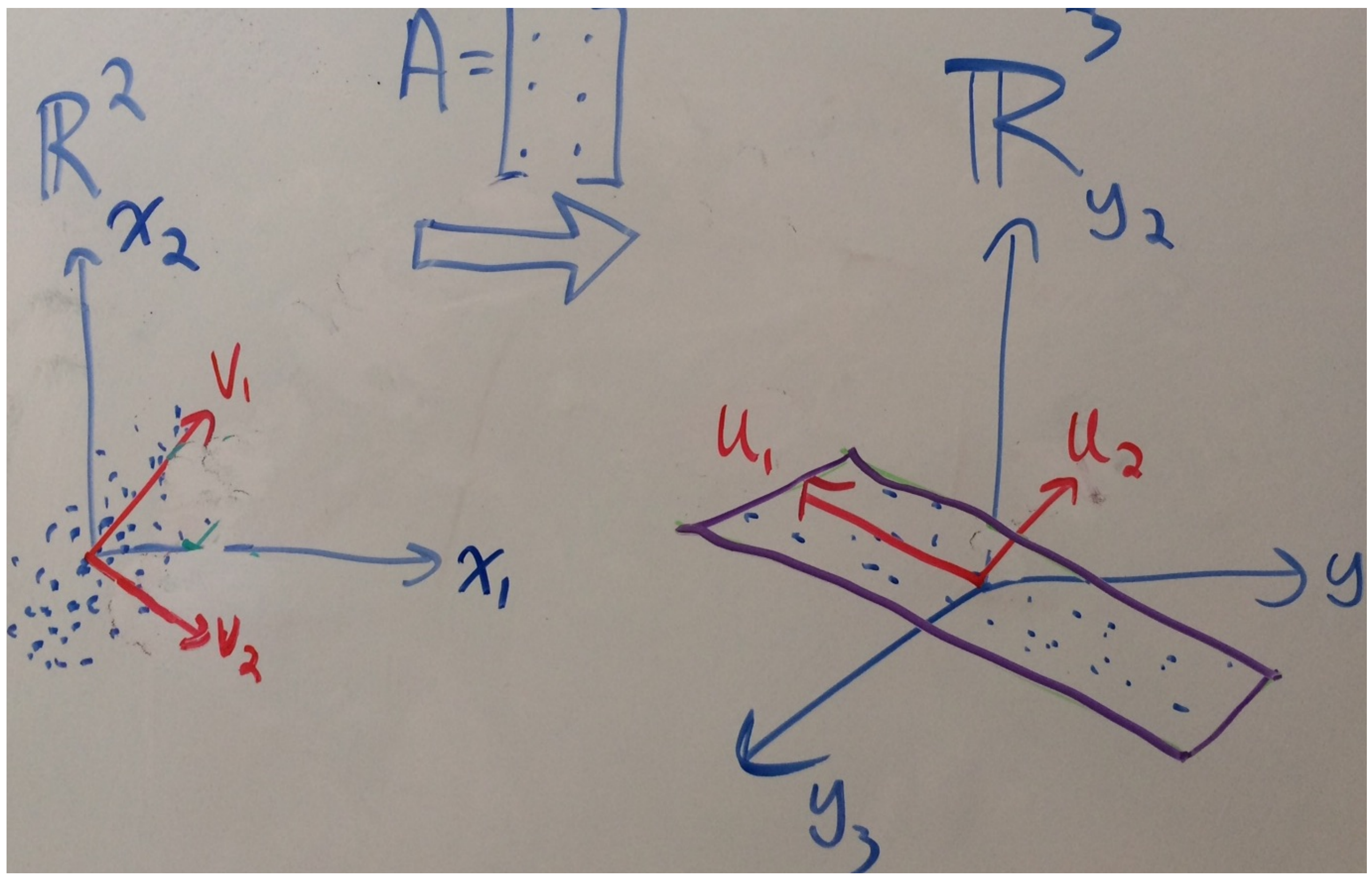
Texture

- SVD Matlab demo
- Texture
- Bag-of-words
- (Spatial) pyramid matching

Recall: SVD

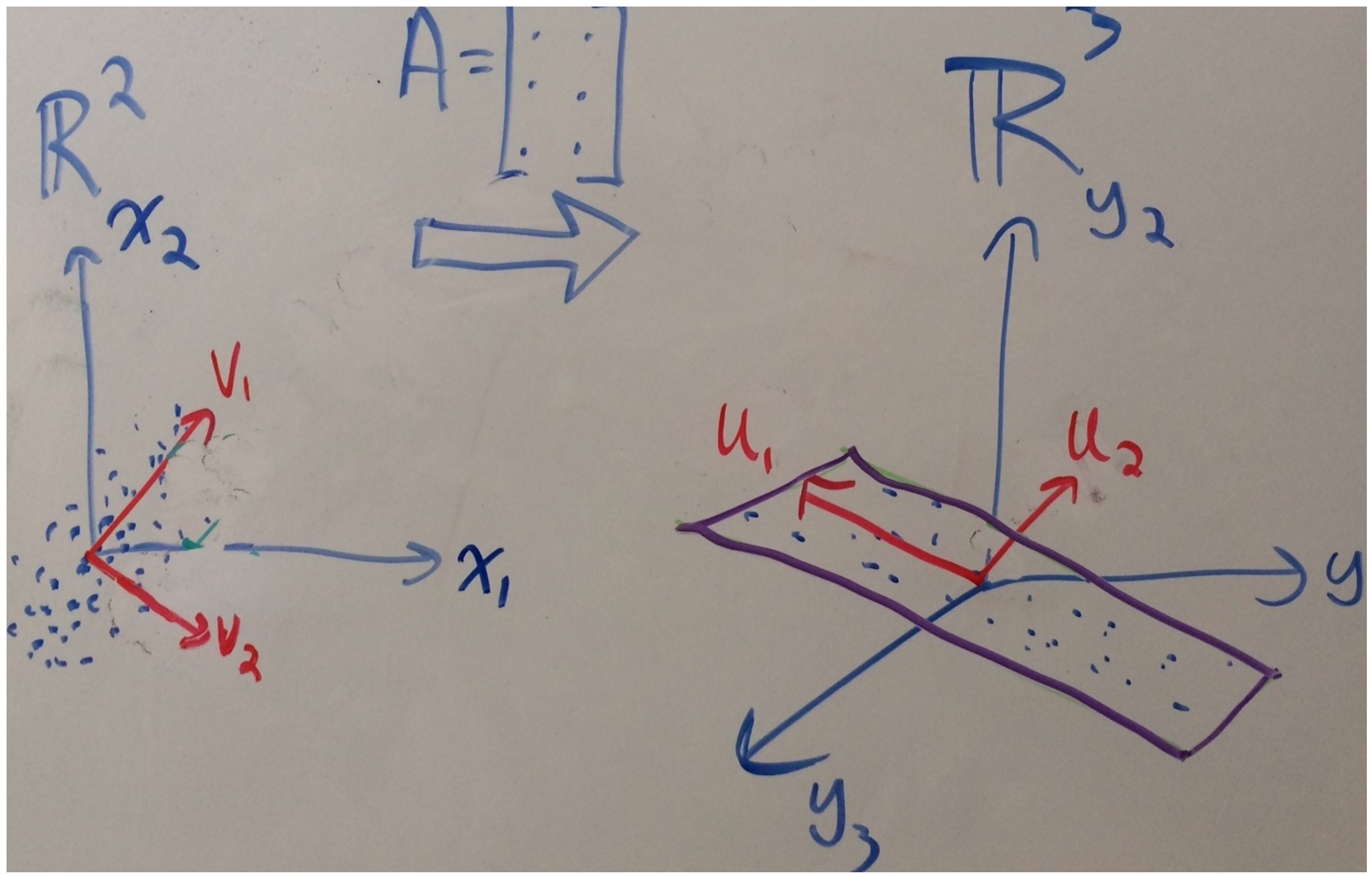


$$y = Ax$$

$$y = U\Sigma V^T x$$

Recall: SVD

$$y = U\Sigma V^T x$$



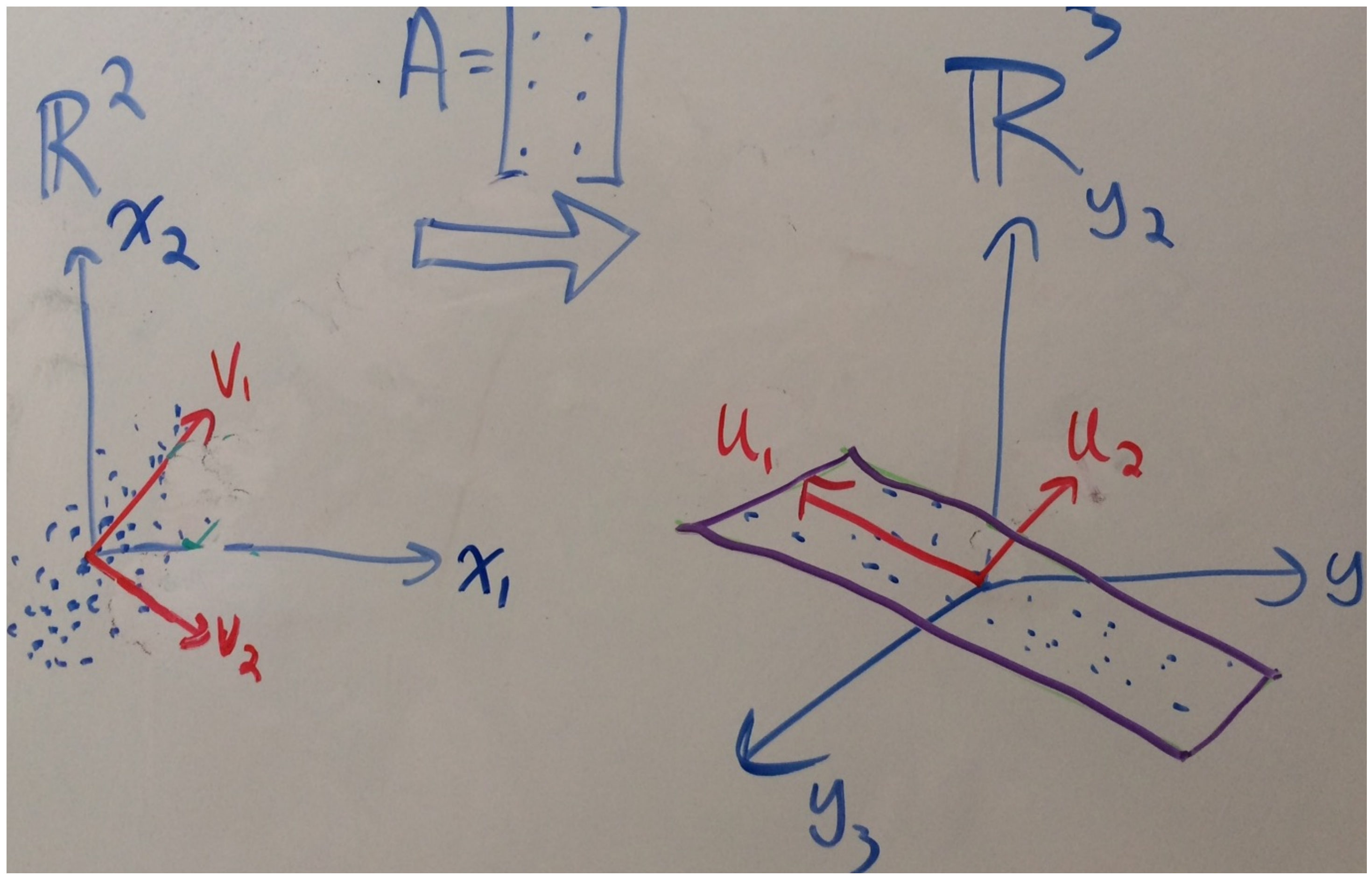
$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 \dots \\ 0 & \sigma_2 & 0 \dots \\ 0 & 0 & \sigma_3 \dots \\ \vdots & \vdots & \vdots \end{bmatrix}$$

$$U = \begin{bmatrix} u_1 & u_2 \dots u_m \end{bmatrix}, U^T U = I$$

$$V = \begin{bmatrix} v_1 & v_2 \dots v_n \end{bmatrix}, V^T V = I$$

Recall: SVD

$$y = U\Sigma V^T x$$



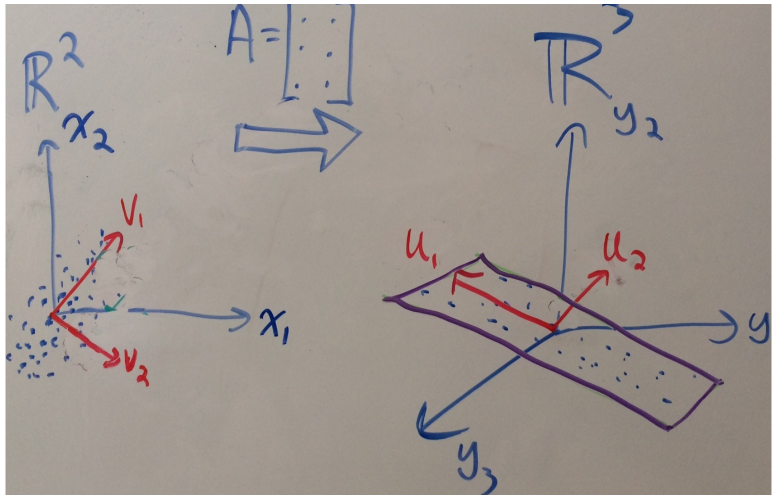
Notation: u_i = left singular vector, sigma_i = singular values, v_i = right singular vectors

Any linear operator can be thought of as mapping from R^n to R^m

1. projection (with right singular vectors)
2. scaling (with singular values),
3. reconstruction (with left singular vectors)

Recall: SVD

Immediate consequences (by appealing to geometric intuition)



$$||A||_F = A(:)^T A(:)$$

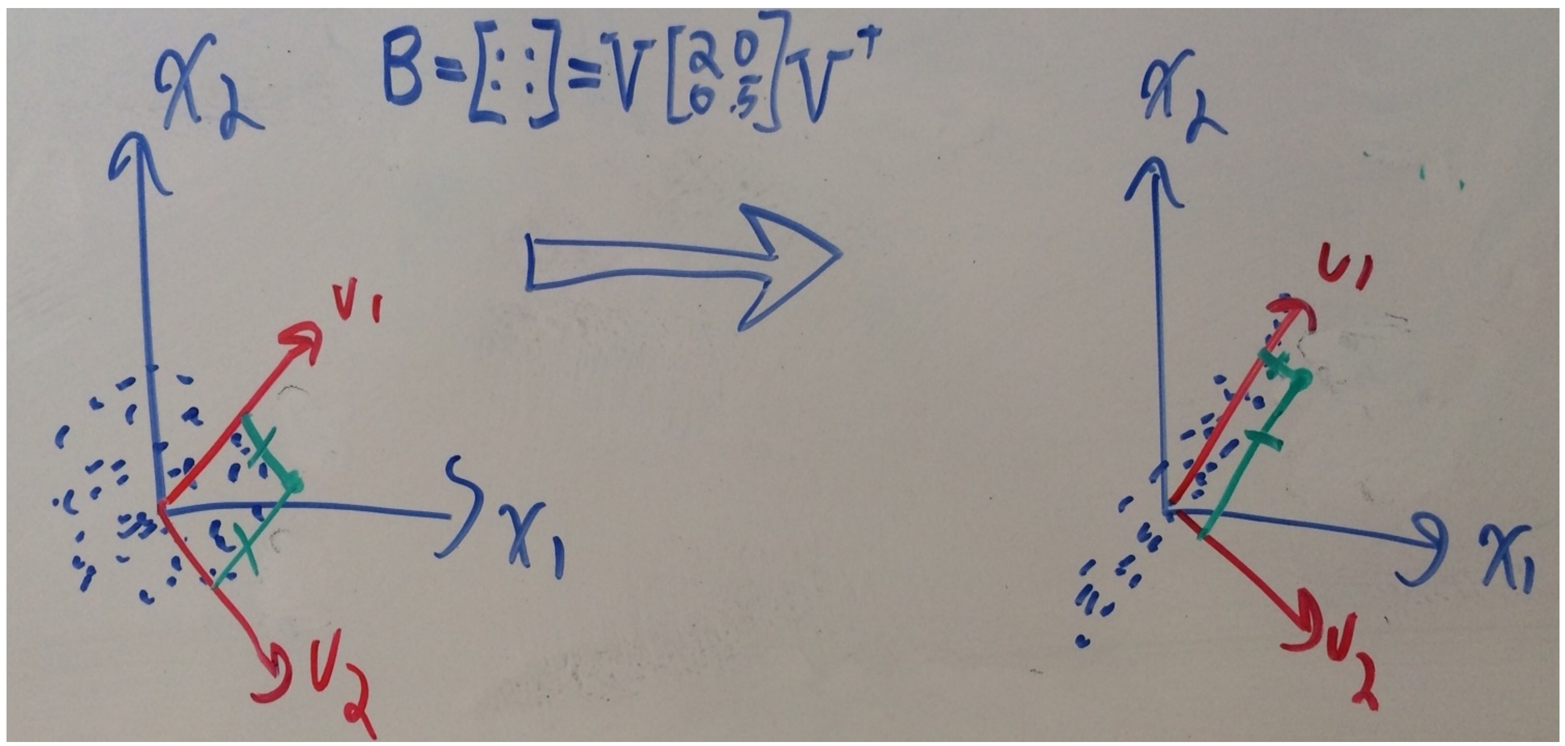
Low rank:
$$\min_{A':rank(A')<=k} ||A - A'||_F = U(:,1:k)\Sigma(1:k,1:k)V(:,1:k)^T$$

Homogenous least-squares:
$$\min_{h:h^T h=1} ||Ah||^2 = V(:,end)$$

Pseudoinverse:
$$A^+ = \operatorname*{argmin}_{A^+} ||A^+ A - I||_F = V \begin{bmatrix} \frac{1}{\sigma_1} & 0 & 0 \dots \\ 0 & \frac{1}{\sigma_2} & 0 \dots \\ 0 & 0 & \frac{1}{\sigma_3} \dots \\ \vdots & \vdots & \vdots \end{bmatrix}^T U^T$$

Extension:
positive-semidefinite (PSD) matrices

Let's construct a square matrix $B = A^T A$



Any PSD matrix can be thought of as mapping from R^n to R^n

1. projection (with eigenvectors)
2. scaling (with eigenvalues),
3. reconstruction (with *same* eigenvectors)

For V = identity, B = scaling
For general V, B = scaling in rotated coordinate system

Matlab demo

Steerable & Separable Filter Banks

(how can we efficiently exploit for convolution?)

Eigenfaces

Texture

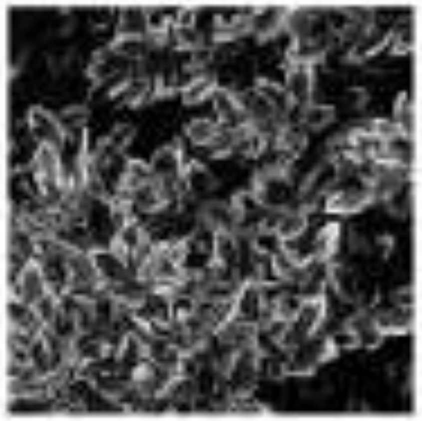
- SVD Matlab demo
- Texture
- Bag-of-words
- (Spatial) pyramid matching

How do we define texture?

Continuum of regularity

stochastic

leaves



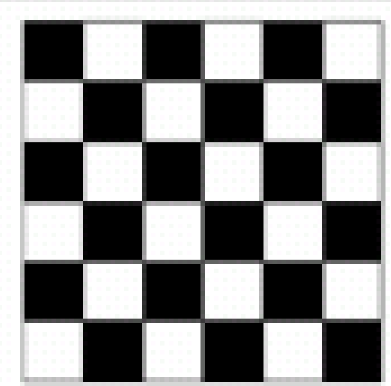
leaves

grass

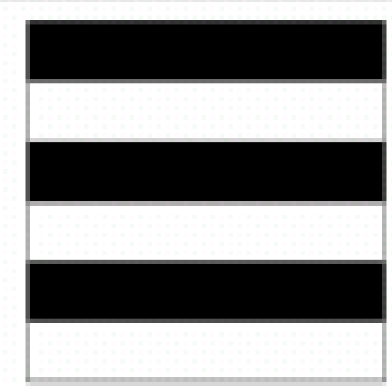
brick

brick

regular



checkerboard



striped pattern

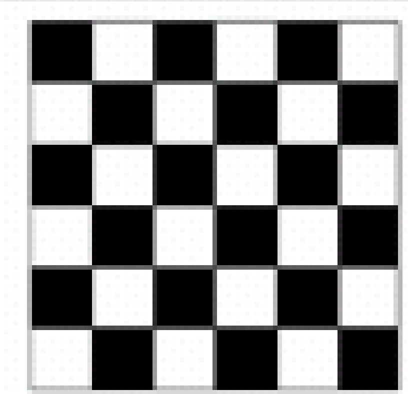
Continuum of regularity

stochastic

leaves

Each pixel is drawn iid from some probability distribution over colors P(I)

regular



checkerboard

Each pixel is color is determined completely by its coordinates w.r.t. the rest of the texture

Continuum of regularity

stochastic

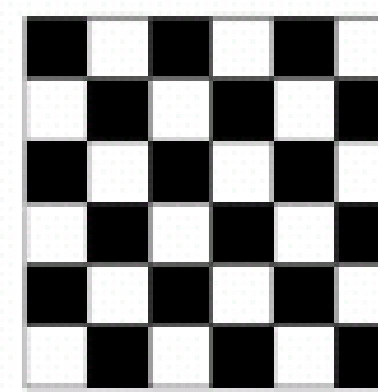
leaves

brick

brick

Each pixel is drawn from some probability distribution over colors conditioned on the value of its neighbors P(I|N)

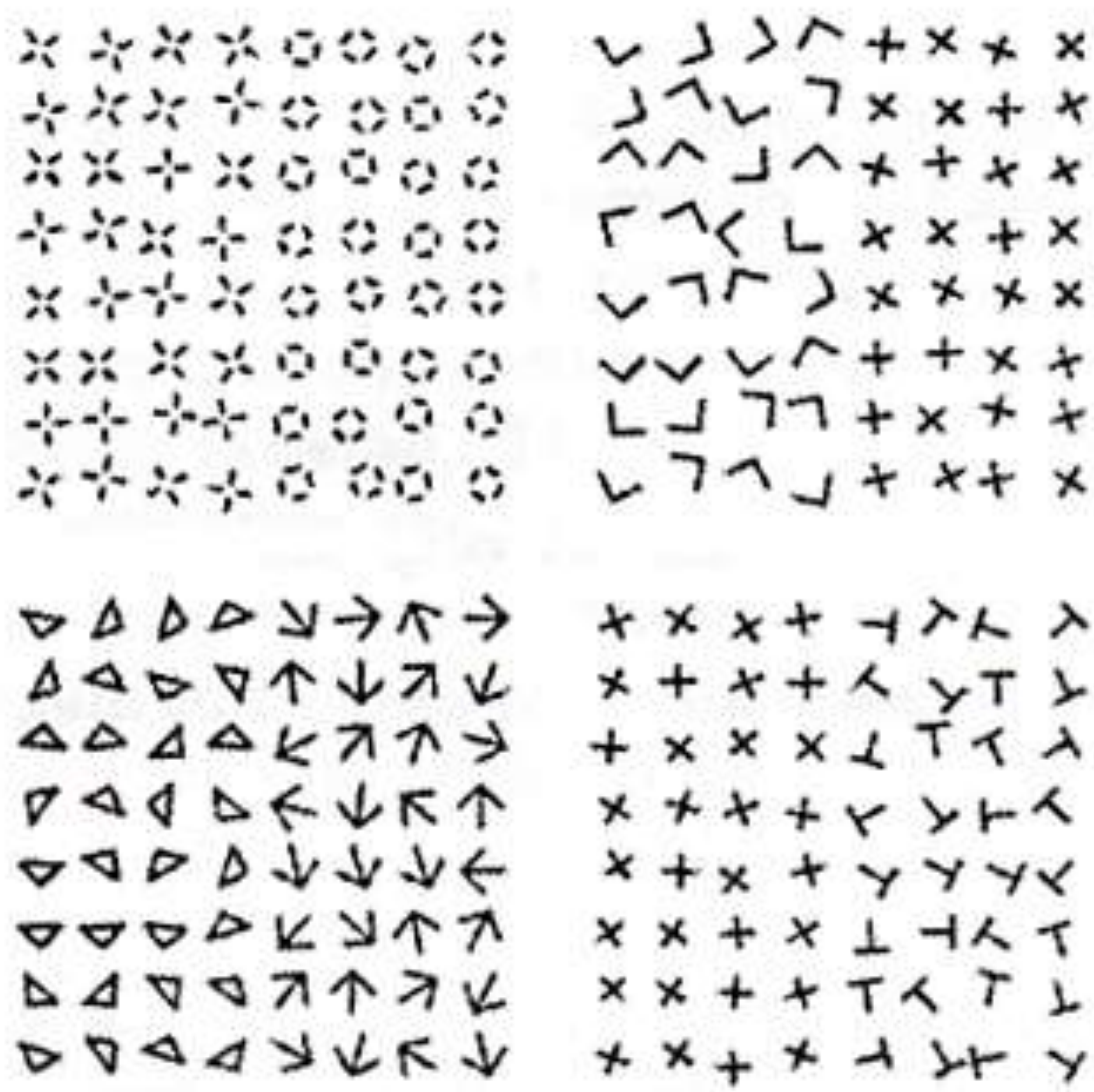
regular



checkerboard

Pre-attentive texture discrimination

(Julesz,1981)



“textons”

160 ms, outside foveal gaze
Instantenous, or effortless texture discrimination

Representing textures

Let’s encode the texture as a distribution over localized visual elements, or “textons”

1. How do we represent a texton?

2. How do we represent a distribution over them?

Version 0

Let's build a discrete probability distribution (or histogram) over small KxK patches, where each pixel can take on one of N = 256 values.

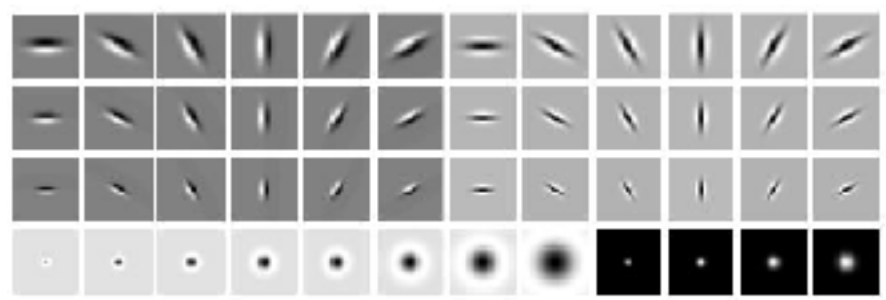
What would be the size of this histogram?

Hopeless! N^{KK}

Version 0

Let's build a discrete probability distribution (or histogram) over small KxK patches, where each pixel can take on one of N = 256 values.

Project KxK patch into M filters



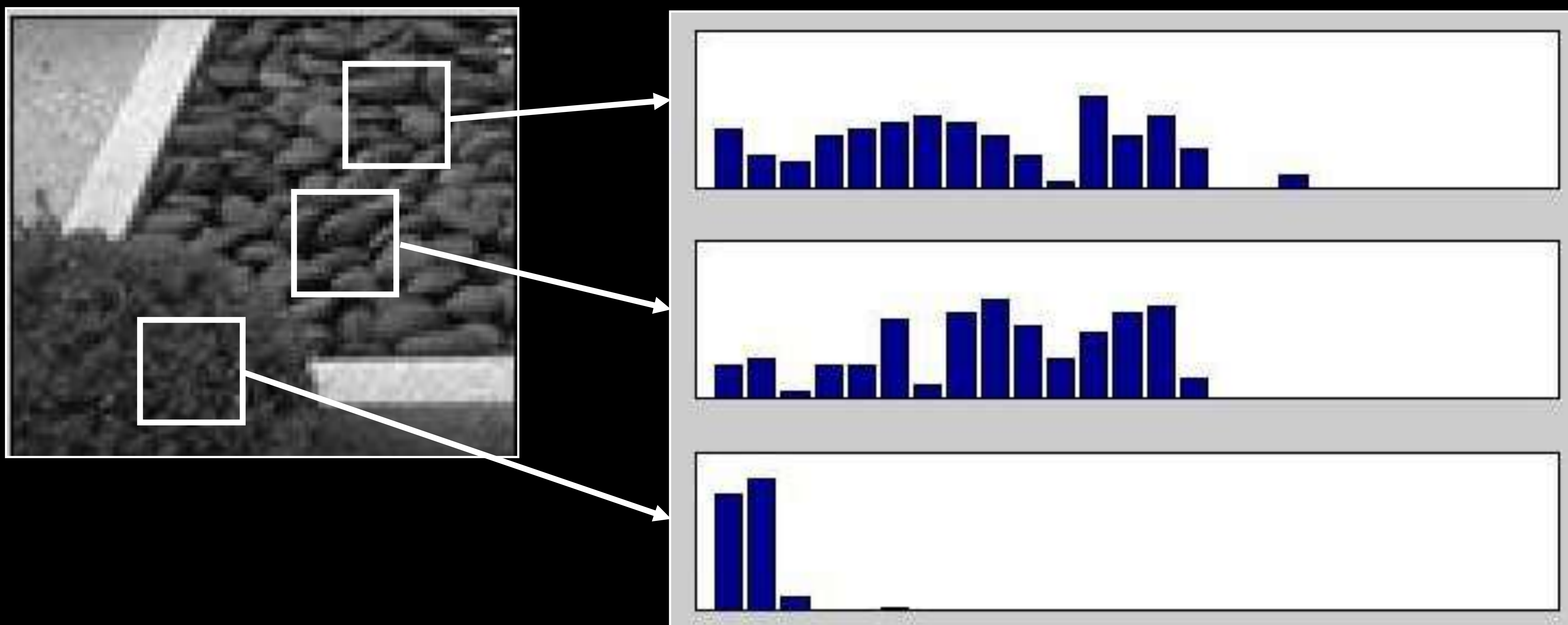
Let's represent marginal distribution instead of joint

P(f1,f2,…) = N^M table entries

P(f1)P(f2)P(f3) = N*M table entries

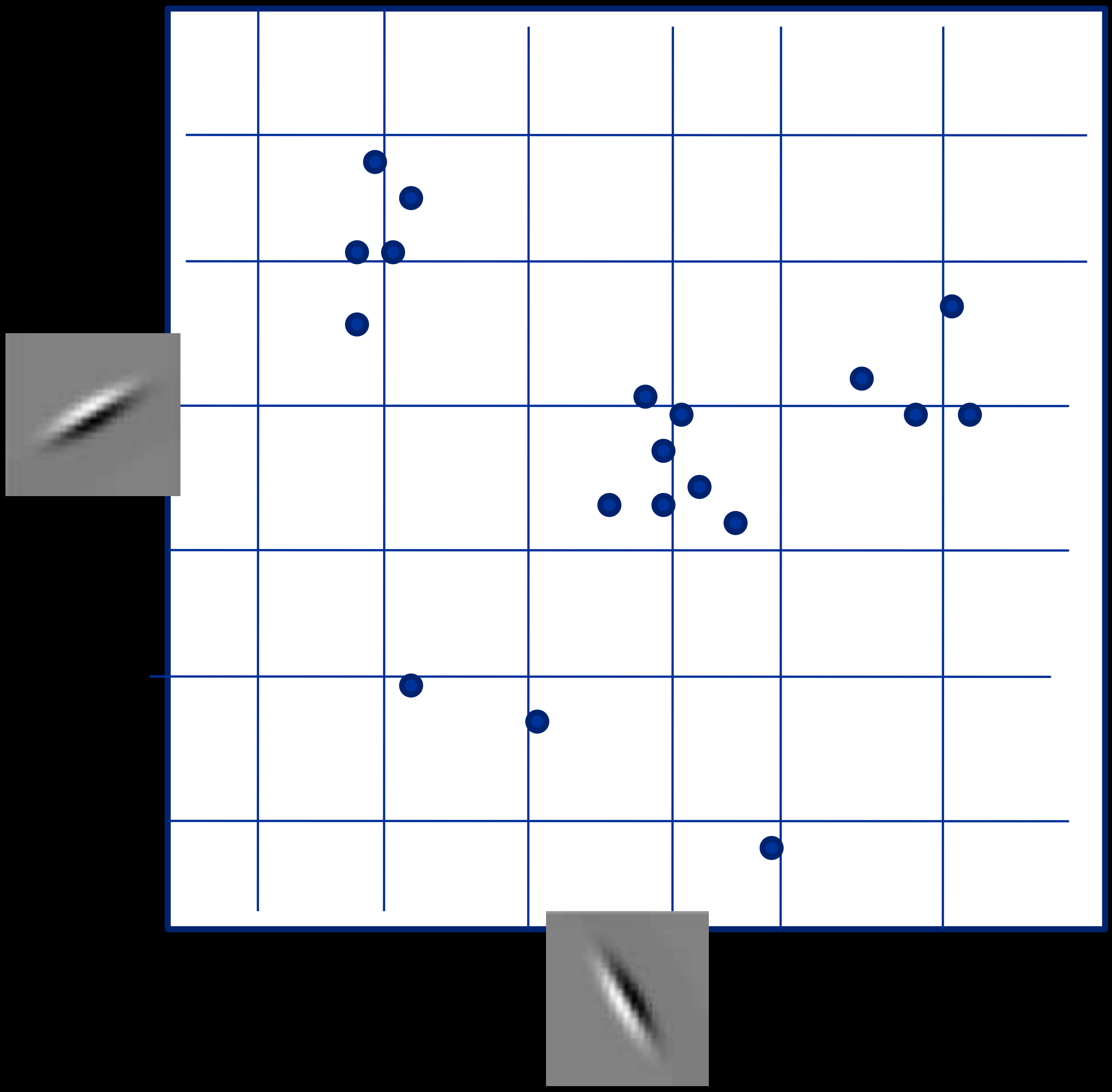
Histogram of filter responses

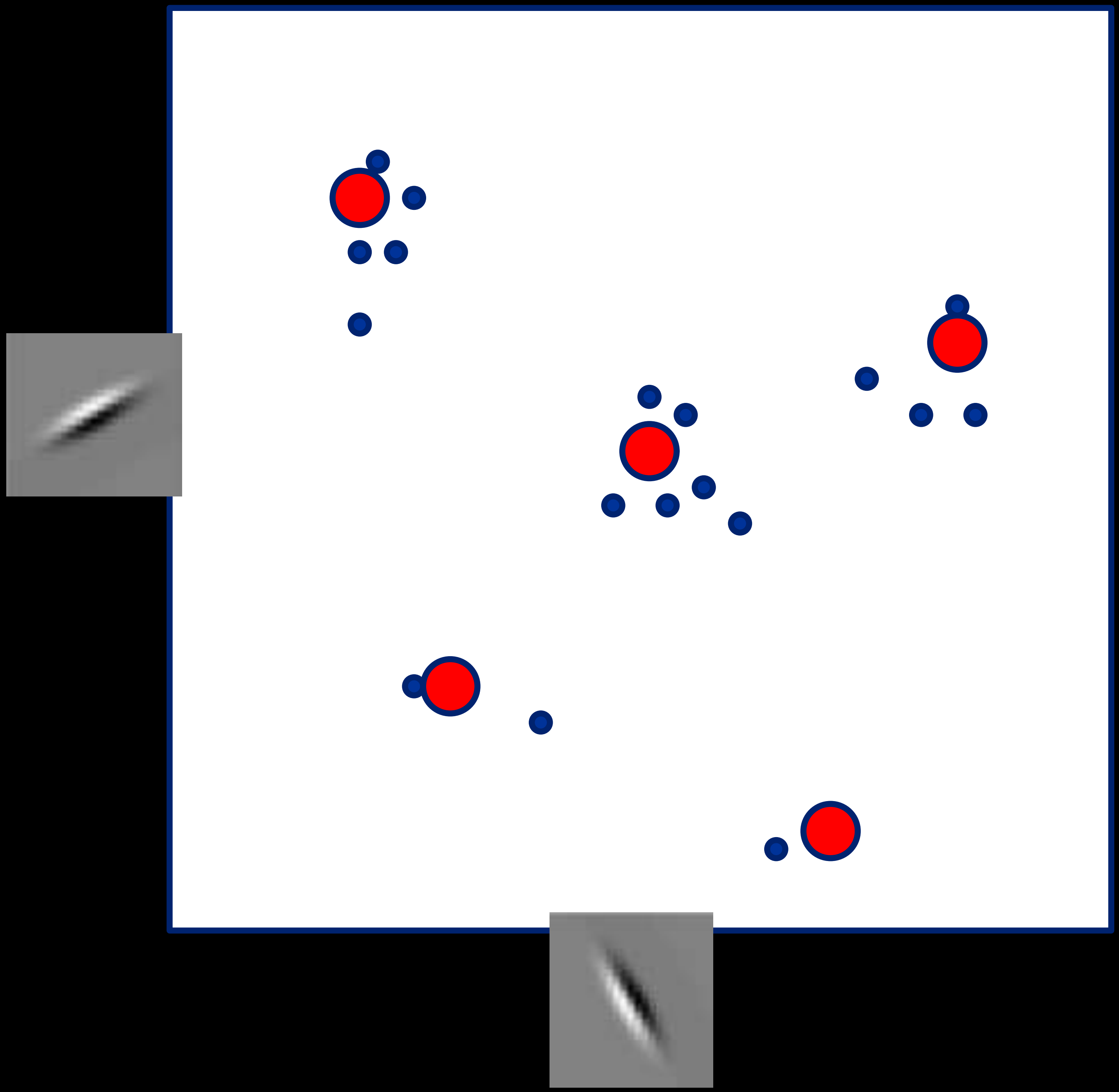
P(I * )

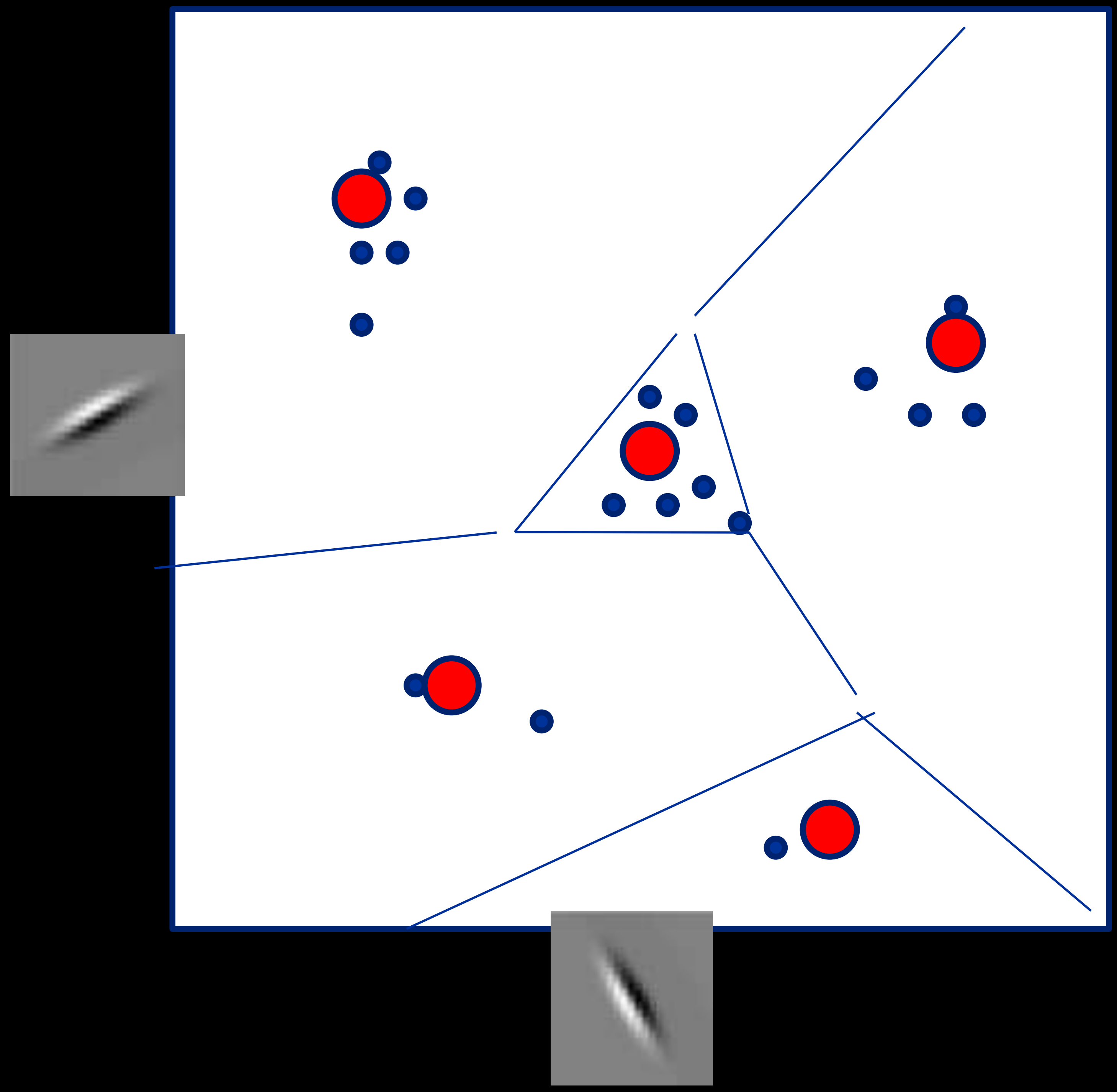


Use collection of histograms for a set of filters to represent texture

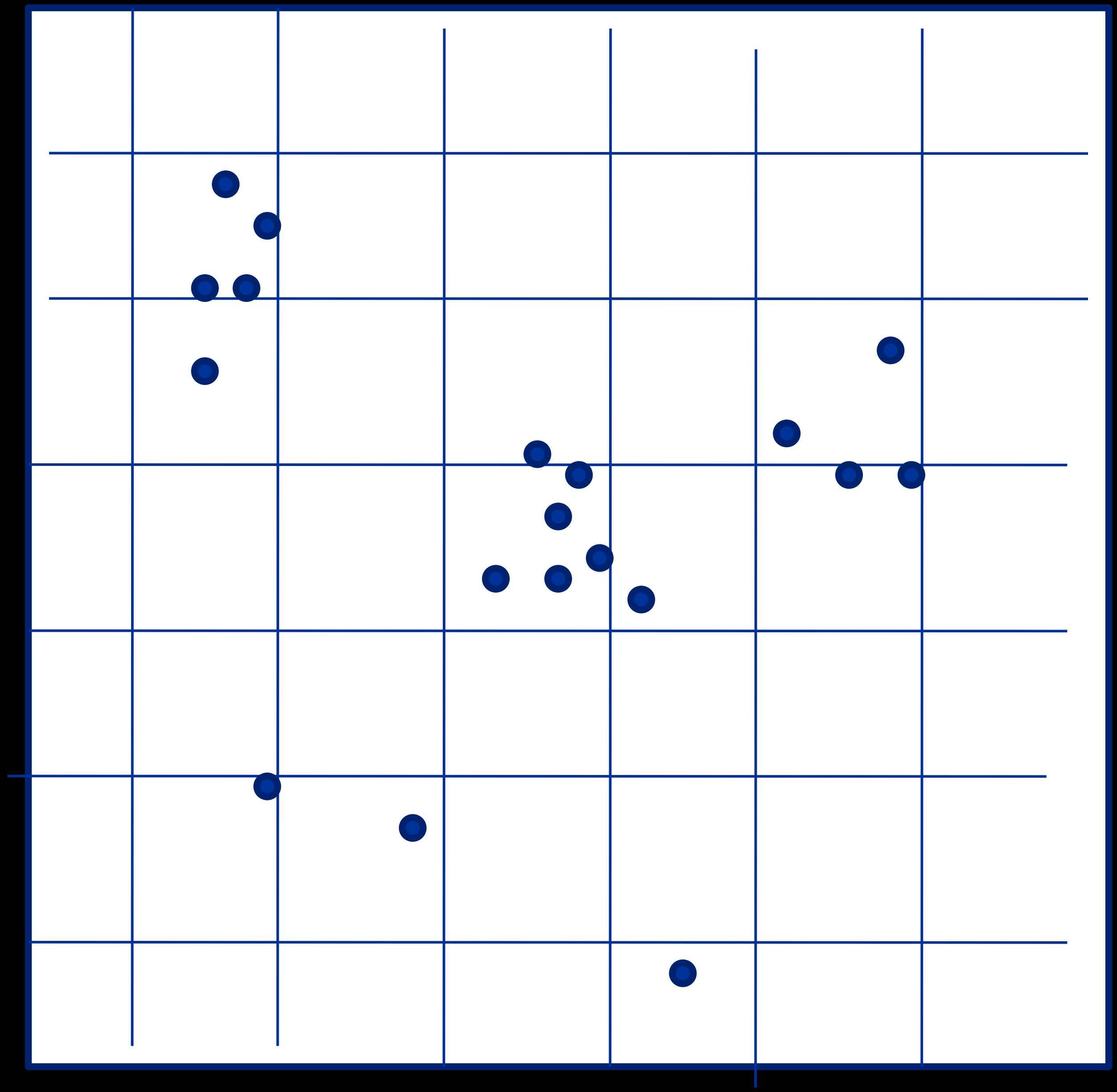
Is there an efficient way to capture joint statistics of filter responses?

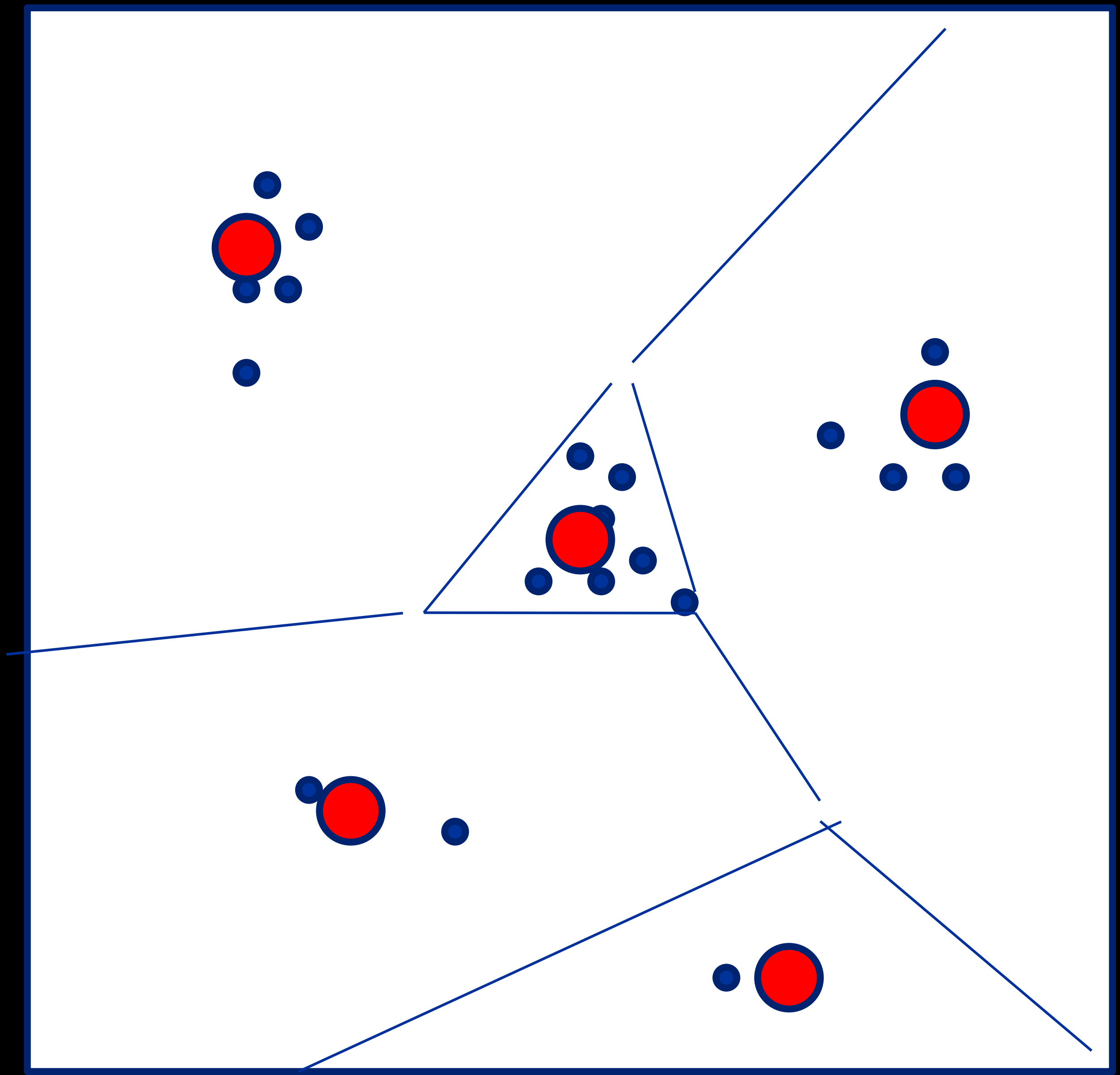






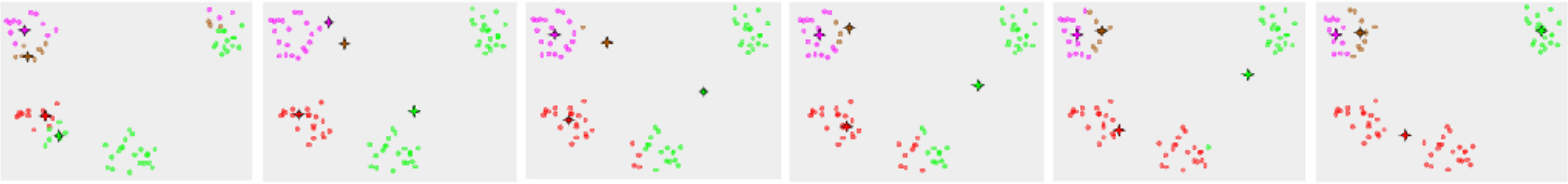
Capture joint statistics via histograms of vector quantized features





K-means

visual intuition



Cost function

$$\min_{Z,D} C(Z,D,X) \quad \text{where} \quad C(Z,D,X) = \sum_i ||x_i - d_{z_i}||^2$$

x_i: ith input vector (to be clustered)

$z_i \in \{1 \ldots K\}$: ith label

d_k: kth dictionairy element (or mean)

Coordinate descent optimization

$$\min_{Z,D} C(Z, D, X) \quad \text{where} \quad C(Z, D, X) = \sum_i ||x_i - d_{z_i}||^2$$

1. $\min_{Z} C(Z, D, X)$

2. $\min_{D} C(Z, D, X)$

Training vs testing

Training: $\min_{Z,D} C(Z, D, X_{\text{train}})$

Testing: $\min_{Z} C(Z, D, X_{\text{test}})$

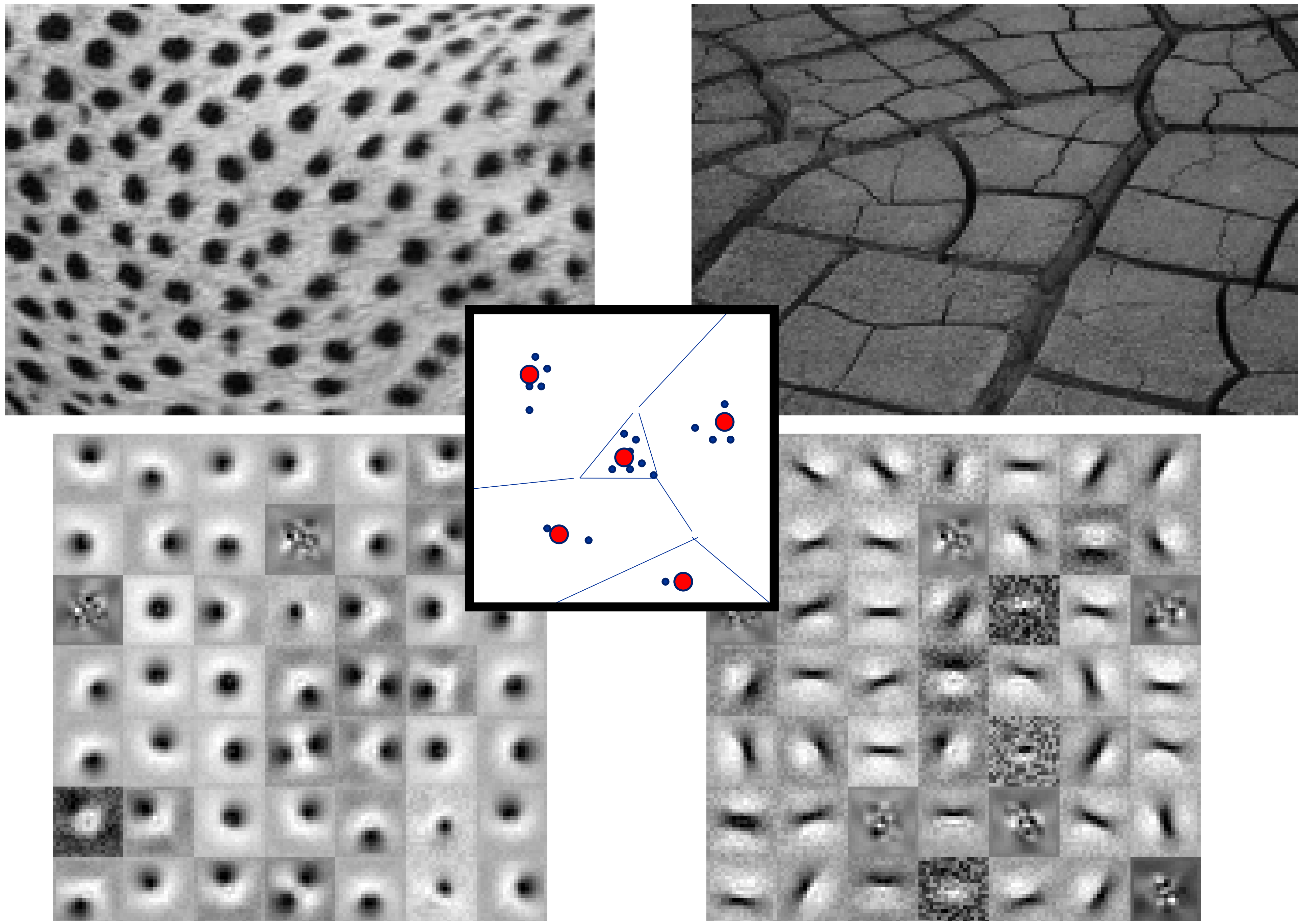
Question: how can we visualize mean (or texton)

Let B be a matrix of vectorized filters: $B = [b_1, \dots b_t]$

Given an vectorized image patch p_i,
compute filter responses with a linear projection: $x_i = B^T p_i$

Given a mean d_i,
compute corresponding image with: $\mathrm{pseudoinv}(B^T) d_i$

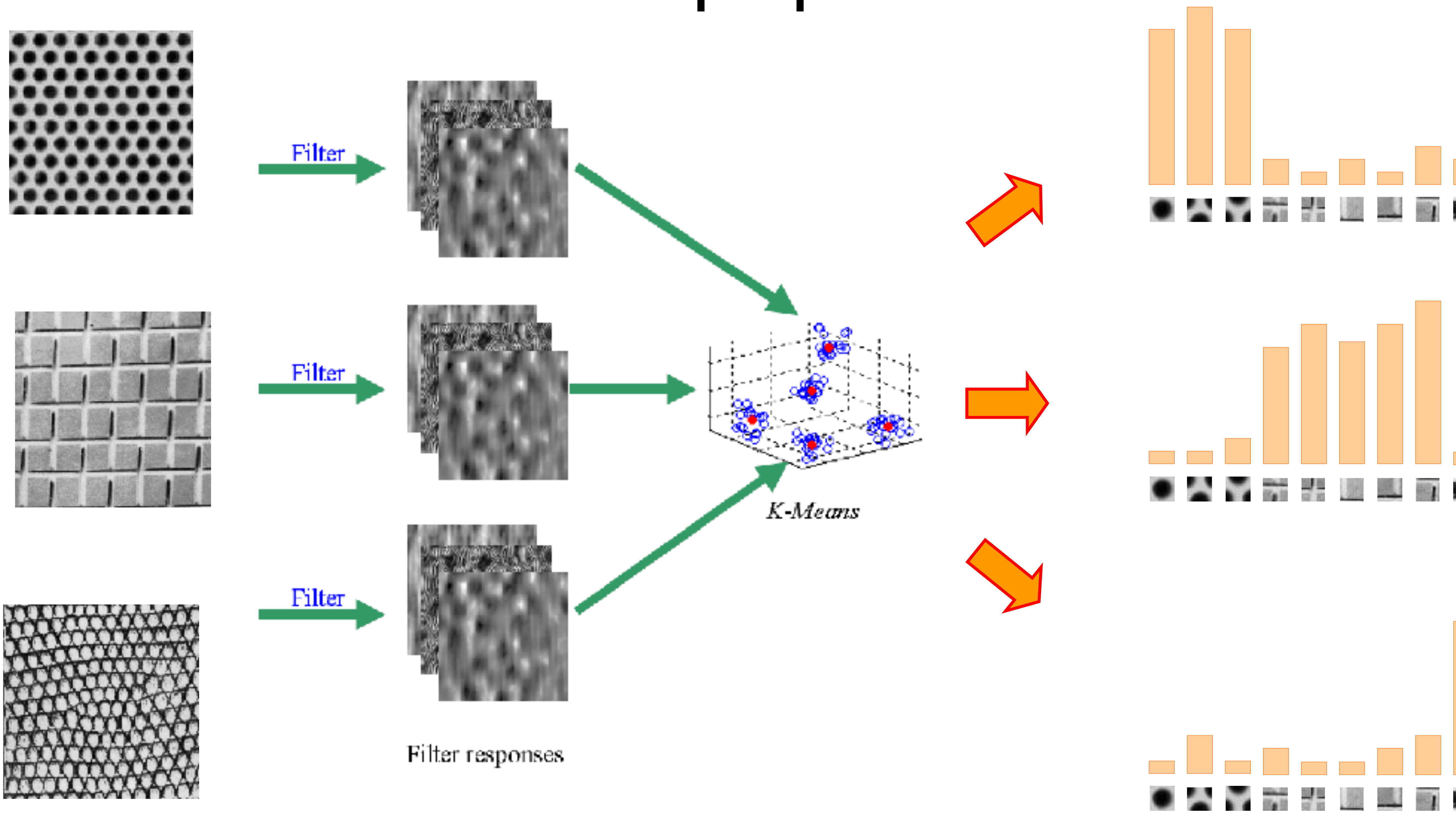
Texton reconstructions



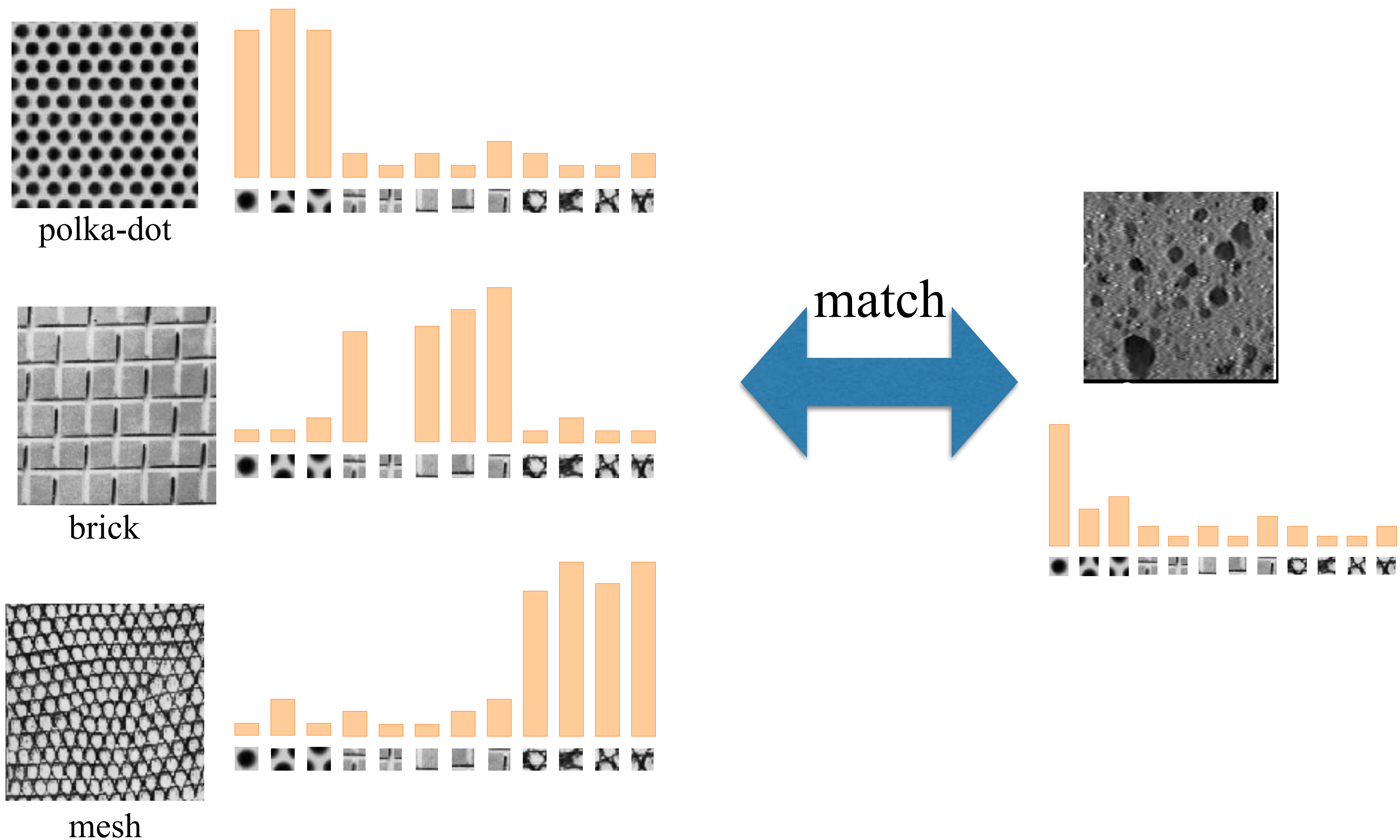
What Are Textons?

Song-Chun Zhu[1], Cheng-en Guo[1], Yingnian Wu[2], and Yizhou Wang[1]

Overall pipeline



Texture recognition



Extenstion: matrix formulation

$$\min_{D,Z} ||X - DZ||_F^2$$

$$X = [x_1, \dots x_n]$$

$$D = [d_1, \dots, d_K]$$

$$Z = [z_1, \dots z_n]$$

K-means: $z_i = [\dots, 0, 1, 0, \dots]$

Sparse reconstructions

$$\min_{D,Z} ||X - DZ||_F^2$$ subject to sparse constraints on Z

$$X = [x_1, \ldots x_n]$$
$$D = [d_1, \ldots, d_K]$$
$$Z = [z_1, \ldots z_n]$$

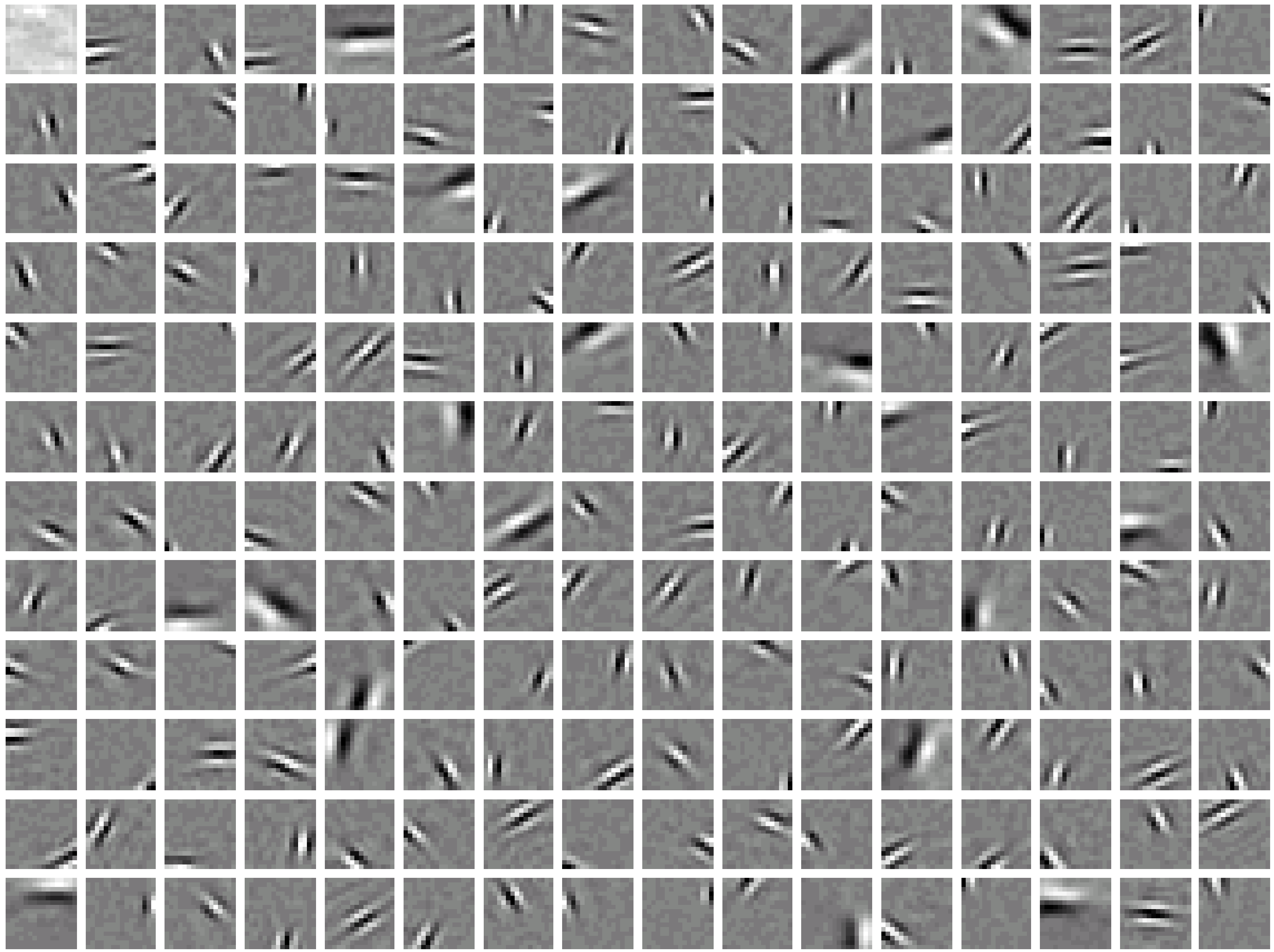
K-means: $z_i = [\ldots, 0, 1, 0, \ldots]$

L0 sparse-coding: $||z_i||_0 \leq M$ (greedy algorithms known as “matching pursuit”)

L1 sparse-coding: $||z_i||_1 \leq M$ (convex program)

L1 sparse dictionary learning

a.



- **Emergence of Simple-Cell Receptive Field Properties by Learning a Sparse Code for Natural Images.**
  Olshausen BA, Field DJ (1996). *Nature*, 381: 607-609

Sparse reconstructions

$$\min_{D,Z} ||X - DZ||_F^2$$ subject to sparse constraints on Z

$$X = [x_1, \dots x_n]$$
$$D = [d_1, \dots, d_K]$$
$$Z = [z_1, \dots z_n]$$

K-means: $z_i = [\dots, 0, 1, 0, \dots]$

L0 sparse-coding: $||z_i||_0 \leq M$

L1 sparse-coding: $||z_i||_1 \leq M$

Can be written equivalently as: $\min_{D,Z} ||X - DZ||_F^2 + R(Z)$

Sparse reconstructions

$$\min_{D,Z} ||B^T I - DZ||_F^2$$ subject to sparse constraints on Z

$$I = [i_1, \ldots x_n]$$
$$D = [d_1, \ldots, d_K]$$
$$Z = [z_1, \ldots z_n]$$
$$B = [b_1, \ldots b_t]$$

i_n is the n^{th} image patch
b_i is the t^{th} filter in the filter bank

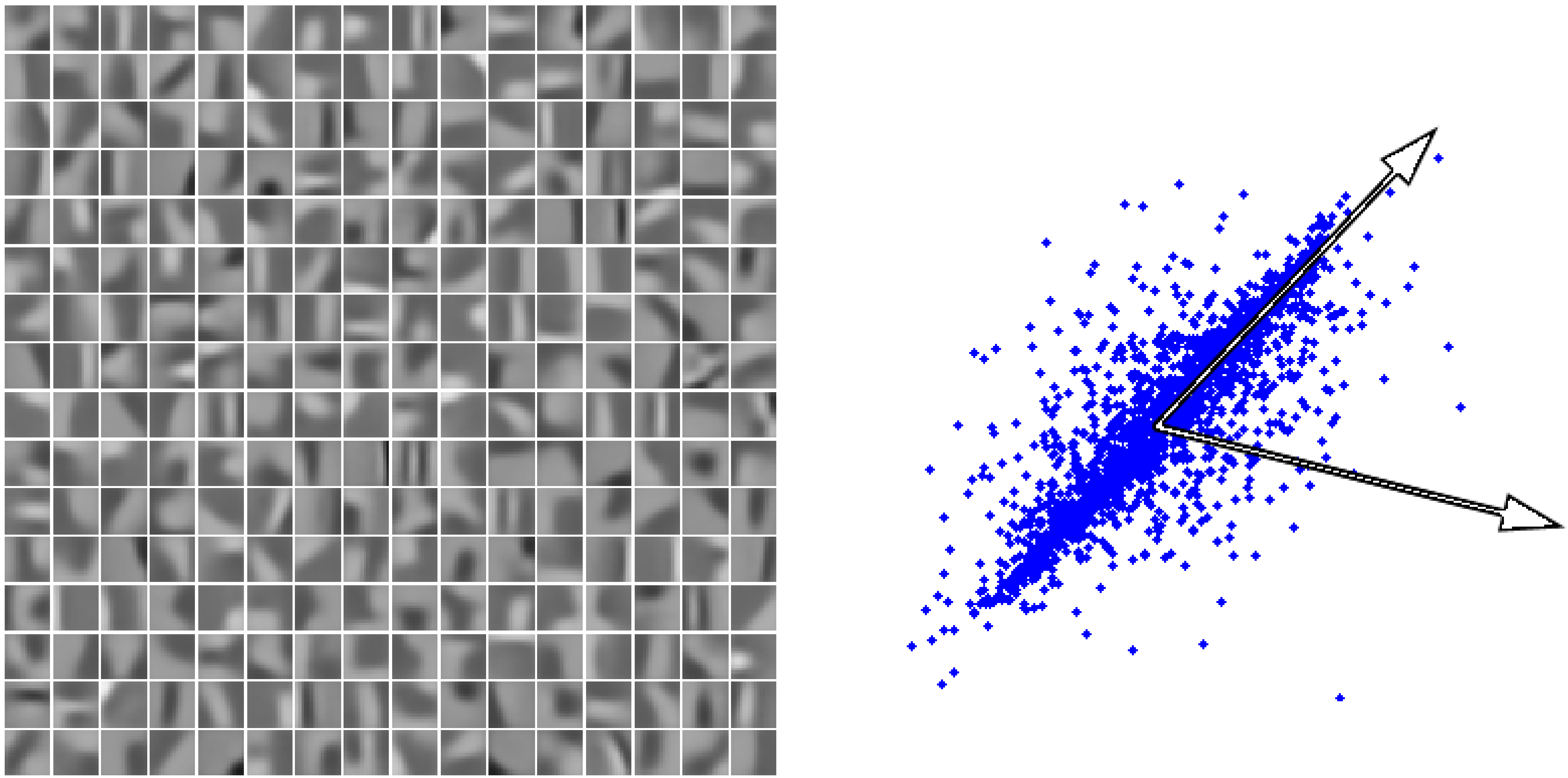
Learning Feature Representations with K-means

Adam Coates and Andrew Y. Ng

Stanford University, Stanford CA 94306, USA
{acoates,ang}@cs.stanford.edu

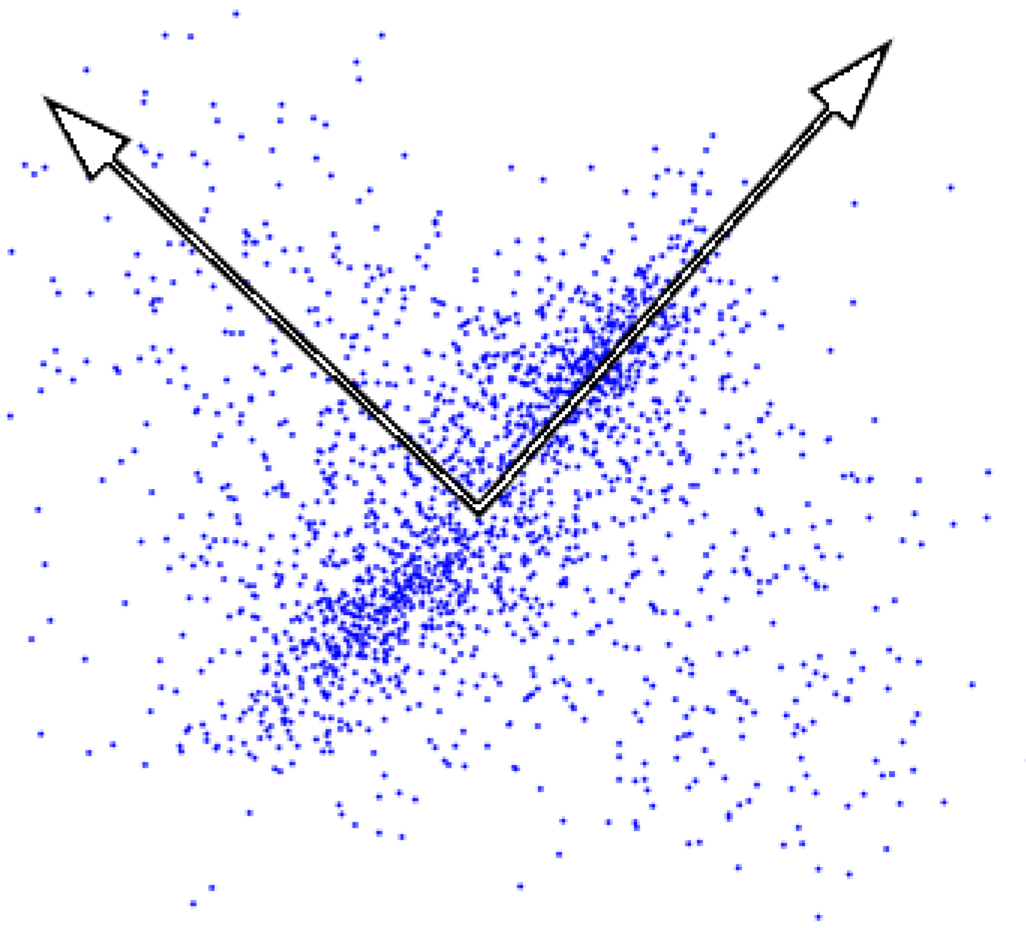
Originally published in: G. Montavon, G. B. Orr, K.-R. Müller (Eds.), Neural Networks: Tricks of the Trade, 2nd edn, Springer LNCS 7700, 2012.

K-means dictionary learning

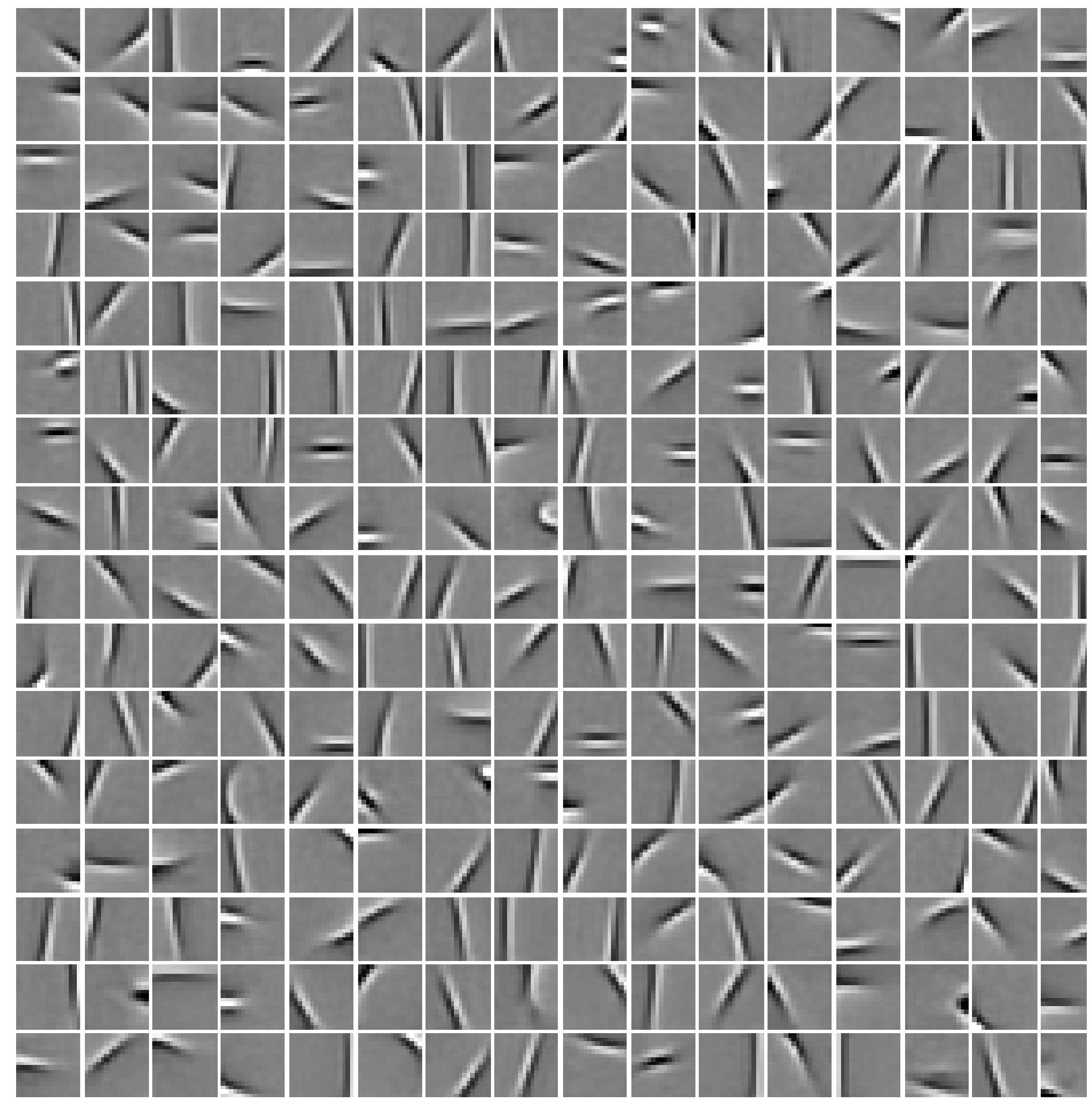


(a)

K-means dictionary learning with whitened images



(b)



(c)

Whitening: choose linear transformation B such that $E[(B^TX)(X^TB)] = \text{Identity}$

Dictionary learning

$$\min_{D,Z} ||X - DZ||_F^2$$

Some folks claim that k-means should always be replaced by sparse coding (never hurts, sometimes better)

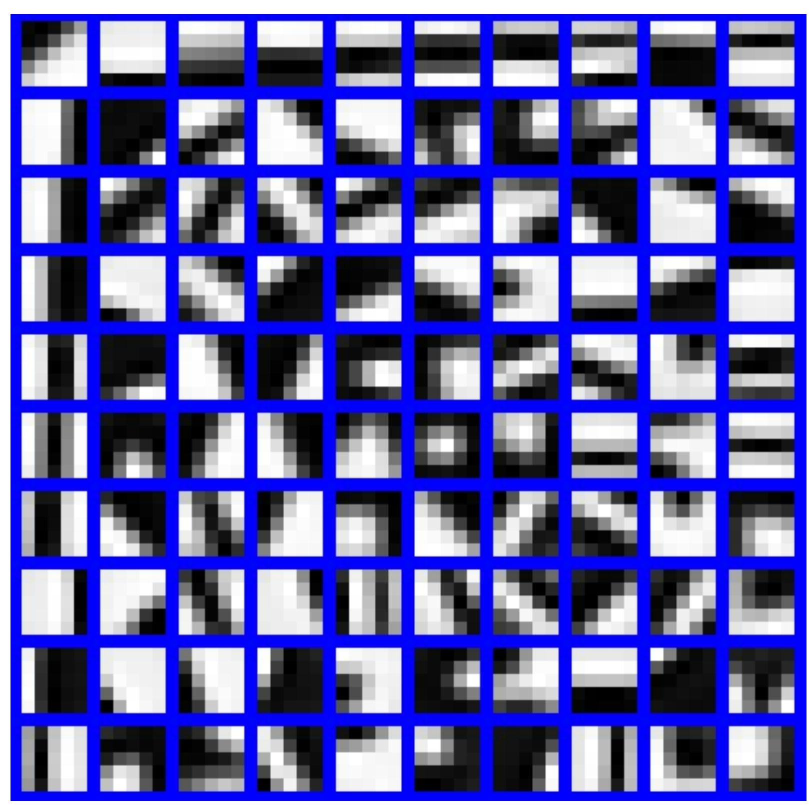
… k-means is far simpler, right?

“In between” k-means and sparse coding

$$\min_{D,Z} ||X - DZ||_F^2$$ subject to sparse constraints on Z

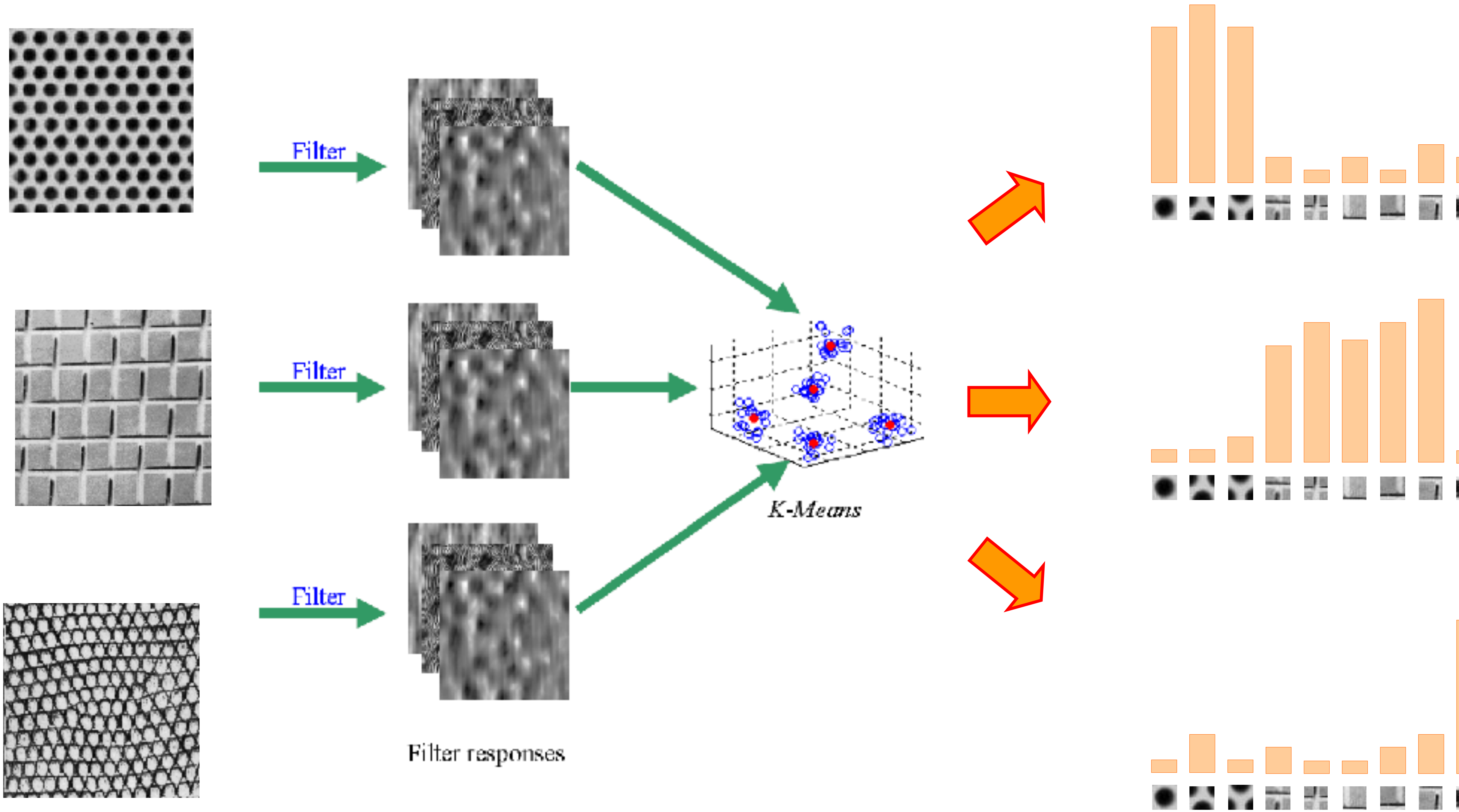
K-means: $z_i = [\ldots, 0, 1, 0, \ldots]$

L0 sparse-coding: $||z_i||_0 \leq M$ (For M = 1, we can solve for Z in closed form)

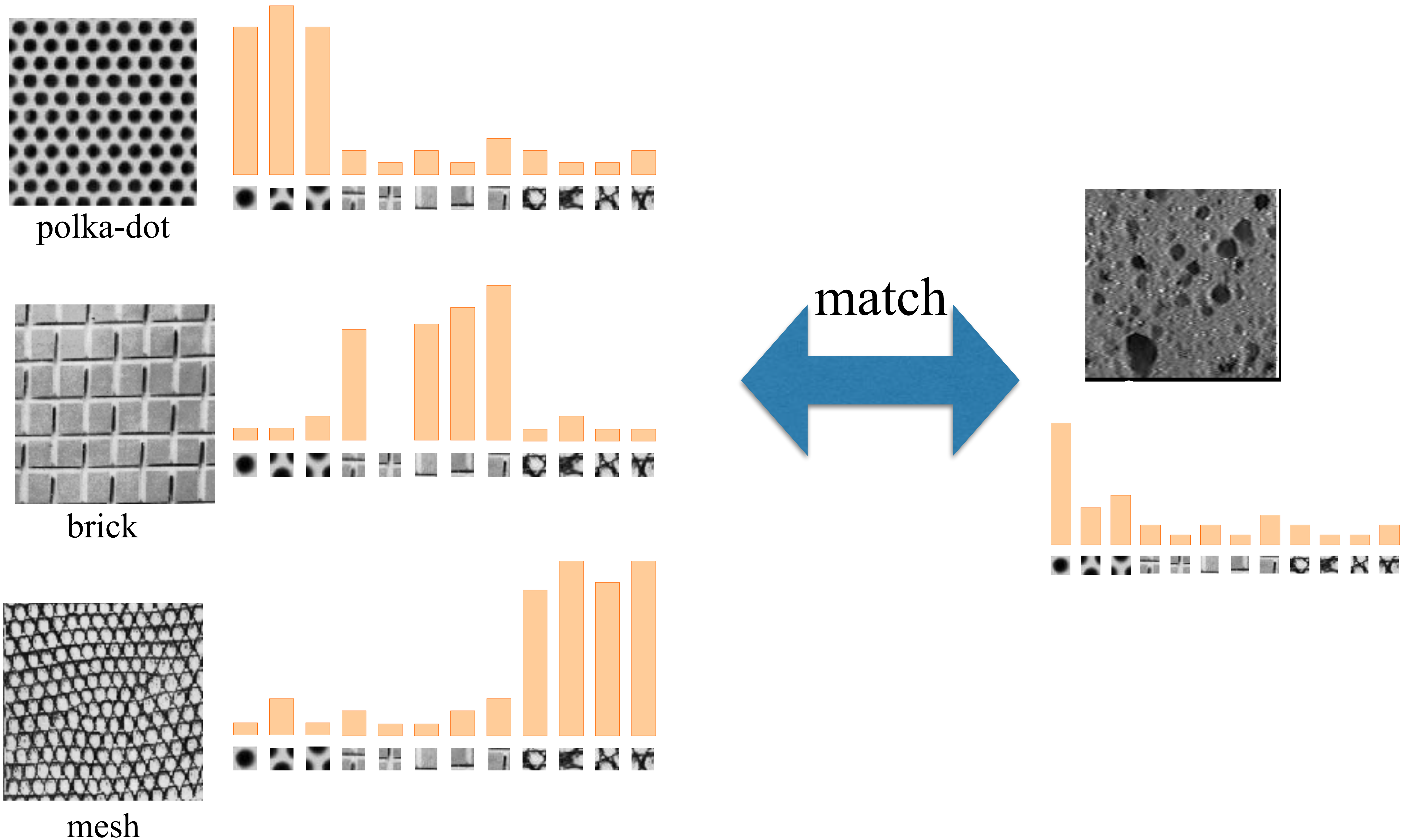


Generalizes k-means with cosine distance to allow for negative coefficients

Histograms of Sparse Codes for Object Recognition, CVPR13



Filter
Filter
Filter
K-Means
Filter responses


polka-dot
brick
mesh
match

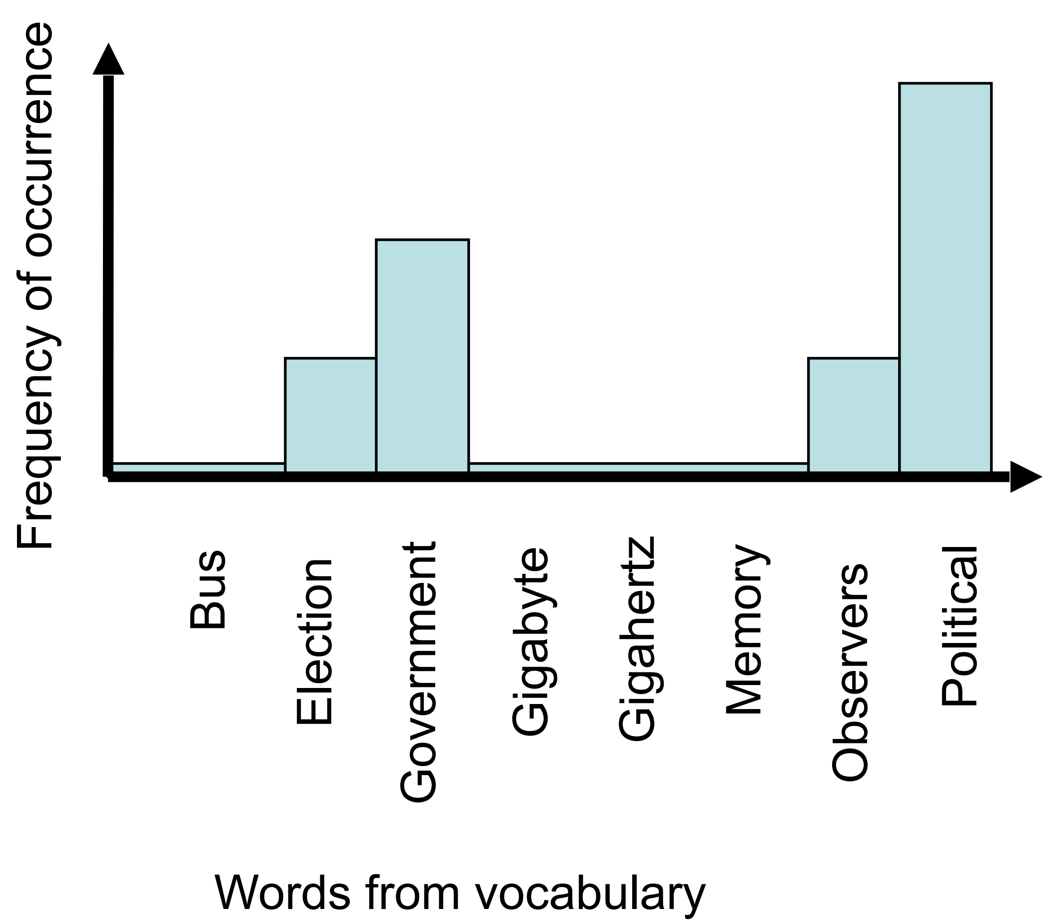
Texture

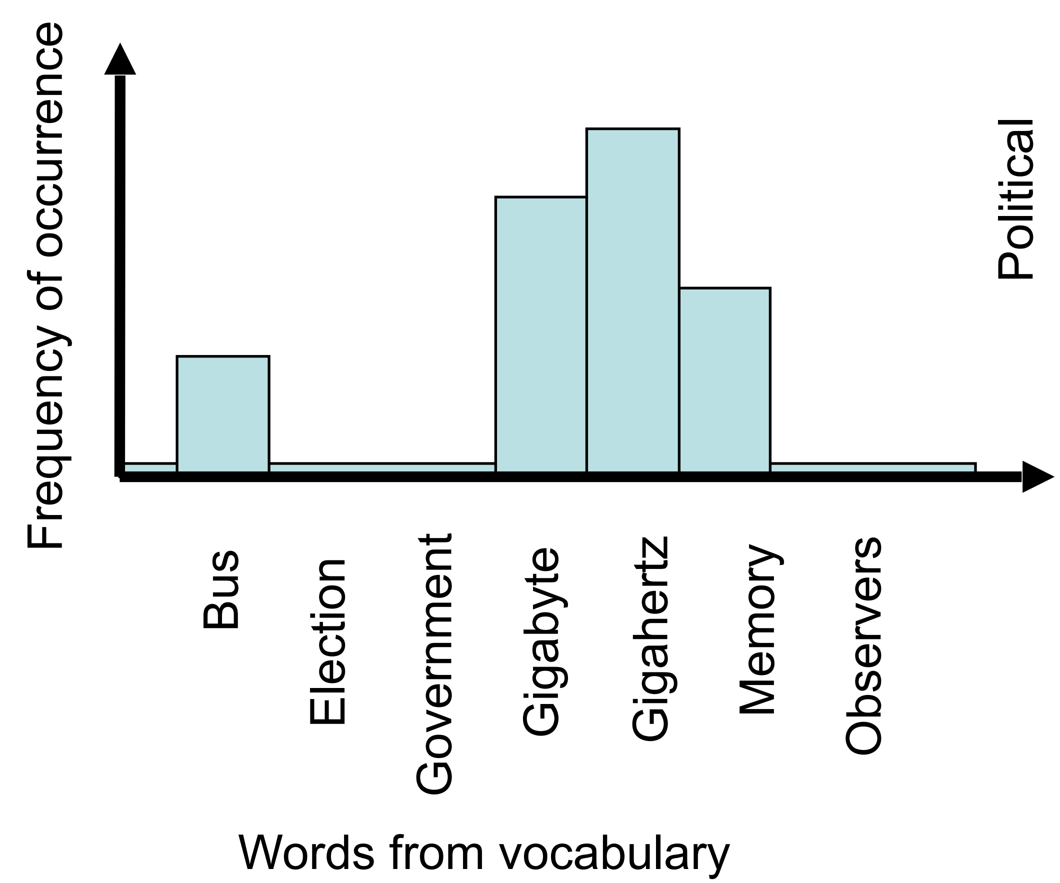
- SVD Matlab demo
- Texture
- Bag-of-words
- (Spatial) pyramid matching

Analogy with “bag-of-words” for document processing

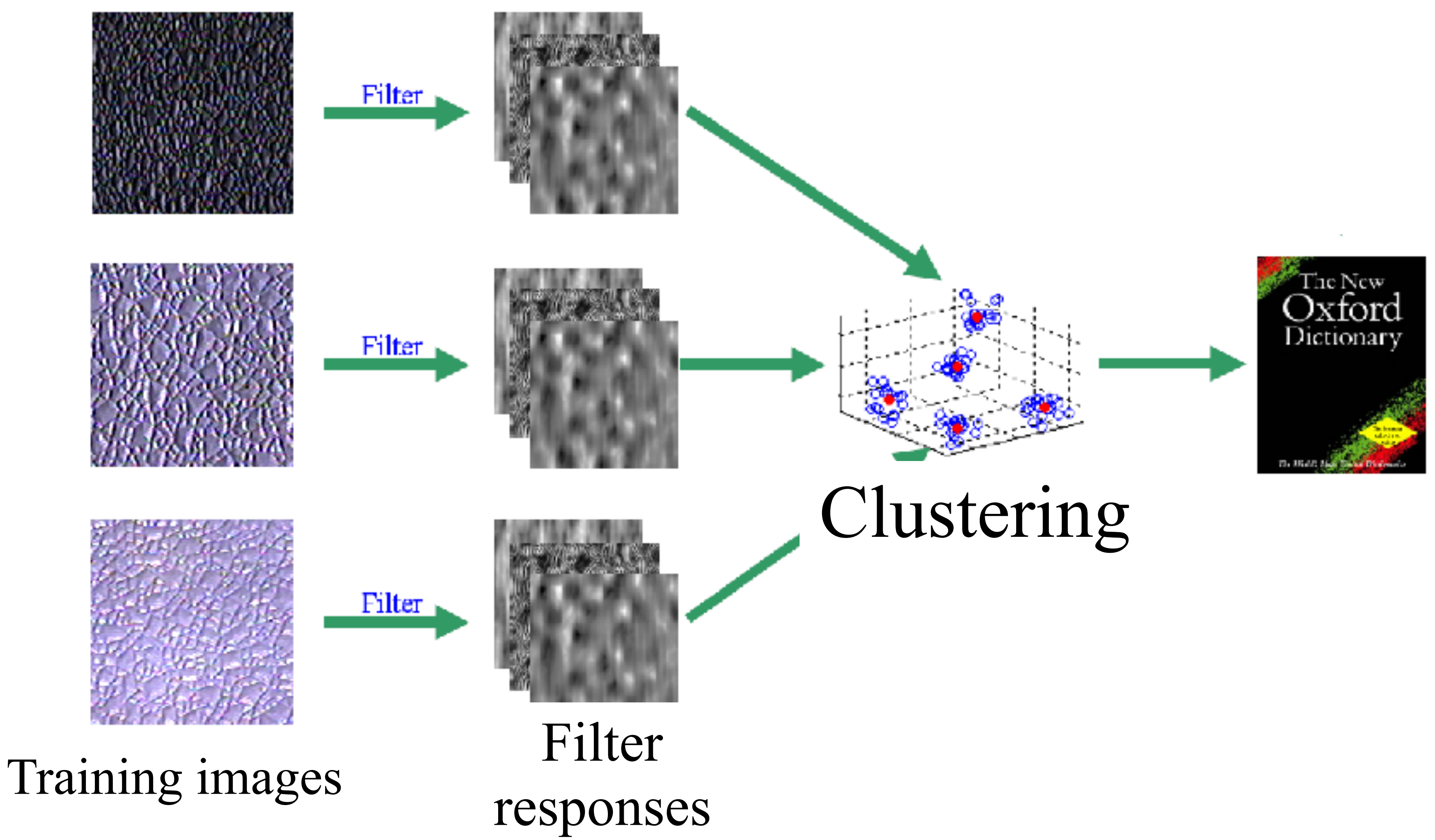
Political observers say that the government of Zorgia does not control the political situation. The government will not hold elections …

The ZH-20 unit is a 200Gigahertz processor with 2Gigabyte memory. Its strength is its bus and high-speed memory……





Bag-of-visual-words



Object → Bag of ‘words’

ICCV 2005 short course, L. Fei-Fei

Recognition with bag-of-words

- Summarize entire image based on its distribution (histogram) of word occurrences.
- Compare to stored library of images (or class-specific *models*)

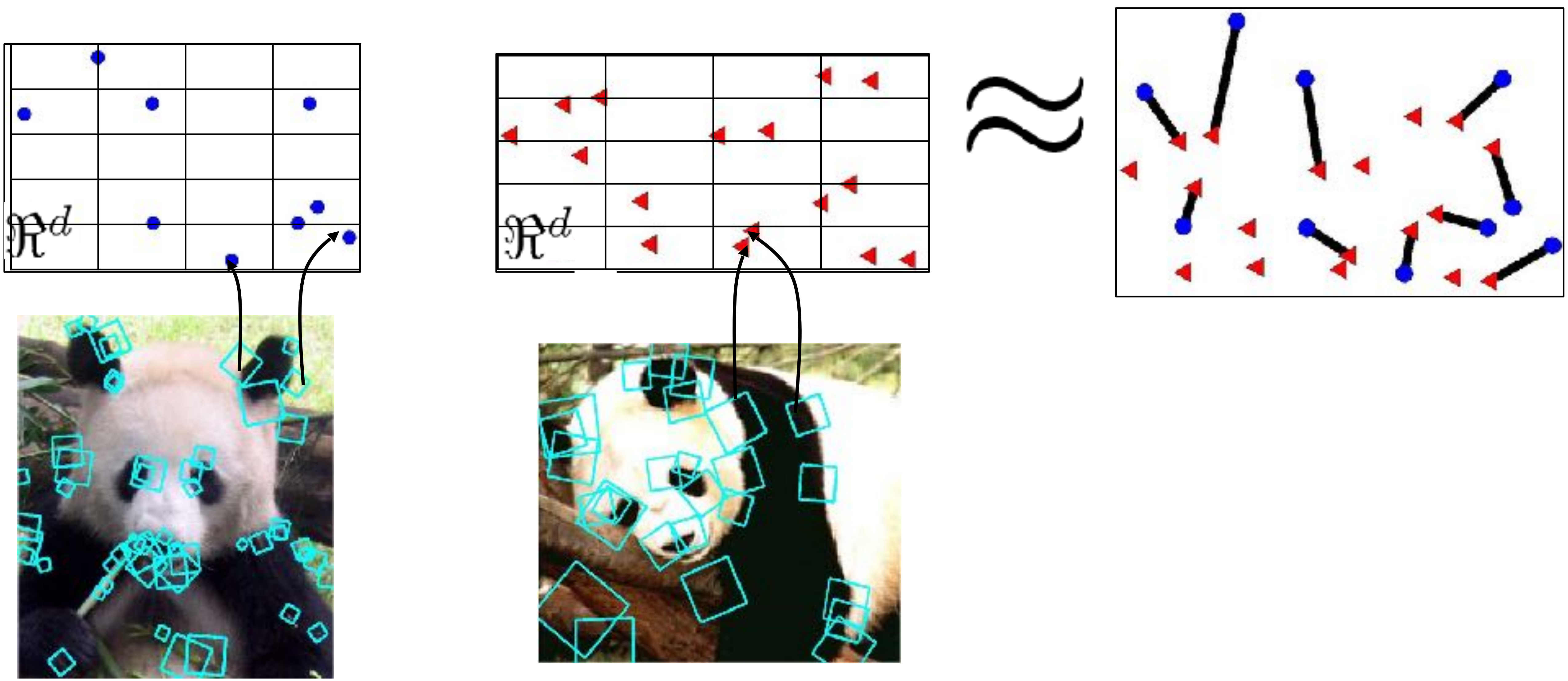
Image credit: Fei-Fei Li

Texture

- SVD Matlab demo
- Texture
- Bag-of-words
- (Spatial) pyramid matching

Digression: alternative to quantization

Approximate matching with histogram similarity

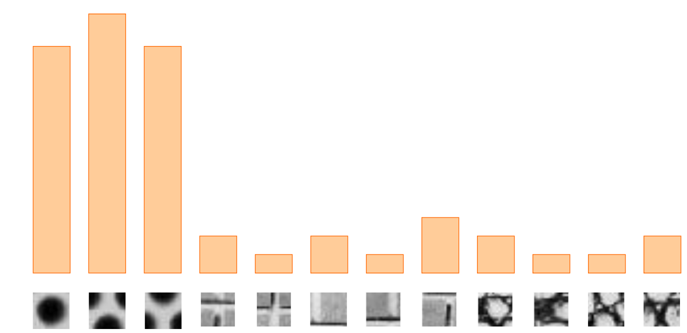


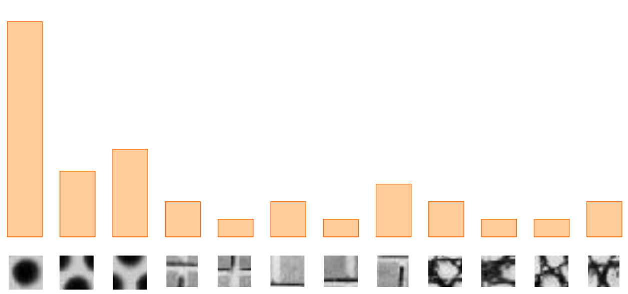
$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots, \vec{\mathbf{x}}_m\}$
$\vec{\mathbf{x}}_i \in \Re^d$

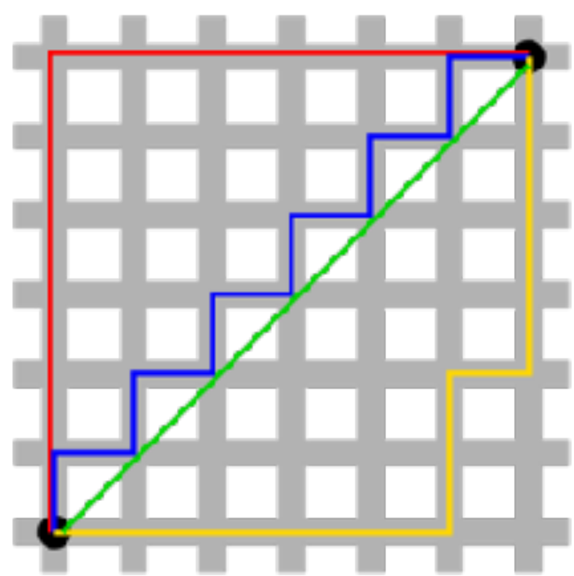
$\mathbf{Y} = \{\vec{\mathbf{y}}_1, \ldots, \vec{\mathbf{y}}_n\}$
$\vec{\mathbf{y}}_i \in \Re^d$

Aside: what's the "right" way to compare histograms?

$$D_r(x,y) = \Big(\sum_i (x_i - y_i)^r\Big)^{\frac{1}{r}}$$







euclidean (r=2) or manhattan (r=1)

Aside: what's the "right" way to compare histograms?

euclidean (r=2)
or manhattan (r=1)

$$D_r(x,y) = \Big(\sum_i (x_i - y_i)^r\Big)^{\frac{1}{r}}$$

chi-squared distance
[derived from chi-squared text in statistics]

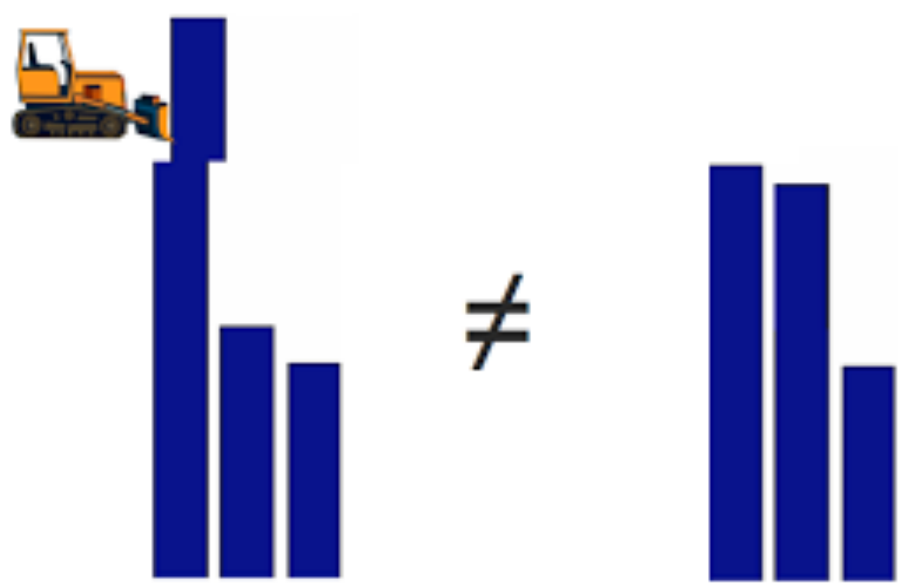
$$Chi(x,y) = \sum_i \frac{(x_i - y_i)^2}{x_i + y_i}$$

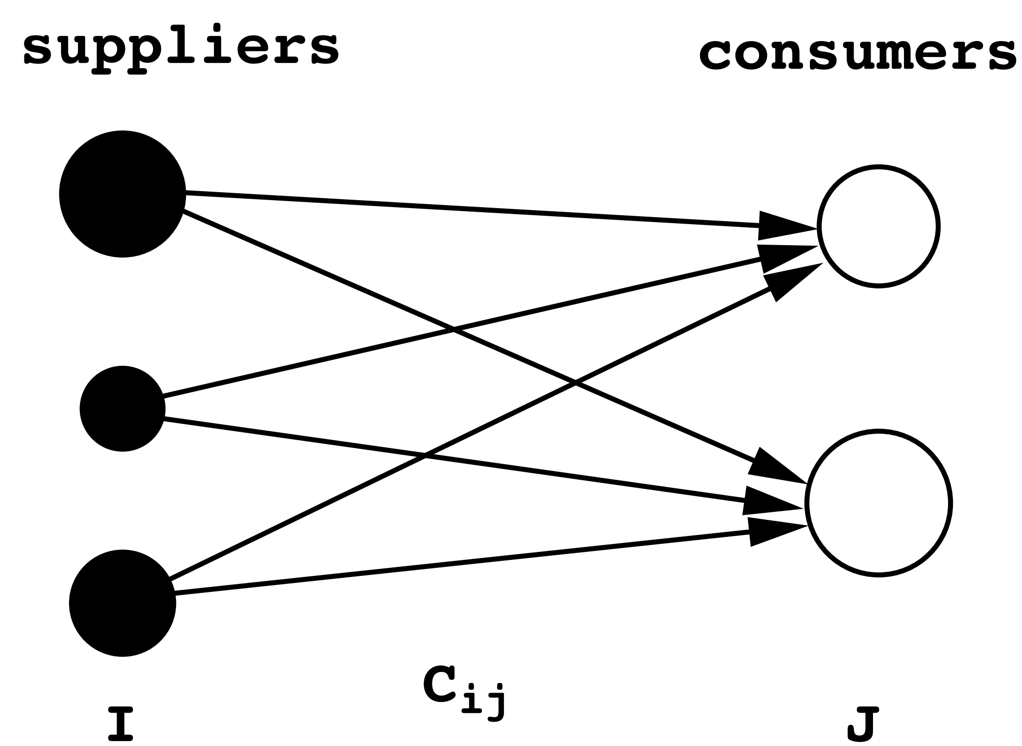
K-L divergence
[log probability of seeing x under model y]

$$D_{KL}(x,y) = \sum_i x_i \log \frac{x_i}{y_i}$$

Earth mover's distance

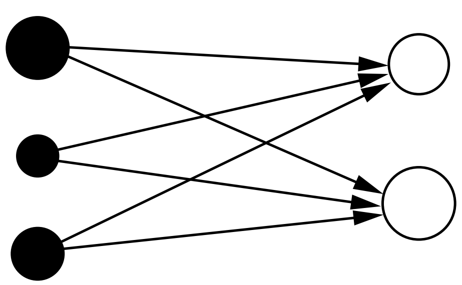
Cast as "transportation problem"





Earth mover's distance

Bipartite network flow



$$\min_{f_{ij}} \sum_{ij} c_{ij} f_{ij} \quad s.t.$$

$$f_{ij} \geq 0$$

$$\sum_i f_{ij} = y_j$$

$$\sum_j f_{ij} = x_i$$

$$EMD(\mathbf{x}, \mathbf{y}) = \sum_{ij} c_{ij} f_{ij}$$

https://www.cs.duke.edu/~tomasi/papers/rubner/rubnerTr98.pdf

Similarity Kernals

[sometimes more intuitive to define than distance functions]

$$D(x, y) = K(x, x) + K(y, y) - 2K(x, y)$$

$$K_{lin}(x, y) = \sum_i x_i y_i$$

[what's corresponding distance function?]

Similarity Kernals

[sometimes more intuitive to define than distance functions]

$$D(x, y) = K(x, x) + K(y, y) - 2K(x, y)$$

$$K_{lin}(x, y) = \sum_i x_i y_i$$

$$K_{int}(x, y) = \sum_i \min(x_i, y_i)$$

What happens if x,y are binary vectors?

Similarity Kernals

[sometimes more intuitive to define than distance functions]

$$D(x,y) = K(x,x) + K(y,y) - 2K(x,y)$$

$$K_{lin}(x,y) = \sum_i x_i y_i$$

$$K_{int}(x,y) = \sum_i \min(x_i, y_i)$$

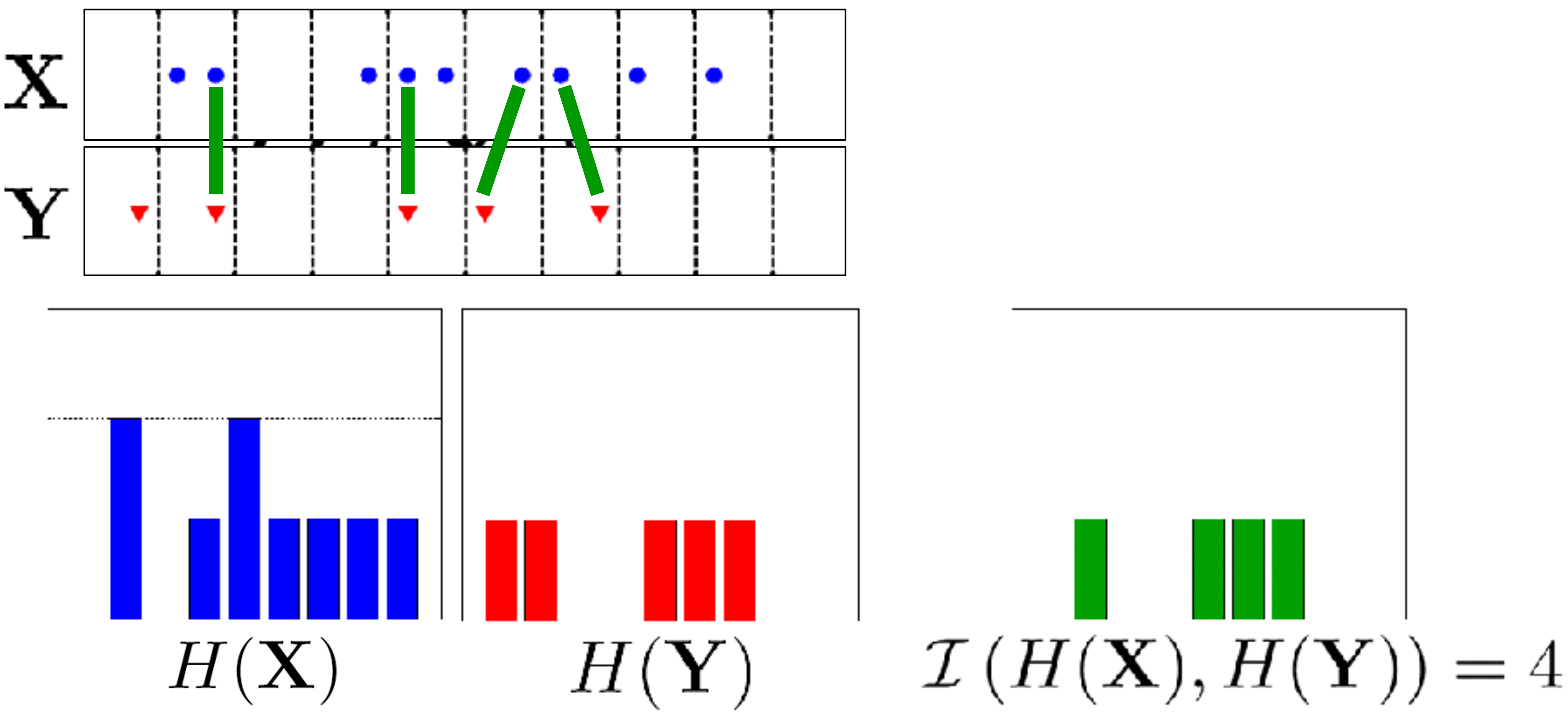
$$K_{bat}(x,y) = \sum_i \sqrt{x_i y_i}$$

It turns out, we can compute transformations f(x) and f(y) such that L2 distance in transformed space corresponds to these kernals (allows use of linear predictors)

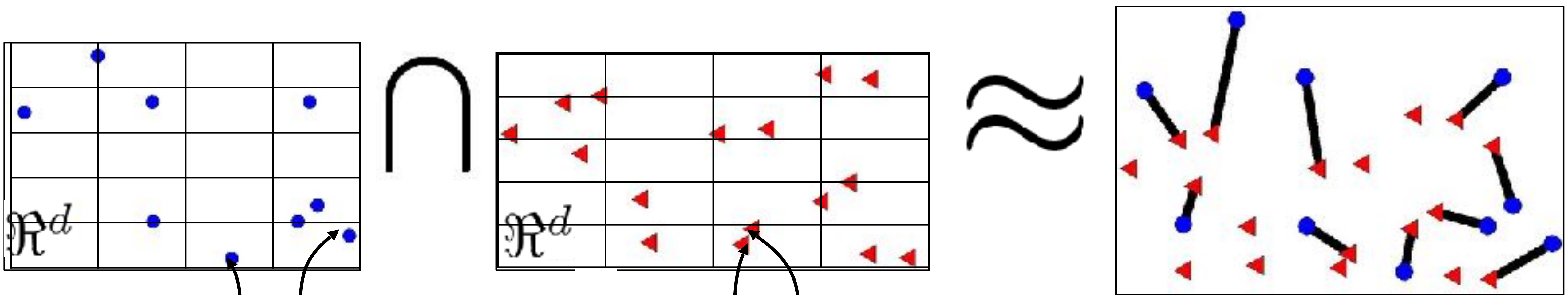
http://www.robots.ox.ac.uk/~vgg/software/homkermap/

Histogram intersection kernal

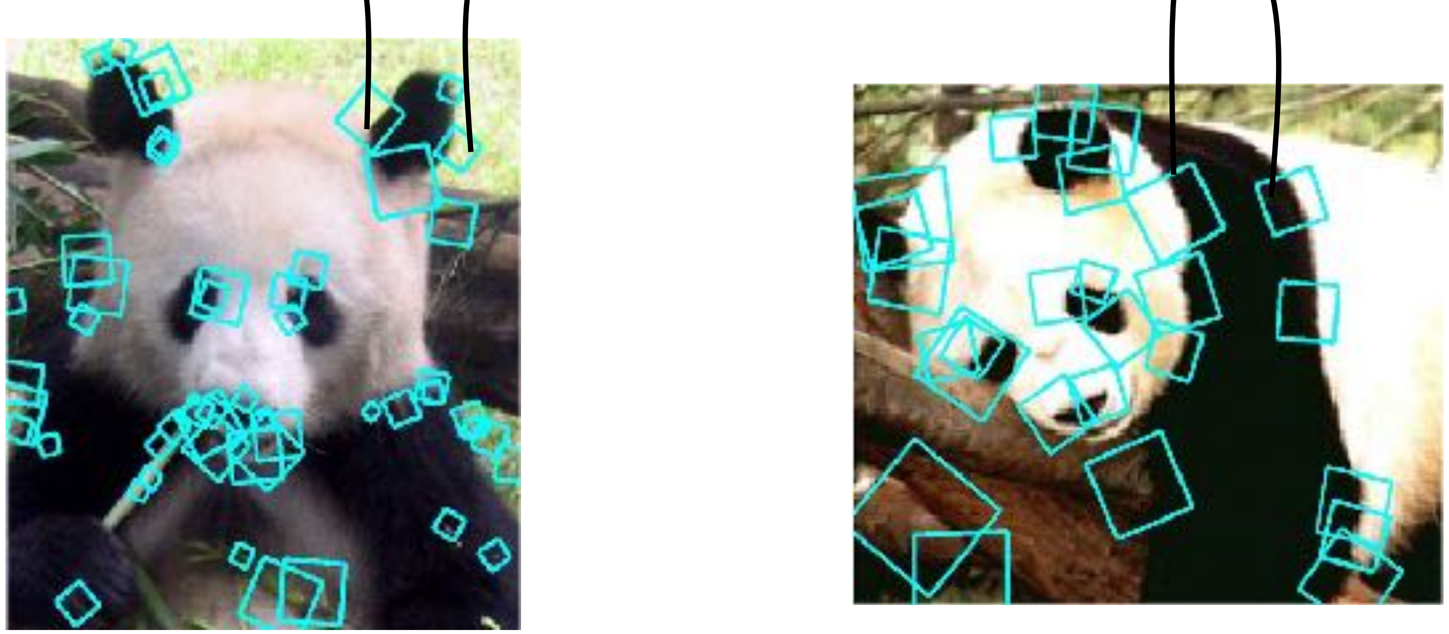
$$\mathcal{I}(H(\mathbf{X}), X(\mathbf{Y})) = \sum_k \min(H(\mathbf{X}_k), X(\mathbf{Y}_k))$$



Back to correspondence matching

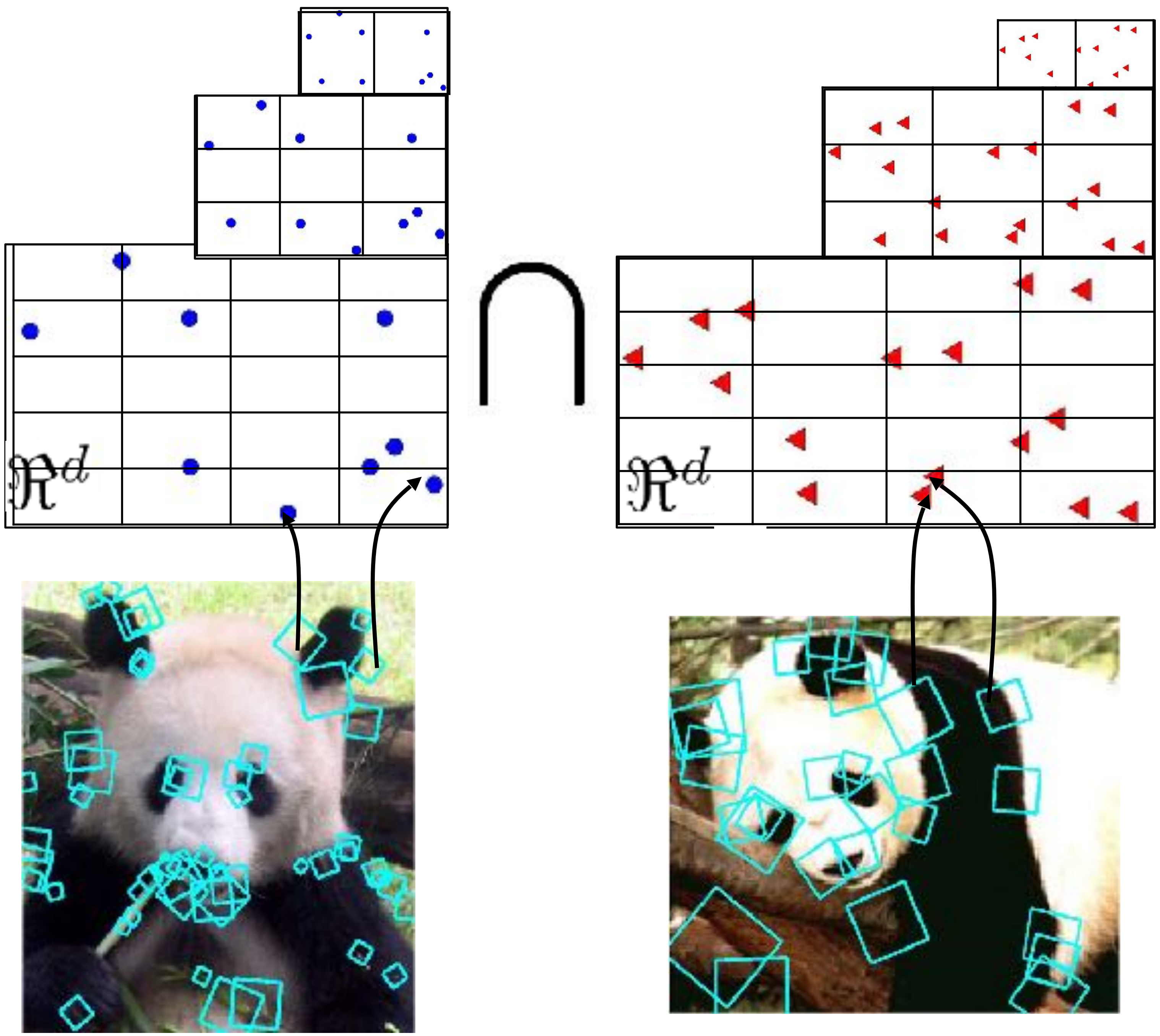


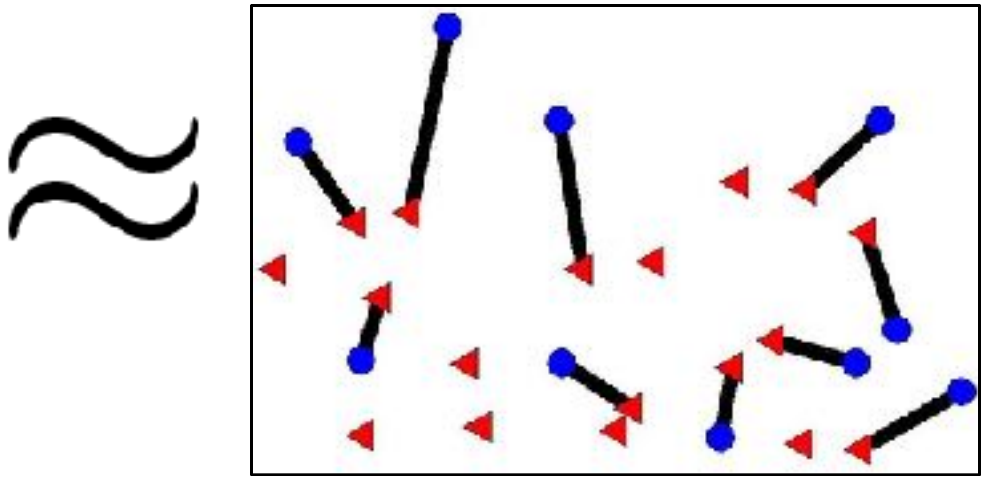
approximate match



But what about bin effects (partial credit for near matches)?

Back to correspondence matching





approximate match

Count matches obtained from larger bins

Counting new matches

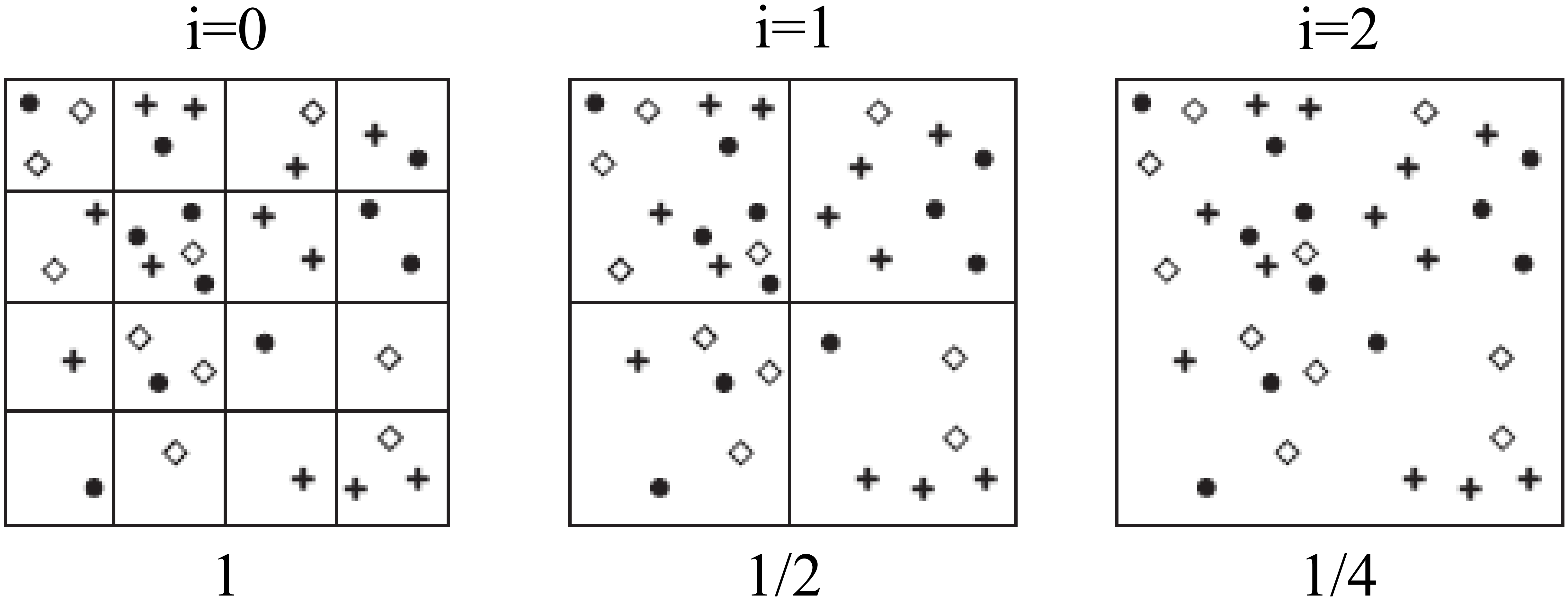
Histogram intersection

$$\mathcal{I}\left(H(\mathbf{X}), H(\mathbf{Y})\right) = \sum_{j=1}^{r} \min\left(H(\mathbf{X})_j, H(\mathbf{Y})_j\right)$$

$$N_i = \underbrace{\mathcal{I}\left(H_i(\mathbf{X}), H_i(\mathbf{Y})\right)}^{\text{matches at this level}} - \underbrace{\mathcal{I}\left(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})\right)}^{\text{matches at previous level}}$$

Difference in histogram intersections across levels counts *number of new pairs* matched

Giving partial credit for new matches



Weight new matches inversely porportional to bin size

$$\frac{1}{2^i}$$

Pyramid match kernel

$$K_\Delta\left(\overbrace{\Psi(\mathbf{X}),\Psi(\mathbf{Y})}^{\text{histogram pyramids}}\right) =$$

$$\sum_{i=0}^{L}\underbrace{\frac{1}{2^i}}_{\substack{\text{measure of difficulty of a}\\ \text{match at level } i}}\Big(\underbrace{\mathcal{I}\left(H_i(\mathbf{X}),H_i(\mathbf{Y})\right)-\mathcal{I}(H_{i-1}(\mathbf{X}),H_{i-1}(\mathbf{Y}))}_{\text{number of newly matched pairs at level } i}\Big)$$

- Weights inversely proportional to bin size
- Normalize kernel values to avoid favoring large sets

Spatial Pyramid Matching

Quantize features into words, but build pyramid in space

Nifty way to encode constraints like “eye” words lie near top of image

Original images

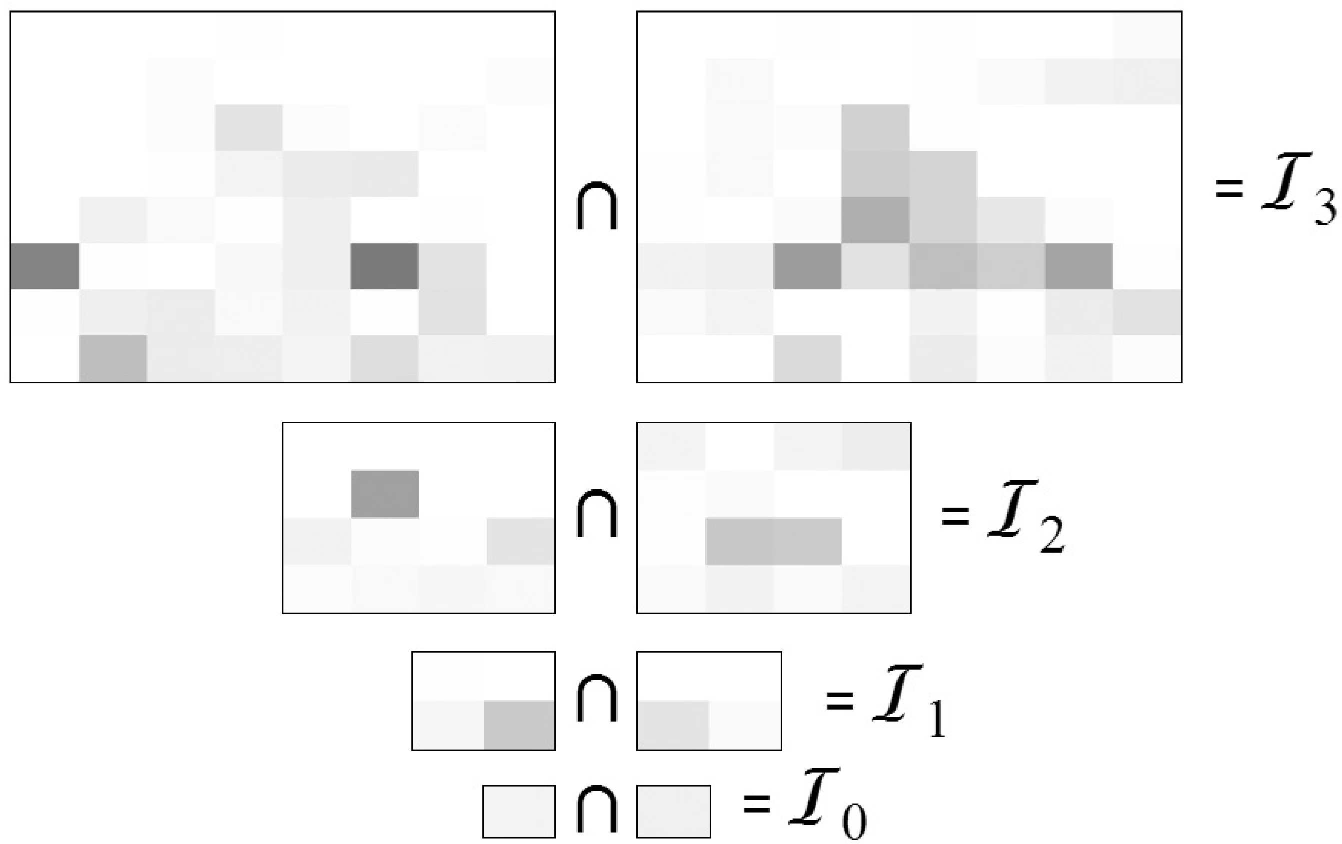
Feature histograms:

Level 3

Level 2

Level 1

Level 0



Total weight (value of *pyramid match kernel*): $\mathcal{I}_3 + \frac{1}{2}(\mathcal{I}_2 - \mathcal{I}_3) + \frac{1}{4}(\mathcal{I}_1 - \mathcal{I}_2) + \frac{1}{8}(\mathcal{I}_0 - \mathcal{I}_1)$

Texture

- SVD Matlab demo
- Texture
- Bag-of-words
- (Spatial) pyramid matching