CS 1674: Intro to Computer Vision

Texture Representation + Image Pyramids

Prof. Adriana Kovashka
University of Pittsburgh
September 14, 2016

Reminders/Announcements

- HW2P due tonight, 11:59pm
- HW3W, HW3P out
- Shuffle!

Plan for today

- Filtering
  - Representing texture
  - Application to subsampling
  - Image pyramids
- Detecting interesting content (start)

Convolution vs. correlation

Convolution

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u, j-v]$$

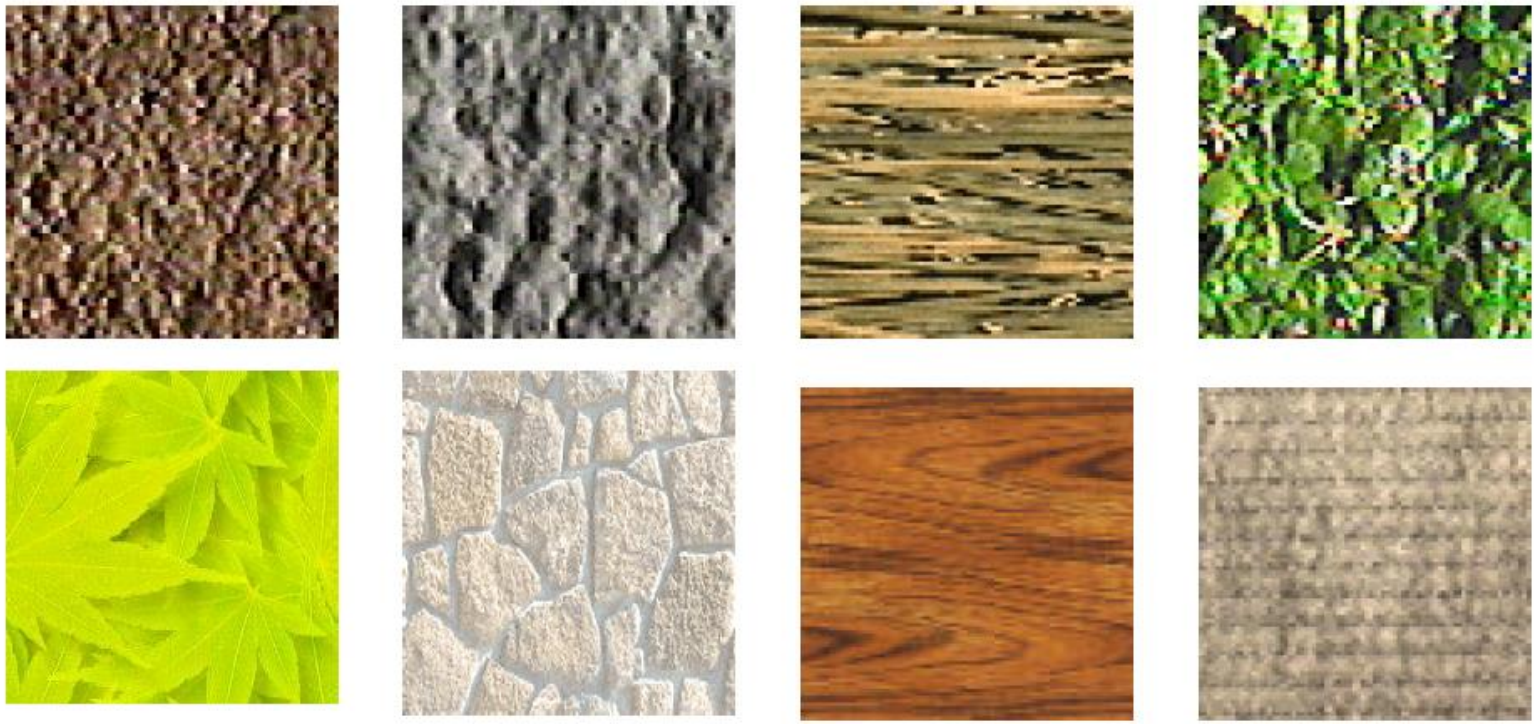
$$G = H \star F$$

Cross-correlation

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u, j+v]$$

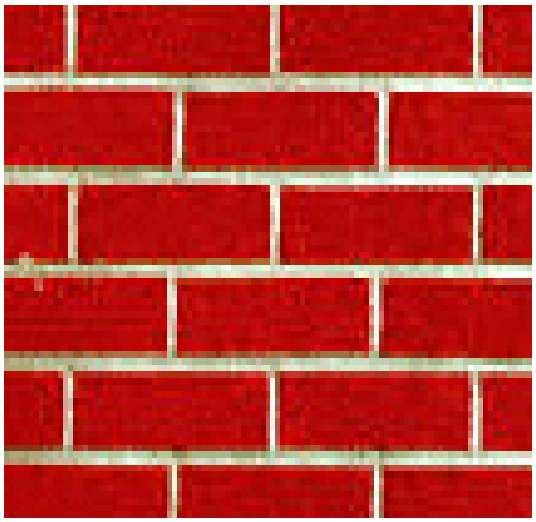
$$G = H \otimes F$$

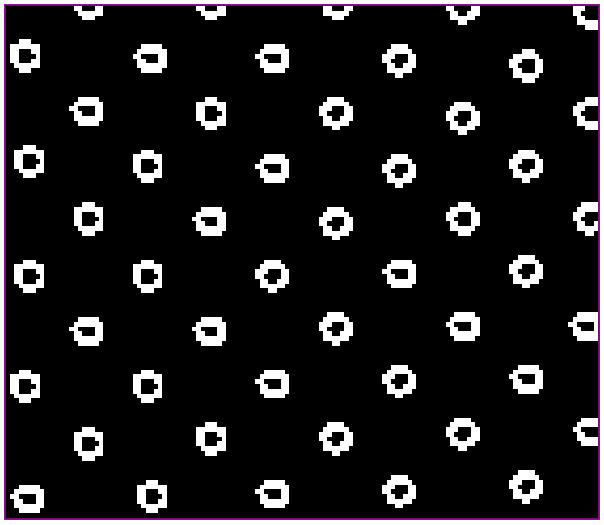
Texture

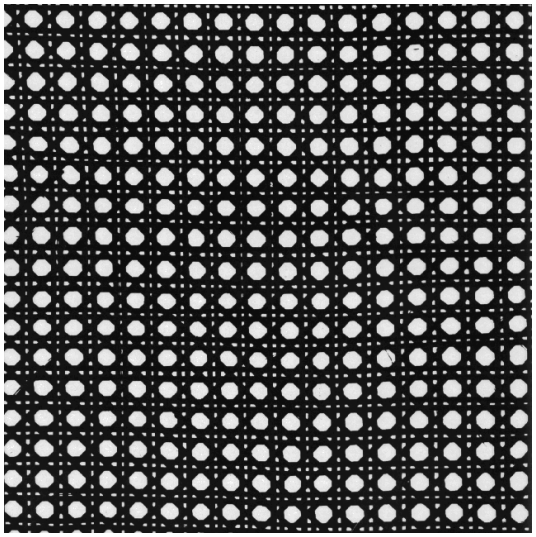


What defines a texture?

Includes: more regular patterns

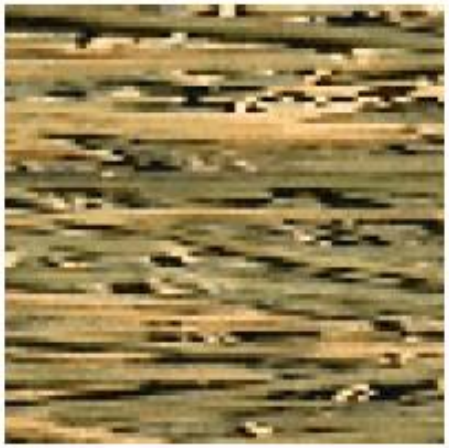
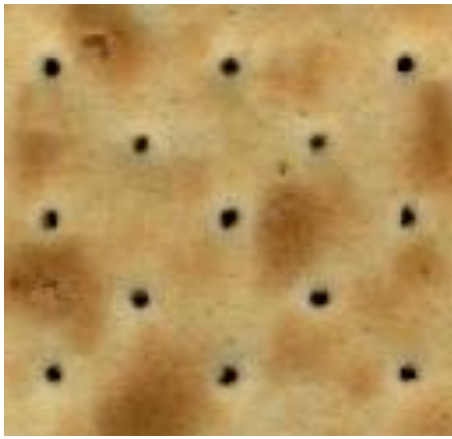






Includes: more random patterns



Kristen Grauman

Kristen Grauman

http://animals.nationalgeographic.com/

Why analyze texture?

- Important for how we perceive objects
- Often indicative of a material's properties
- Can be important appearance cue, especially if shape is similar across objects
- To represent objects, we want a feature one step above "building blocks" of filters, edges

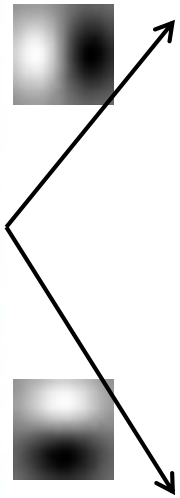
Adapted from Kristen Grauman

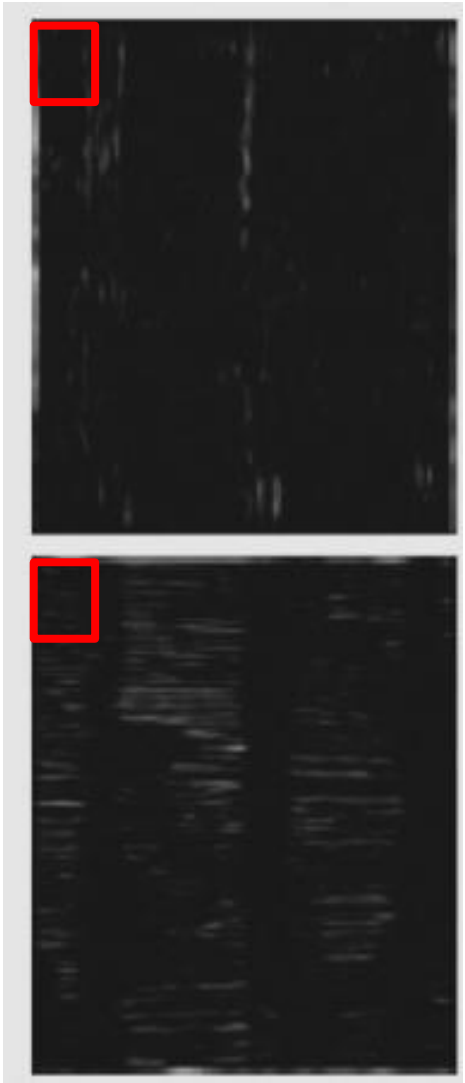
Texture representation

- Textures are made up of repeated local patterns, so:
  - Find the patterns
    - Use filters that *look like* patterns (spots, bars, raw patches…)
    - Consider magnitude of response
  - Describe their statistics within each local window
    - Mean, standard deviation
    - Histogram

Texture representation: example

original image





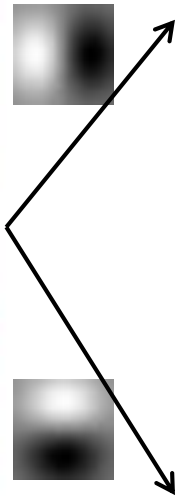
derivative filter responses, squared

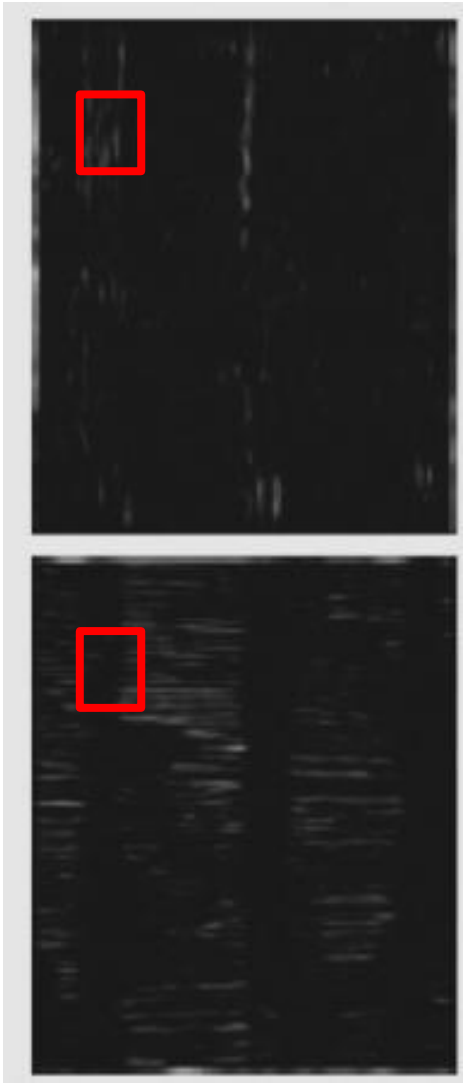
	mean d/dx value	mean d/dy value
Win. #1	4	10

statistics to summarize patterns in small windows

Texture representation: example

original image





derivative filter responses, squared

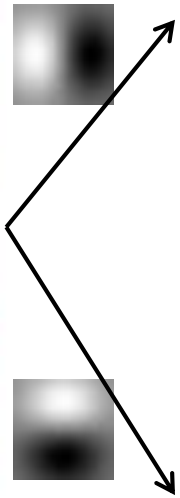
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7

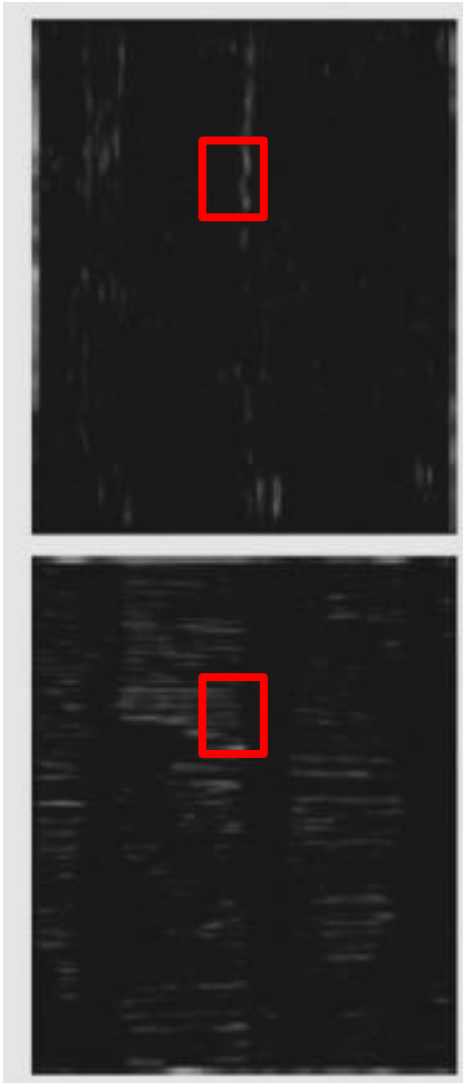
⋮

statistics to summarize patterns in small windows

Texture representation: example

original image





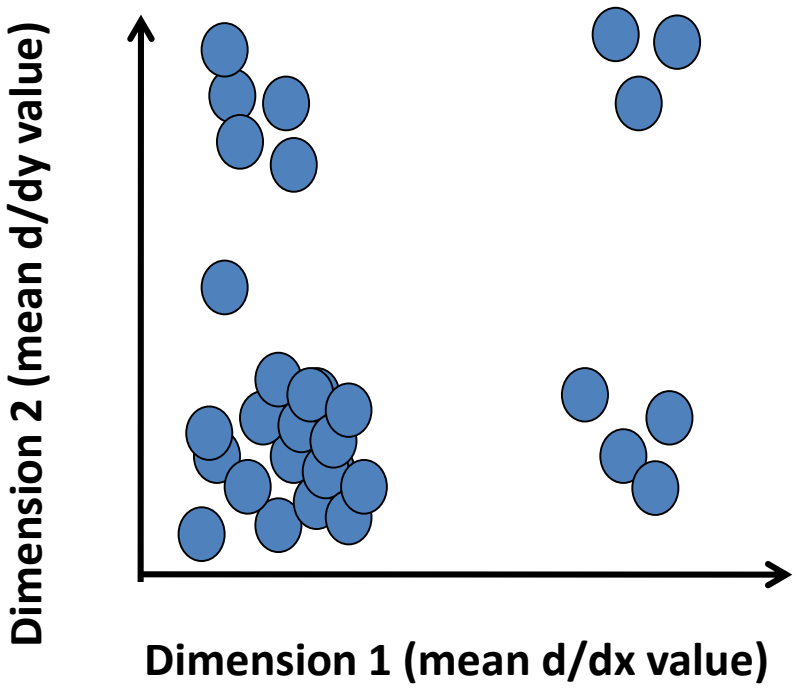
derivative filter responses, squared

	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20

⋮

statistics to summarize patterns in small windows

Texture representation: example

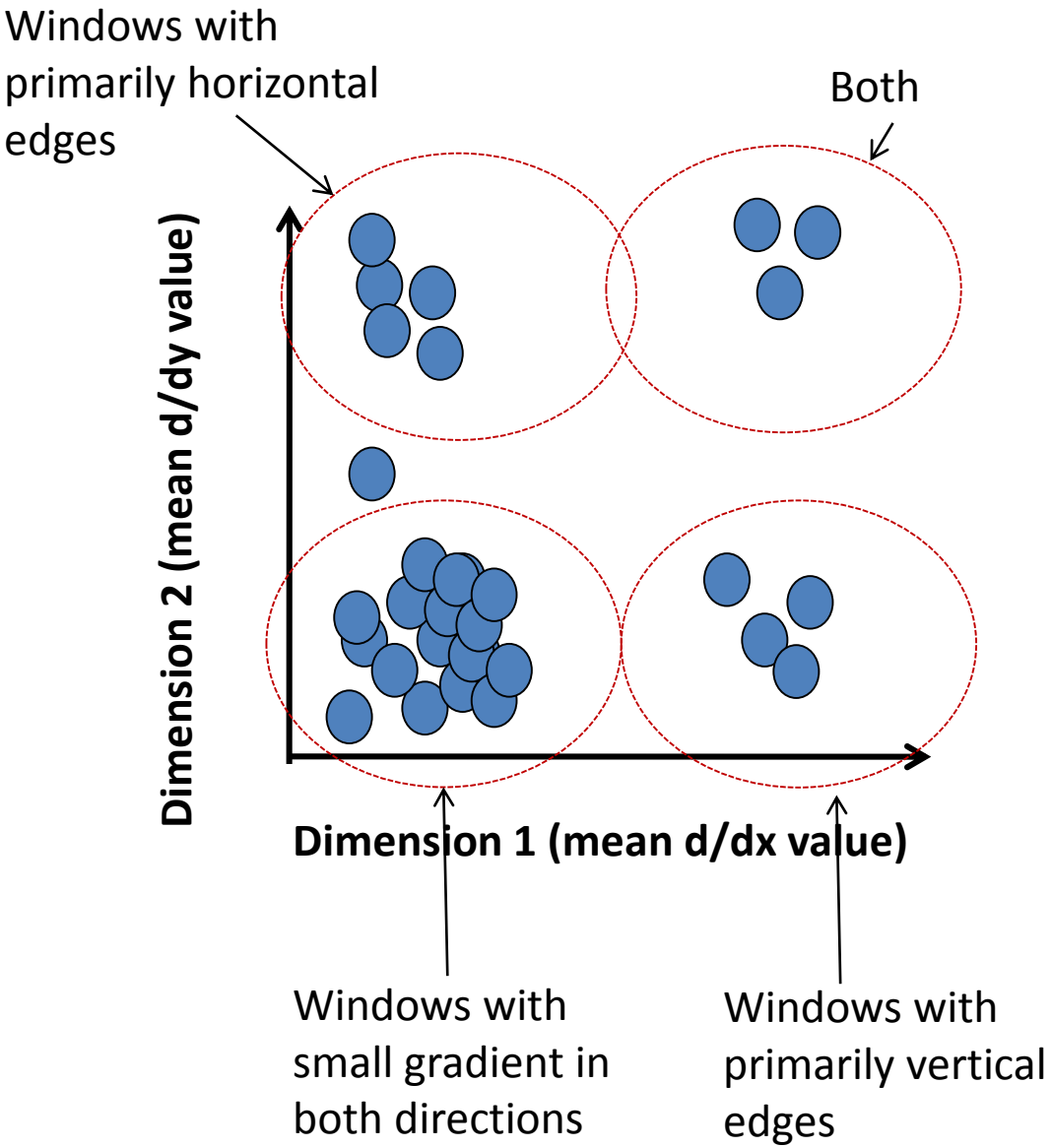


	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20

⋮

statistics to summarize patterns in small windows

Texture representation: example

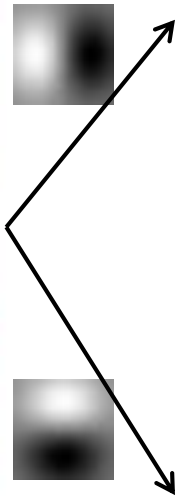


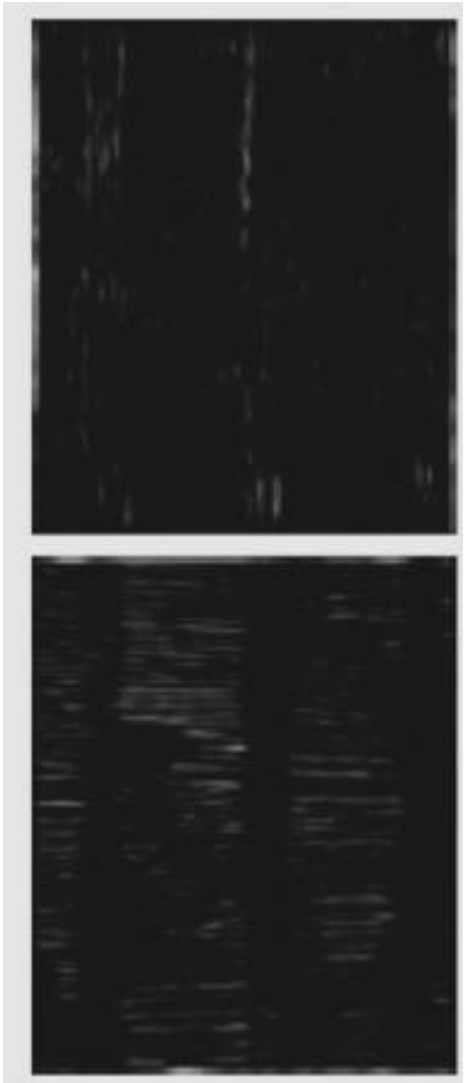
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20
	⋮	

statistics to summarize patterns in small windows

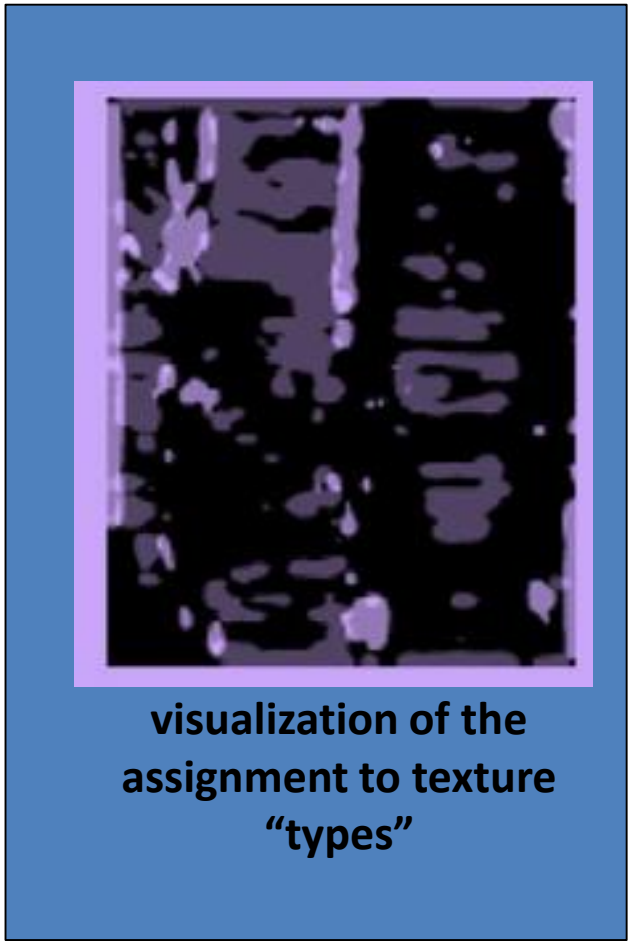
Texture representation: example

original image

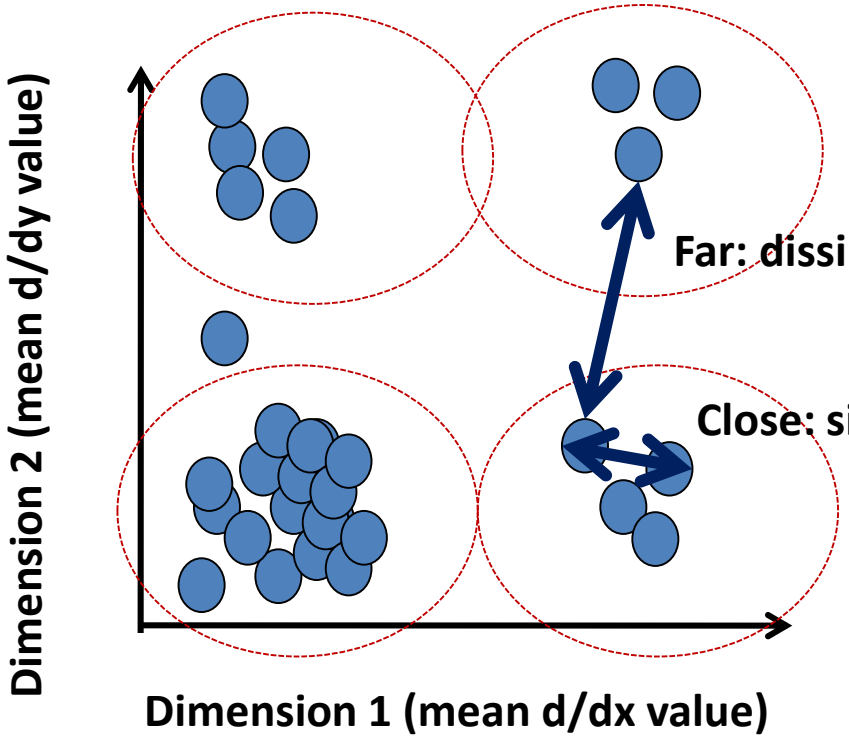




derivative filter responses, squared



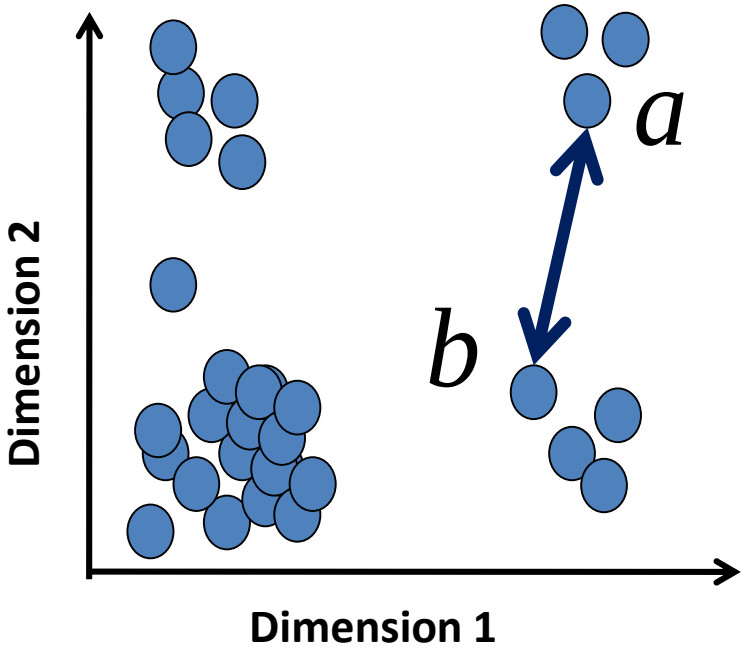
Texture representation: example



	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20
	⋮	

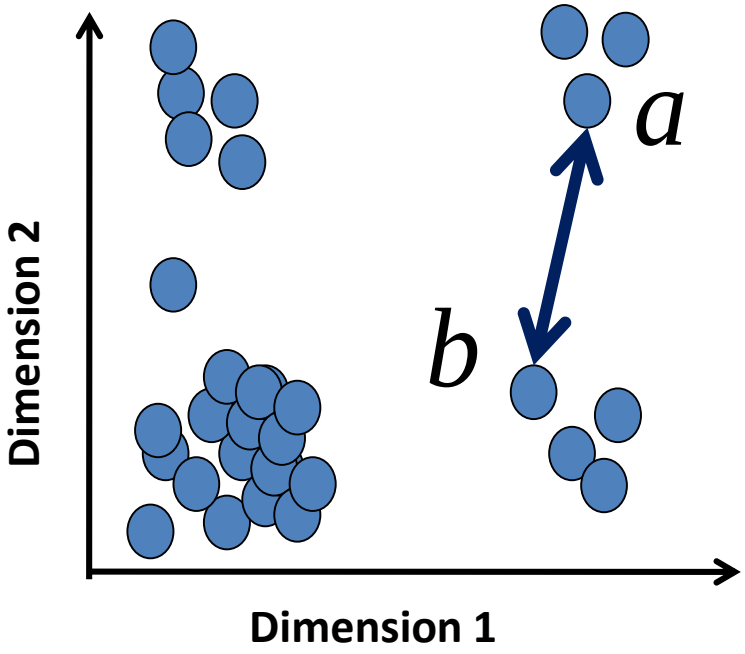
statistics to summarize patterns in small windows

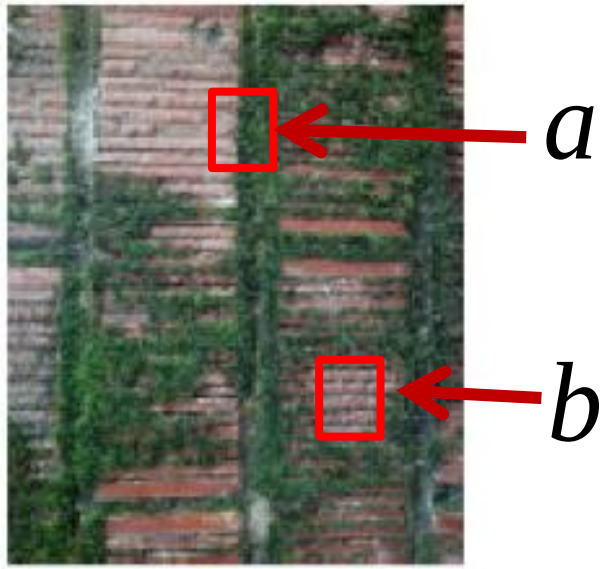
Computing distances using texture



$$D(a,b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2}$$

Texture representation: example





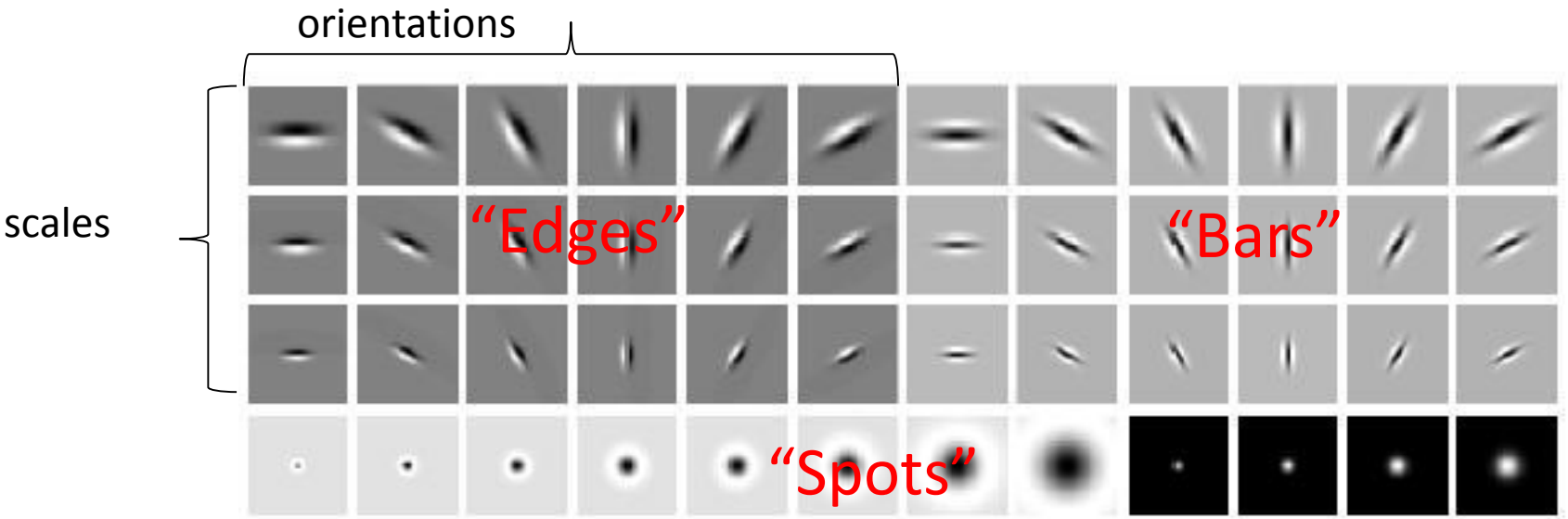
Distance reveals how dissimilar texture from window a is from texture in window b.

Filter banks

- Our previous example used two filters, and resulted in a 2-dimensional feature vector to describe texture in a window.
  - x and y derivatives revealed something about local structure.
- We can generalize to apply a collection of multiple (*d*) filters: a "filter bank"
- Then our feature vectors will be *d*-dimensional.
  - still can think of nearness, farness in feature space

Adapted from Kristen Grauman

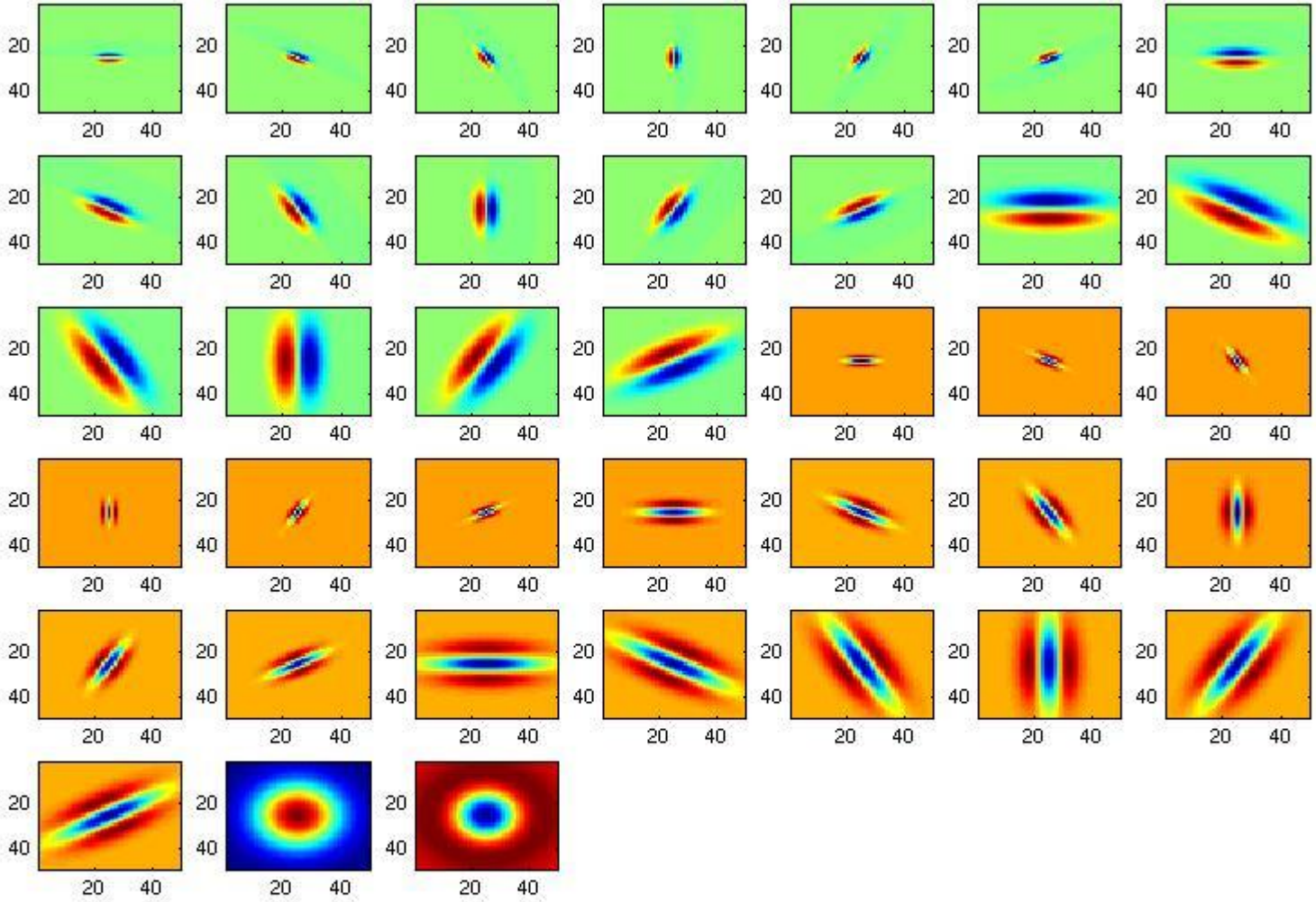
Filter banks



- What filters to put in the bank?
  - Typically we want a combination of scales and orientations, different types of patterns.

Matlab code available for these examples:
http://www.robots.ox.ac.uk/~vgg/research/texclass/filters.html

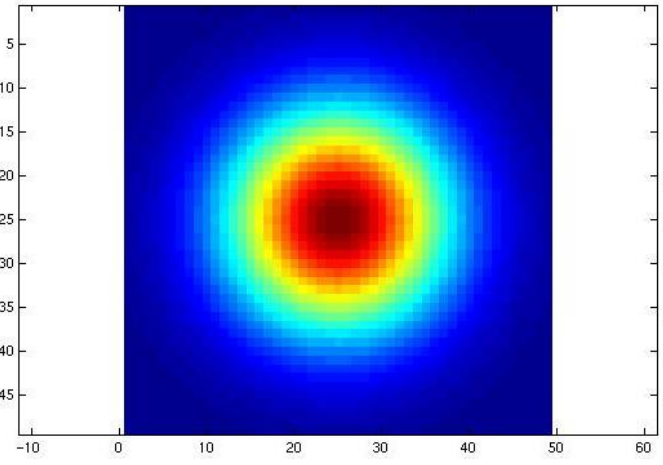
Filter bank

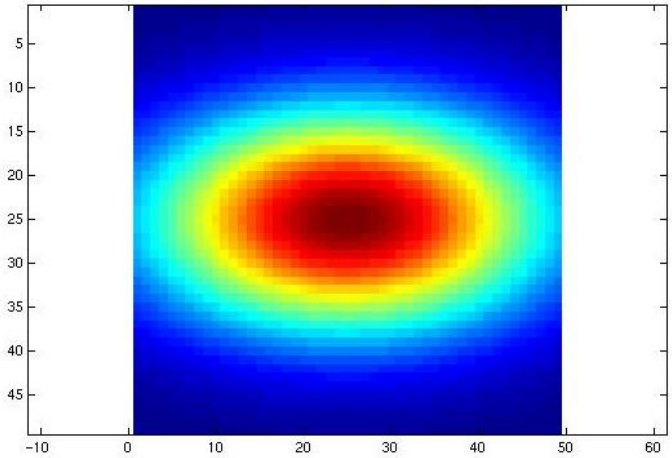


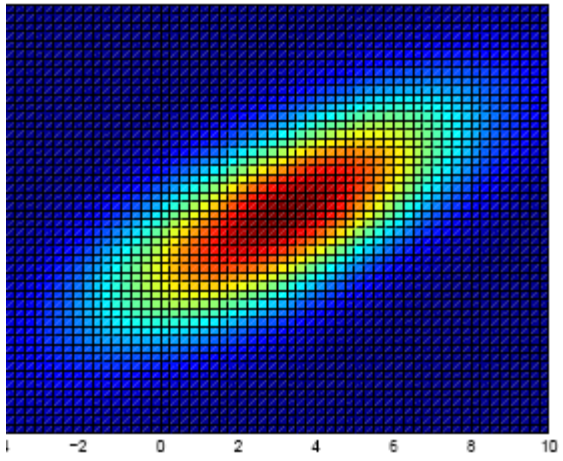
Kristen Grauman

Multivariate Gaussian

$$p(x;\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right).$$





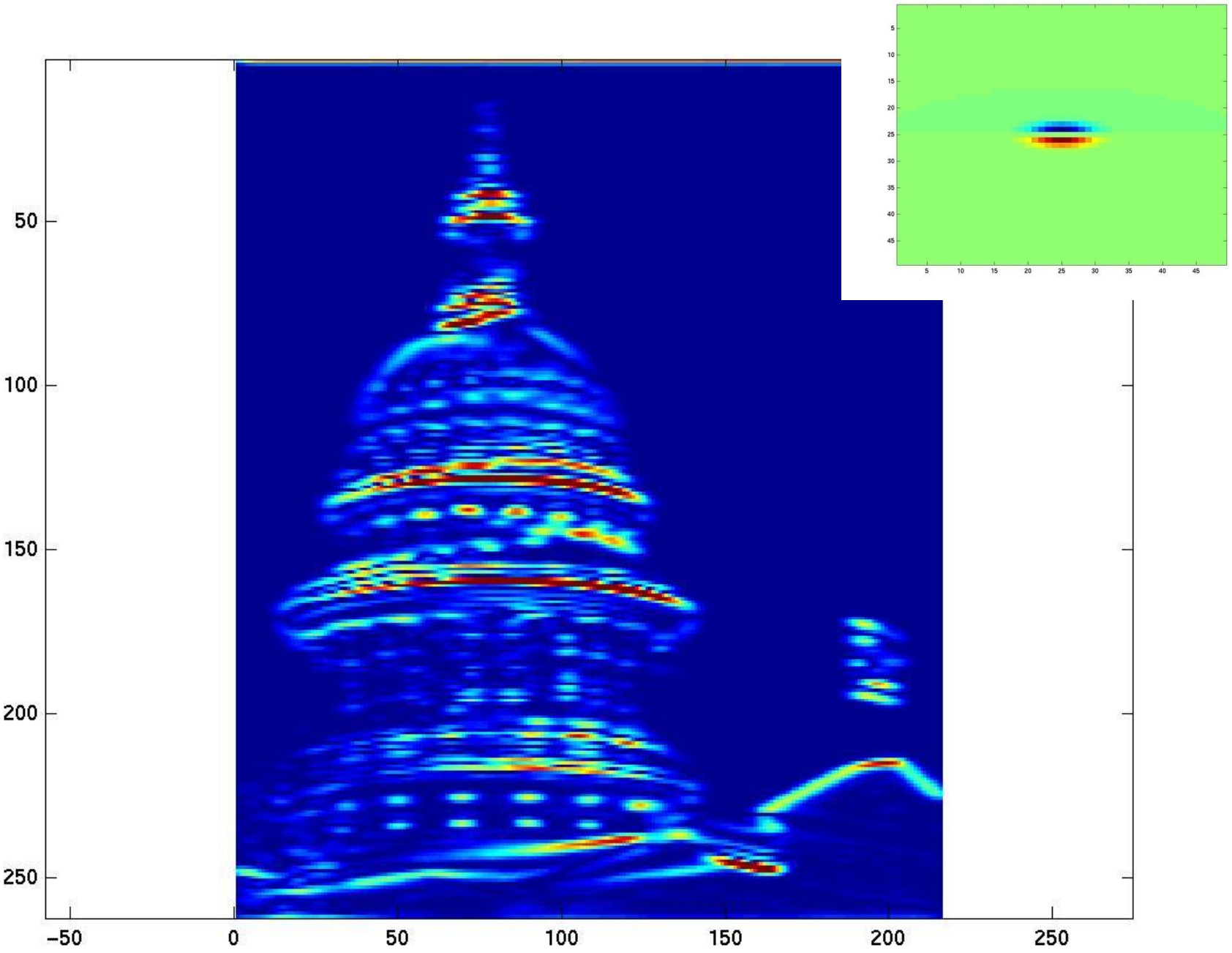


$$\Sigma = \begin{bmatrix} 9 & 0 \\ 0 & 9 \end{bmatrix}$$

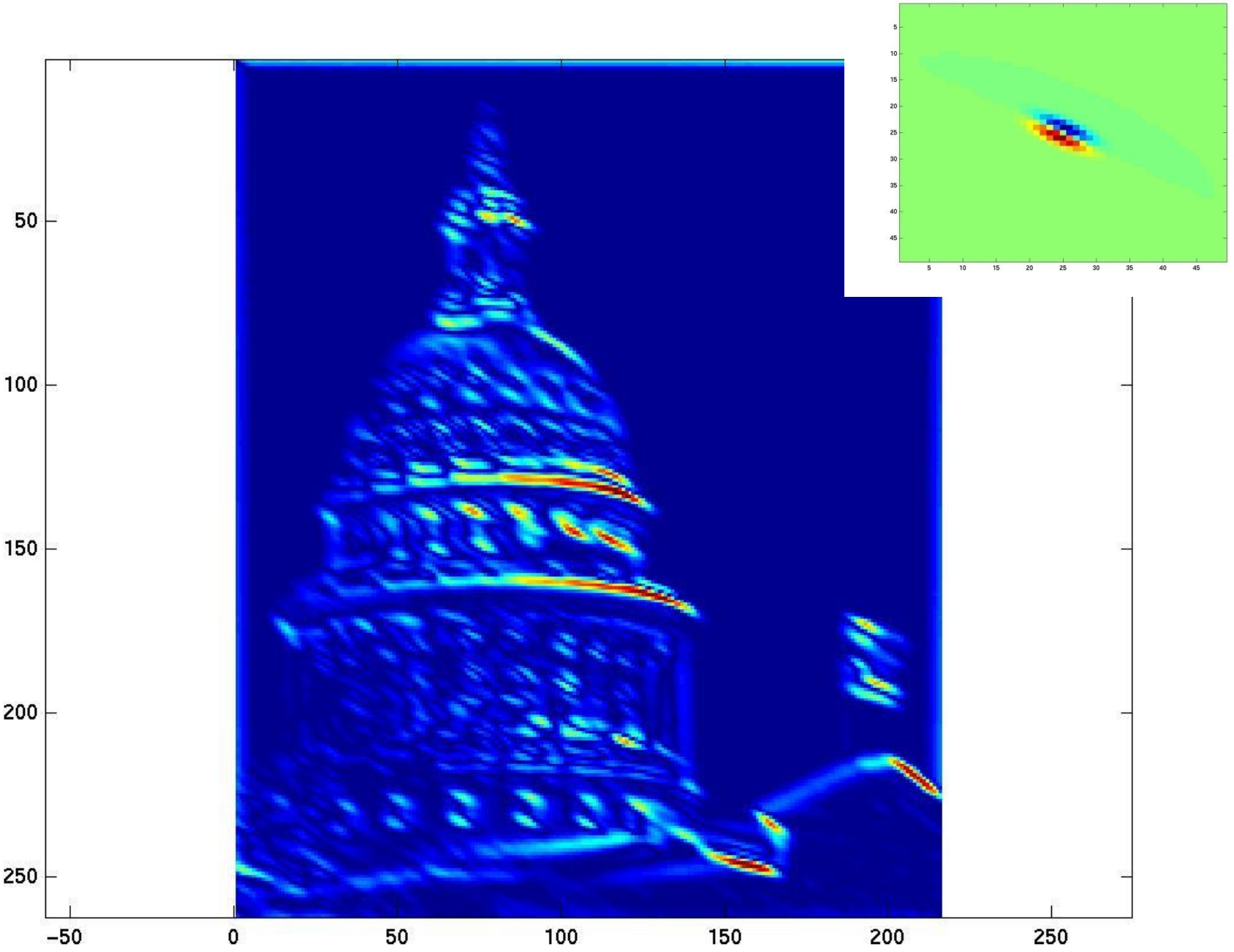
$$\Sigma = \begin{bmatrix} 16 & 0 \\ 0 & 9 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 10 & 5 \\ 5 & 5 \end{bmatrix}$$

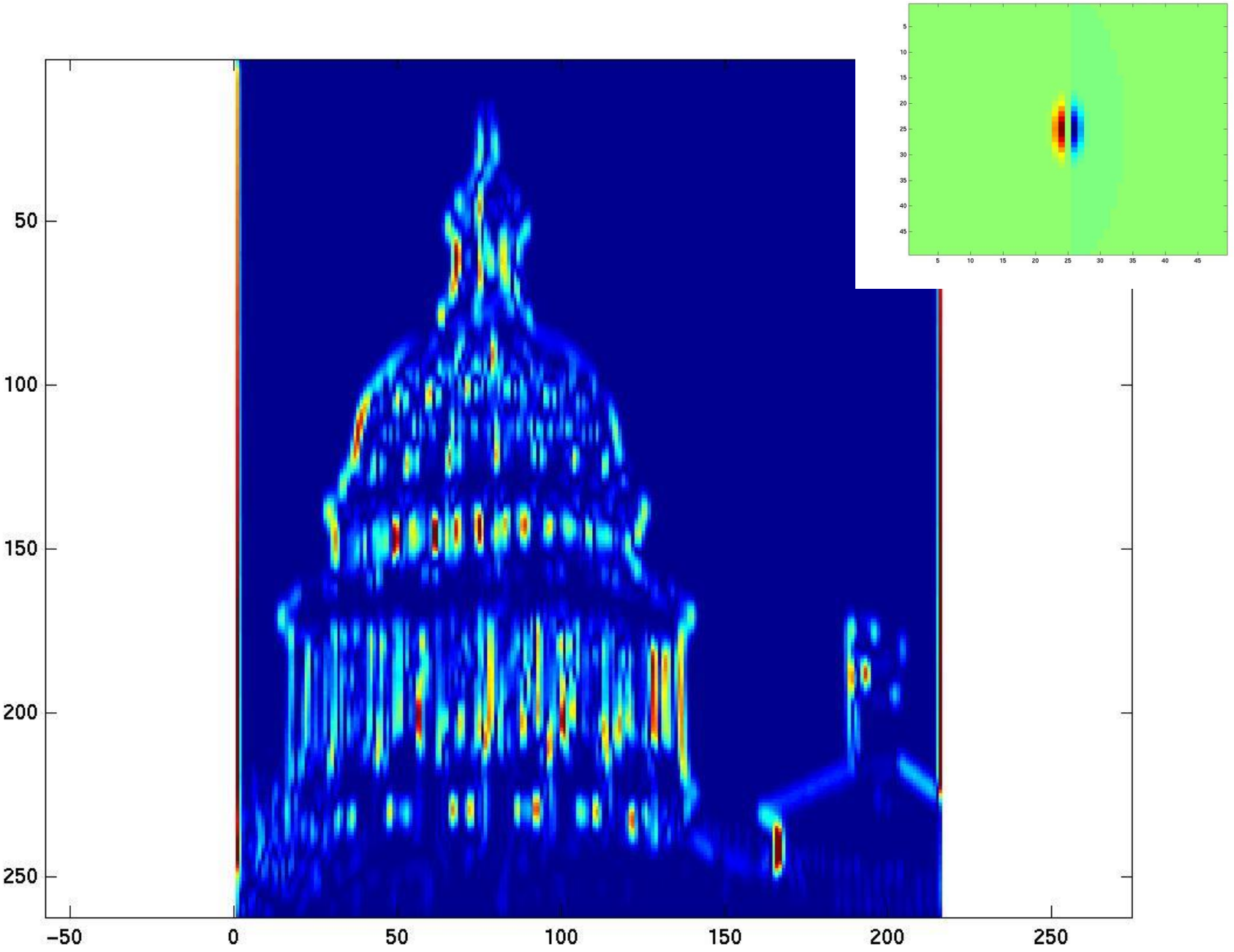
Image from http://www.texasexplorer.com/austincap2.jpg



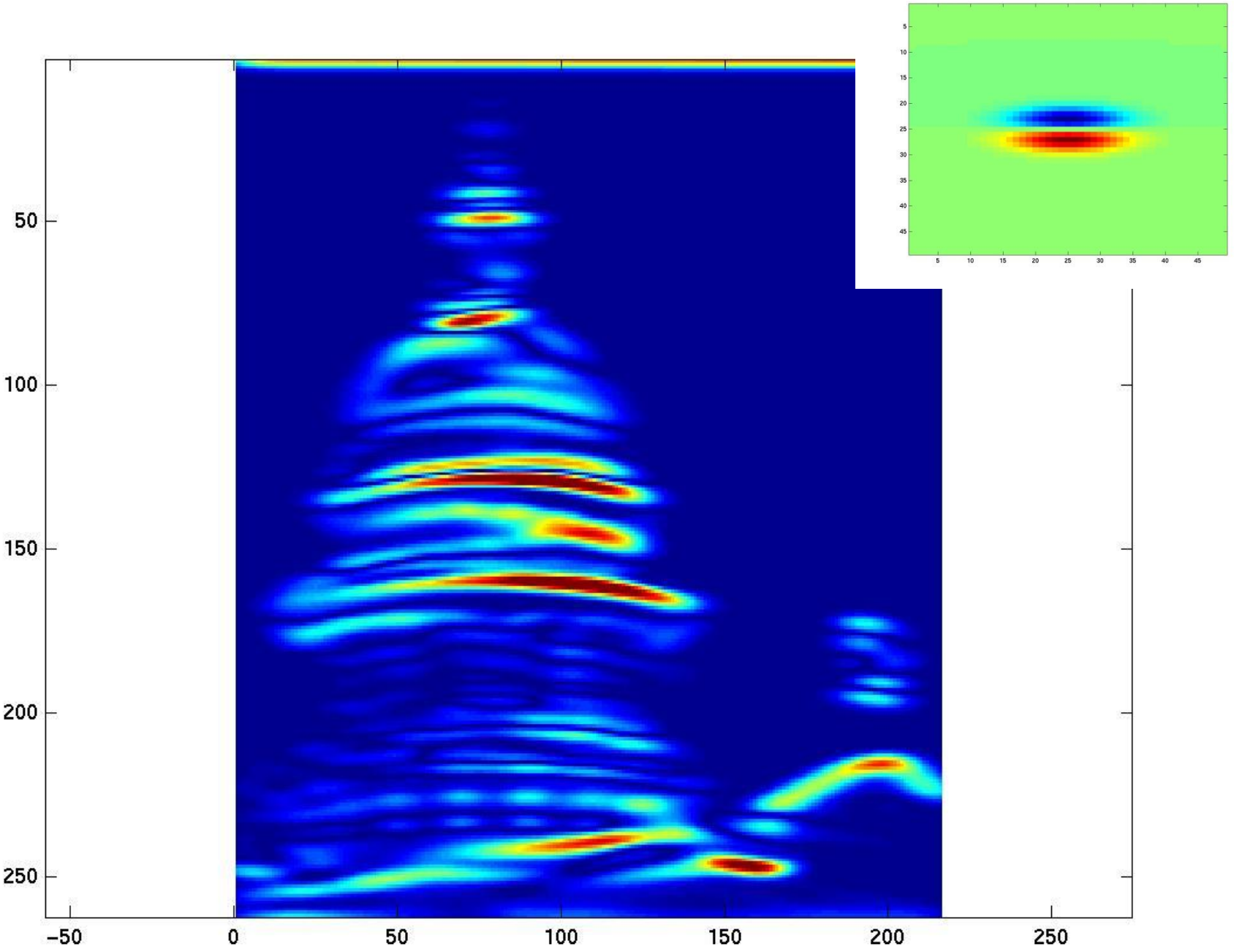
Kristen Grauman



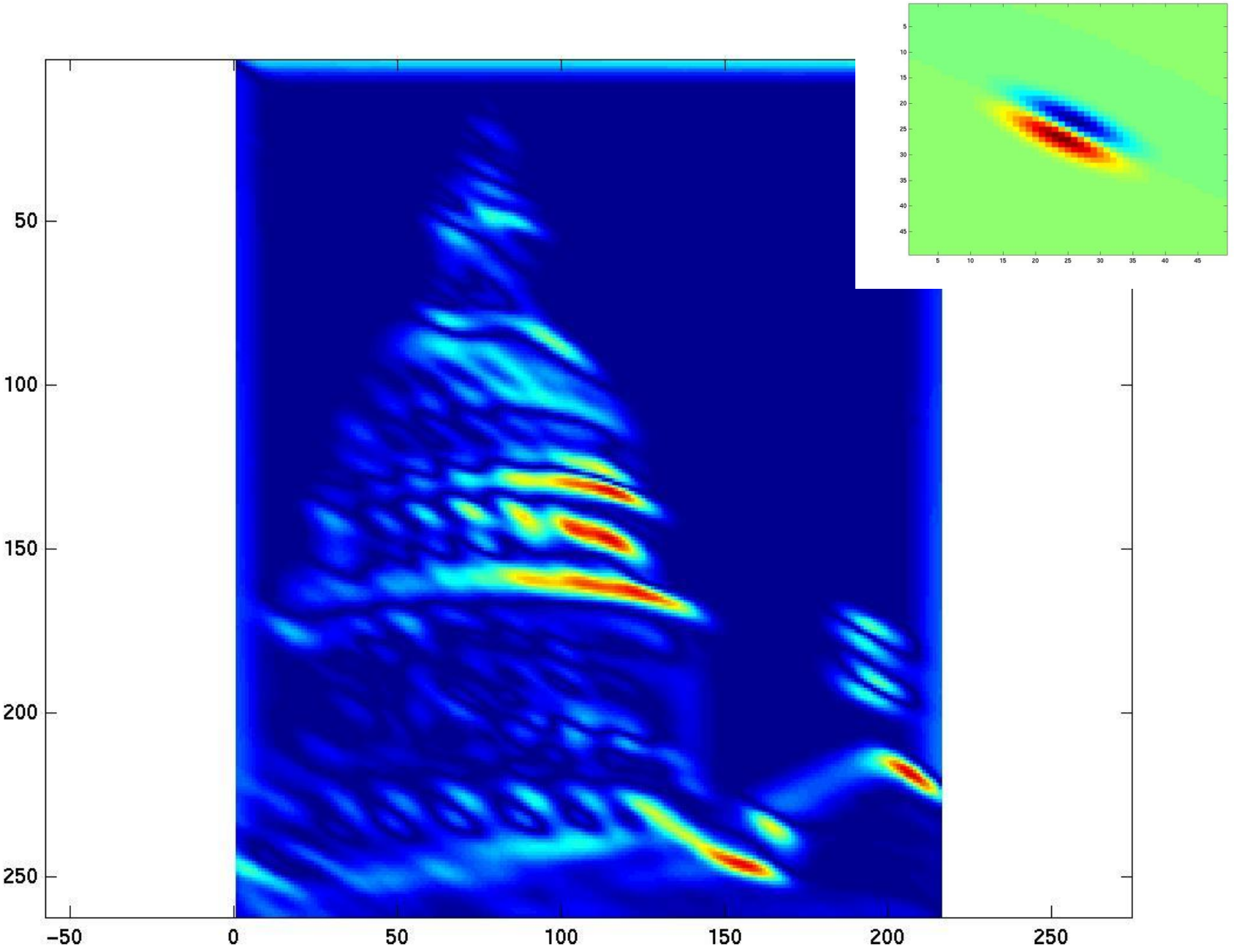
Kristen Grauman



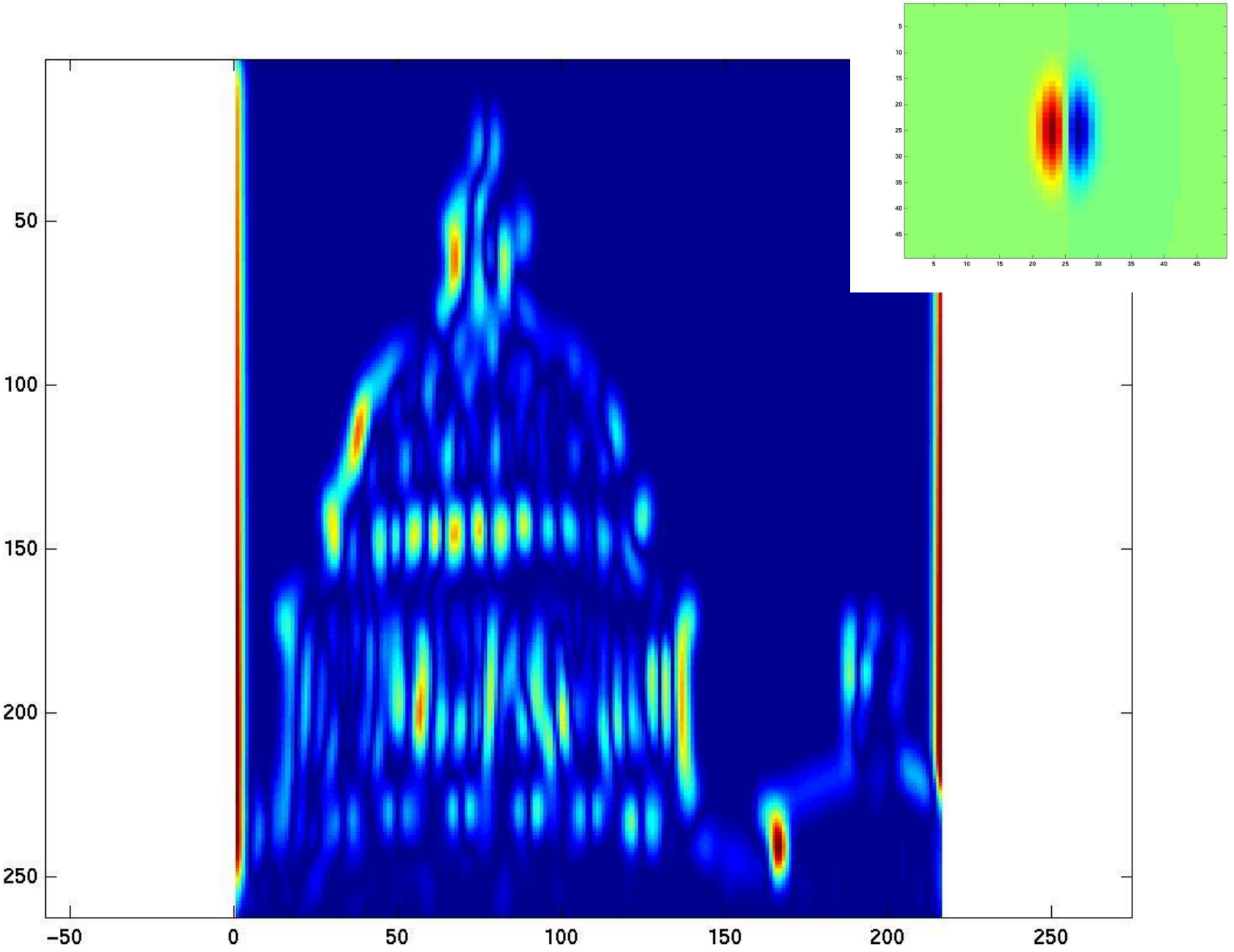
Kristen Grauman



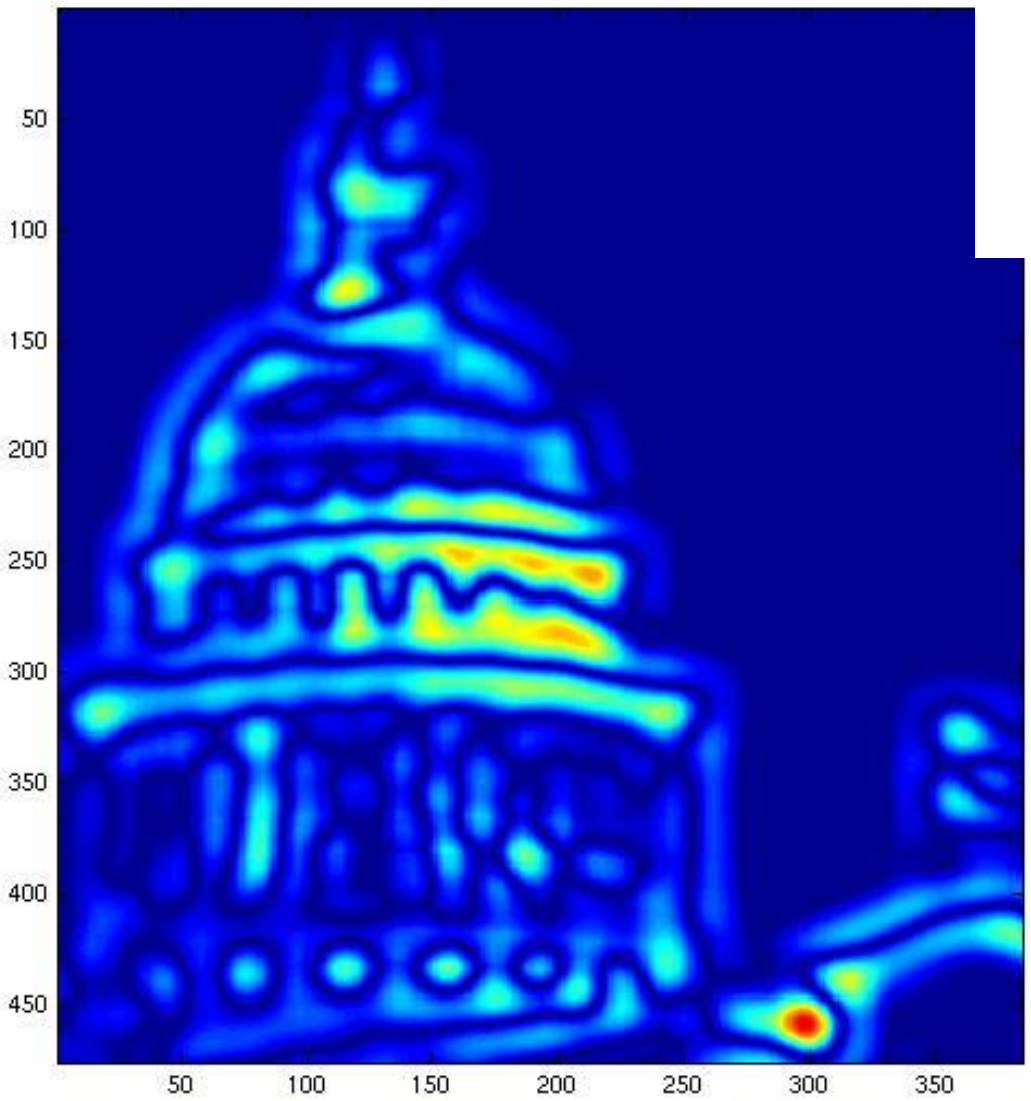
Kristen Grauman

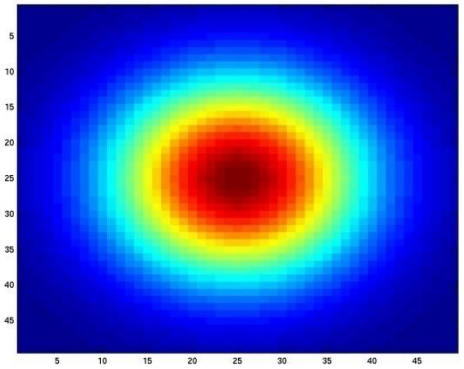


Kristen Grauman



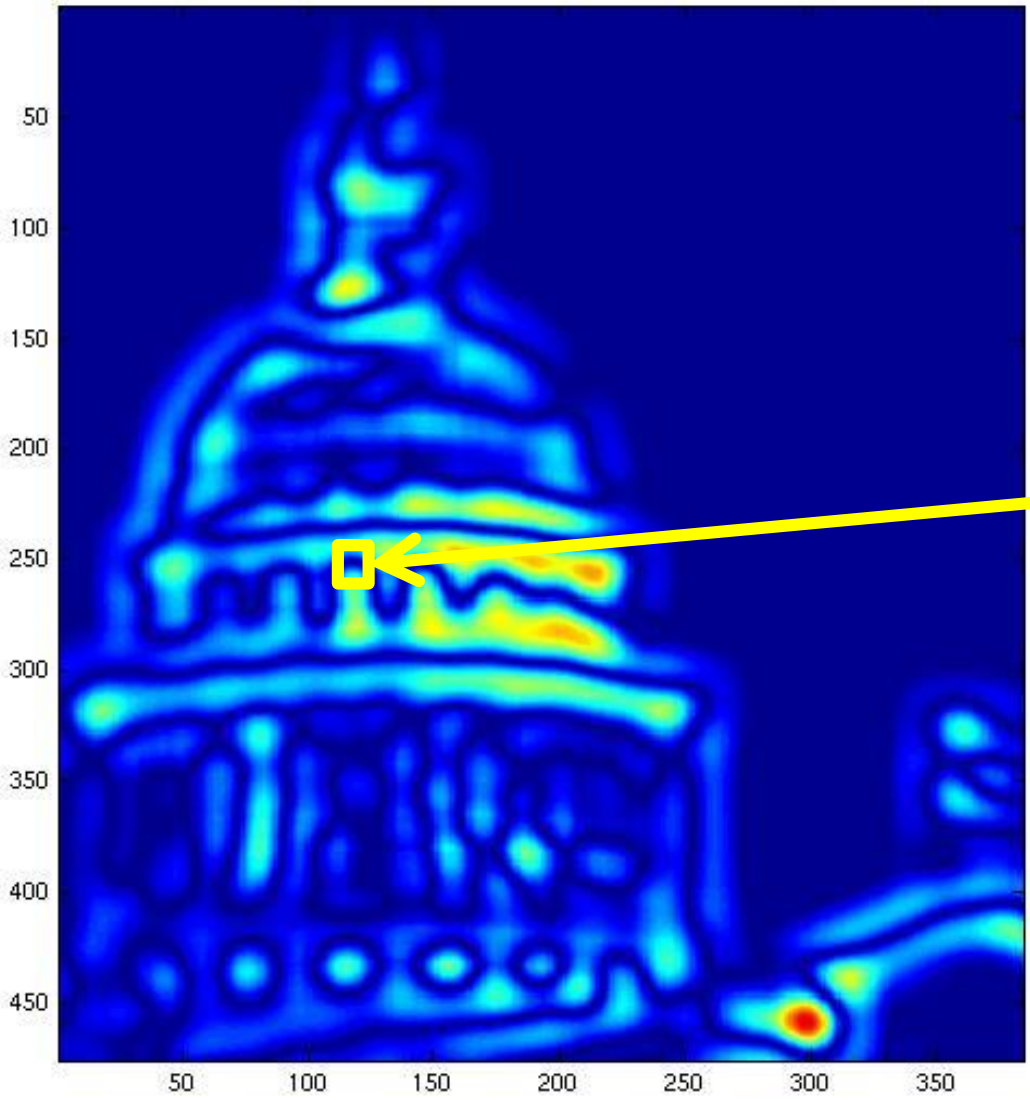
Kristen Grauman

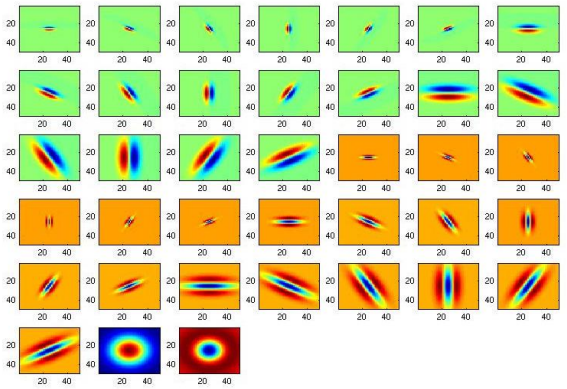




Kristen Grauman

Vectors of texture responses





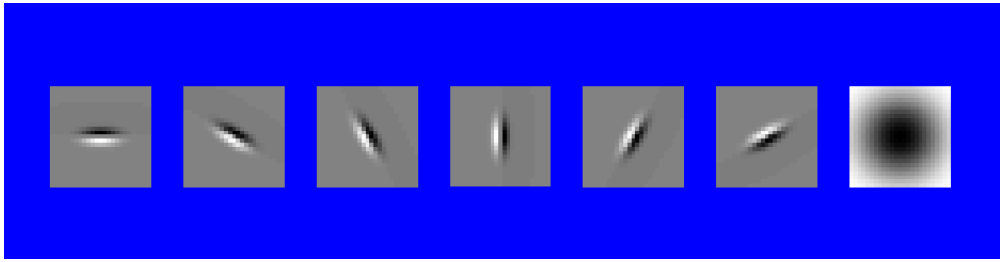
[r1, r2, …, r38]

We can form a "feature vector" from the list of responses at each pixel; gives us a representation of the pixel, image.

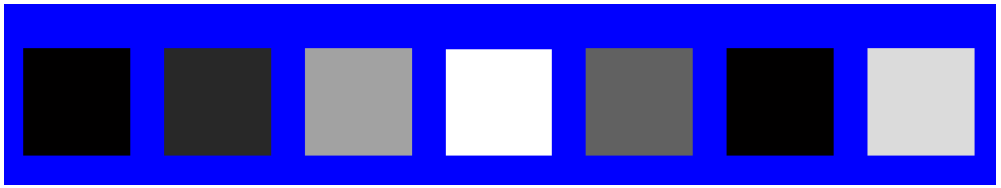
Adapted from Kristen Grauman

You try: Can you match the texture to the response?

Filters

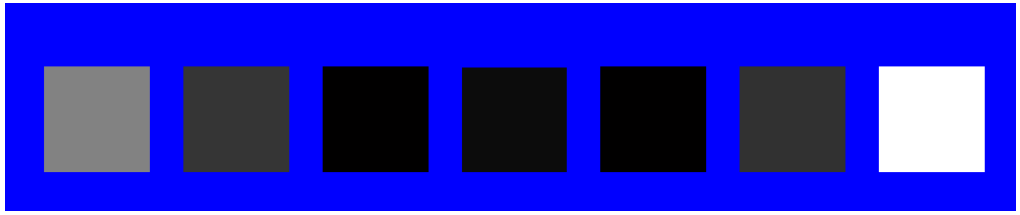


1



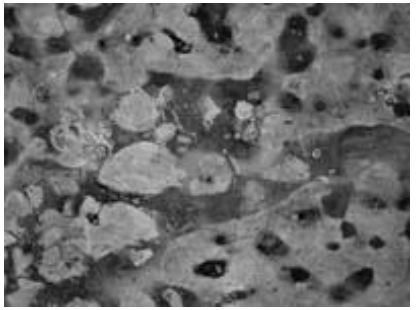
2

3



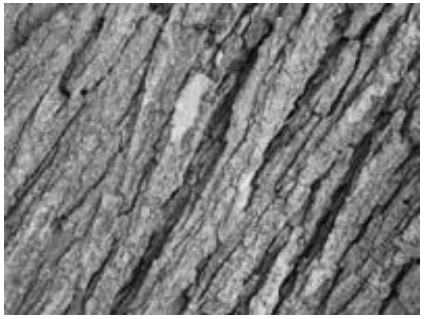
Mean abs responses

A



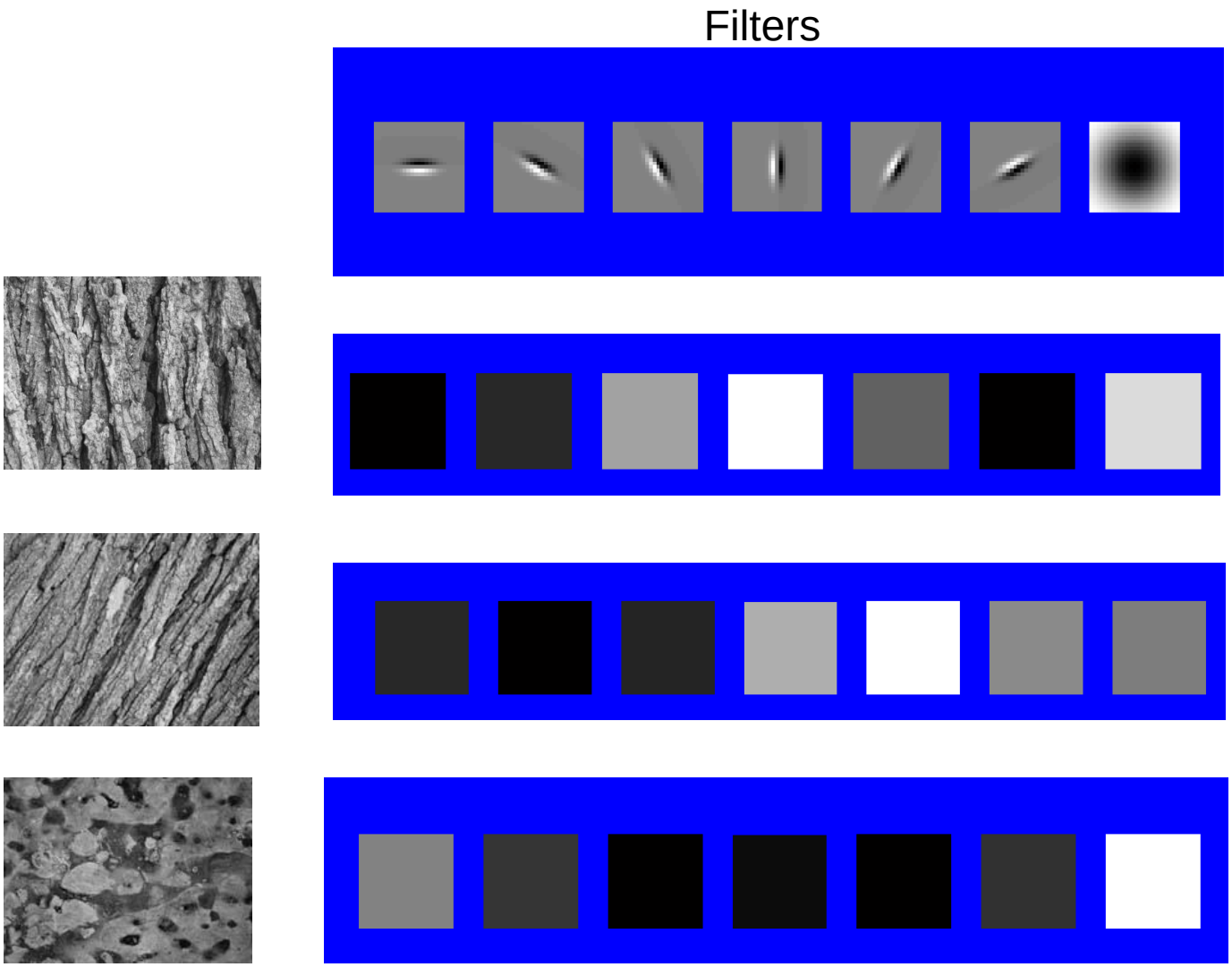
B

C



Derek Hoiem

Representing texture by mean abs response



Vectors of texture responses

- The frequency of image patch responses in an image tells us something about the image
- If my patches tend to respond strongly to blobs, and another image's patches respond strongly to vertical edges, then that image and I are probably of different materials
- How to capture?
- One idea: Concatenate [r1, …, r38] from the previous slide, for all image pixels
- Problems?

Means or histograms of feature responses

1) Instead of concatenating, compute the mean across all image pixels to each of the filters
2) Instead of concatenating, count:
   - Assume all responses between 0 and 1
   - How many times within the image did I see an r1 response between 0 and 0.1?
   - … between 0.1 and 0.2?
   - How many times did I see r2 response between 0 and 0.1?
   - …
   - Concatenate these counts into a *histogram*

- Tradeoffs?

Classifying materials, “stuff”

Figure by Varma & Zisserman

1-minute break

Plan for today

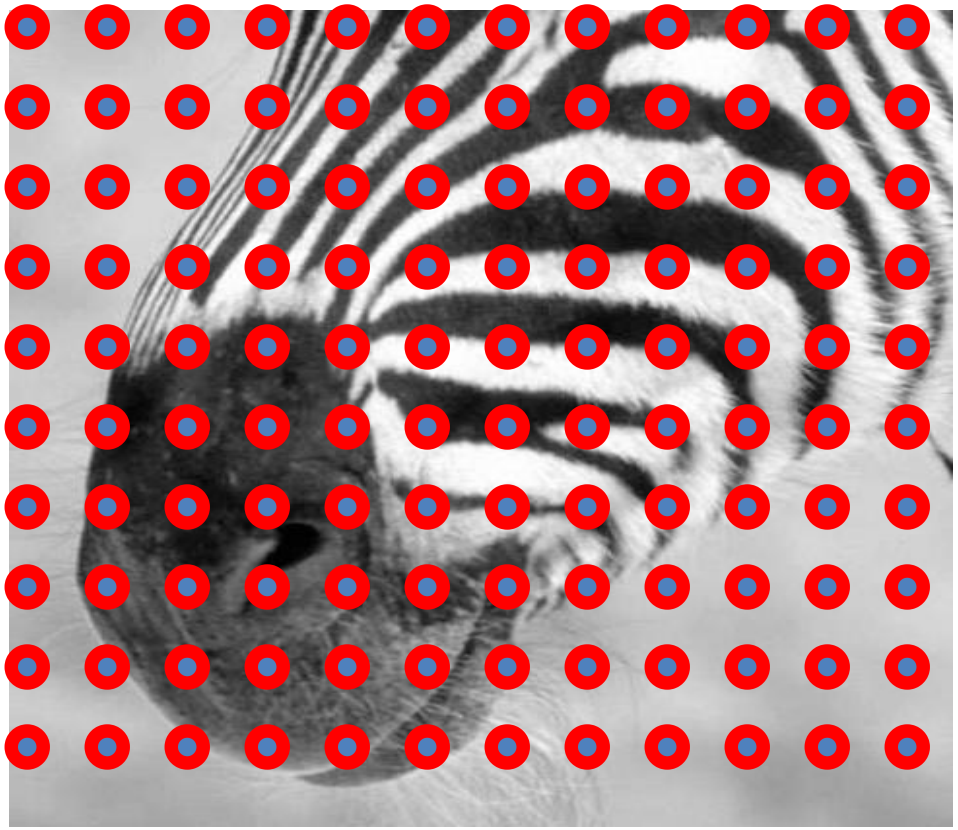
- Filtering
  - Representing texture
  - Application to subsampling
  - Image pyramids
- Detecting interesting content (start)

Sampling

Why does a lower resolution image still make sense to us? What do we lose?

Image: http://www.flickr.com/photos/igorms/136916757/

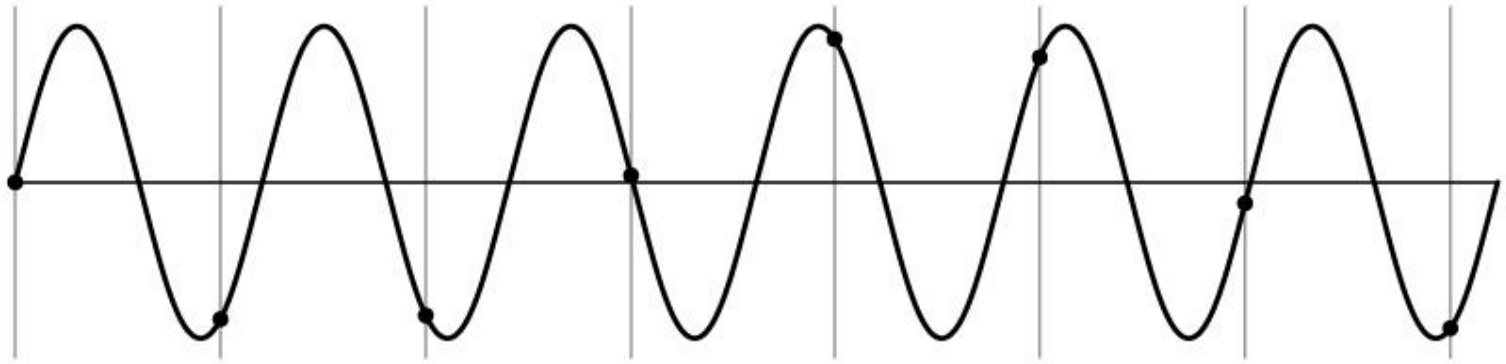
Subsampling by a factor of 2



Throw away every other row and column to create a 1/2 size image

Aliasing problem

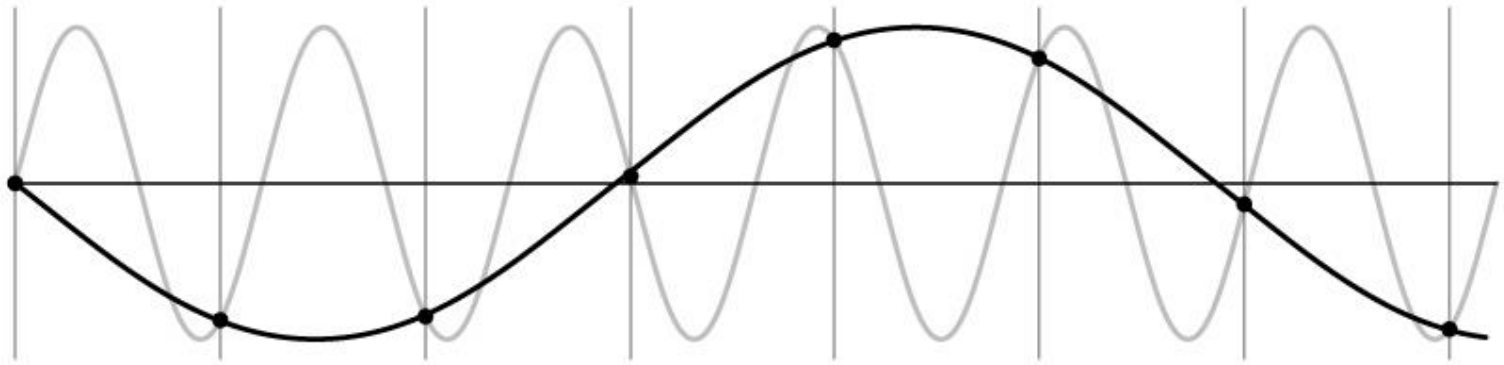
- 1D example (sinewave):



Source: S. Marschner

Aliasing problem

- 1D example (sinewave):



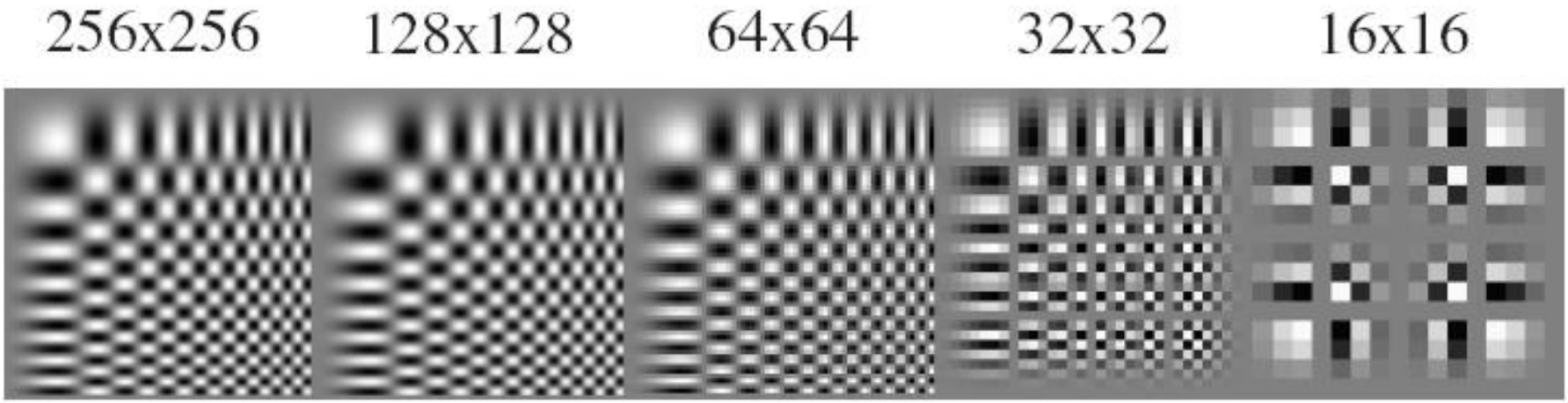
Source: S. Marschner

Aliasing problem

- Sub-sampling may be dangerous….
- Characteristic errors may appear:
  - “Wagon wheels rolling the wrong way in movies”
  - “Striped shirts look funny on color television”

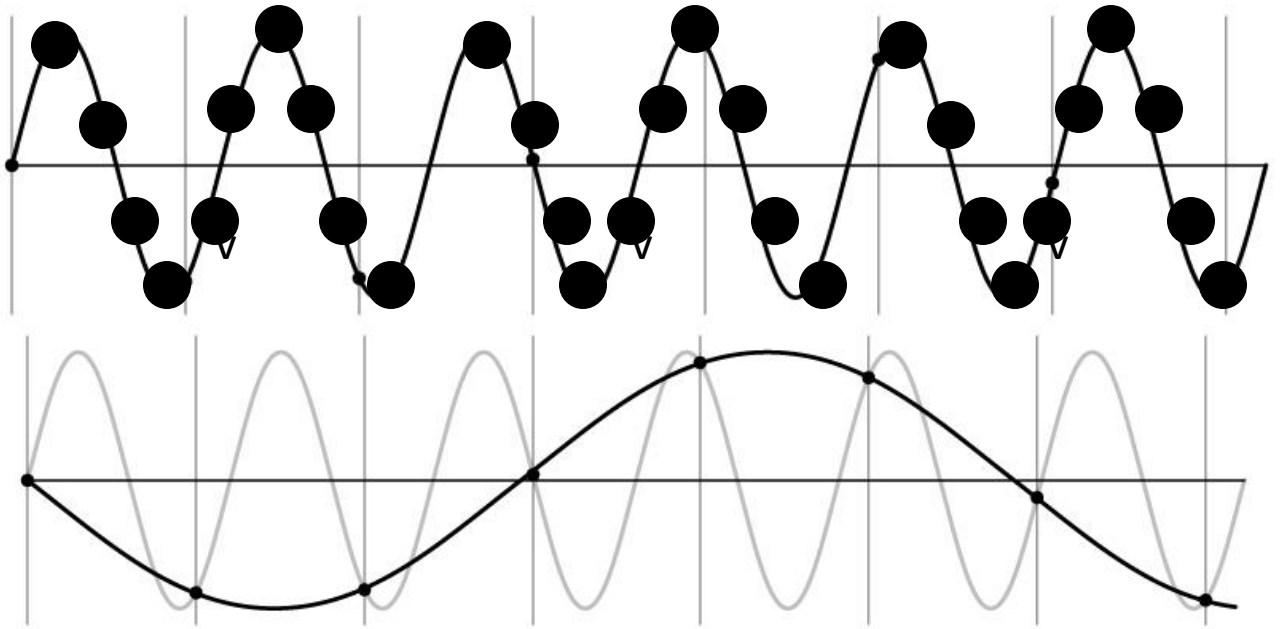
Source: D. Forsyth

Sampling and aliasing



Nyquist-Shannon Sampling Theorem

- When sampling a signal at discrete intervals, the sampling frequency must be $\geq 2 \times f_{max}$
- f_{max} = max frequency of the input signal
- This will allows to reconstruct the original perfectly from the sampled version



Anti-aliasing

Solutions:

- Sample more often

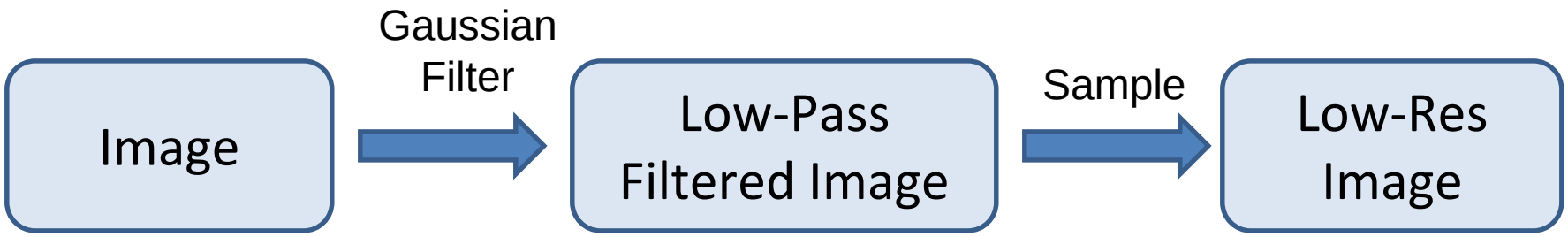
- Get rid of all frequencies that are greater than half the new sampling frequency
  - Will lose information
  - But it's better than aliasing
  - Apply a smoothing filter

Algorithm for downsampling by factor of 2

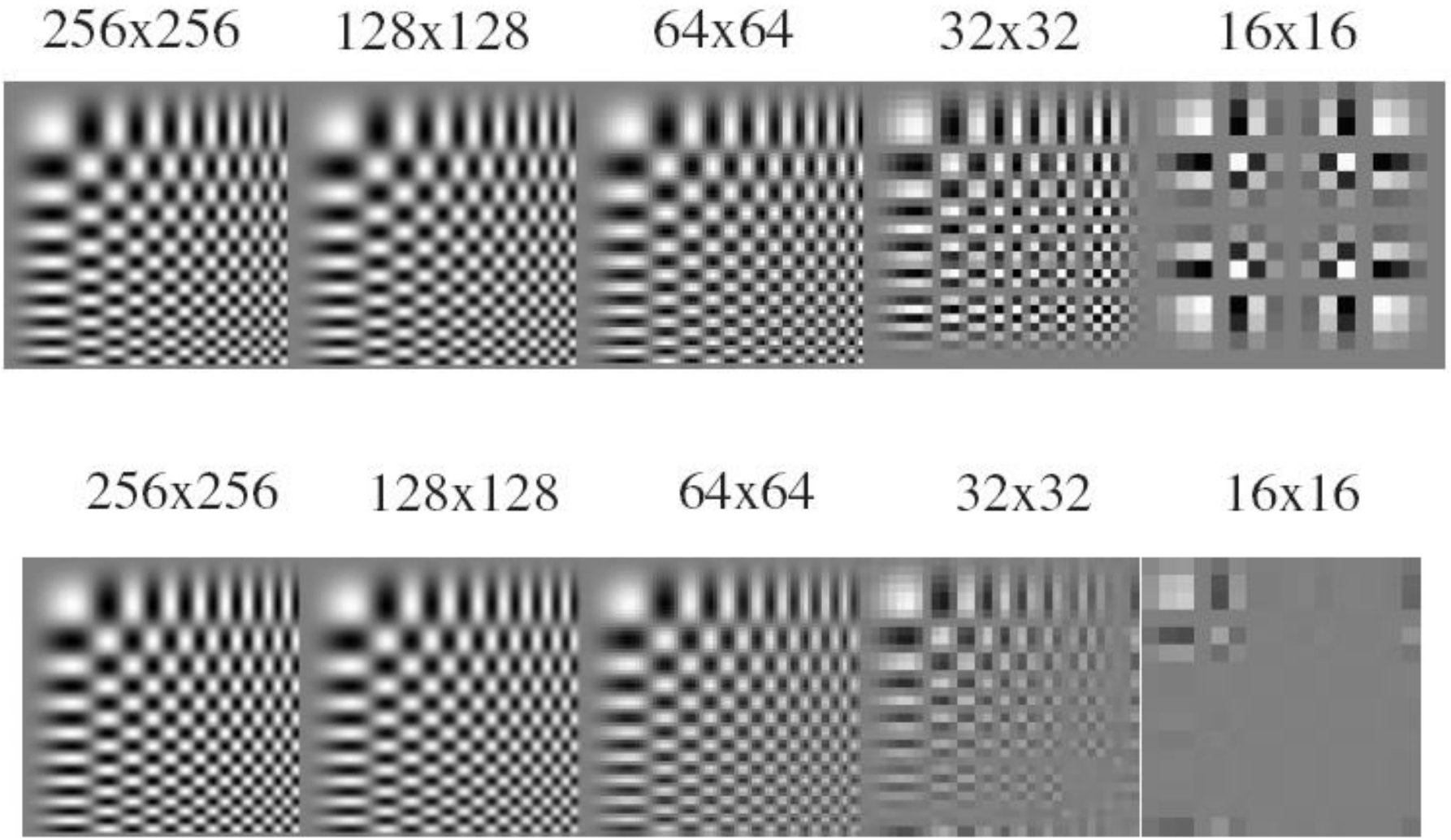
1. Start with image(h, w)
2. Apply low-pass filter

   im_blur = imfilter(image, fspecial('gaussian', 7, 1))
3. Sample every other pixel

   im_small = im_blur(1:2:end, 1:2:end);

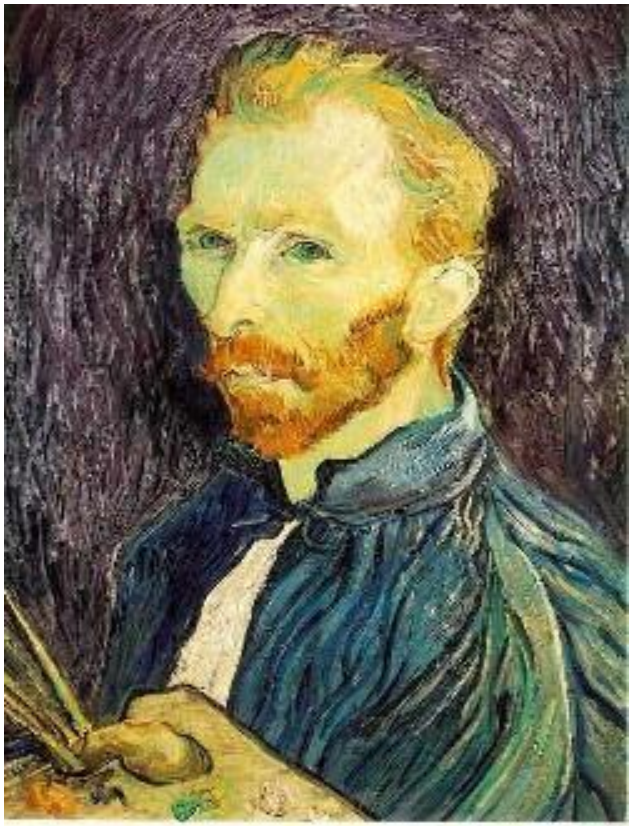


Anti-aliasing



Forsyth and Ponce 2002

Subsampling without pre-filtering



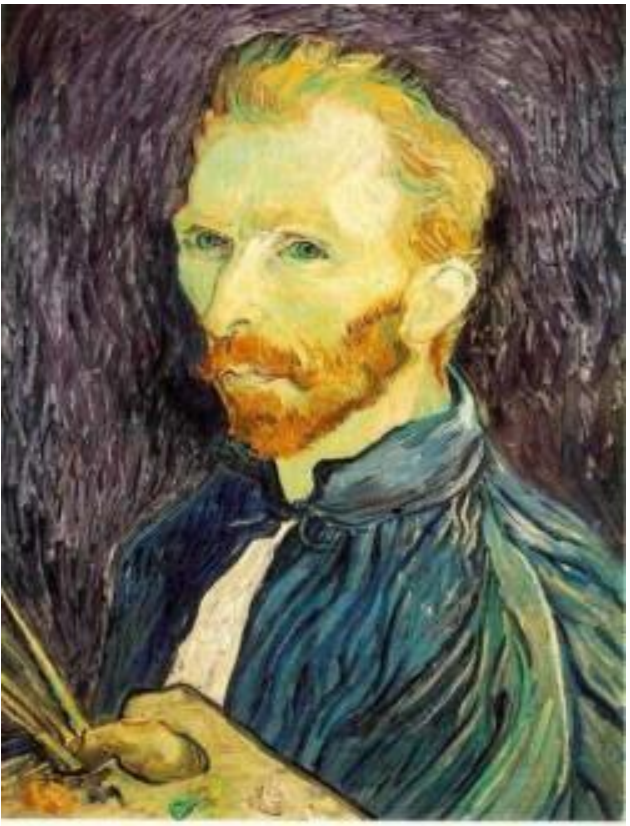
1/2

1/4 (2x zoom)

1/8 (4x zoom)

Slide by Steve Seitz

Subsampling with Gaussian pre-filtering



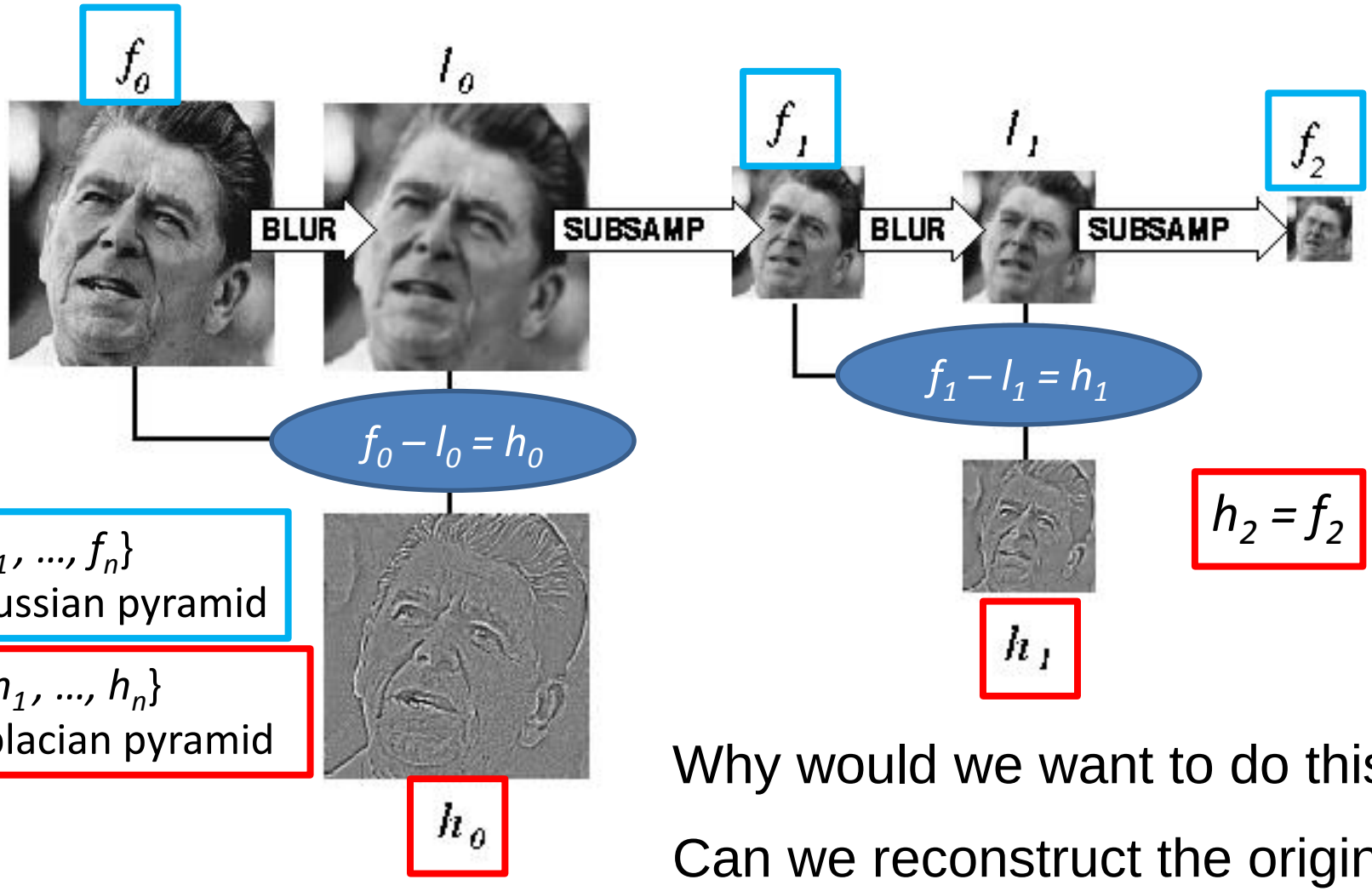
Gaussian 1/2

G 1/4

G 1/8

Slide by Steve Seitz

Subsampling away…



$\{f_0, f_1, \ldots, f_n\}$
= Gaussian pyramid

$\{h_0, h_1, \ldots, h_n\}$
= Laplacian pyramid

Why would we want to do this?

Can we reconstruct the original from the Laplacian pyramid?

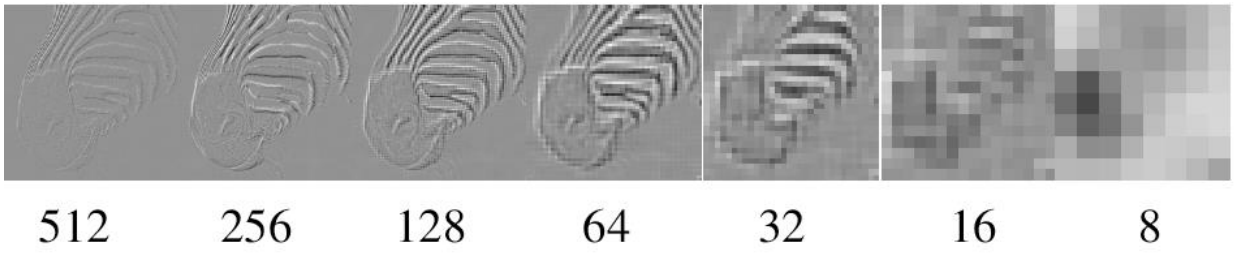
http://sepwww.stanford.edu/~morgan/texturematch/paper_html/node3.html

Gaussian pyramid

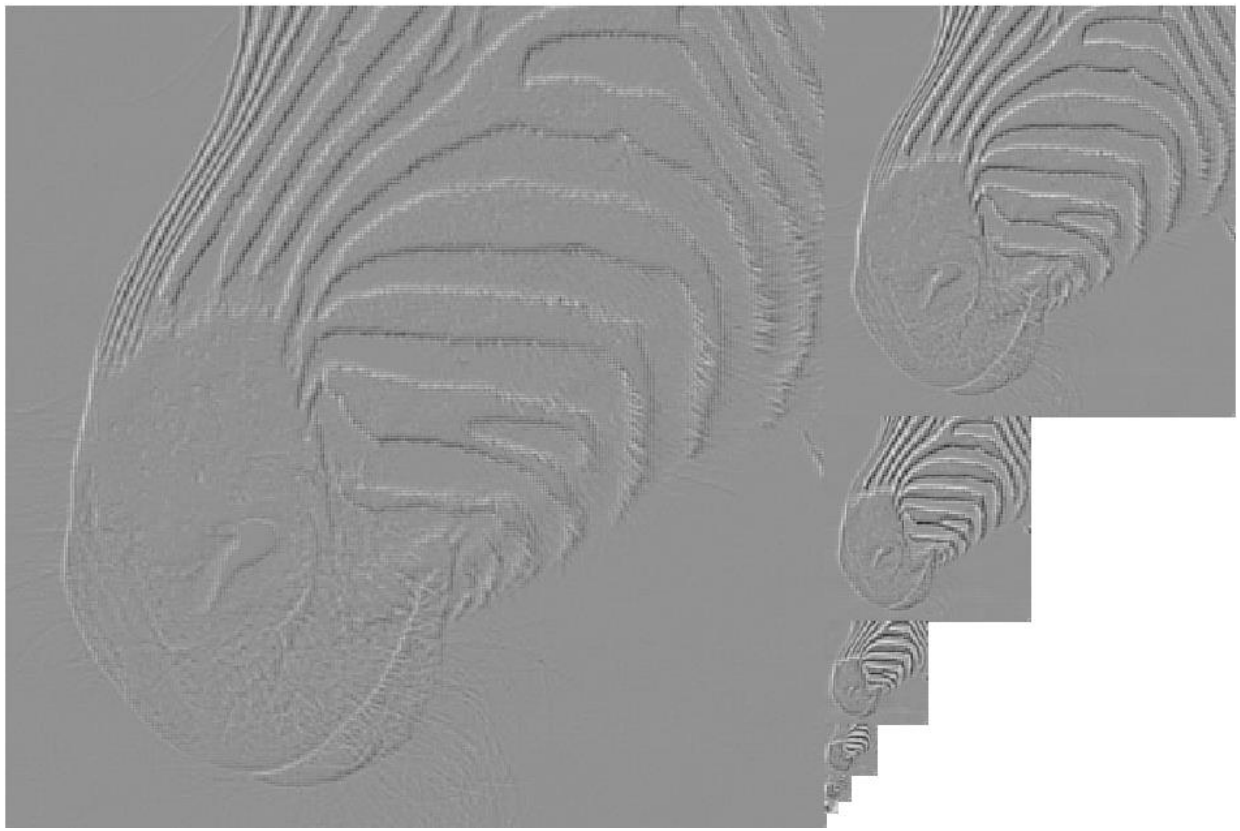
512 256 128 64 32 16 8

Source: Forsyth

Laplacian pyramid

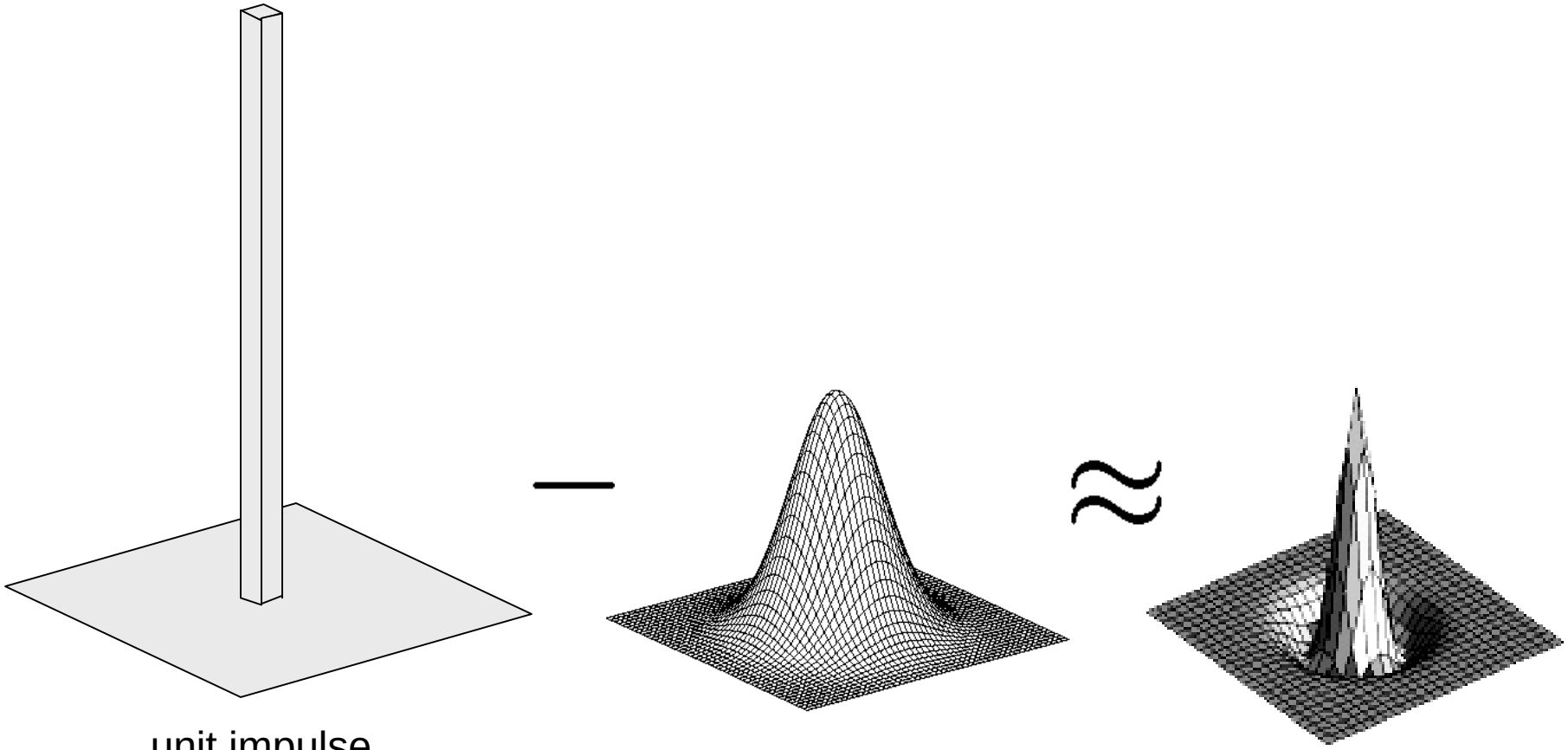


512 256 128 64 32 16 8



Source: Forsyth

Laplacian filter

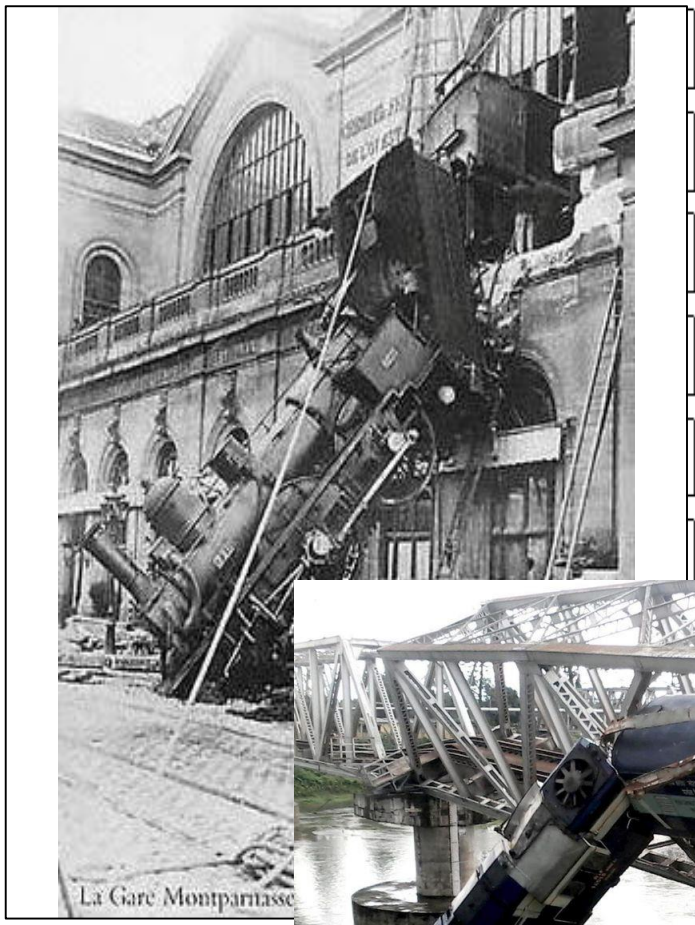


Source: Lazebnik

Plan for today

- Filtering
  - Representing texture
  - Application to subsampling
  - Image pyramids
- Detecting interesting content (start)

An image is a set of pixels…



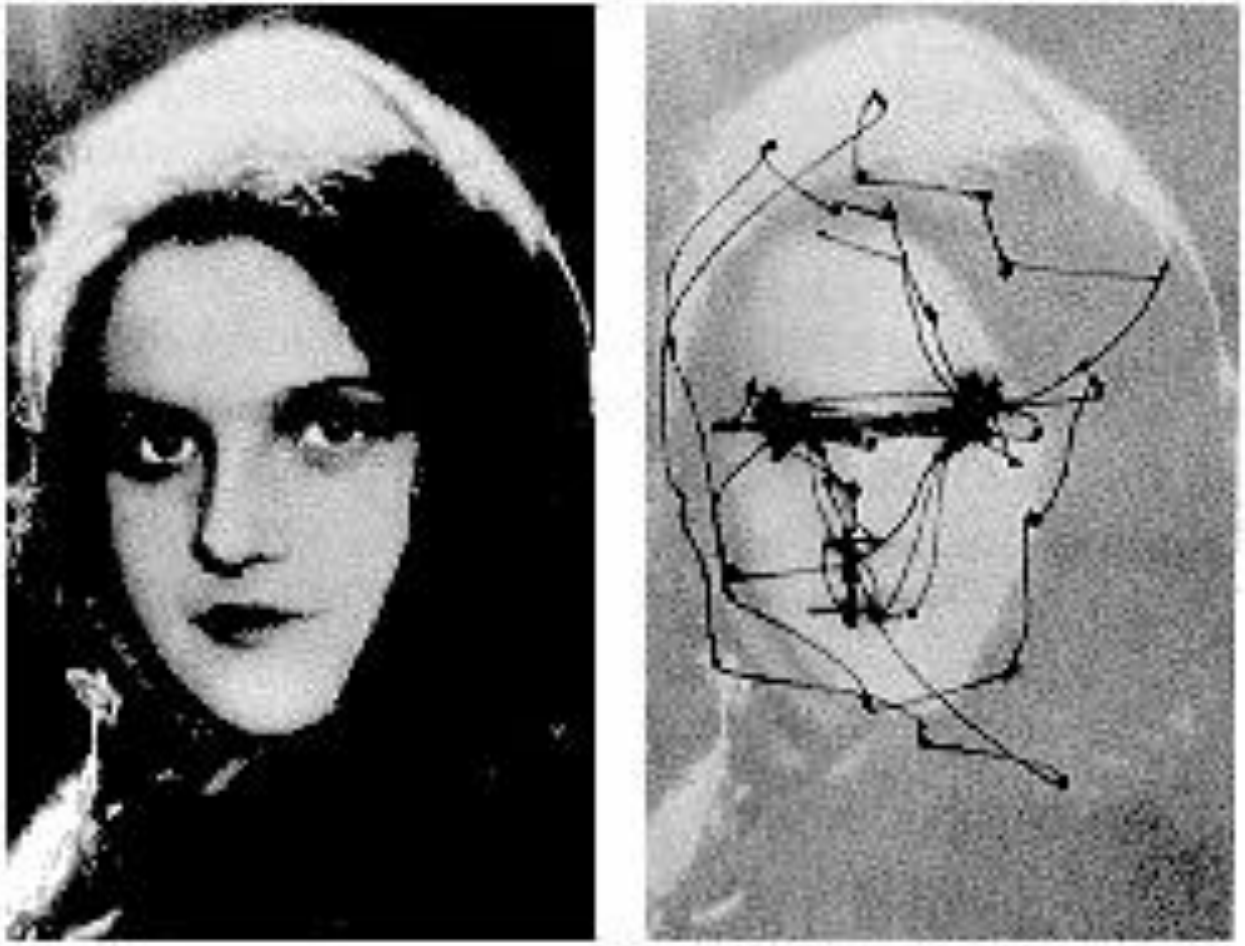
6	9	8
5	6	7
4	7	6
3	4	5
2	5	4
1	2	3

Adapted from S. Narasimhan

Problems with pixel representation

- Not invariant to small changes
  - Translation
  - Illumination
  - etc.
- Some parts of an image are more important than others
- What do we want to represent?

Human eye movements



Yarbus eye tracking

Choosing distinctive interest(ing) points

- If you wanted to meet a friend would you say
  a) “Let’s meet on campus.”
  b) “Let’s meet on Green street.”
  c) “Let’s meet at Green and Wright.”
  – Corner detection

- Or if you were in a secluded area:
  a) “Let’s meet in the Plains of Akbar.”
  b) “Let’s meet on the side of Mt. Doom.”
  c) “Let’s meet on top of Mt. Doom.”
  – Blob (valley/peak) detection

Choosing interest(ing) points

Where would you tell your friend to meet you?

Choosing interest(ing) points

Where would you tell your friend to meet you?

Interest points

- Suppose you have to click on some point, go away and come back after I deform the image, and click on the same points again.
  - Which points would you choose?