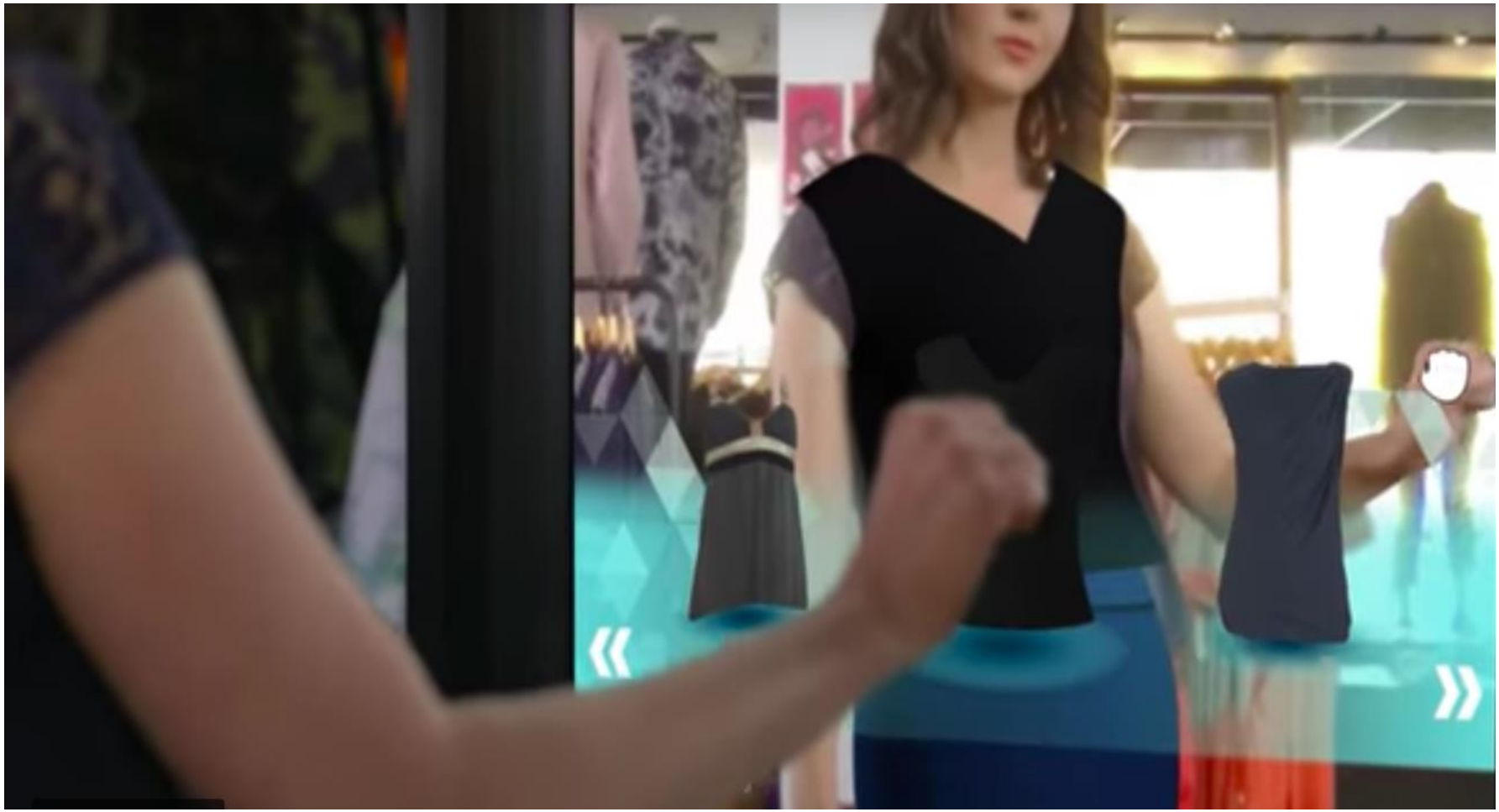
TEXTURE ANALYSIS

Class 7: Artificial Vision

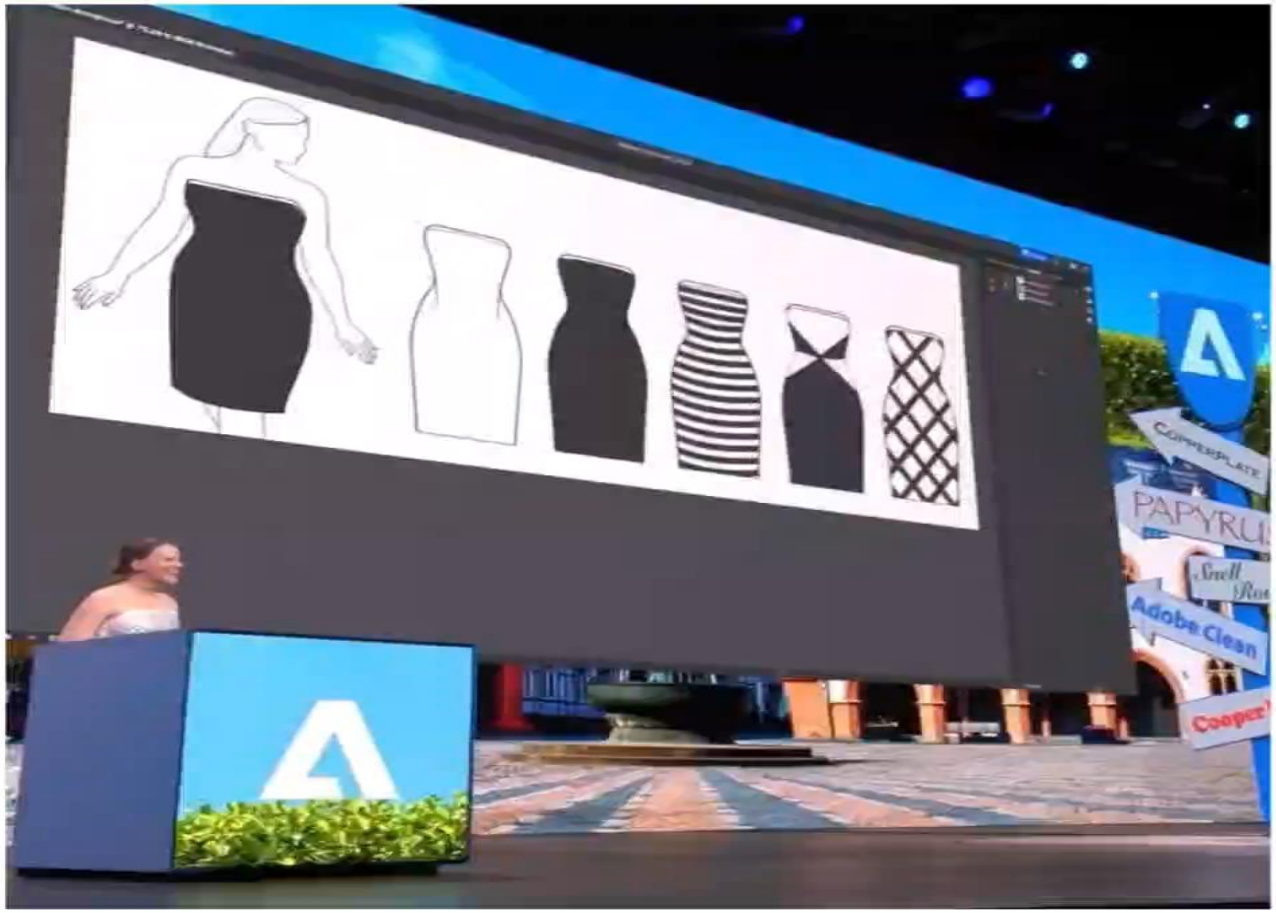
Petia Radeva

Why texture analysis?



Other virtual dressing apps

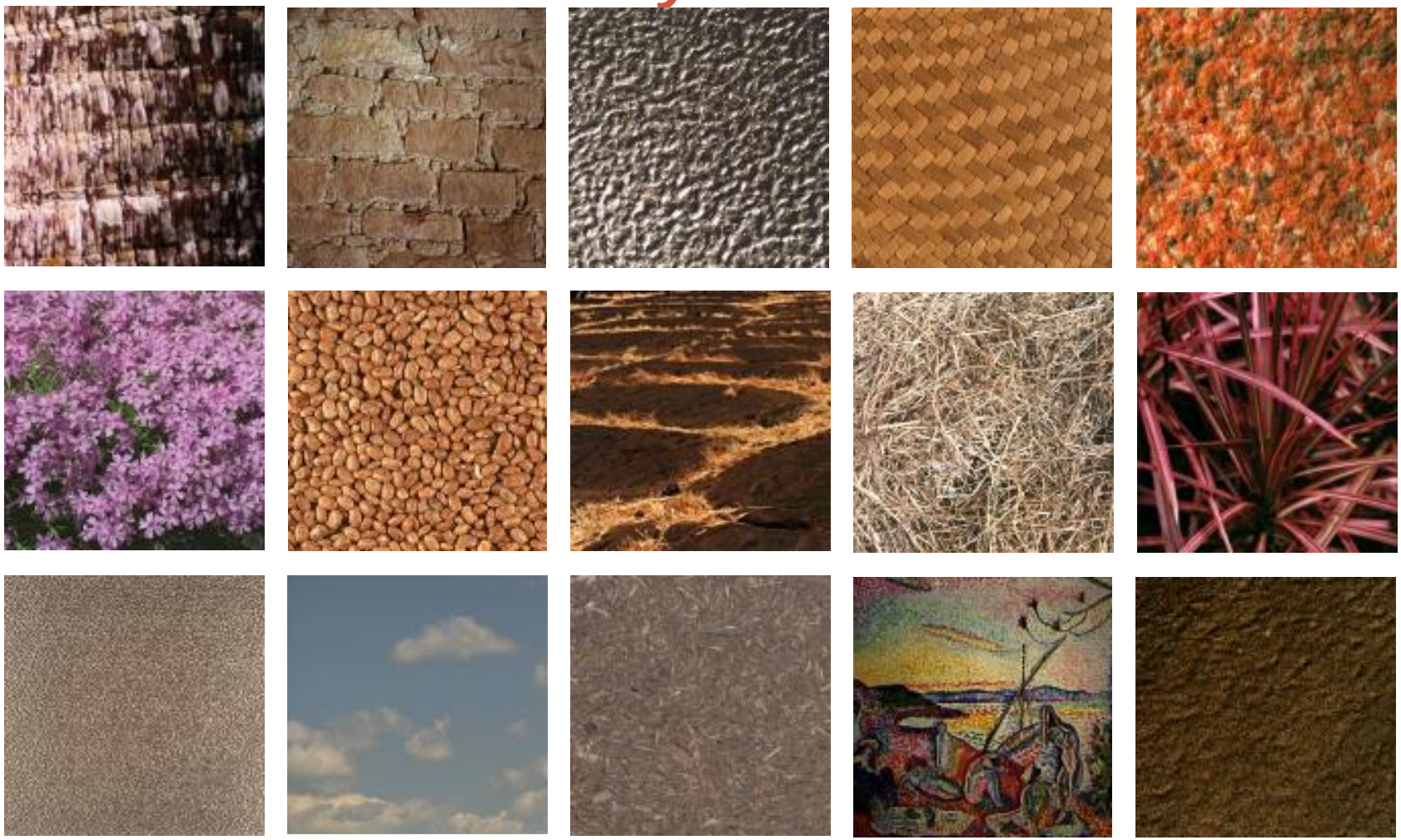
What's next in dressing?



Index

- What is a texture in an image?
- Discriminative textures and pre-attentivity
- Computational methods
  - Gaussian Filters
  - LBP
- Applications
- Exercise

What textures do you know?!

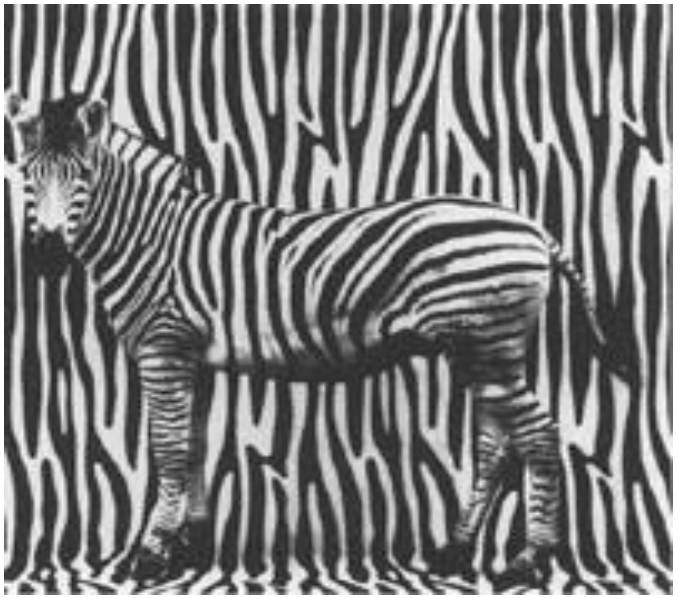


What does define a texture?

Why shell we use texture analysis?

Automatic analysis and perception of textures

- Image classification
- Image Segmentation
- Object recognition
- Perception of surface orientation, etc.



What applications can be defined for the texture analysis?

Defect Inspection

- textile
- cork
- Wooden furniture
- ceramic
- skin

Analysis of scenes

Improved images and photos

- Printers, cameras

Texture syntesis

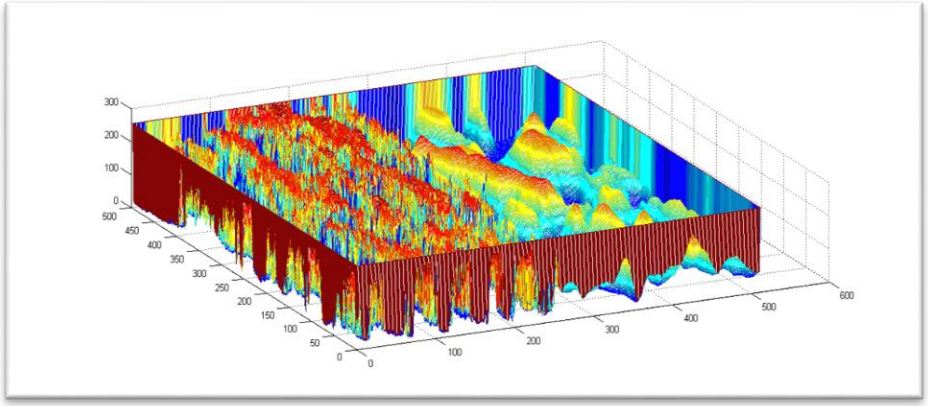
What is a texture?

Visual texture - when the uniformity of the region is perceived as a series of variations in the intensity.

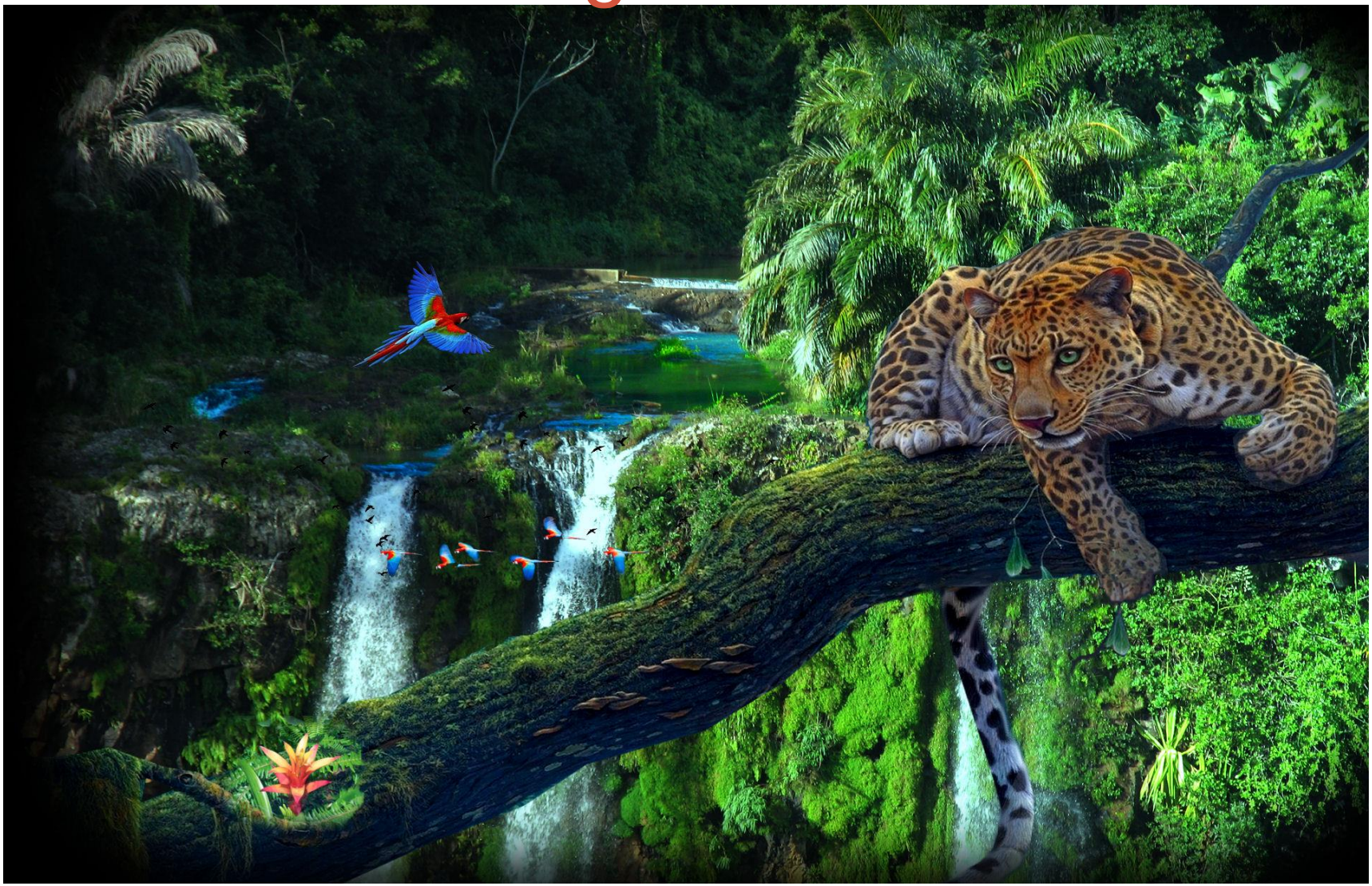
Causes of the visual texture appearance:

- Changes in reflectance.
- Changes in the micro-orientation of a surface.

Relevance: source of information about the scene, useful for certain visual processes (segmentation and perception of orientation).



Discriminating texture



…and for this image?

Index

- What is the texture of an image?
- Discriminative textures and pre-attentivity
- Computational methods
  - Gaussian Filters
  - LBP
- Applications
- Exercise

Computational methods for texture discrimination

Assumptions

Texture (stochastic and structural) is a property of regions:

- contextual property;

- description of the distribution of gray levels in regions more or less extensive.

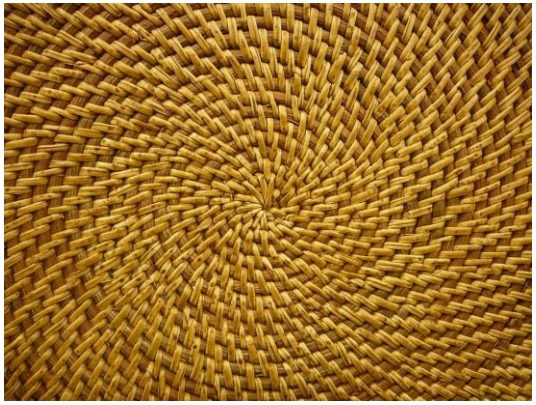
The texture can be perceived at different scales or levels of resolution.

A region presents a texture when the number of primitive patterns is large (not easily countable).

Texture representation

Assumptions

- Consider textures as made up of repeated local patterns, so:
  1. Find the patterns
     - Use filters that look like patterns (spots, bars, raw patches…)
     - Consider magnitude of response

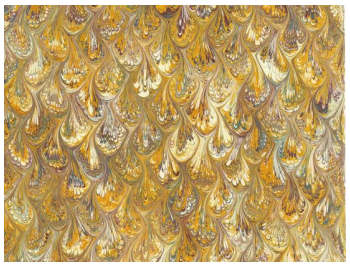


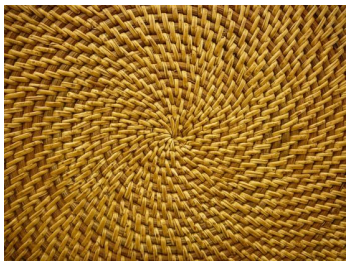
  2. Describe their statistics within each local window
     - Mean, standard deviation
     - Histogram
     - Histogram of “prototypical” feature occurrences (textons)

Texture representation

Assumptions

- Consider textures as made up of repeated local patterns, so:
  1. Find the patterns
     - Use filters that look like patterns (spots, bars, raw patches…)
     - Consider magnitude of response





  2. Describe their statistics within each local window
     - Mean, standard deviation
     - Histogram
     - Histogram of “prototypical” feature occurrences (textons)

What kind of filters to use?

Buildings

Forest

Sunset

Different textures have different attributes -> we need different filters!

Recall

- We looked for edges convolving with filters like:

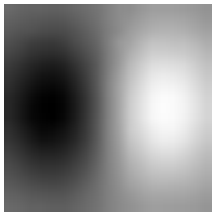
Sobel: $M_x =$

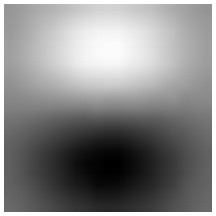
-1	0	1
-2	0	2
-1	0	1

; $M_y =$

1	2	1
0	0	0
-1	-2	-1

Derivatives of
of Gaussians

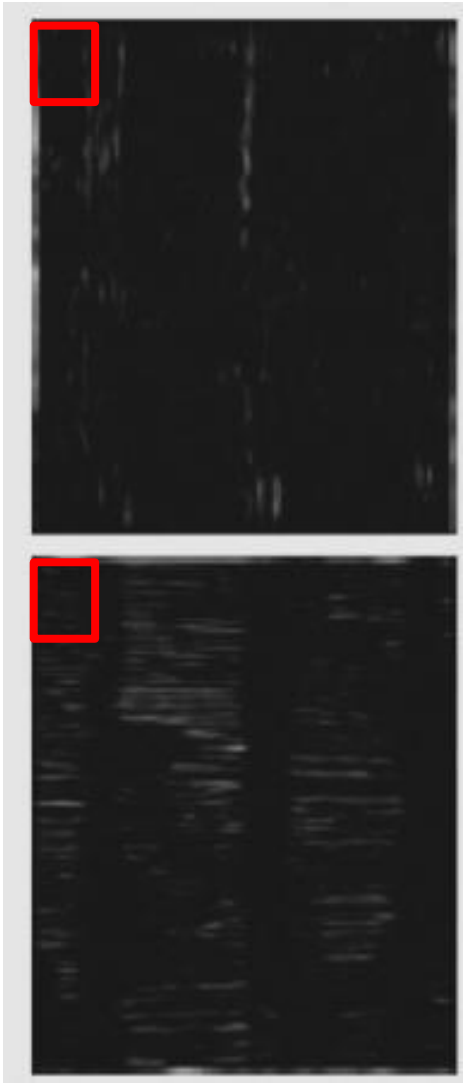




Convolution filters (kernels) are “imitations” of structures we are looking for.

Texture representation: example

original image



derivative filter responses, squared

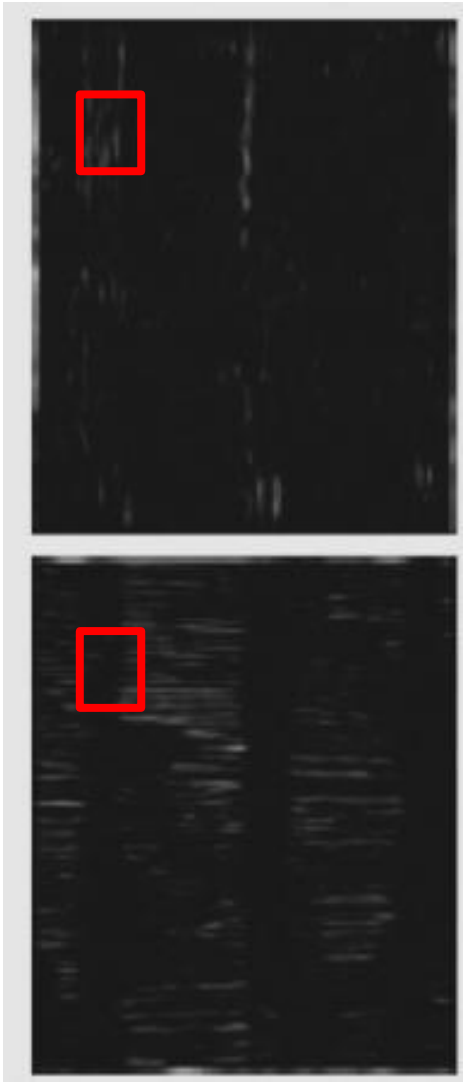
	mean d/dx value	mean d/dy value
Win. #1	4	10

⋮

statistics to summarize patterns in small windows

Texture representation: example

original image



derivative filter responses, squared

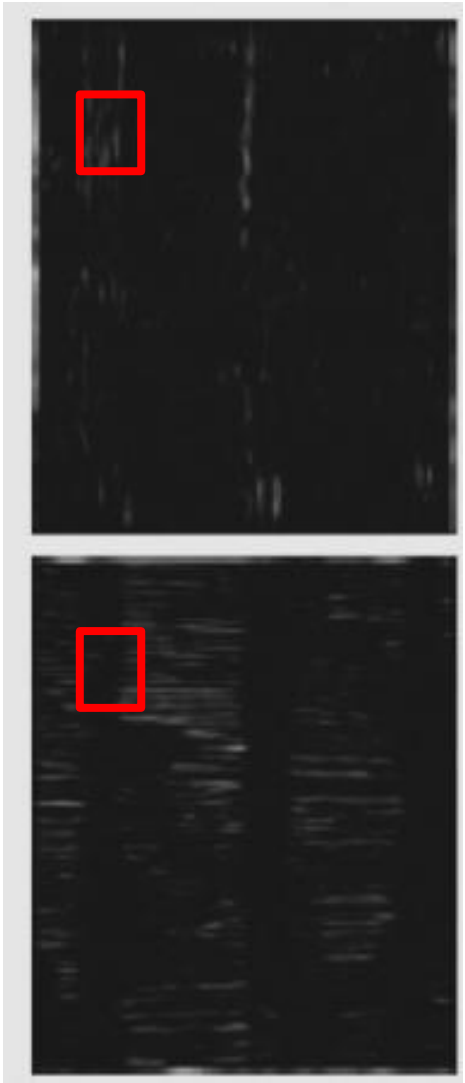
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7	

⋮

statistics to summarize patterns in small windows

Texture representation: example

original image



derivative filter responses, squared

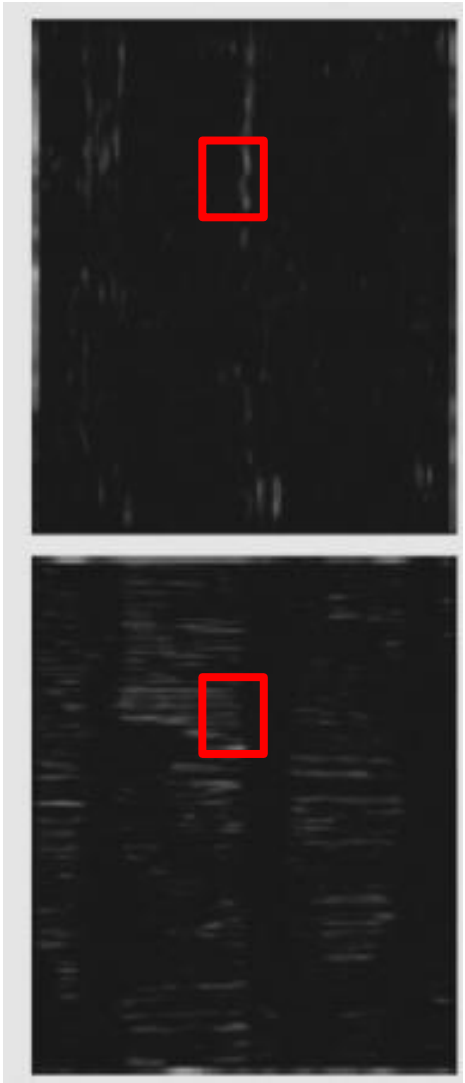
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7	

⋮

statistics to summarize patterns in small windows

Texture representation: example

original image



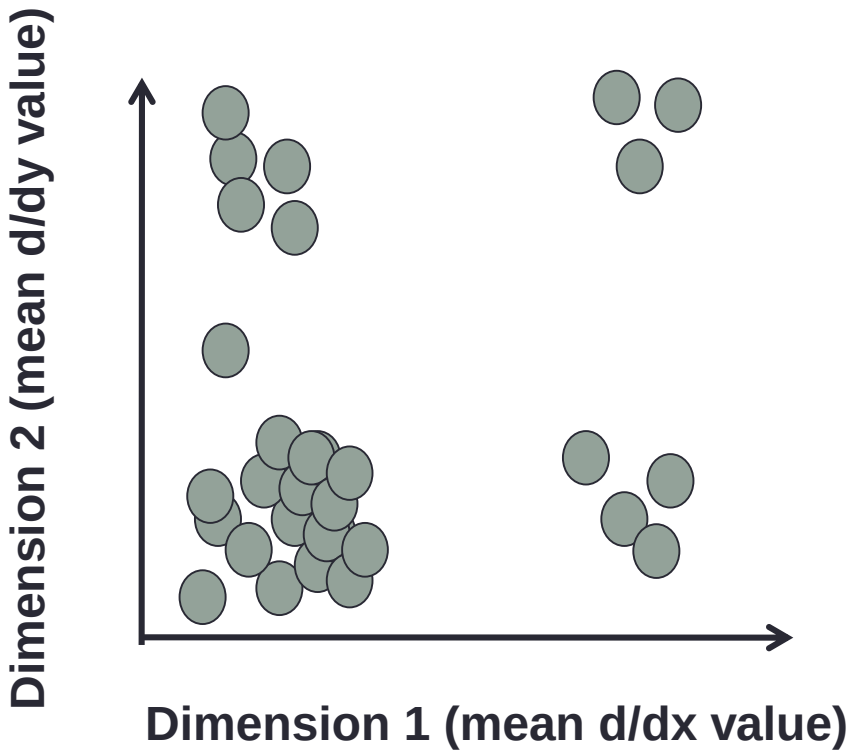
derivative filter responses, squared

	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

⋮

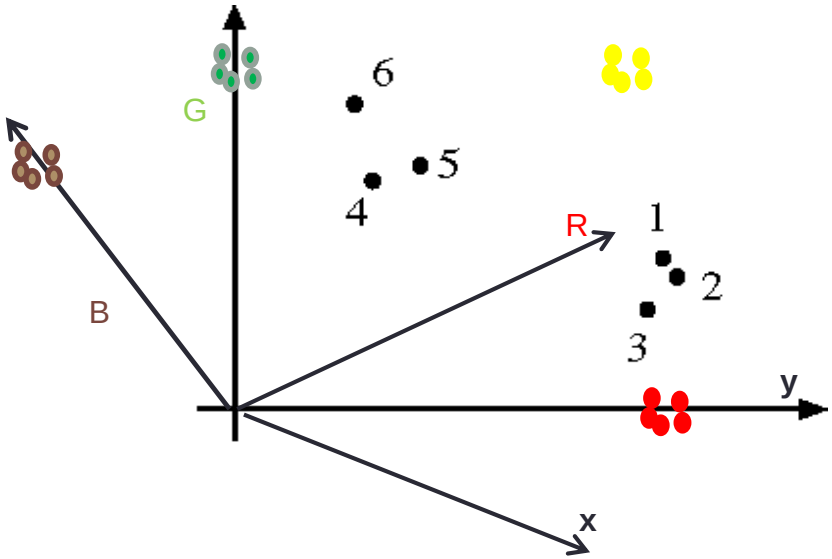
statistics to summarize patterns in small windows

Remimber: a simple segmentation algorithm

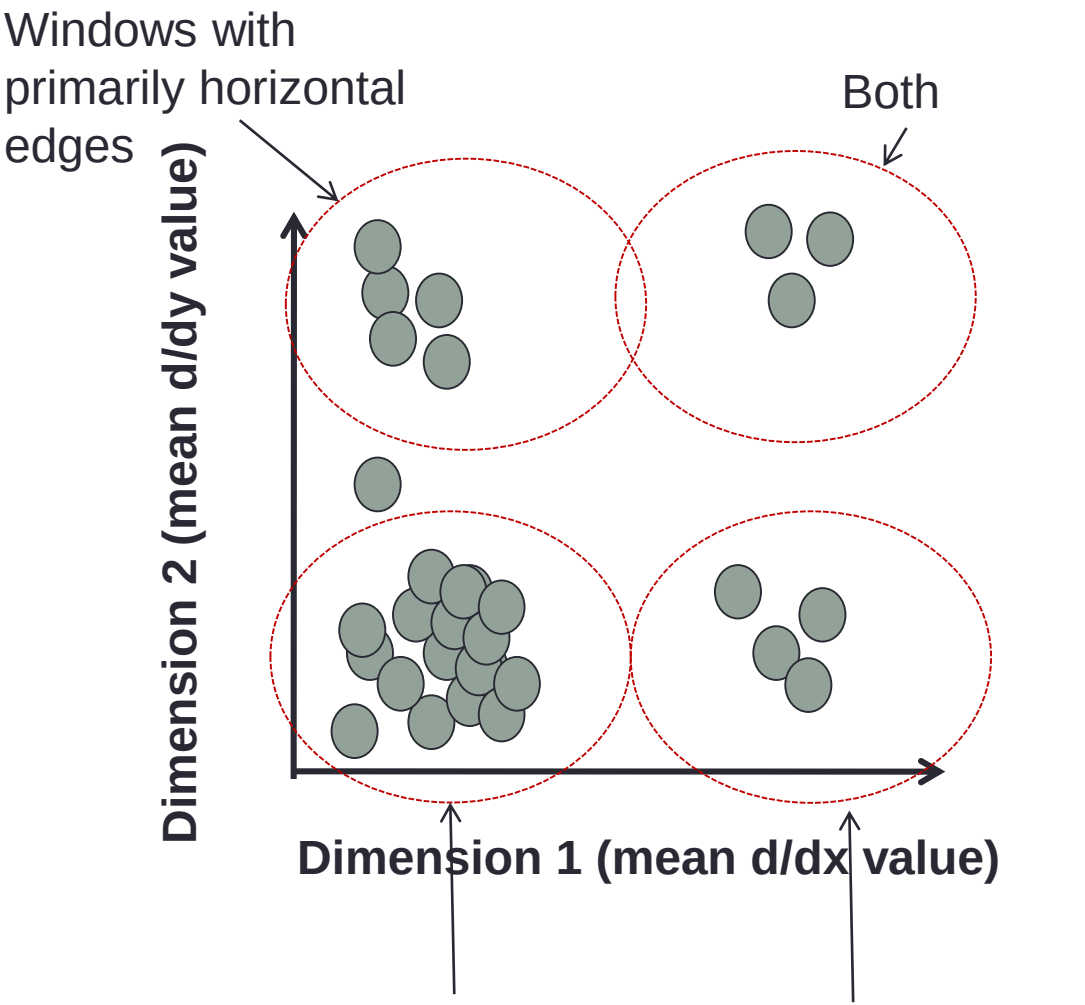
- Each pixel of the image is described by a vector in the feature space:

  z = (R,G,B,x,y), …

- Run a clustering algorithm using some distance between pixels (e.g. Euclidean distance)



Texture representation: example



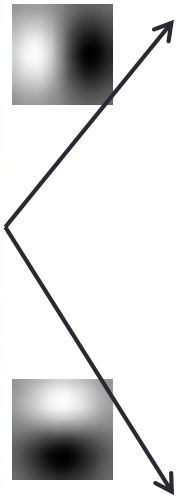
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20

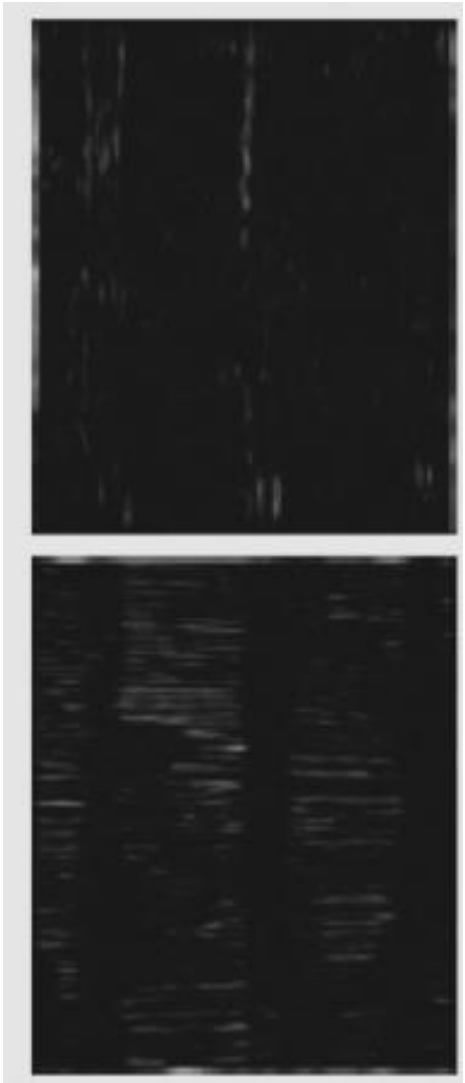
⋮

statistics to summarize patterns in small windows

Texture representation: example

original image

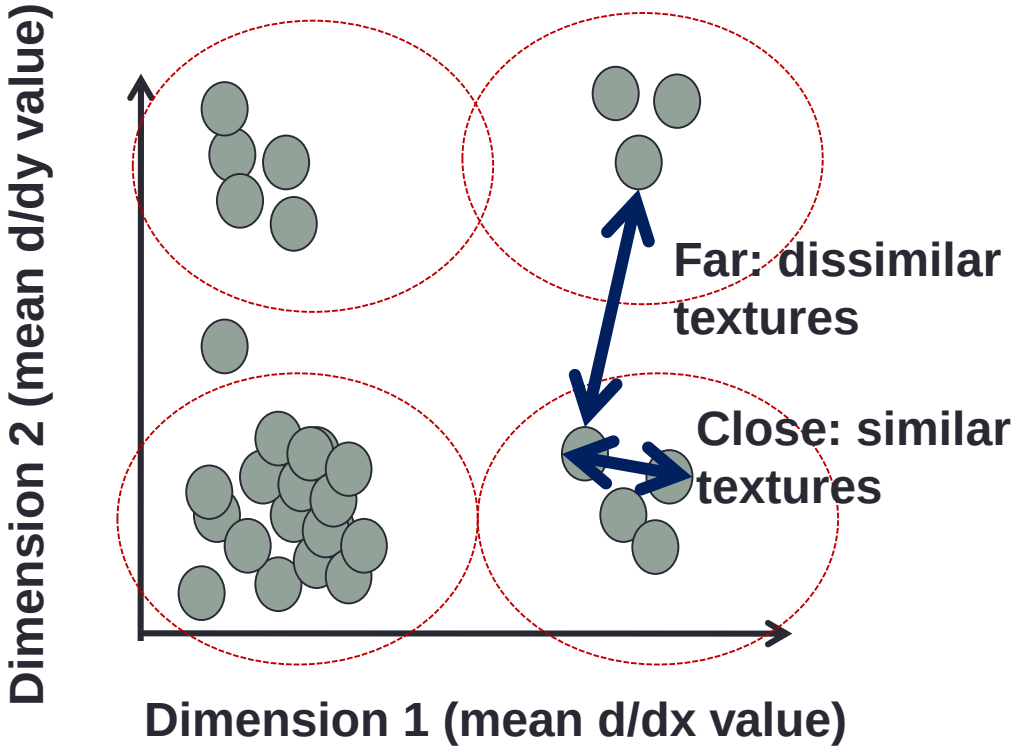




derivative filter responses, squared

visualization of the assignment to texture “types”

Texture representation: example

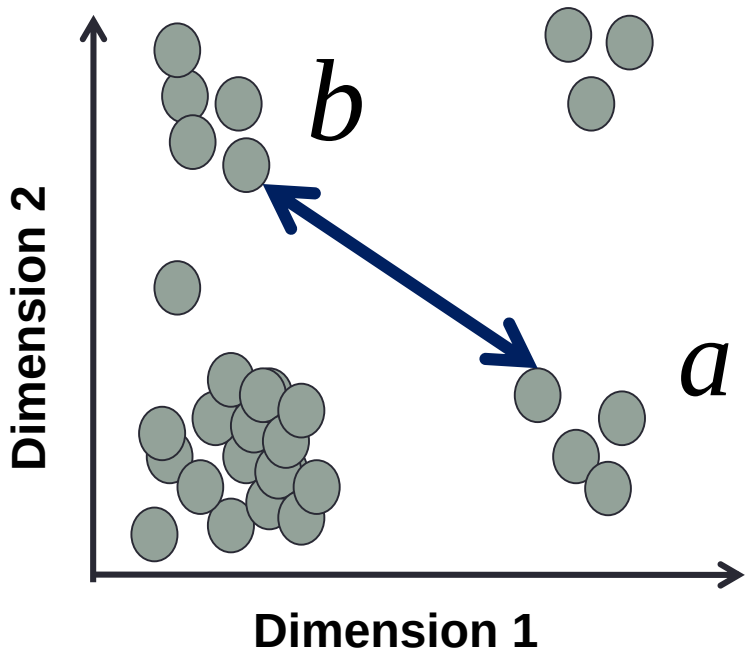


	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20

⋮

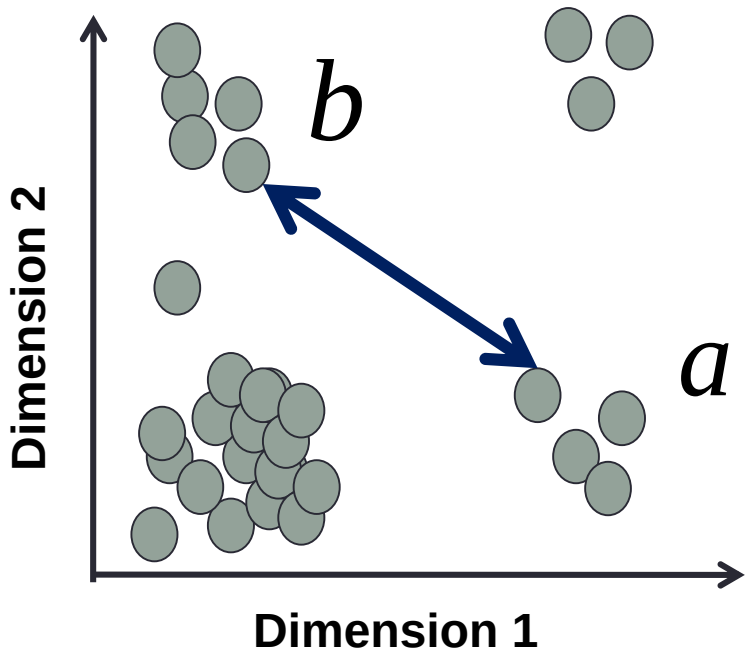
statistics to summarize patterns in small windows

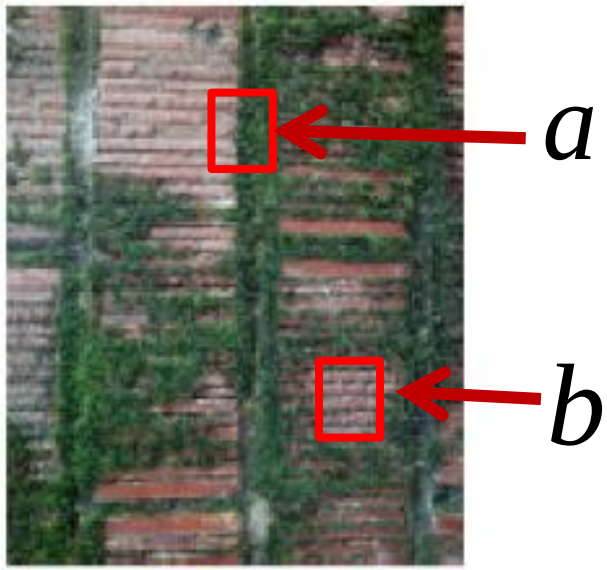
Texture representation: example

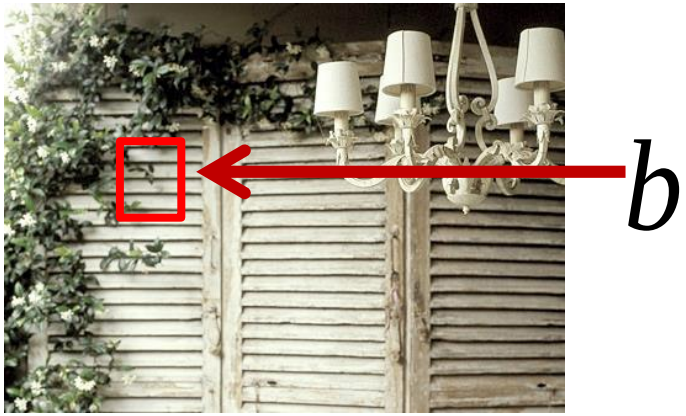


$$D(a,b)=\sqrt{(a_1-b_1)^2+(a_2-b_2)^2}$$

Texture representation: example







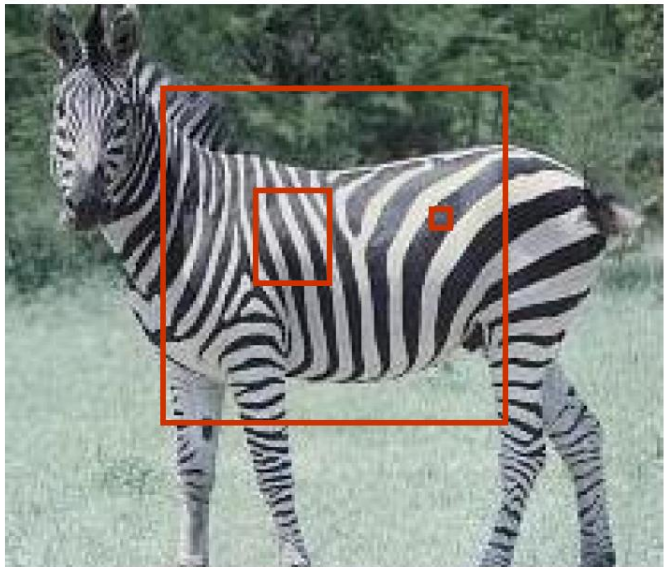
Distance reveals how dissimilar is texture of window a is from texture in window b.

If the whole image contains the same texture, what would be the feature space representation looks like?

Kristen Grauman

Texture representation: window scale

- We’re assuming we know the relevant window size for which we collect these statistics.

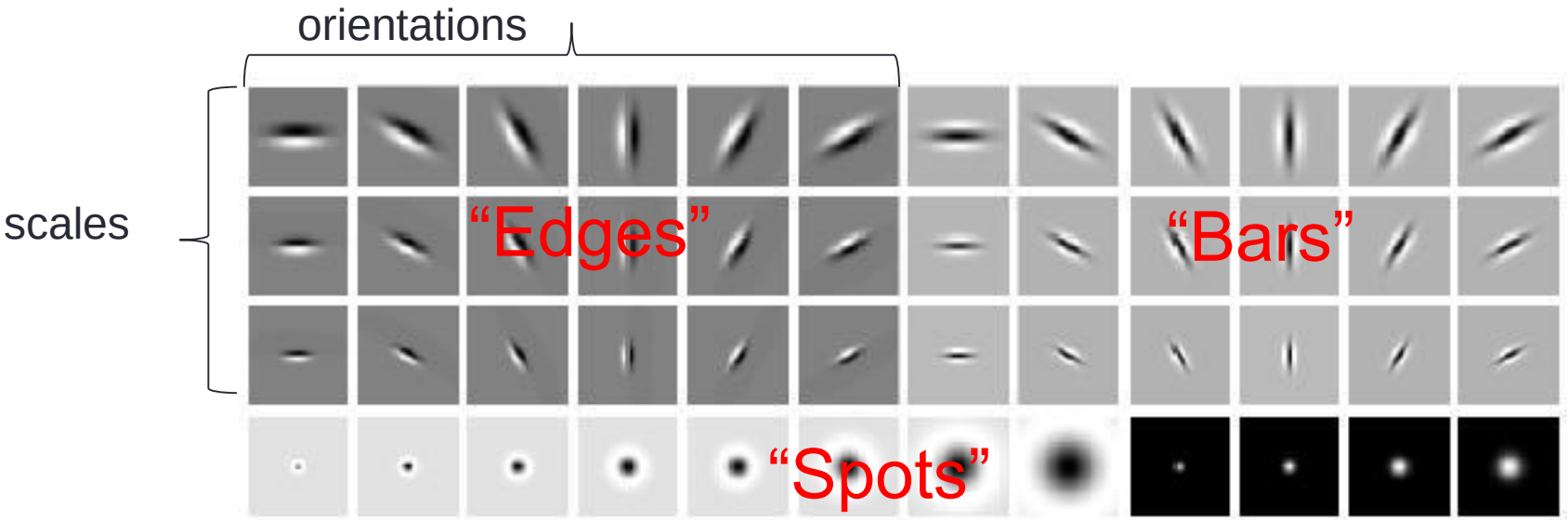


Possible to perform scale selection by looking for window scale where texture description is not changing.

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

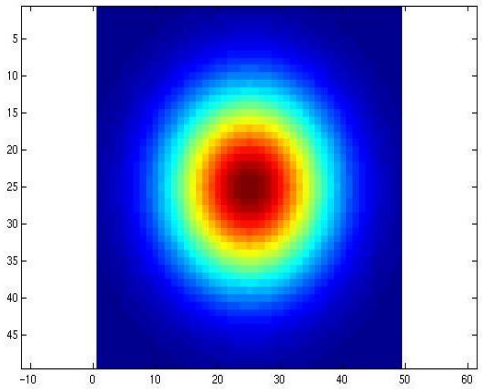
Filter banks



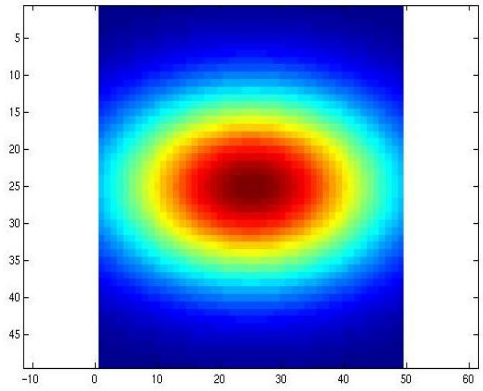
- What filters to put in the bank?
  - Typically, we want a combination of scales and orientations, different types of patterns.

How to generate filters: using a Gaussian function

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

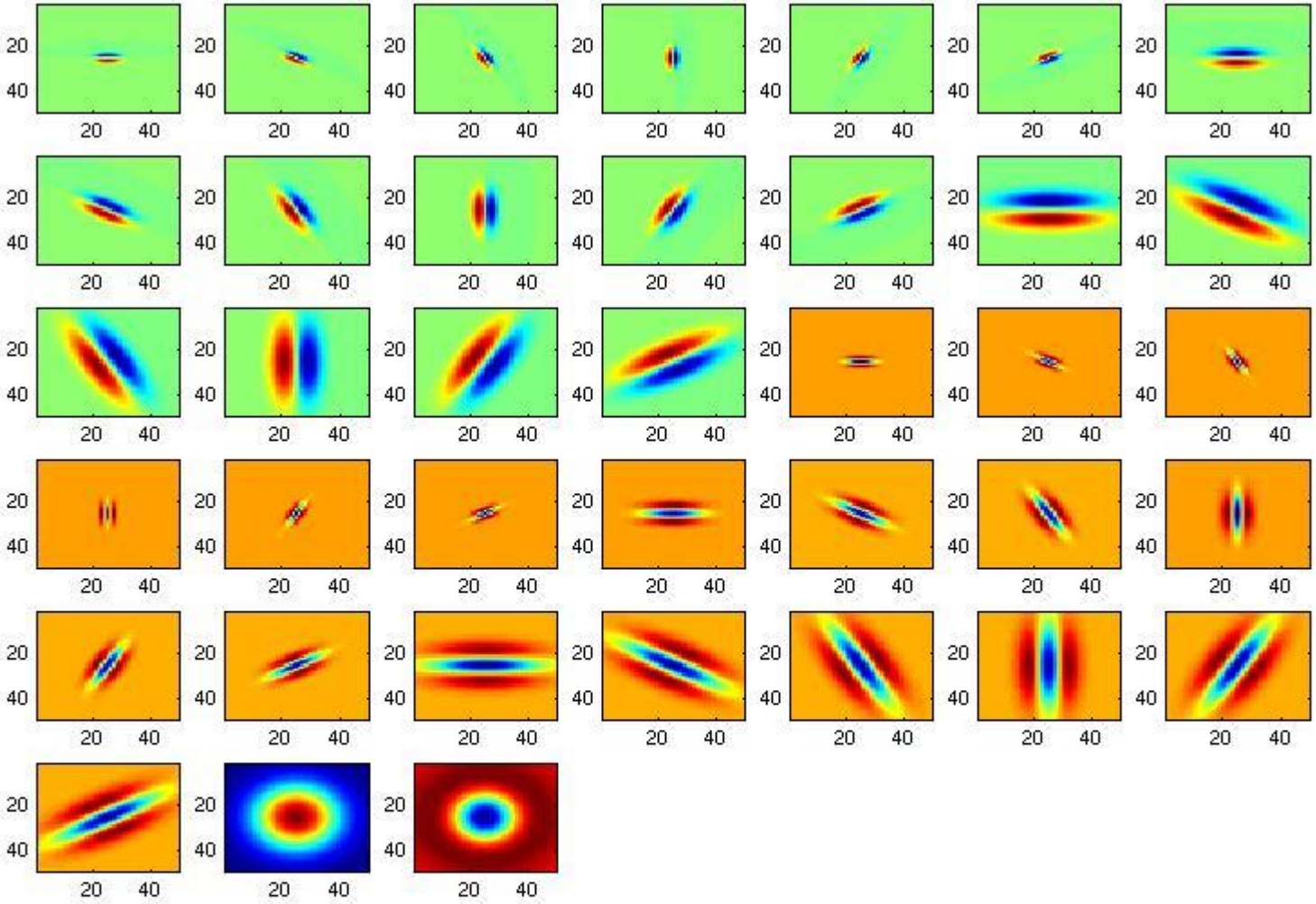


$$\Sigma = \begin{bmatrix} 9 & 0 \\ 0 & 9 \end{bmatrix}$$



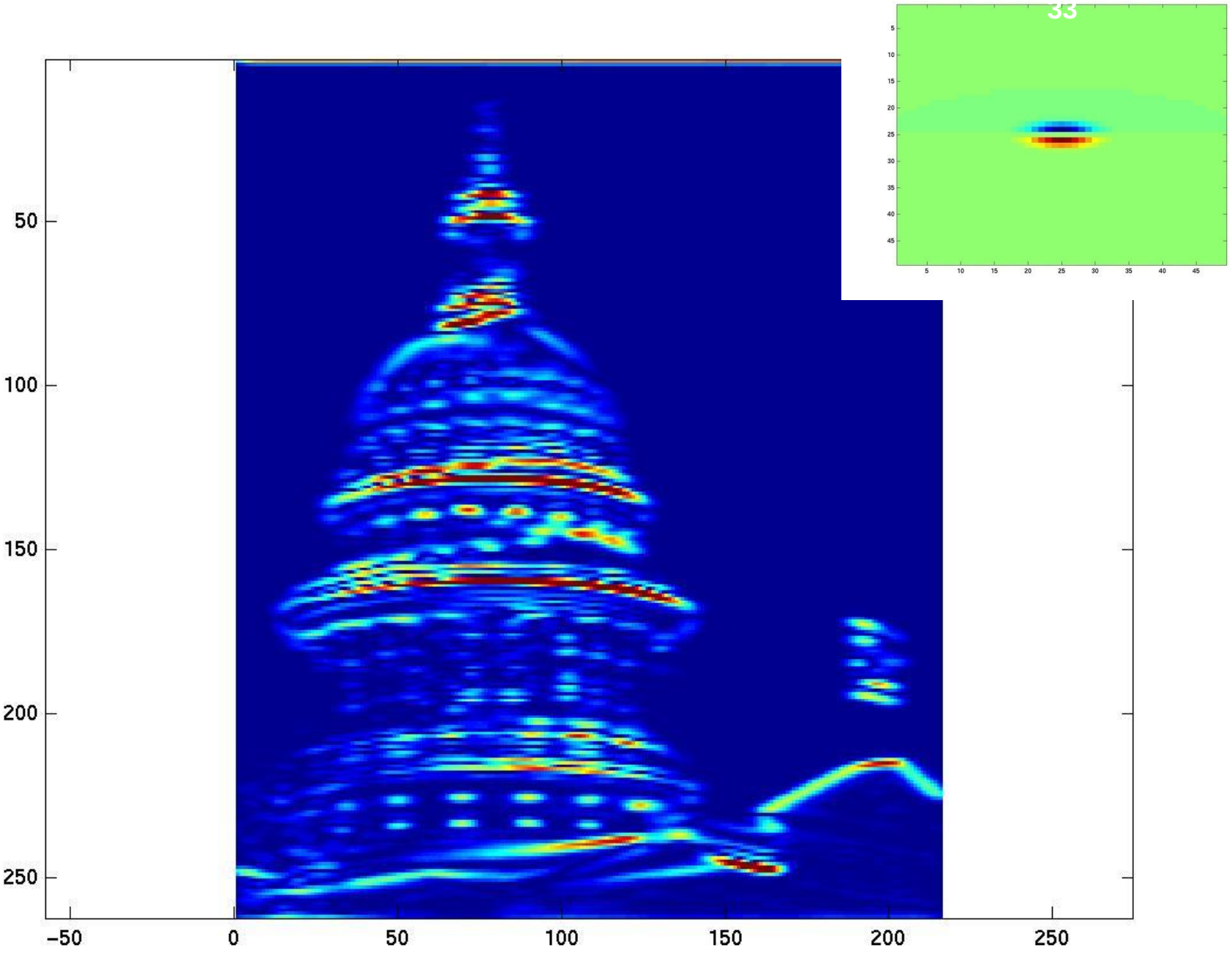
$$\Sigma = \begin{bmatrix} 16 & 0 \\ 0 & 9 \end{bmatrix}$$

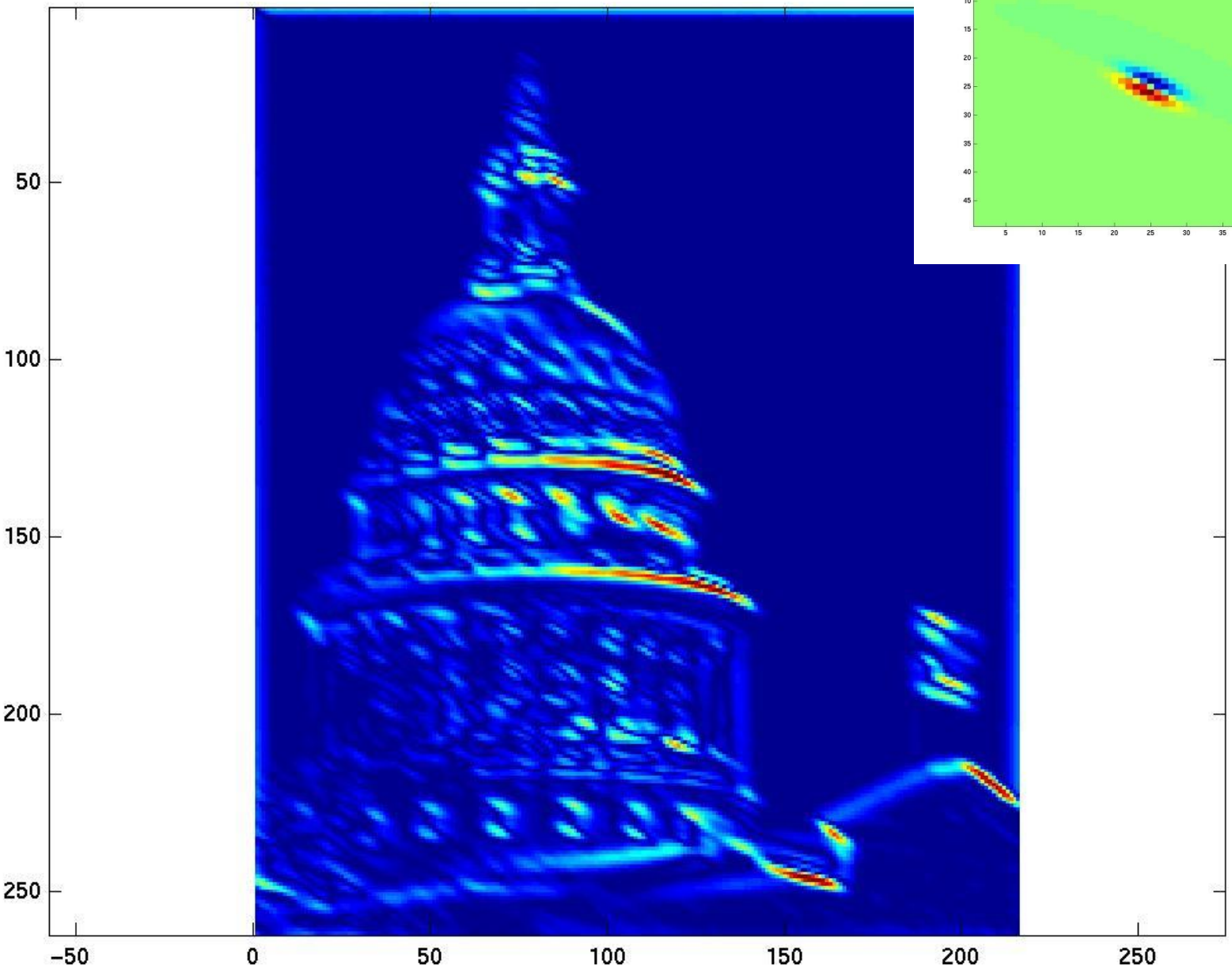
Filter bank

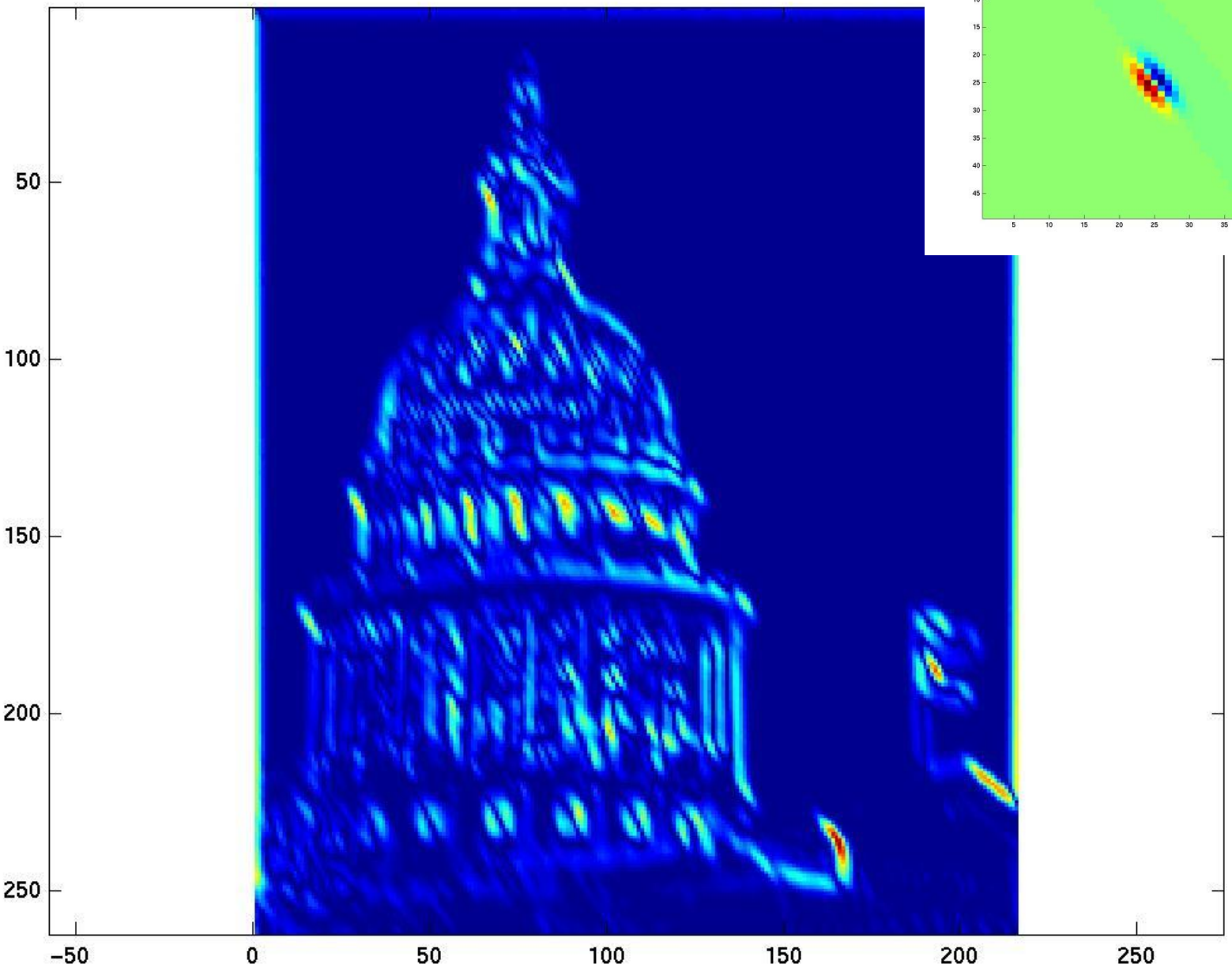


Example

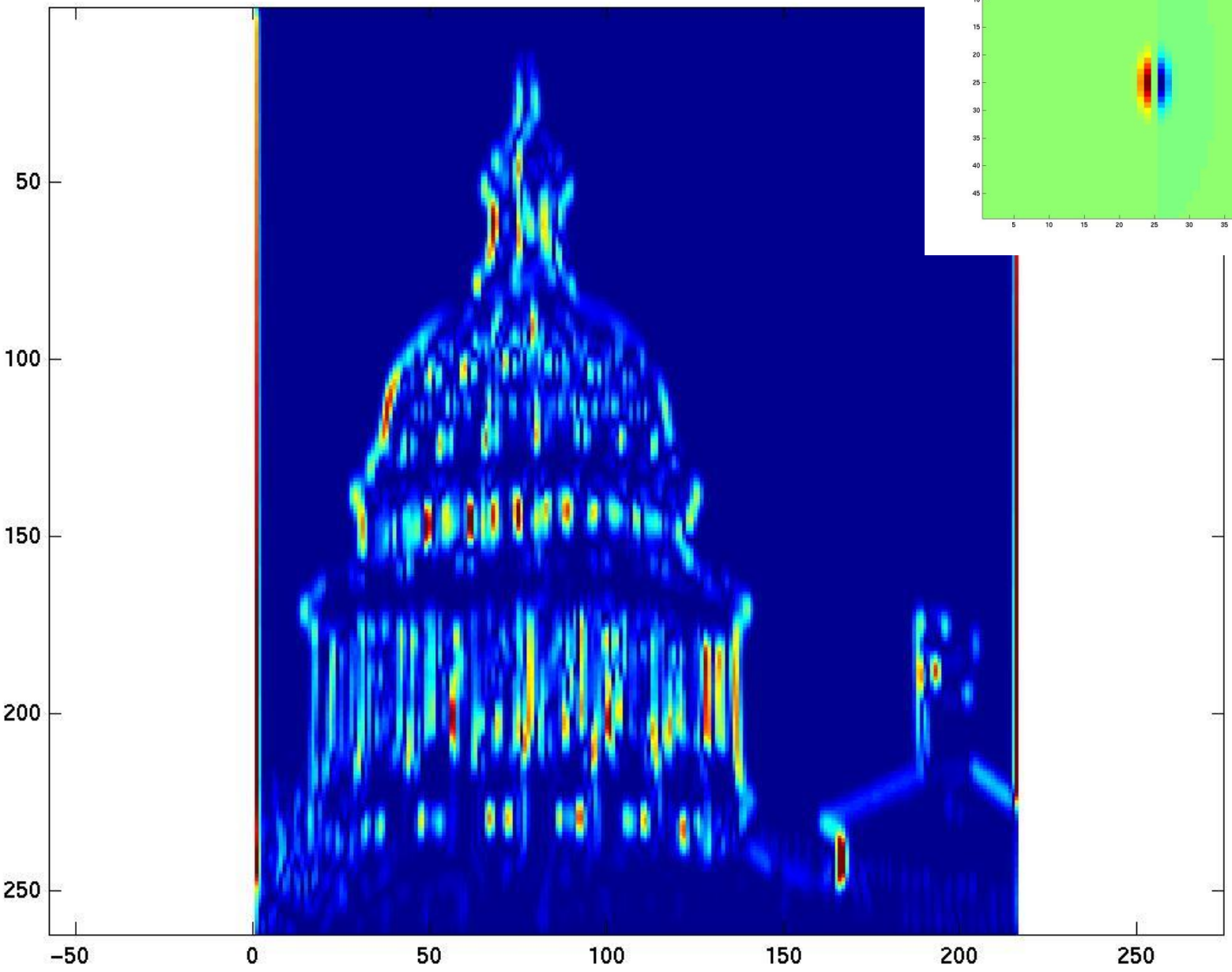
Image from http://www.texasexplorer.com/austincap2.jpg


33
5
10
15
20
25
30
35
40
45
50
100
150
200
250
−50
0
50
100
150
200
250

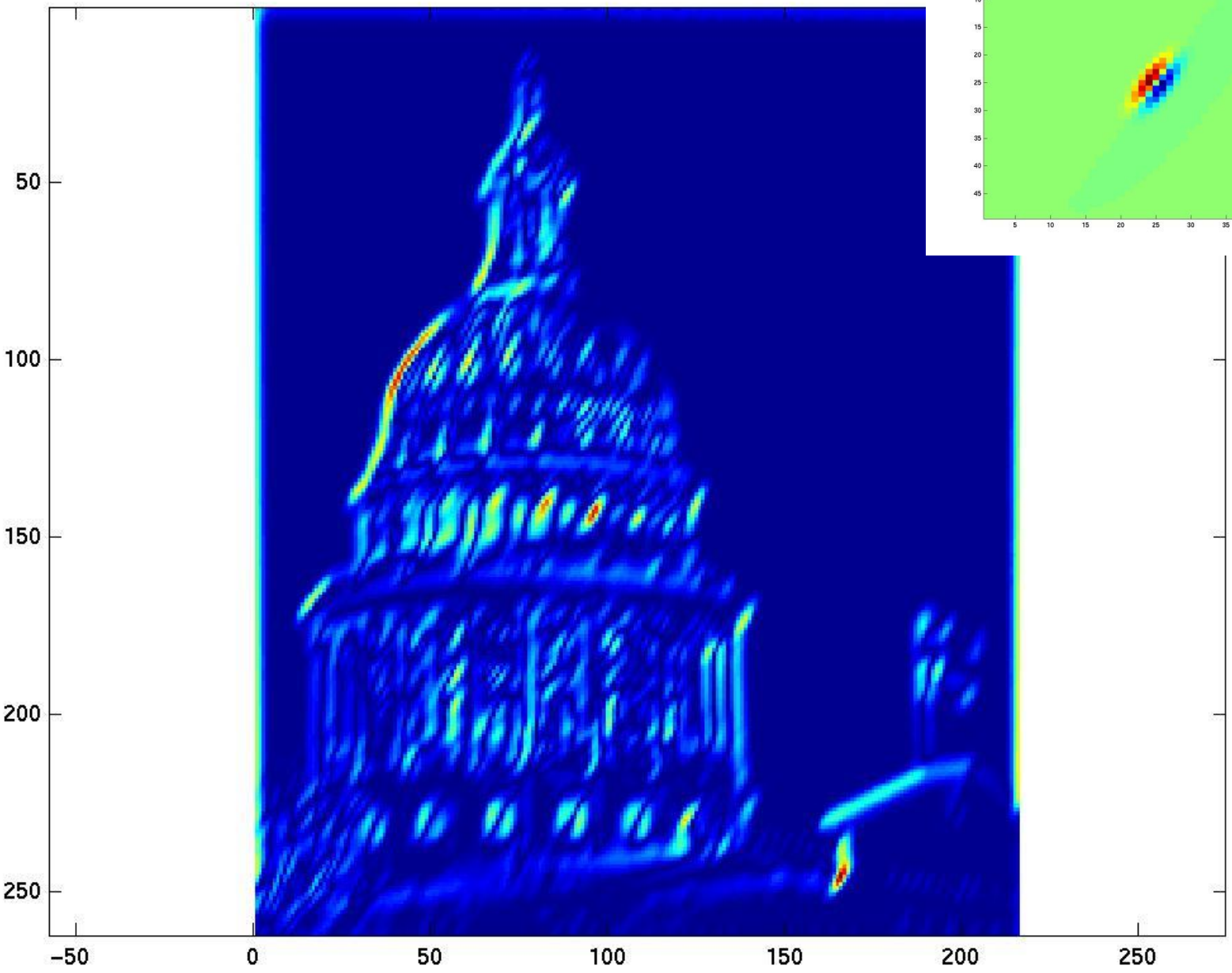

50
100
150
200
250
−50
0
50
100
150
200
250


50
100
150
200
250
−50
0
50
100
150
200
250

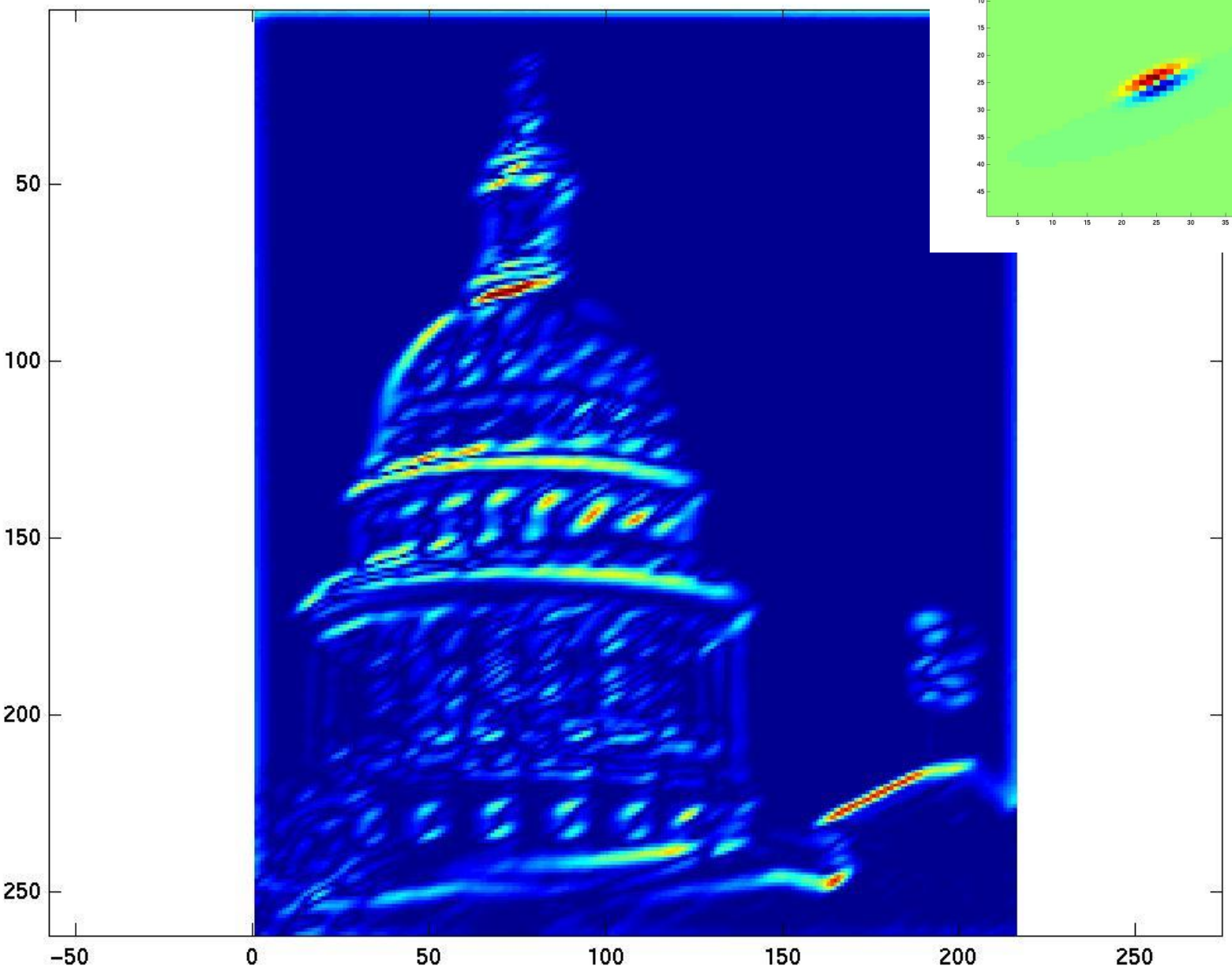
5
10
15
20
25
30
35
40
45
5
10
15
20
25
30
35
40
45

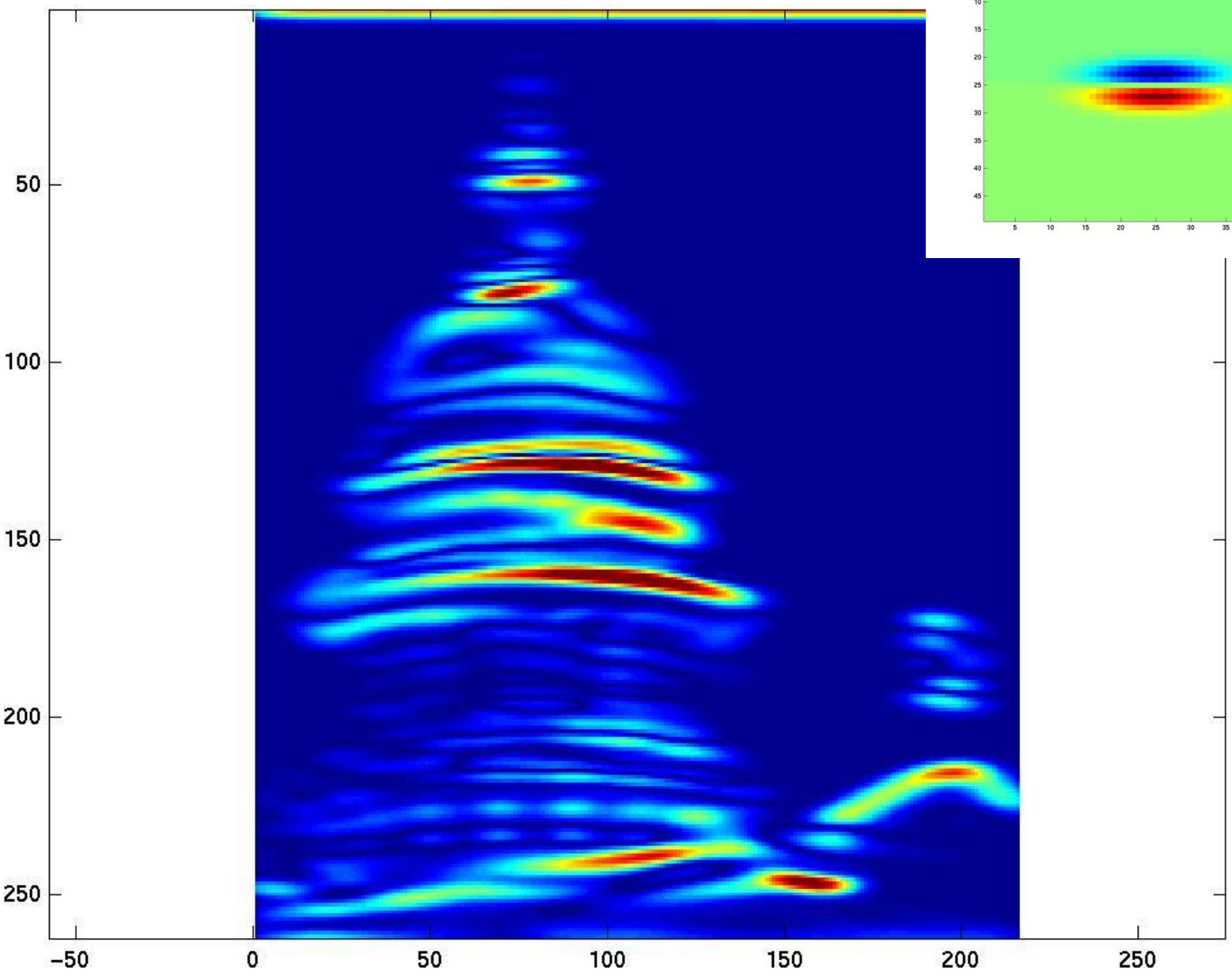

50
100
150
200
250
−50
0
50
100
150
200
250

5
10
15
20
25
30
35
40
45

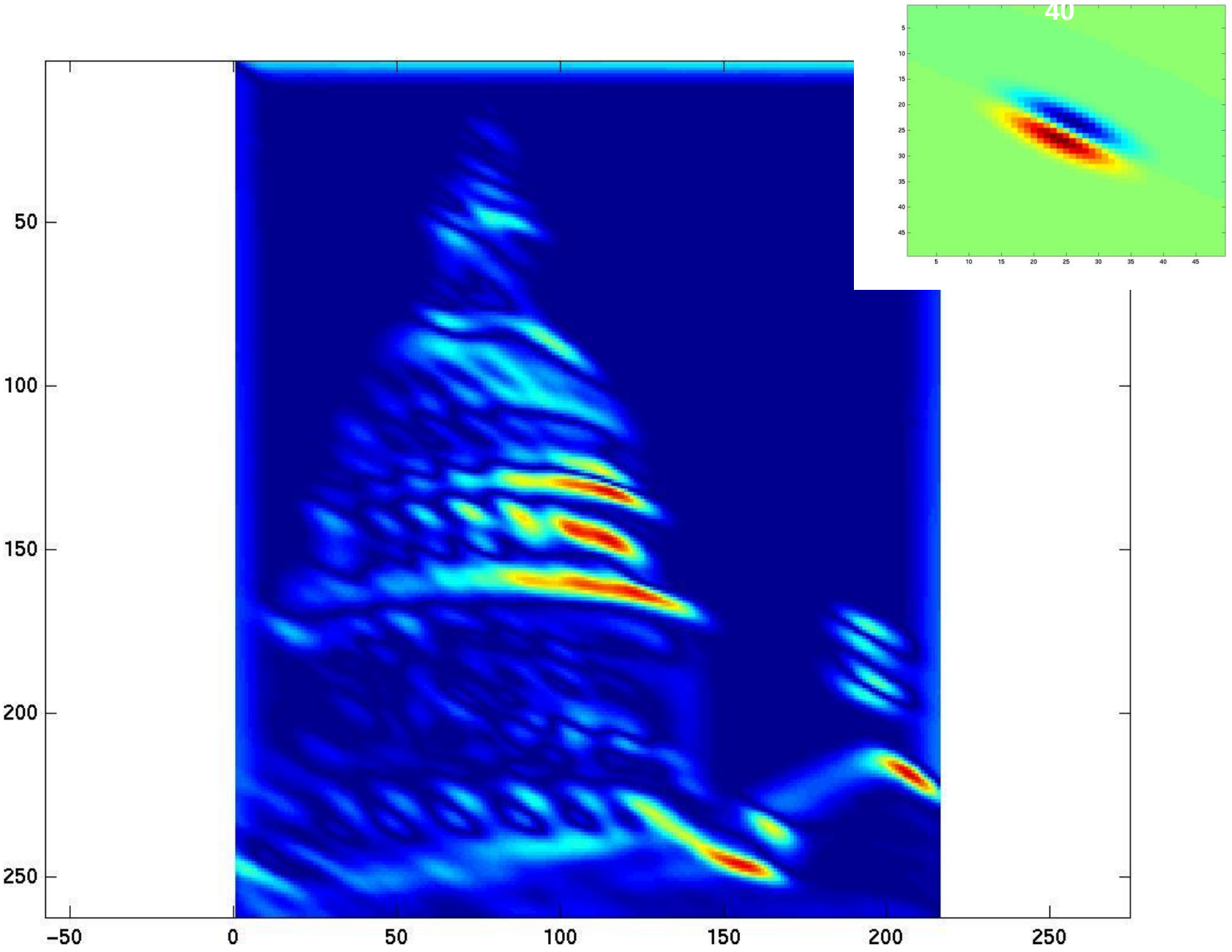

50
100
150
200
250
−50
0
50
100
150
200
250

5
10
15
20
25
30
35
40
45
5
10
15
20
25
30
35
40
45

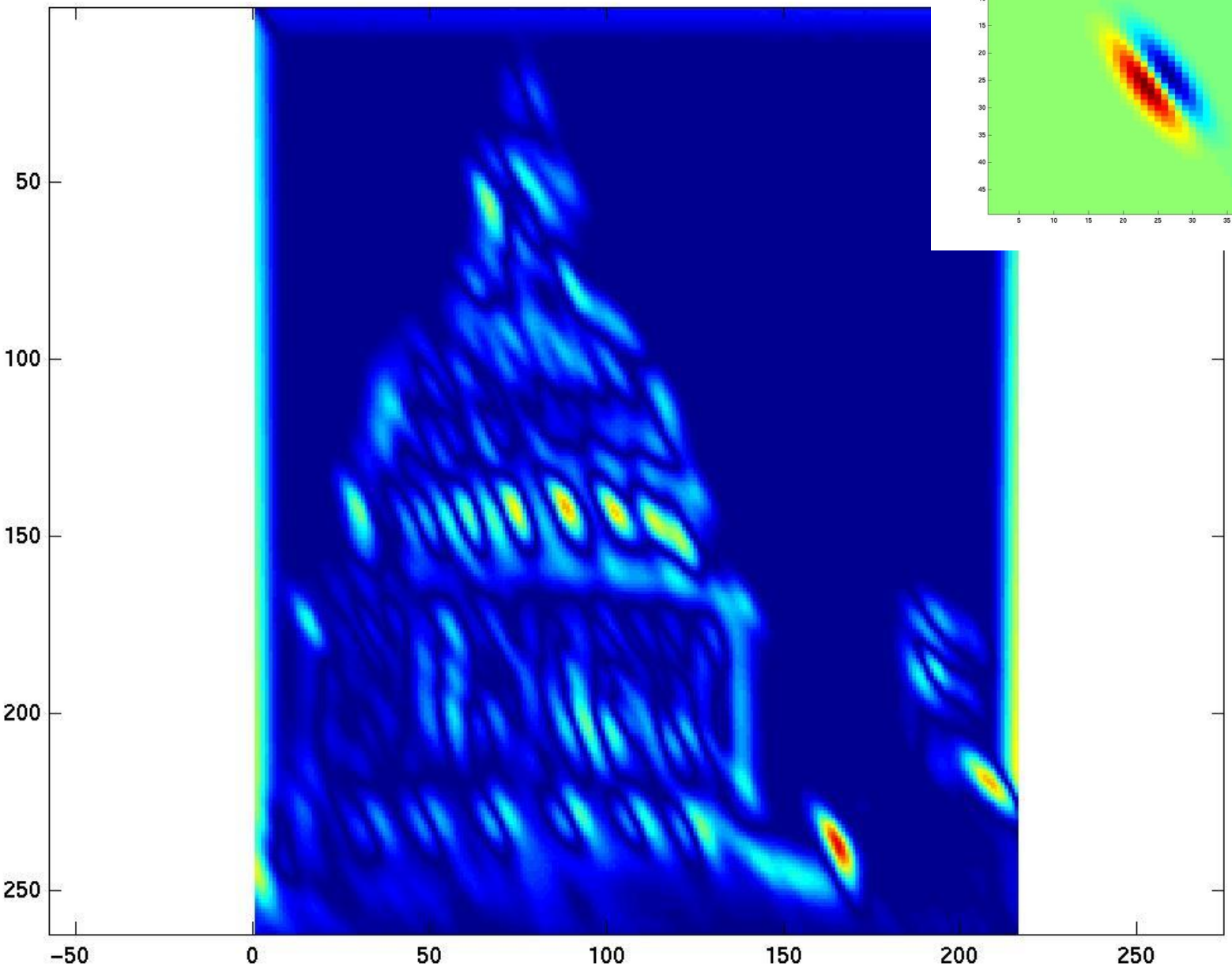

50
100
150
200
250
−50
0
50
100
150
200
250

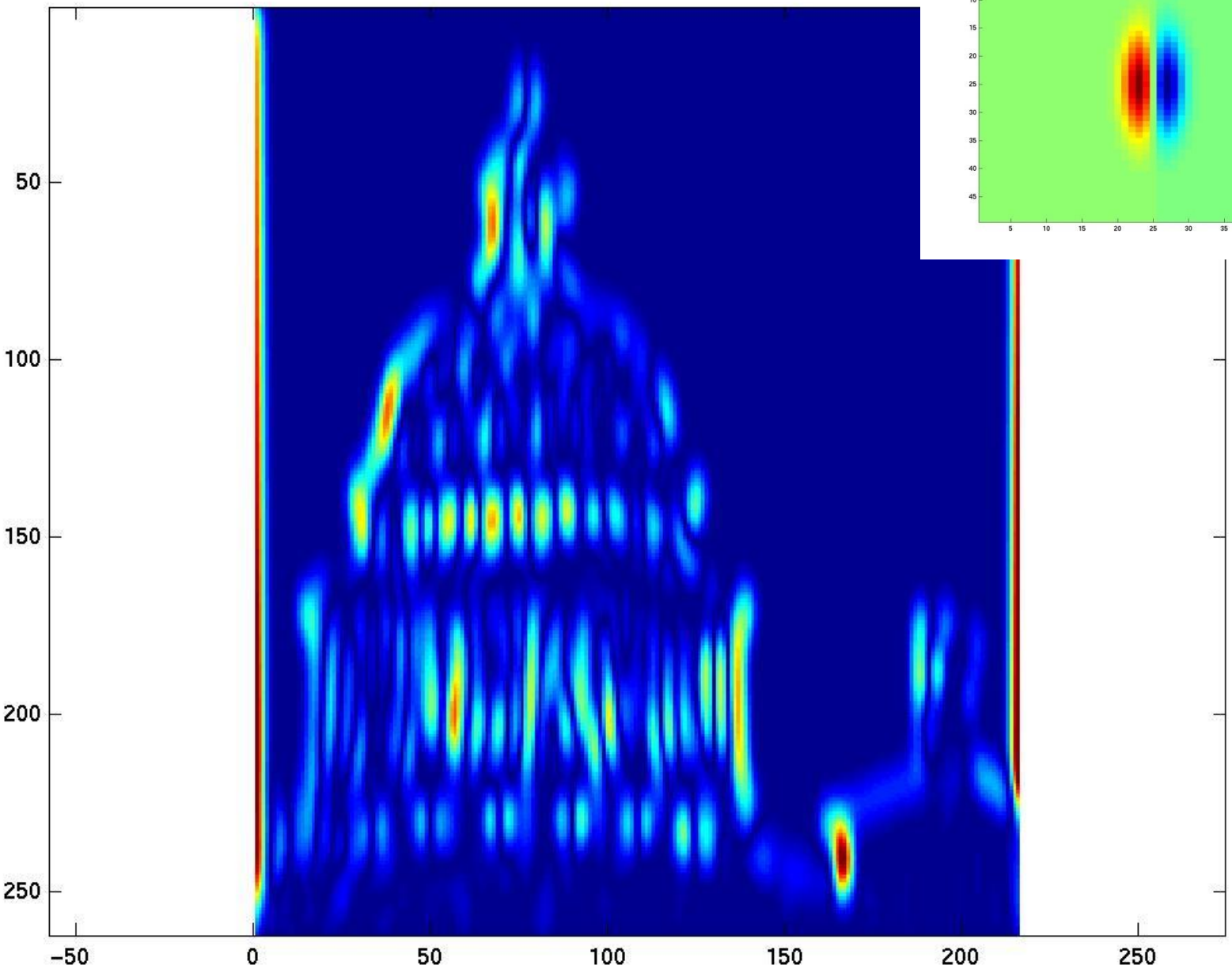

50
100
150
200
250
−50
0
50
100
150
200
250

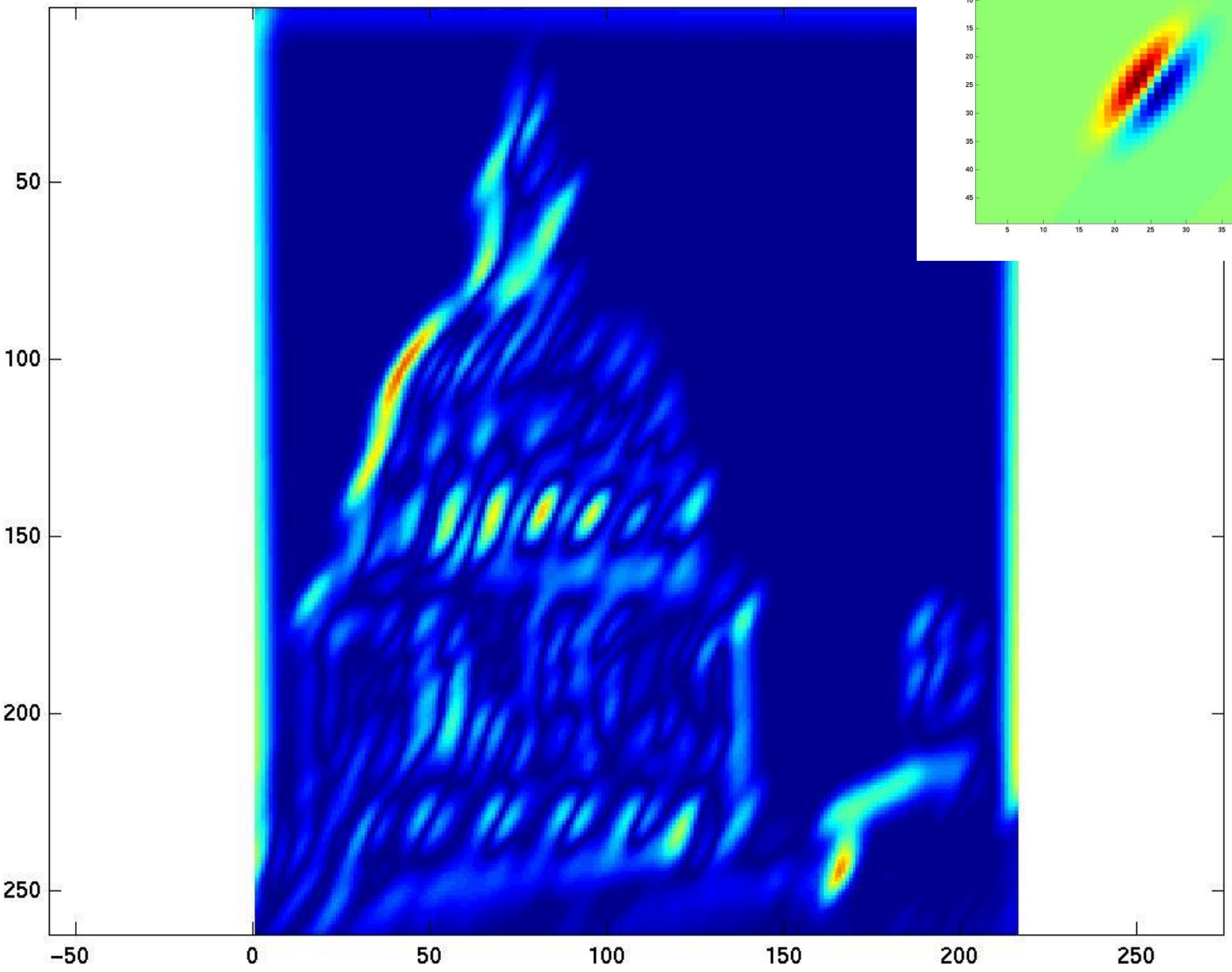
5
10
15
20
25
30
35
40
45



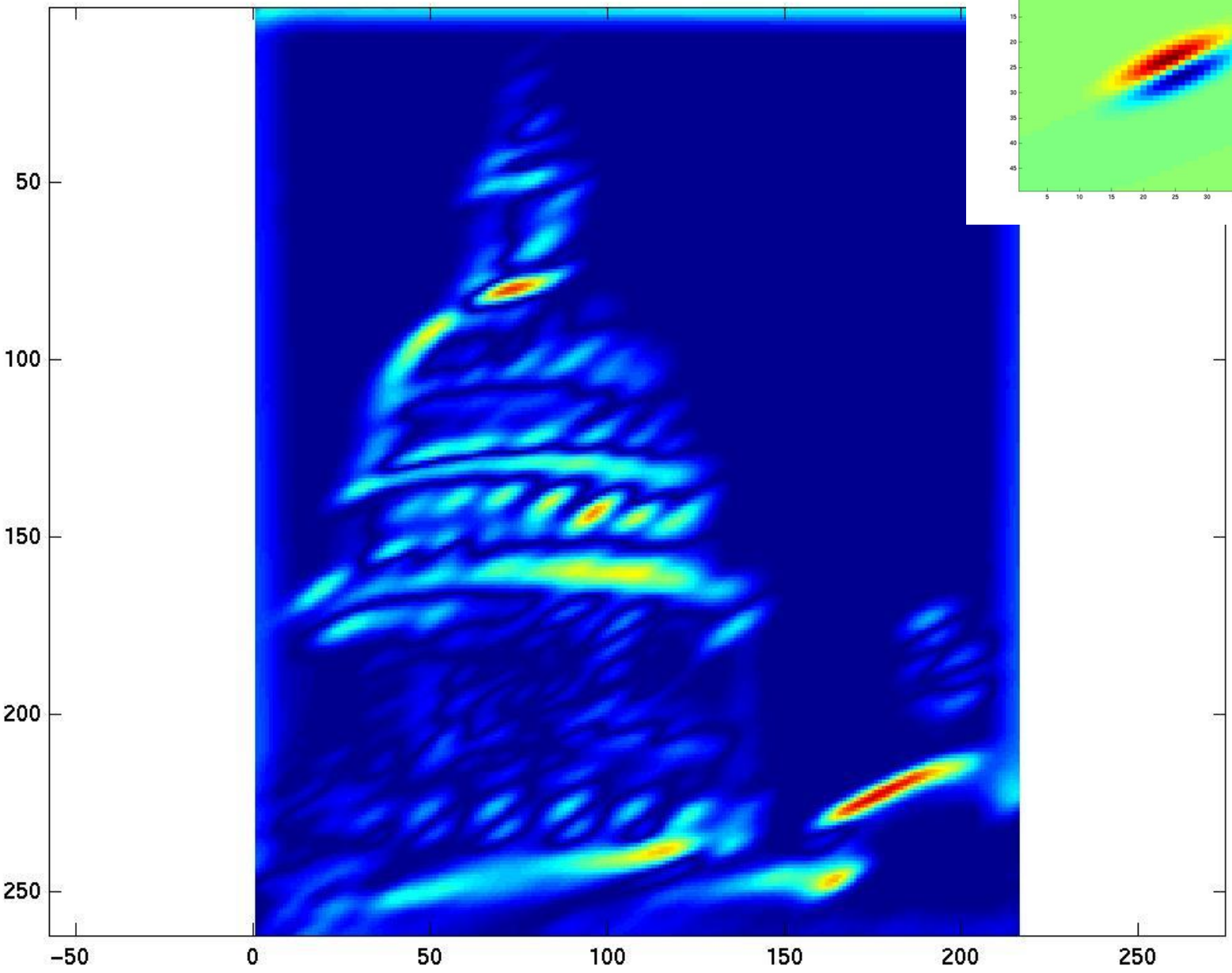
Kristen Grauman

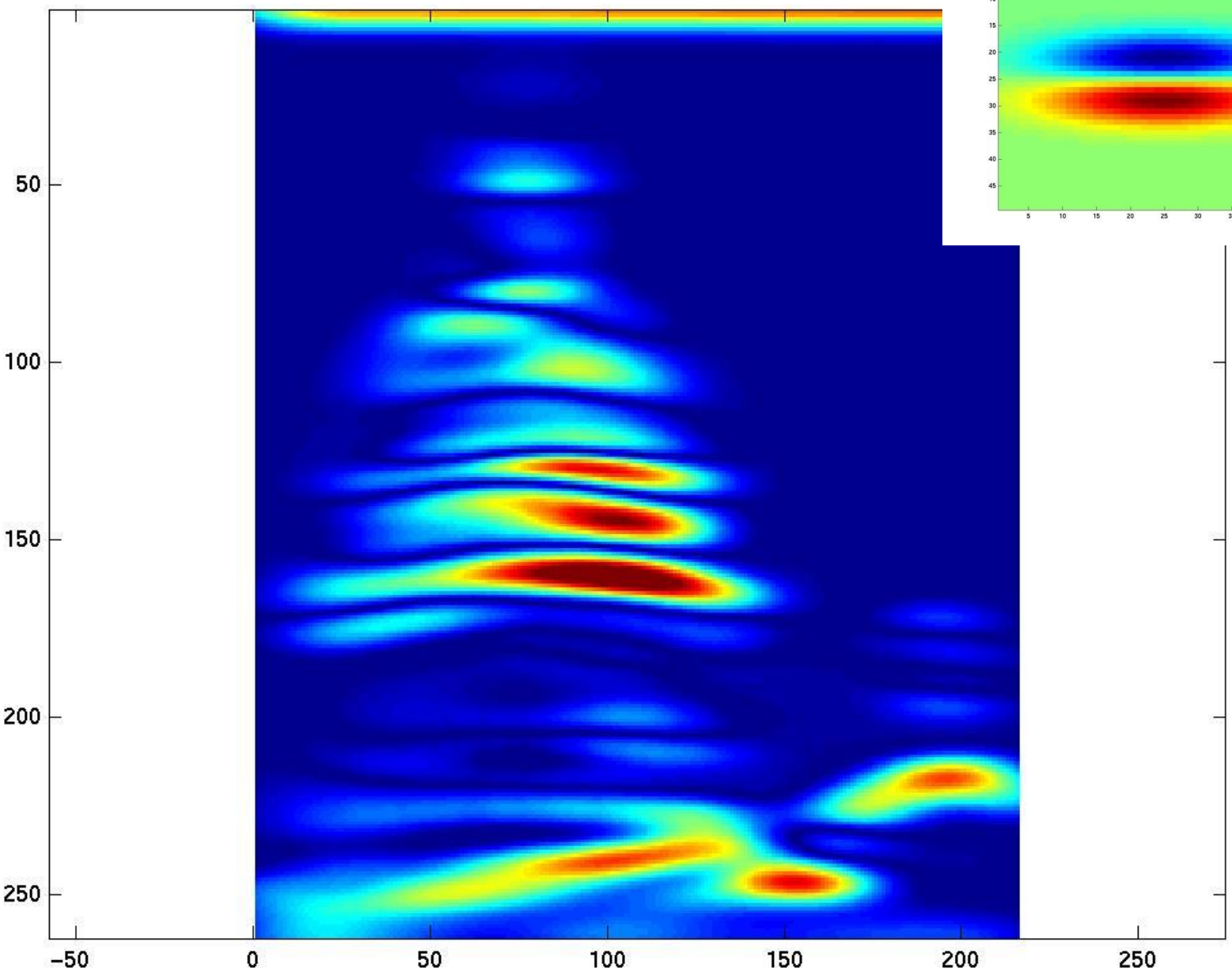

50
100
150
200
250
−50
0
50
100
150
200
250


50
100
150
200
250
−50
0
50
100
150
200
250

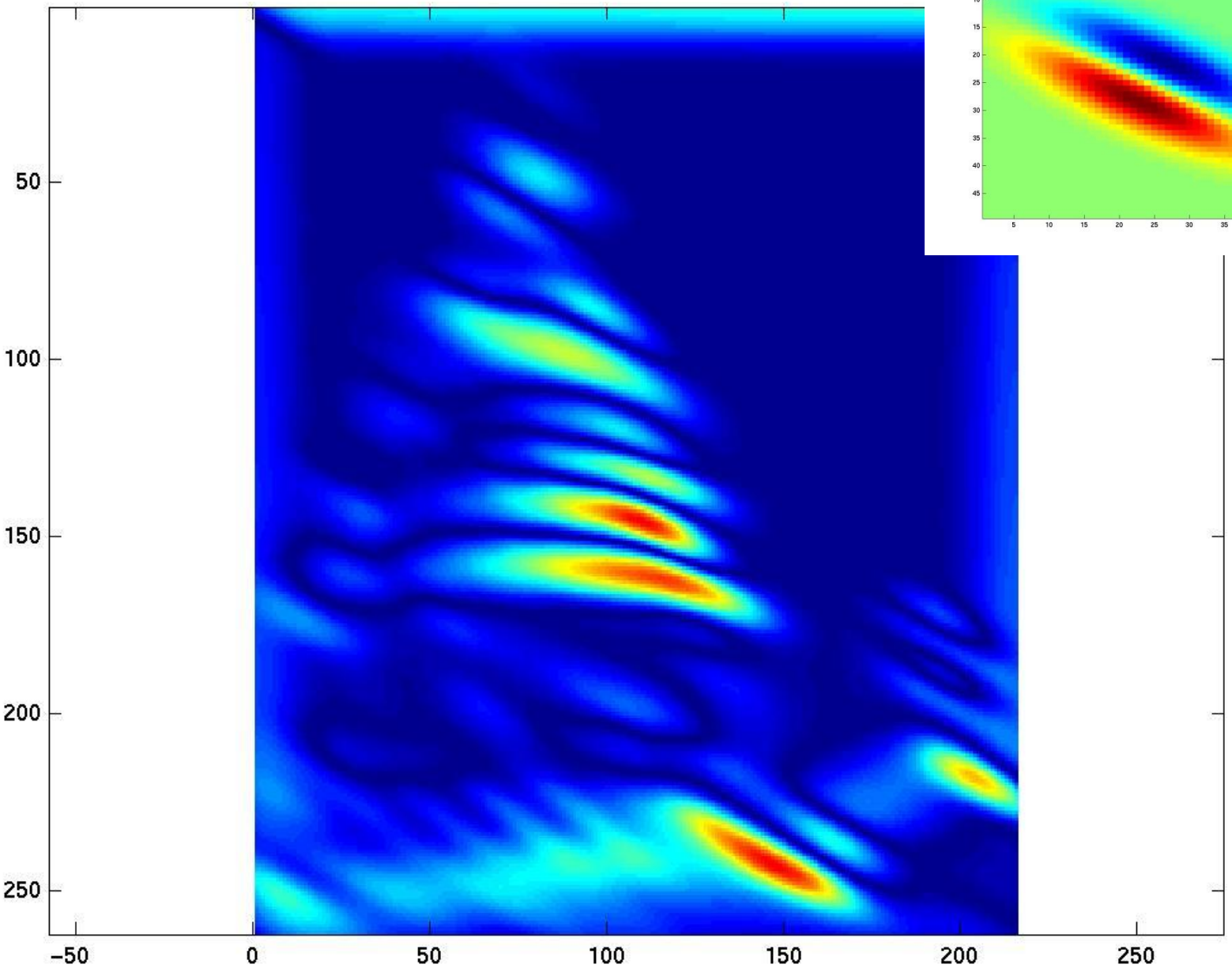

50
100
150
200
250
−50
0
50
100
150
200
250

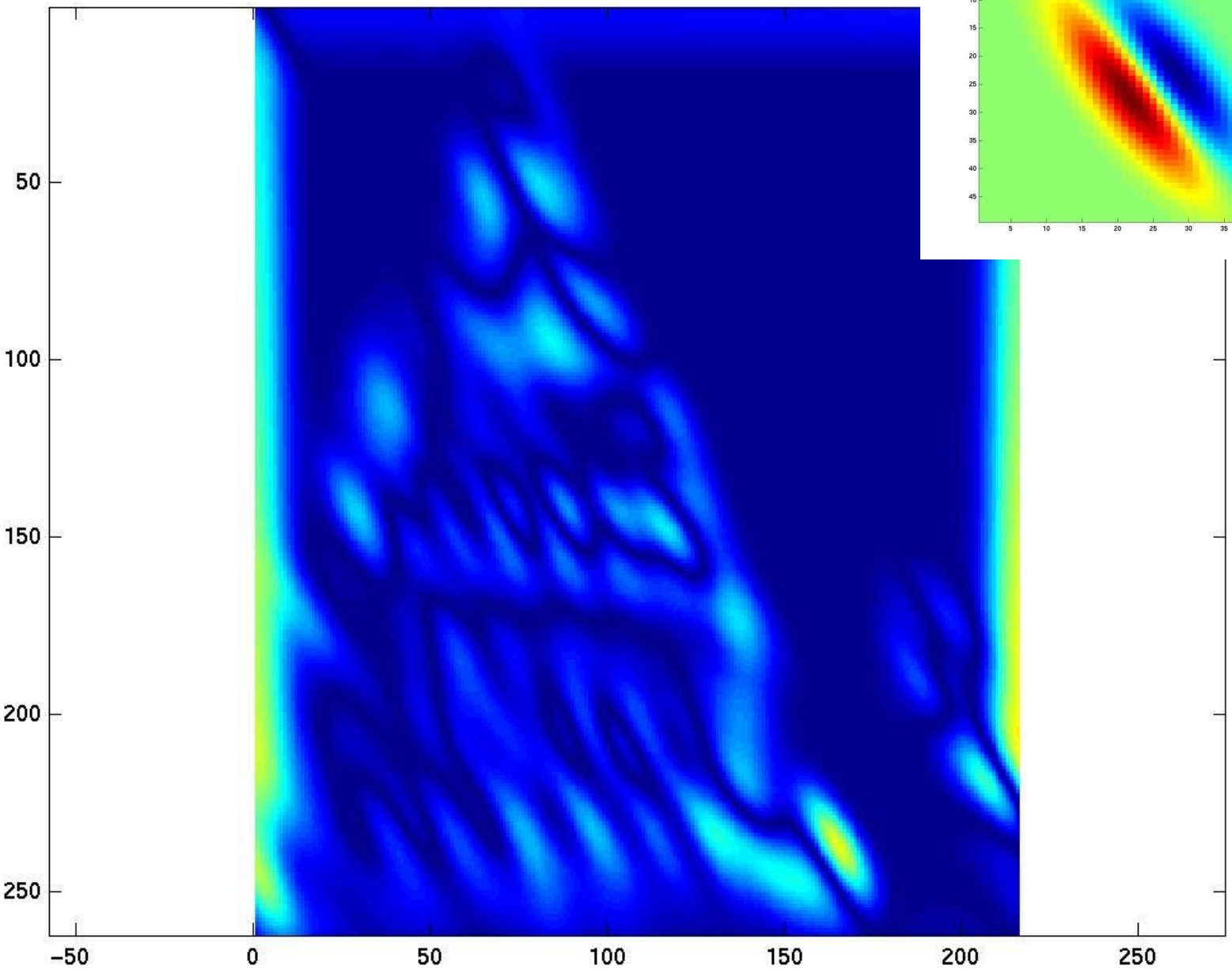
5
10
15
20
25
30
35
40
45

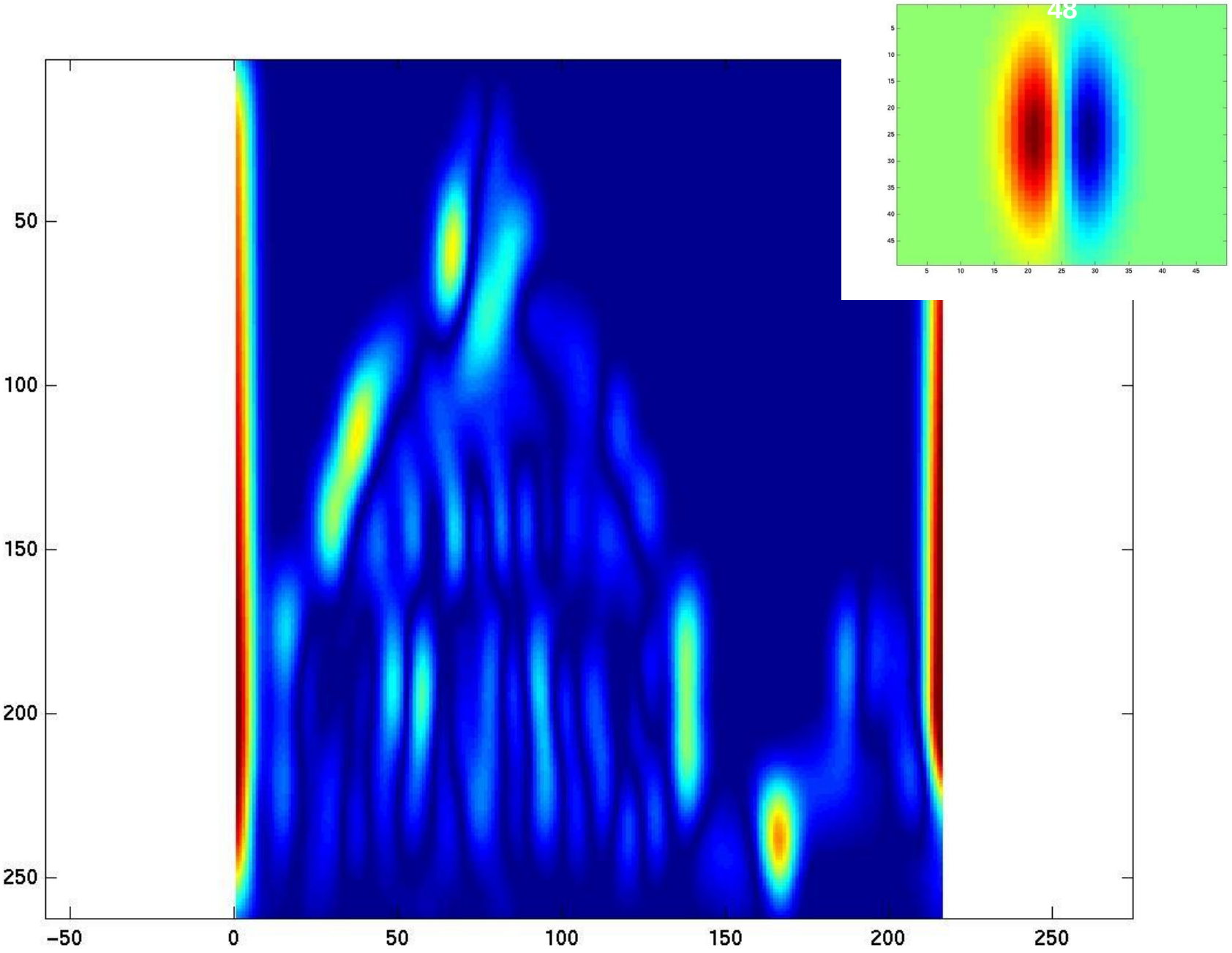

50
100
150
200
250
−50
0
50
100
150
200
250

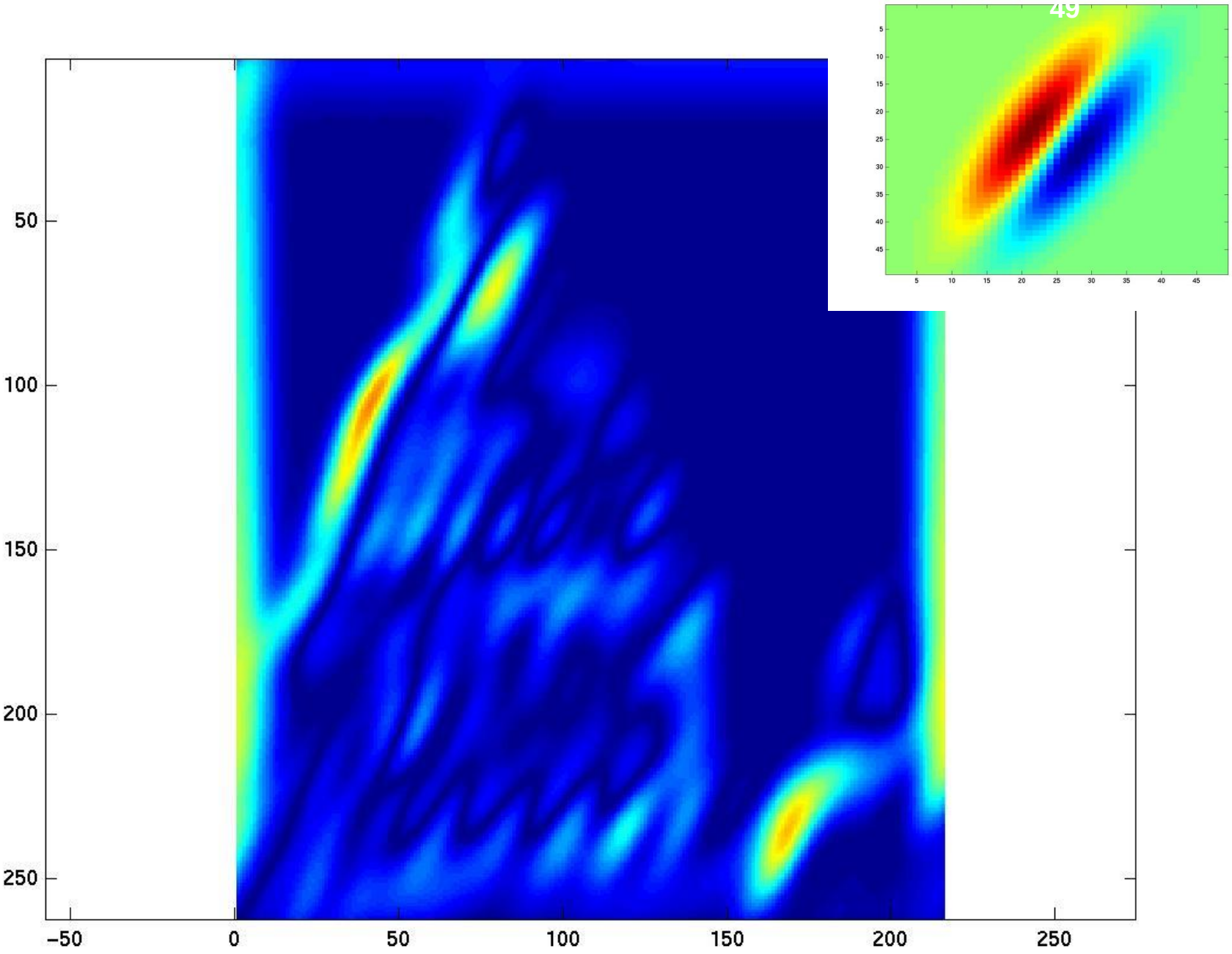

50
100
150
200
250
−50
0
50
100
150
200
250

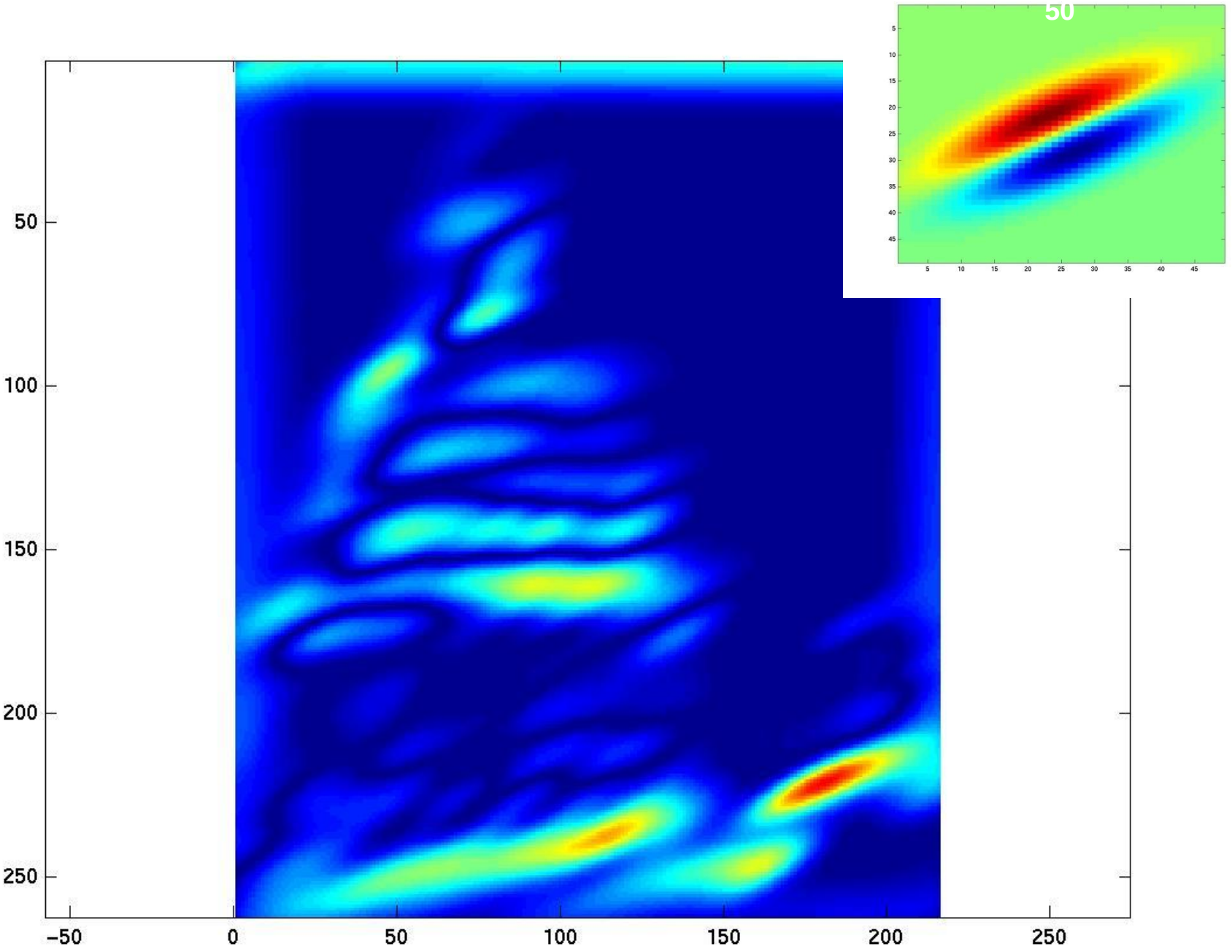
5
10
15
20
25
30
35
40
45

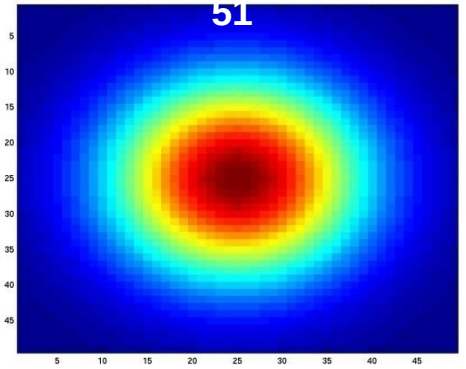

50
100
150
200
250
−50
0
50
100
150
200
250

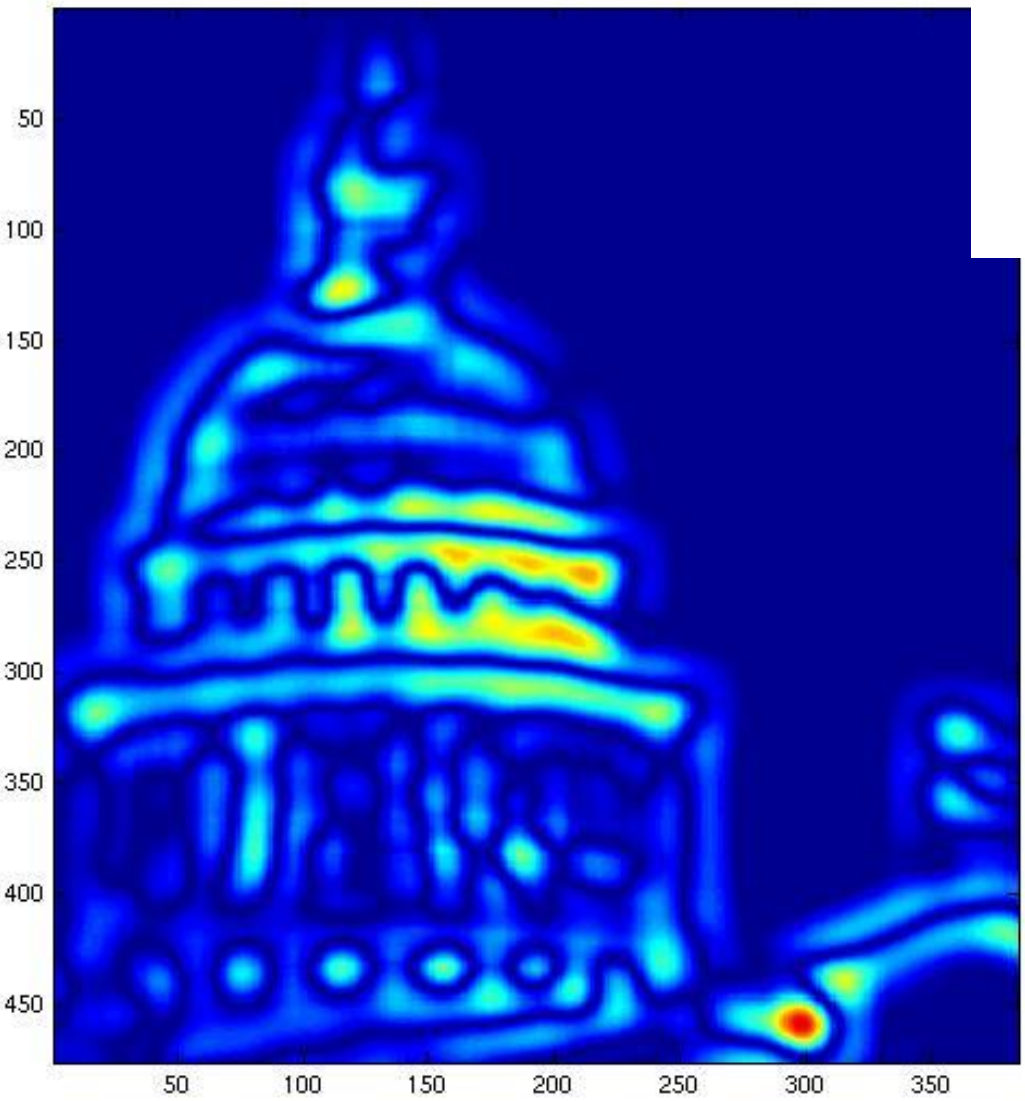

50
100
150
200
250
−50
0
50
100
150
200
250


50
100
150
200
250
−50
0
50
100
150
200
250

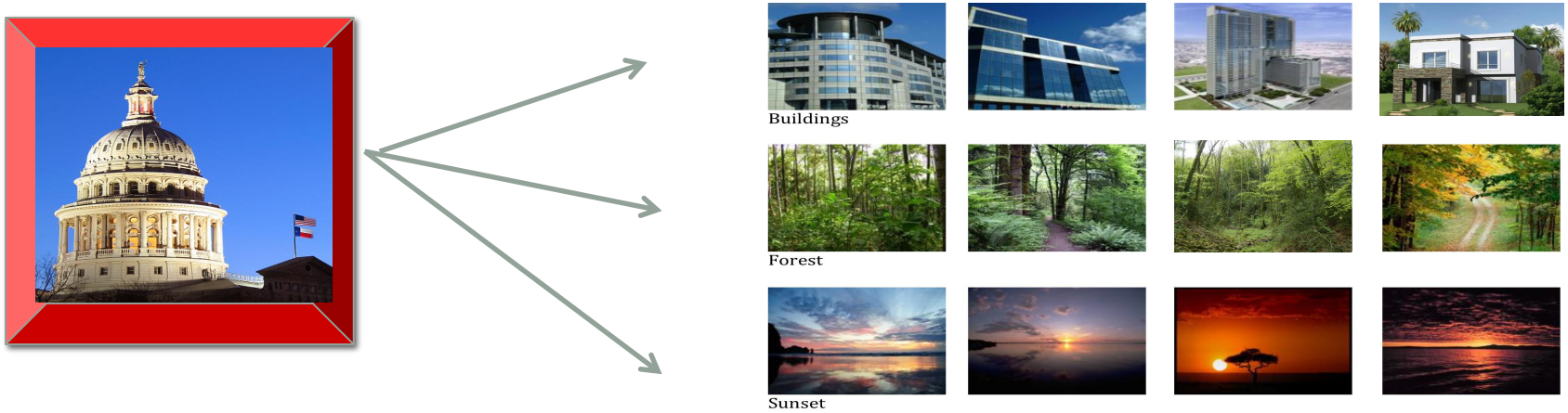

49
50
100
150
200
250
−50
0
50
100
150
200
250

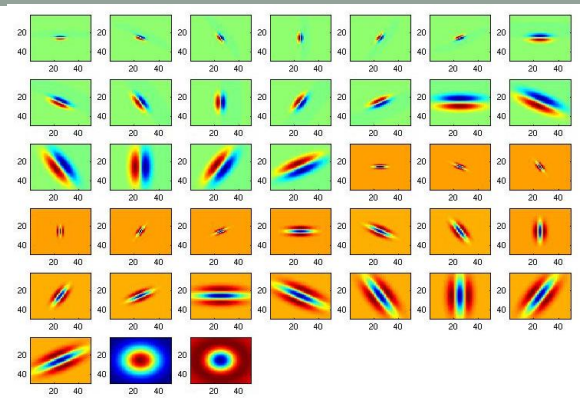

50
5
10
15
20
25
30
35
40
45
5
10
15
20
25
30
35
40
45
50
100
150
200
250
−50
0
50
100
150
200
250

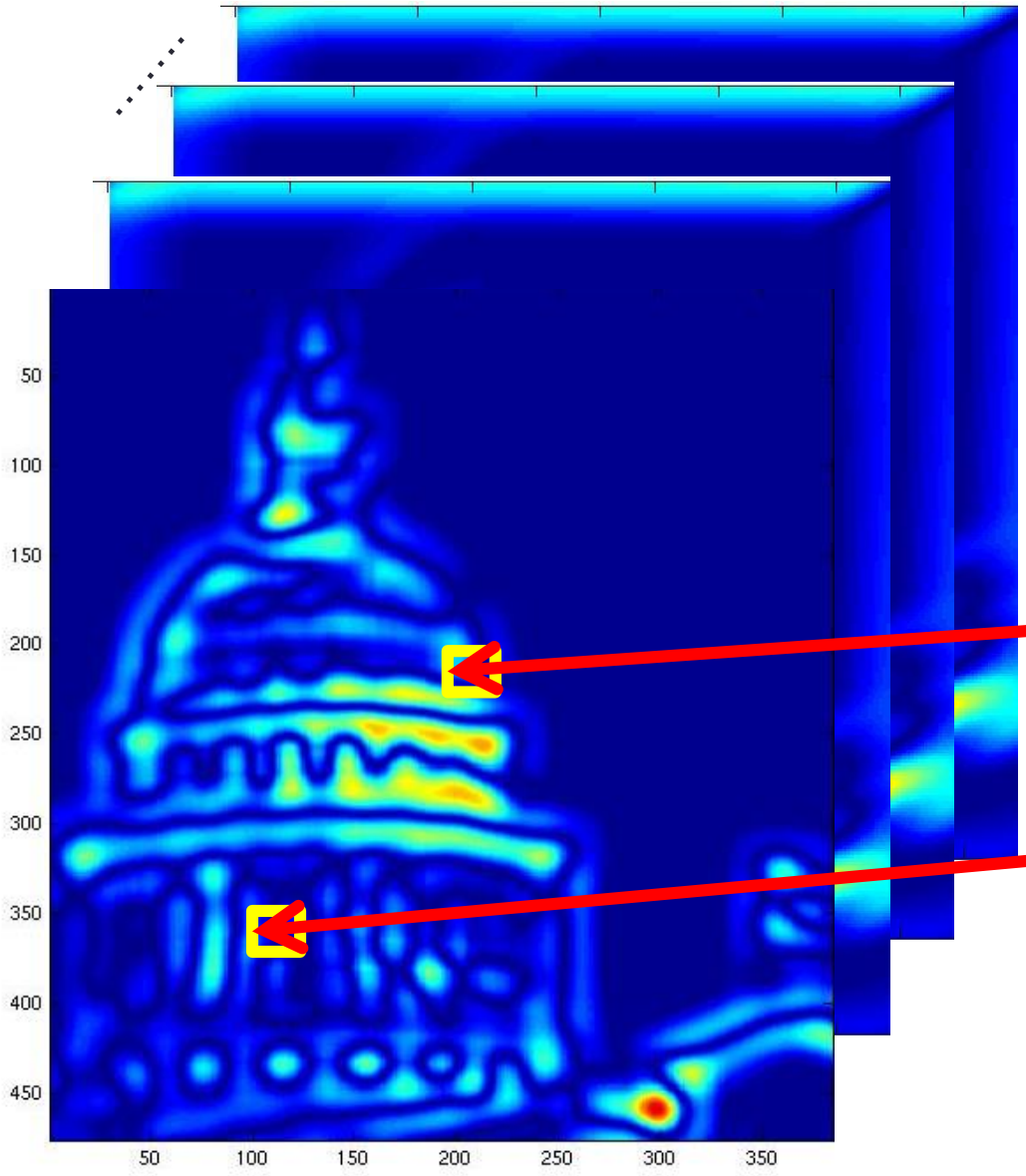

5
10
15
20
25
30
35
40
45
5 10 15 20 25 30 35 40 45


50
100
150
200
250
300
350
400
450
50 100 150 200 250 300 350

How to recover the image with the most similar texture?





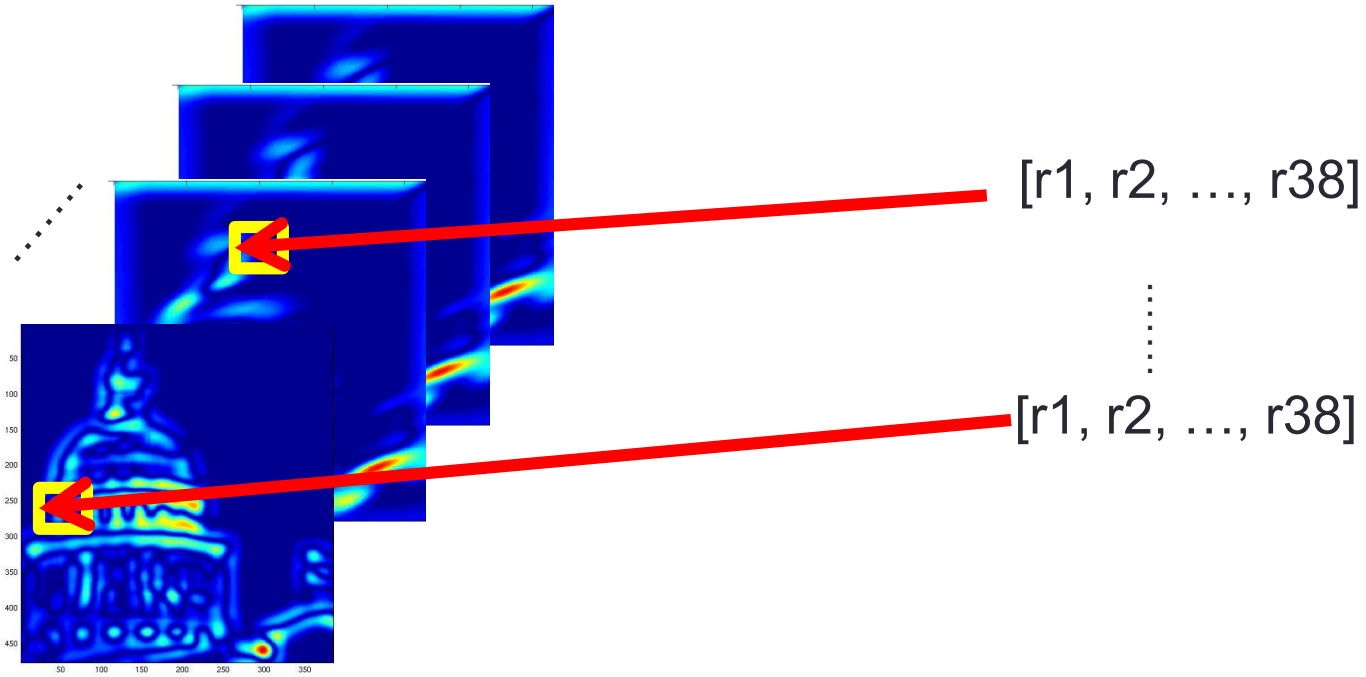


We can form a feature vector from the list of responses at each pixel.

[r1, r2, …, r38]

[r1, r2, …, r38]

How to go from pixel representation to image representation?



A simple way to represent the whole image is to get the mean abs value of each feature:

$$I \rightarrow f(I)=[mean_{\text{all pixels}}(|r1|), mean_{\text{all pixels}}(|r2|), \ldots, mean_{\text{all pixels}}(|r38|)]$$

Remember: Image retrieval based on HOG

Given an image (query), find all similar images in the database.

Image descriptor:

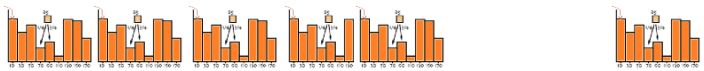
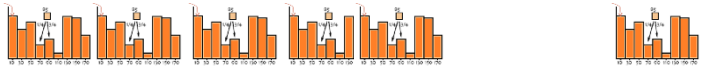


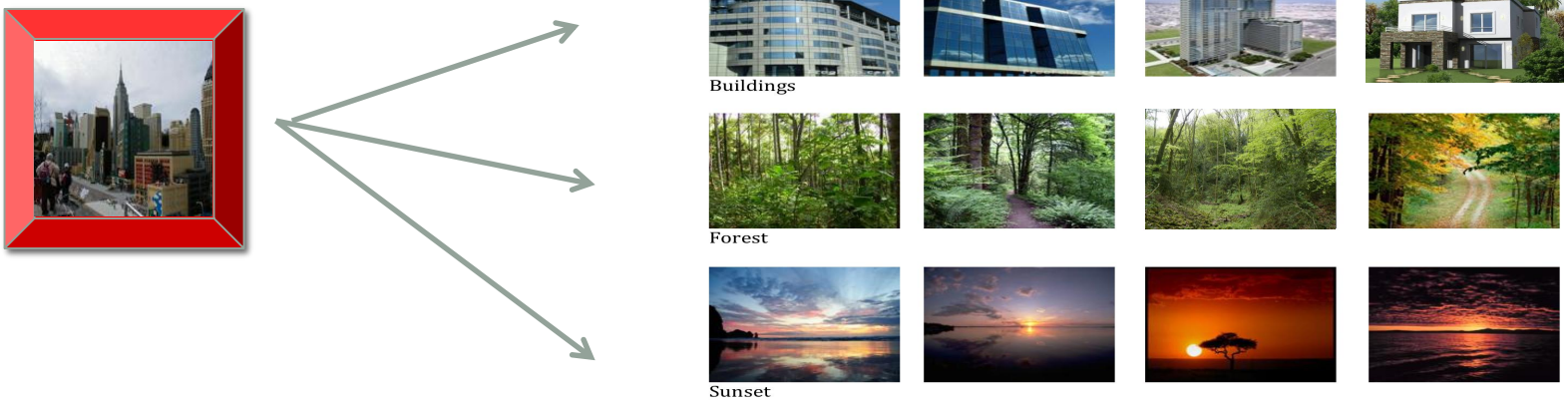
Image descriptor:



How to recover the image with the most similar texture?

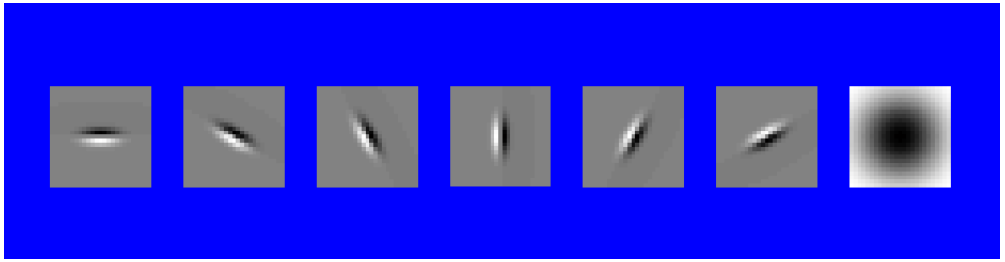
We are looking for the minimal Euclidean distance (L_2) between the feature vectors (f(I)) of the query image and the database images

$$D(f(I_1), f(I_2)) = \sqrt{\sum_{i=1}^{d} (f_i(I_1) - f_i(I_2))^2}$$

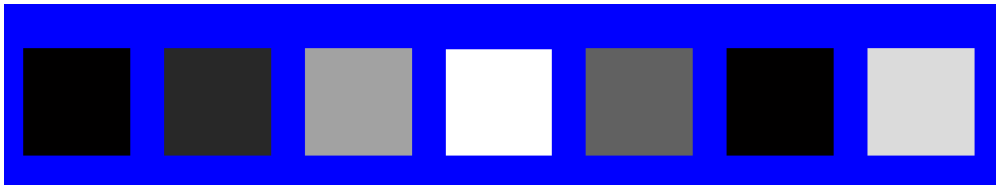


Exercise: Can you match the texture to the response?

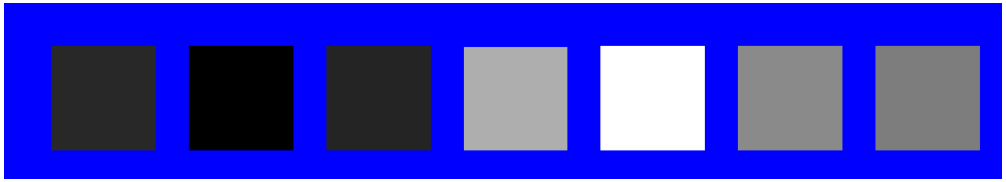
Filters



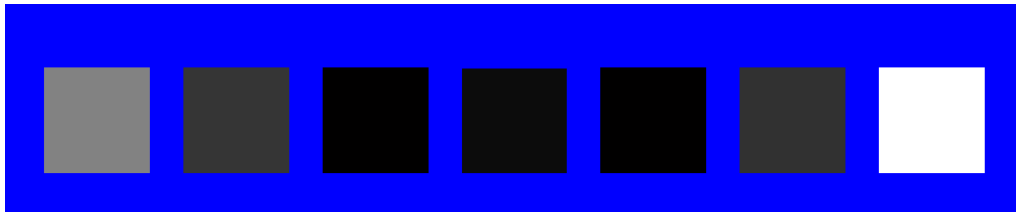
1



2

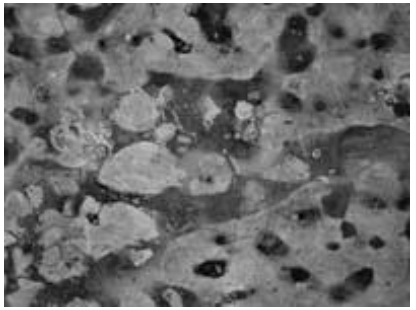


3



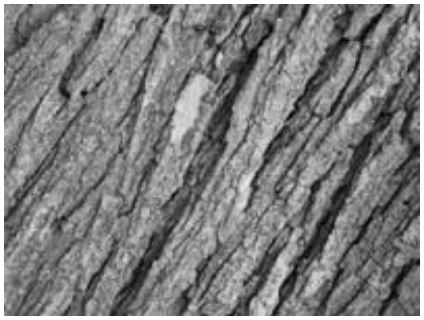
Mean abs responses

A



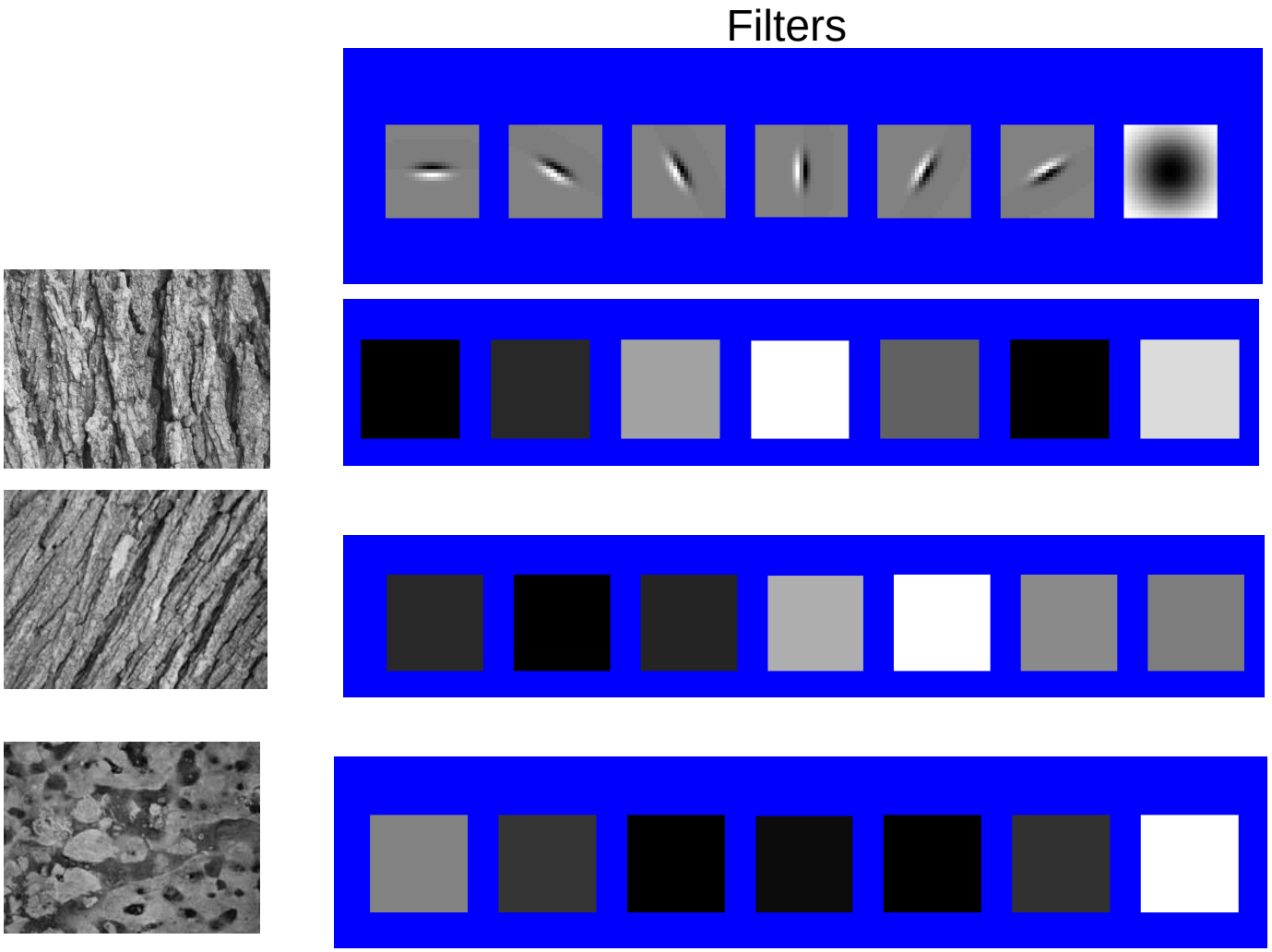
B

C



Derek Hoiem

Exercise: Representing texture by mean abs response



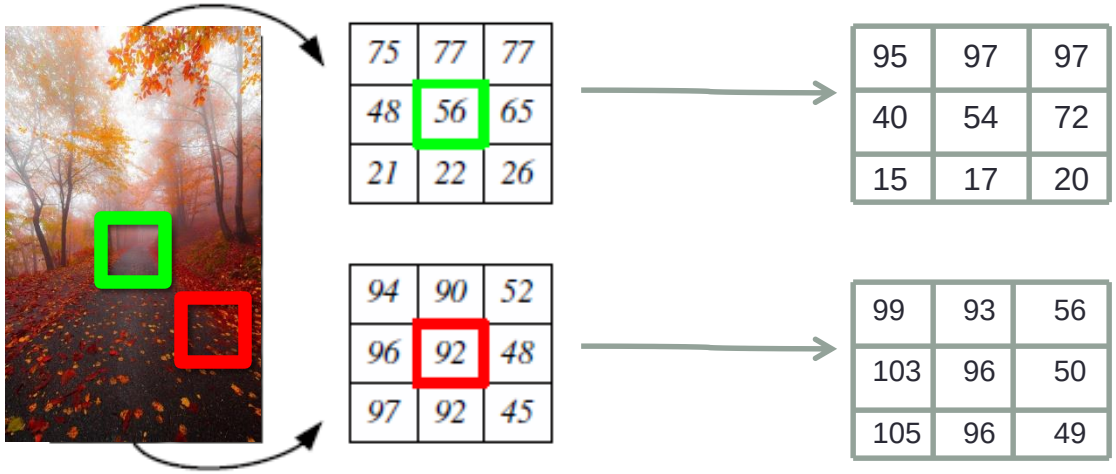
Is the Gaussian filter-based texture recognition contrast invariant?

Derek Hoiem

What is the alternative to recognition of textures?

Recognition of Local Binary Patterns textures - Local Binary Patterns (LBP)

What will happen with the filter convolutions if we change the image contrast?

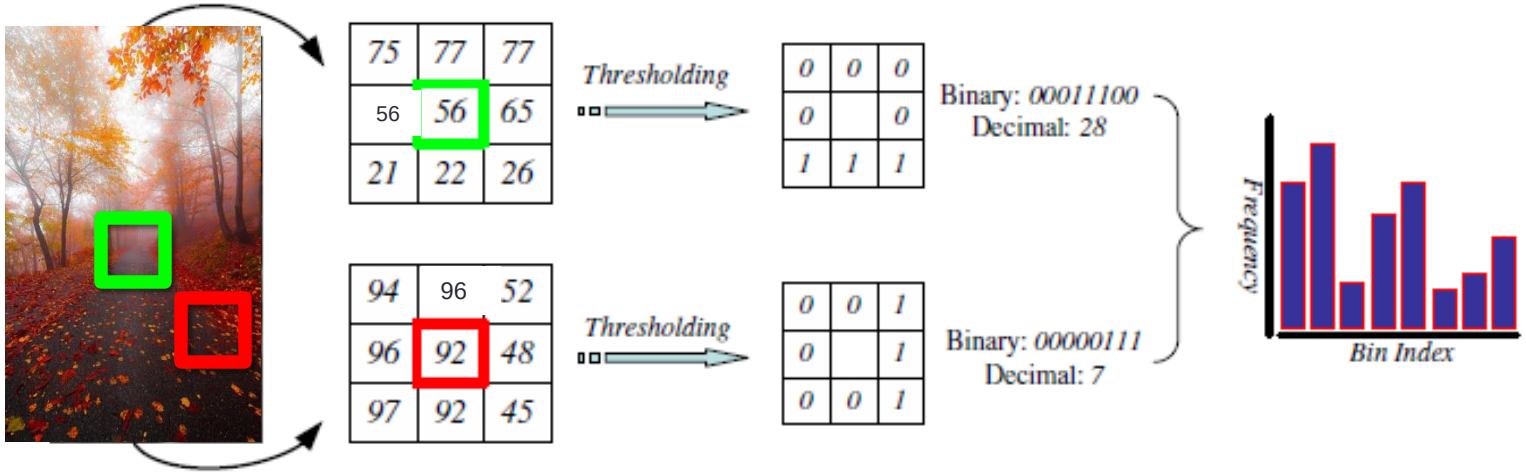


What is not changed?

Recognition of Local Binary Patterns textures (LBP)

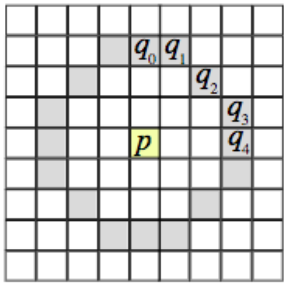
Df: The **LBP operator** assigns a label to each pixel of an image by comparing (for example, is it larger than the neighbor?) each neighbor with the value of the central pixel.

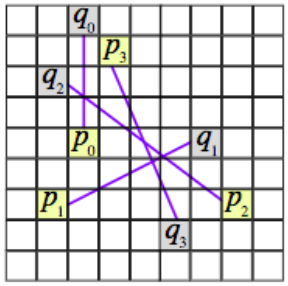
The result is considered as a binary number.



Note: If you use p = 8 neighbors, how many different codes can be generated?

Remember: ?!?! descriptor





$$\tau(p;x,y) = \begin{cases} 1 & \text{if } p(x) \geq p(y) \\ 0 & \text{otherwise} \end{cases}$$

$$\text{feature vector } f_n(p) := \sum_{1 \leq i \leq n} 2^{i-1} \ \tau(p;x_i,y_i)$$

$$\text{vector length } n = 256 \text{ bit}$$

Pixel location p and 256 pairs of pixel locations around p;

$f_n(p)=\tau(p_0,q_0)\cdot 2^0+ \tau(p_1,q_1)\cdot 2^1+..+ \tau(p_{255},q_{255})\cdot 2^{255}$

Which pairs provide best information for a given patch?

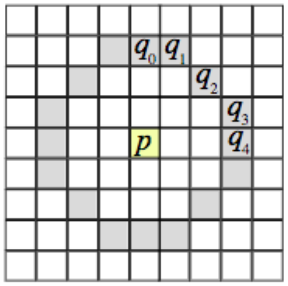
- maximize variance and thus minimize correlation.

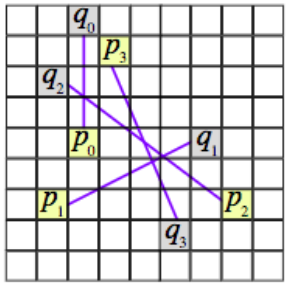
?!?!? learns which pairs to consider

- input: reference image series / output: static list of pairs

Rublee, Ethan, et al. "ORB: An efficient alternative to SIFT or SURF." *2011 International conference on computer vision*. Ieee, 2011.

Remember: ORB descriptor





$$\tau(p;x,y) = \begin{cases} 1 & \text{if } p(x) \geq p(y) \\ 0 & \text{otherwise} \end{cases}$$

$$\text{feature vector } f_n(p) := \sum_{1 \leq i \leq n} 2^{i-1} \ \tau(p;x_i,y_i)$$

$$\text{vector length } n = 256 \text{ bit}$$

Pixel location p and 256 pairs of pixel locations around p;

$$f_n(p)=\tau(p_0,q_0)\cdot 2^0+ \tau(p_1,q_1)\cdot 2^1+..+ \tau(p_{255},q_{255})\cdot 2^{255}$$

Which pairs provide best information for a given patch?

- maximize variance and thus minimize correlation.

ORB learns which pairs to consider

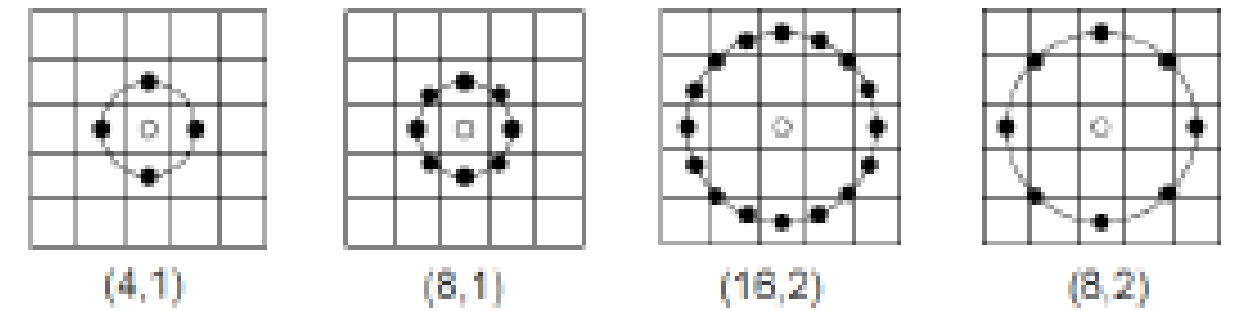
- input: reference image series / output: static list of pairs

Rublee, Ethan, et al. "ORB: An efficient alternative to SIFT or SURF." *2011 International conference on computer vision*. Ieee, 2011.

Extension of Local Binary Patterns

To deal with textures at different scales, LBP operator can be extended to use neighborhoods of different sizes.

- Circular neighborhoods and bilinear interpolation of pixel values -> any radius and number of samples of the neighborhood.



Note: If you use p = 16 neighbors, how many different codes can be generated?

Do we need so much different codes?

Uniform LBP

Df: A **LBP is called uniform** iff there are at most two transitions on the binary loop function.

Circular Binary Pattern	# of bitwise transitions	Uniform pattern?
11111111	0	Yes
00001111	1	Yes
01110000	2	Yes
11001110	3	No
11001001	4	No

For example :

00000000 (0 transition), 11100011 (2 transitions) -> uniform
01010000 (4 transitions), 01110101 (5 transitions) -> no uniform

Each uniform pattern -> its proper bin of the histogram.

All transitions are accumulated in a non-uniform bin (eg bin 0).

Uniform LBP

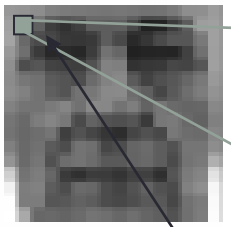
Considering P = 8, we get eight patterns (bits), all possible values are 256 (i.e. 2^8), but only 58 are uniform.

We construct a histogram for the 58 bins + one bin accumulating all non-uniform patterns.

A histogram of 59 bins represents 76.95% of the reduction of the image feature vector (256 bins histogram).

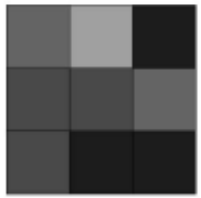
Uniform Label (decimal)	Uniform Label (binary)	Number of Transitions	$LBP^{u2}_{(8,R)}$ histogram bin
non uniform patterns	non uniform patterns	>2	0
0	00000000	0	1
1	00000001	1	2
2	00000010	2	3
3	00000011	1	4
4	00000100	2	5
⋮	⋮	⋮	⋮
251	11111011	2	54
252	11111100	1	55
253	11111101	2	56
254	11111110	1	57
255	11111111	0	58

Uniform LBP- Algorithm



16x16

Original image



Sub-Image

85	99	21
54	54	62
57	12	13

Sub-Image

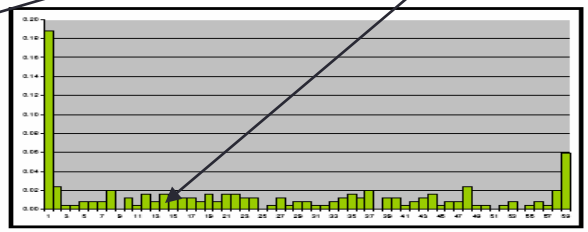
0	0	1
0		0
0	1	1

01011000

LBP index $(13)_{10}$

Repeat for each pixel and store in the histogram

14x14



LPB texture descriptor

[r1, r2, …, r59]

Properties of LBP vs. Gaussian filters

Pros
- Simple theory
- Computational simplicity
- Fast
- Compact
- Invariant to image illumination changes

Cons
- Can be too simple sometimes
- Which neighbours to consider to be decided.

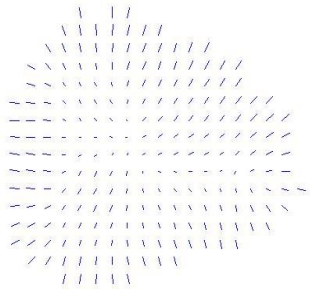
Pros
- Able to capture wide scope of textures
- Computational simplicity
- Compact
- No parameters

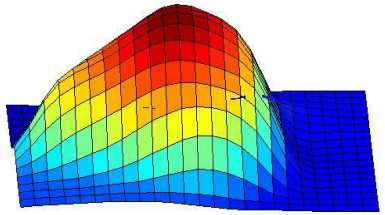
Cons
- Non-Invariant to image illumination changes

Example uses of texture in Computer vision (optional)

Shape from texture

- Use deformation of texture from point to point to estimate surface shape





Pics from A. Loh: http://www.csse.uwa.edu.au/~angie/phdpics1.html

Textures and Orientation

The geometric variations of a surface produce three effects on the elements of texture:

- Change in the density of the elements.
- Foreshortening, or change in size due to perspective.
- Rescaling.

The foreshortening is only caused by the orientation of the surface!!!.

Idea: The foreshortening can be characterized by changes in the local distribution of the contour orientation of the textons.

Classifying materials, “stuff”

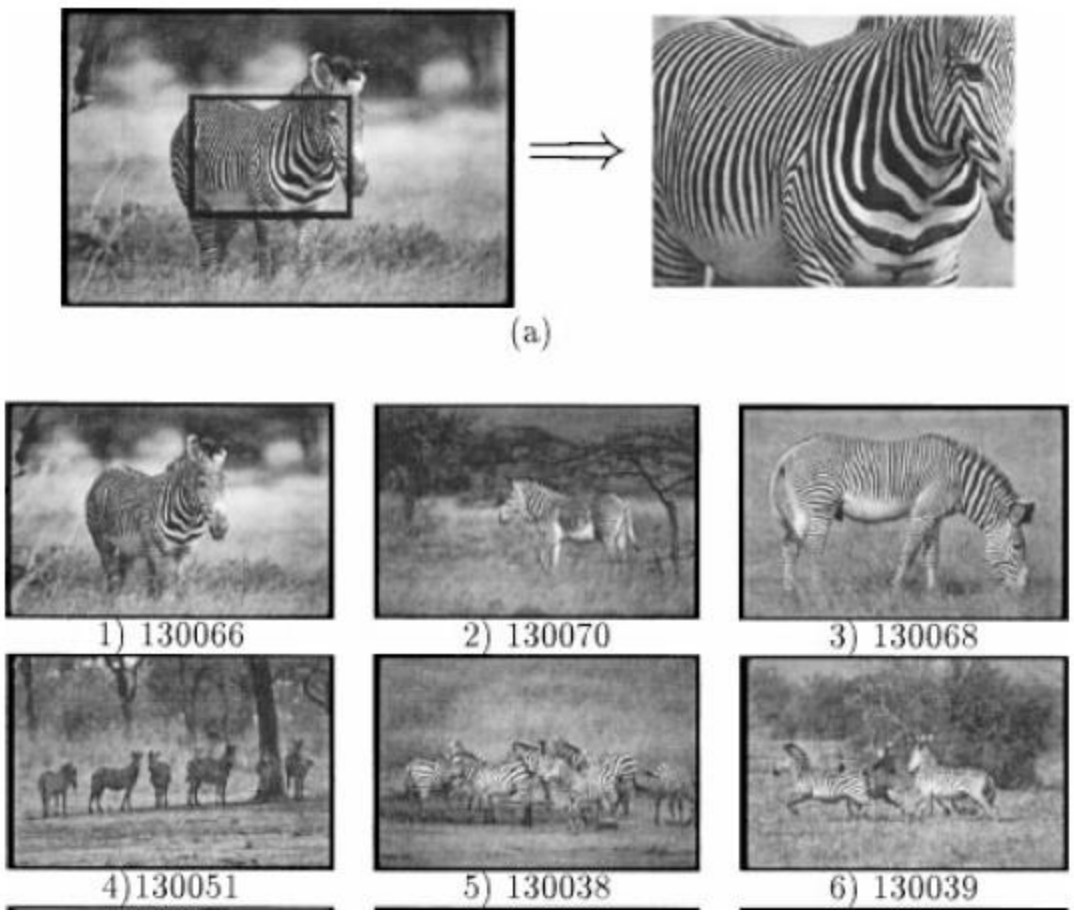
Figure by Varma & Zisserman

Scene characterization

Characterizing scene categories by texture

L. W. Renninger and J. Malik. When is scene identification just texture recognition? Vision Research 44 (2004) 2301–2311

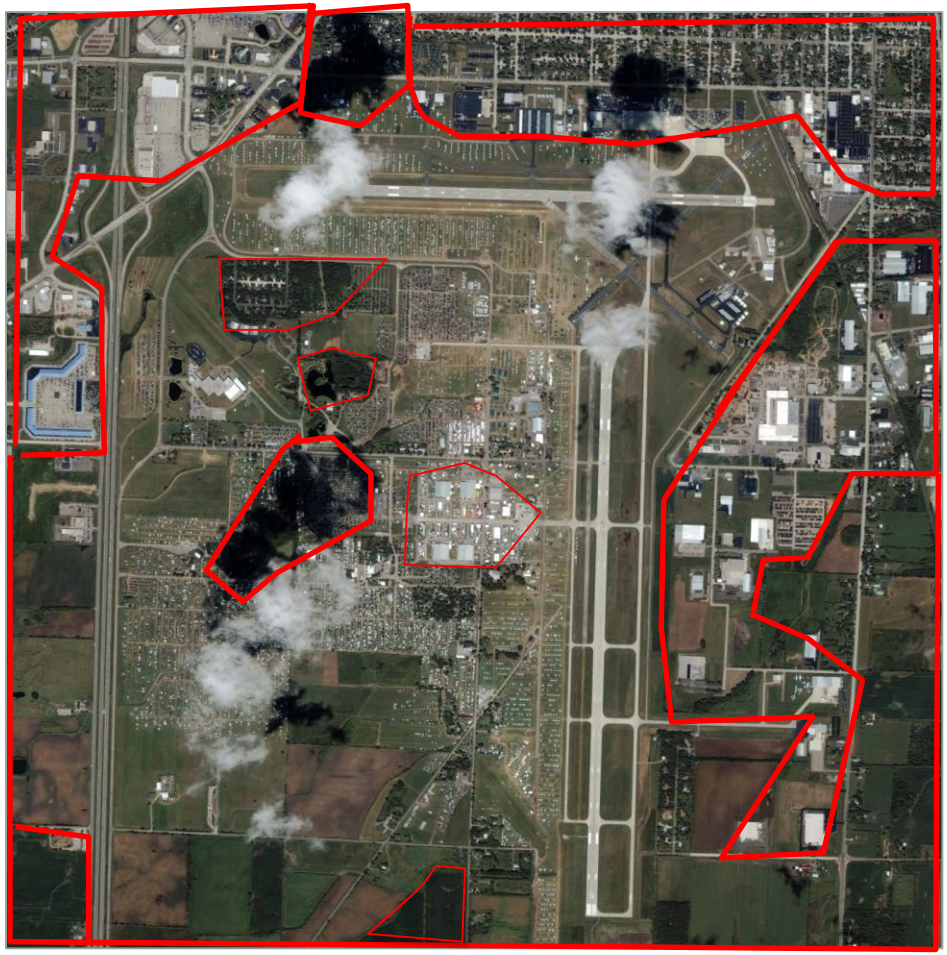
Image retrieval



Texture features for image retrieval

Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99-121, November 2000,

Image segmentation



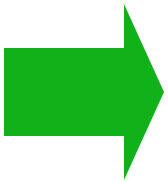
Segmenting aerial imagery by textures

Kristen Grauman

http://www.airventure.org/2004/gallery/images/073104_satellite.jpg

Texture synthesis

- Goal: create new samples of a given texture
- Many applications: virtual environments, hole-filling, texturing surfaces



Synthesizing textures for 3D models.

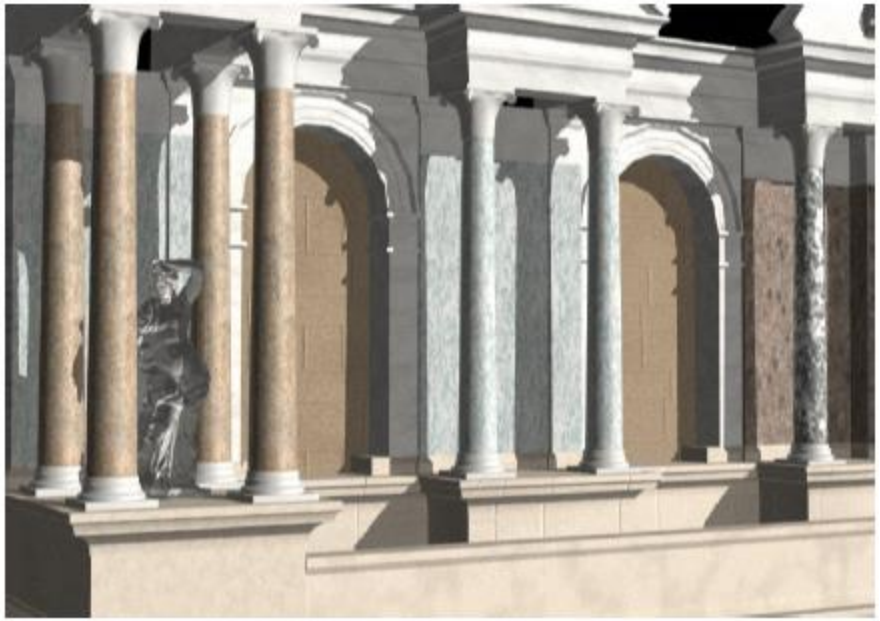


Figure 12. The Nymphaeum at the upper agora of Sagalassos with differently textured pillars. Overview of one half of the building (symmetric)

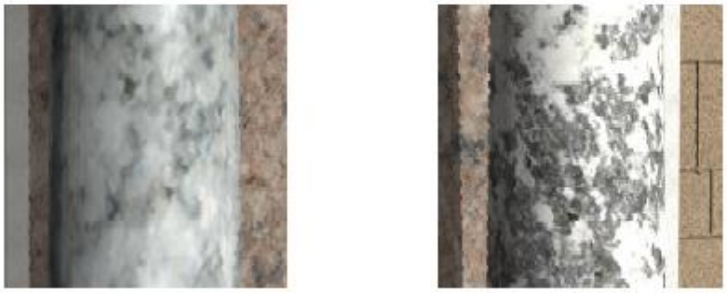


Figure 14. Nymphaeum pillars and back wall fragments in detail

Summary

- **Texture** is a useful property that is often indicative of materials, appearance cues.
- **Texture representations** attempt to summarize repeating patterns of local structure
- **Filter banks** useful to measure redundant variety of structures in local neighborhood
  - Feature spaces can be multi-dimensional
- **LBP** are fast, compact and illumination invariant texture descriptors.
- A lot of **texture applications** to other CV problems and real applications.

Test – Determine true or false

- Image descriptors are designed to solve Computer Vision problems like image retrieval or object classification. ✓
- The bank of filters based on Derivatives of Gaussians (DoG) are image descriptors that measure texture presence in the images ✓
- The bank of filters DoG are image descriptors similar to HOG image descriptors. The difference is that HOG measure the statistics of textured regions while DoG describe configurations of structures detected by the image gradient. ✗
- LBP similar to DoG are invariant to orientation but not to illumination change. ✗
- The dimensionality of the DoGs depends on the scale, orientation and shape captured by the derivatives. ✓
- The dimensionality of the LBP depends on the neighbourhood considered and if they are uniform or not. ✓

Test –

Problems/Image descriptors	DoG	LBP	HOG	SIFT
Image smoothing				
Edge detection	✓			
Image retrieval	✓	✓	✓	✓
Scene classification	✓	✓	✓	✓
Object detection			✓	✓
Object recognition			✓	✓
Segmentation	✓	✓		
Texture synthesis	✓	✓		
Shape from texture	✓	✓		
Orientation extraction	✓	✓		