Segmentation and Grouping

CS 600.361/600.461

Instructor: Greg Hager
Based on slides from Nicolas Padoy

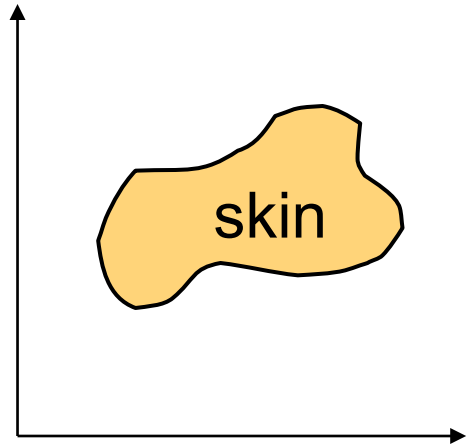
(Adapted from slides by Rick Szeliski and Kristen Grauman)

Outline

- Reminders
- Segmentation and grouping
  - K-means clustering
  - Features
  - Textures
- CV projects

Reminders

Skin detection



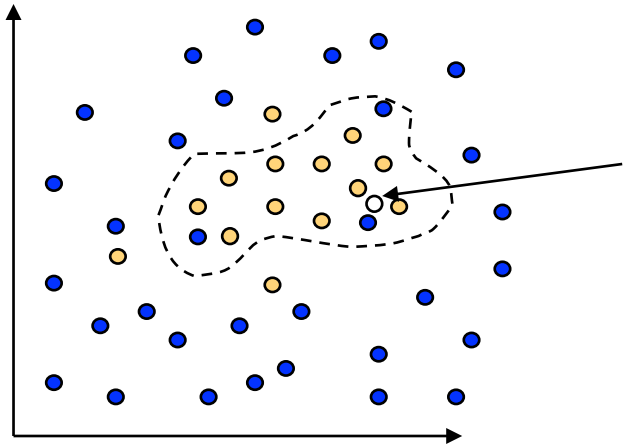
Skin pixels have a distinctive range of colors

- Corresponds to region(s) in RGB color space

Skin classifier

- A pixel X = (R,G,B) is skin if it is in the skin (color) region
- How to find this region?

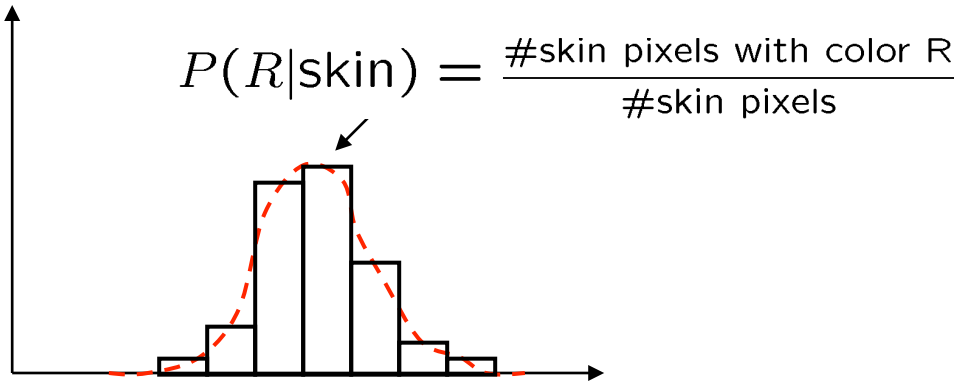
Skin classifier



Given X = (R,G,B): how to determine if it is skin or not?

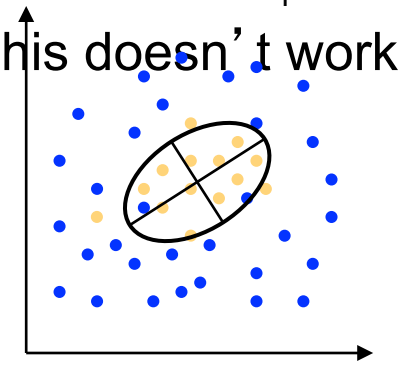
- Nearest neighbor
  - find labeled pixel closest to X
- Find plane/curve that separates the two classes
  - popular approach: Support Vector Machines (SVM)
- Data modeling
  - fit a probability density/distribution model to each class

Learning conditional PDF's



We can calculate P(R | skin) from a set of training images

- It is simply a histogram over the pixels in the training images
  - each bin R_i contains the proportion of skin pixels with color R_i
- This doesn't work as well in higher-dimensional spaces. Why not?



Approach: fit parametric PDF functions

- common choice is rotated Gaussian
  - center $\mathbf{c} = \overline{X}$
  - covariance $\sum_X (X - \overline{X})(X - \overline{X})^T$

Bayes rule

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)}$$

In terms of our problem:

what we measure (**likelihood**)

domain knowledge (**prior**)

$$P(\text{skin}|R) = \frac{P(R|\text{skin})\ P(\text{skin})}{P(R)}$$

what we want (**posterior**)

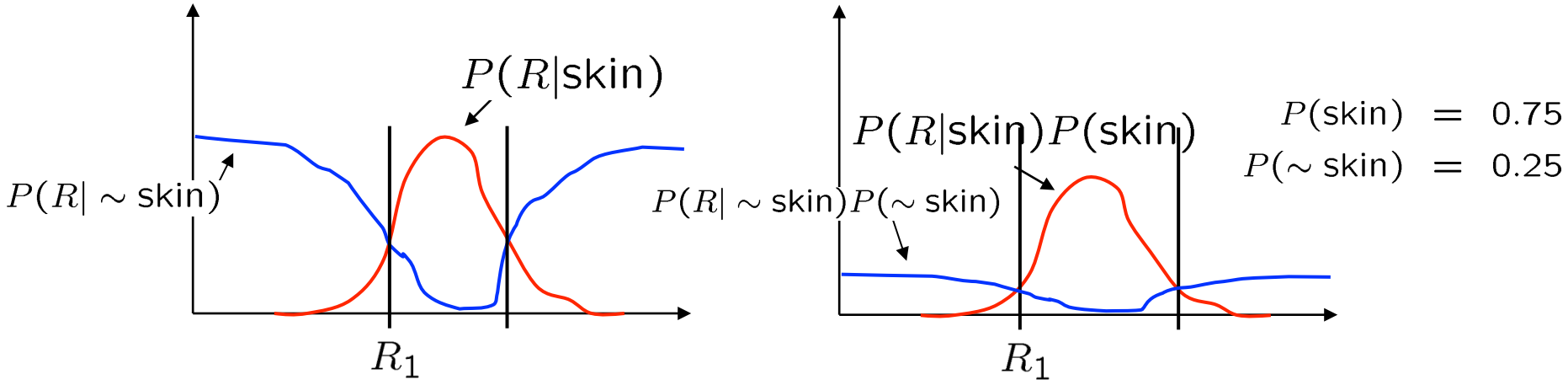
normalization term

$$P(R) = P(R|\text{skin})P(\text{skin}) + P(R|\sim\text{skin})P(\sim\text{skin})$$

What can we use for the prior P(skin)?

- Domain knowledge:
  - P(skin) may be larger if we know the image contains a person
  - For a portrait, P(skin) may be higher for pixels in the center
- Learn the prior from the training set. How?
  - P(skin) is proportion of skin pixels in training set

Bayesian estimation



likelihood

posterior (unnormalized)

Bayesian estimation

- Goal is to choose the label (skin or ~skin) that maximizes the posterior ↔ minimizes probability of misclassification
  - this is called **Maximum A Posteriori (MAP) estimation**

Eigenfaces for recognition

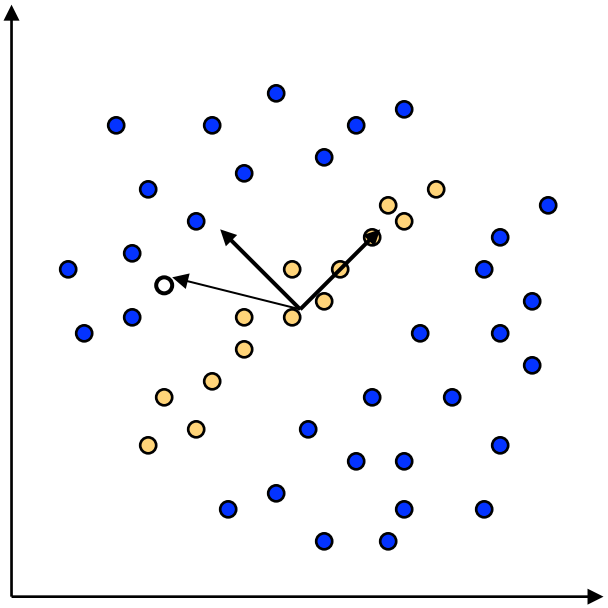
Matthew Turk and Alex Pentland

J. Cognitive Neuroscience

1991

(Adapted from slides by Rick Szeliski)

Linear subspaces



Consider the variation along direction **v** among all of the orange points:

$$var(\mathbf{v}) = \sum_{\text{orange point } \mathbf{x}} \|(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}} \cdot \mathbf{v}\|^2$$

What unit vector **v** minimizes *var*?

$$\mathbf{v}_2 = min_{\mathbf{v}} \{var(\mathbf{v})\}$$

What unit vector **v** maximizes *var*?

$$\mathbf{v}_1 = max_{\mathbf{v}} \{var(\mathbf{v})\}$$

$$\begin{aligned} var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}} \cdot \mathbf{v}\| \\ &= \sum_{\mathbf{x}} \mathbf{v}^{\mathrm{T}} (\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}} \mathbf{v} \\ &= \mathbf{v}^{\mathrm{T}} \left[\sum_{\mathbf{x}} (\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}} \right] \mathbf{v} \\ &= \mathbf{v}^{\mathrm{T}} \mathbf{A} \mathbf{v} \quad \text{where } \mathbf{A} = \sum_{\mathbf{x}} (\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathrm{T}} \end{aligned}$$

Solution: $\mathbf{v}_1$ is eigenvector of **A** with *largest* eigenvalue
$\mathbf{v}_2$ is eigenvector of **A** with *smallest* eigenvalue

Principal component analysis

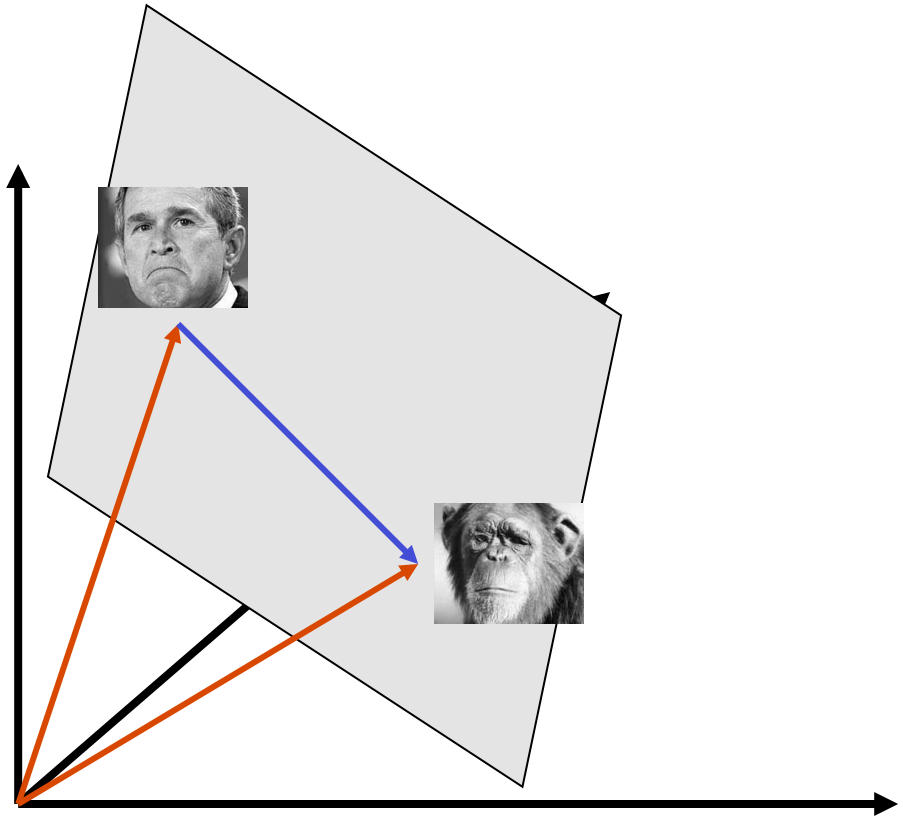
Suppose each data point is N-dimensional

- Same procedure applies:

$$\begin{aligned} var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x}-\overline{\mathbf{x}})^{\mathrm{T}} \cdot \mathbf{v}\| \\ &= \mathbf{v}^{\mathrm{T}}\mathbf{A}\mathbf{v} \quad \text{where } \mathbf{A} = \sum_{\mathbf{x}} (\mathbf{x}-\overline{\mathbf{x}})(\mathbf{x}-\overline{\mathbf{x}})^{\mathrm{T}} \end{aligned}$$

- The eigenvectors of **A** define a new coordinate system
  - eigenvector with largest eigenvalue captures the most variation among training vectors **x**
  - eigenvector with smallest eigenvalue has least variation
- We can compress the data using the top few eigenvectors
  - corresponds to choosing a "linear subspace"
    - » represent points on a line, plane, or "hyper-plane"
  - these eigenvectors are known as the ***principal components***

Dimensionality reduction



The set of faces is a “subspace” of the set of images

- We can find the best subspace using PCA
- This is like fitting a “hyper-plane” to the set of faces
  - spanned by vectors $\mathbf{v_1}$, $\mathbf{v_2}$, ..., $\mathbf{v_K}$
  - any face $\mathbf{x} \approx \bar{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_k\mathbf{v_k}$

Projecting onto the eigenfaces

The eigenfaces $\mathbf{v_1}$, ..., $\mathbf{v_K}$ span the space of faces

- A face is converted to eigenface coordinates by

$$\mathbf{x} \rightarrow (\underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v_1}}_{a_1}, \underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v_2}}_{a_2}, \ldots, \underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v_K}}_{a_K})$$

$$\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K}$$

Define $W = [\mathbf{v}_1\ \mathbf{v}_2 \cdots \mathbf{v}_k]$ then $\begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{pmatrix} = W^{\mathbf{T}}(x - \bar{x})$

Recognition with eigenfaces

Algorithm

1. Process the image database (set of images with labels)
   - Run PCA—compute eigenfaces
   - Calculate the K coefficients for each image
2. Given a new image (to be recognized) **x**, calculate K coefficients

$$\mathbf{x} \rightarrow (a_1, a_2, \ldots, a_K)$$

3. Detect if x is a face

$$\|\mathbf{x} - (\bar{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K})\| < \text{threshold}$$

4. If it is a face, who is it?
   - Find closest labeled face in database
     - » nearest-neighbor in K-dimensional space

Fischerfaces

Belhumeur, Hespanha, Kriegman

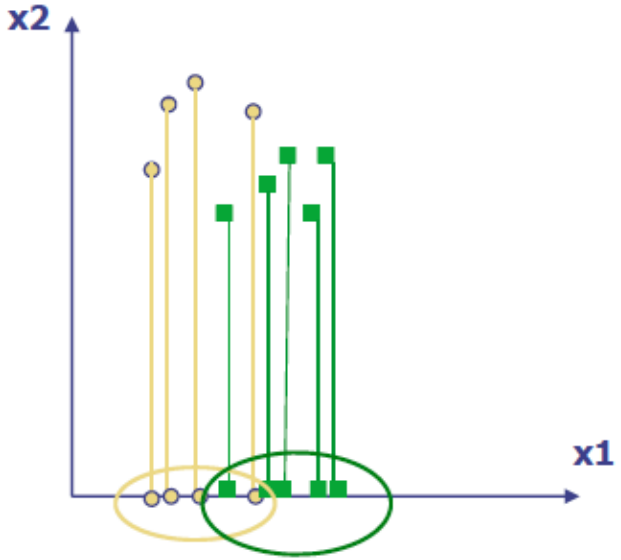
PAMI

1997

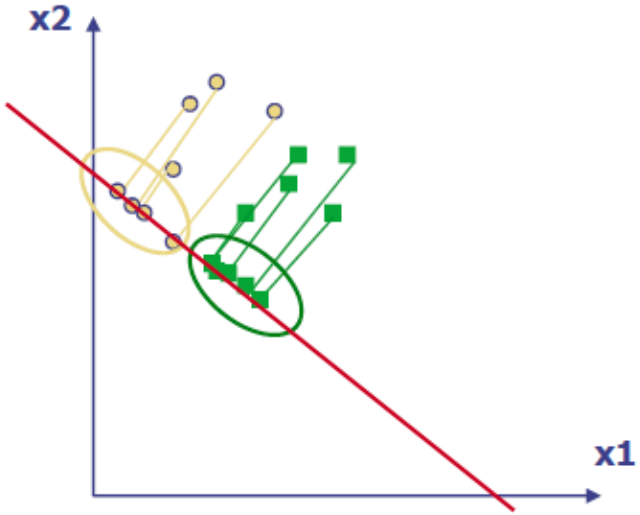
(Adapted from slides by Fei-fei Li)

Illustration of the projection

- Using two classes as example:



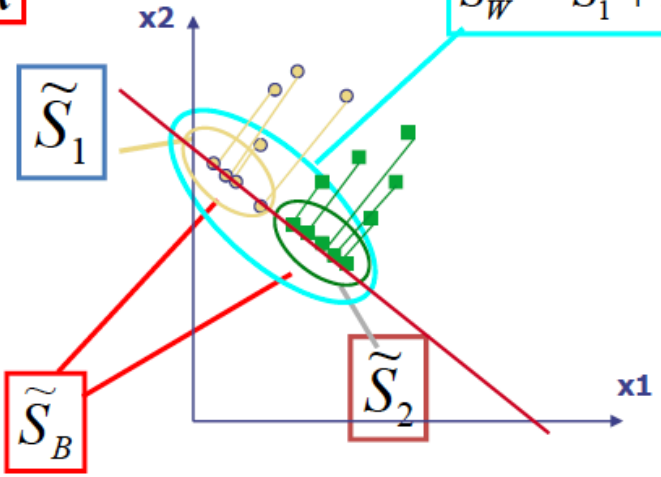
Poor Projection



Good

Illustration

$$y_k = W^T x_k$$



$$S_W = \sum_{i=1}^{c} S_i$$

$$S_B = \sum_{i=1}^{c} |\chi_i| (\mu_i - \mu)(\mu_i - \mu)^T$$

$$\tilde{S}_W = W^T S_W W$$

$$\tilde{S}_B = W^T S_B W$$

Mathematical formulation

- The desired projection:

$$W_{opt} = \arg\max_{\mathbf{W}} \frac{\left|\tilde{S}_B\right|}{\left|\tilde{S}_W\right|} = \arg\max_{\mathbf{W}} \frac{\left|W^T S_B W\right|}{\left|W^T S_W W\right|}$$

- How is it found ? → Generalized Eigenvectors

$$S_B w_i = \lambda_i S_W w_i \qquad i = 1, \ldots, m$$

- If S_w has full rank, the generalized eigenvectors are eigenvectors of $S_W^{-1} S_B$ with largest eigen-values

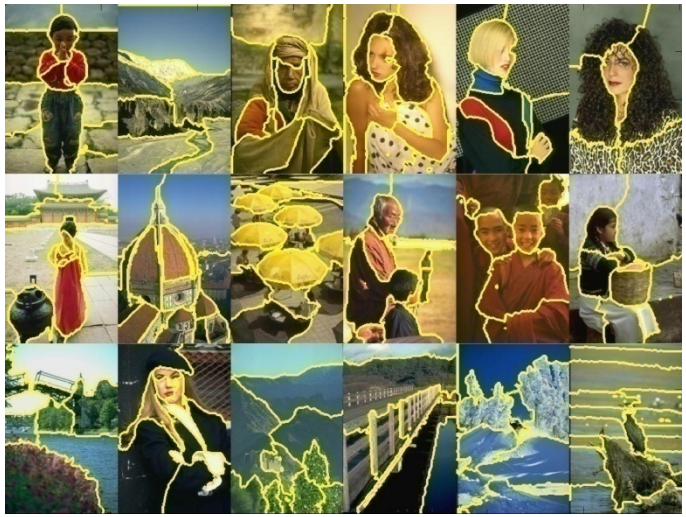
Segmentation and grouping

Grouping in vision

Goals:

- Gather features that belong together
- Obtain an intermediate representation that compactly describes key image or video parts

Examples of grouping in vision



[Figure by J. Shi]

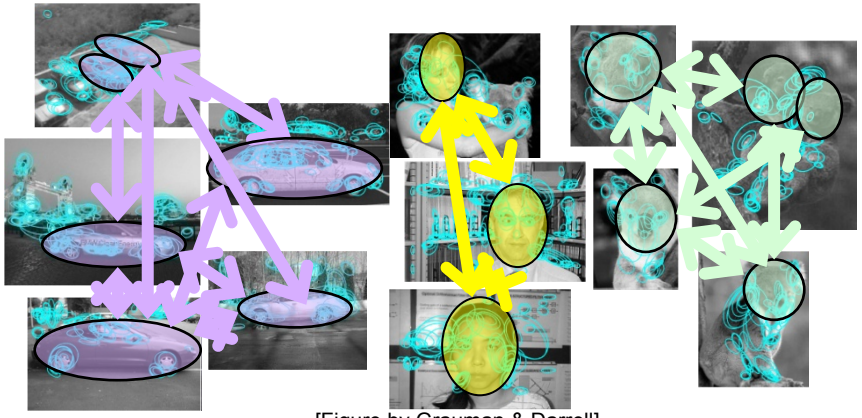
Determine image regions

[http://poseidon.csd.auth.gr/LAB_RESEARCH/Latest/imgs/SpeakDepVidIndex_img2.jpg]

Group video frames into shots

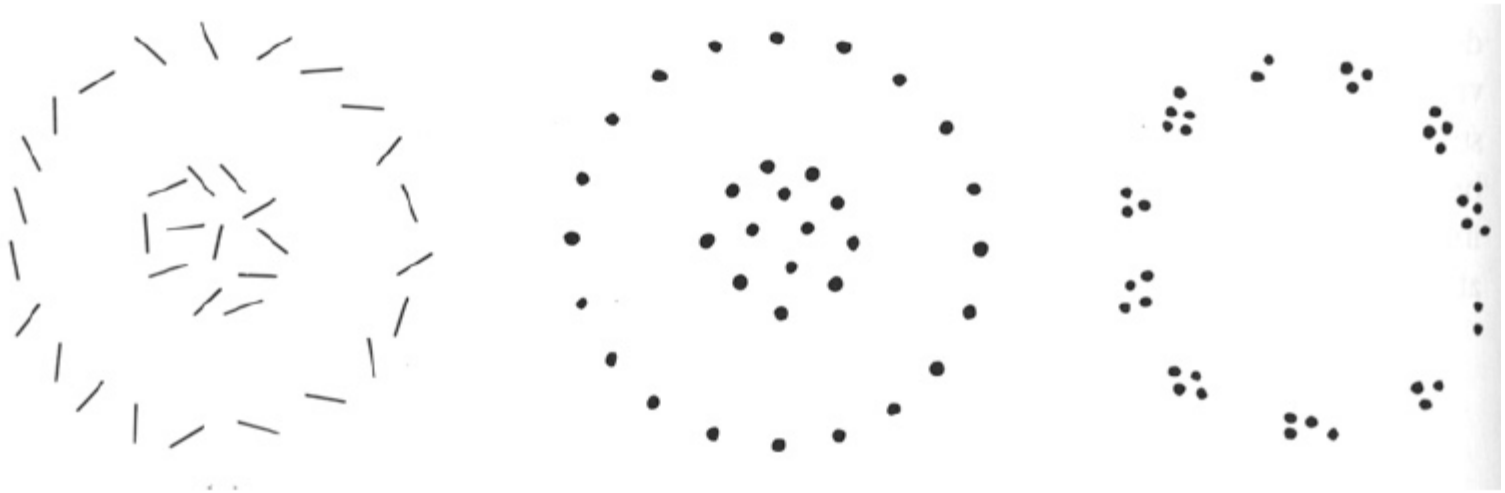
[Figure by Wang & Suter]

Figure-ground

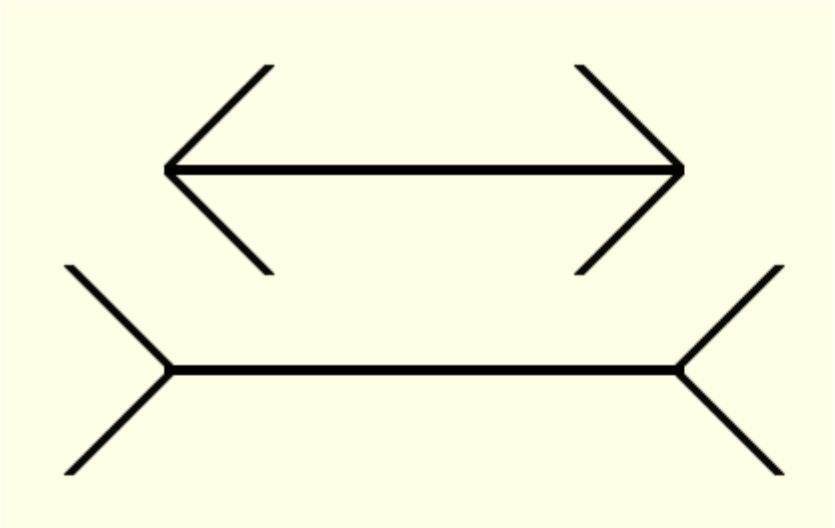


[Figure by Grauman & Darrell]

Object-level grouping



Muller-Lyer illusion



What things should be grouped?
What cues indicate groups?

Gestalt

Gestalt: whole or group

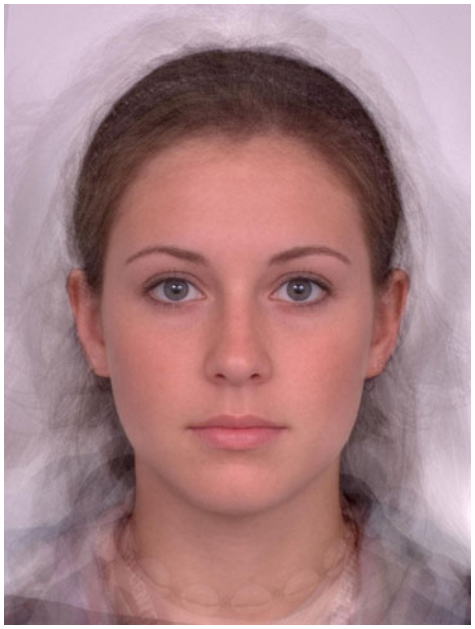
- Whole is greater than sum of its parts
- Relationships among parts can yield new properties/features

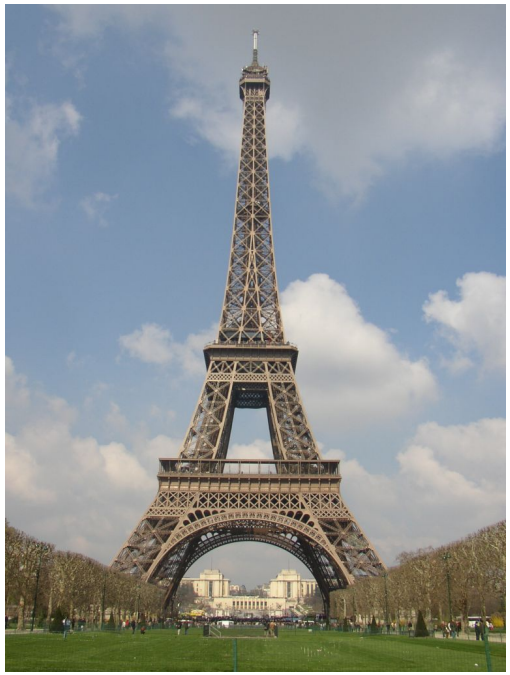
Psychologists identified series of factors that predispose set of elements to be grouped (by human visual system)

Similarity

http://chicagoist.com/attachments/chicagoist_alicia/GEESE.jpg, http://wwwdelivery.superstock.com/WI/223/1532/PreviewComp/SuperStock_1532R-0831.jpg

Symmetry





http://seedmagazine.com/news/2006/10/beauty_is_in_the_processingtim.php

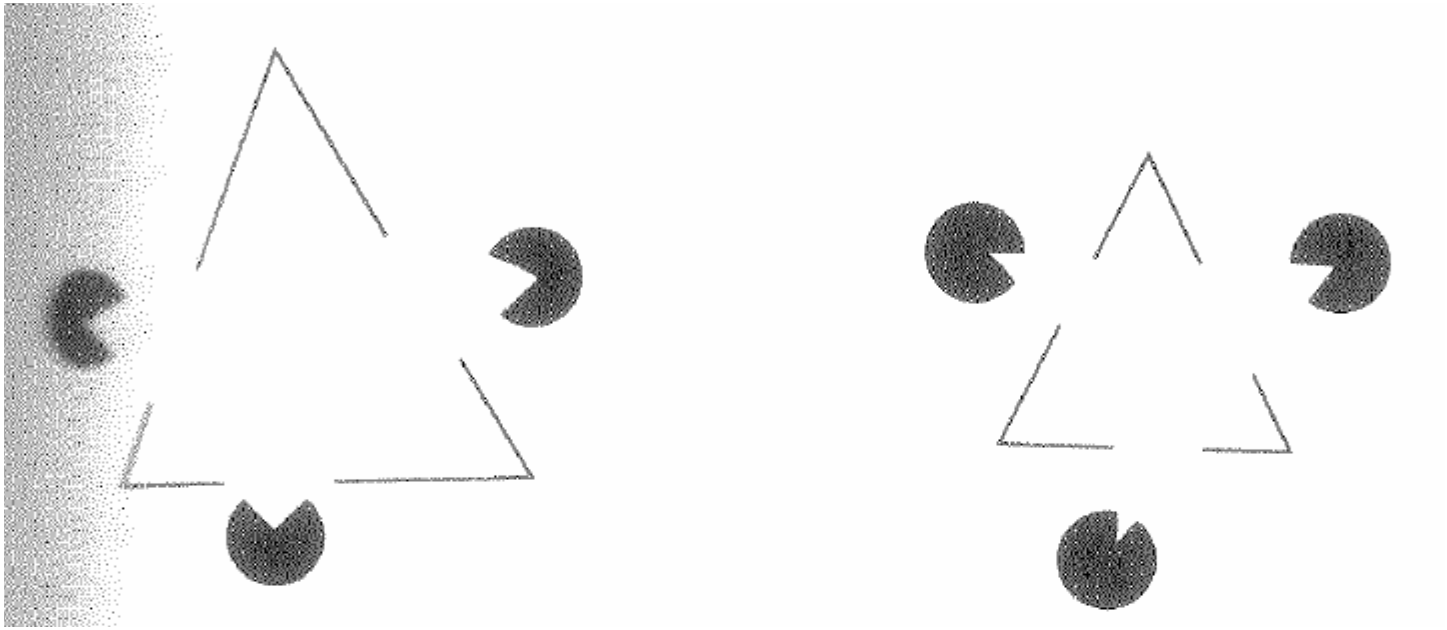
Common fate

Image credit: Arthus-Bertrand (via F. Durand)

Proximity

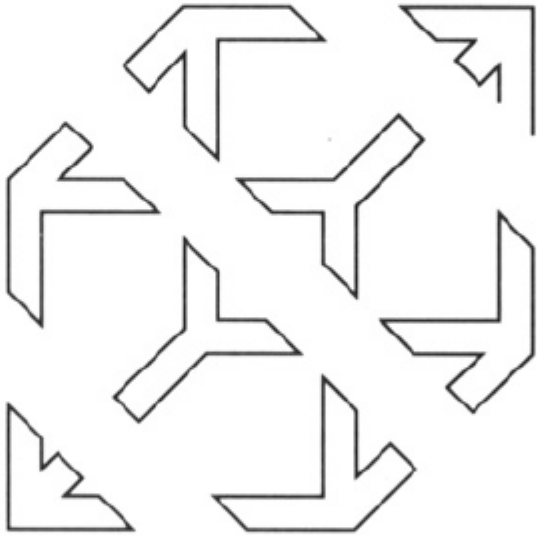
http://www.capital.edu/Resources/Images/outside6_035.jpg

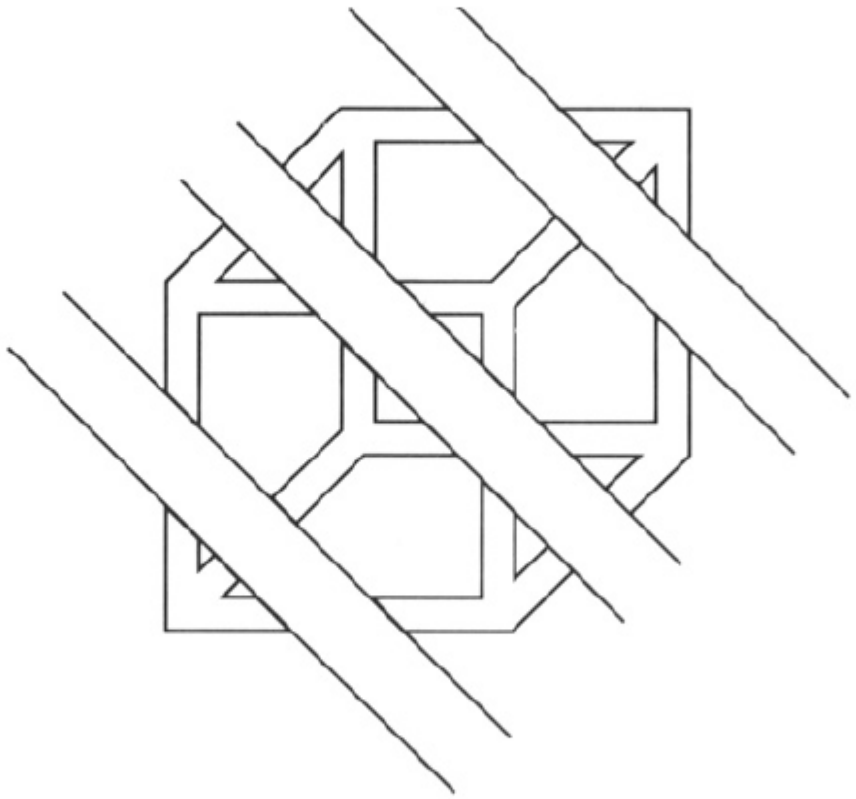
Illusory/subjective contours



Interesting tendency to explain by occlusion

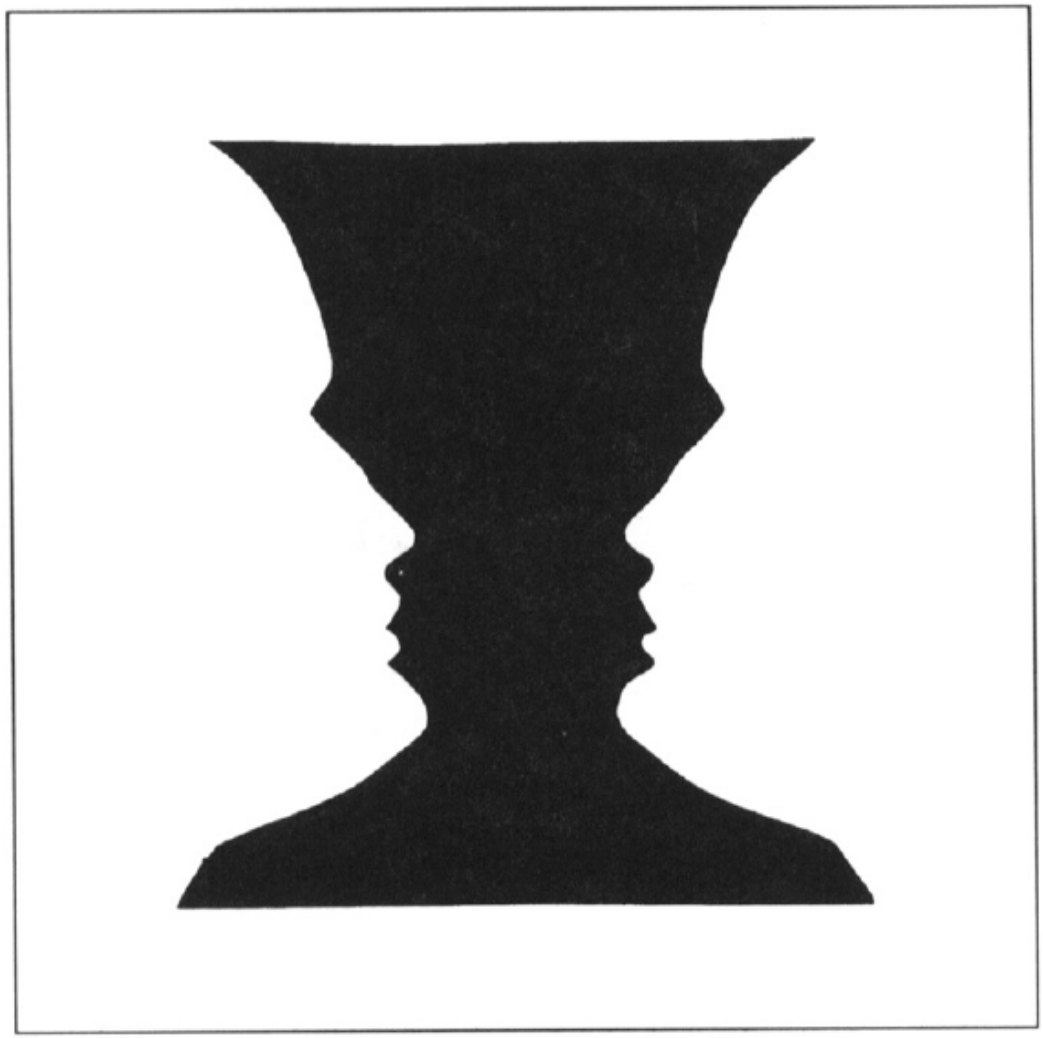
In *Vision*, D. Marr, 1982

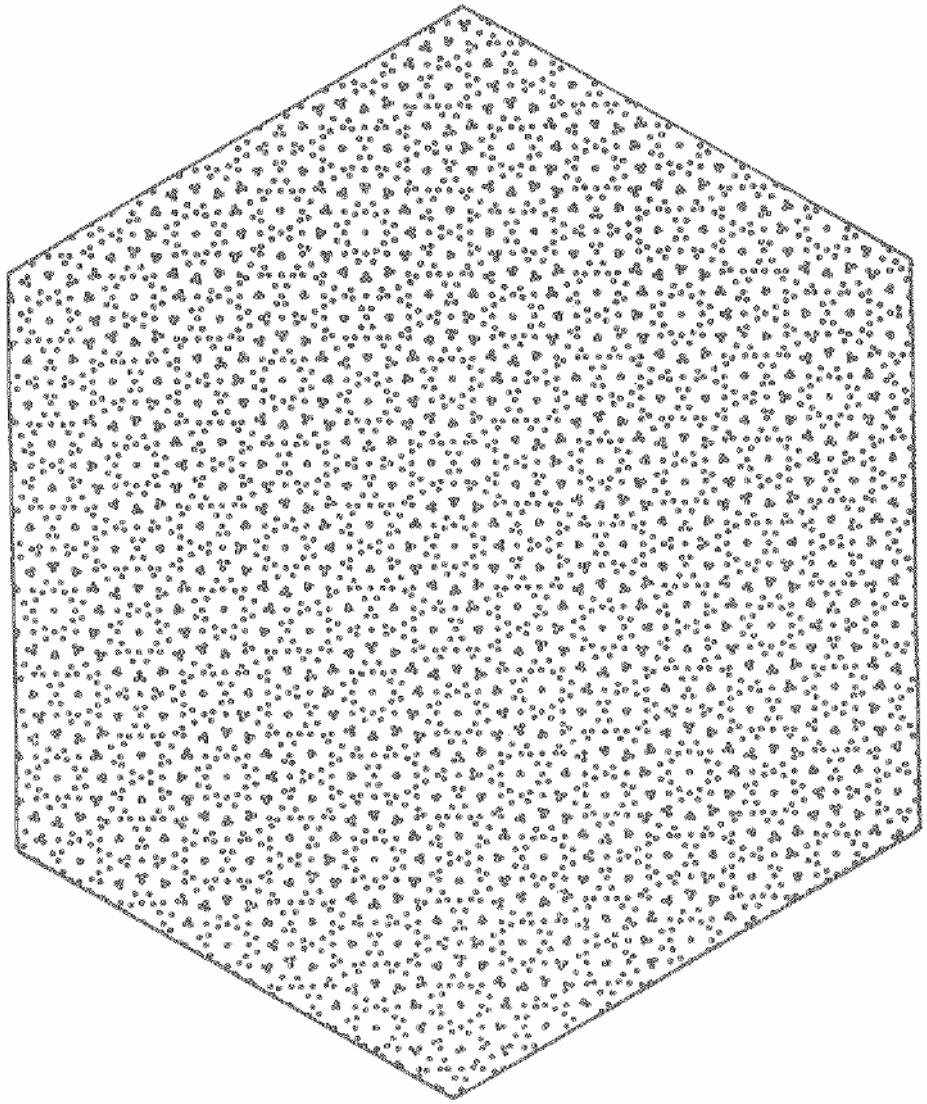




Continuity, explanation by occlusion

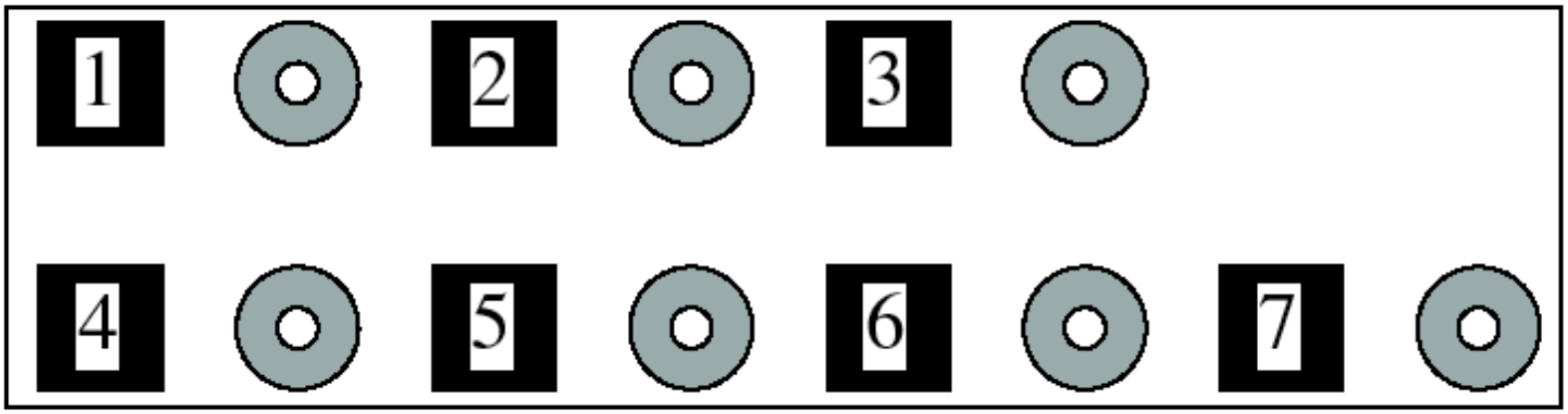
Figure-ground





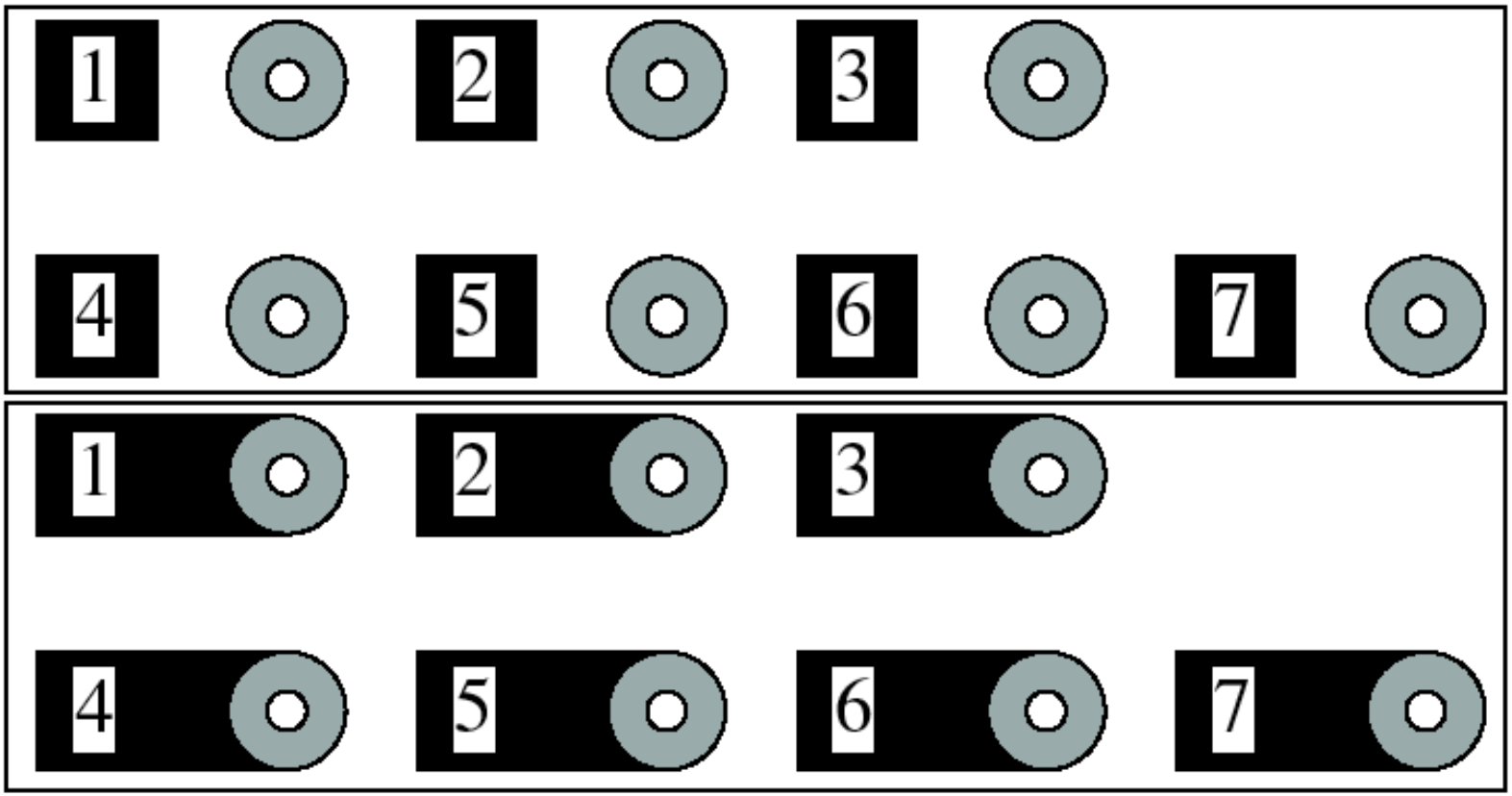
In *Vision*, D. Marr, 1982; from J. L. Marroquin, "Human visual perception of structure", 1976.

Grouping phenomena in real life



Forsyth & Ponce, Figure 14.7

Grouping phenomena in real life



Forsyth & Ponce, Figure 14.7

Gestalt

Gestalt: whole or group

- Whole is greater than sum of its parts
- Relationships among parts can yield new properties/features

Psychologists identified series of factors that predispose set of elements to be grouped (by human visual system)

Inspiring observations/explanations; challenge remains how to best map to algorithms.

Bottom-up segmentation via clustering

Grouping in vision

Goals:

- Gather features that belong together
- Obtain an intermediate representation that compactly describes key image (video) parts

Top down vs. bottom up segmentation

- Top down: pixels belong together because they are from the same object
- Bottom up: pixels belong together because they look similar

Hard to measure success

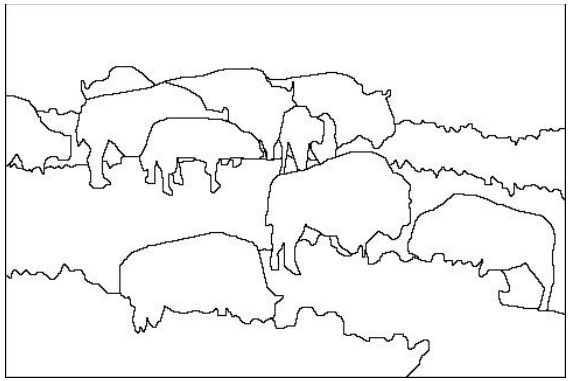
- What is interesting depends on the app.

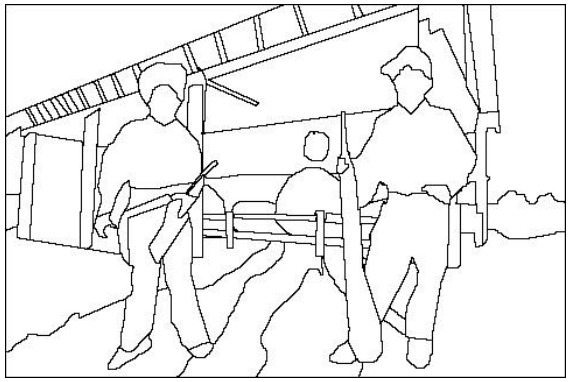
The goals of segmentation

Separate image into coherent “objects”

image

human segmentation





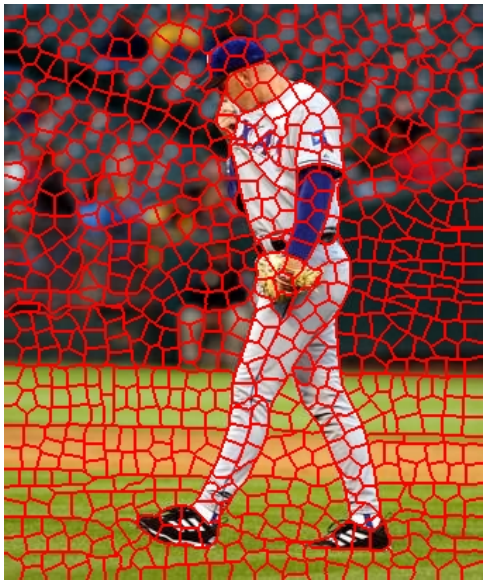
Source: Lana Lazebnik

The goals of segmentation

Separate image into coherent “objects”

Group together similar-looking pixels for efficiency of further processing

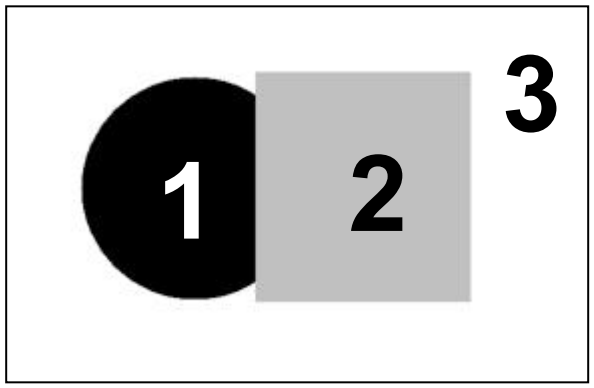
“superpixels”



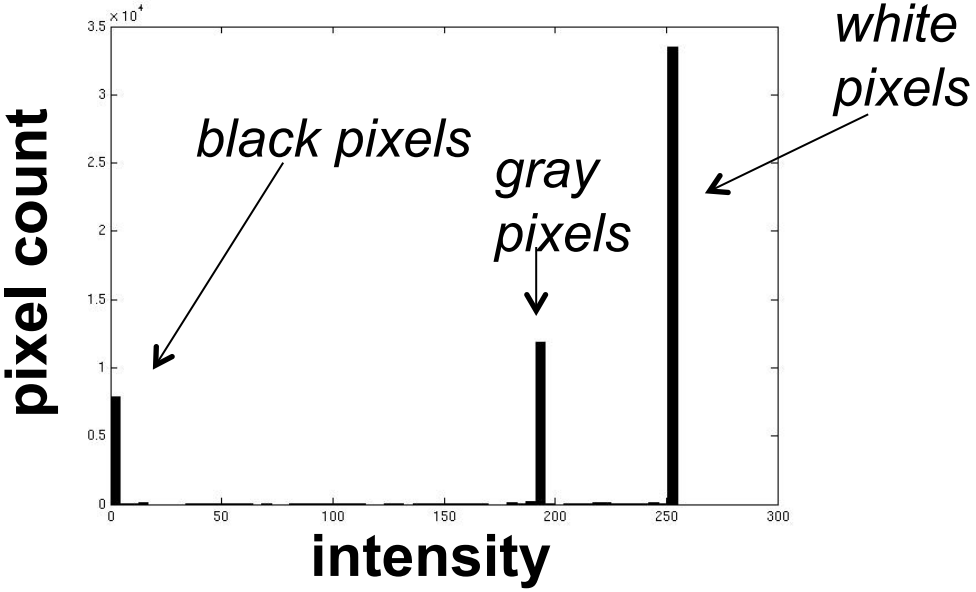
X. Ren and J. Malik. **Learning a classification model for segmentation.** ICCV 2003.

Source: Lana Lazebnik

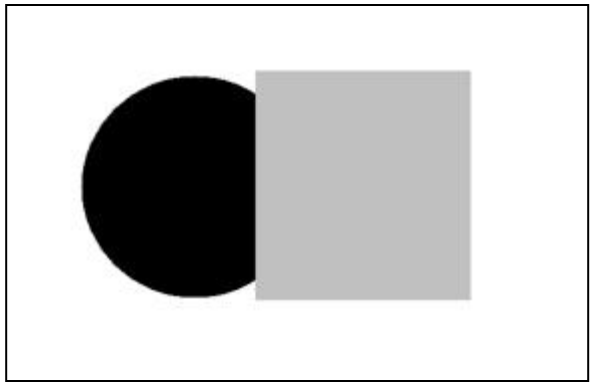
Image segmentation: toy example



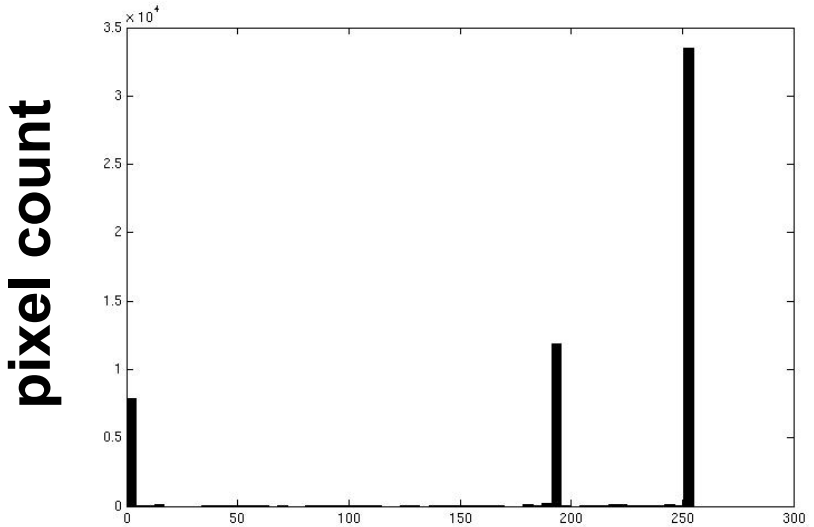
input image

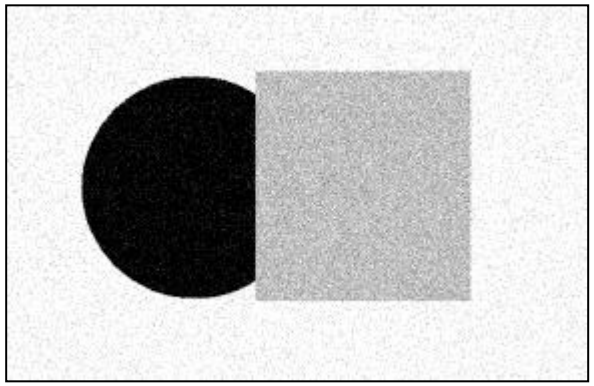


- These intensities define the three groups.
- We could label every pixel in the image according to which of these primary intensities it is.
  - i.e., *segment* the image based on the intensity feature.
- What if the image isn't quite so simple?

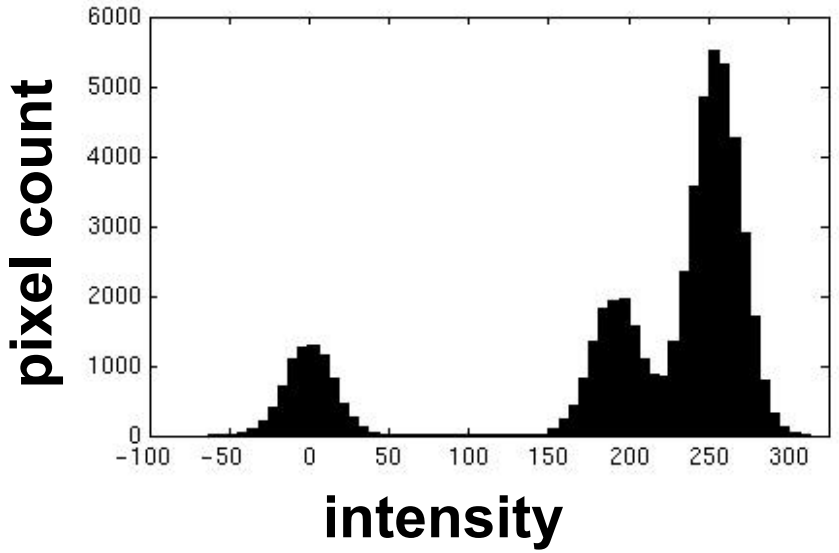


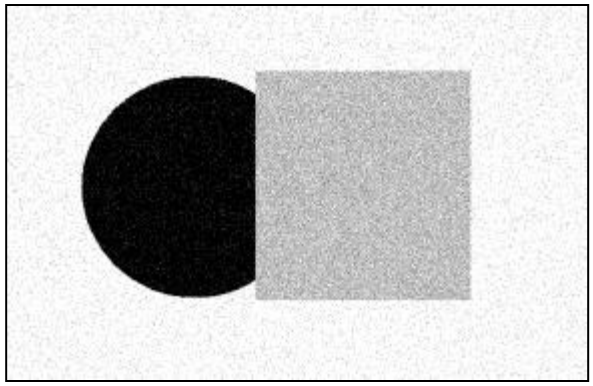
input image



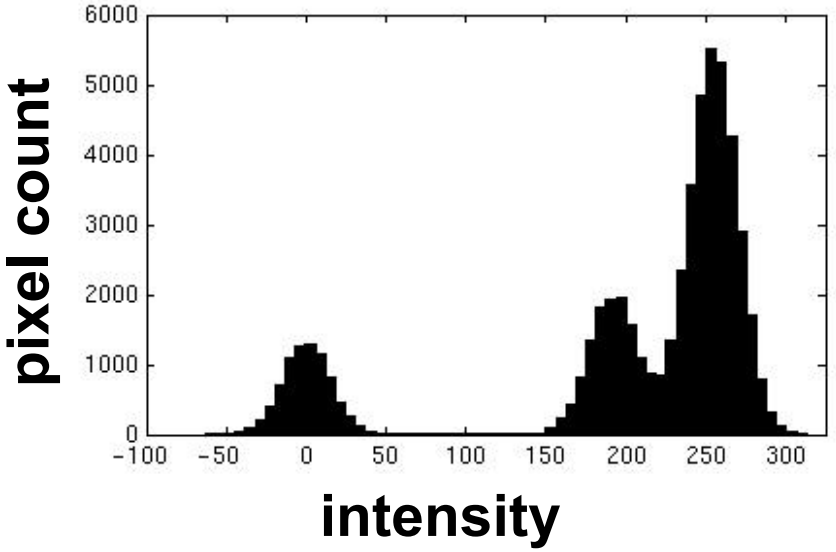


input image

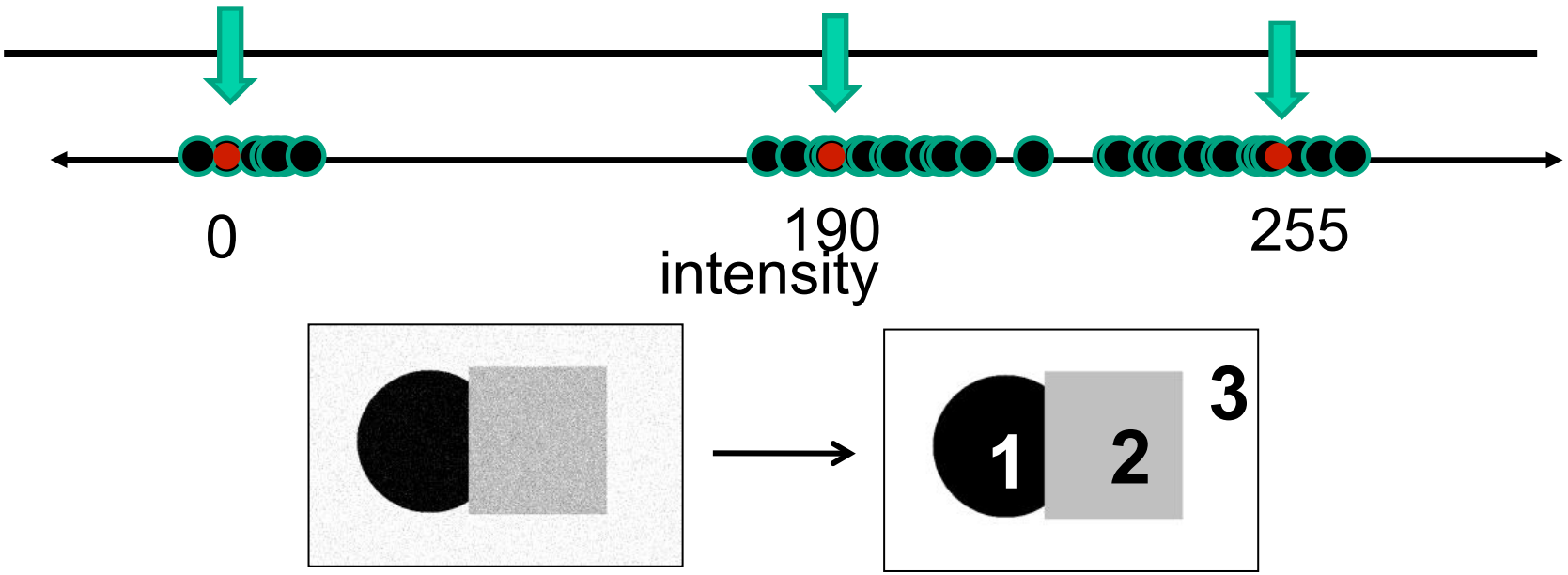




input image



- Now how to determine the three main intensities that define our groups?
- We need to ***cluster.***



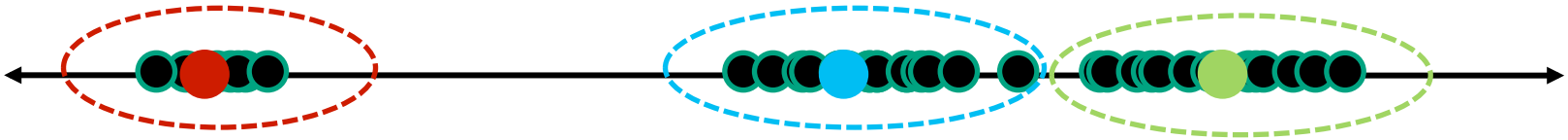
- Goal: choose three "centers" as the representative intensities, and label every pixel according to which of these centers it is nearest to.
- Best cluster centers are those that minimize SSD between all points and their nearest cluster center ci:

$$\sum_{\text{clusters } i} \sum_{\text{points p in cluster } i} \|p - c_i\|^2$$

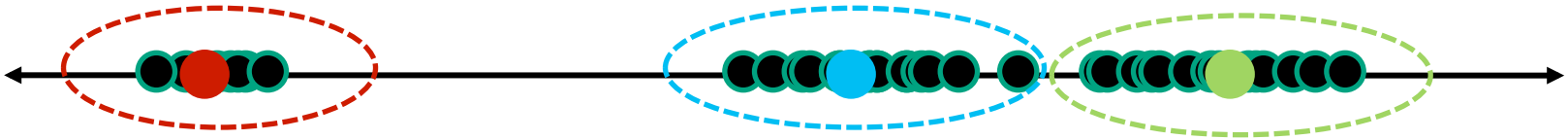
Clustering

With this objective, it is a "chicken and egg" problem:

- If we knew the **cluster centers**, we could allocate points to groups by assigning each to its closest center.



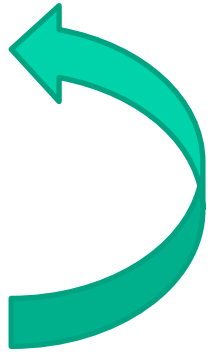
- If we knew the **group memberships**, we could get the centers by computing the mean per group.



K-means clustering

Basic idea: randomly initialize the *k* cluster centers, and iterate between the two steps we just saw.

1. Randomly initialize the cluster centers, c_1, ..., c_K
2. Given cluster centers, determine points in each cluster
   - For each point p, find the closest c_i. Put p into cluster i
3. Given points in each cluster, solve for c_i
   - Set c_i to be the mean of points in cluster i
4. If c_i have changed, repeat Step 2



Properties

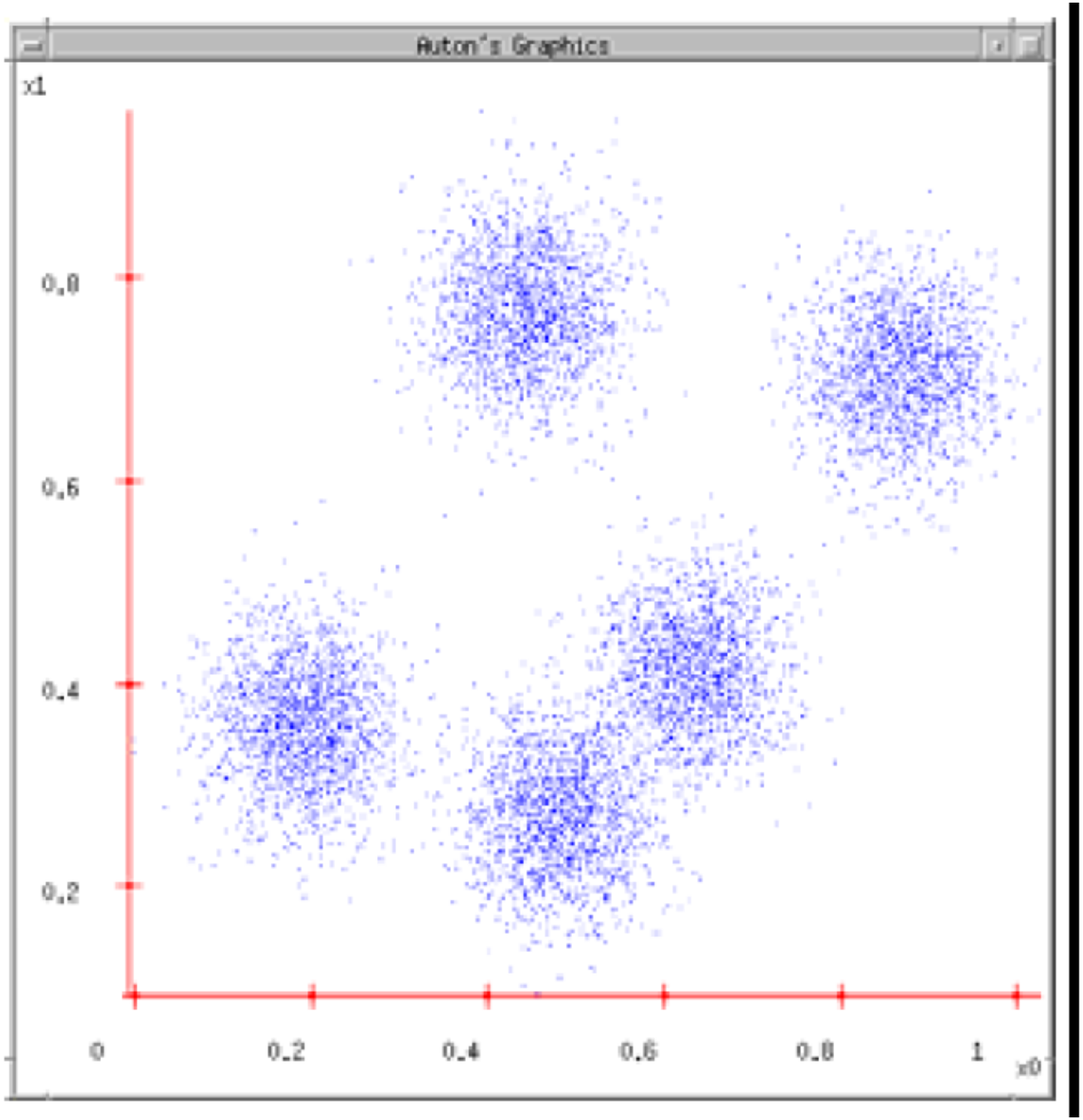
- Will always converge to *some* solution
- Can be a "local minimum"
  - does not always find the global minimum of objective function:

$$\sum_{\text{clusters } i} \sum_{\text{points p in cluster } i} ||p - c_i||^2$$

Source: Steve Seitz

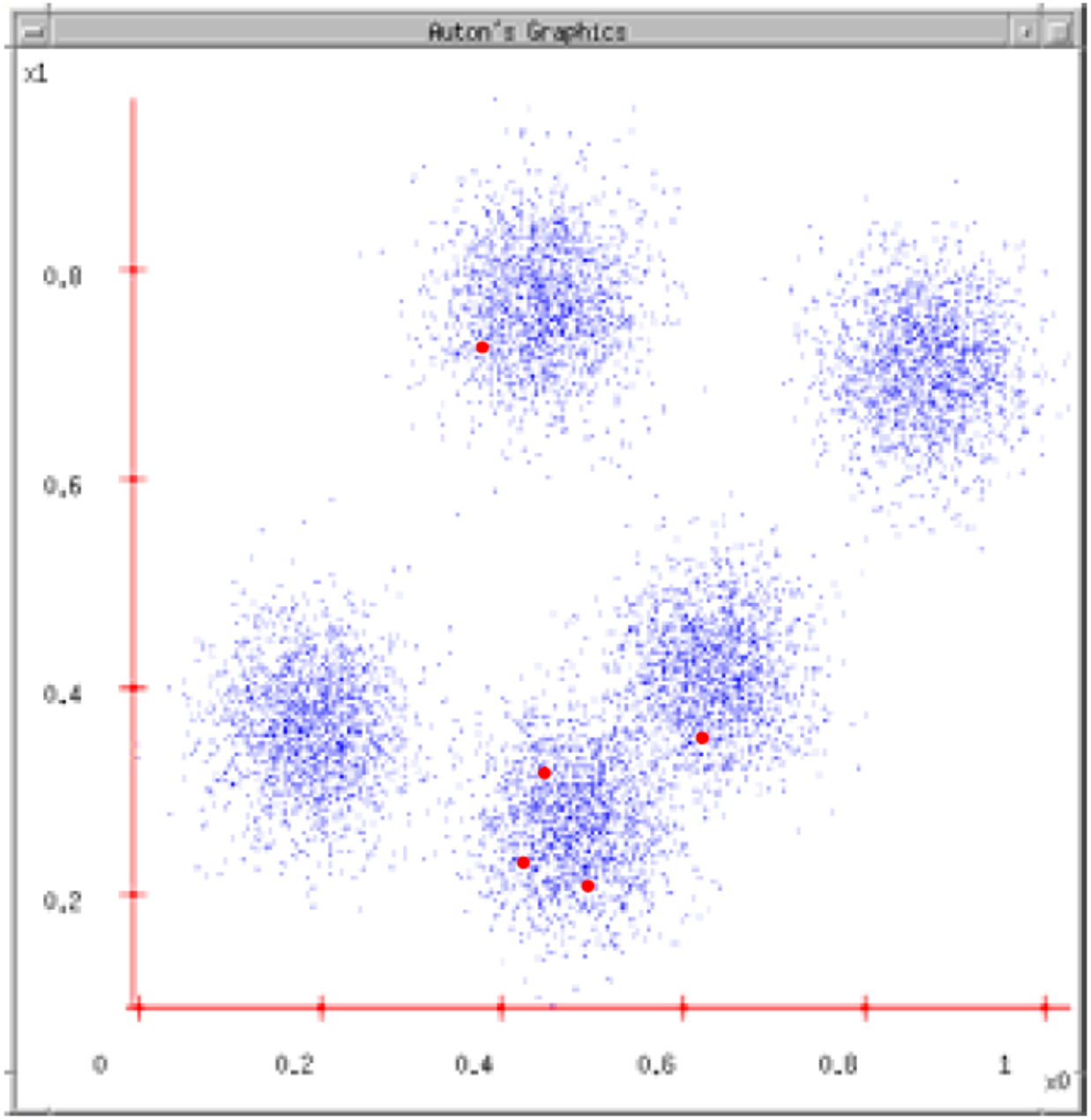
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



K-means

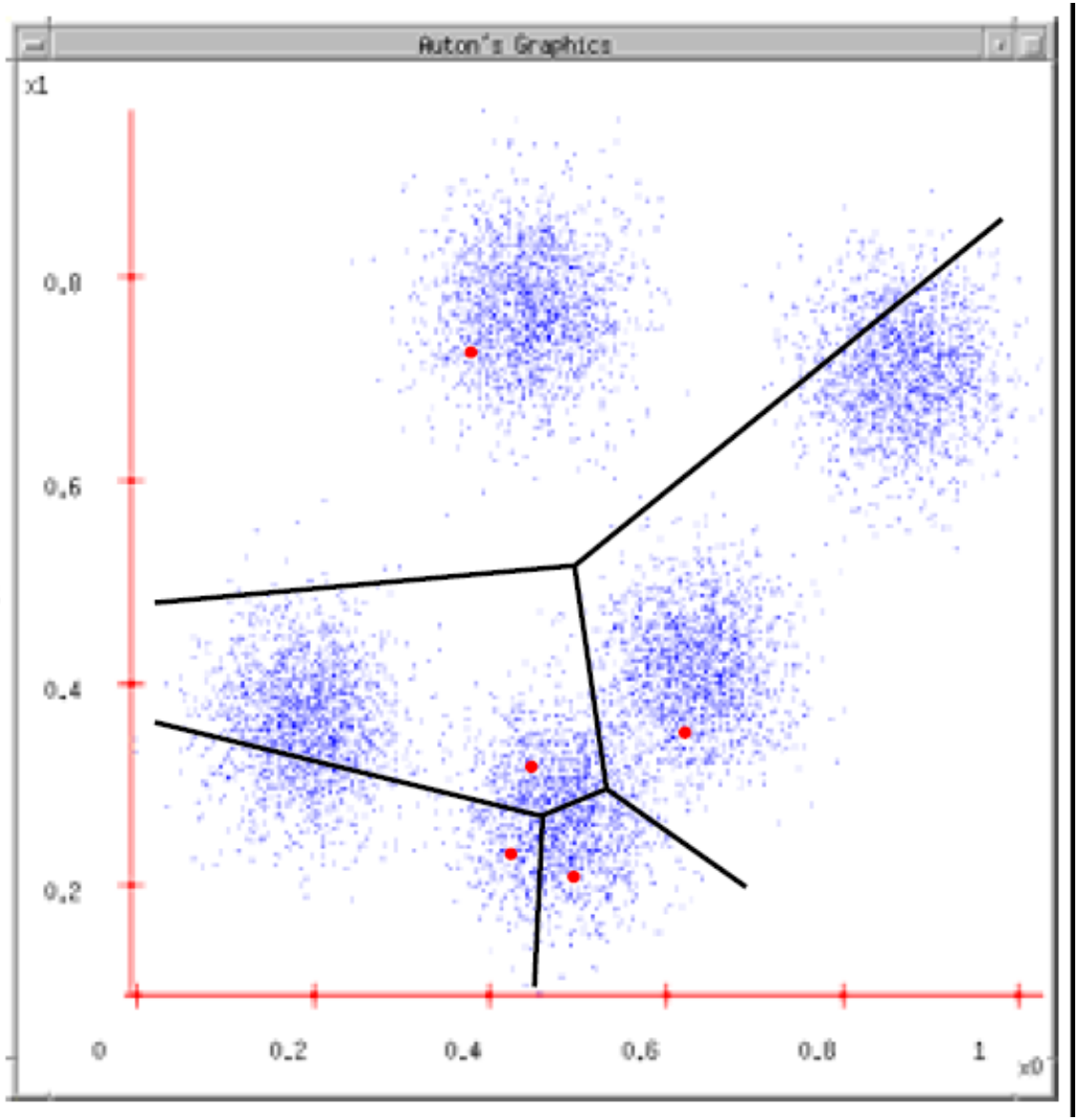
1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



Andrew Moore

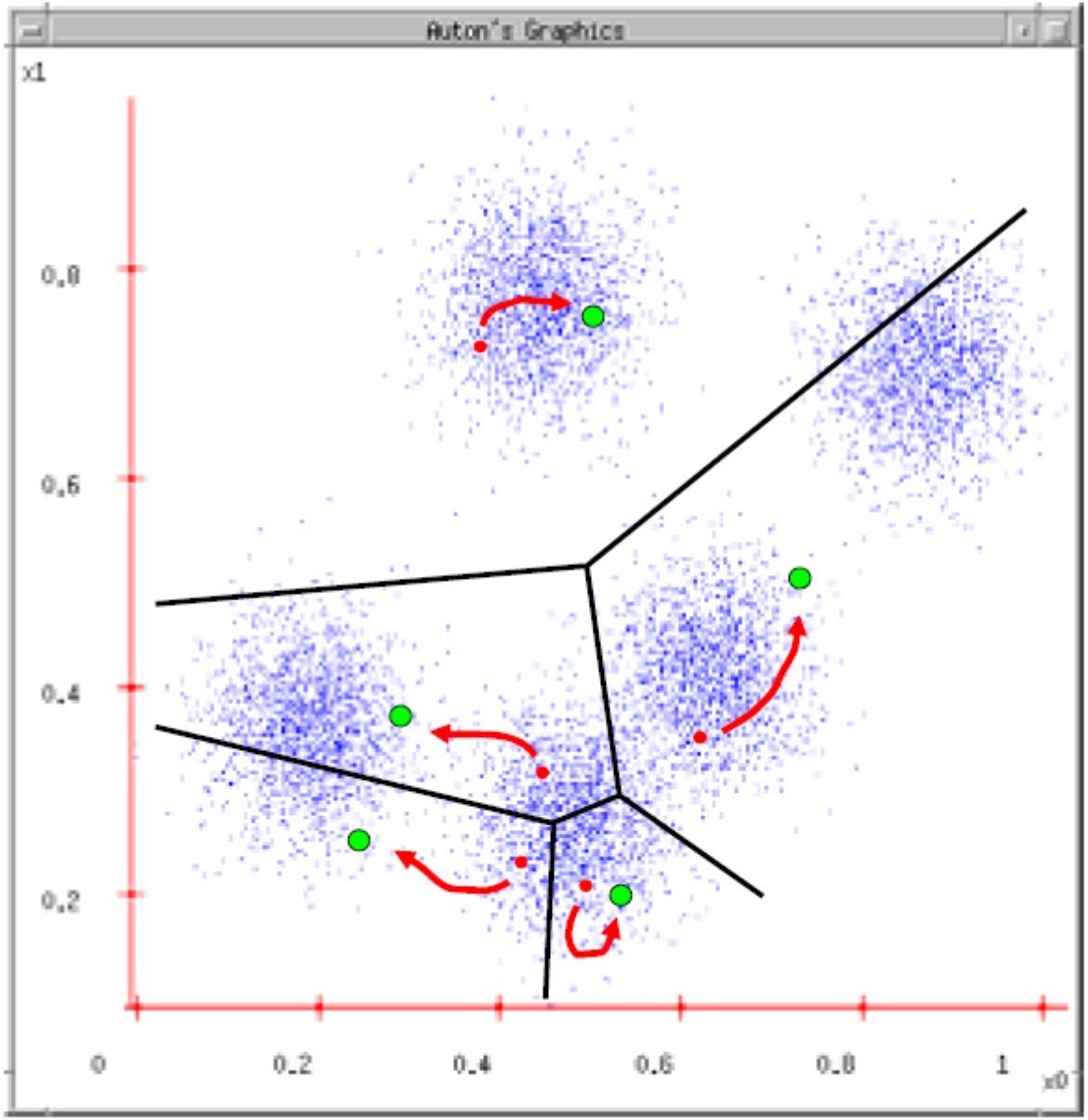
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



K-means

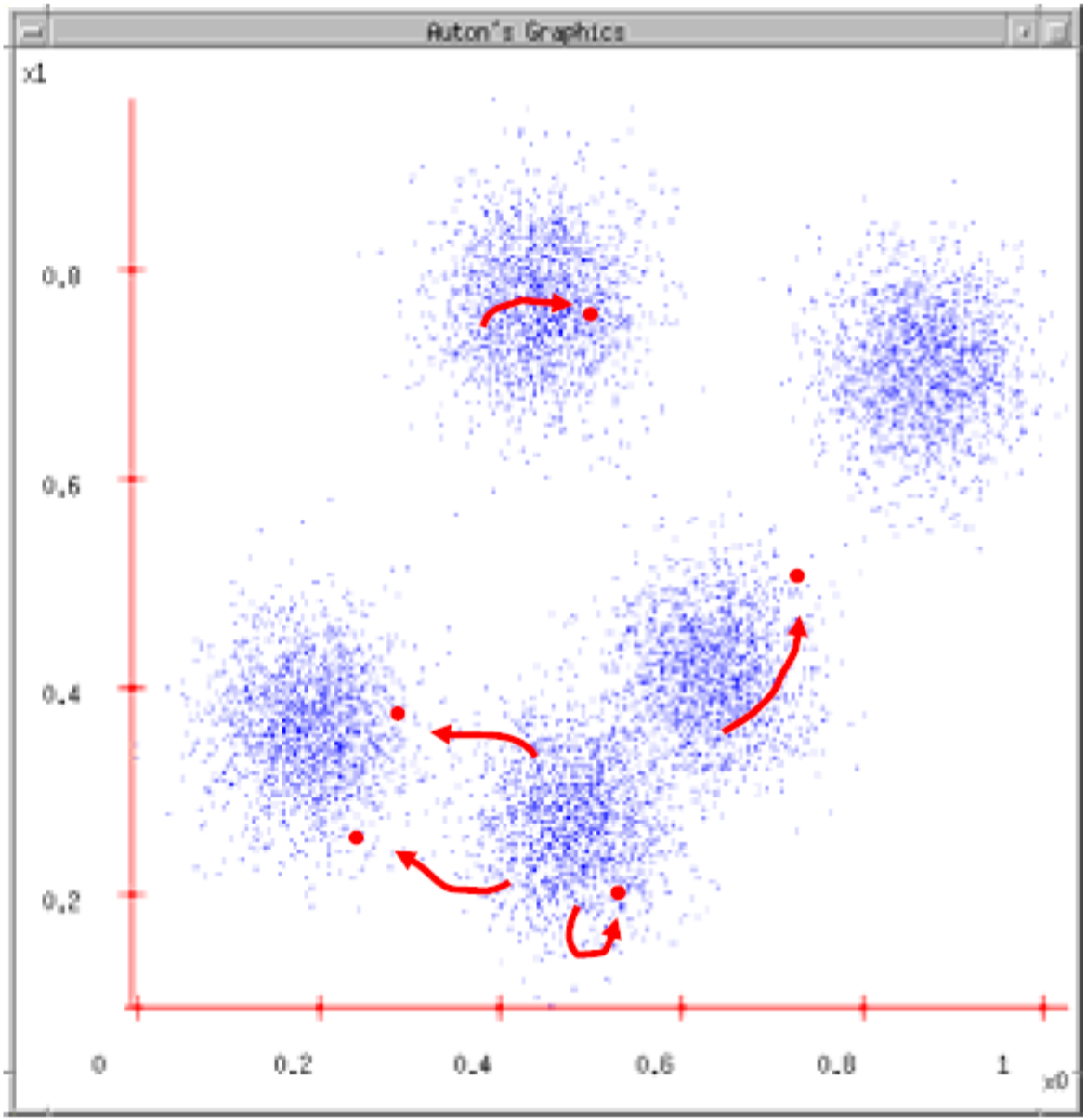
1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns



Andrew Moore

K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns...
5. ...and jumps there
6. ...Repeat until terminated!



Andrew Moore

K-means clustering

Java demo:

http://kovan.ceng.metu.edu.tr/~maya/kmeans/index.html

http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/AppletKM.html

K-means: pros and cons

Pros

Simple, fast to compute

Converges to local minimum of within-cluster squared error

Cons/issues

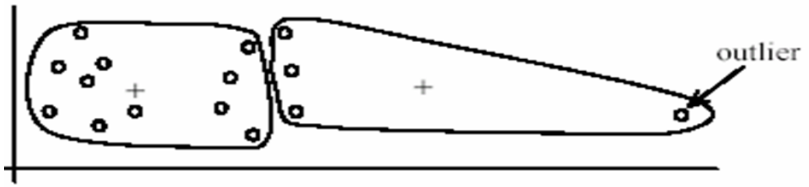
Setting k?

Sensitive to initial centers

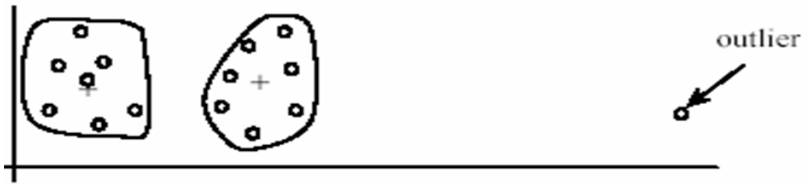
Sensitive to outliers

Detects spherical clusters

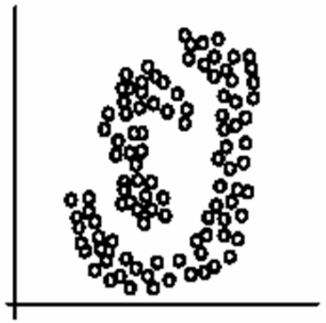
Assuming means can be computed



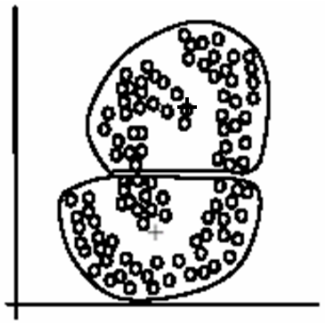
(A): Undesirable clusters



(B): Ideal clusters



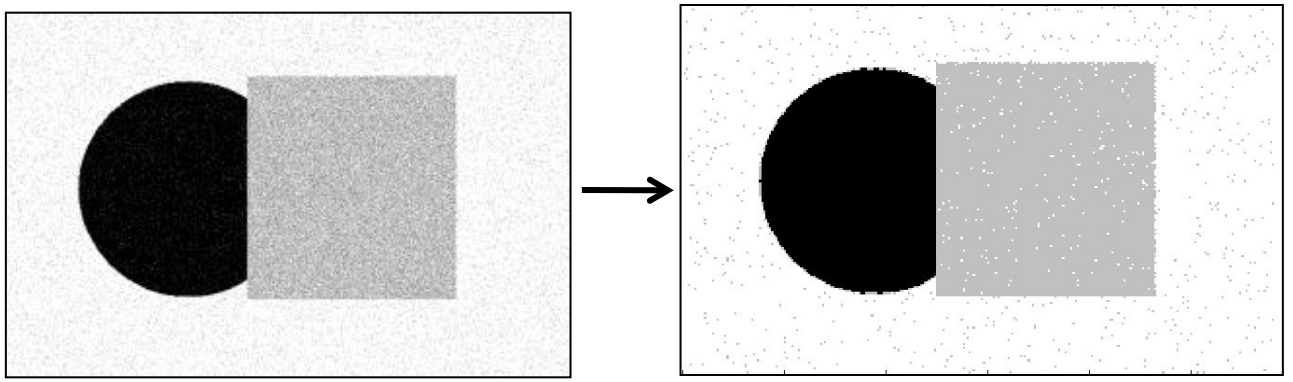
(A): Two natural clusters



(B): *k*-means clusters

An aside: Smoothing out cluster assignments

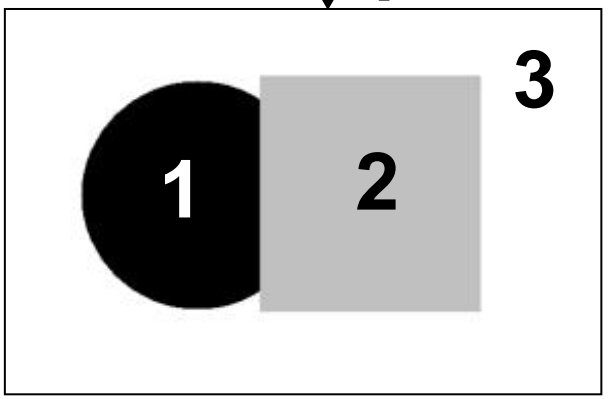
Assigning a cluster label per pixel may yield outliers:



original

labeled by cluster center's intensity

?



- How to ensure they are spatially smooth?

Segmentation as clustering

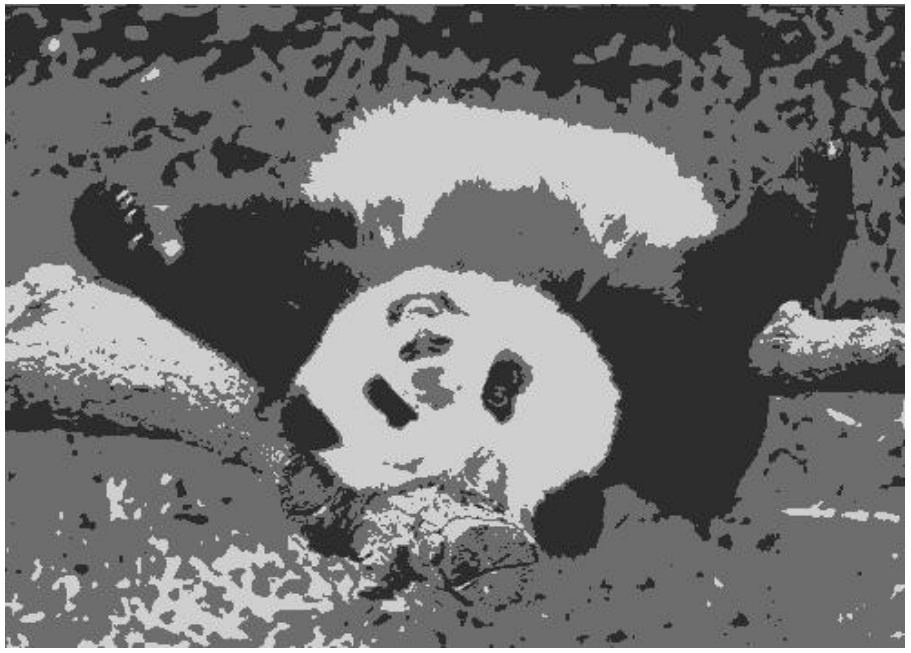
Depending on what we choose as the *feature space*, we can group pixels in different ways.

Grouping pixels based on **intensity** similarity

Feature space: intensity value (1-d)

K=2

K=3

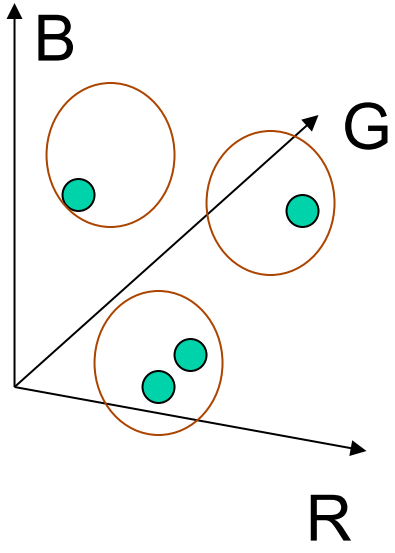


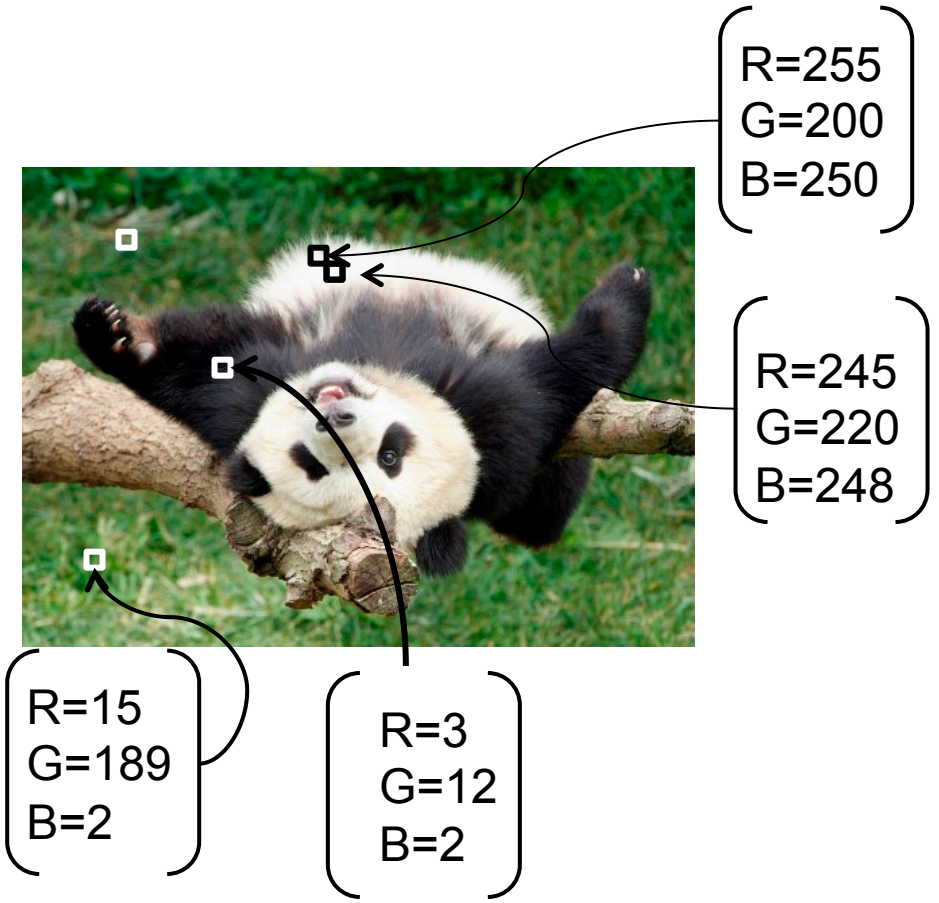
quantization of the feature space;
segmentation label map

Segmentation as clustering

Depending on what we choose as the *feature space*, we can group pixels in different ways.

Grouping pixels based on **color** similarity





Feature space: color value (3-d)

Kristen Grauman

Segmentation as clustering

Depending on what we choose as the *feature space*, we can group pixels in different ways.

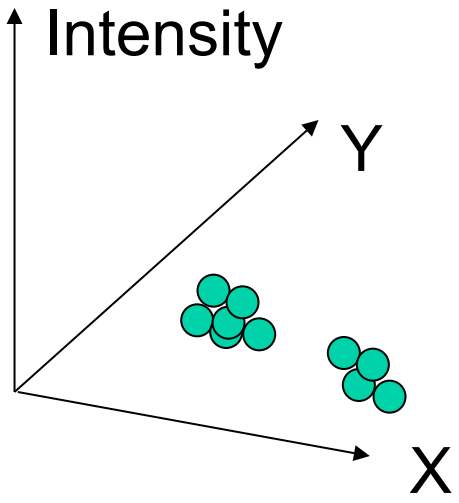
Grouping pixels based on **intensity** similarity

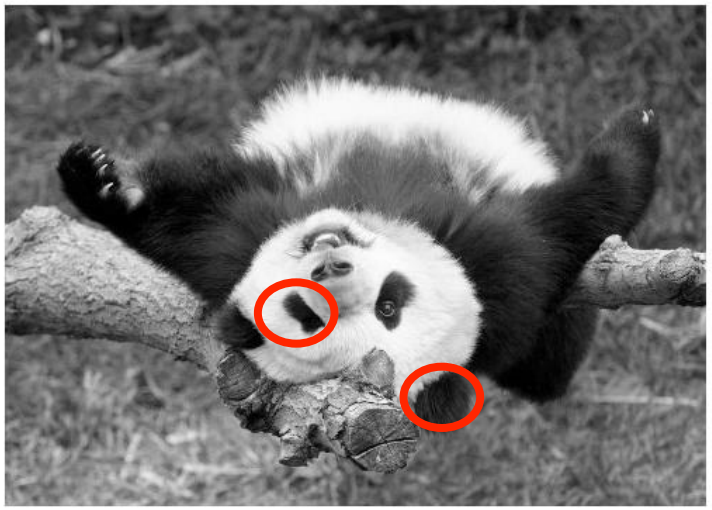
Clusters based on intensity similarity don't have to be spatially coherent.

Segmentation as clustering

Depending on what we choose as the *feature space*, we can group pixels in different ways.

Grouping pixels based on **intensity+position** similarity





Both regions are black, but if we also include **position (x,y),** then we could group the two into distinct segments; way to encode both similarity & proximity.

Segmentation as clustering

- Color, brightness, position alone are not enough to distinguish all regions…

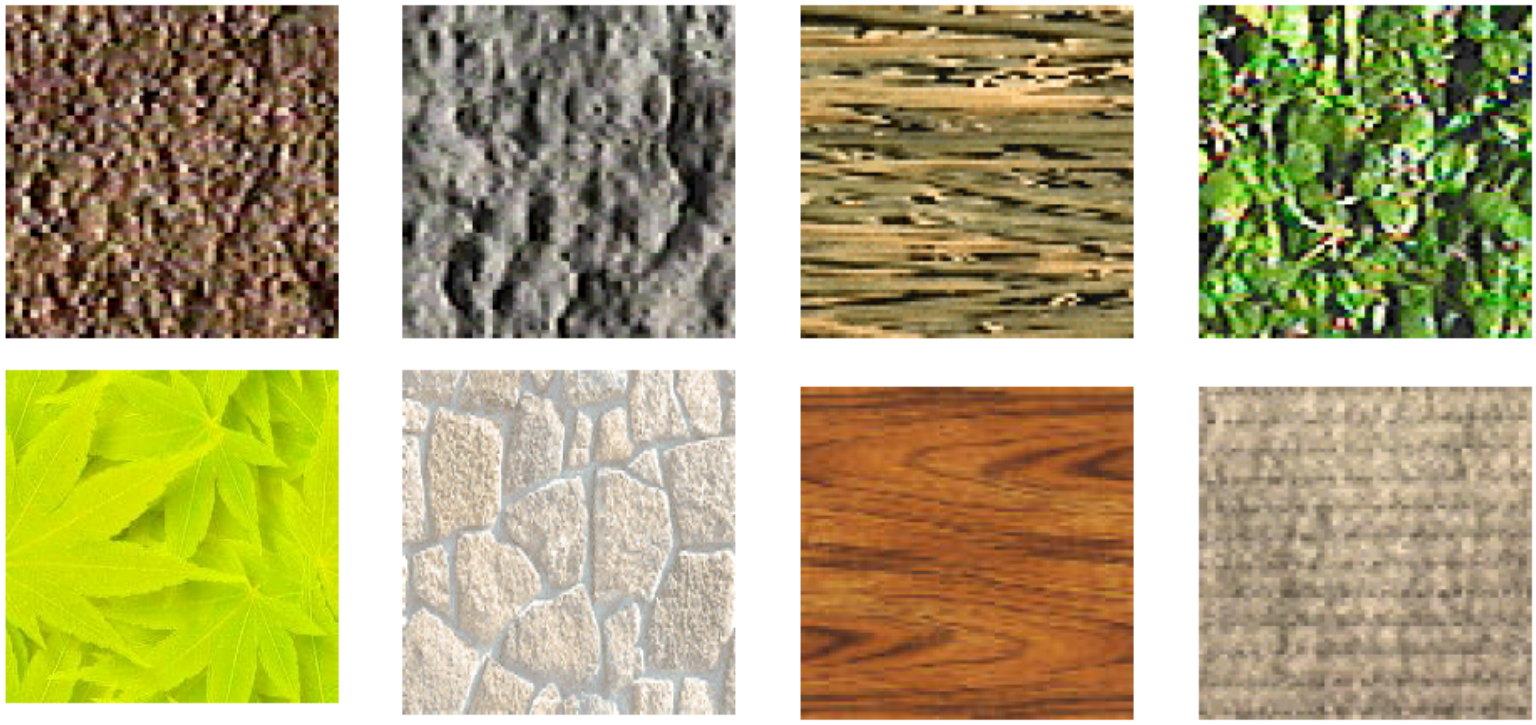


Textures

Texture cues for segmentation/classification

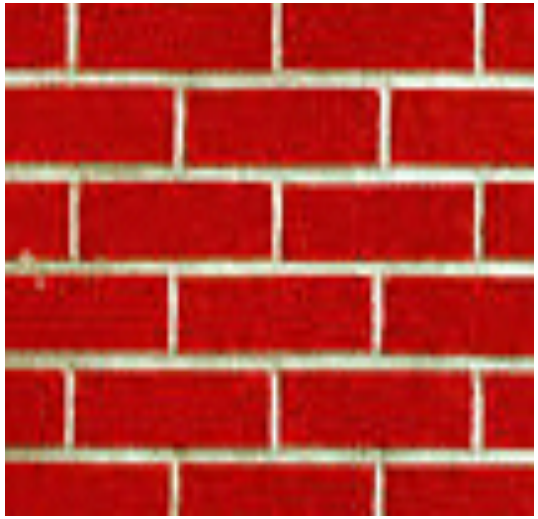
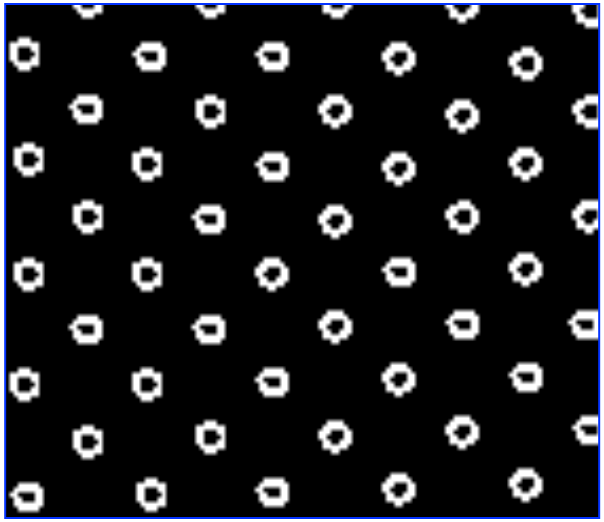
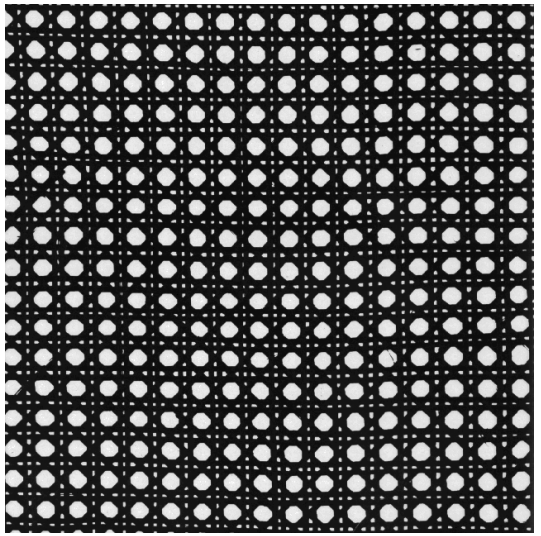
- Analyze, represent texture
- Group image regions with consistent texture

Texture



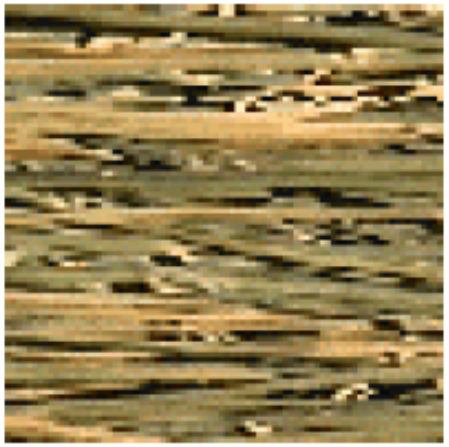
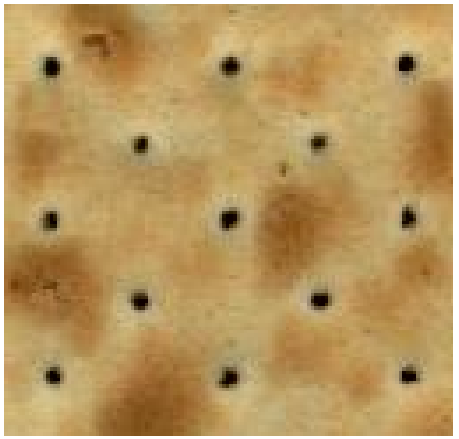
What defines a texture?

Includes: more regular patterns

Includes: more random patterns



What kind of response will we get with an edge detector for these images?

Images from Malik and Perona, 1990

…and for this image?

Image credit: D. Forsyth

Why analyze texture?

Importance to perception:

Often indicative of a material's properties

Can be important appearance cue, especially if shape is similar across objects

Aim to distinguish between shape, boundaries, and texture

Technically:

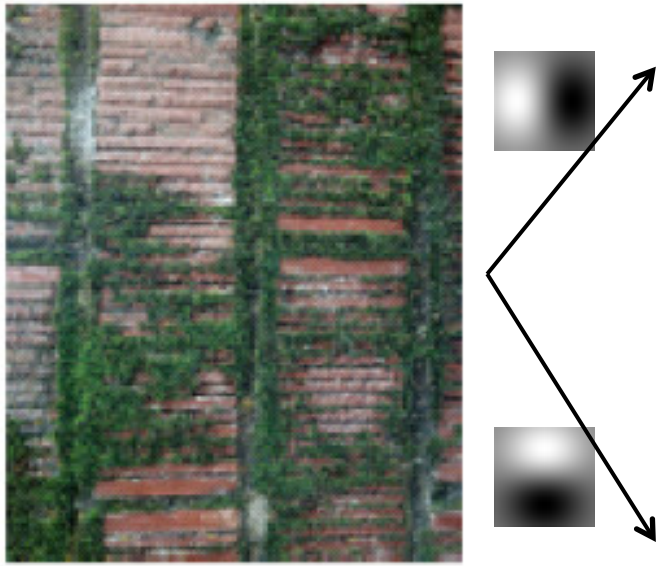
Representation-wise, we want a feature one step above "building blocks" of filters, edges.

Texture representation

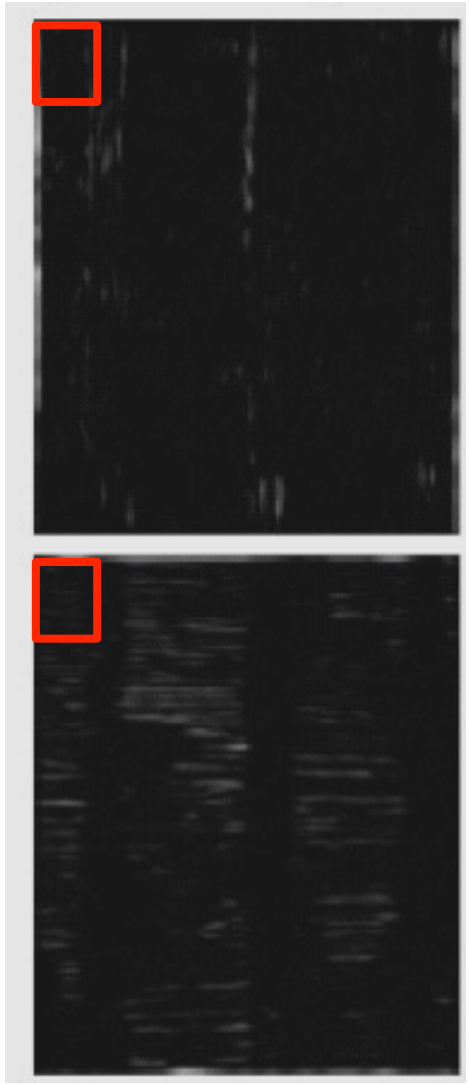
Textures are made up of repeated local patterns, so:

- 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
  - Histogram of “prototypical” feature occurrences

Texture representation: example



original image



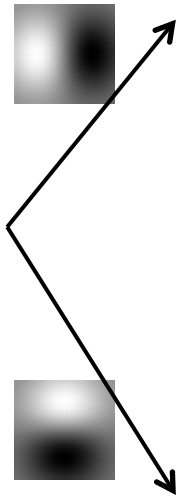
derivative filter responses, squared

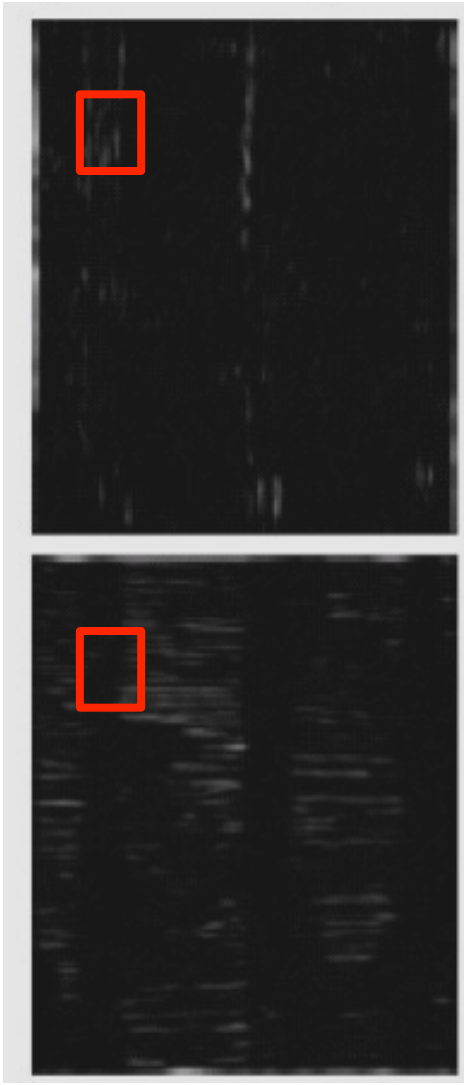
	mean d/dx value	mean d/dy value
Win. #1	4	10

statistics to summarize patterns in small windows

Texture representation: example

original image





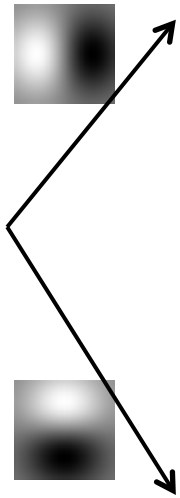
derivative filter responses, squared

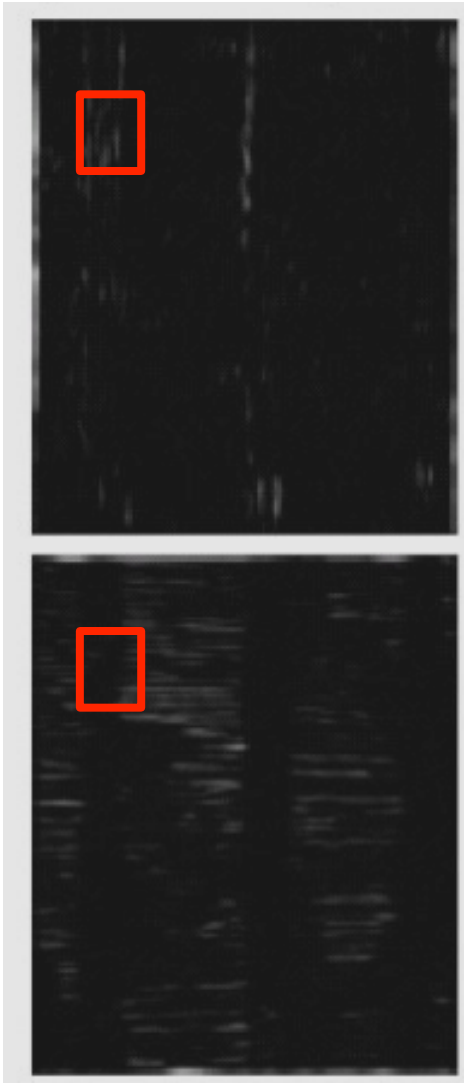
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7	

statistics to summarize patterns in small windows

Texture representation: example

original image



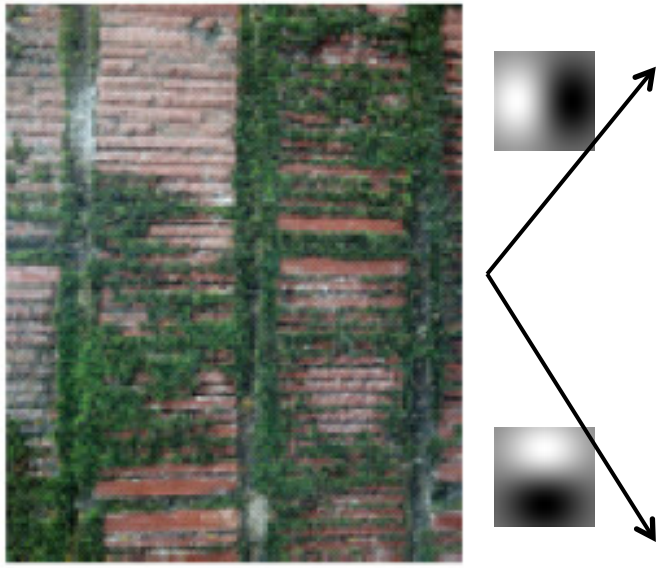


derivative filter responses, squared

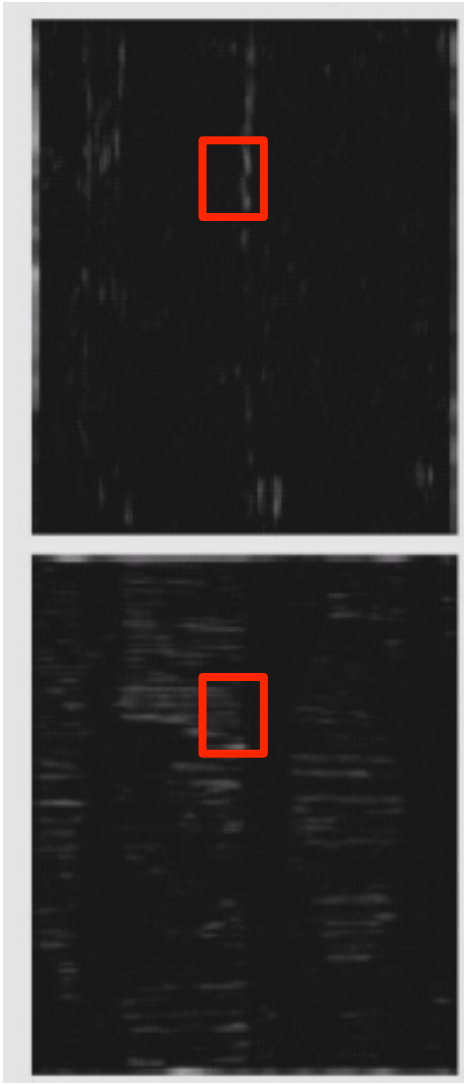
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7	

statistics to summarize patterns in small windows

Texture representation: example



original image



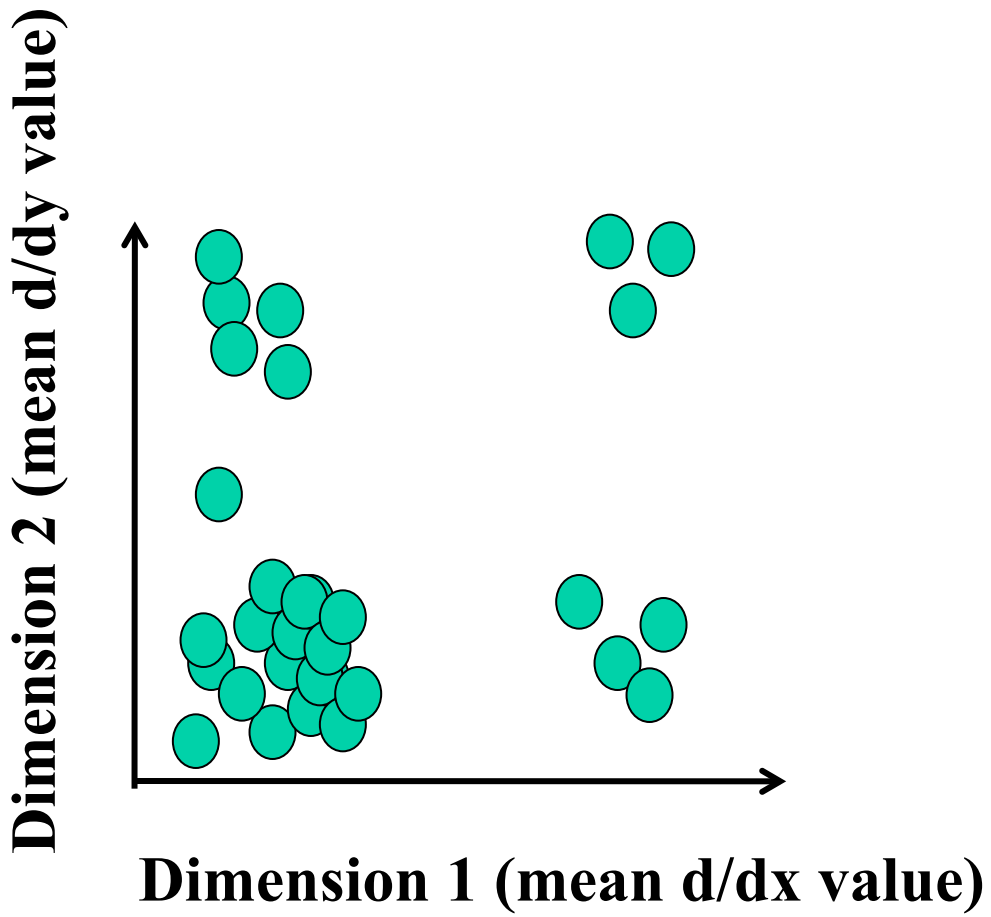
derivative filter responses, squared

	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

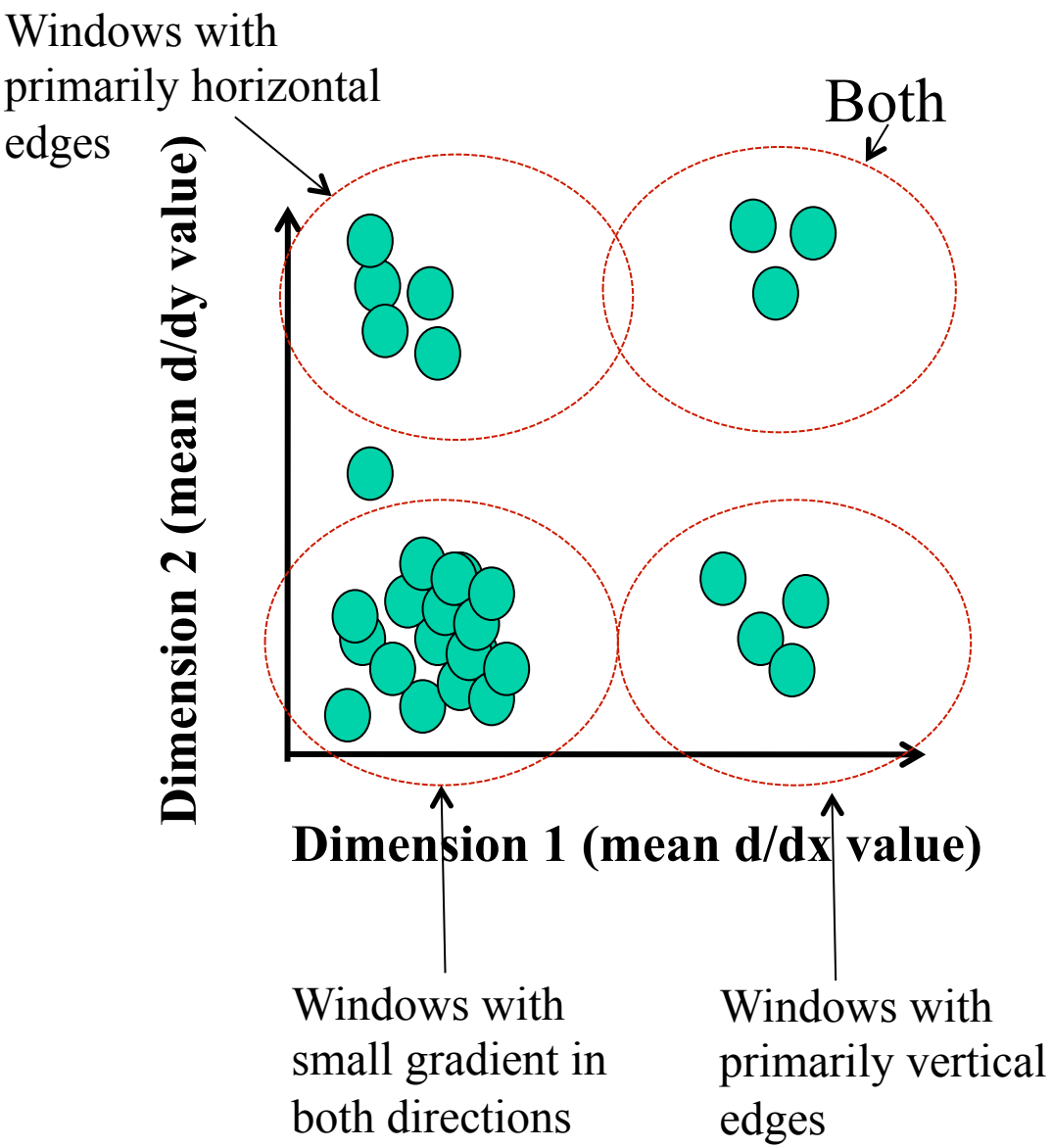
Texture representation: example



	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

Texture representation: example



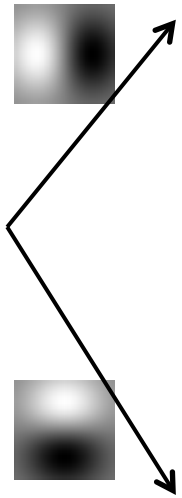
	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20

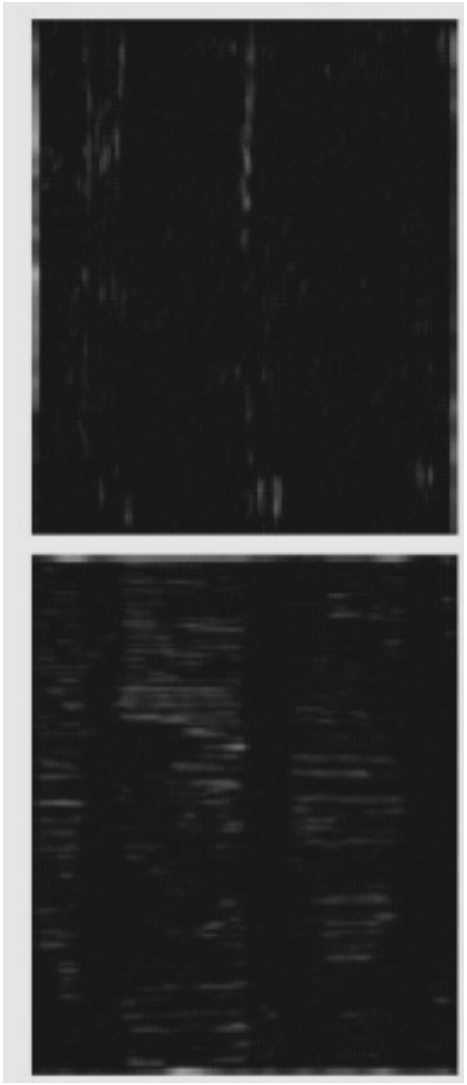
⋮

statistics to summarize patterns in small windows

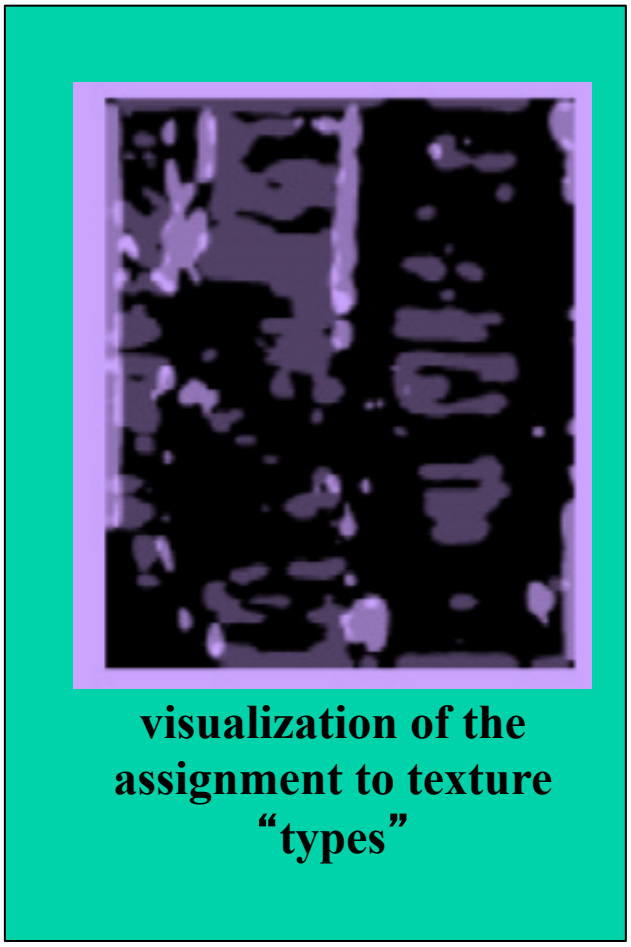
Texture representation: example

original image



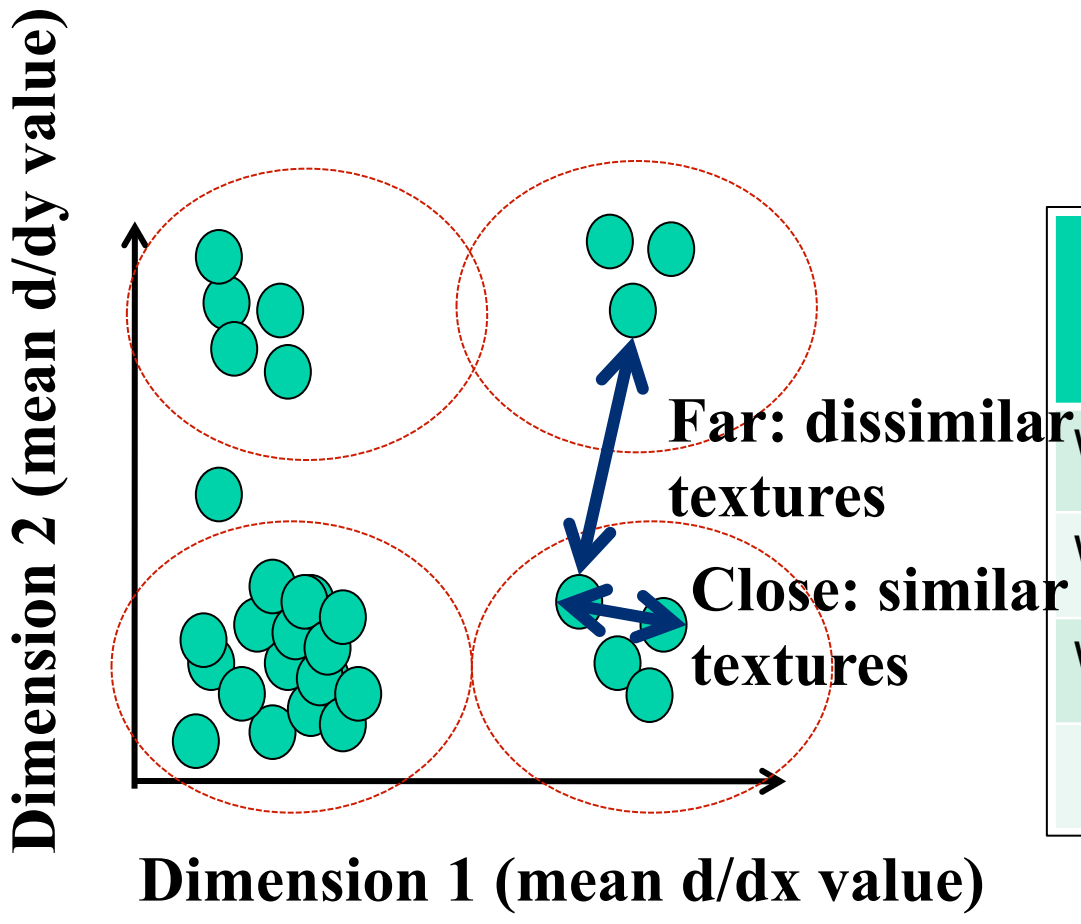


derivative filter responses, squared



visualization of the assignment to texture "types"

Texture representation: example

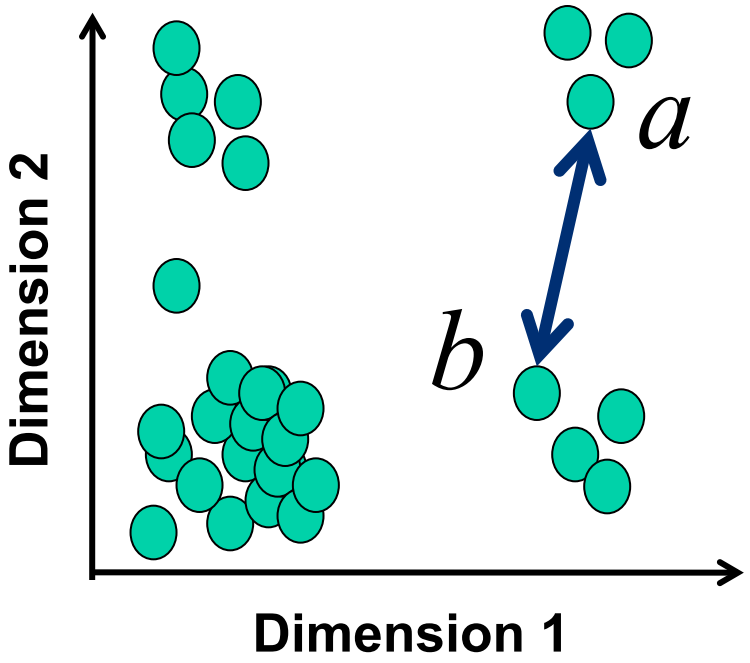


	mean d/dx value	mean d/dy value
Win. #1	4	10
Win.#2	18	7
⋮		
Win.#9	20	20

⋮

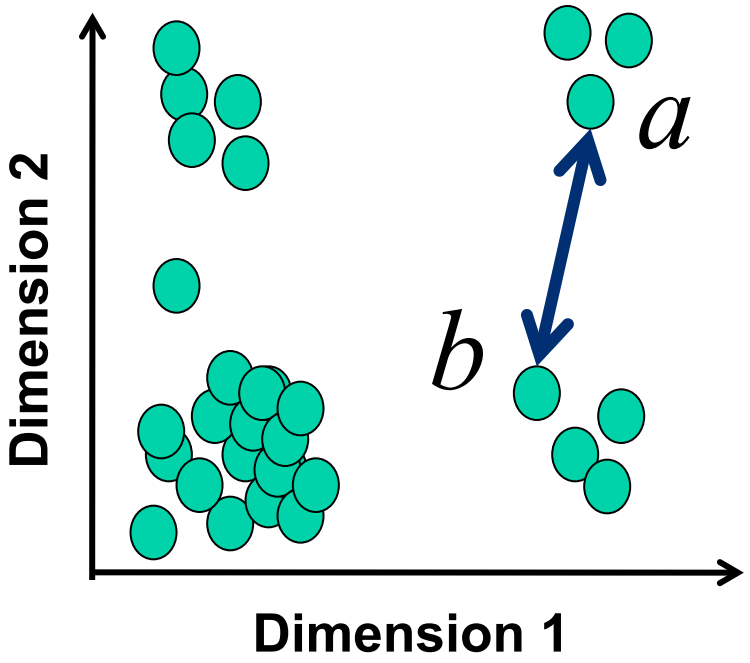
statistics to summarize patterns in small windows

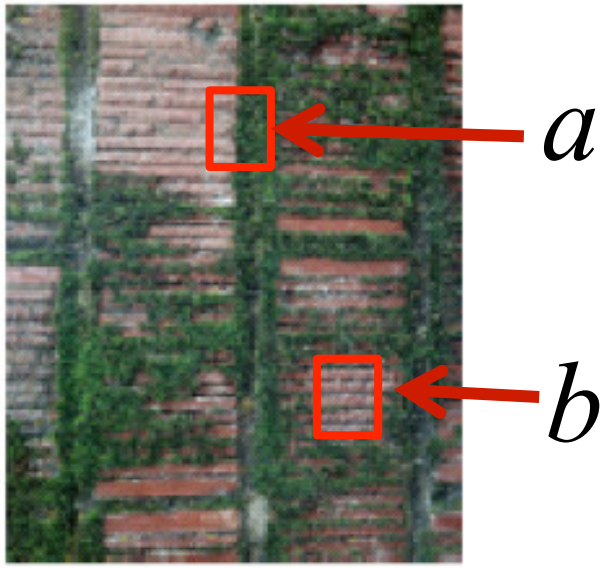
Texture representation: example

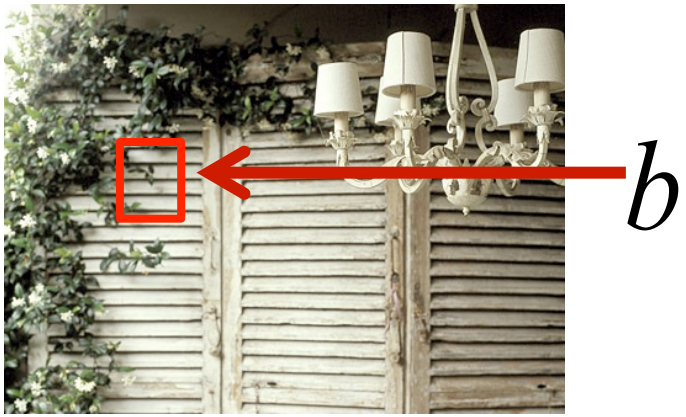


$$D(a,b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2}$$

Texture representation: example



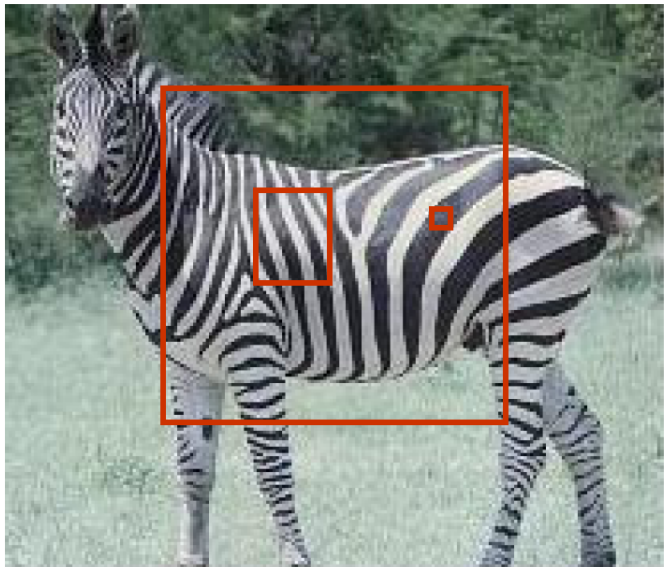




Distance reveals how dissimilar texture from window a is from texture in window b.

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 not changing.

Filter banks

Our previous example used two filters, and resulted in a 2-dimensional feature vector to describe texture in a window.

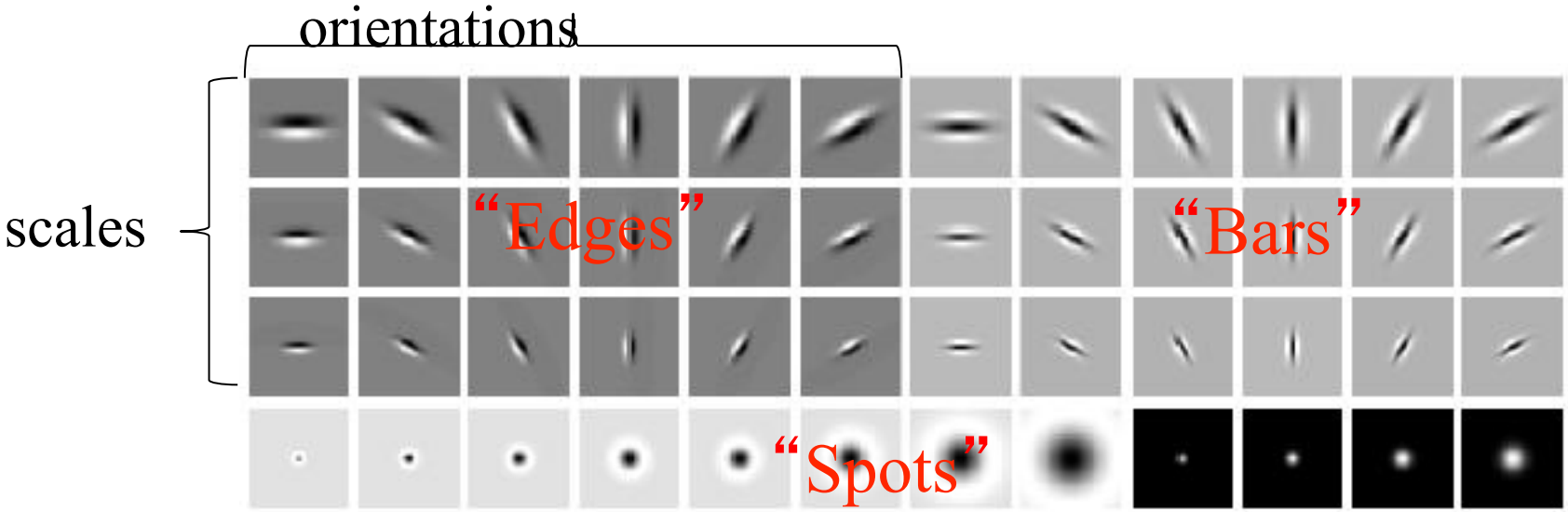
- x and y derivatives revealed something about local structure.

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

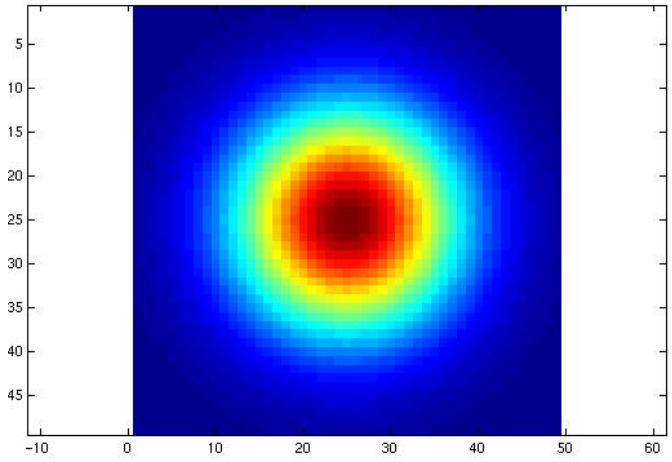


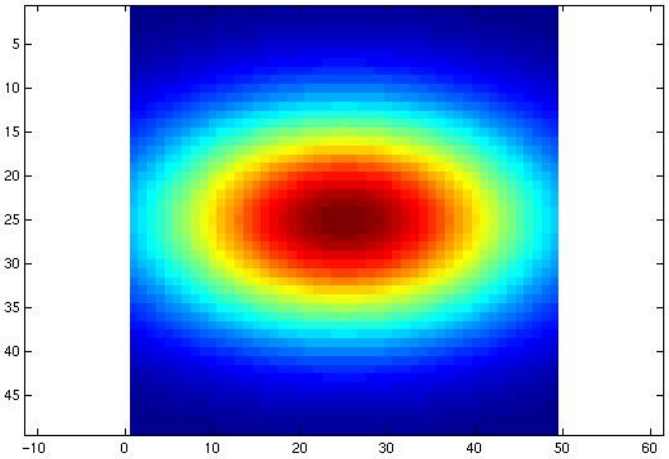
What filters to put in the bank?

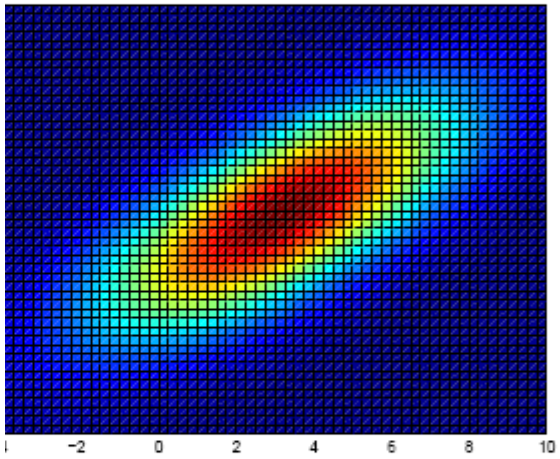
- Typically we want a combination of scales and orientations, different types of patterns.

Multivariate Gaussian

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







$$\Sigma = \begin{bmatrix} 9 & 0 \\ 0 & 9 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} 16 & 0 \\ 0 & 9 \end{bmatrix} \qquad \Sigma = \begin{bmatrix} 10 & 5 \\ 5 & 5 \end{bmatrix}$$

Filter bank

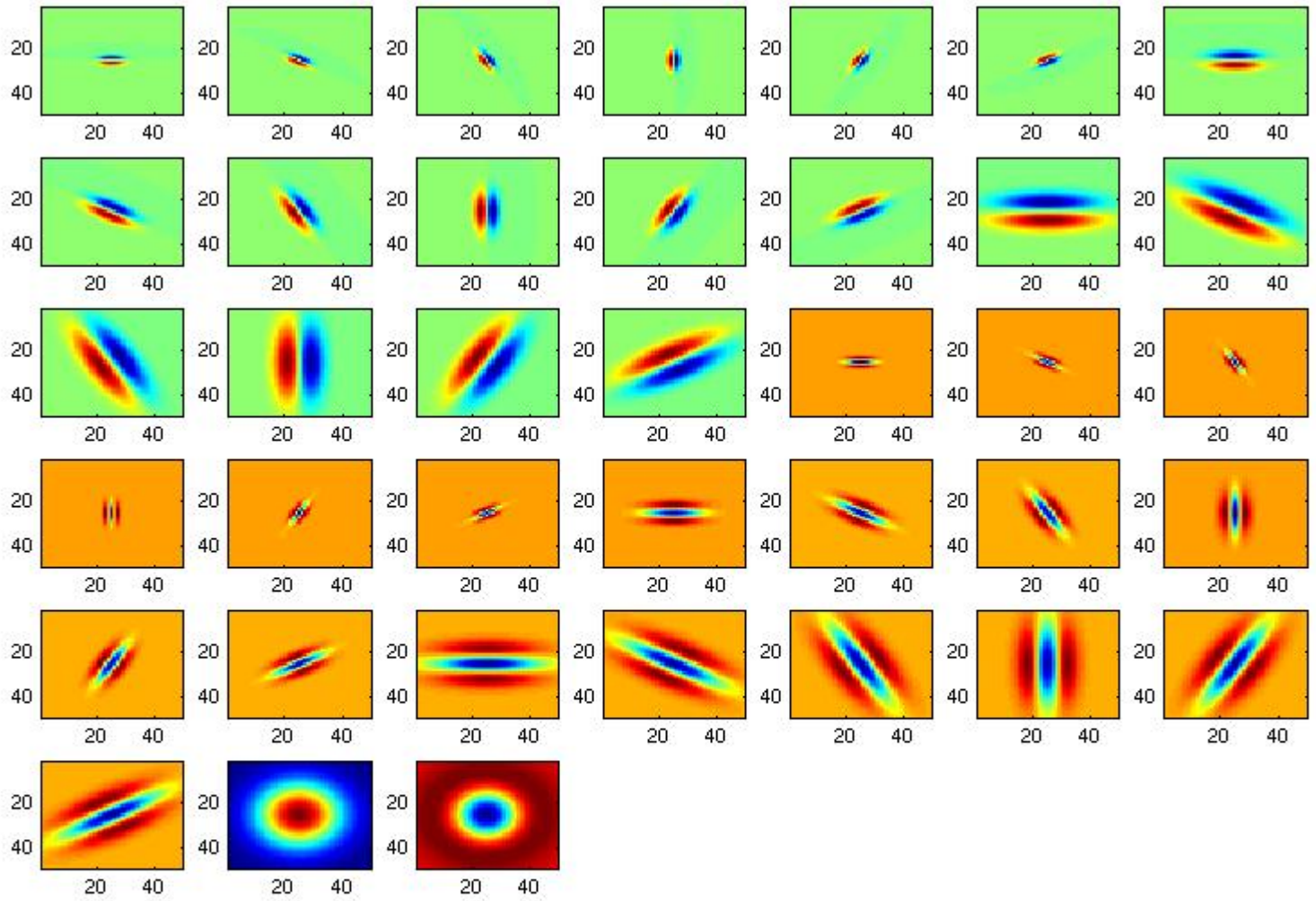
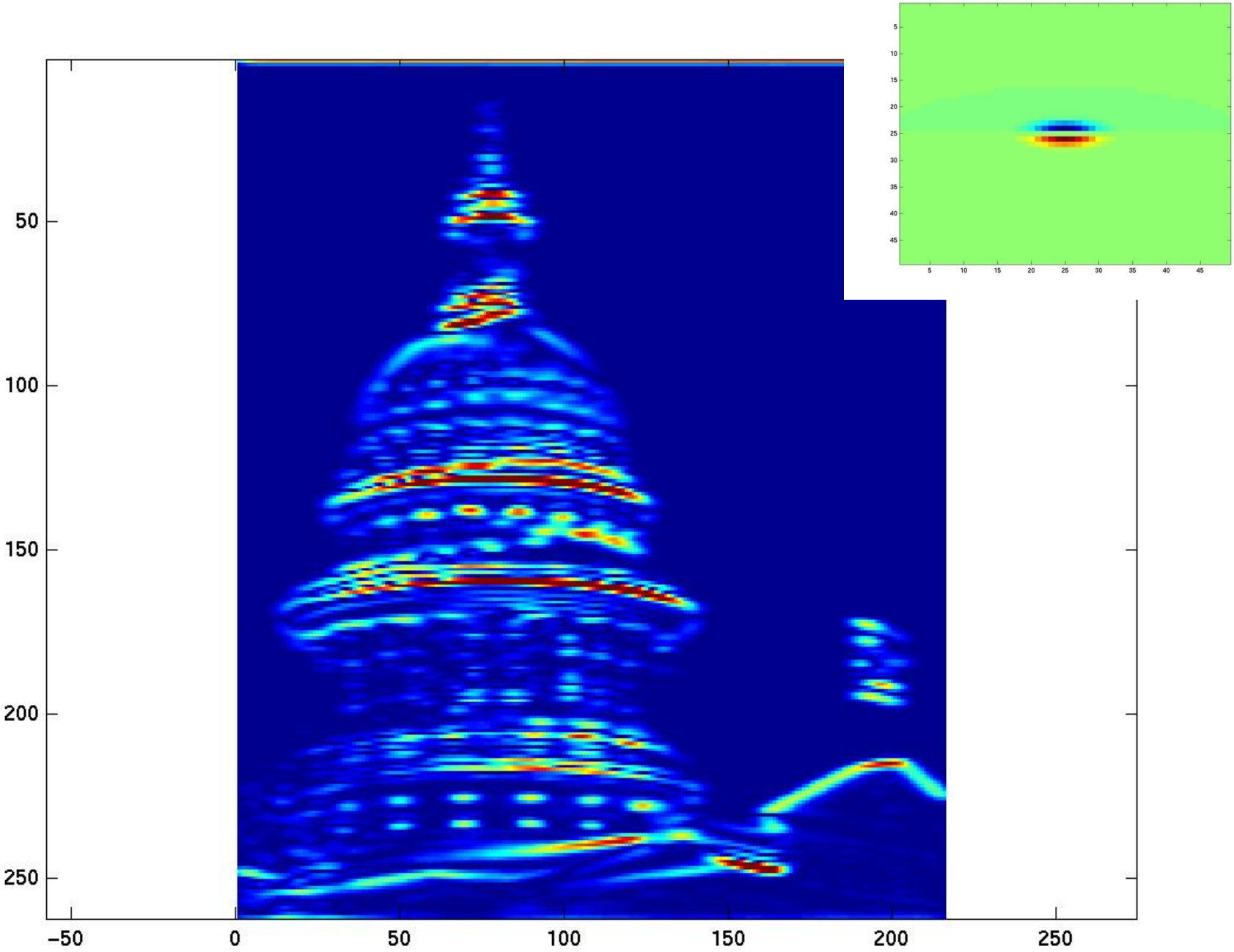
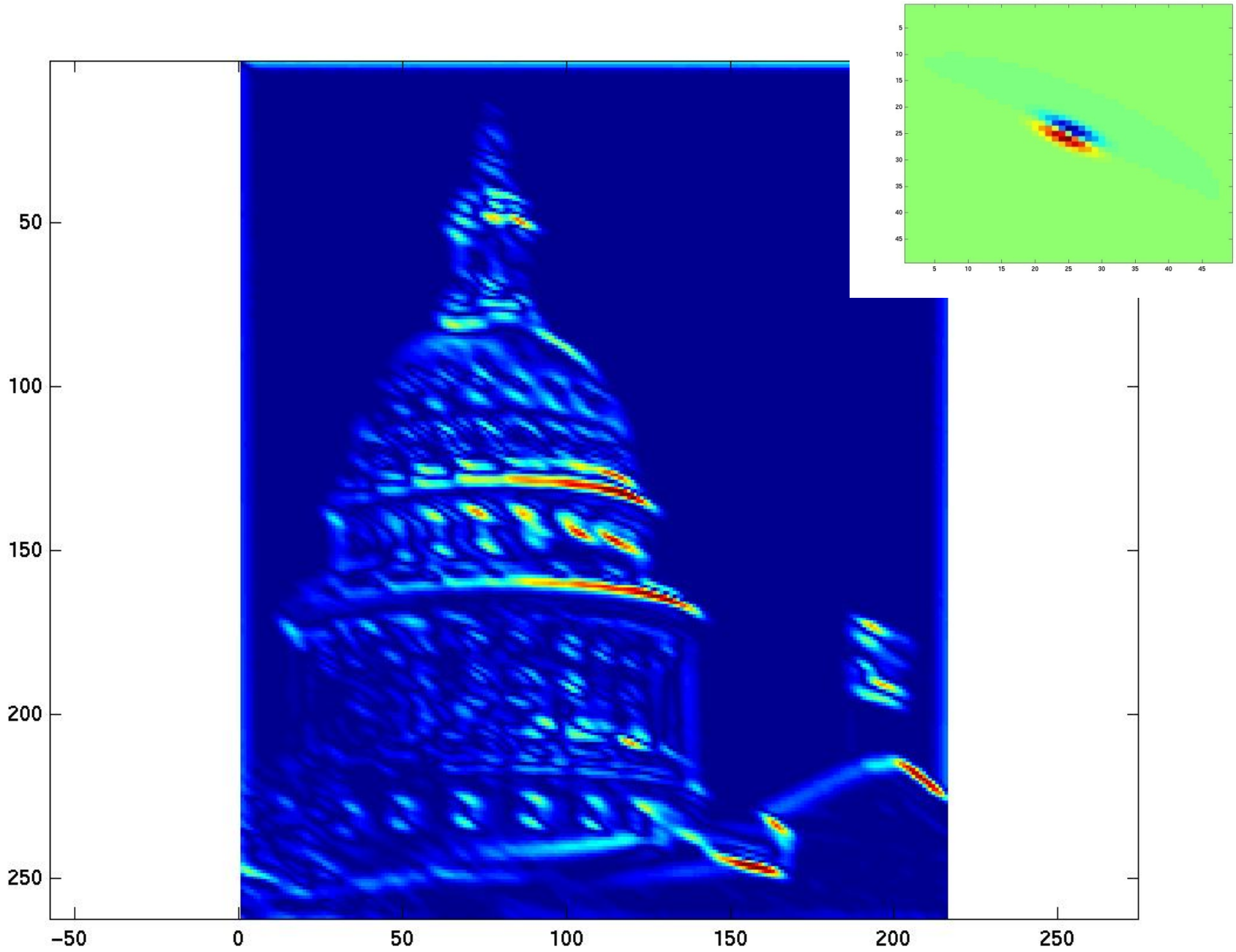


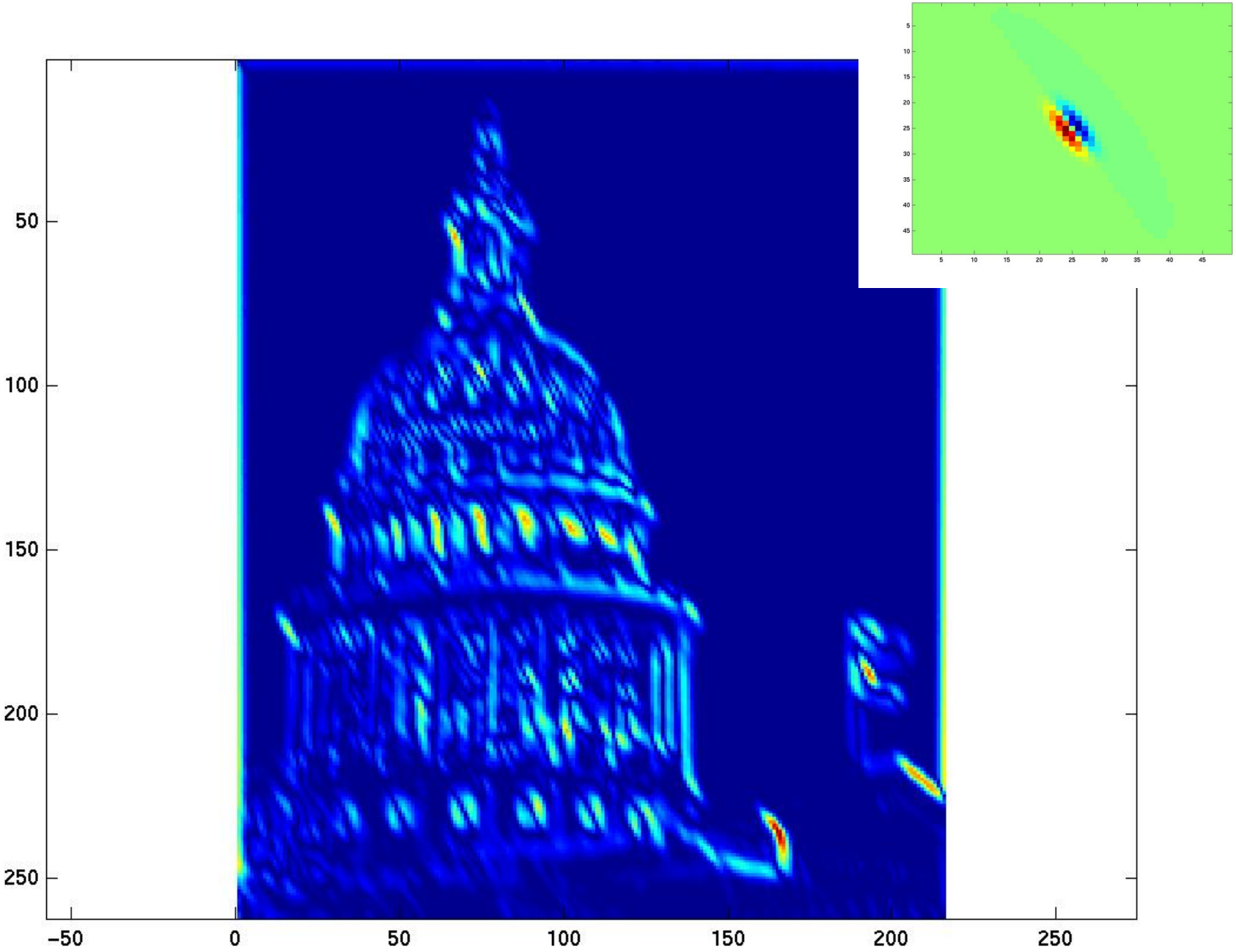
Image from http://www.texasexplorer.com/austincap2.jpg



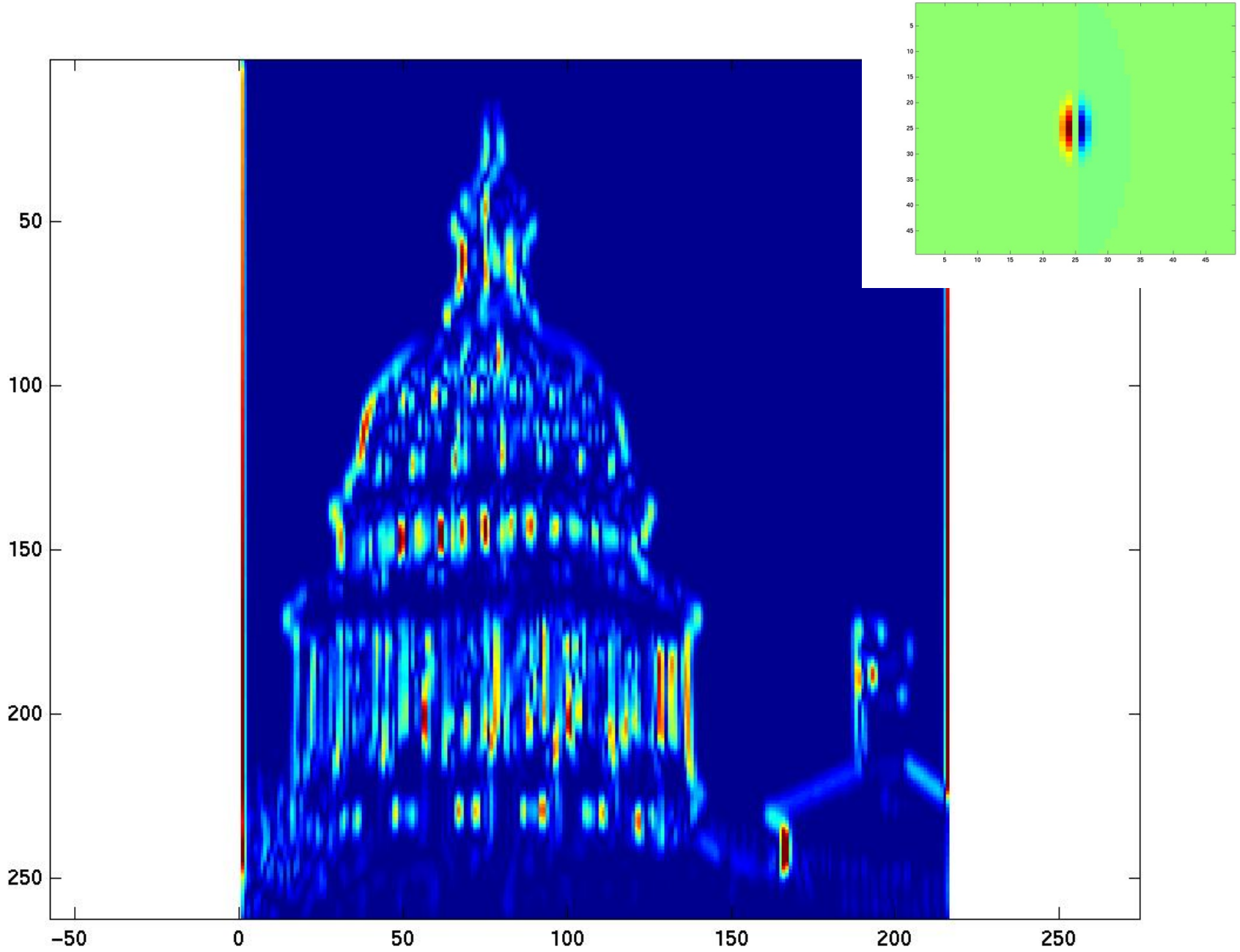
Kristen Grauman



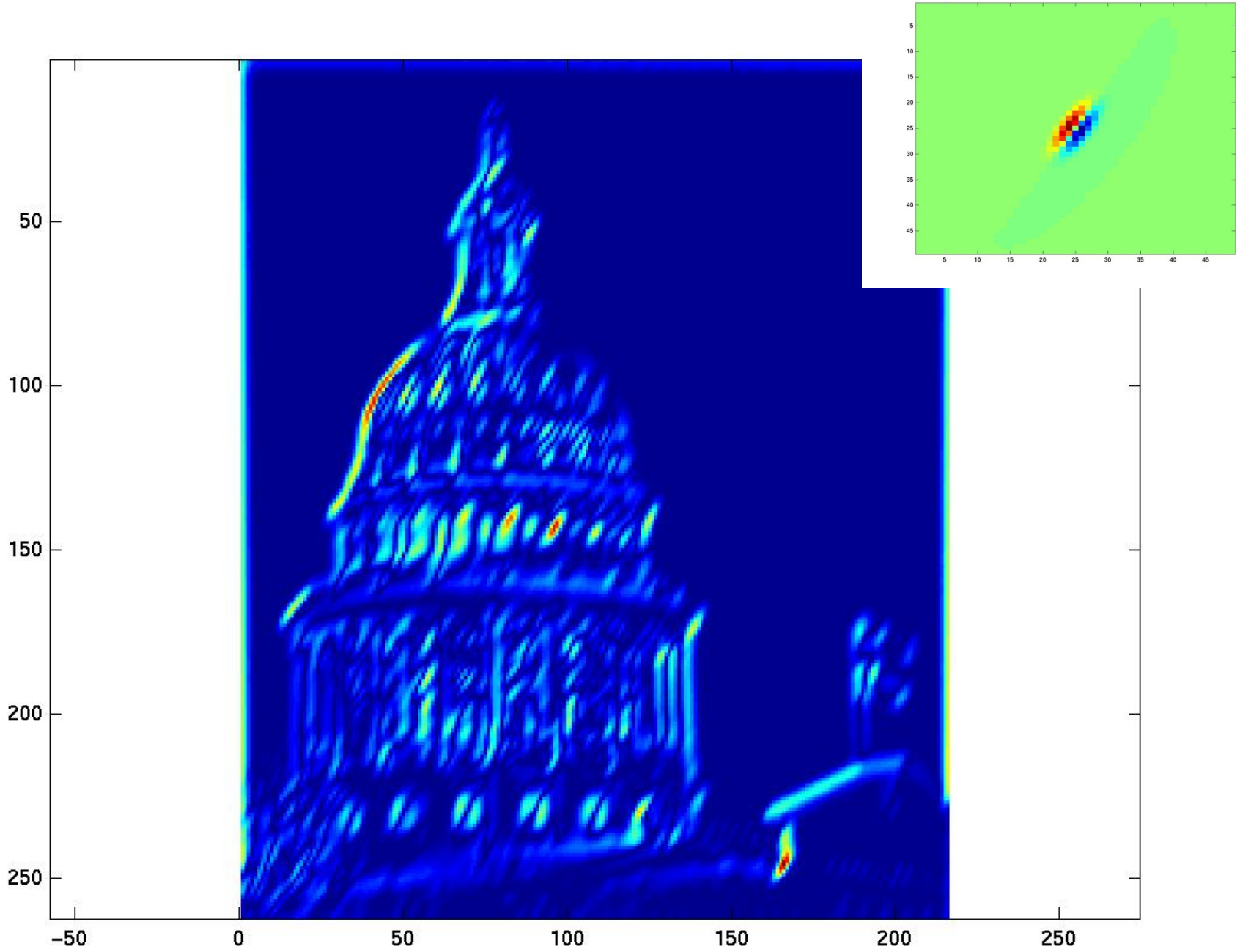
Kristen Grauman



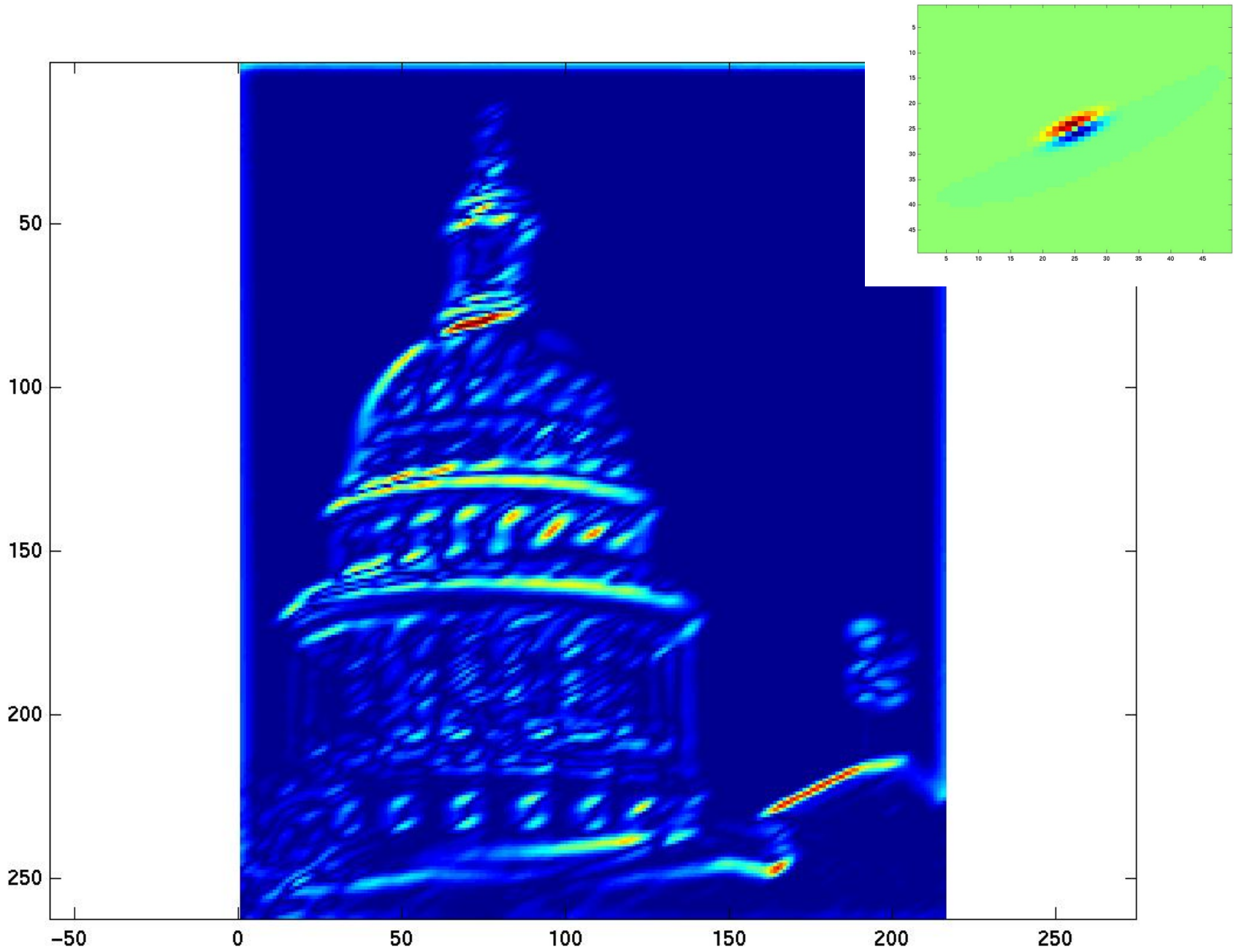
Kristen Grauman



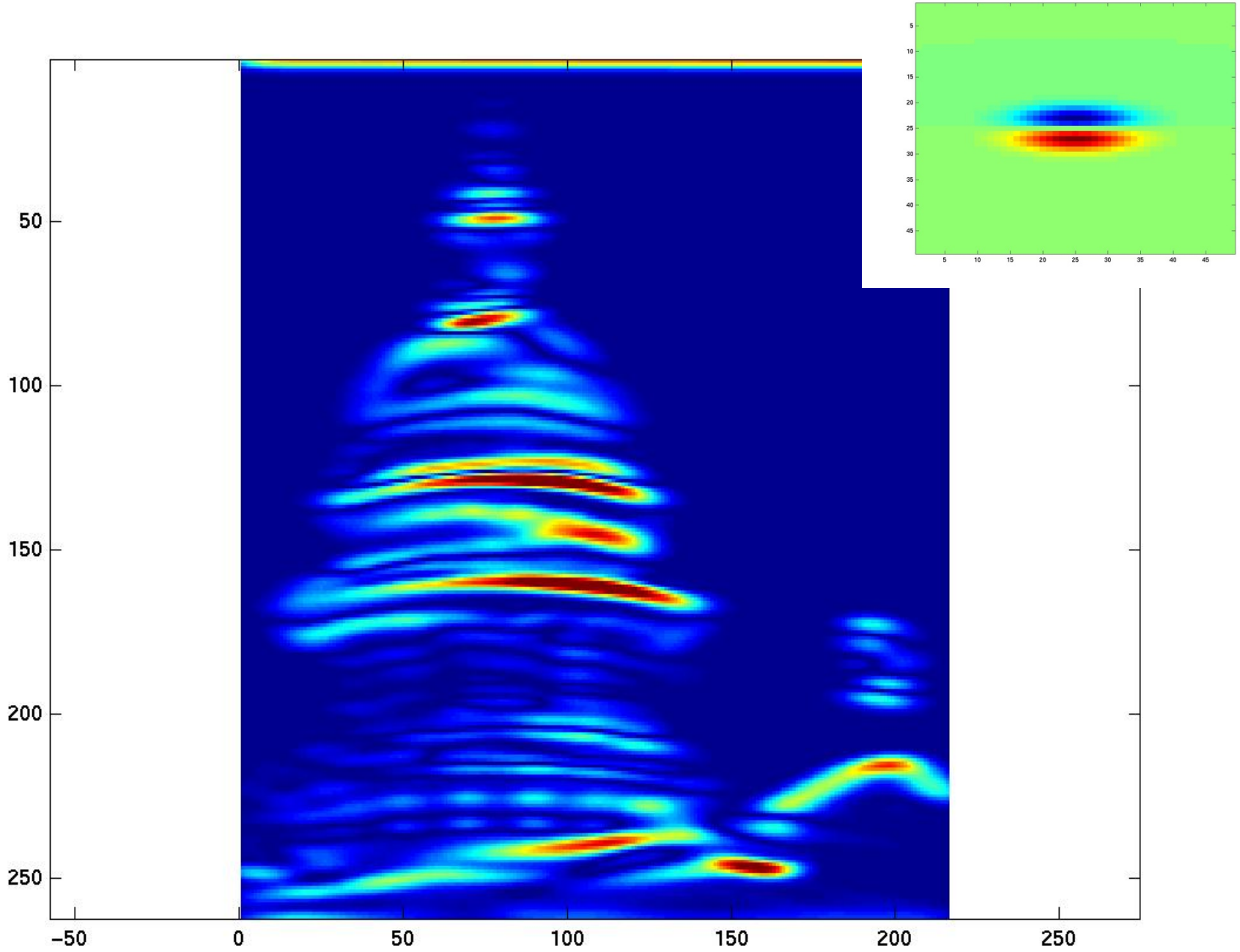
Kristen Grauman



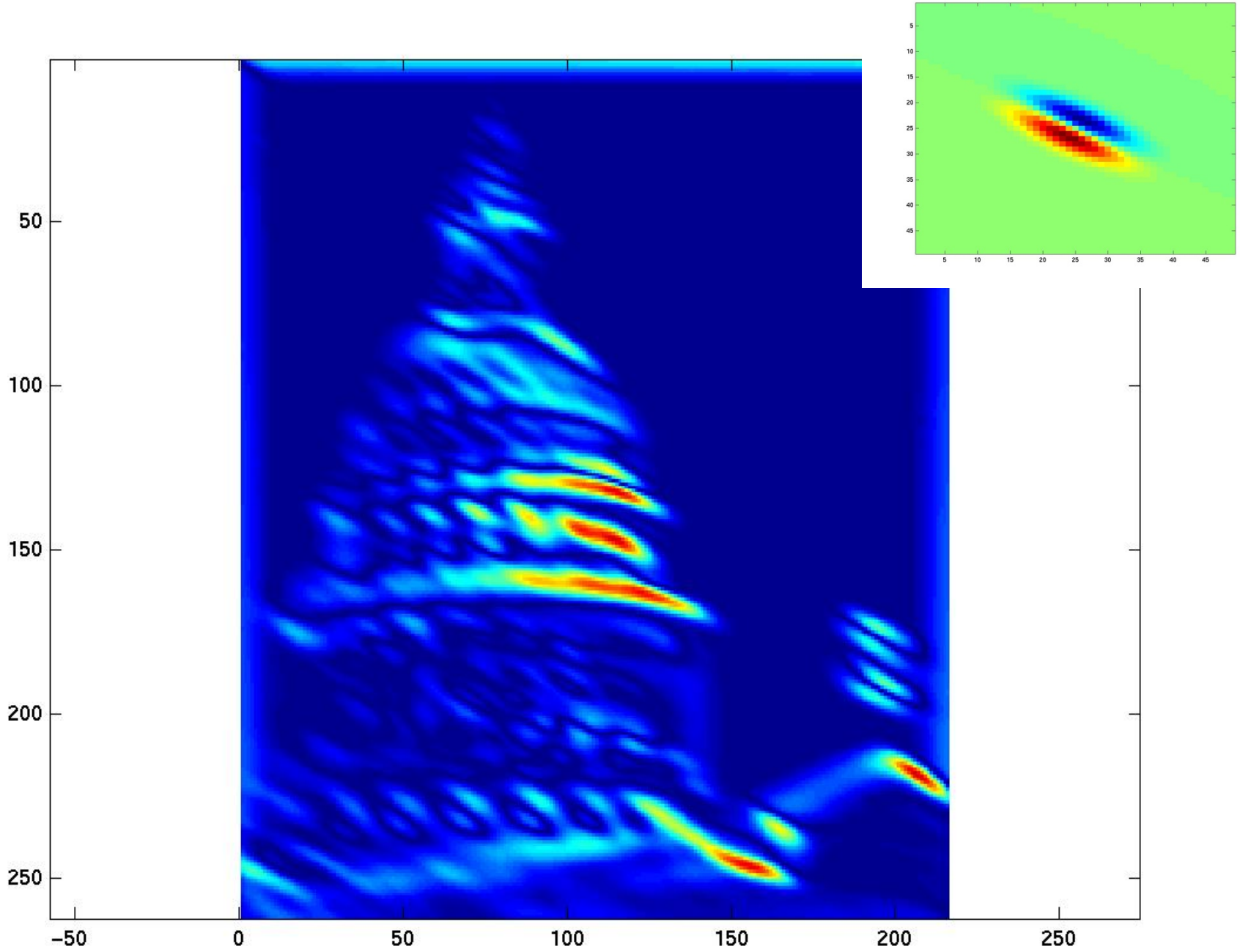
Kristen Grauman



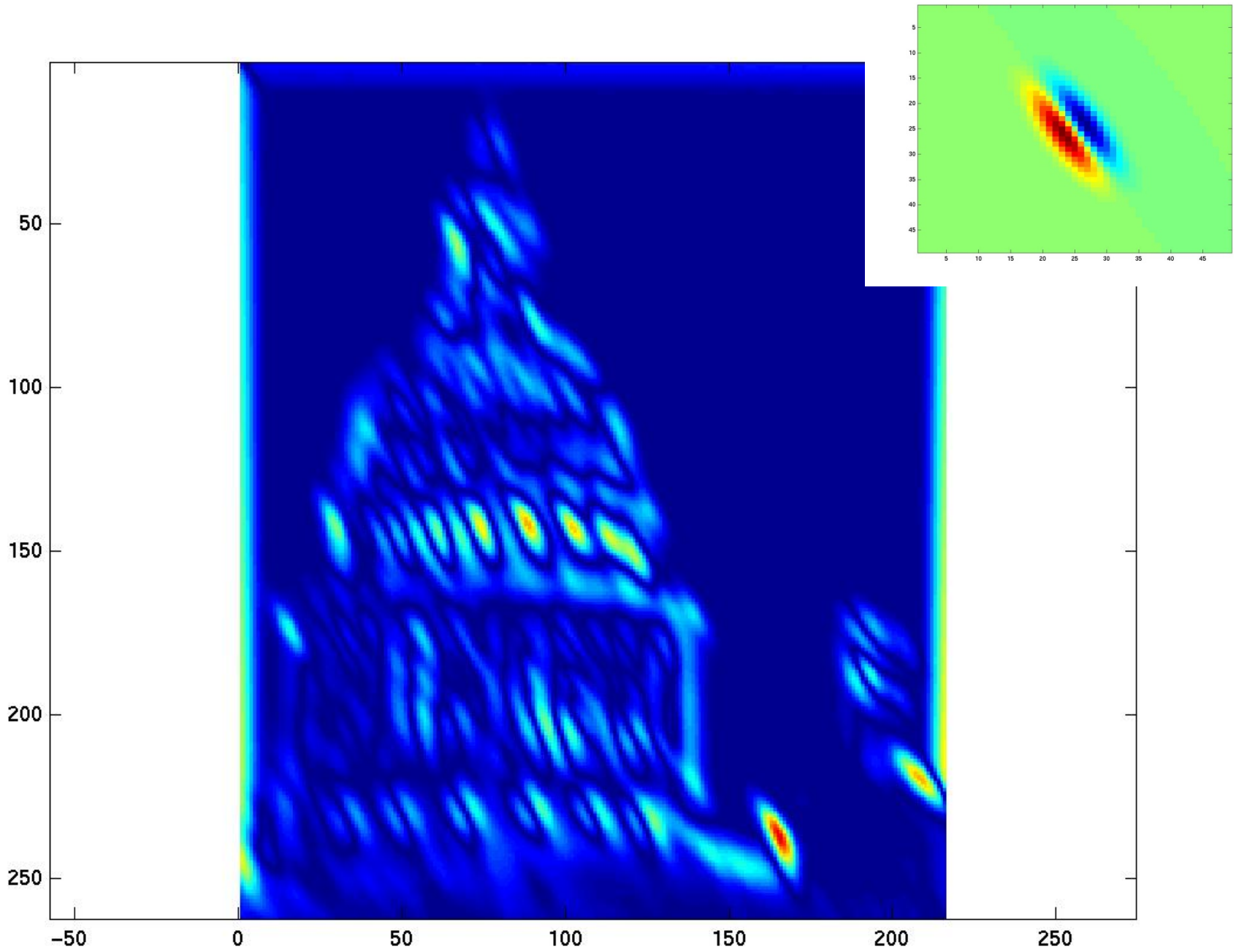
Kristen Grauman



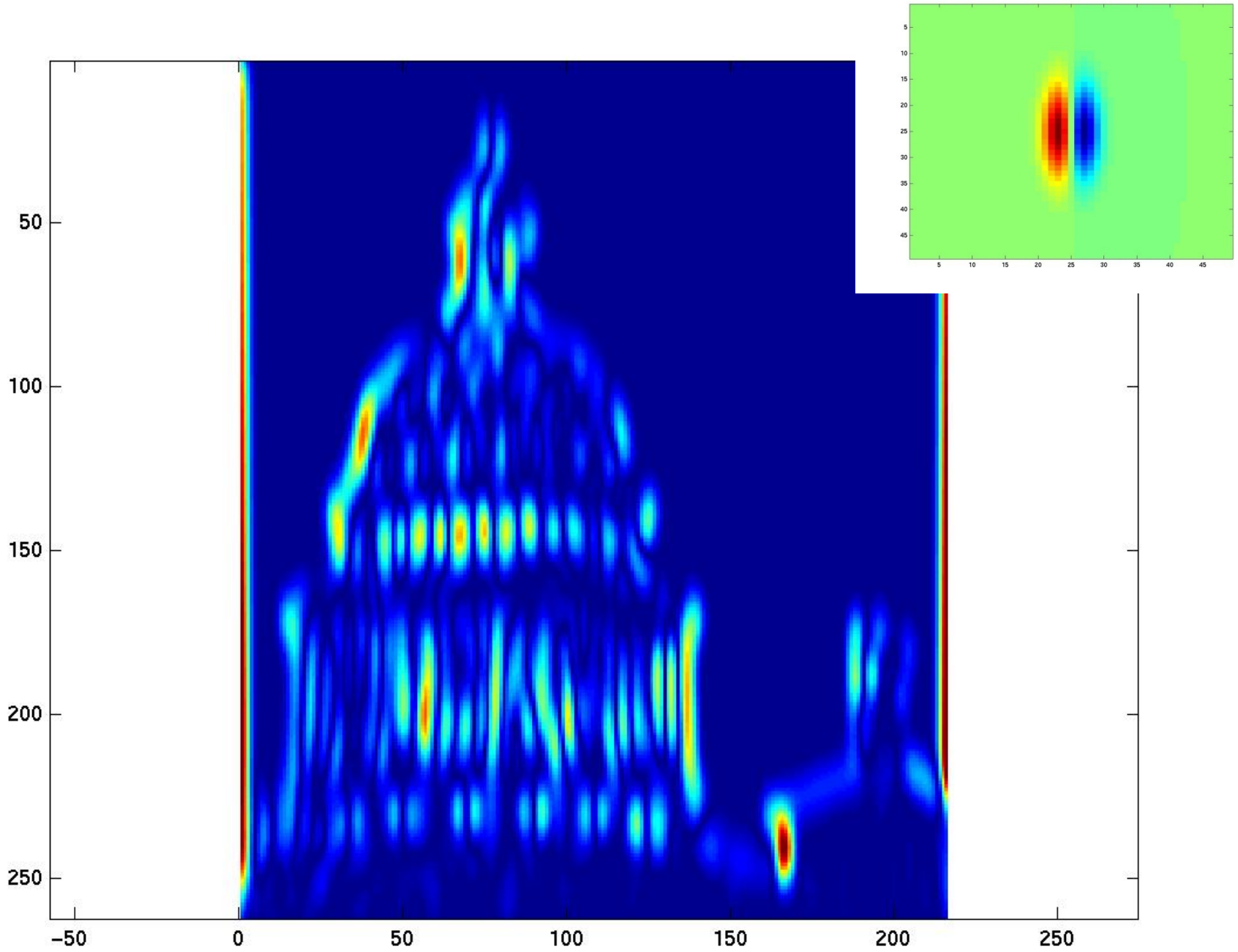
Kristen Grauman



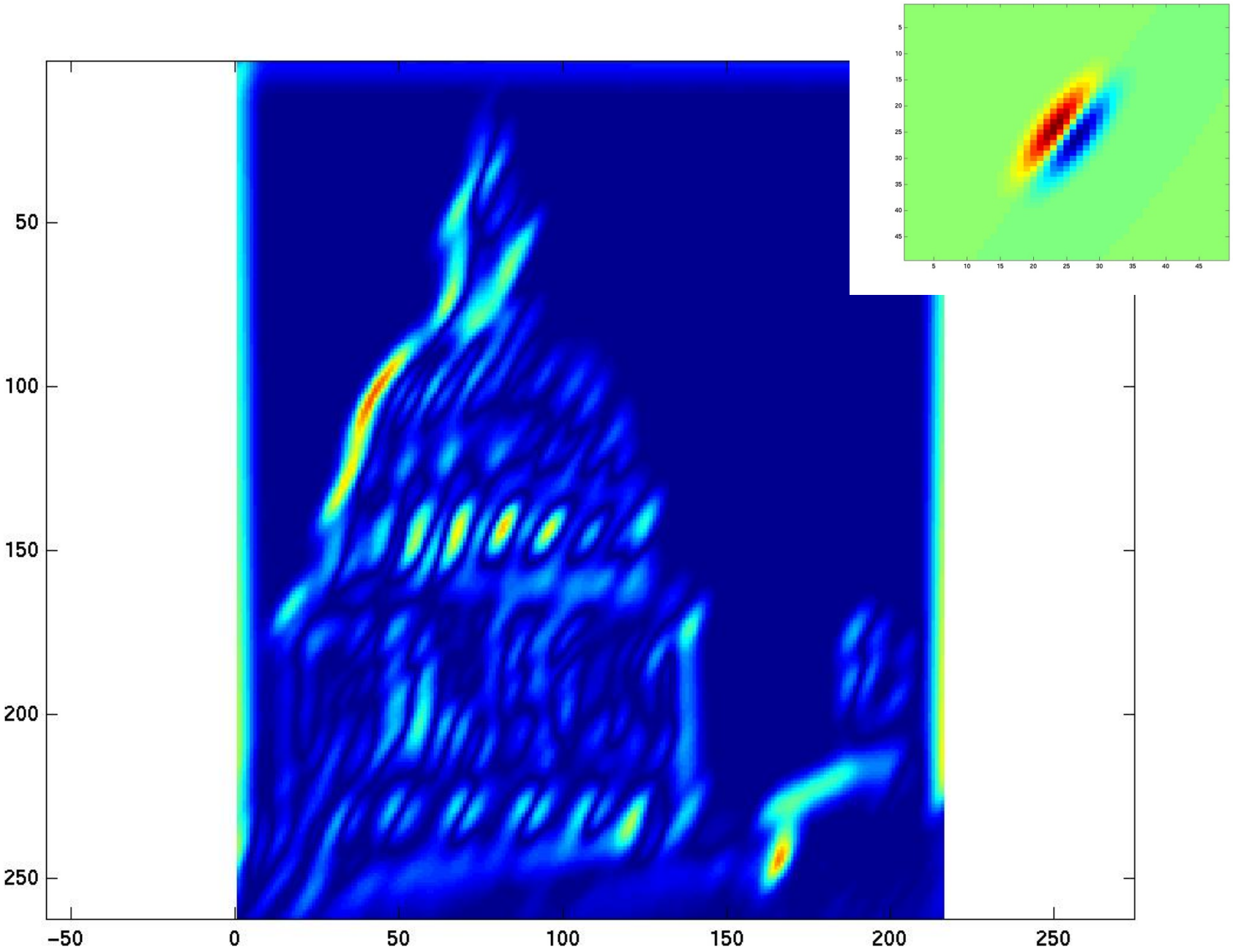
Kristen Grauman



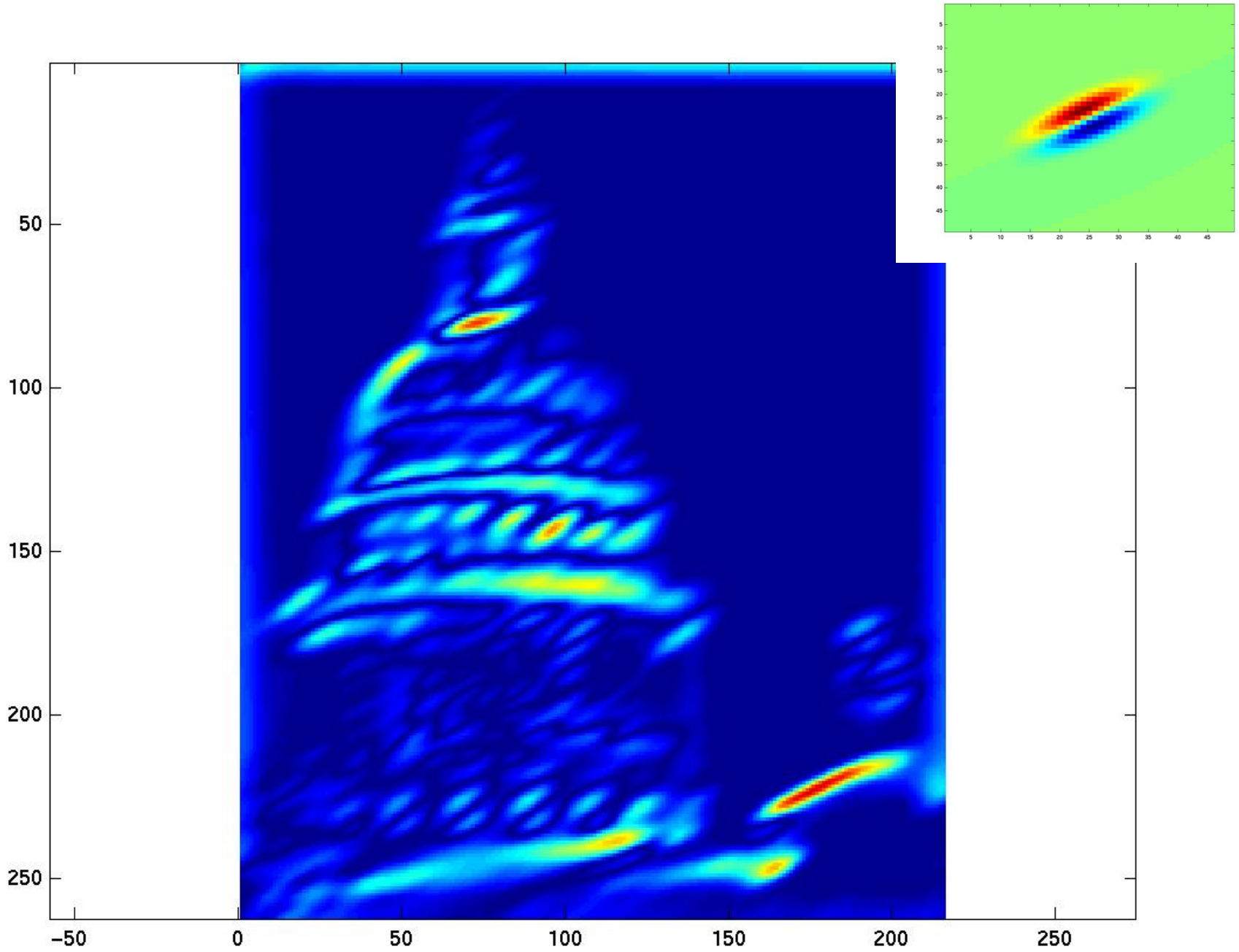
Kristen Grauman



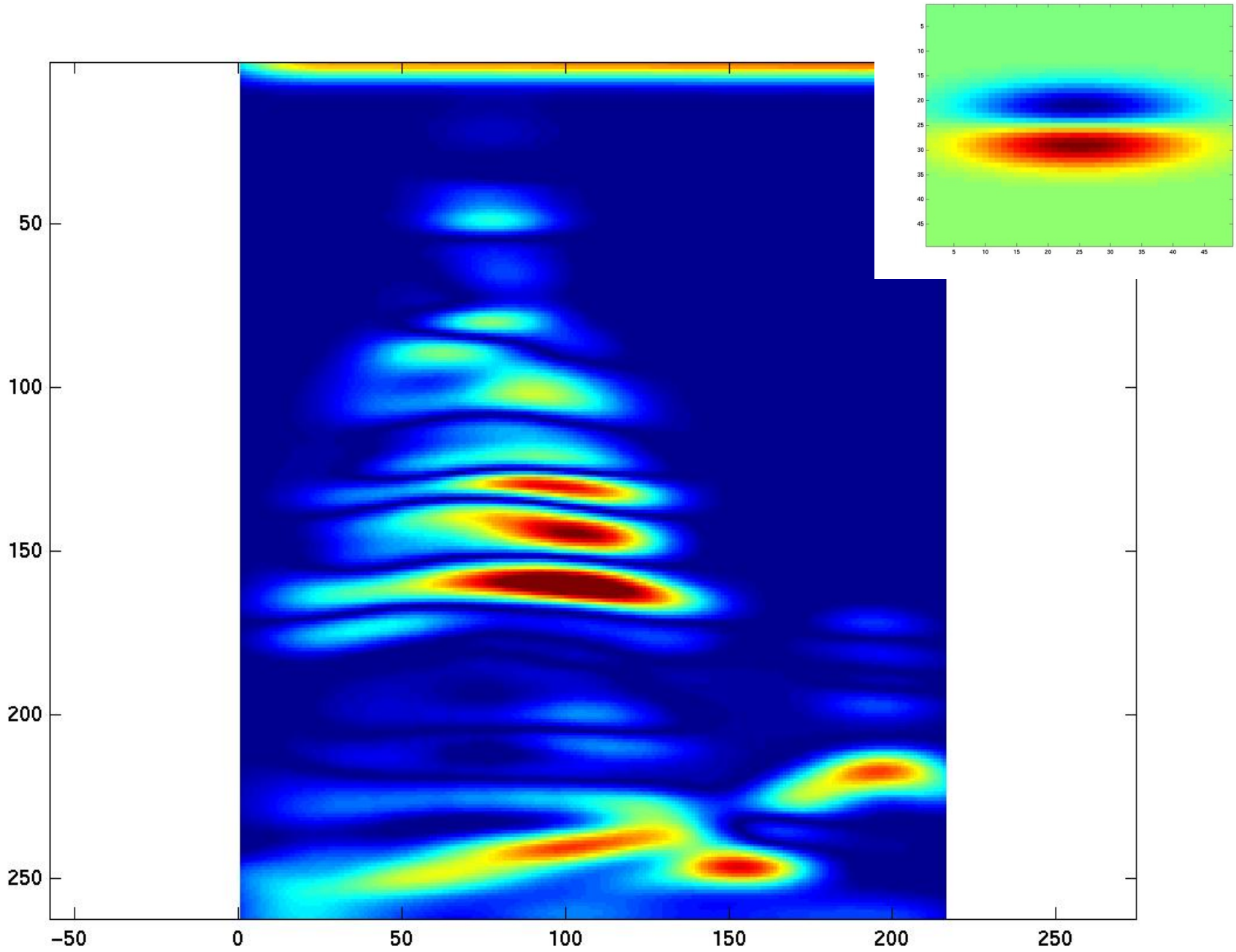
Kristen Grauman



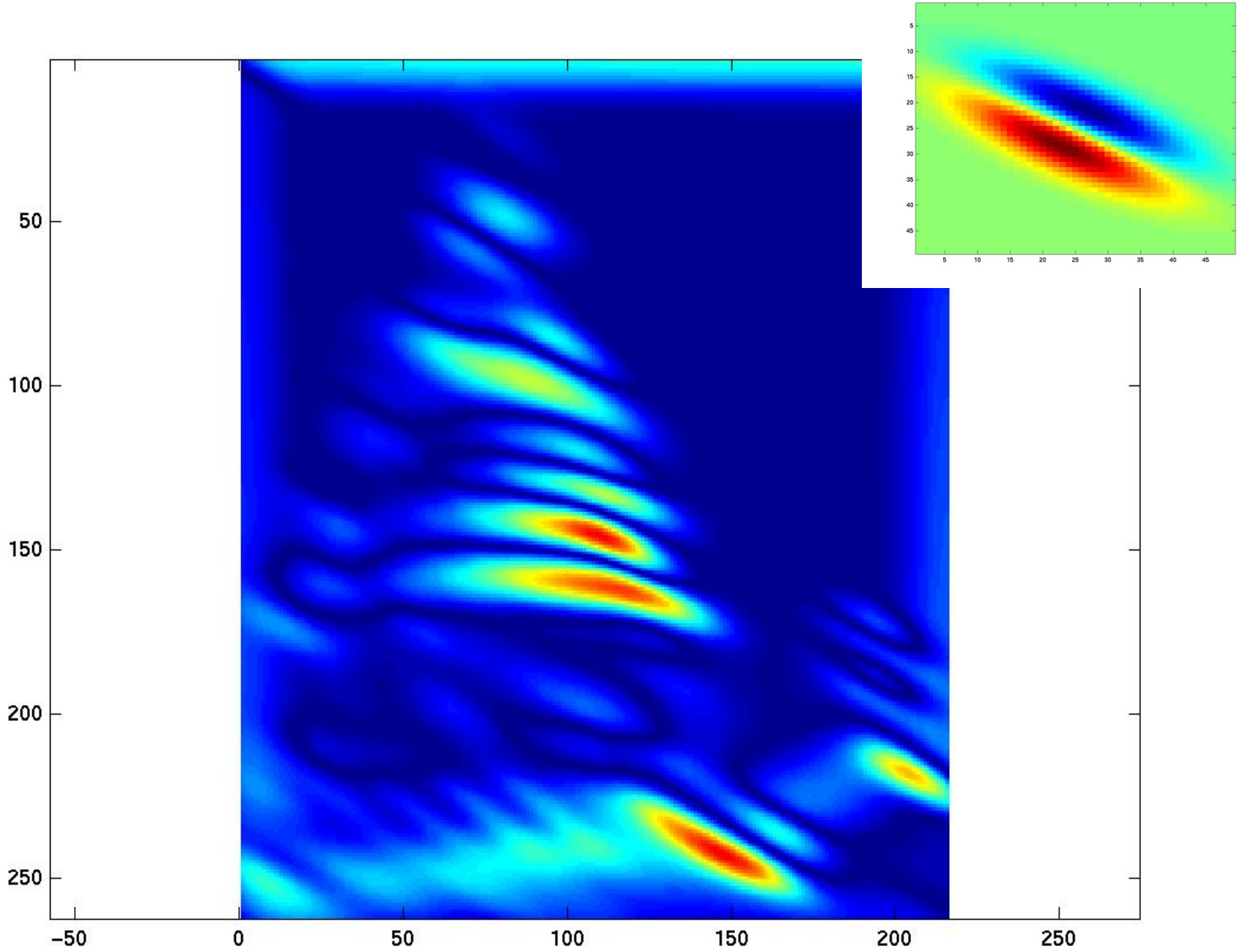
Kristen Grauman



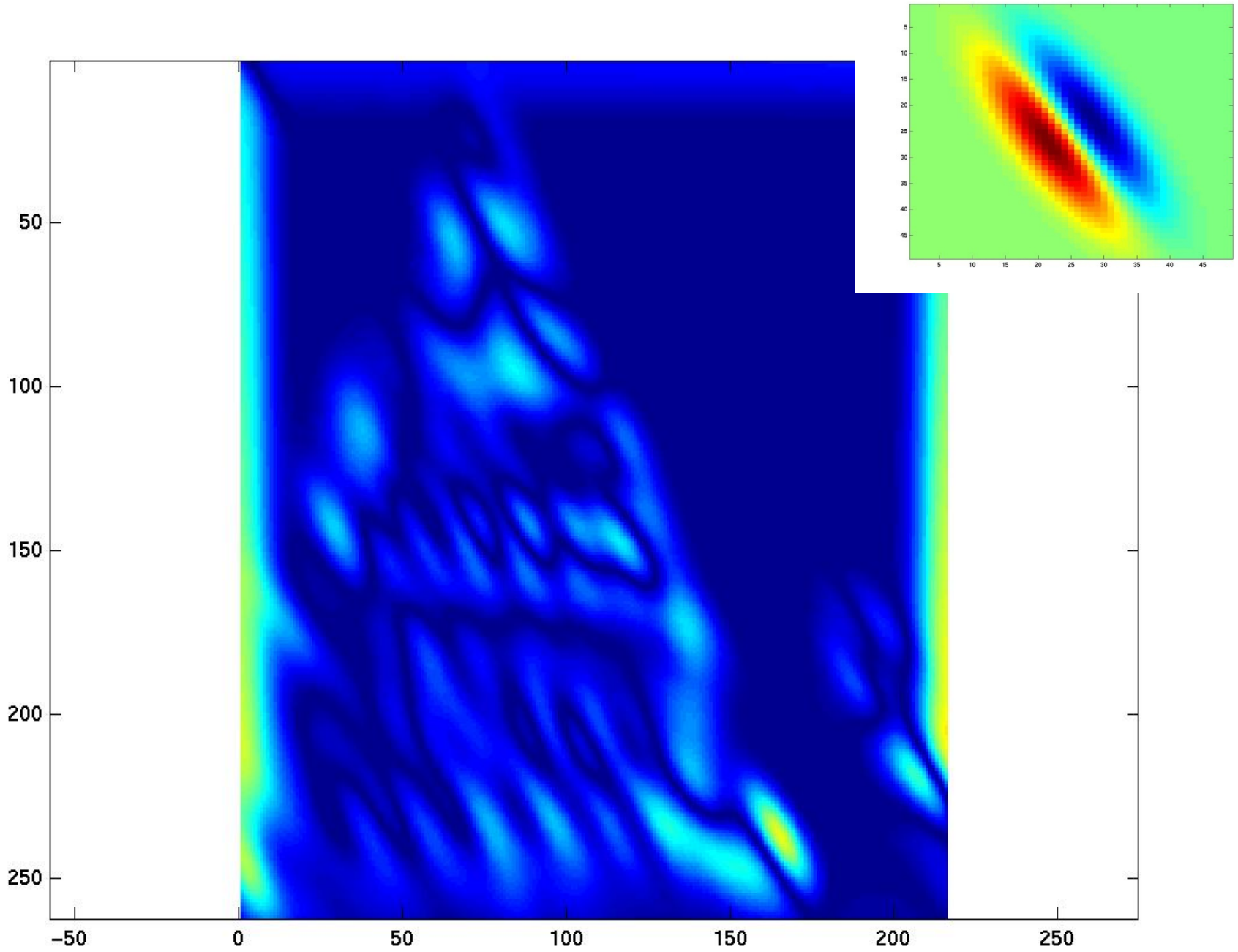
Kristen Grauman



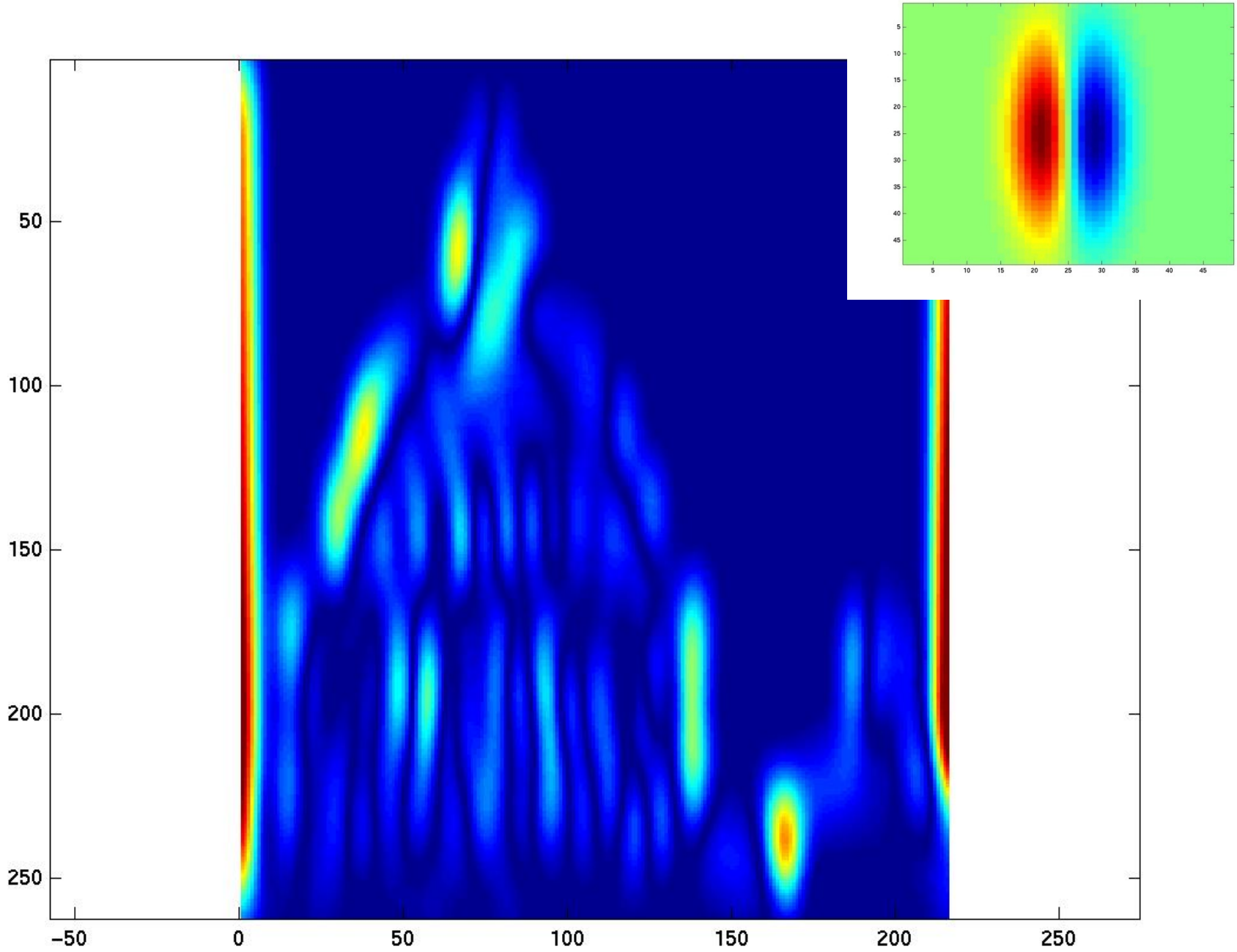
Kristen Grauman



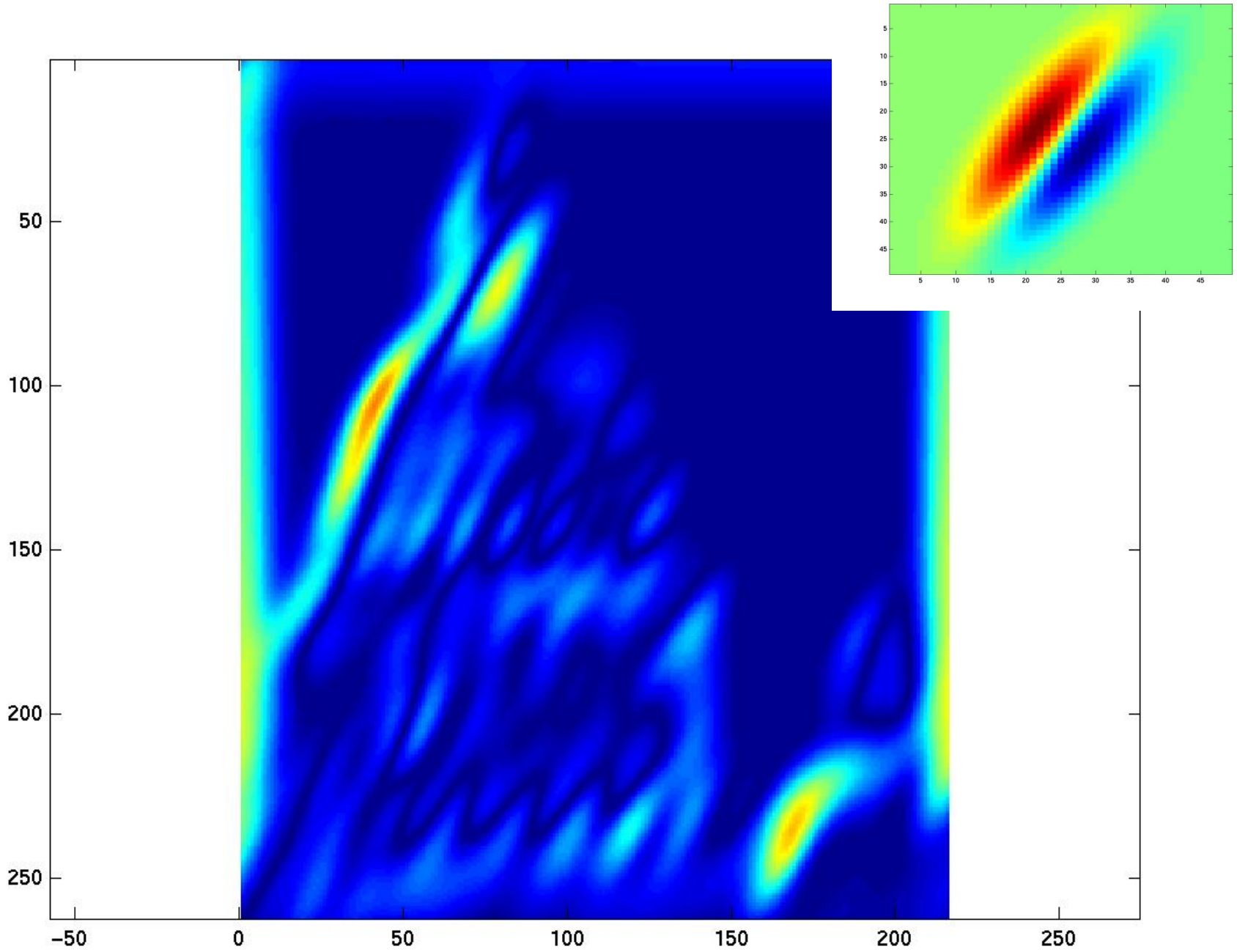
Kristen Grauman



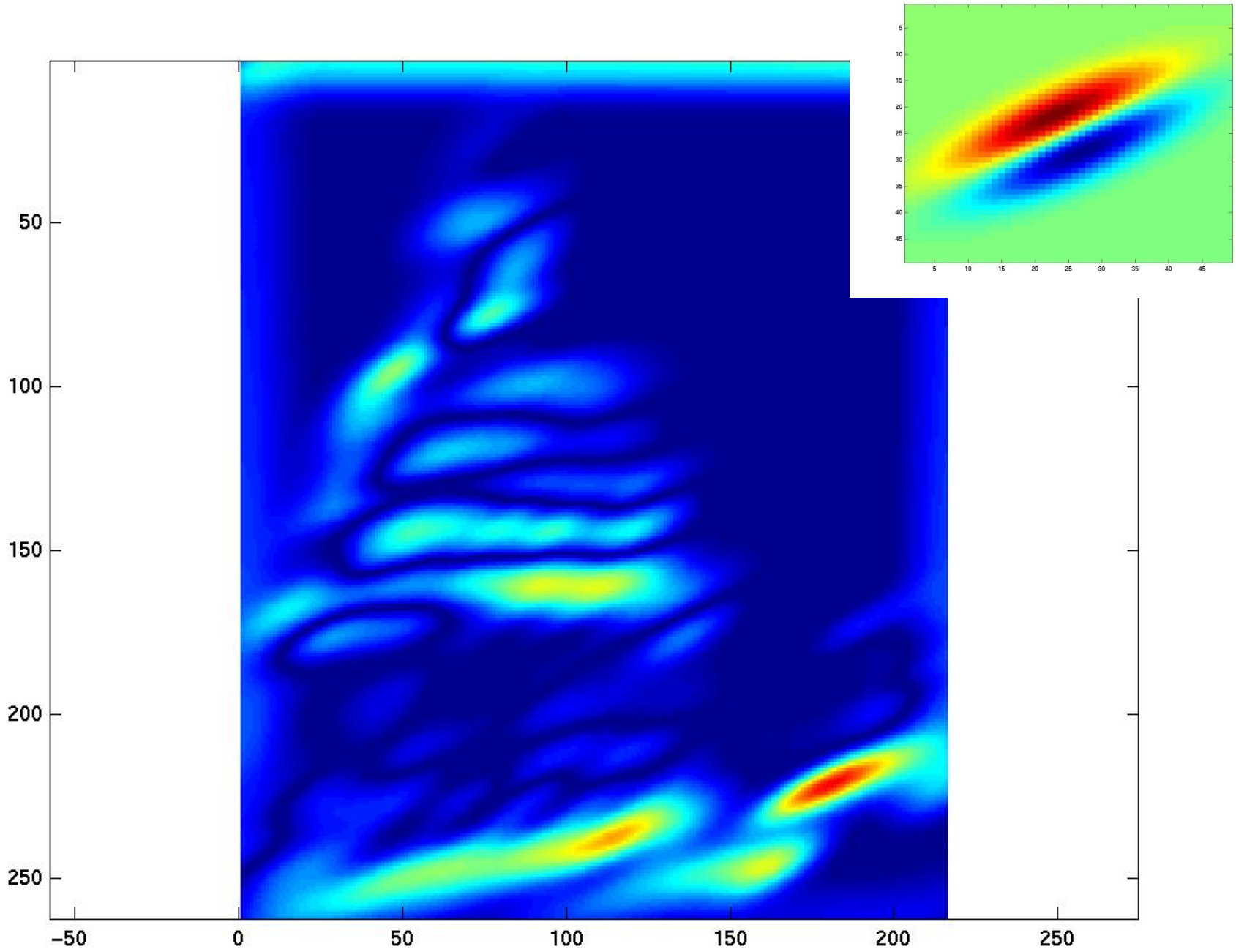
Kristen Grauman



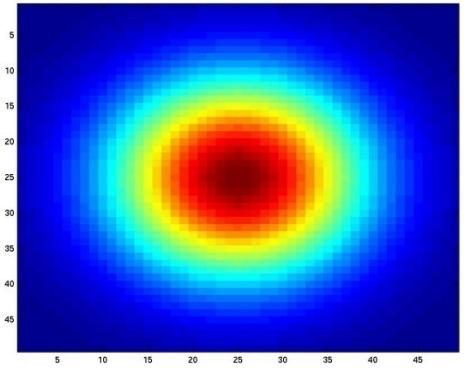
Kristen Grauman



Kristen Grauman

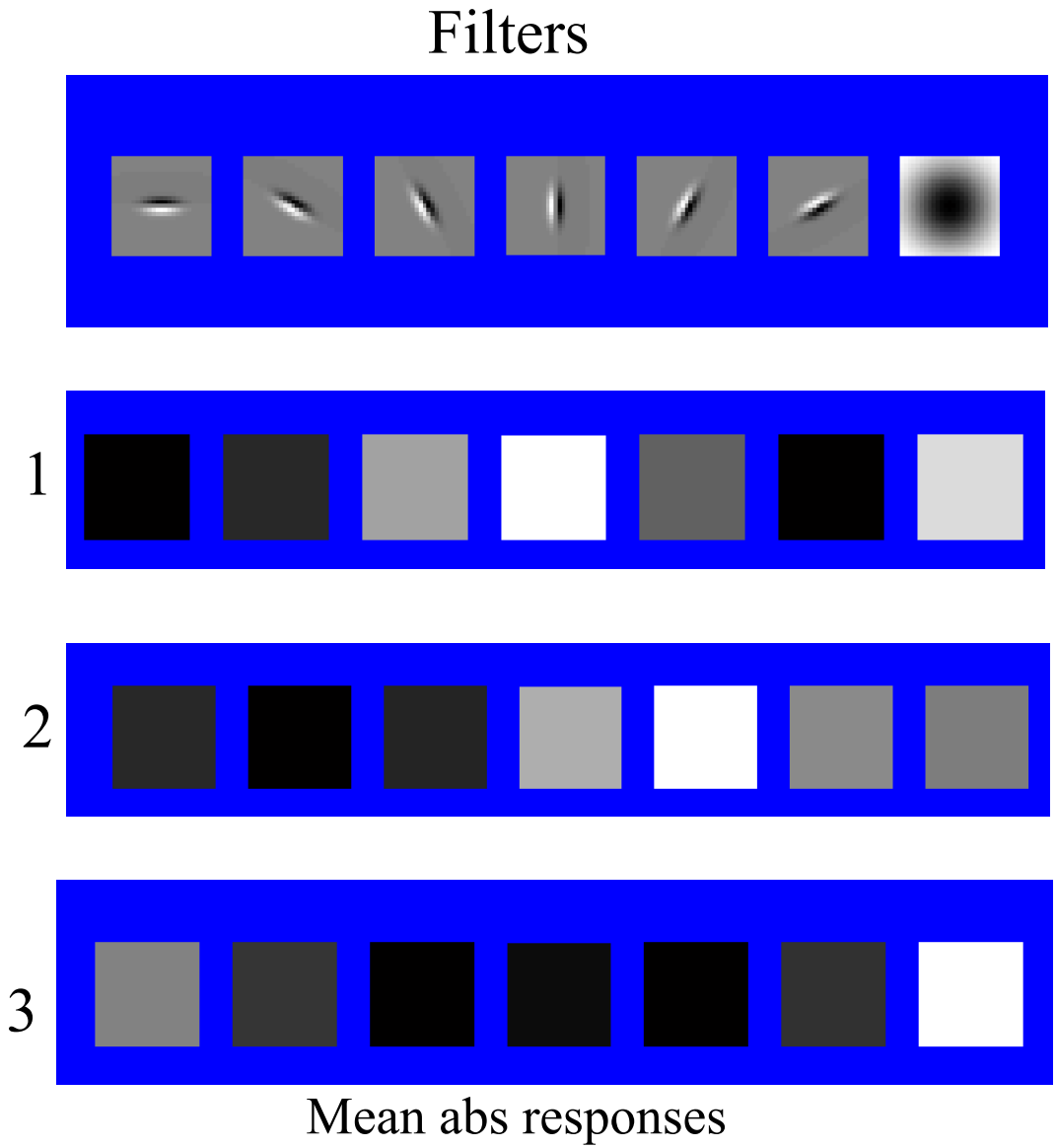


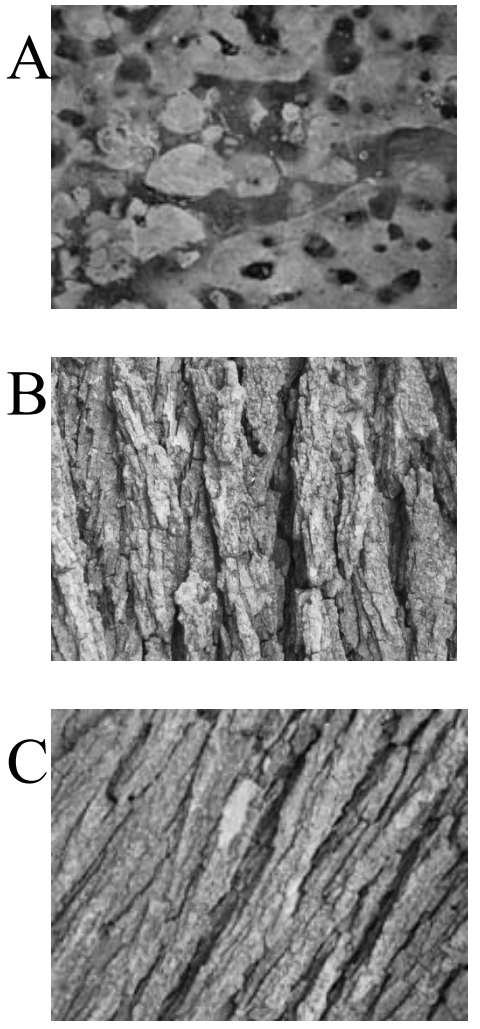
Kristen Grauman



Kristen Grauman

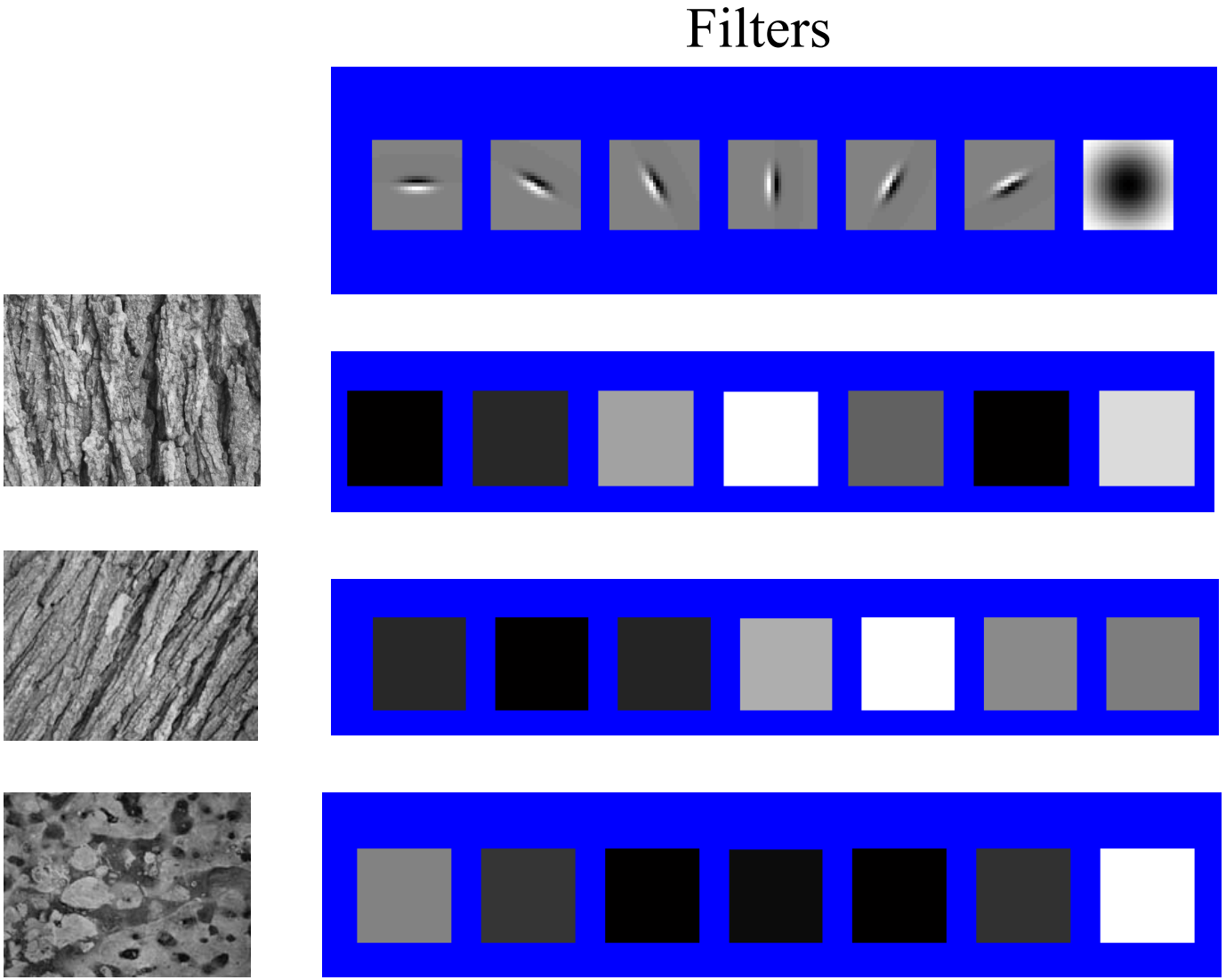
You try: Can you match the texture to the response?



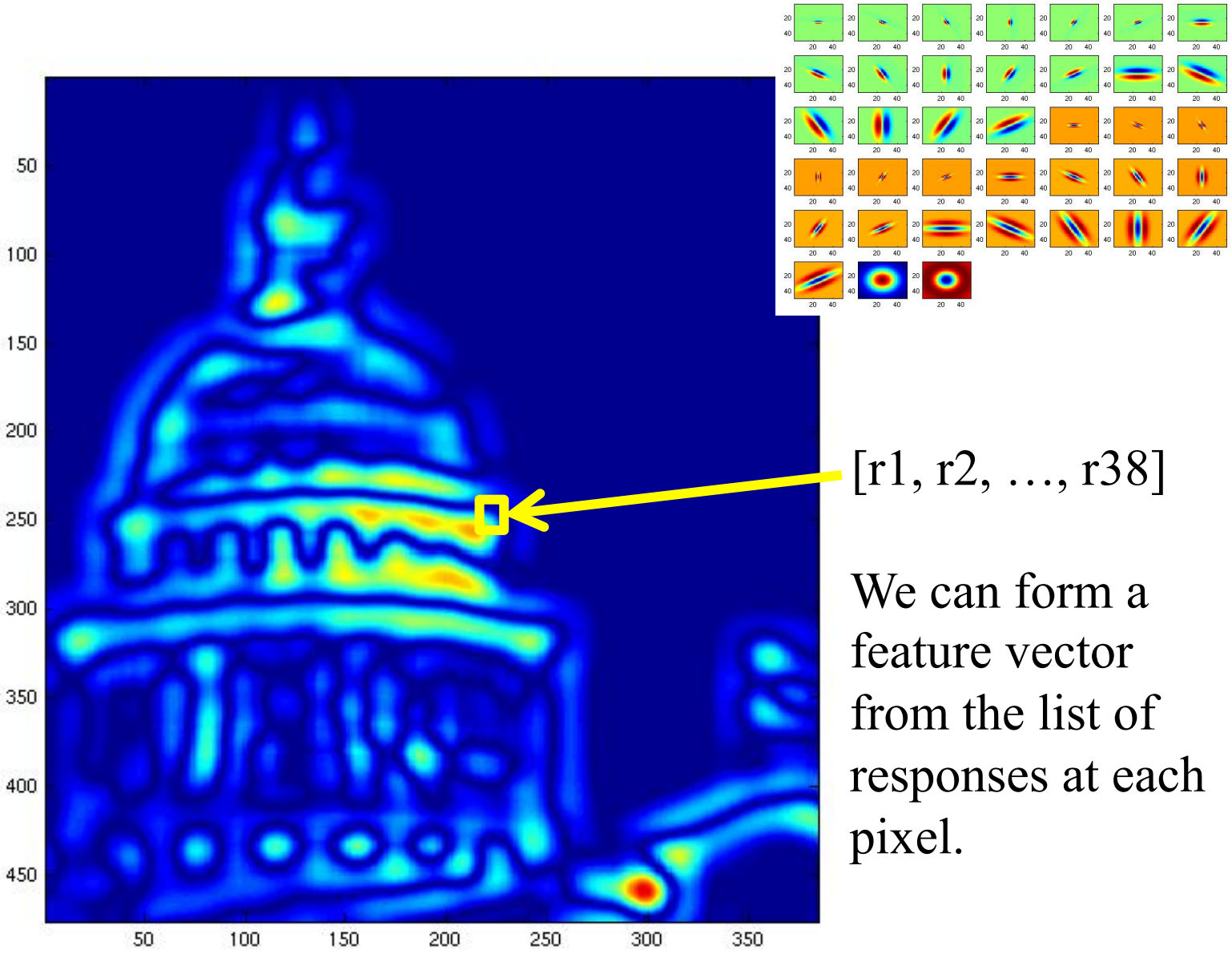


Derek

Representing texture by mean abs response



Derek



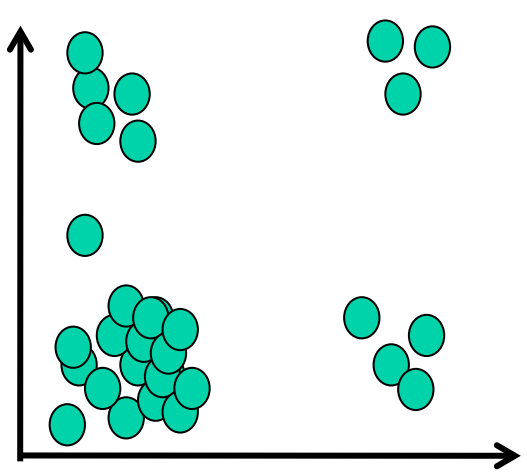
[r1, r2, …, r38]

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

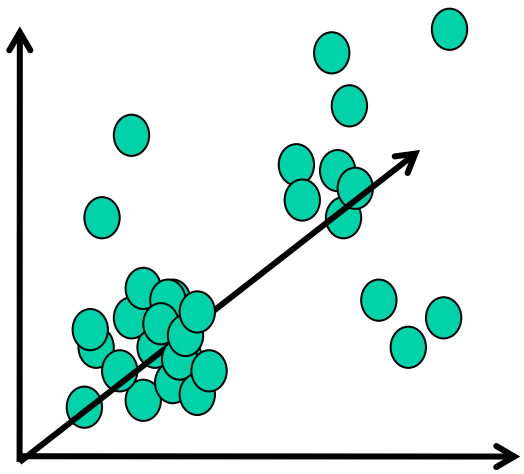
d-dimensional features

$$D(a,b) = \sqrt{\sum_{i=1}^{d} (a_i - b_i)^2}$$

Euclidean distance (L_2)

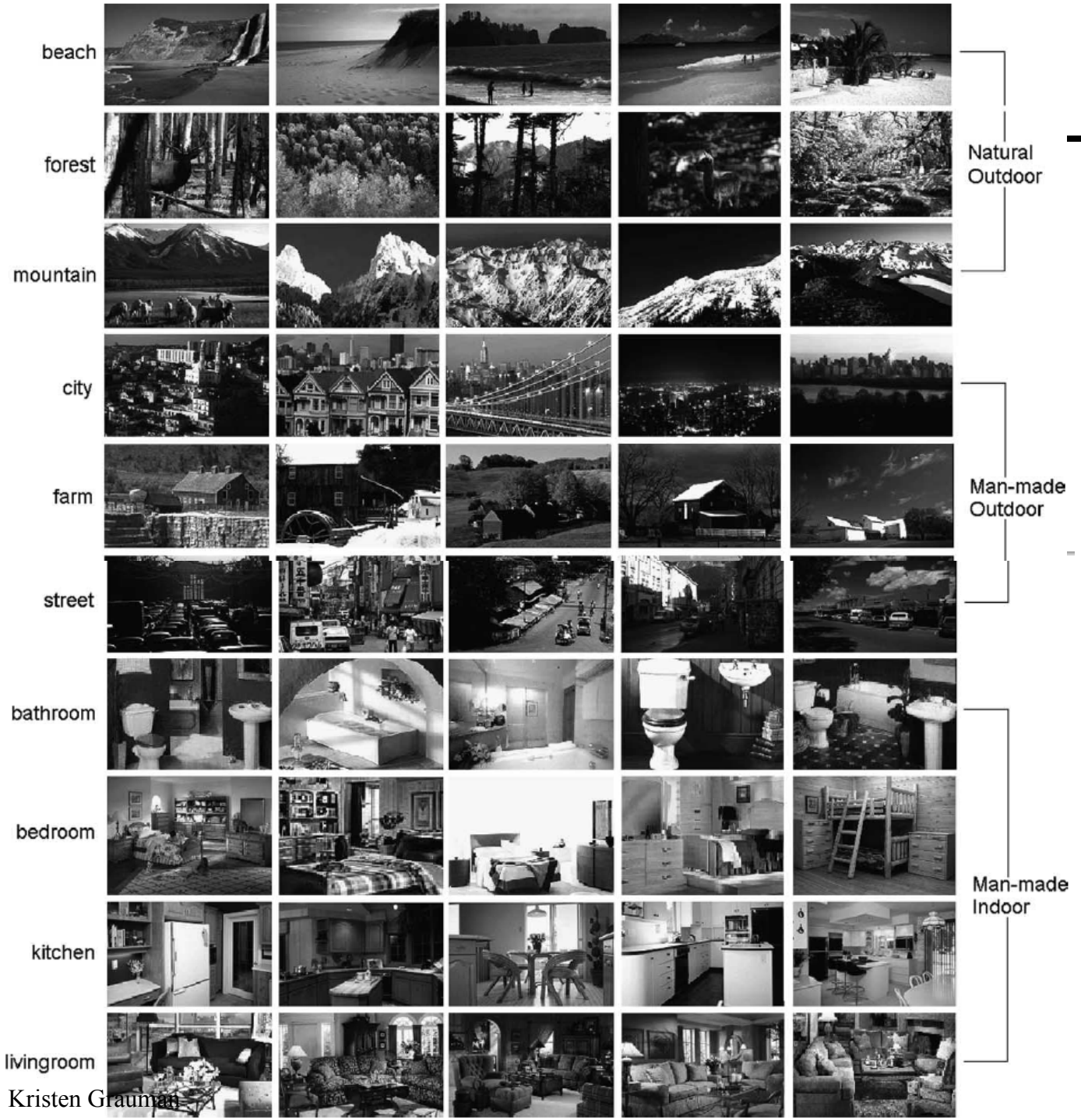


2d



3d

. . .



Example: characterizing scene categories by texture

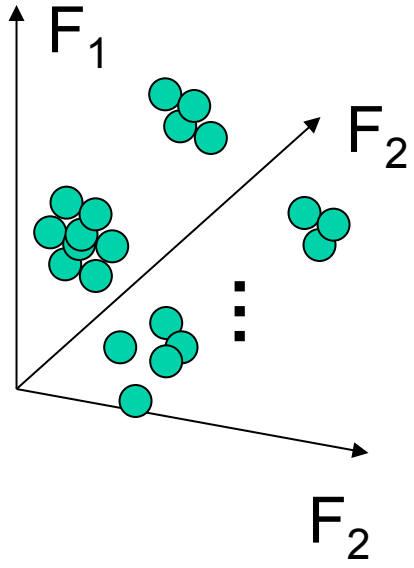
L. W. Renninger and J. Malik. When is scene identification just texture recognition? Vision Research 44 (2004) 2301–2311

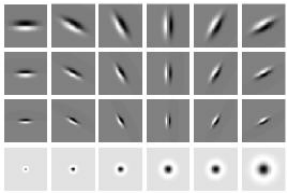
Back to segmentation

Segmentation as clustering

Depending on what we choose as the *feature space*, we can group pixels in different ways.

Grouping pixels based on **texture** similarity





Filter bank of 24 filters

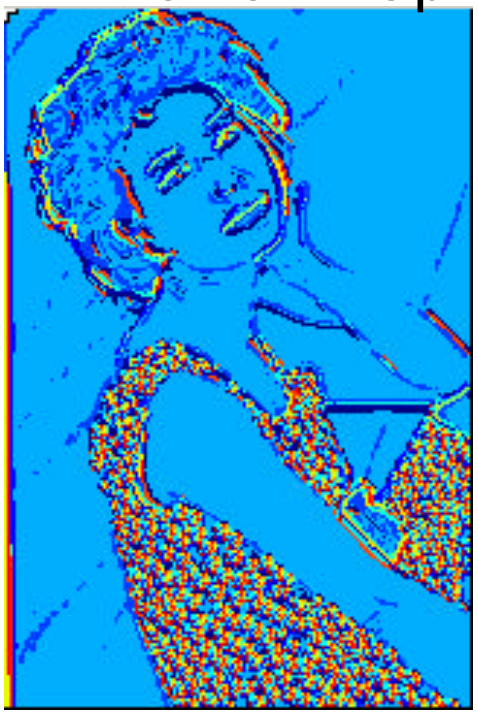
Feature space: filter bank responses (e.g., 24-d)

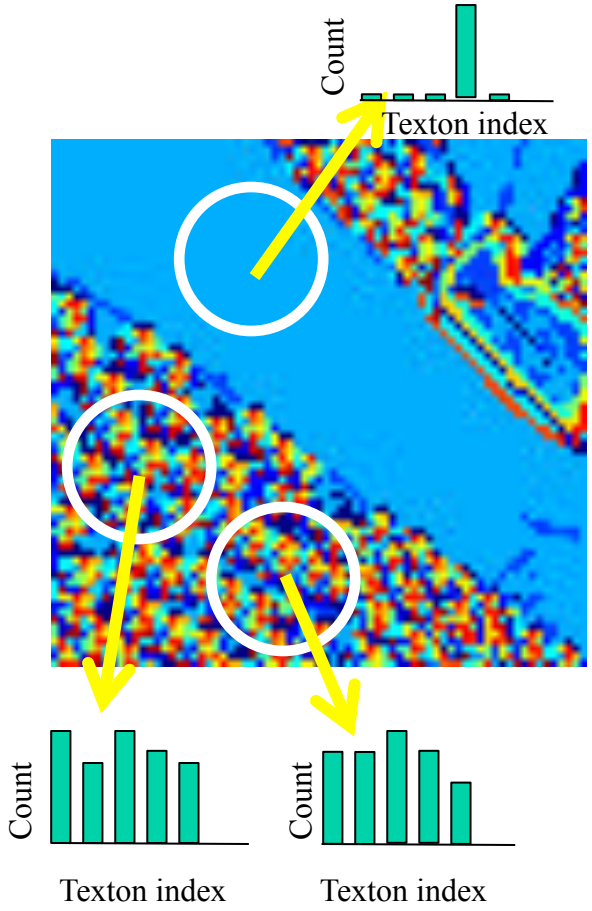
Segmentation with texture features

- Find “textons” by **clustering** vectors of filter bank outputs
- Describe texture in a window based on *texton histogram*

Image

Texton map

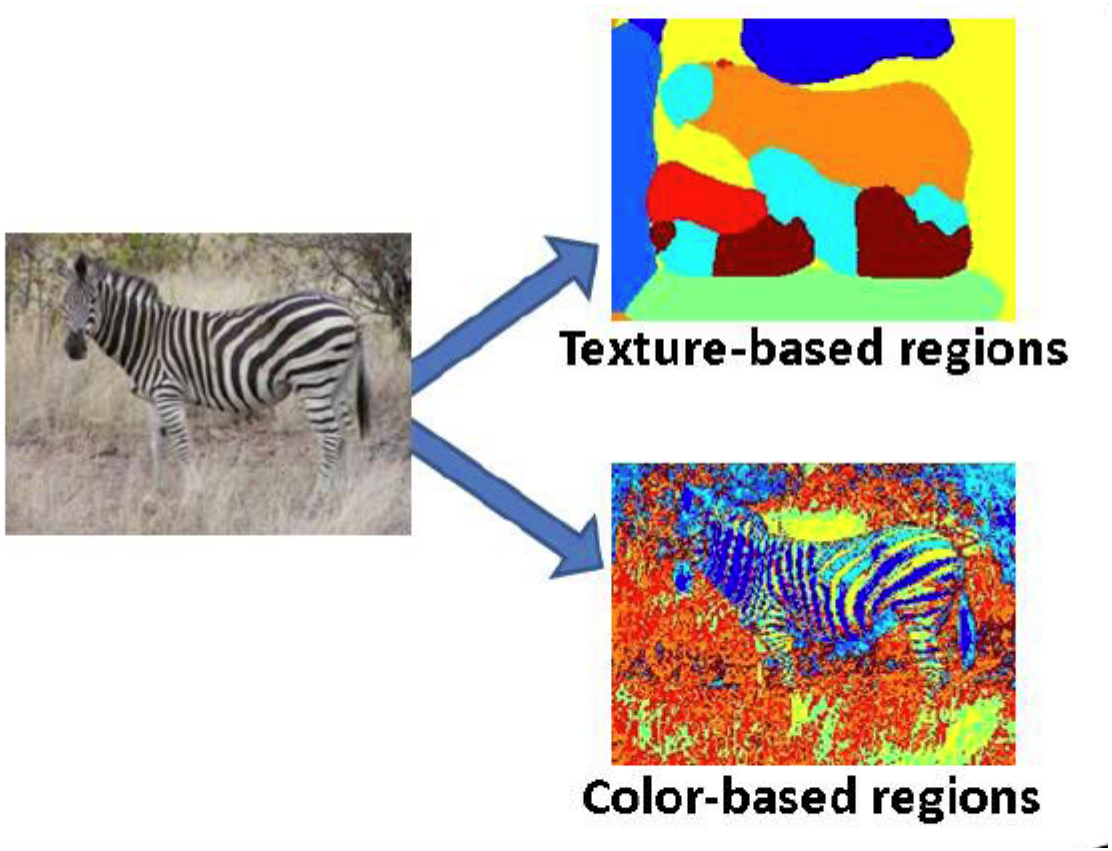




Malik, Belongie, Leung and Shi. IJCV 2001.

Adapted from Lana Lazebnik

Image segmentation example



Pixel properties vs. neighborhood properties

These look very similar in terms of their color distributions (histograms).

How would their *texture* distributions compare?

Material classification example

For an image of a single texture, we can classify it according to its global (image-wide) texton histogram.

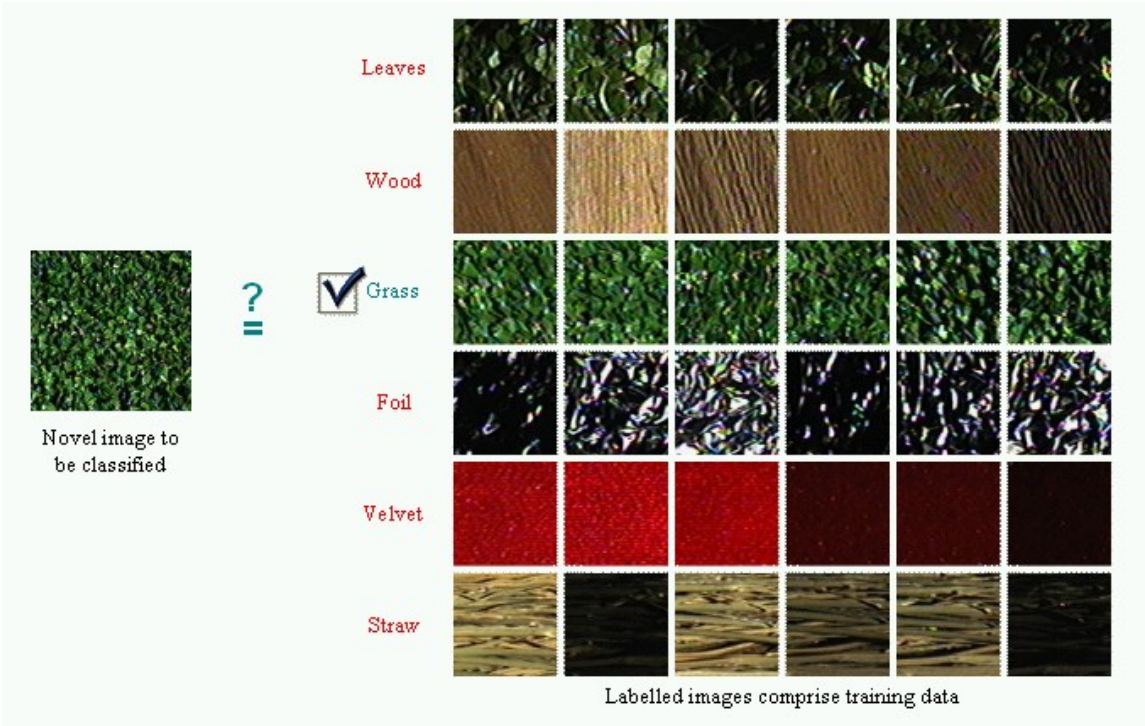


Figure from Varma & Zisserman, IJCV 2005

Material classification example

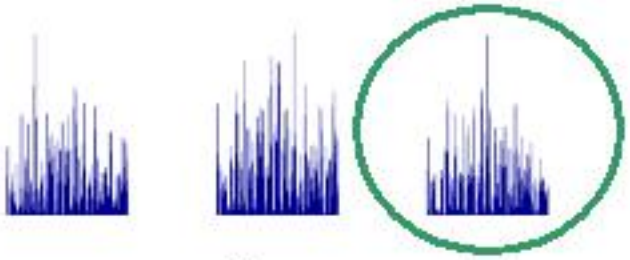
Nearest neighbor classification: label the input according to the nearest known example's label.

Plastic

Novel Image

Model

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



Grass

Manik Varma
http://www.robots.ox.ac.uk/~vgg/research/texclass/with.html

Conclusion

- Today: Segmentation
  - K-means
  - Features
  - Textons

- Tuesday
  - More on segmentation
  - Classification

Computer Vision Projects

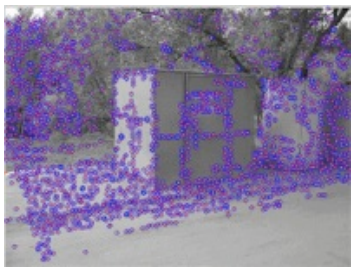
Organization

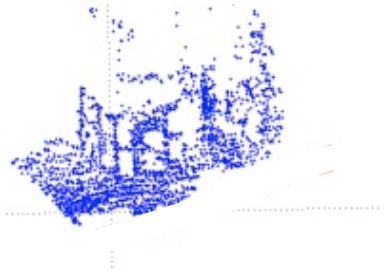
- Groups of 2 or 3 students
  - If one member is 461, project will be graded using the 461 scheme
- 6 weeks
- Schedule
  - Oct 28 – one page description
    - Project choice, team members, tentative plan
  - Nov 14 – Progress report
    - Current results, current issues, pictures of the object for project #1
  - Nov 29 / Dec 1 – Short presentation in class
    - A few slides with pictures/videos to describe approach + results
  - Dec 02 – Code and final report
    - Comment the code and follow the instructions in the document

Projects – choice between

- (1) Reconstruction of a Hopkins landmark
  - Calibrate your camera
  - Select an **interesting** object of any size
  - Take pictures from different views
  - Compute relative motion between views
  - Triangulate corresponding features
  - Show your results, e.g. make a movie containing different views of the 3D point cloud

  - **Baseline**: an imperfect but recognizable reconstruction
  - **600.461**: show results towards a dense reconstruction





Matlab viewer

Project (2) - Video

Projects – choice between

- (2) Object detection and classification in a video sequence
  - Detect moving objects (blobs),
  - Stabilize video if needed, by recovering affine transformation between frames
  - Implement suitable feature vectors to characterize objects, using appearance and motion
  - Generate training data [25 last seconds of video only for testing]
  - Classify the moving objects
  - Display your results using bounding boxes with ID and class color

  - **Baseline**: an imperfect but visually acceptable detection of moving cars and yellow taxis
  - **600.461**: show results towards detection of the other classes

Allowed material

- All literature (of course!)
- Code from previous assignments
- The following external Matlab libraries/toolboxes (if needed)
  - VLFeat
  - Piotr's toolbox
  - libSVM
  - HMM/Kalman toolbox from K. Murphy
  - Matlab camera calibration toolbox (gui for calibrating internal camera parameters only)

Computer vision projects

- Code needs to run on the CS undergrad lab computers
- Please insert comments and instructions in the code
- Feel free to use the approaches that you prefer
- More information (including grading) in the project description document

Be creative and have fun !