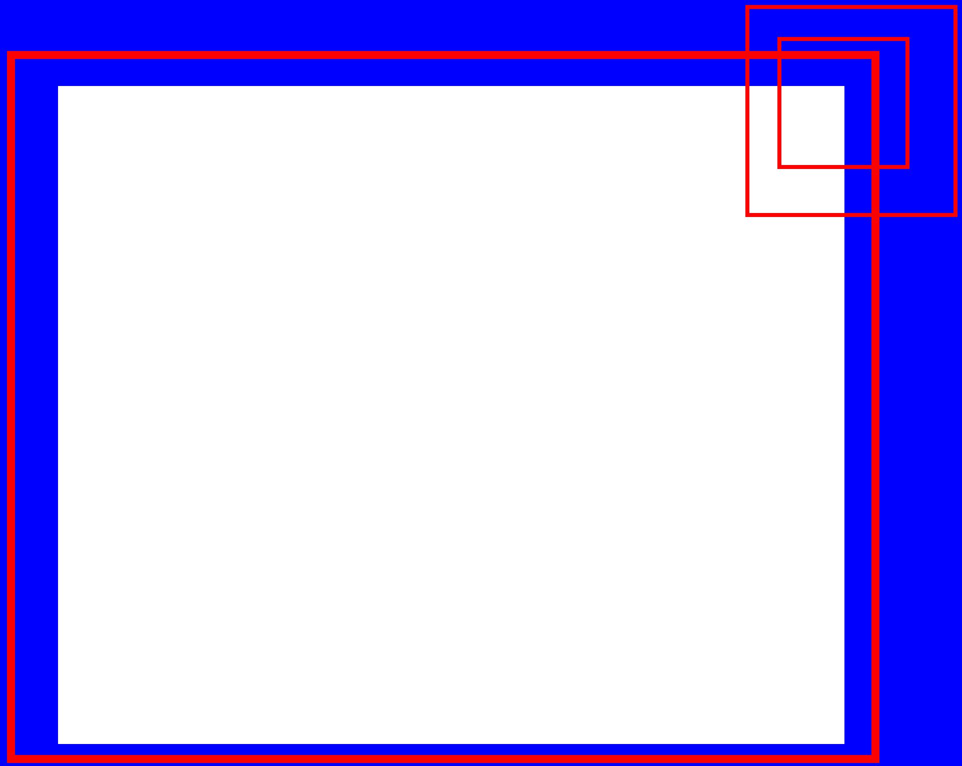
But at a corner, moving a little changes the scenery

Move this square a bit and the region under it looks the same.

Same here.

A corner (in computer vision) is technically defined as a region where the intensity pattern changes a lot with any small translatiion.

However, a corner isn’t well localized in scale.

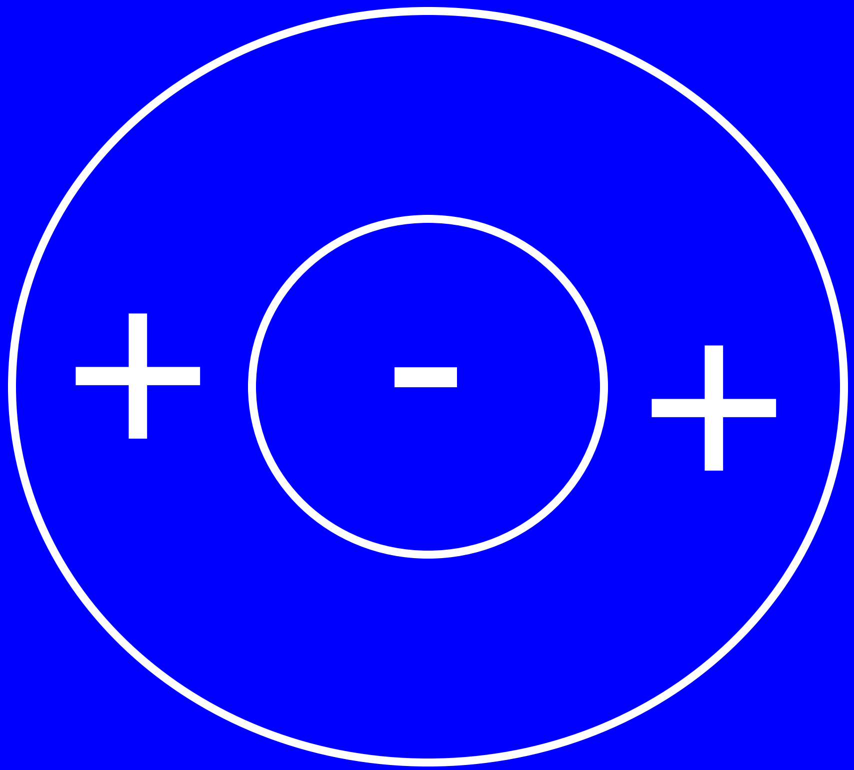


For that we need blobs

What is a good blob detector?

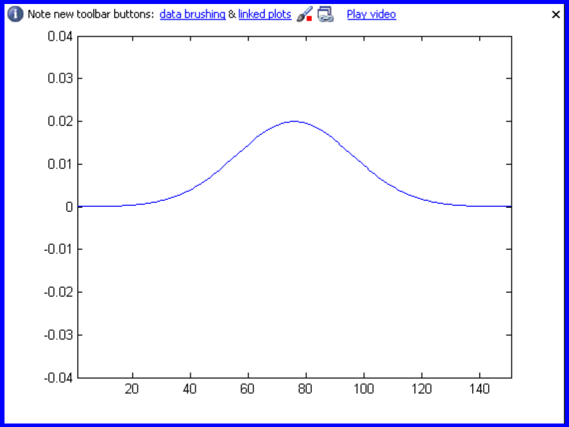
- A filter that has versions at multiple scales.
- The biggest response should be when the filter has the same location and scale as the blob.

Center-Surround Filter

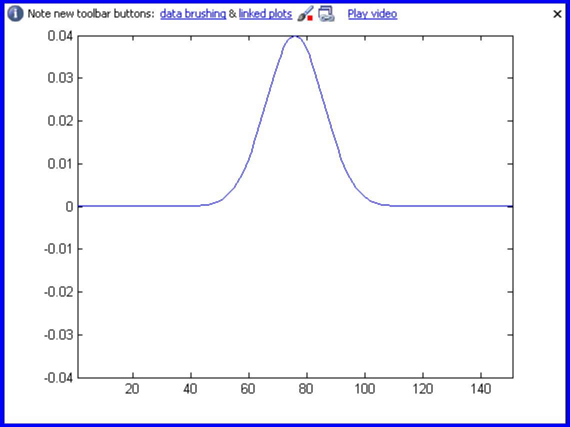


- When does this have biggest response?
  - When inside is as dark as possible
  - And outside is as light as possible.
  - Ie, a dark spot.
  - Note, this locates position and scale.
- Similar filters are in animals (eg., frog).

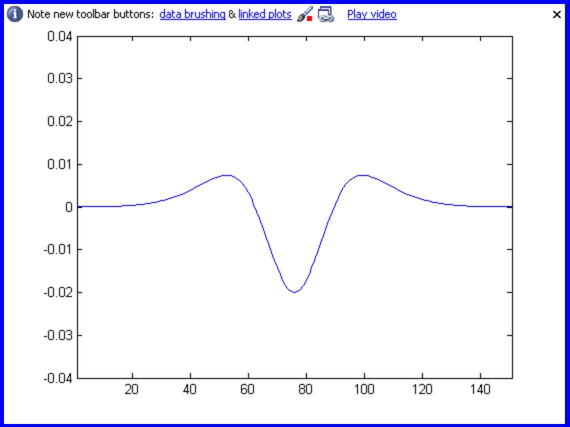
Difference of Gaussian



-



=



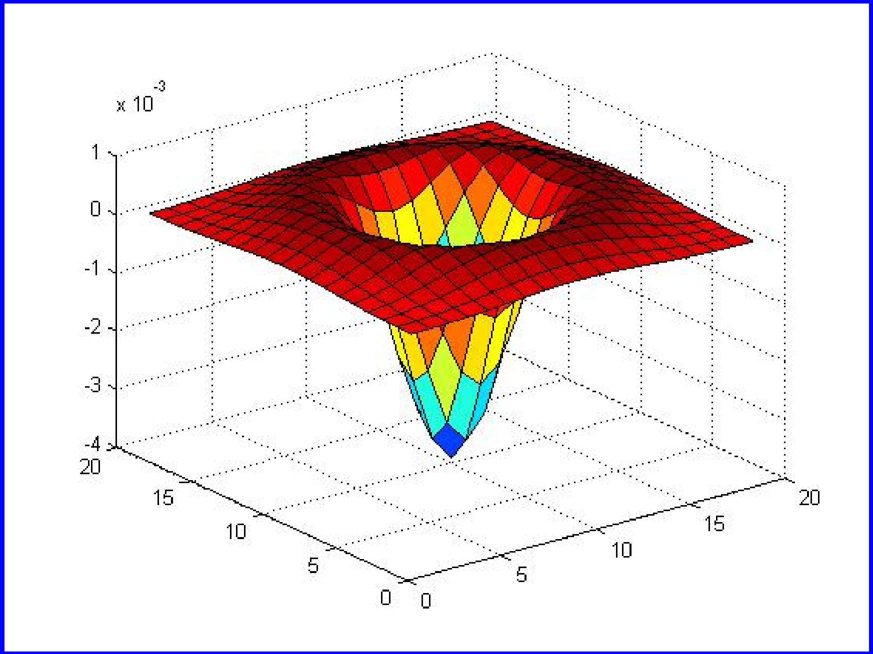
Basic Algorithm

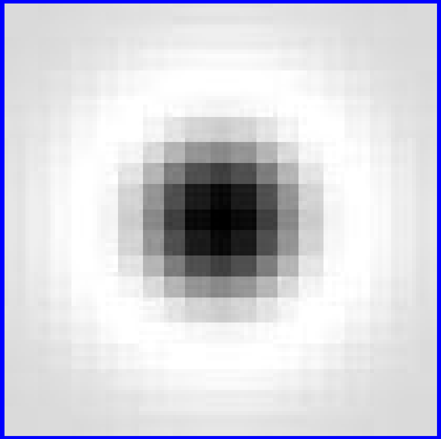
- Filter with Gaussian at different scales
  - This is done by just repeatedly filtering with the same Gaussian.
- Subtract image filtered at one scale with image filtered at previous scale.
- Look for local extrema
  - A pixel is bigger (smaller) than all eight neighbors, and all nine neighboring pixels at neighboring scales.

Which Scales

- More scales can produce greater accuracy.
- But also more expense.
- We are taking a derivative, so need to be careful about denominator.
  - It turns out that we should increase scale multiplicatively. Sigma, k*sigma, k*k*sigma….
  - Sigma = 1.6 produces reasonable results.
  - k = cuberoot(2). (These values are heuristic).

- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D





$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

(Slides from Lazebnik)

Efficient implementation

- Approximating the Laplacian with a difference of Gaussians:

$$L = \sigma^2 \left(G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right)$$

(Laplacian)

$$DoG = G(x, y, k\sigma) - G(x, y, \sigma)$$

(Difference of Gaussians)