Deep Learning for Computer Vision

Spring 2019

http://vllab.ee.ntu.edu.tw/dlcv.html (primary)

https://ceiba.ntu.edu.tw/1072CommE5052_ (grade, etc.)

FB: DLCV Spring 2019

Yu-Chiang Frank Wang 王鈺強, Associate Professor

Dept. Electrical Engineering, National Taiwan University

2019/03/06

TA Hours

李宇哲

Mon. 13:30 ~ 15:30

BL-527

楊福恩

Tue. 13:00 ~ 15:00

BL-527

李元顥

Wed. 13:00 ~ 15:00

BL-527

劉致廷

Wed. 15:00 ~ 17:00

BL-421

吳致緯

Thu. 10:00 ~ 12:00

BL-421

郭冠軒

Thu. 14:00 ~ 16:00

BL-527

陳尚甫

Fri. 13:00 ~ 15:00

BL-527

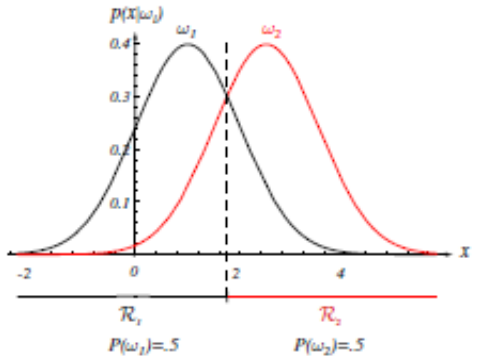
- FB: **DLCV Spring 2019**
- Feedback are welcome (to TA/me).

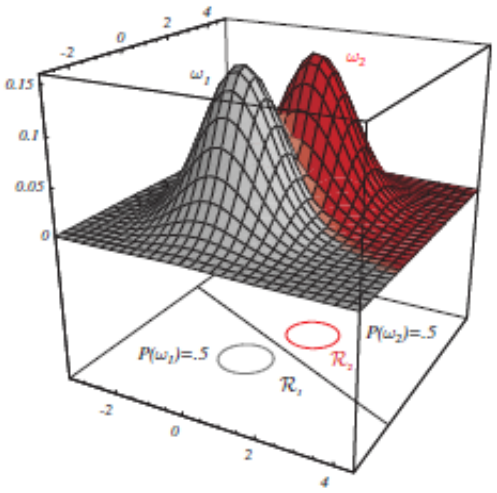
Tight (yet tentative) Schedule

Week	Date	Topic	Remarks
0	2/20	Course Logistics + Intro to Computer Vision	
1	2/27	**Machine Learning 101**	
2	3/06	**Image Representation & Recognition**	**HW #1 out**
3	3/13	Intro to Neural Networks + CNN (I)	
4	3/20	Intro to Neural Networks + CNN (II) Tutorial on Python, GitHub, etc.	**HW #1 due**
5	3/27	Detection & Segmentation	HW #2 out
6	4/03	Spring Break!	
7	4/10	Generative Models	
8	4/17	Visualization and Understanding NNs	HW #3 out, HW #2 due
9	4/24	Transfer Learning for Visual Analysis	
10	5/01	Recurrent NNs and Seq-to-Seq Models (I)	Team Up for Final Projects
11	5/08	Guest Lecture	HW #4 out, HW #3 due
12	5/15	Recurrent NNs and Seq-to-Seq Models (I)	
13	5/22	Learning Beyond Images (2D/3D, depth, etc.)	
14	5/29	Deep Reinforcement Learning for Visual Apps	HW #4 due
15	6/05	Final Project Checkpoint	
16	6/12 or 6/19	Final Presentation	

What We Covered Last Week…

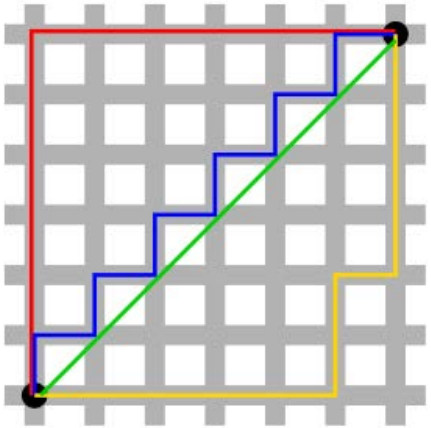
- From Probability to Bayes Decision Rule
- Brief Review of Linear Algebra & Linear System
- Unsupervised vs. Supervised Learning
  - Clustering & Dimension Reduction
  - Training, testing, & validation
  - Linear Classification





Revisit of Clustering

- Similarity:
  - A key component/measure to perform data clustering
  - Inversely proportional to distance
  - Example distance metrics:
    - Euclidean distance (L2 norm): $d(x, z) = \|x - z\|_2 = \sqrt{\sum_{i=1}^{D}(x_i - z_i)^2}$
    - Manhattan distance (L1 norm): $d(x, z) = \|x - z\|_1 = \sum_{i=1}^{D}|x_i - z_i|$



K-Means Clustering (cont'd)

- Remarks
  - Recall that, we have $\|x\|_p = \left(\sum_{i=1}^{d} x_i^p\right)^{1/p}$
    - L_2/Euclidean norm $\|x\|_2$
    - L_1 norm $\|x\|_1 = \sum_{i=1}^{d} |x_i|$
    - Maximum norm $\|x\|_\infty = \max\{x_1, \cdots, x_d\}$
    - L_0 norm: $\|x\|_0 = \lim_{p \to 0} \left\{(x_1)^p + \cdots + (x_d)^p\right\}^{\frac{1}{p}}$
      $= \#$ of non-zero elements (out of d)

"sparsity" $\|\alpha_i\|_0 = 1$

s.t. $\|\alpha_i\|_2 = 1$

最終目的一樣，只是 $\|\alpha_i\|_1 = 1$

1-norm 比較好學

$\sum \|x_i - D\alpha_i\|_2$

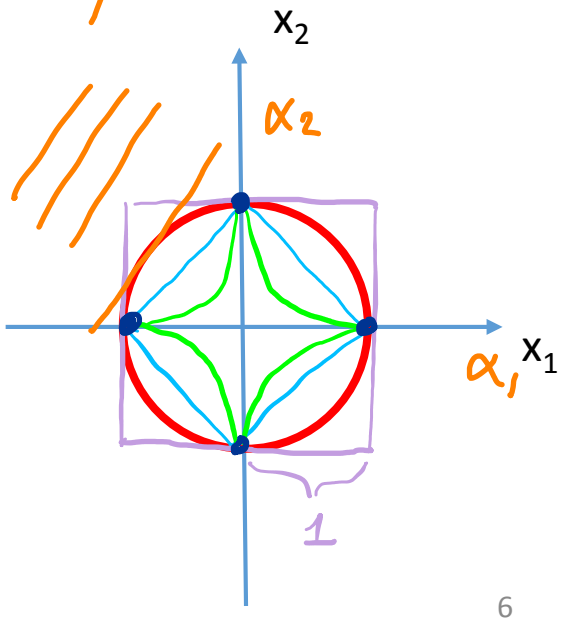
smooth

$\|x\|_\infty = 1$

$\|x\|_2 = 1$ ↑ 代表六維度都有值

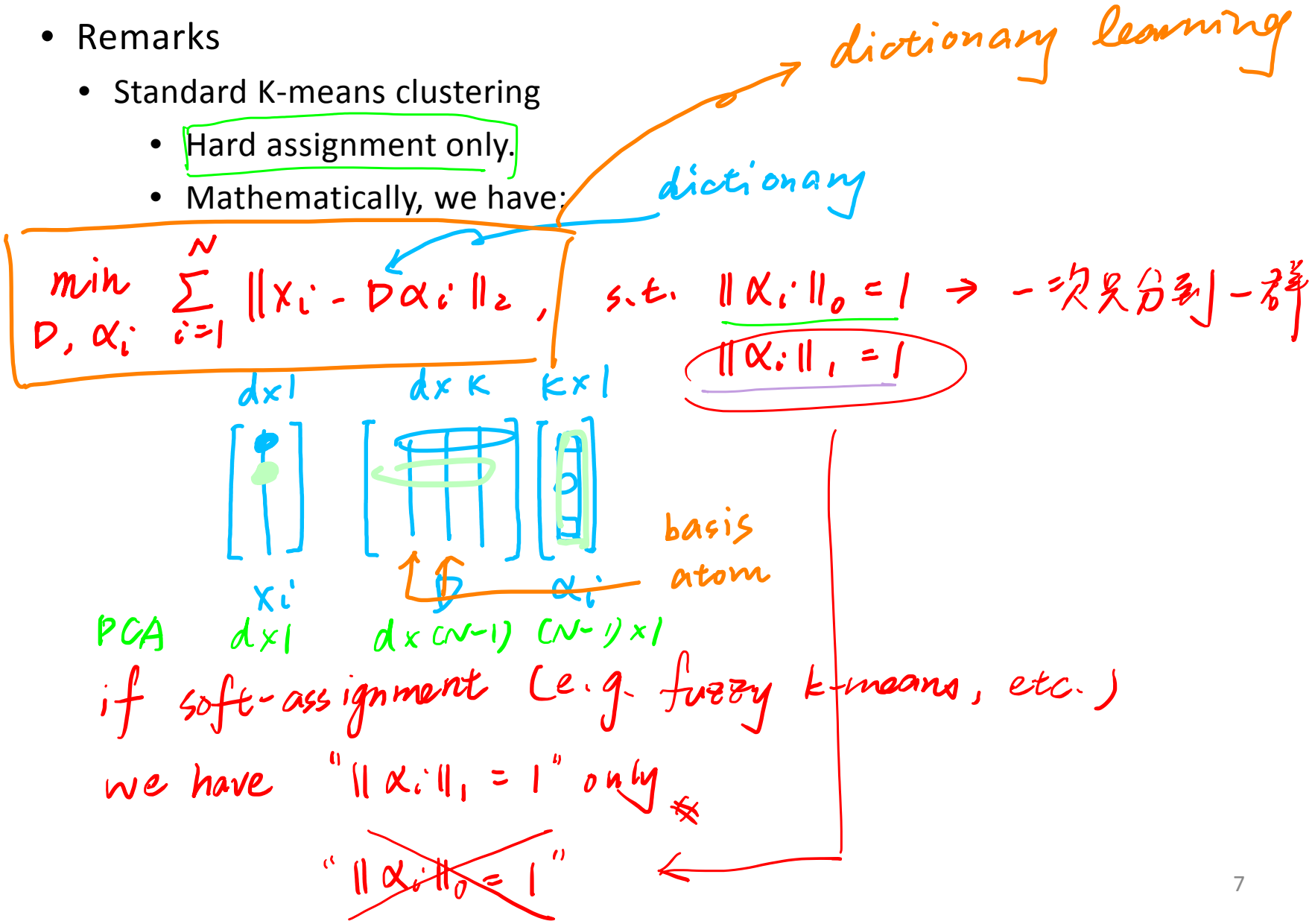
$\|x\|_1 = 1$

$\|x\|_0 = 1$



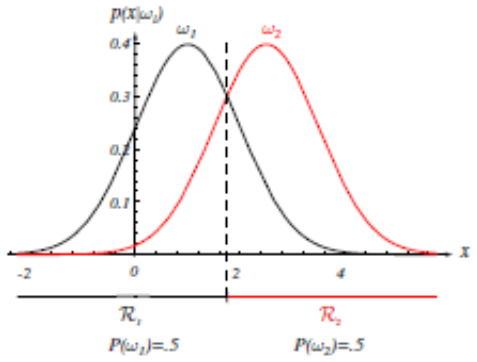
K-Means Clustering (cont'd)

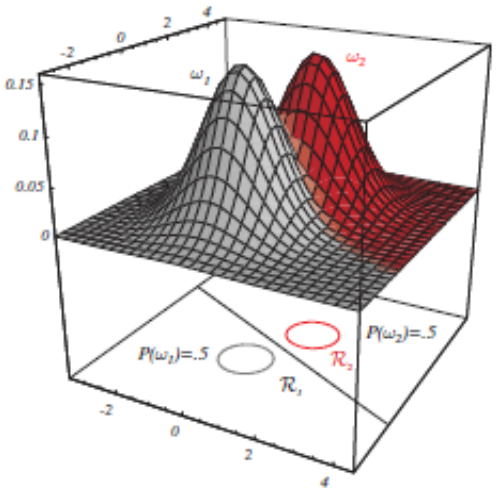
- Remarks
  - Standard K-means clustering
    - Hard assignment only.
    - Mathematically, we have:



What We Covered Last Week…

- From Probability to Bayes Decision Rule
- Brief Review of Linear Algebra & Linear System
- Unsupervised vs. Supervised Learning
  - Clustering & Dimension Reduction
  - Training, testing, & validation
  - Linear Classification





Eigenanalysis & PCA (cont'd)

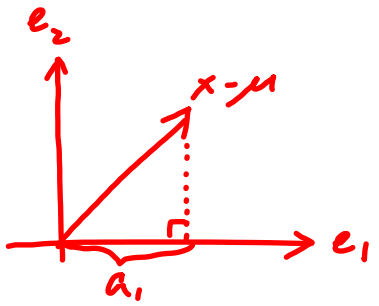
- A *d* x *d* covariance matrix contains a maximum of *d* eigenvector/eigenvalue pairs.
  - Assuming you have N images of size M x M pixels, we have dimension $d = M^2$.
  - With the rank of ∑ as N-1, we have at most N-1 non-zero eigenvalues.
  - How dimension reduction is realized? how to reconstruct the input data?

$$\mathbb{R}^d \rightarrow x_i - \mu = \sum_{i=1}^{d} \alpha_i e_i$$

e_1^T (投影)

$$= \sum_{i=1}^{N-1} a_i e_i, \quad N-1 \ll d$$

$$a = \begin{bmatrix} a_1 \\ \vdots \\ a_{N-1} \end{bmatrix} \in \mathbb{R}^{N-1}$$



$$= a_1 e_1 + a_2 e_2 + a_3 e_3 + \cdots + a_{N-1} e_{N-1}$$

e_1^T

$$a' = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$$

- Expanding a signal via eigenvectors as bases
  - With symmetric matrices (e.g., covariance matrix), eigenvectors are orthogonal.
  - They can be regarded as unit basis vectors to span any instance in the d-dim space.

PCA for Data Whitening

$$\Sigma = \frac{1}{N}\sum_{i=1}^{N}\{(x_i - \mu)(x_i - \mu)^T\}$$

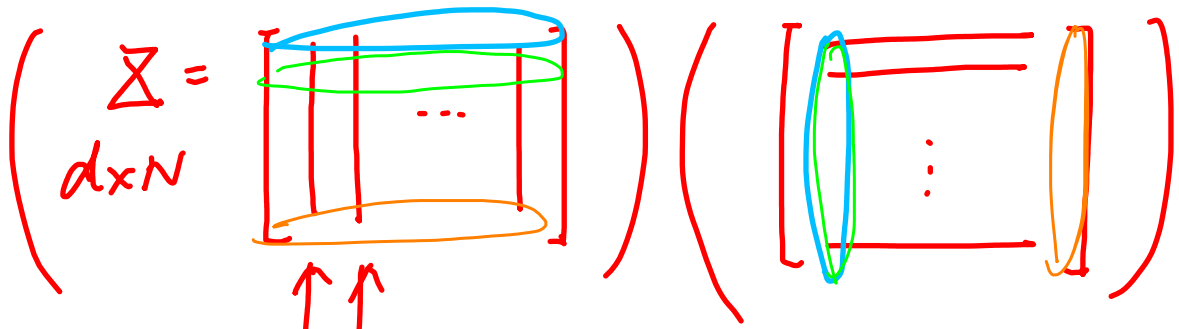
$$= \frac{1}{N}\{\square + \square + \cdots + \square\}$$

$$= X \cdot X^T = \square$$

$$\Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_d \end{bmatrix} \in \mathbb{R}^{d \times d}$$

$$E = [e_1, \cdots, e_d] \in \mathbb{R}^{d \times d}$$

X ($d \times N$) $\quad X^T$ ($N \times d$)



$x_1 - \mu$

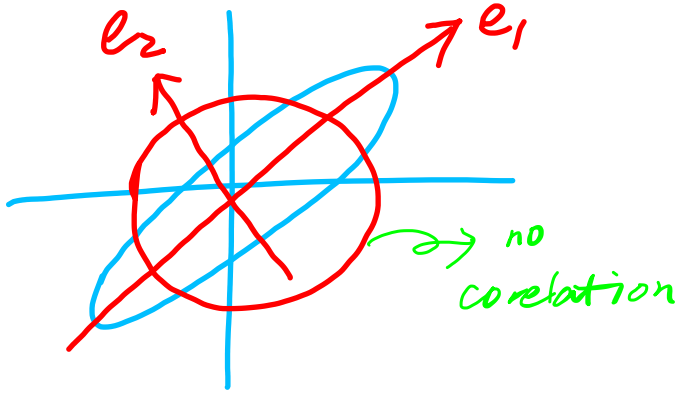
$x_2 - \mu$

$$\Sigma \cdot E = \begin{bmatrix} \cdots \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 & & \\ & \lambda_2 & 0 \\ & 0 & \ddots \\ & & \lambda_d \end{bmatrix}$$

$$= \square$$

Singular Value Decomp.

PCA vs. SVD



$$\Sigma \cdot E = E \cdot \Lambda$$

$$E^T \Sigma \cdot \Sigma^T \cdot E = E^T E \cdot \Lambda = I\Lambda = \begin{bmatrix} \ddots \end{bmatrix}$$

$$\Sigma' = (\Sigma')(\Sigma')^T = \Lambda$$

→ data whitening

$$\underset{d\times N}{\Sigma} = \underset{d\times d}{U} \cdot \underset{d\times N}{S} \cdot \underset{N\times N}{V^T}$$

$$\begin{cases} U^T U = I \\ V^T V = I \end{cases}, \quad S = d\begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_d \end{bmatrix}_N$$

$$\Sigma \cdot \Sigma^T = (U \cdot S \cdot V^T)(V \cdot S^T \cdot U^T)$$

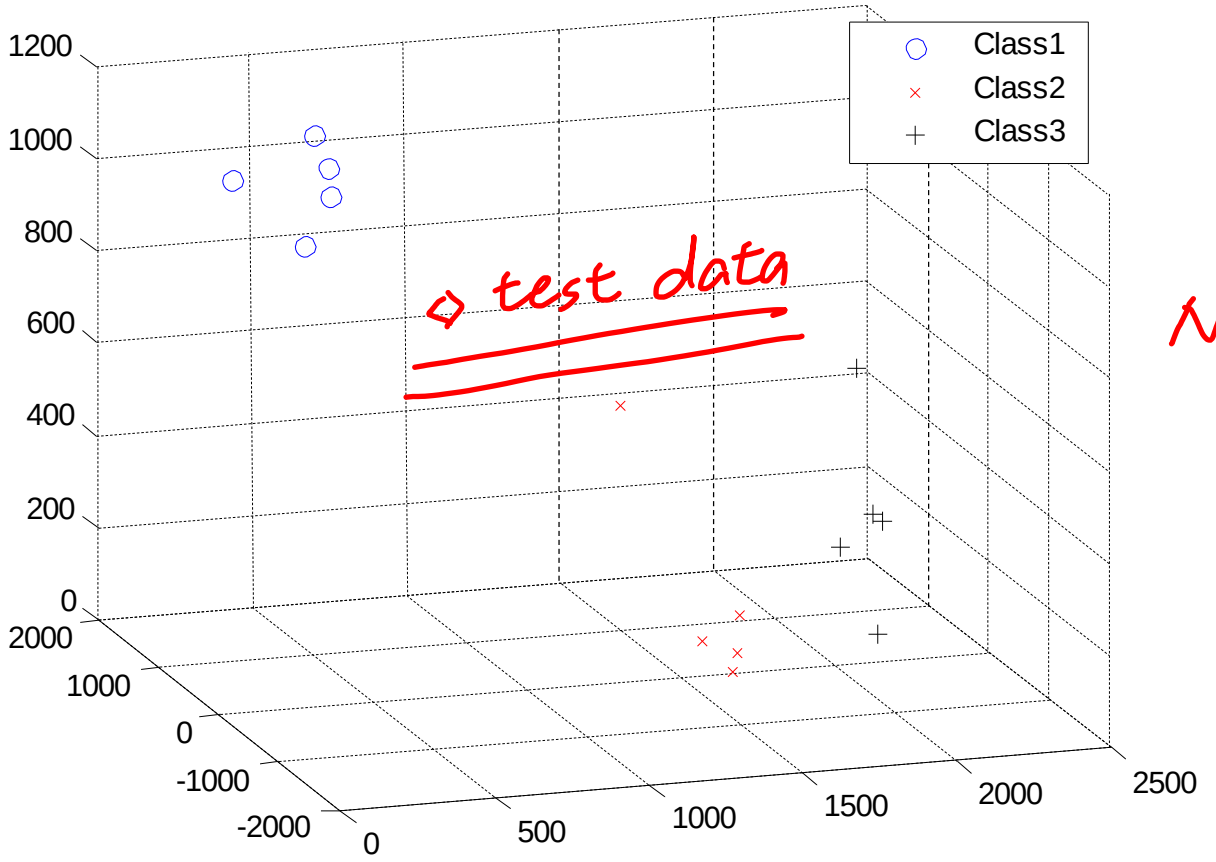
$$= \underset{d\times d}{U} \underset{d\times d}{S'} \underset{d\times d}{U^T}$$

$$S' = \begin{bmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_d^2 \end{bmatrix}$$

$$\Sigma \cdot \Sigma^T U = U S'$$

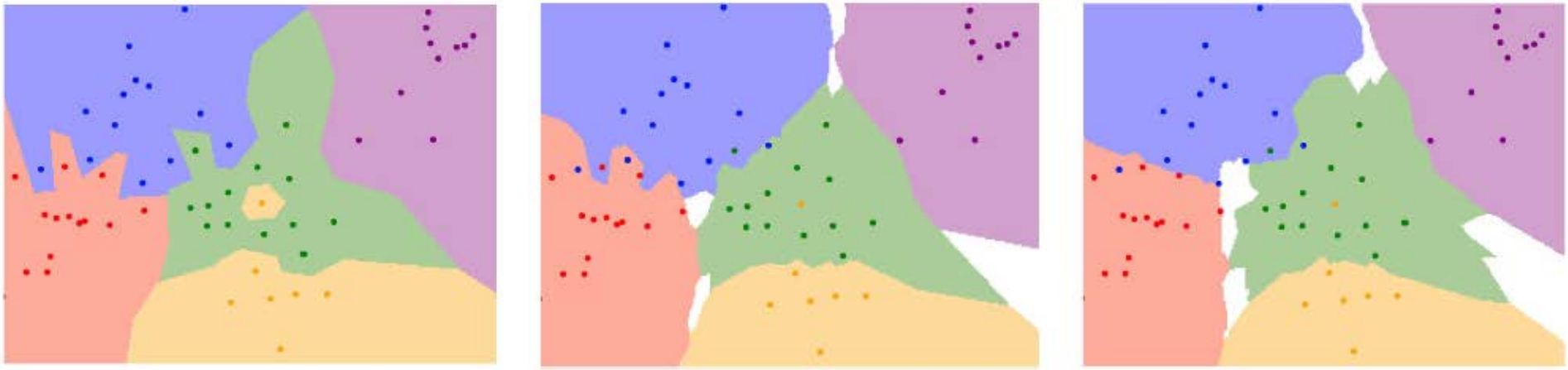
Final Remarks

- Linear & unsupervised dimension reduction
- PCA can be applied as a feature extraction/preprocessing technique.
  - E.g,, Use the top 3 eigenvectors to project data into a 3D space for classification.



Final Remarks (cont'd)

- How do we classify? For example…
  - Given a test face input, project into the same 3D space (by the same top-3 eigenvectors).
  - The resulting vector in the 3D space is the feature for this test input.
  - We can do a simple Nearest Neighbor (NN) classification with Euclidean distance, which calculates the distance to all the projected training data in this space.
  - If NN, then the label of the closest training instance determines the classification output.
  - If k-nearest neighbors (k-NN), then k-nearest neighbors need to vote for the decision.



k = 1 k = 3 k = 5

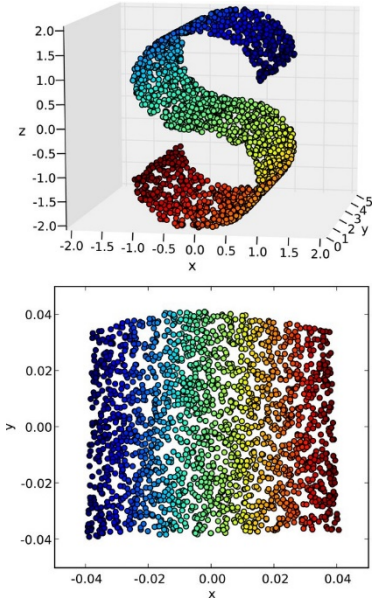
Demo available at http://vision.stanford.edu/teaching/cs231n-demos/knn/

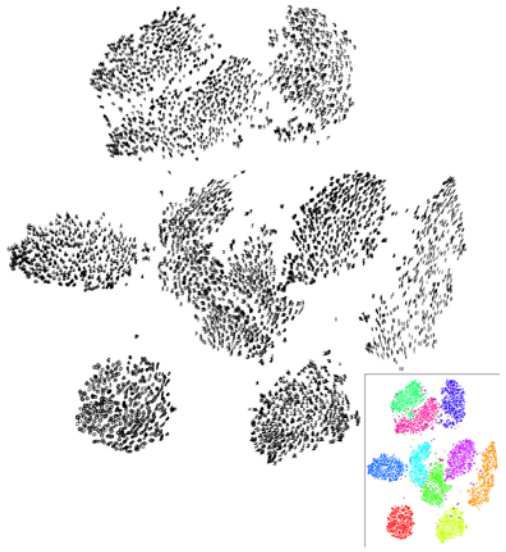
Image credit: Stanford CS231n

Final Remarks on PCA

- If labels for each data is provided → Linear Discriminant Analysis (LDA)
  - LDA is also known as Fisher's discriminant analysis.
  - Eigenface vs. Fisherface (IEEE Trans. PAMI 1997)
- If linear DR is not sufficient, and non-linear DR is of interest…
  - Isomap, locally linear embedding (LLE), etc.
  - **t-distributed stochastic neighbor embedding (t-SNE)** (by G. Hinton & L. van der Maaten)

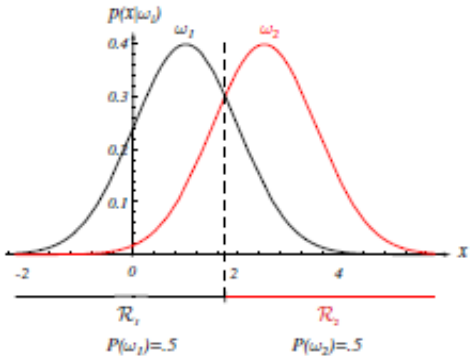
投影後還是鄰居

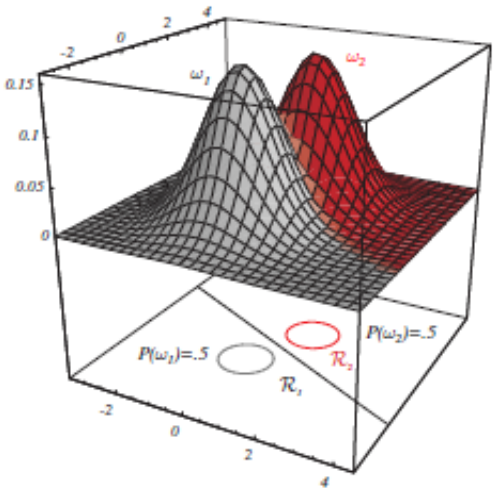




What's to Be Covered in Part I Today…

- From Probability to Bayes Decision Rule
- Brief Review of Linear Algebra & Linear System
- Unsupervised vs. Supervised Learning
  - Clustering & Dimension Reduction
  - Training, testing, & validation
  - Linear Classification

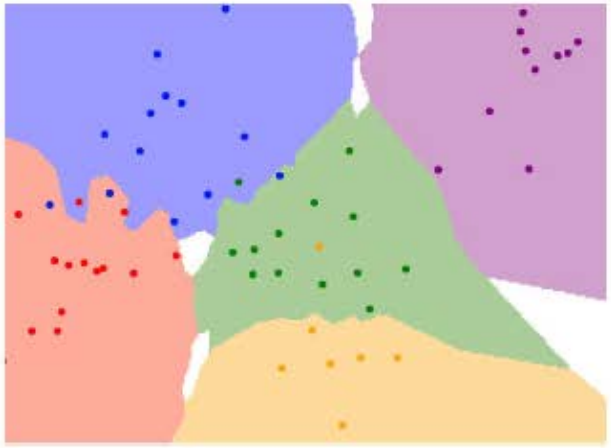




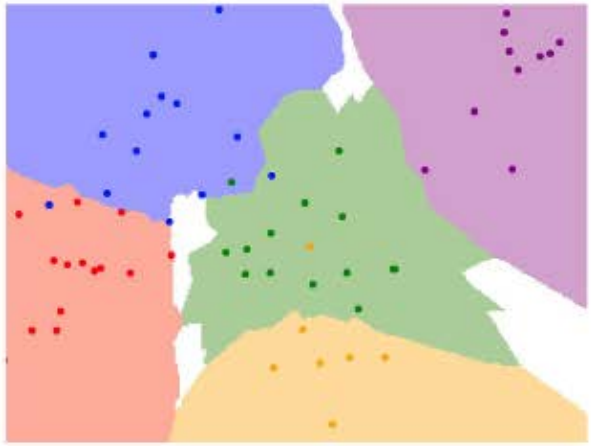
Hyperparameters in ML

- Recall that for k-NN, we need to determine the k value in advance.
  - What is the best k value?
  - And, what is the best distance/similarity metric?
  - Similarly, take PCA for example, what is the best reduced dimension number?
- **Hyperparameters**: choices about the learning model/algorithm of interest
  - We need to determine such hyperparameters instead of learn them.
  - Let's see what we can do and cannot do…

k = 1



k = 3



k = 5

Image credit: Stanford CS231n

How to Determine Hyperparameters?

- Idea #1
  - Let's say you are working on face recognition.
  - You come up with your very own feature extraction/learning algorithm.
  - You take a dataset to train your model, and select your hyperparameters based on the resulting performance.
  - 

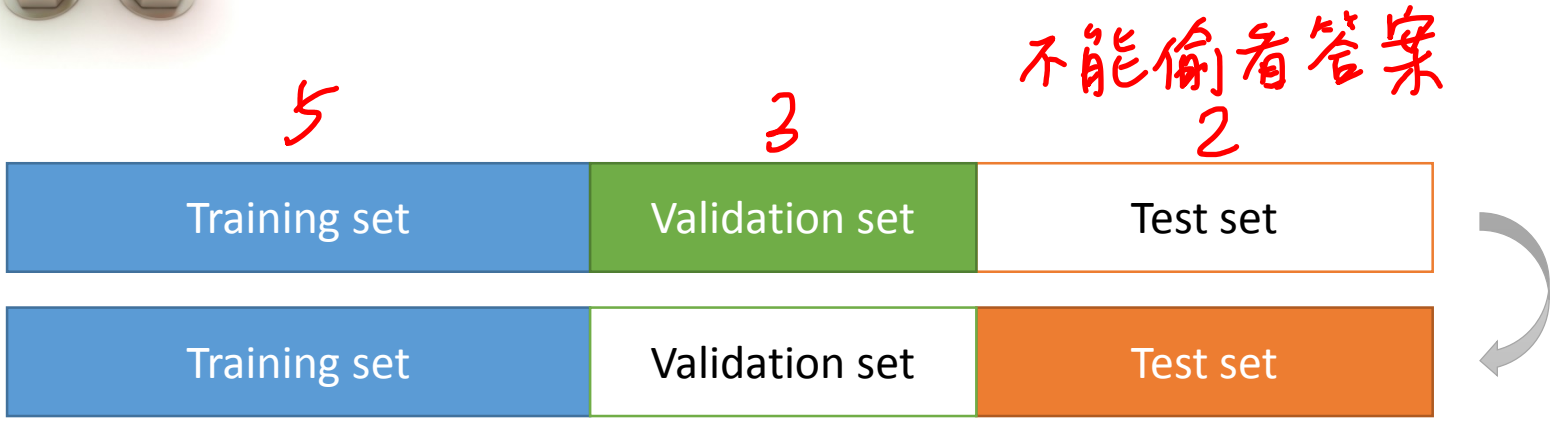
Dataset

How to Determine Hyperparameters? (cont'd)

- Idea #2
  - Let's say you are working on face recognition.
  - You come up with your very own feature extraction/learning algorithm.
  - For a dataset of interest, you split it into training and test sets.
  - You train your model with possible hyperparameter choices, and select those work best on test set data.
  - 

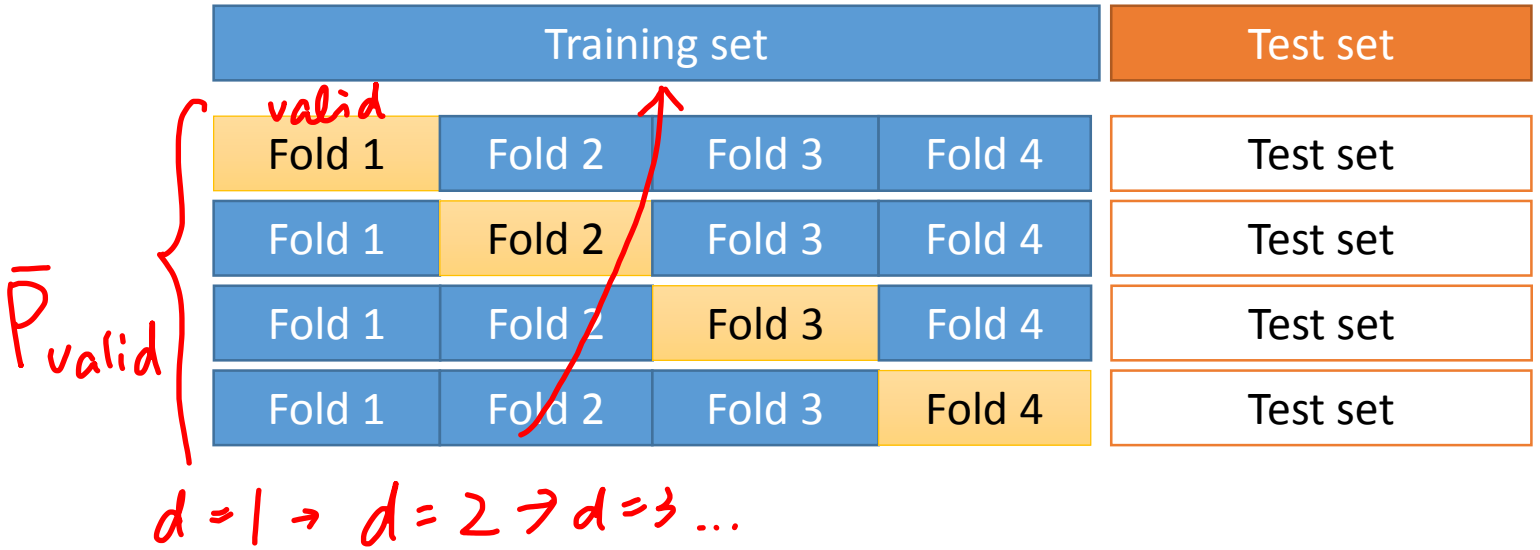
How to Determine Hyperparameters? (cont'd)

- Idea #3
  - Let's say you are working on face recognition.
  - You come up with your very own feature extraction/learning algorithm.
  - For the dataset of interest, it is split it into training, validation, and test sets.
  - You train your model with possible hyperparameter choices, and select those work best on the validation set.
  - 



How to Determine Hyperparameters? (cont'd)

- Idea #3.5
  - What if only training and test sets are given, not the validation set?
  - Cross-validation (or *k-fold* cross validation)
    - Split the training set into k folds with a hyperparameter choice
    - Keep 1 fold as validation set and the remaining k-1 folds for training
    - After each of k folds is evaluated, report the average validation performance.
    - Choose the hyperparameter(s) which result in the highest average validation performance.
  - Take a 4-fold cross-validation as an example… (k = 4)



Minor Remarks on NN-based Methods

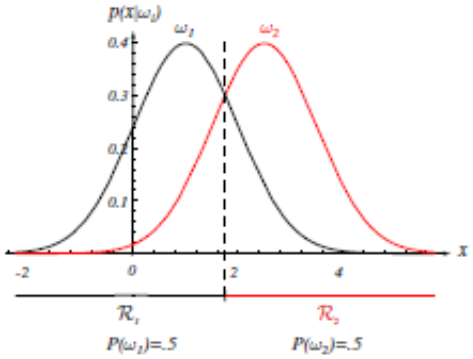
- In fact, k-NN (or even NN) is not of much interest in practice. Why?
  - Choice of distance metrics might be an issue. See example below.
  - Measuring distances in high-dimensional spaces might not be a good idea.
  - Moreover, NN-based methods require lots of and !
    (That is why NN-based methods are viewed as *data-driven* approaches.)

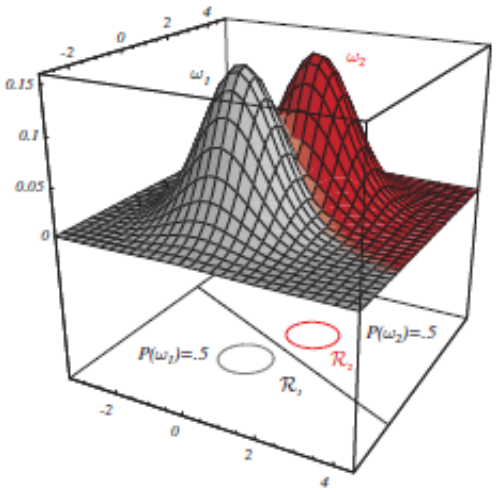
All three images have the same Euclidean distance to the original (left) one.

Image credit: Stanford CS231n

What We Covered Last Week…

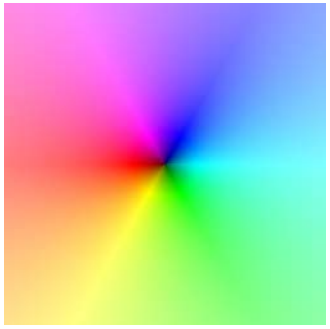
- From Probability to Bayes Decision Rule
- Brief Review of Linear Algebra & Linear System
- Unsupervised vs. Supervised Learning
  - Clustering & Dimension Reduction
  - Training, testing, & validation
  - Linear Classification (move to the next lecture)

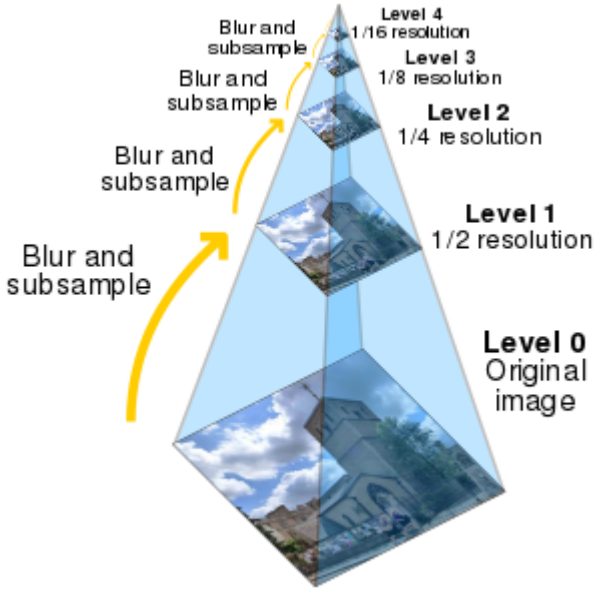




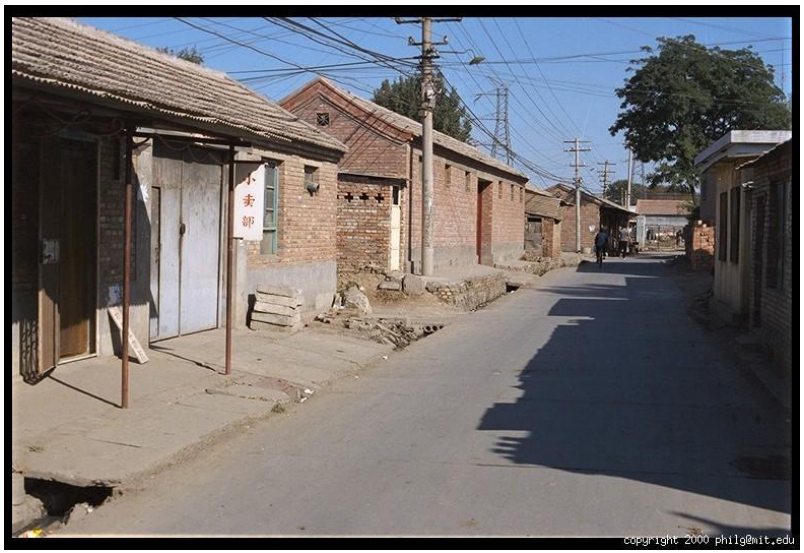
What's to Be Covered in Part II Today…

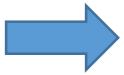
- Image Representation
  - Color Space
  - Image Filtering
  - Image Pyramid
- General Framework for Image Classification





Example Color Image

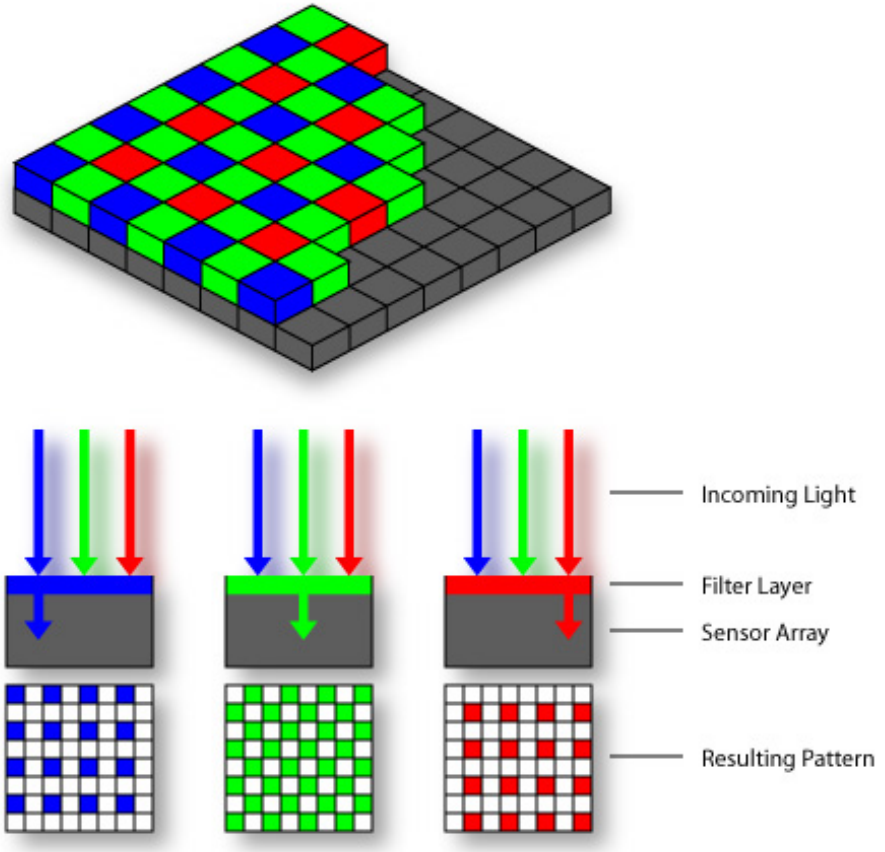




Color Sensing

- Bayer Grid/Pattern

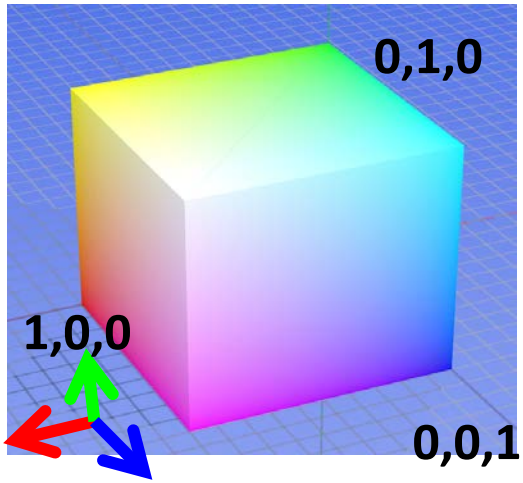
- Estimate RGB at ‘G’ cells from neighboring values



Slide credit: Steve Seitz

Example Color Space: RGB

- Default Color Space



R
(G=0,B=0)

G
(R=0,B=0)

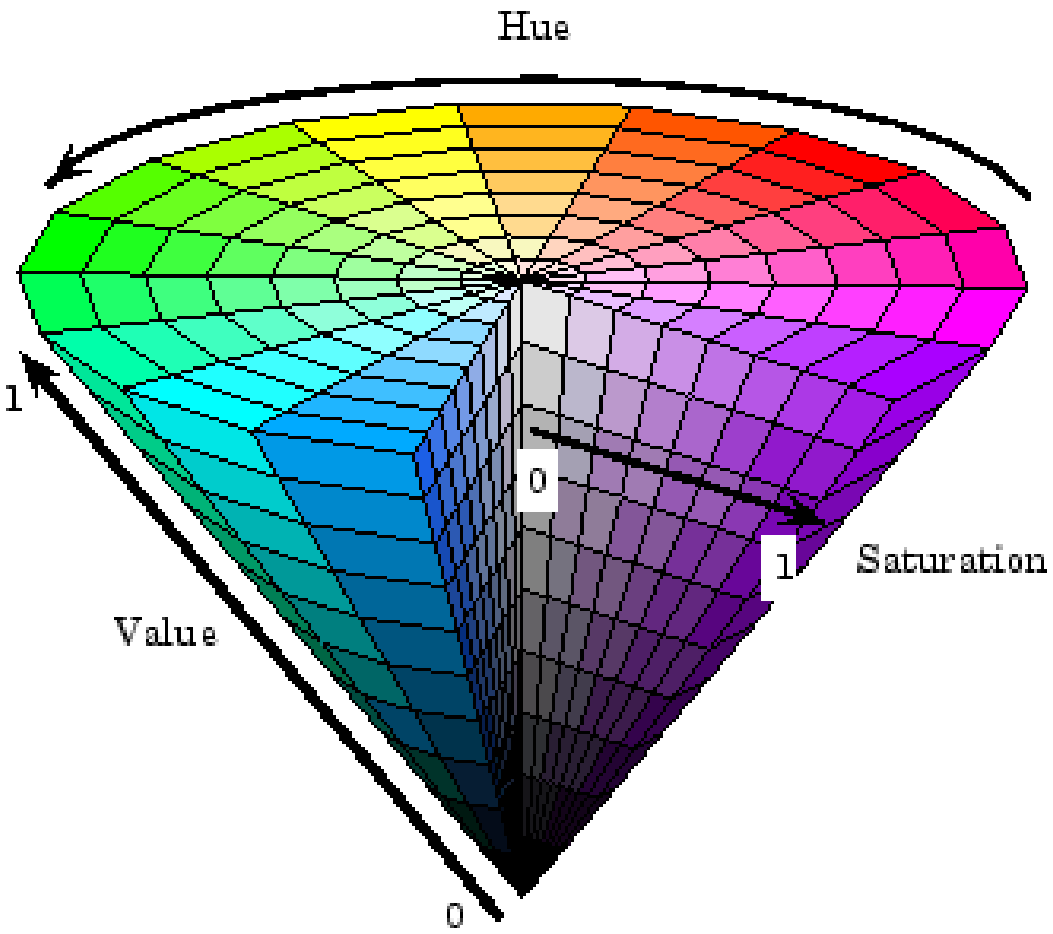
B
(R=0,G=0)

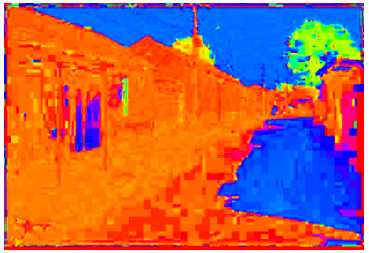
- Remarks
  - Easy for devices
  - But not perceptual
  - Where do the grays live?
  - Where is hue and saturation?

Image from: http://en.wikipedia.org/wiki/File:RGB_color_solid_cube.png

Example Color Space: HSV

- Intuitive Color Space



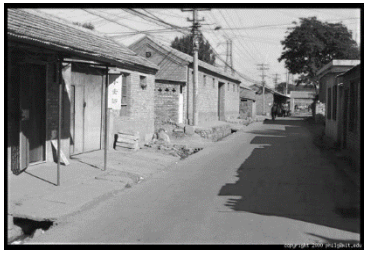


H

(S=1,V=1)

S

(H=1,V=1)

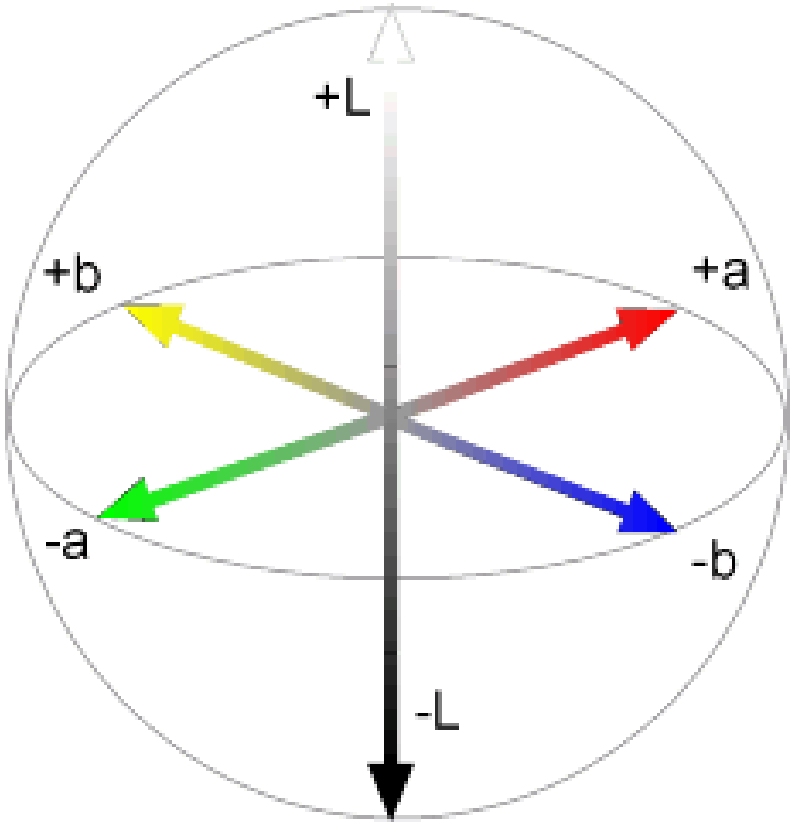


V

(H=1,S=0)

Example Color Space: L*a*b*

- “Perceptually uniform/linear” color space



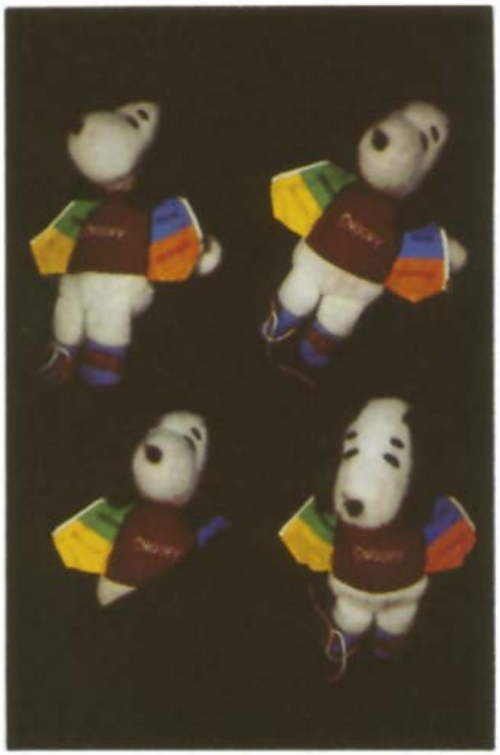
L
(a=0,b=0)

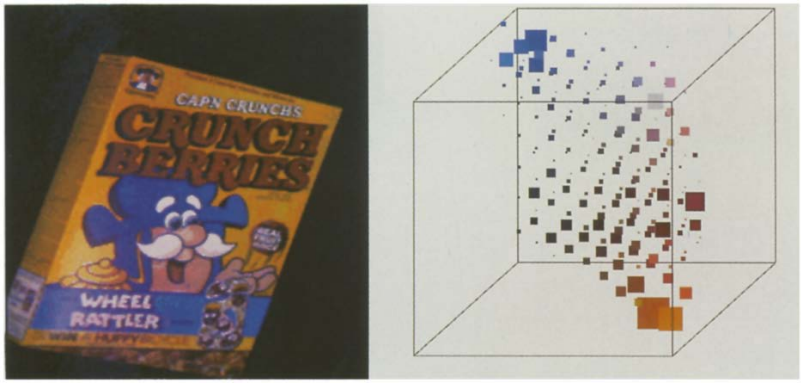
a
(L=65,b=0)

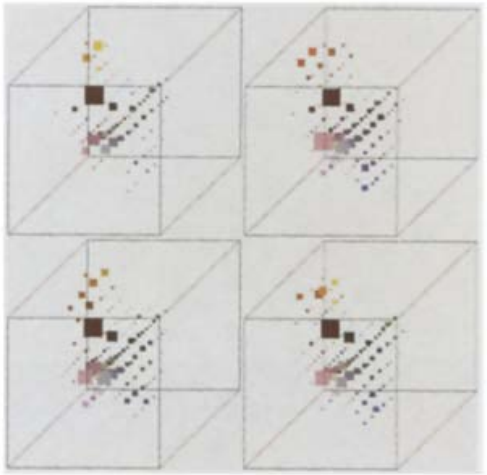
b
(L=65,a=0)

Uses of Color in Computer Vision

- Color histograms for indexing and retrieval



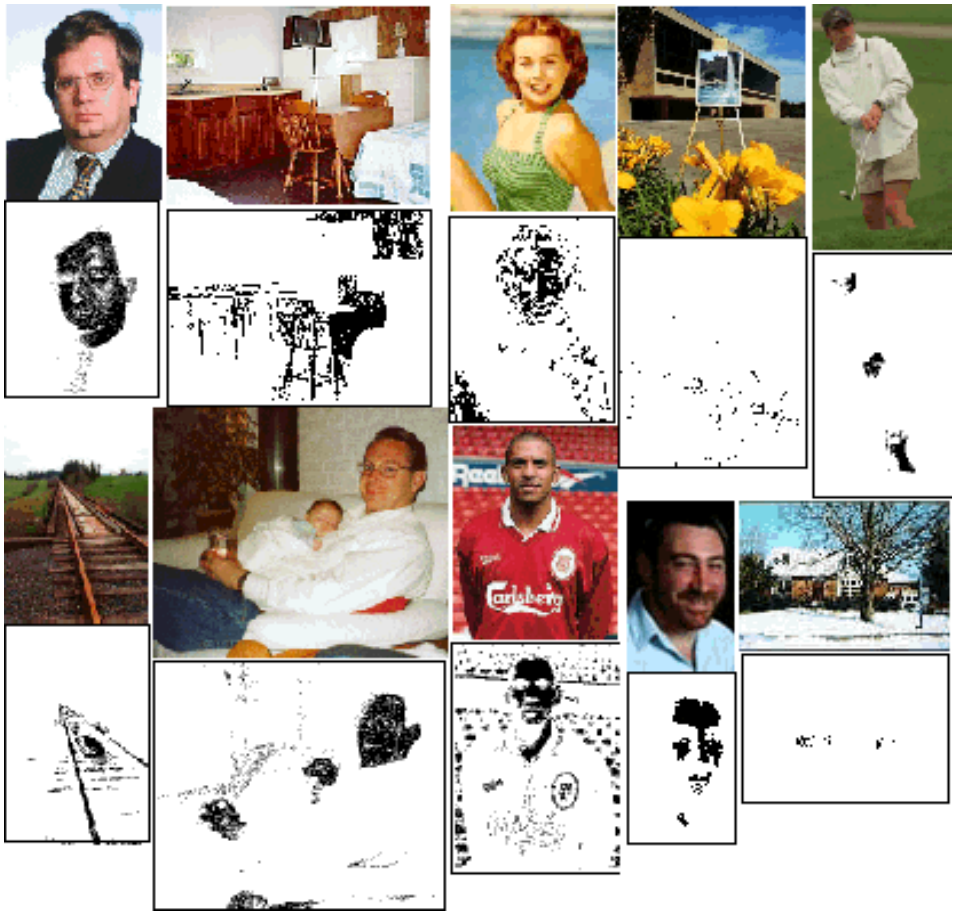




Swain and Ballard, Color Indexing, IJCV 1991

Uses of Color in Computer Vision

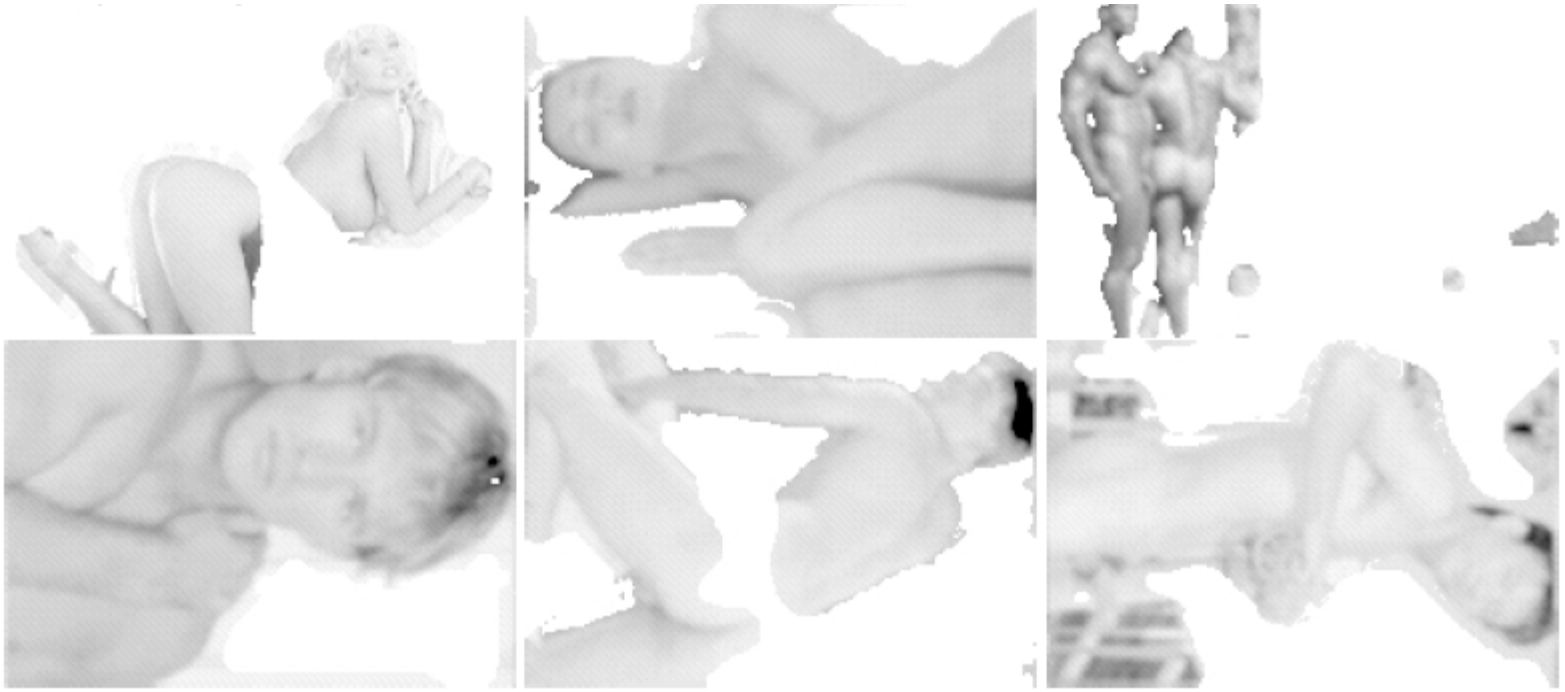
- Skin Detection



M. Jones and J. Rehg, Statistical Color Models with Application to Skin Detection, IJCV 2002.

Uses of Color in Computer Vision

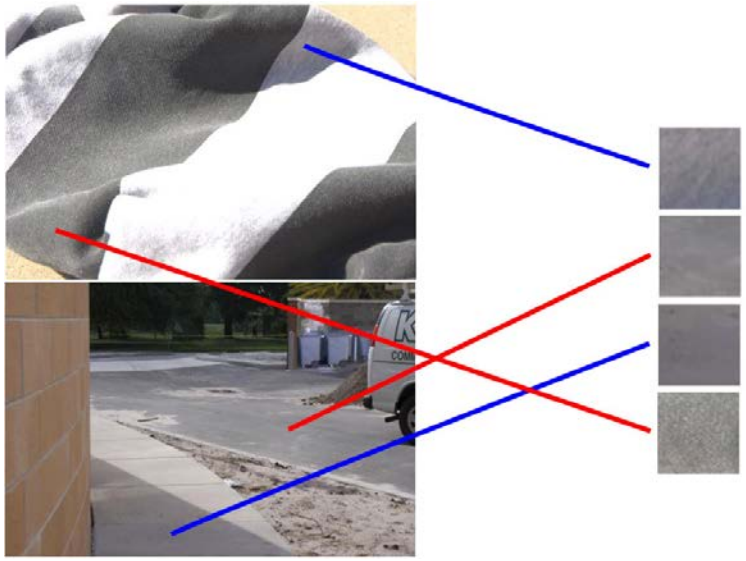
- Nudity detection



Forsyth and Fleck, ``Automatic Detection of Human Nudes,'' *IJCV*, 1999

Light

- A pixel's brightness is determined by
  - Light source (strength, direction, color)
  - Surface orientation
  - Surface material and albedo
  - Reflected light and shadows from surrounding surfaces
  - Gain on the sensor
- A pixel's brightness tell us nothing but itself.



Slide credit: J.-B. Huang

Light (cont'd)

- Yet, we are able to interpret images from lightness.
  - For nearby pixels/points, most factors do not change much.
  - The information is mainly contained in *local differences* of brightness

Slide credit: J.-B. Huang

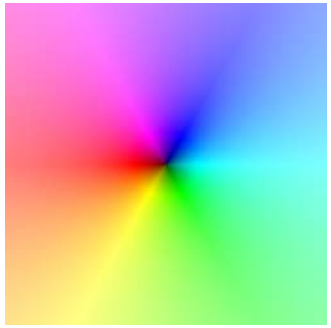
What differences in intensity tell us about shape?

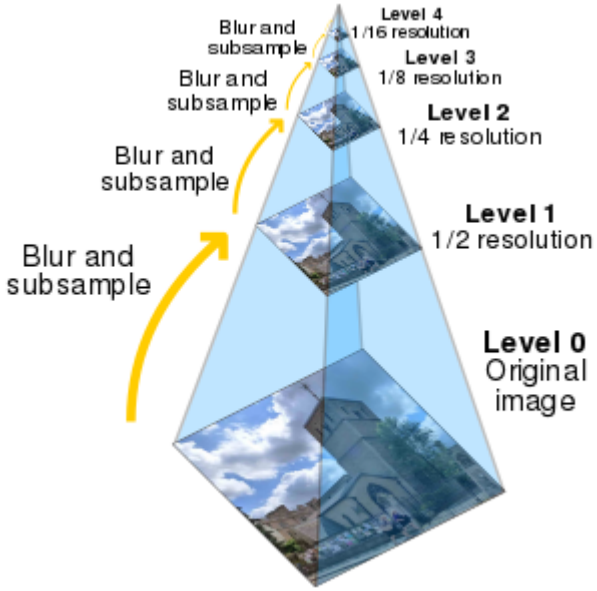
- Changes in surface normal
- Texture
- Proximity
- Indents and bumps
- Grooves and creases

Photos Koenderink slides on image texture and the flow of light

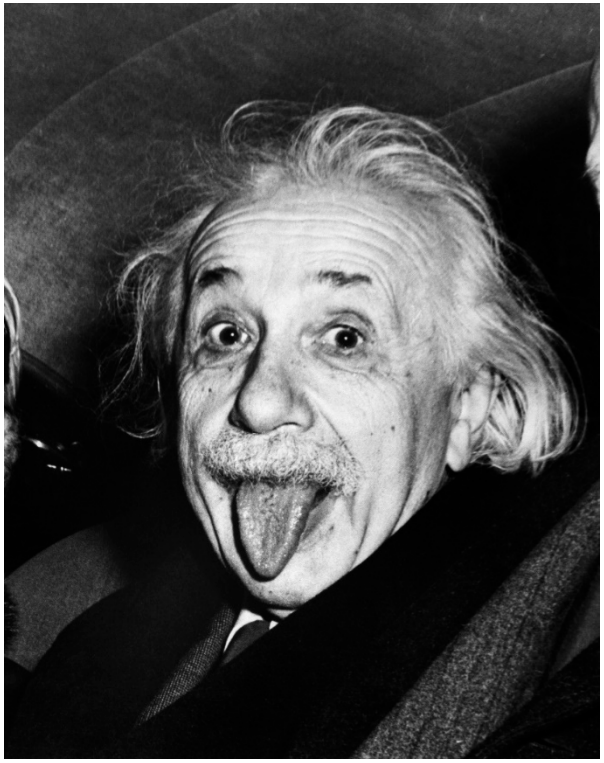
What's to Be Covered in Part II Today…

- Image Representation
  - Color Space
  - Image Filtering
  - Image Pyramid



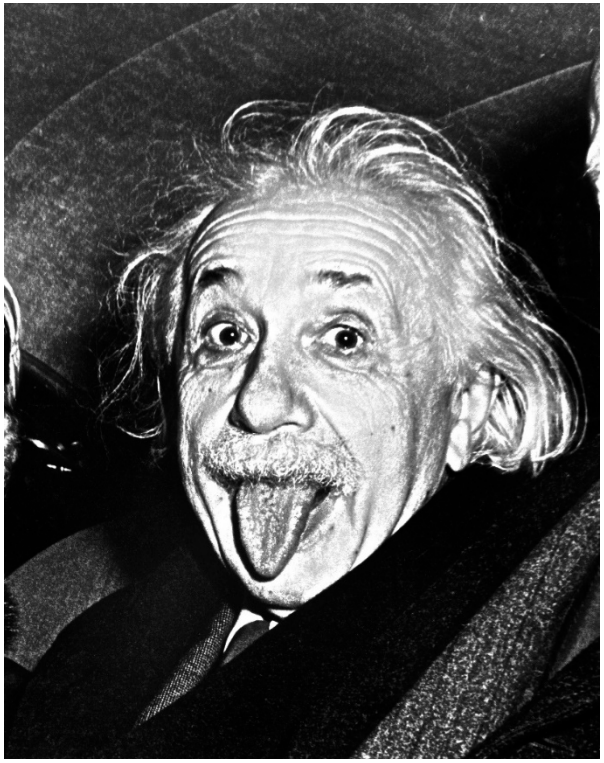


Why Image Filtering?



Input

Smoothing



Sharpening

https://en.wikipedia.org/wiki/Albert_Einstein_in_popular_culture#/media/File:Einstein_tongue.jpg

Why Image Filtering?

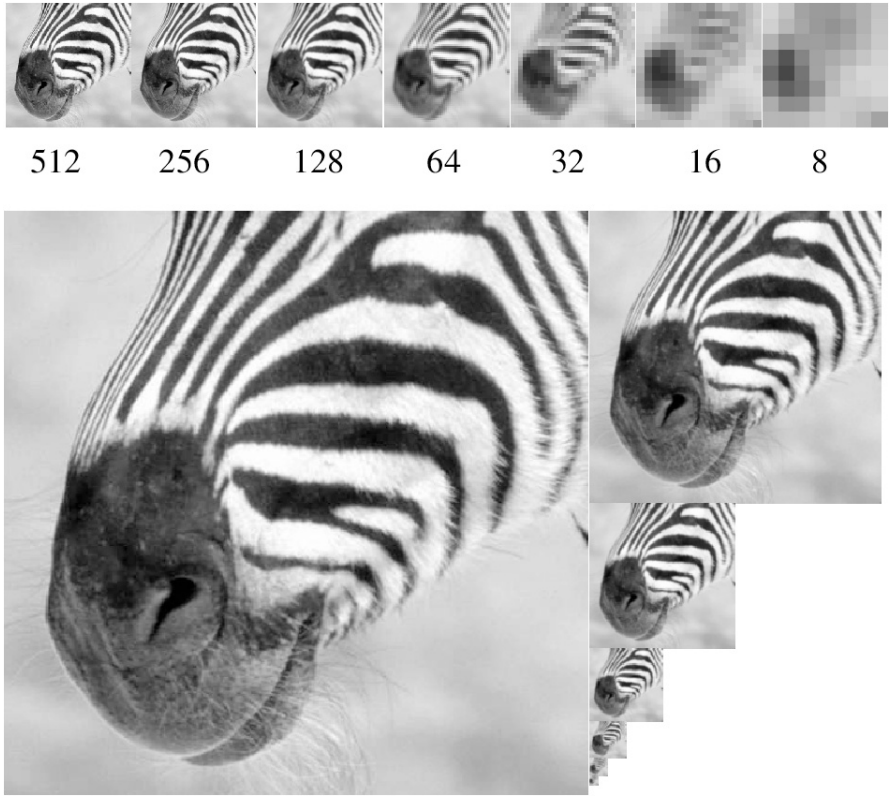


Image Pyramid

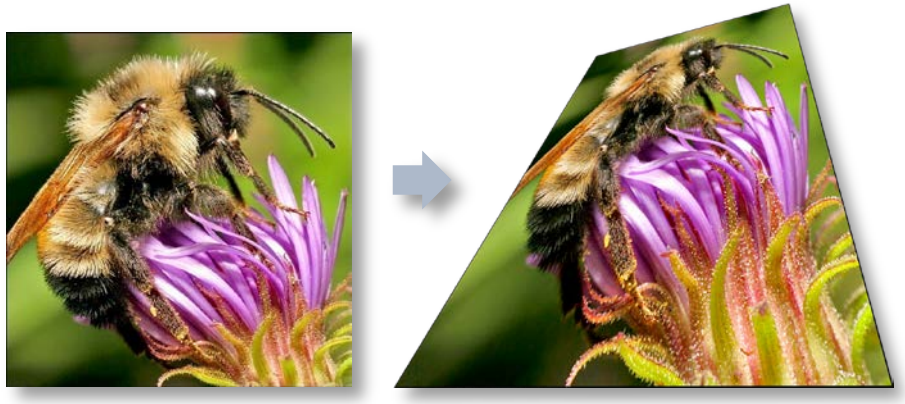
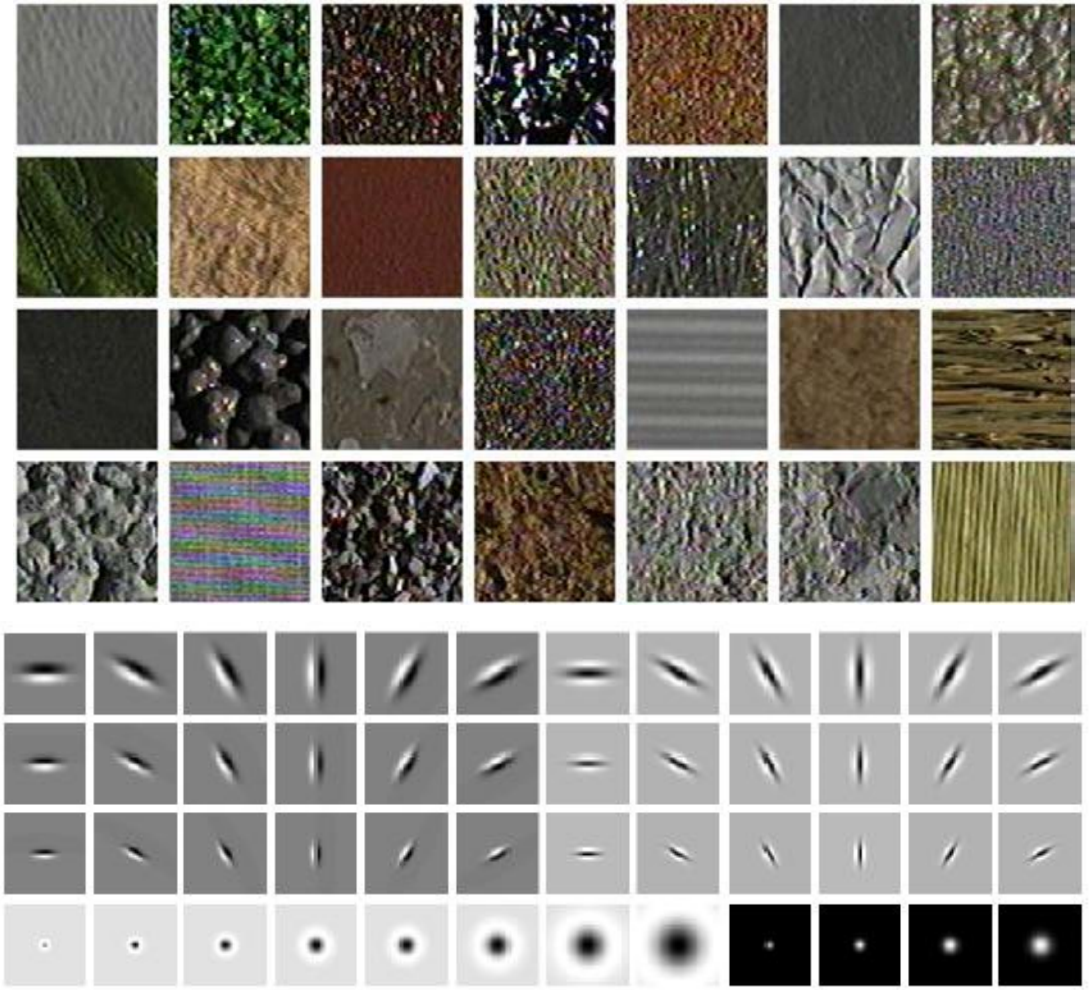


Image interpolation/resampling

Image credit: D. Forsyth & N. Snavely

Why Image Filtering?

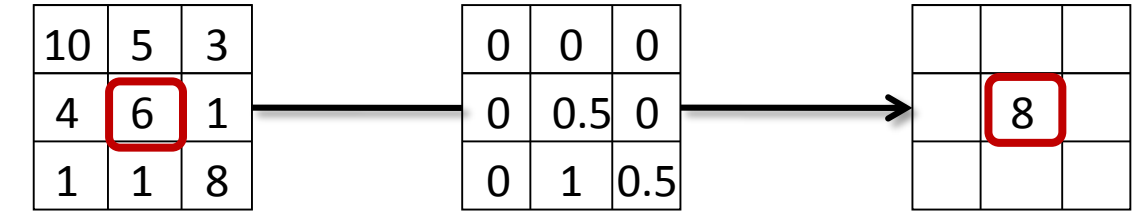
Representing textures with filter banks



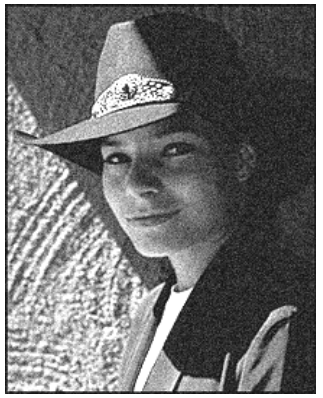
LM filter bank. Code here

What's Image Filtering?

- Image Filtering
  - For each pixel of interest, compute the function of local neighborhood and output the new value.
  - Generally (while not always true), same function applied to each position
- Linear filtering
  - function is a weighted sum/difference of pixel values



- Example applications
  - Enhance images
    - Denoise, smooth, increase contrast, etc.
  - Extract information from images
    - Texture, edges, distinctive points, etc.
  - Detect patterns



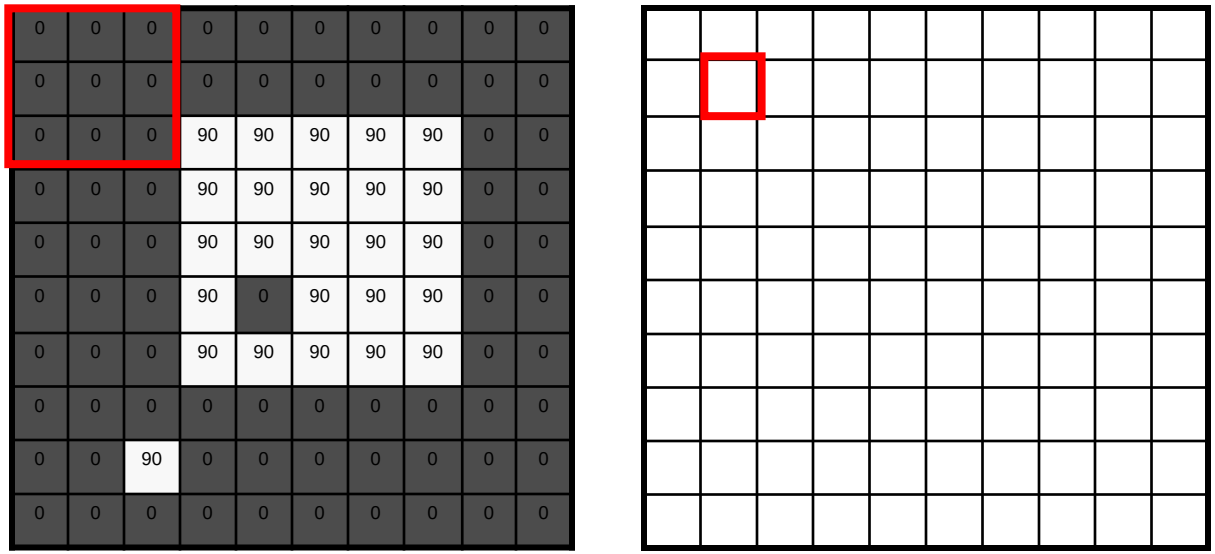
Slide credit: D. Hoiem

Example: Image Filtering with a Box Filter

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$ $h[.,.]$



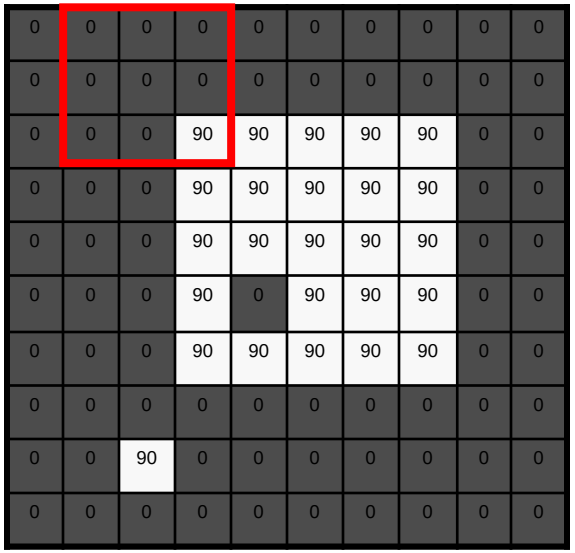
$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Slide credit: S. Seitz

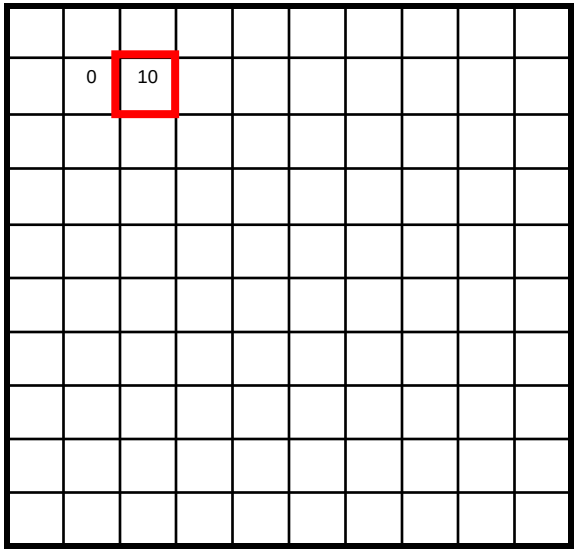
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



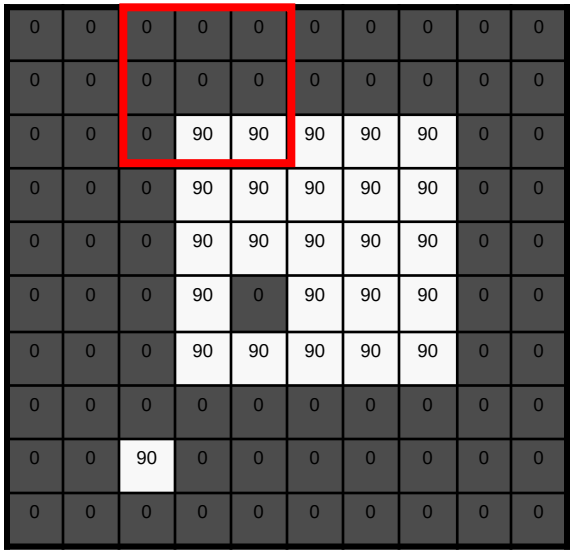
$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Slide credit: S. Seitz

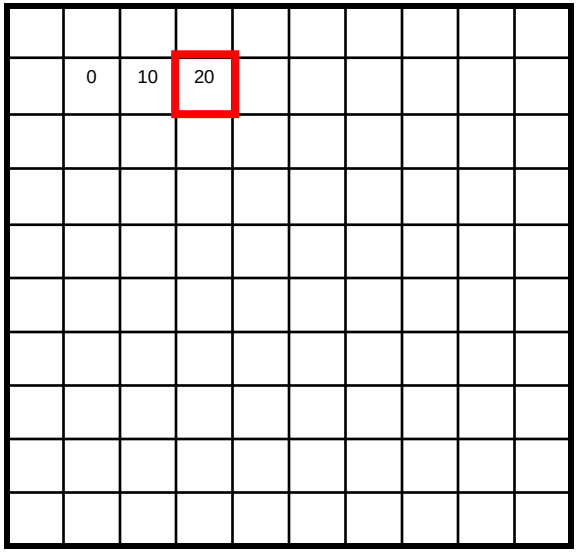
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Slide credit: S. Seitz

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30					

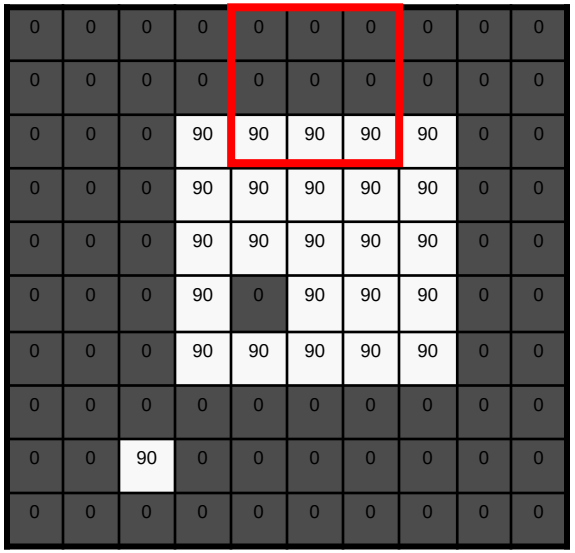
$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Slide credit: S. Seitz

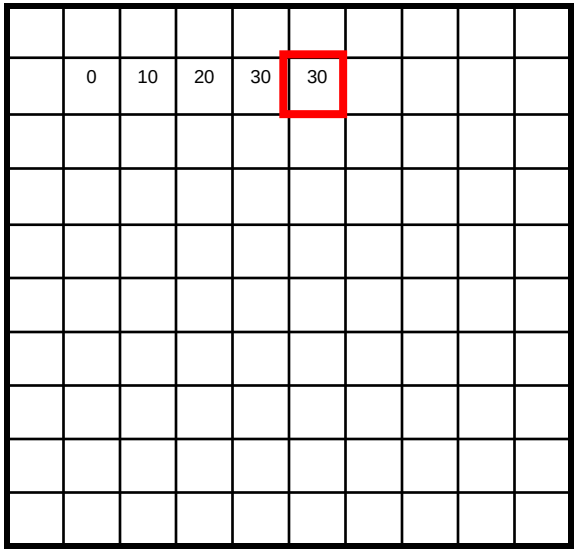
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



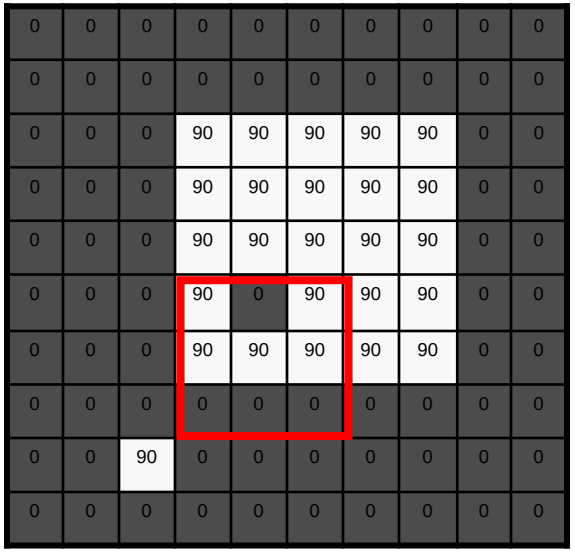
$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Slide credit: S. Seitz

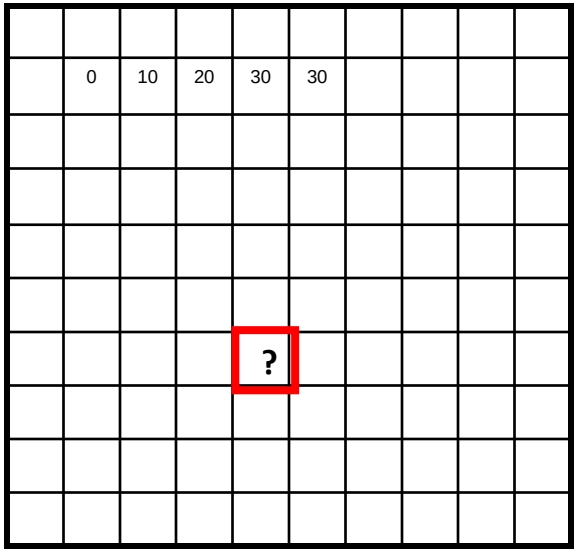
$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$



$h[.,.]$



$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Slide credit: S. Seitz

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30	30				
						?			
				50					

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Slide credit: S. Seitz

$g[\cdot,\cdot]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$h[m,n] = \sum_{k,l} g[k,l]\ f[m+k,n+l]$$

Slide credit: S. Seitz

Example: Image Filtering with a Box Filter

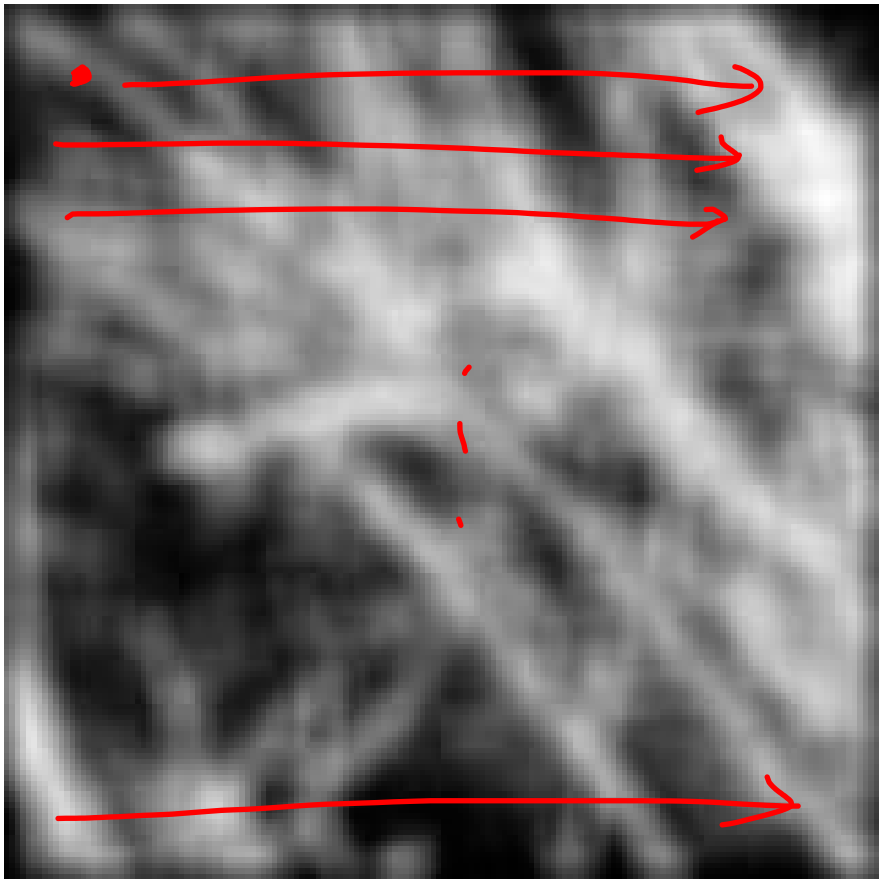
- What does Box Filter do?
  - Replaces each pixel with an average of its neighborhood
  - Performs image smoothing
  - Achieves image denoising? Not necessarily…Why?

$$g[\cdot,\cdot]$$

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Slide credit: K. Grauman

Smoothing with a Box Filter



Correlation Filtering

For the averaging window with size 2k+1 x 2k+1:

$$G[i,j] = \underbrace{\frac{1}{(2k+1)^2}}_{\text{Attribute uniform weight to each pixel}} \underbrace{\sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]}$$

Now generalize to allow different weights depending on neighboring pixel's relative position:

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

Slide credit: K. Grauman

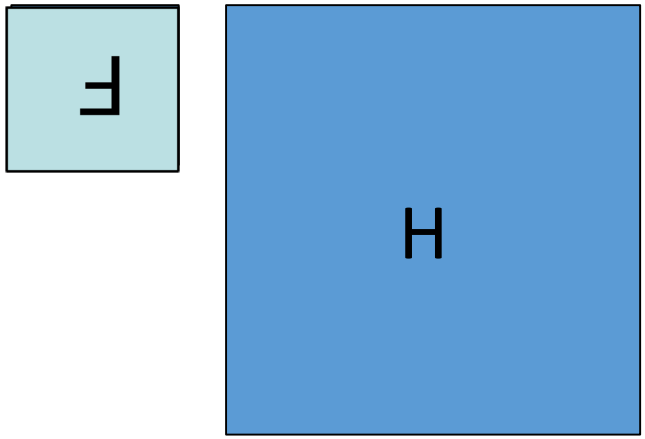
Convolution

- Convolution:
  - Flip the filter in both dimensions (bottom to top, right to left)
  - Then apply cross-correlation

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

$$G = H \star F$$

Notation for convolution operator



Slide credit: K. Grauman

Convolution vs. Correlation

Convolution

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

```
G=conv2(H,F);
```

$$G = H \star F$$

Cross-correlation

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

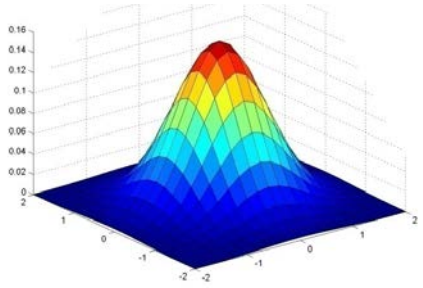
```
G=filter2(H,F);
```
or
```
G=imfilter(F,H);
```

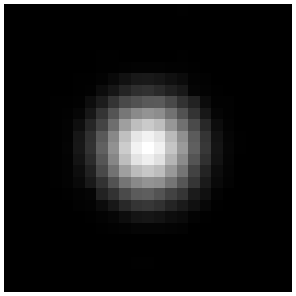
$$G = H \otimes F$$

Slide credit: K. Grauman

Gaussian Filter

- Spatially-weighted averaging





0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

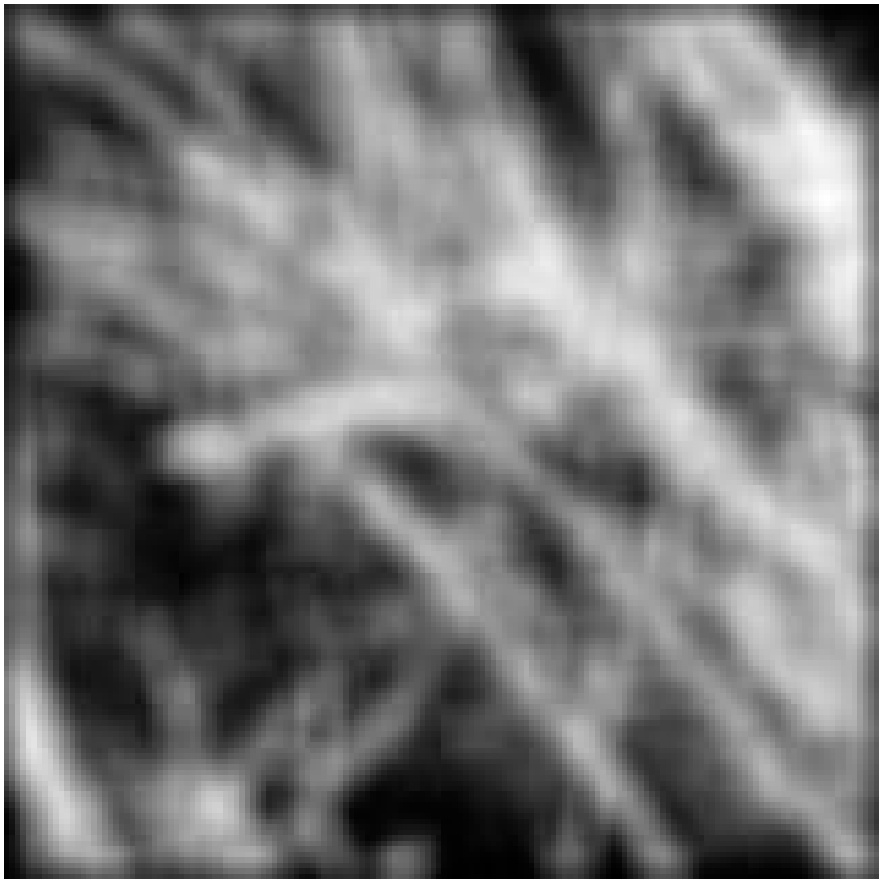
5 x 5, $\sigma = 1$

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

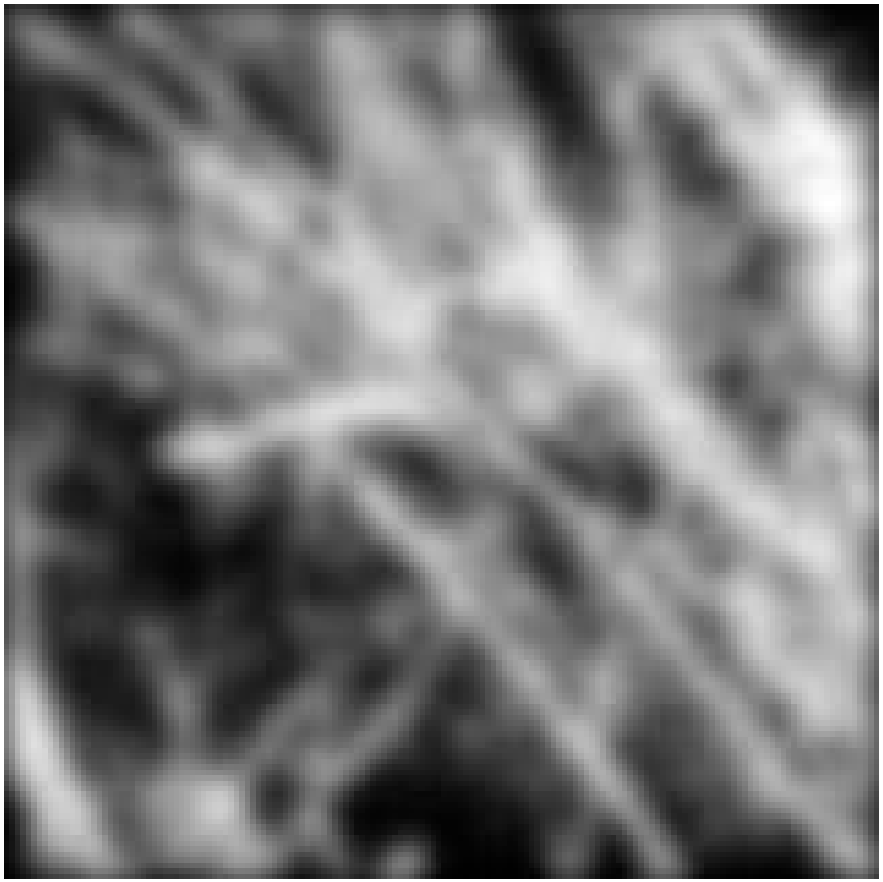
Slide credit: Christopher Rasmussen

Smoothing with a Box Filter

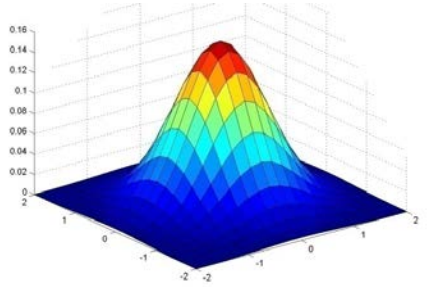
- Grid-like



Smoothing with a Gaussian Filter



Gaussian Filter

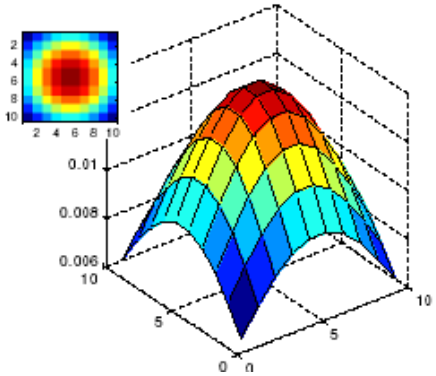


- Remove "high-frequency" components from the image (i.e., low-pass filter)
  - Images become more smooth
- Convolution with self is another Gaussian
  - So can smooth with small-width kernel, repeat, and get same result as larger-width kernel would have
  - Convolving two times with Gaussian kernel of width σ is same as convolving once with kernel of width $\sigma\sqrt{2}$
- *Separable* kernels
  - Factors into product of two 1D Gaussians

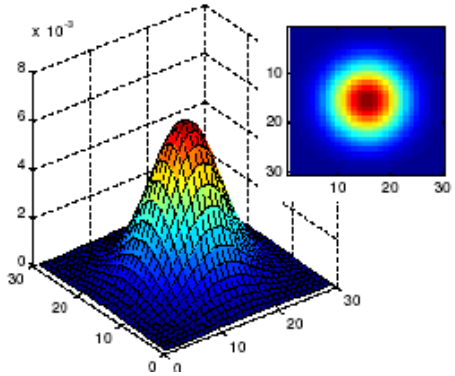
Slide credit: K. Grauman

Gaussian Filter

- What parameters matter here?
- **Size** of filter (or kernel/mask)



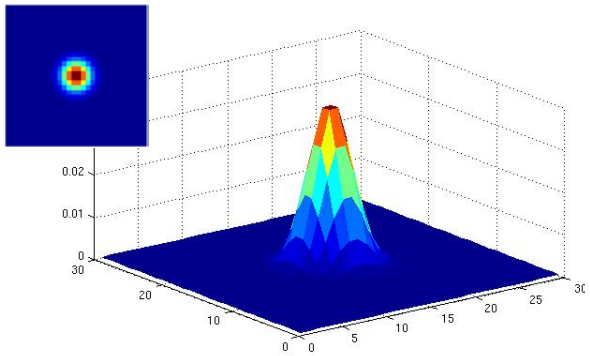
σ = 5 with 10 x 10 kernel



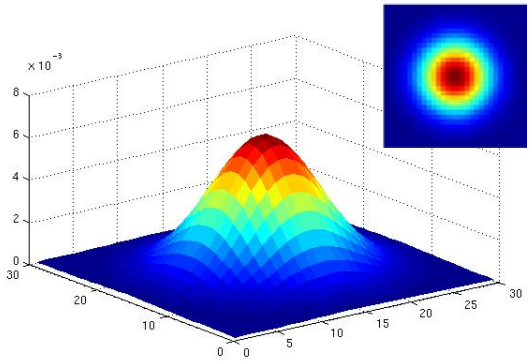
σ = 5 with 30 x 30 kernel

Gaussian Filter

- What parameters matter here?
- **Variance** of Gaussian: determines extent of smoothing



σ = 2
with 30 x 30 kernel



σ = 5
with 30 x 30 kernel

Separability of the Gaussian Filter

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$= \underbrace{\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)}_{m} \underbrace{\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)}_{m}$$

The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

In this case, the two functions are the (identical) 1D Gaussian

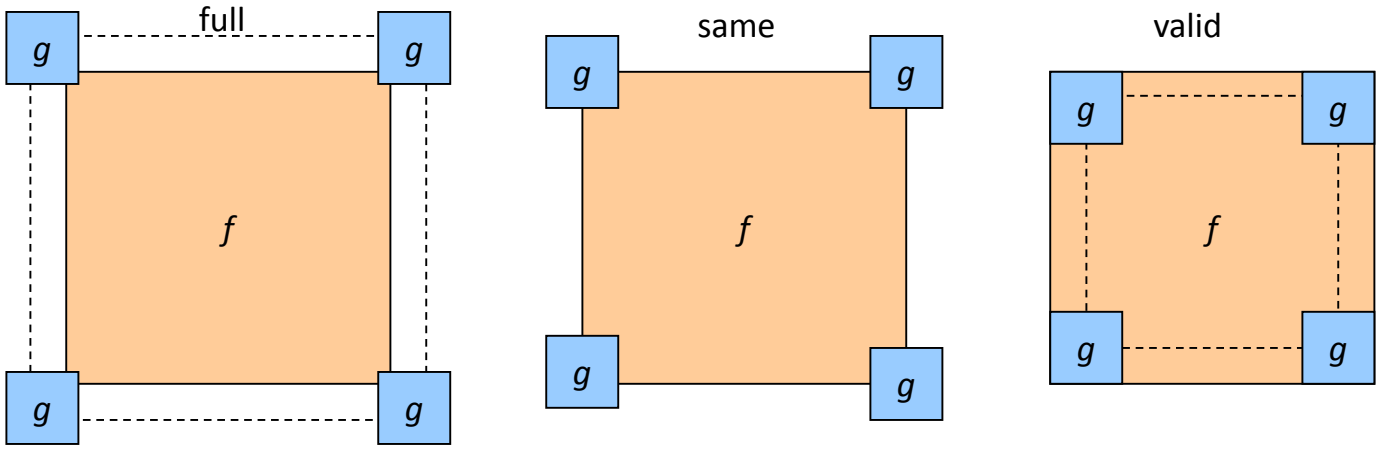
Source: D. Lowe

Separability of the Gaussian Filter

- Why is separability useful in practice?
- Separability means that a 2D convolution can be reduced to two 1D convolutions (one among rows and one among columns)
- What is the complexity of filtering an n×n image with an m×m kernel?
  - $O(n^2m^2)$
- What if the kernel is separable?
  - $O(n^2m) = O(n^2m) + O(n^2m)$

Practical matters

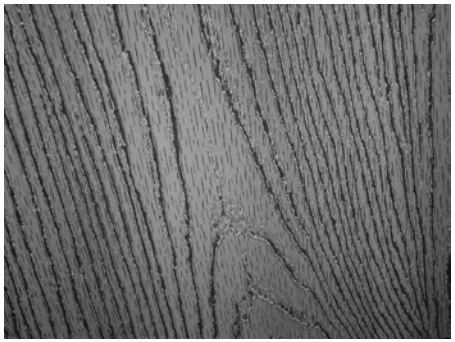
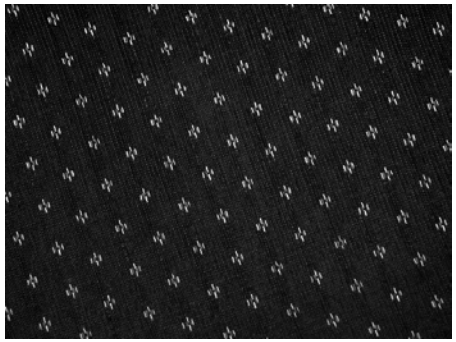
- What is the size of the output?
- MATLAB: filter2(g, f, *shape*)
  - *shape* = ‘full’: output size is sum of sizes of f and g
  - *shape* = ‘same’: output size is same as f
  - *shape* = ‘valid’: output size is difference of sizes of f and g



Source: S. Lazebnik

Applications: Representing Image Texture

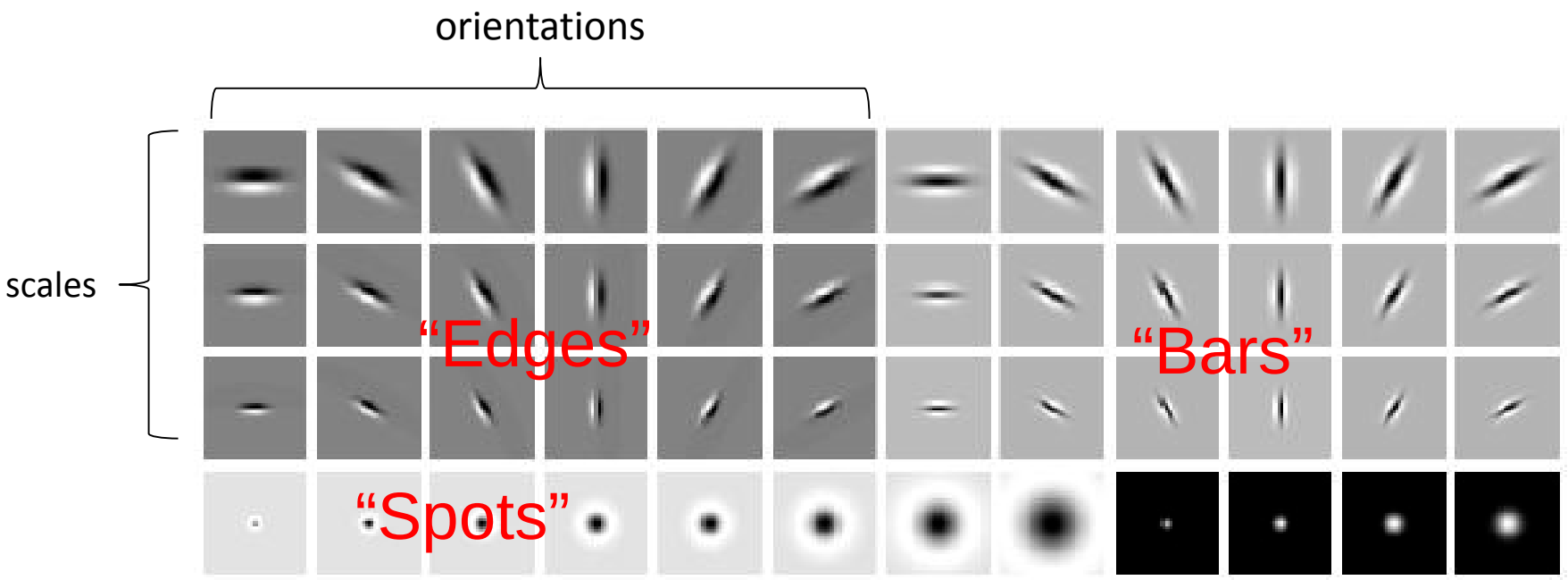
- Texture: Orientation, Material, and Scale


http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

How to Describe Textures?

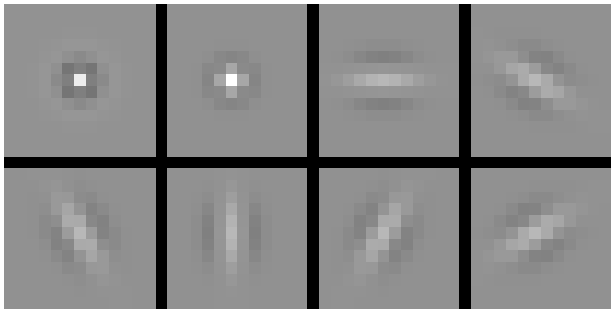
- Filter banks
  - Typically overcomplete representation
  - E.g., wavelet or more complex filters (basis functions)

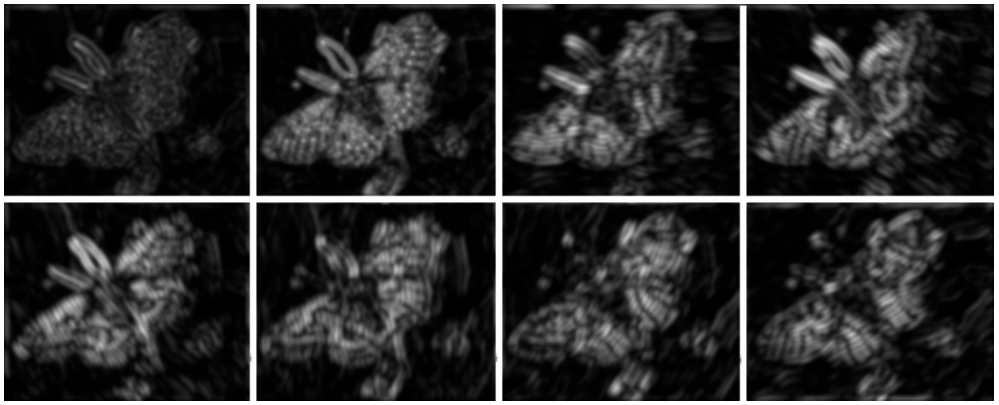


Code for filter banks: www.robots.ox.ac.uk/~vgg/research/texclass/filters.html

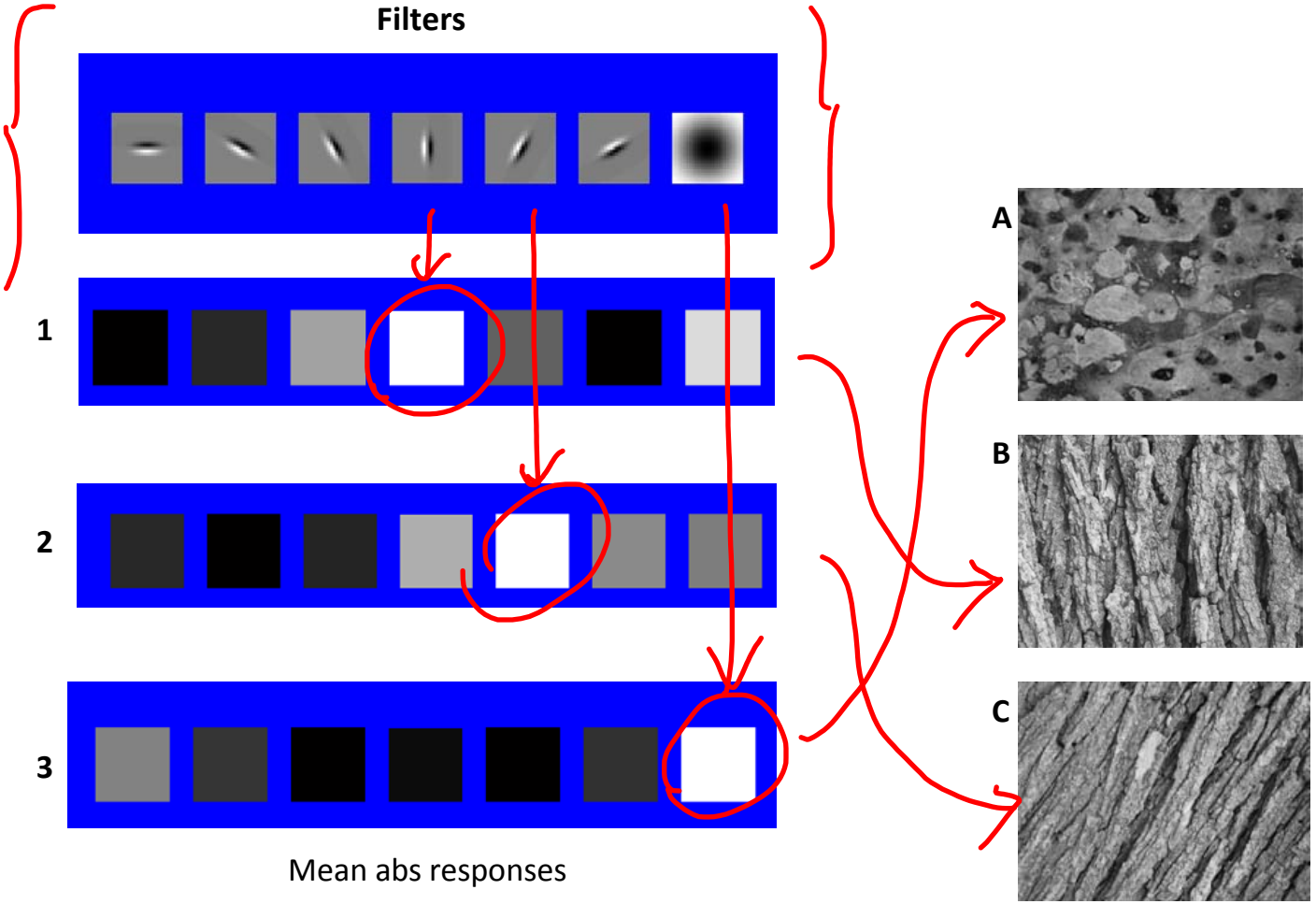
How to Describe Textures?

- Filter banks
  - Typically overcomplete representation
  - E.g., wavelet or more complex filters (basis functions)
  - Example



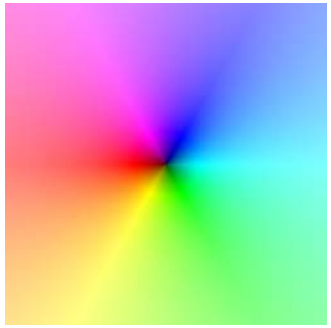


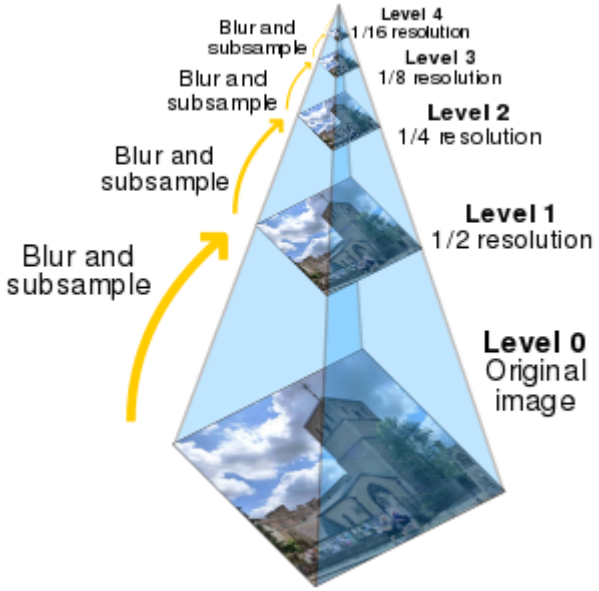
Can you match the filtering response?



What's to Be Covered in Part II Today…

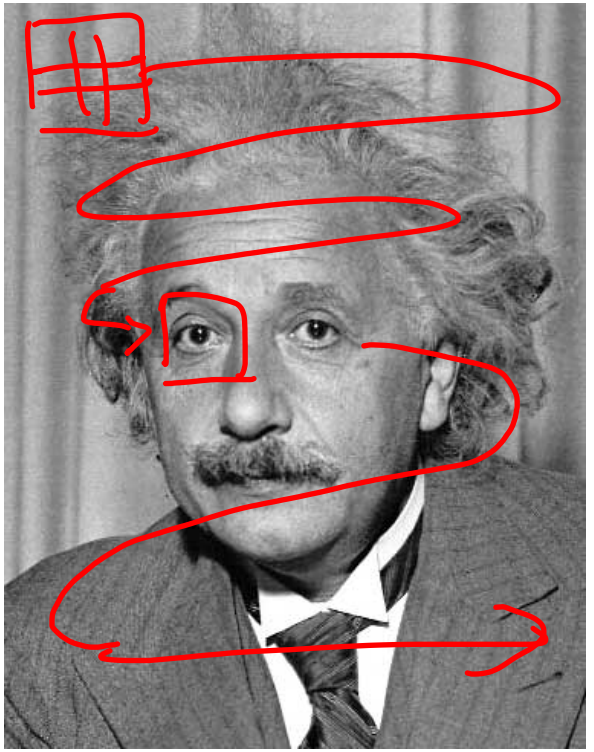
- Image Representation
  - Color Space
  - Image Filtering
  - Image Pyramid





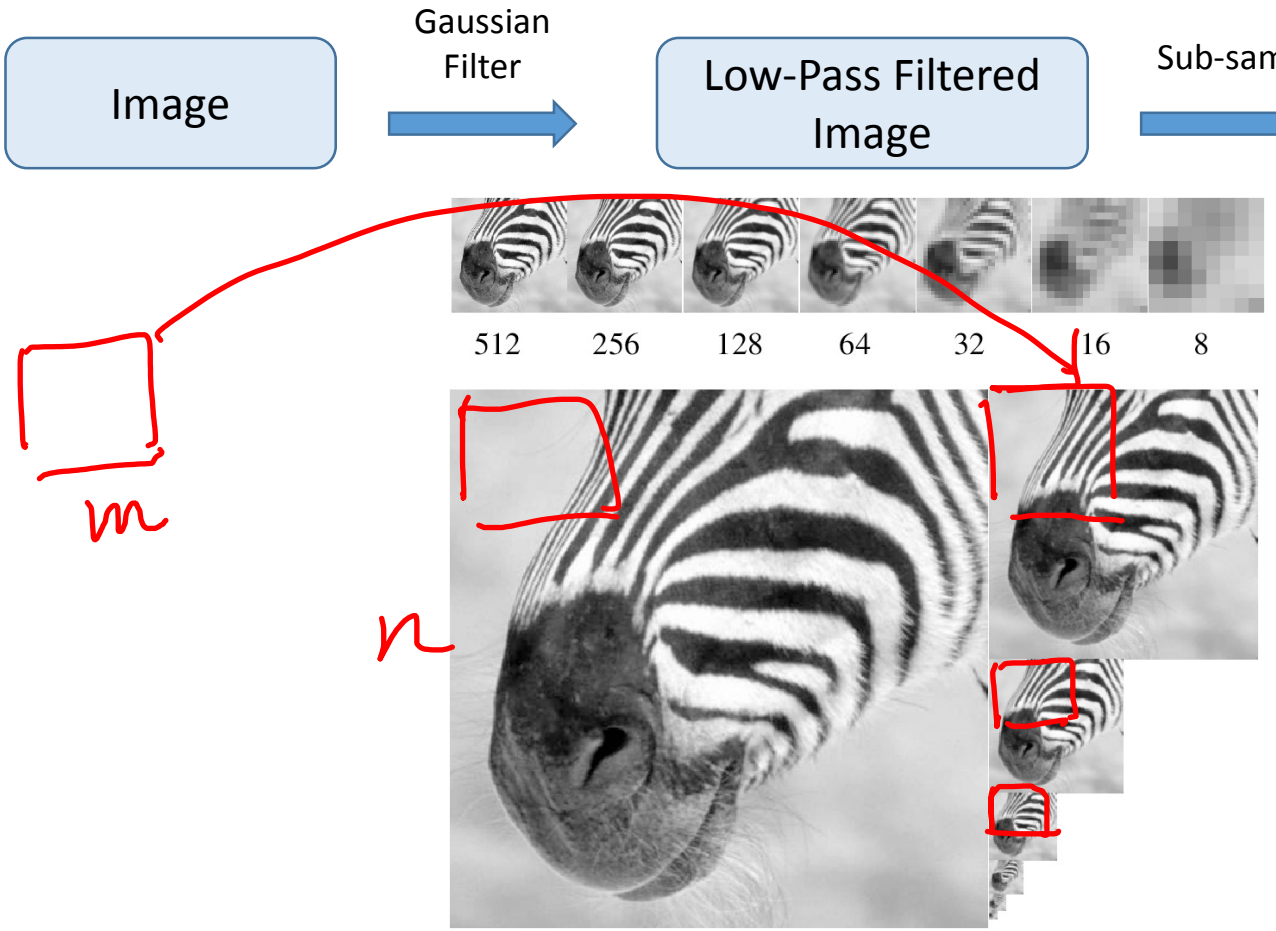
From Template Matching to Image Pyramid

- Goal: find  in an image
- Main challenge: what is a good similarity or distance metric for measuring between two patches? D( , )
  - Cross-Correlation?
  - Normalized cross correlation?
  - Zero-mean correlation?
  - Sum of Square Difference (SSD)?

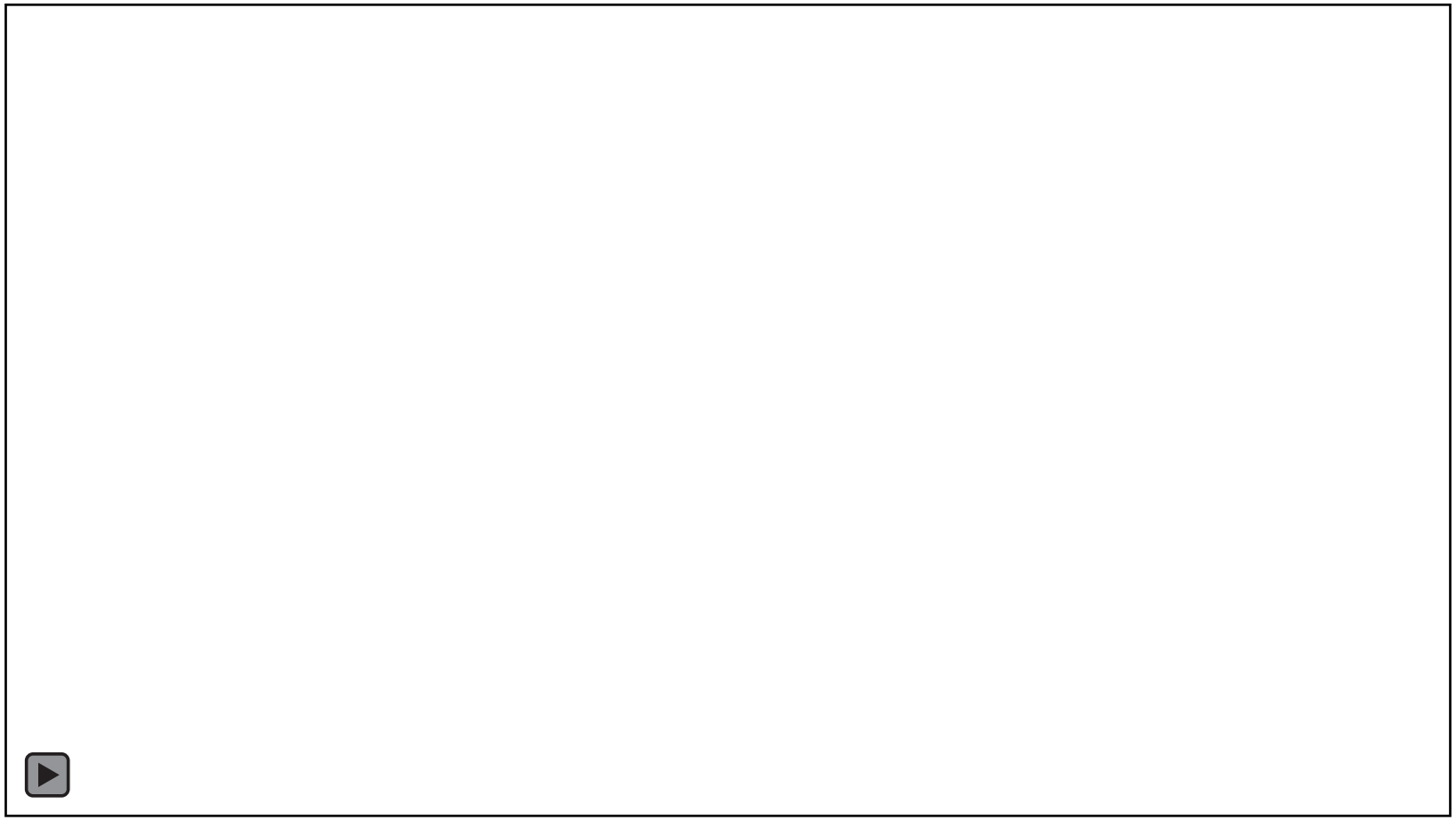


Size Matters…

- Question: What if we need to find *larger* or *smaller* eyes?
- Answer: Search in image pyramid! Why?
- Example: Gaussian pyramid (and why LPF?)

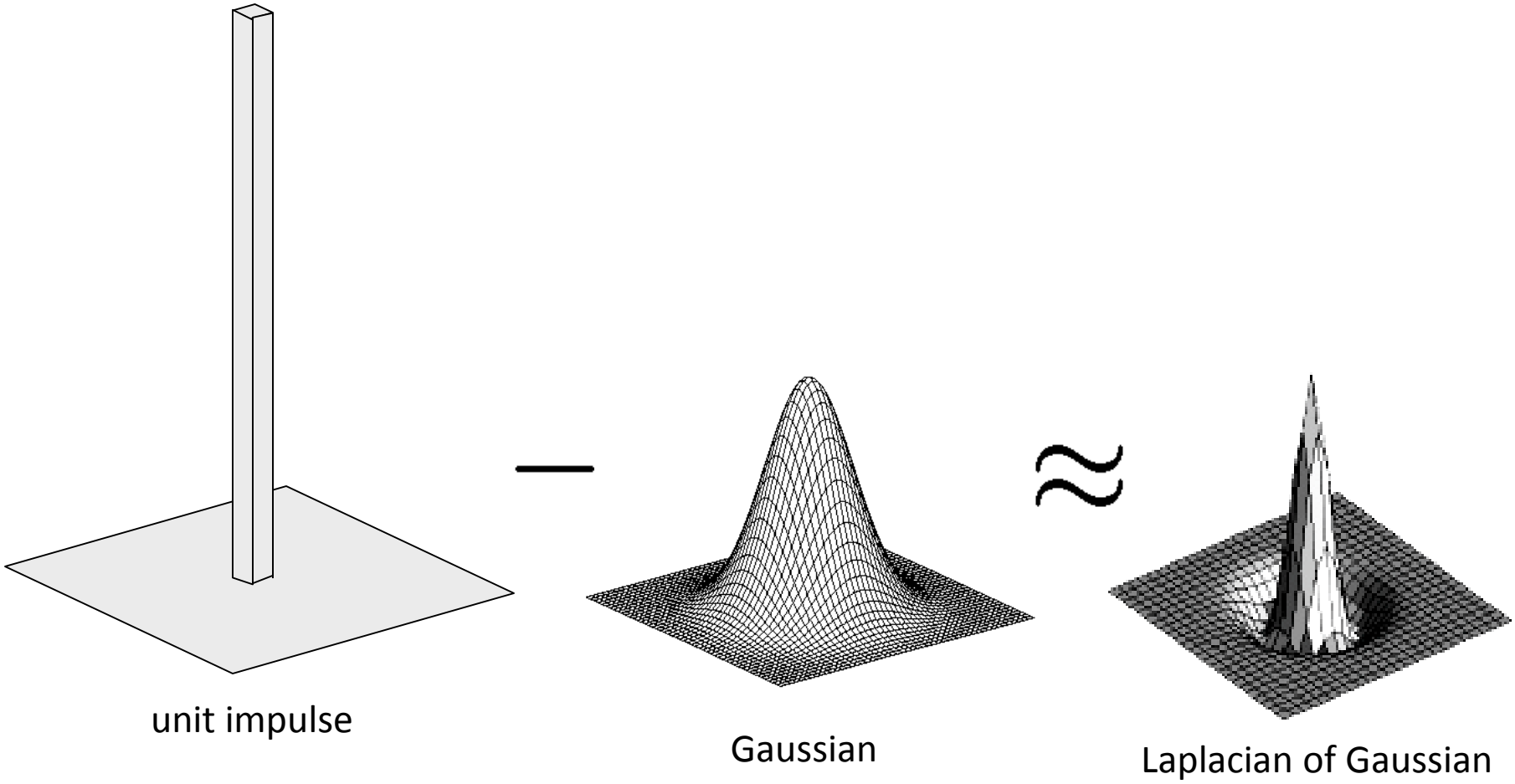


Wagon-Wheel Effect



https://www.youtube.com/watch?v=QOwzkND_ooU

Example: Laplacian filter



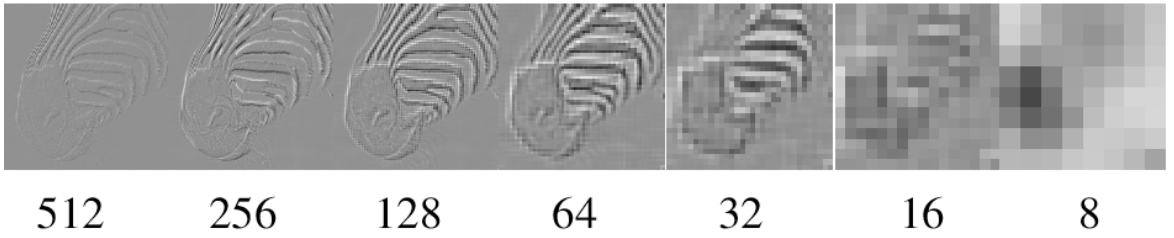
unit impulse

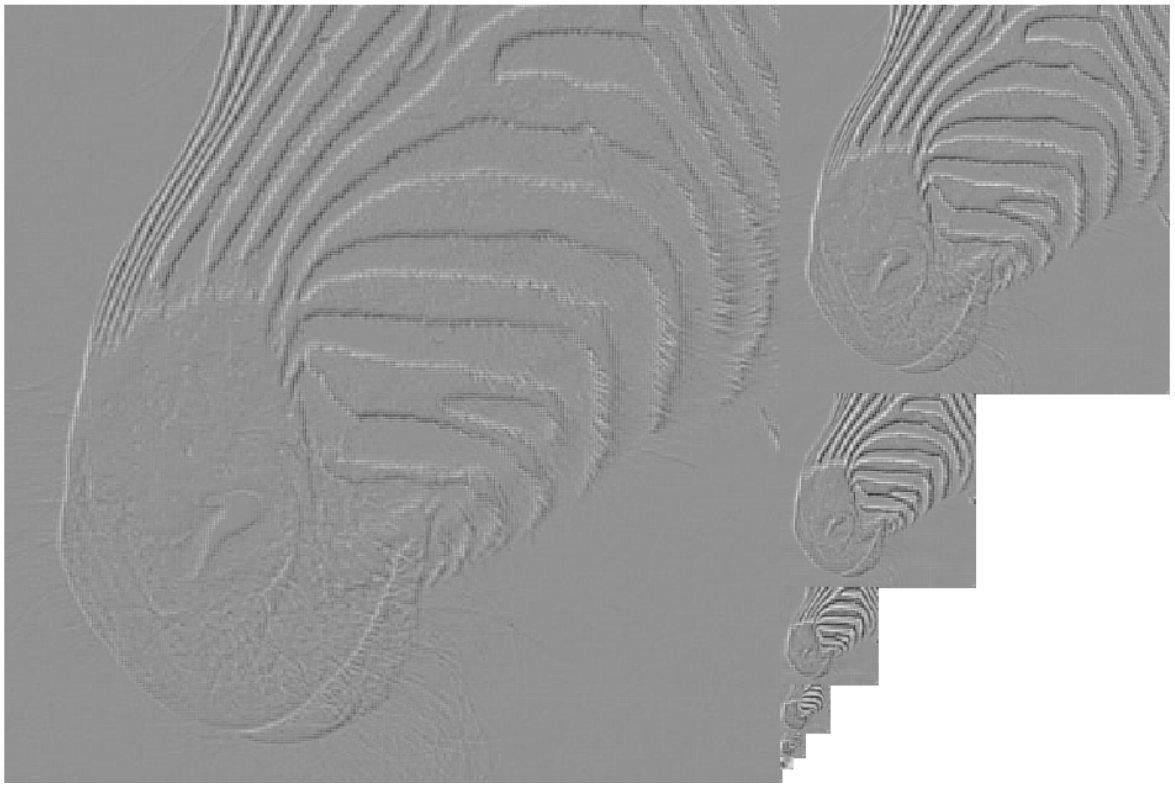
Gaussian

Laplacian of Gaussian

Source: Lazebnik

Laplacian Pyramid

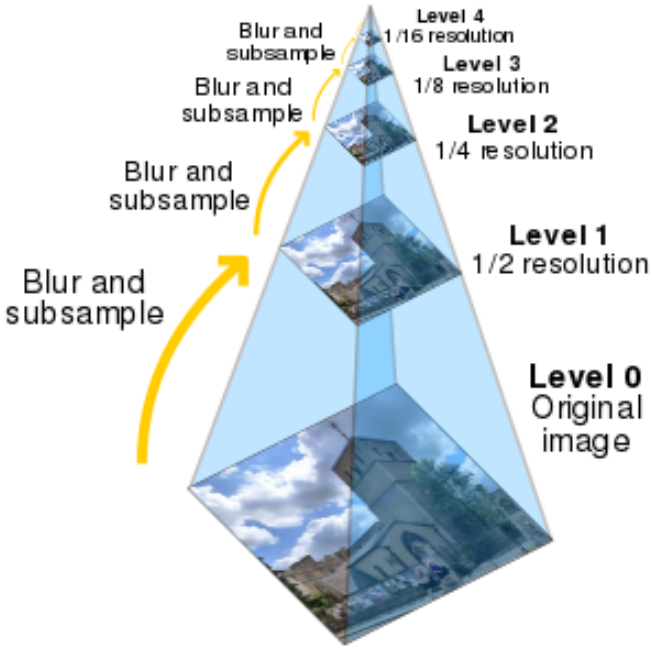




Source: Forsyth

Popular Uses of Image Pyramids

- Object detection
  - Scale search
  - Features
- Detecting stable interest points
- Course-to-fine registration
- Compression




STARBUCKS
COFFEE AND
STARBUCKS
COFFEE
STARBUCKS
COFFEE


STARBUCKS
COFFEE
STARBUCKS
COFFEE
STARBUCKS
COFFEE


STARBUCKS
COFFEE
STARBUCKS
COFFEE

Coarse-to-Fine Image Registration

1. Compute Gaussian pyramid
2. Align with coarse pyramid
3. Successively align with finer pyramids
   - Search smaller range

Why is this faster?

Are we guaranteed to get the same result?

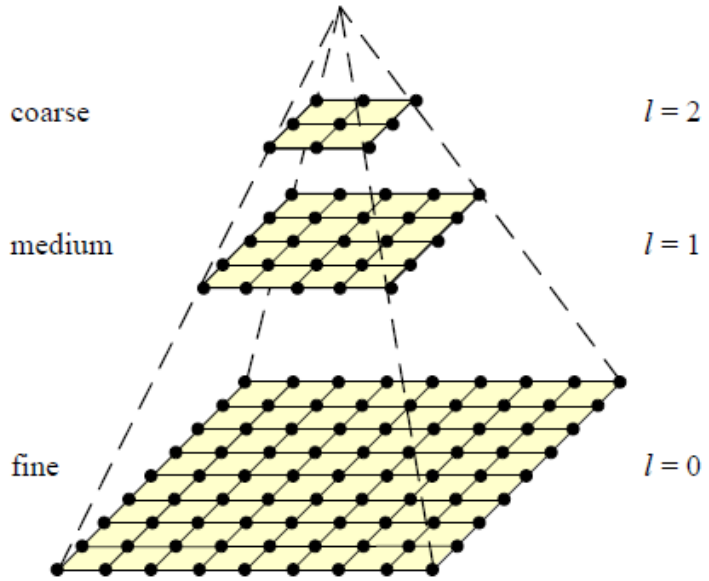
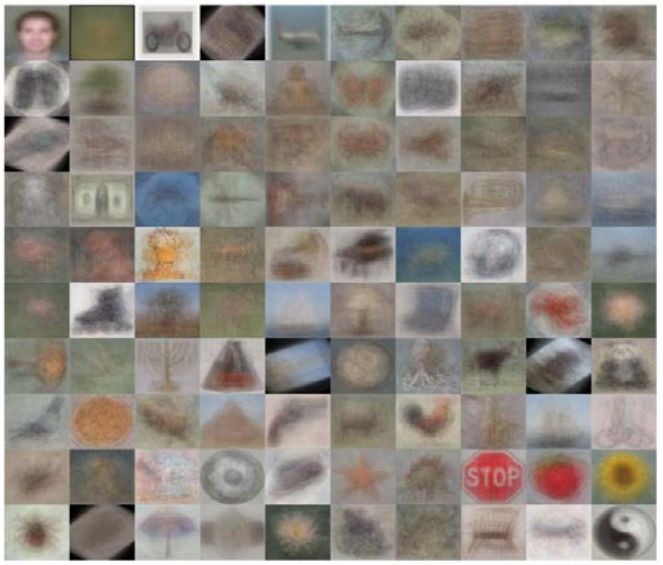


Image Categorization

- Object Recognition

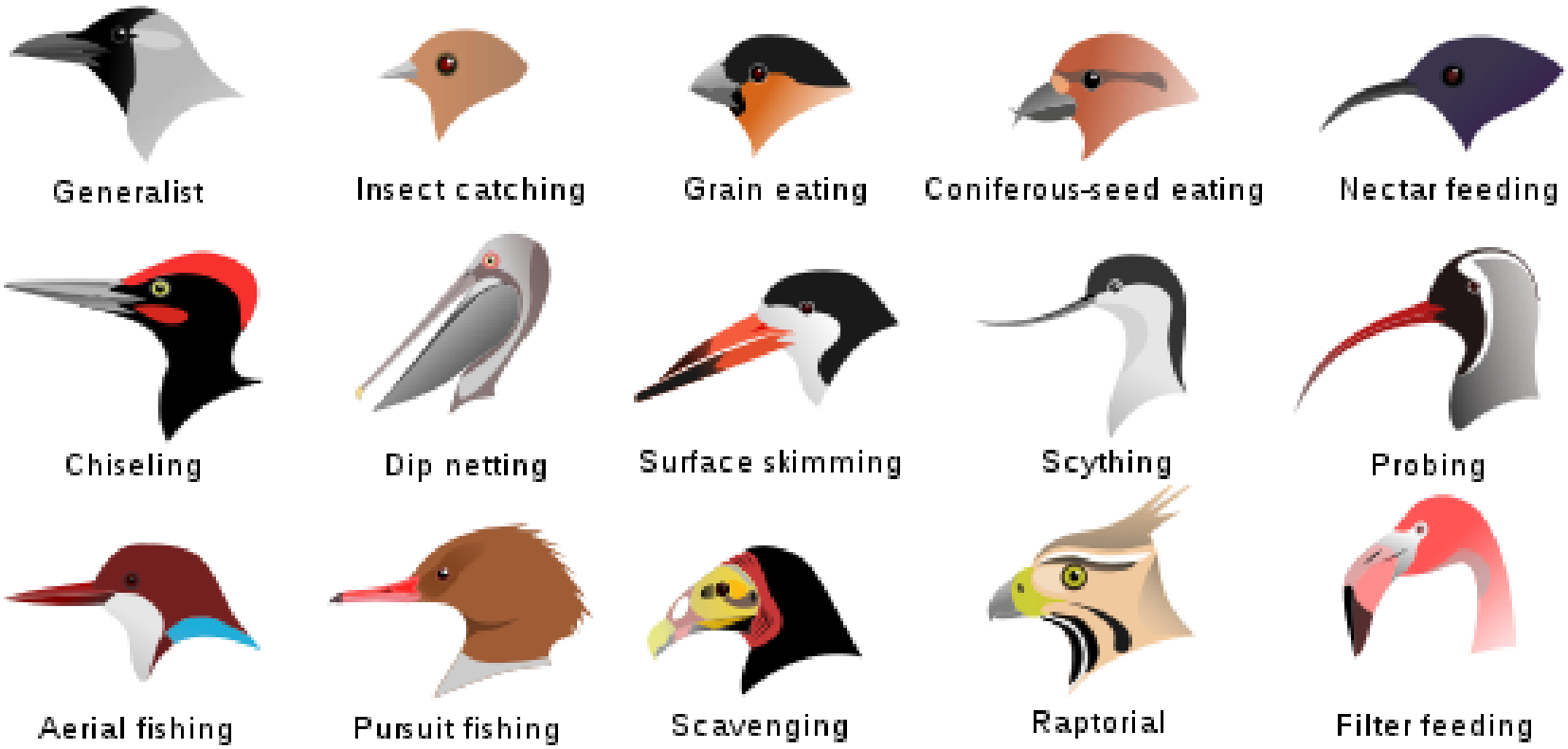
Example images of Caltech 101



Average object images of Caltech 101

Image Categorization

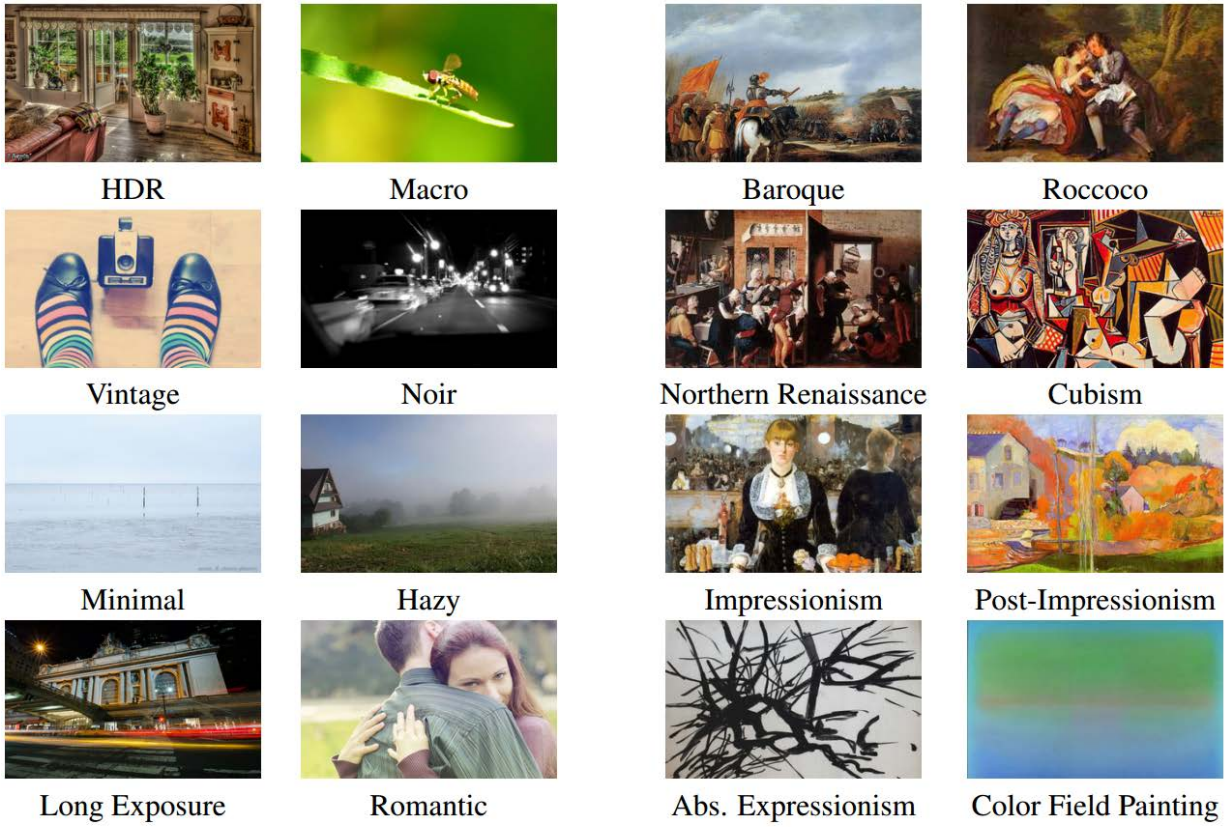
- Fine-grained image classification



Visipedia Project

Image Categorization

- Image style recognition



[Karayev et al. BMVC 2014]

Image Categorization

- Dating historical photos

1940 1953 1966 1977

[Palermo et al. ECCV 2012]

Supervised Learning for Visual Classification

- General framework

Image Data

+

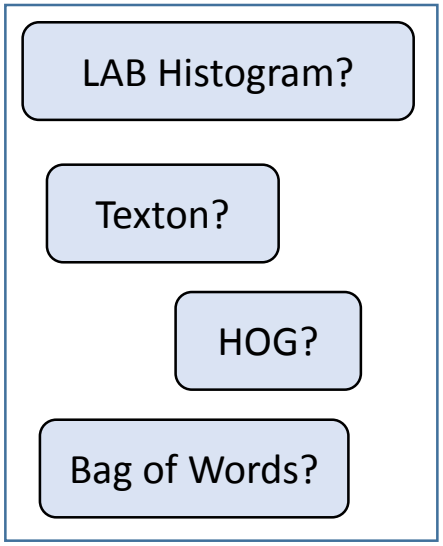
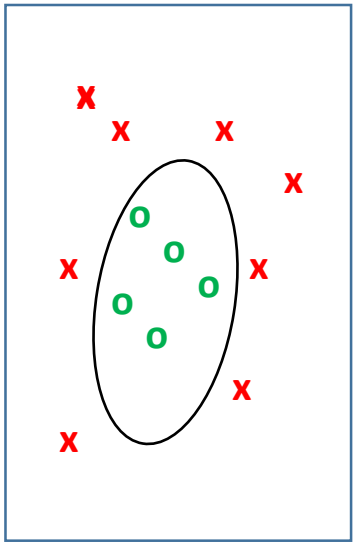


Image Features

+

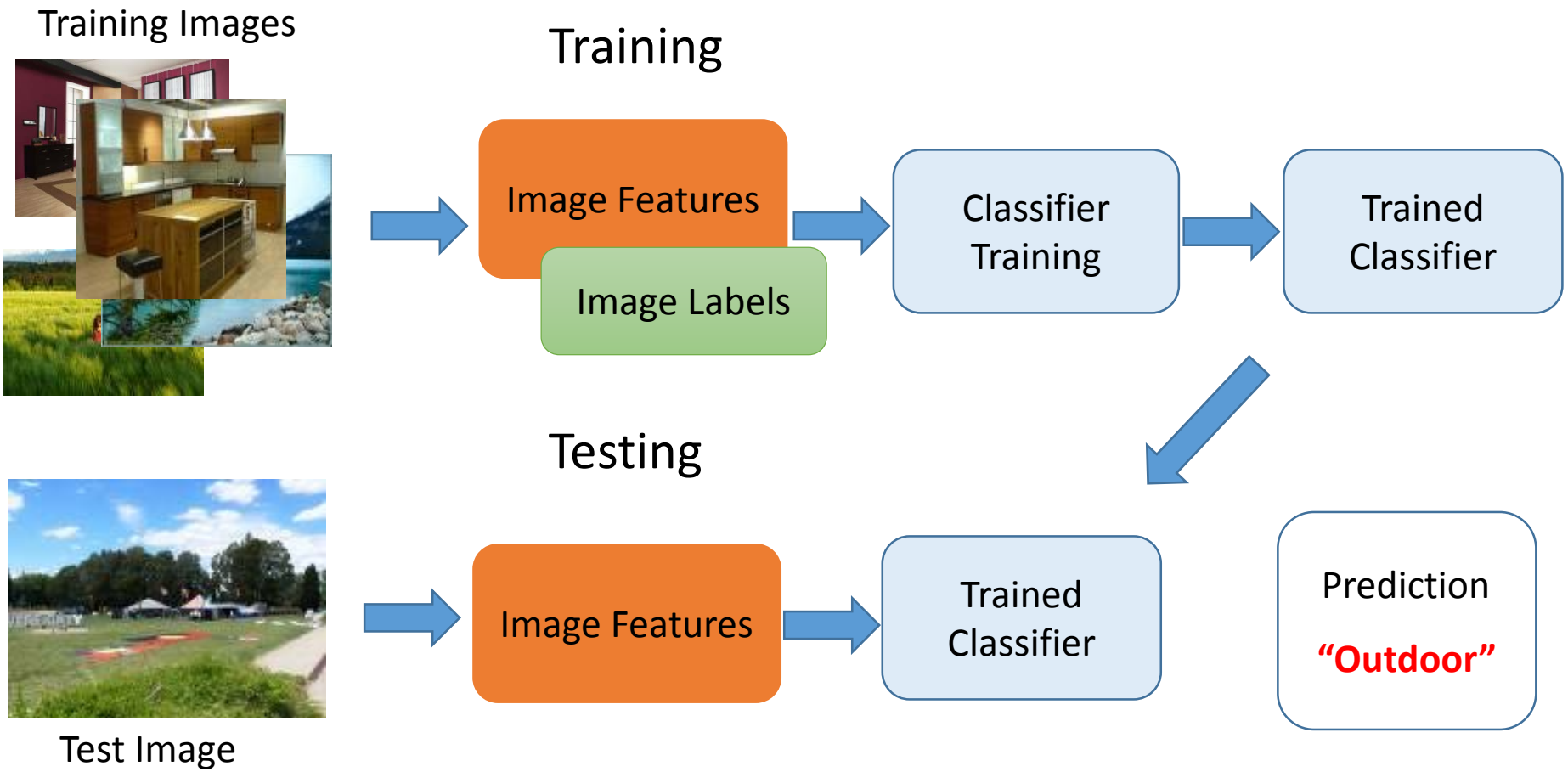


Classifier

= Category label

Supervised Learning for Visual Classification

- Training vs. Testing Phases

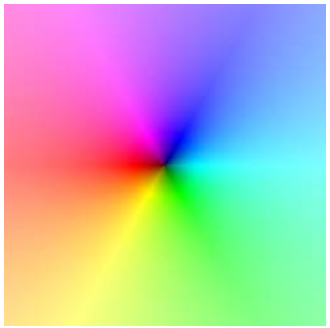


What Are the Right Features?

- Depending on the task of interest!
- Possible choices
  - Object: shape
    - Local shape info, shading, shadows, texture
  - Scene : geometric layout
    - linear perspective, gradients, line segments
  - Material properties: albedo, feel, hardness
    - Color, texture
  - Action: motion
    - Optical flow, tracked points

What's to Be Covered in Part II Today…

- Image Representation
  - Color Space
  - Image Filtering
  - Image Pyramid
- General Framework for Image Classification



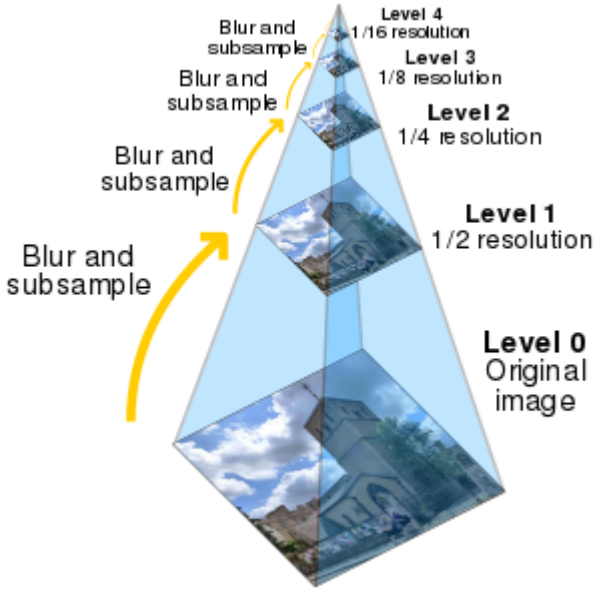
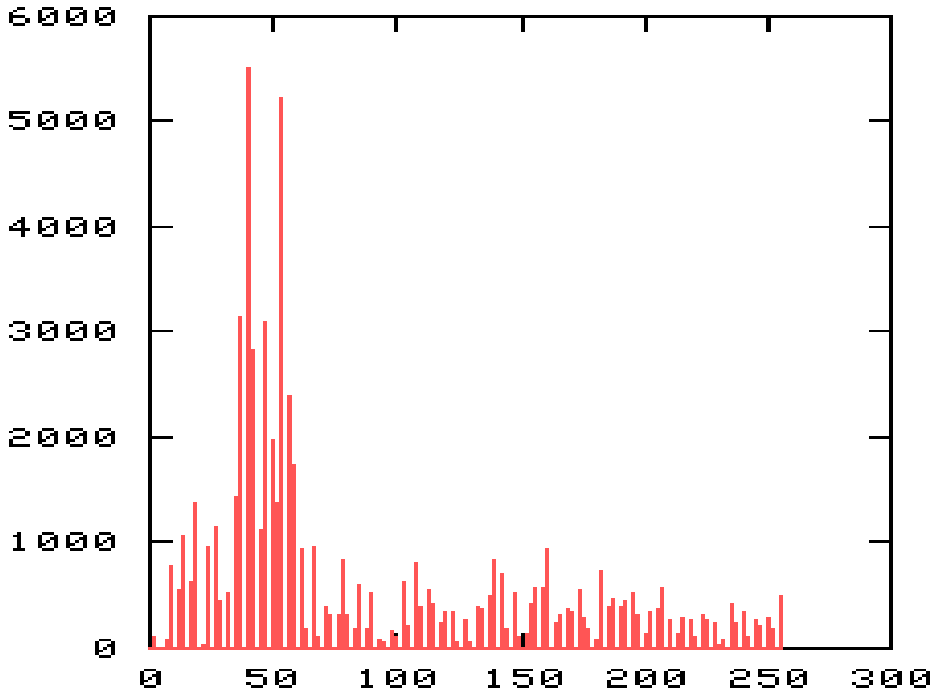
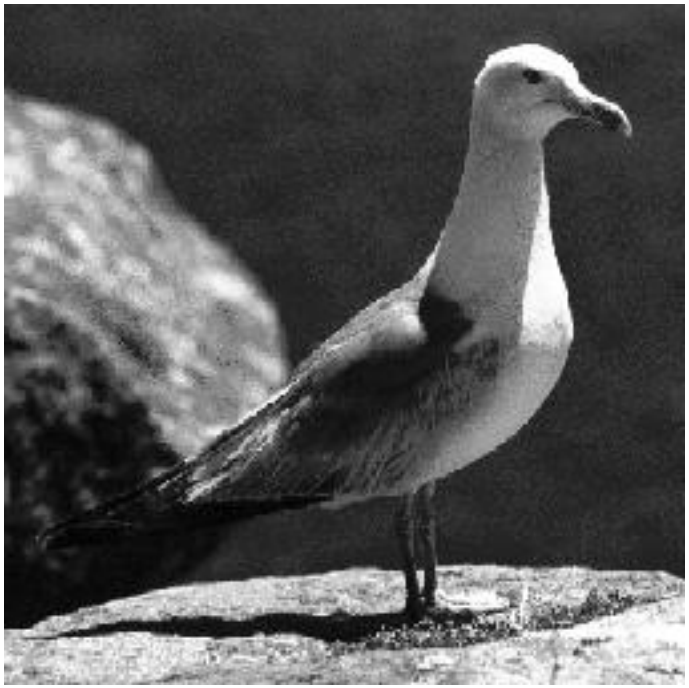


Image Representation: Histograms

- Global histogram
  - Possible to describe color, texture, depth, or even interest points!

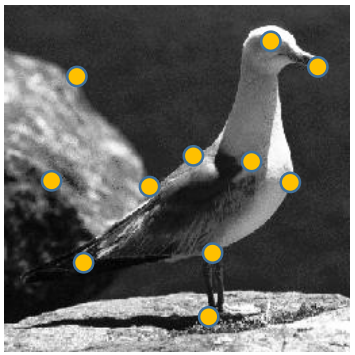




Images from Dave Kauchak

Image Representation: Histograms

- Take images with 2D features/descriptors as an example



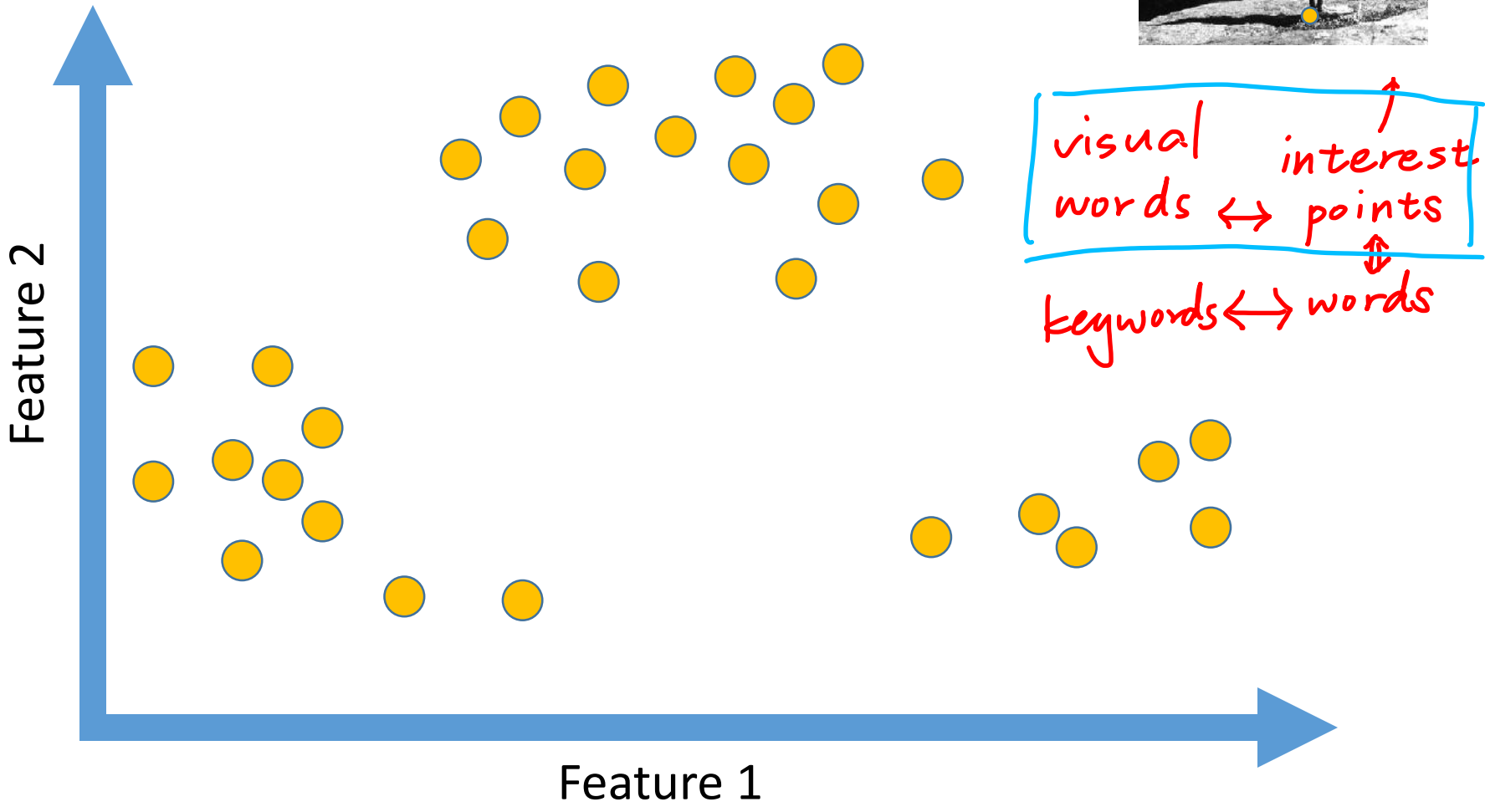


Image Representation: Histograms

- # of occurrence of data in each bin
- Marginal histogram of feature 1

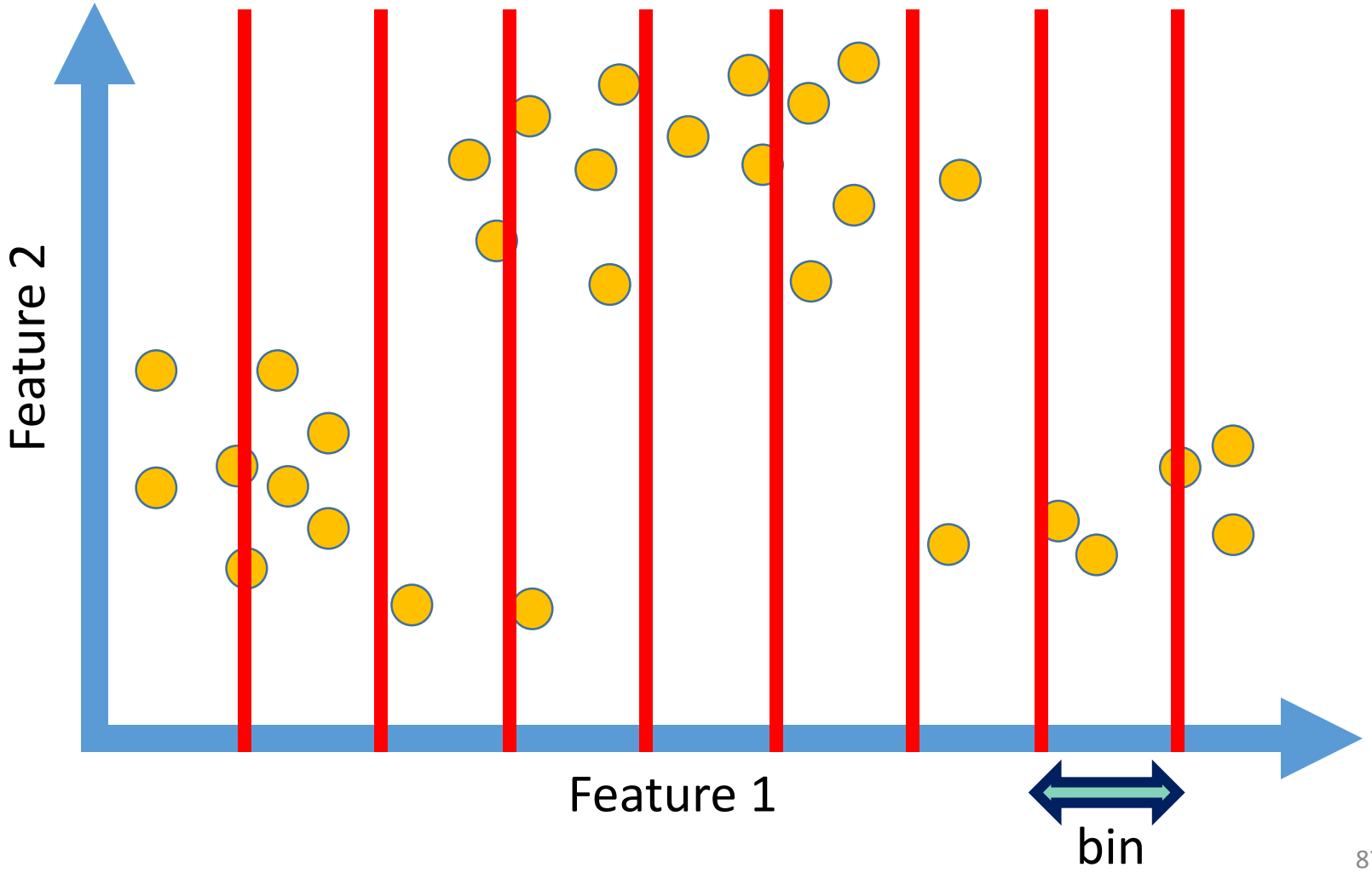


Image Representation: Histograms

- # of occurrence of data in each bin
- Marginal histogram of feature 2

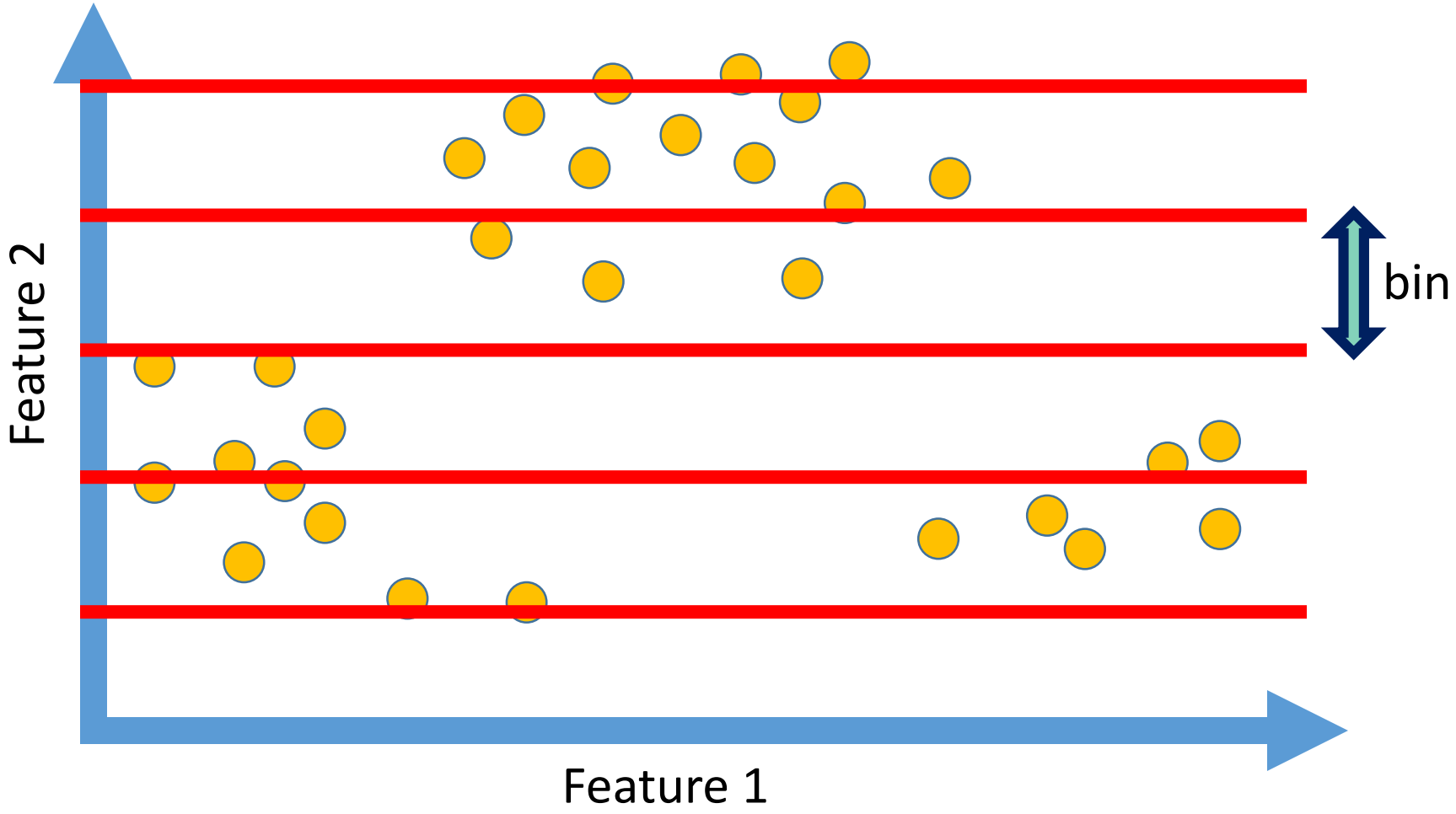


Image Representation: Histograms

- Better modeling (quantization) of multi-dimensional data
- Clustering

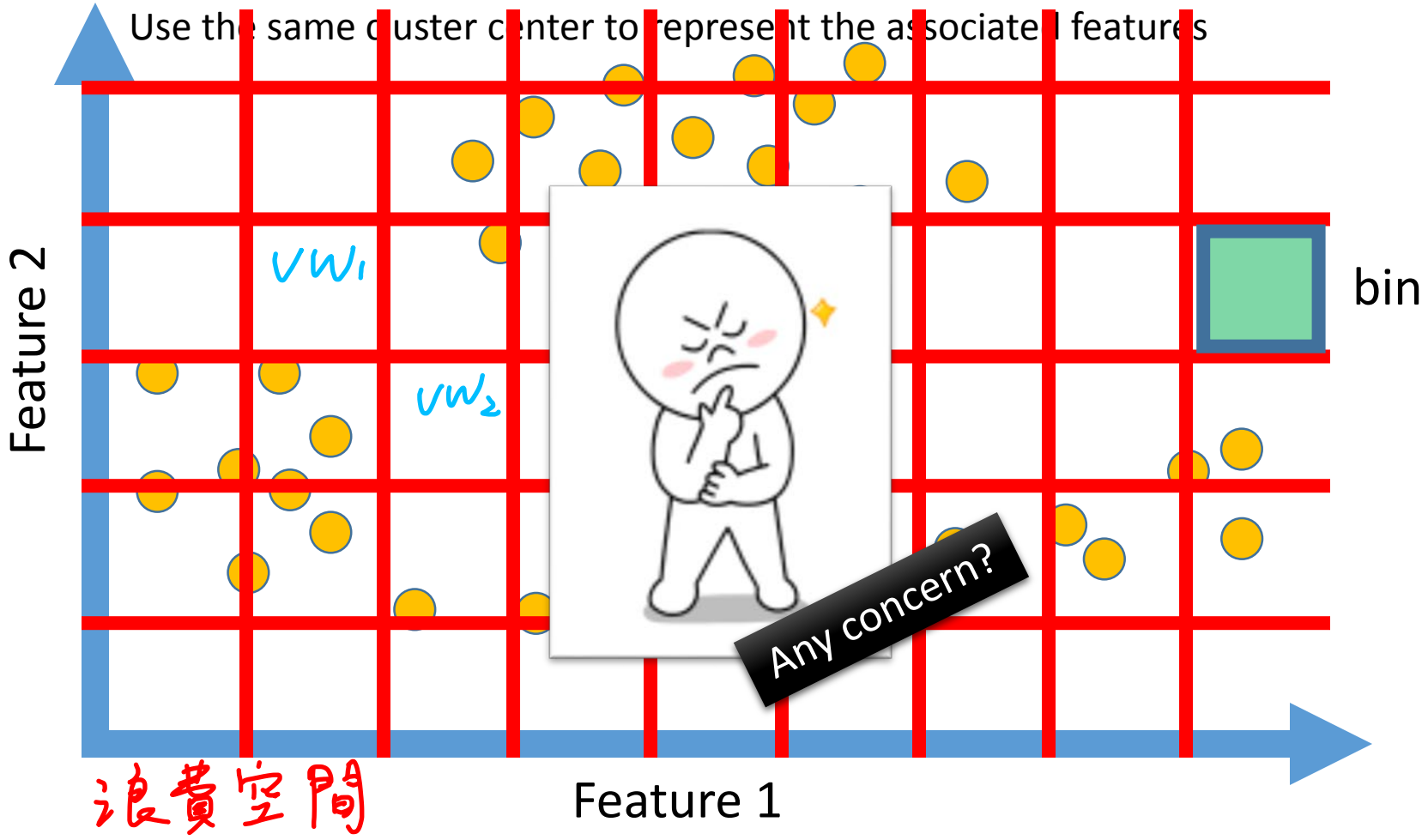
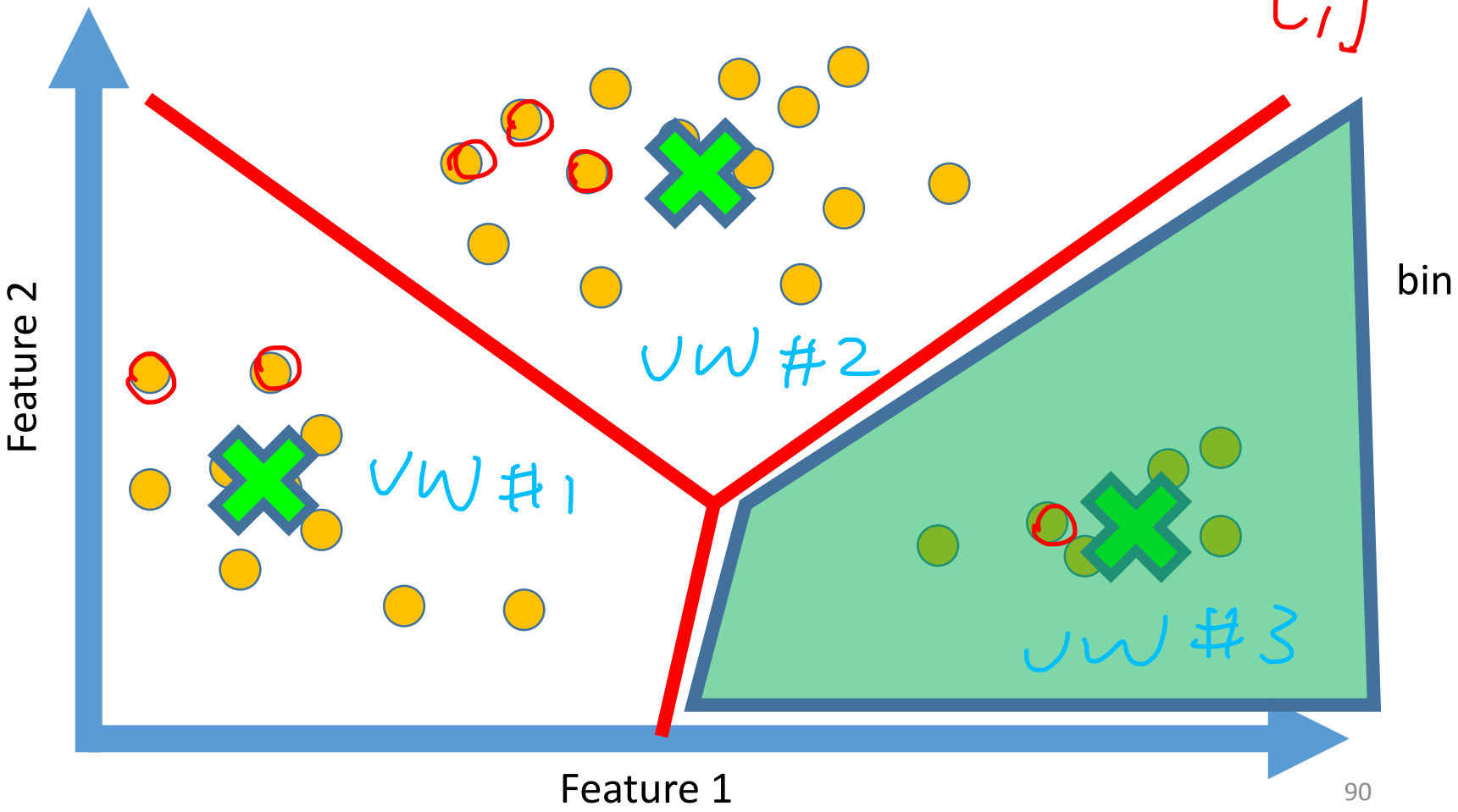


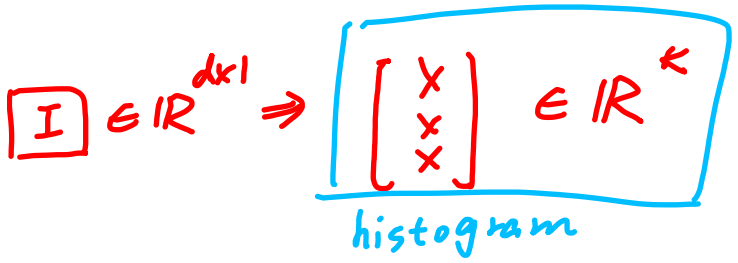
Image Representation: Histograms

- Better modeling (quantization) of multi-dimensional data
- Clustering
  - Use the same cluster center to represent the associated features



Remarks on Histogram-Based Image Representation

- Quantization
  - Grids vs. clusters



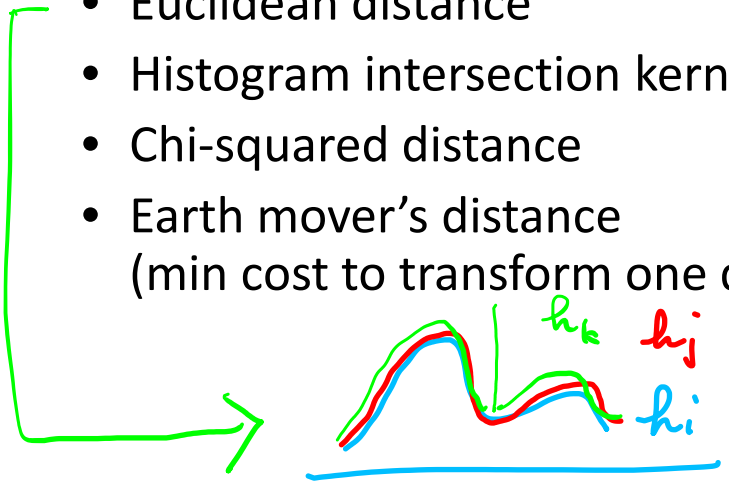
Fewer Bins
Need less data
Coarser representation

More Bins
Need more data
Finer representation

- Possible distance metrics
  - Euclidean distance
  - Histogram intersection kernel
  - Chi-squared distance
  - Earth mover's distance
    (min cost to transform one distribution to another)

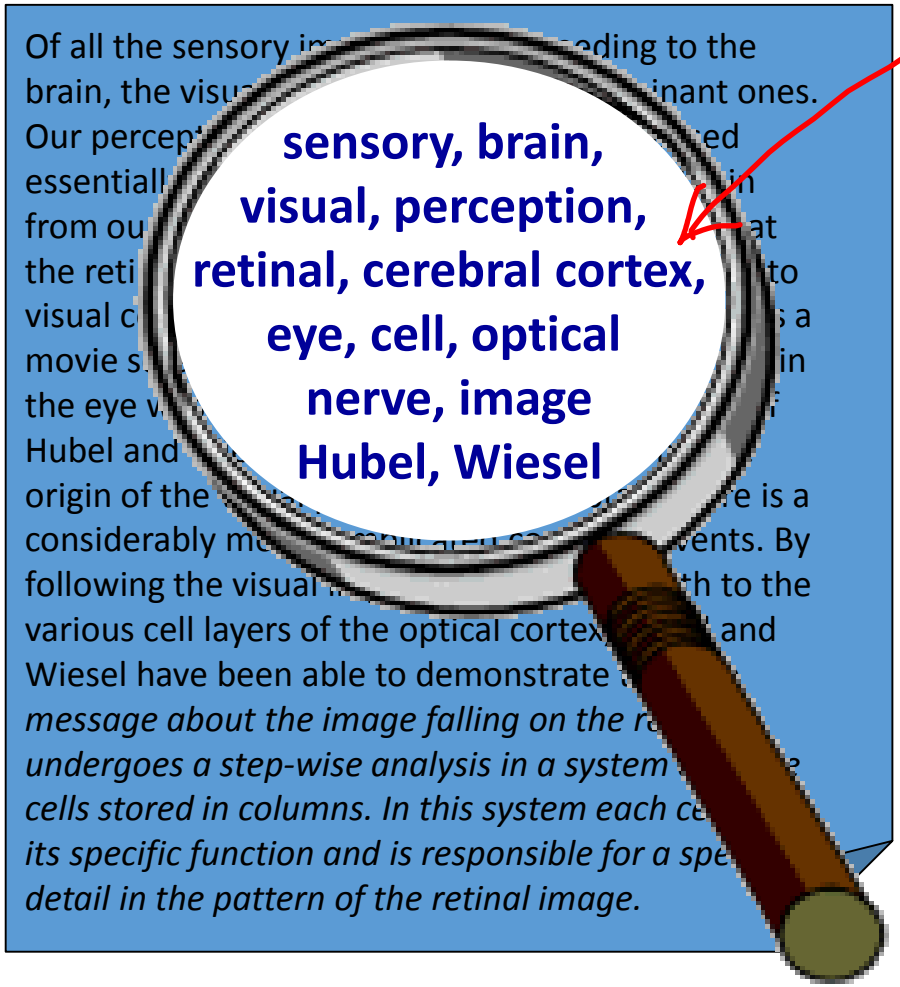
$$\text{histint}(h_i, h_j) = 1 - \sum_{m=1}^{K} \min\left(h_i(m), h_j(m)\right)$$

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

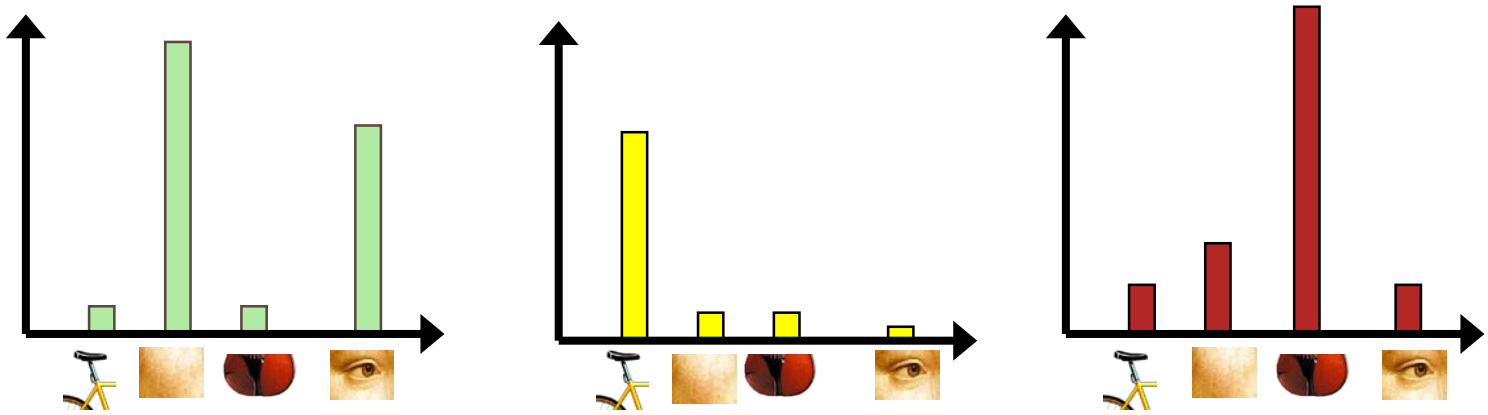


Bag-of-Words Models for Image Classification

- Analogy to document categorization



Bag of Words (or Visual Words)

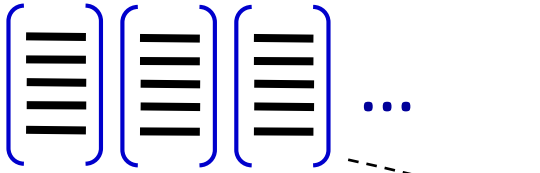


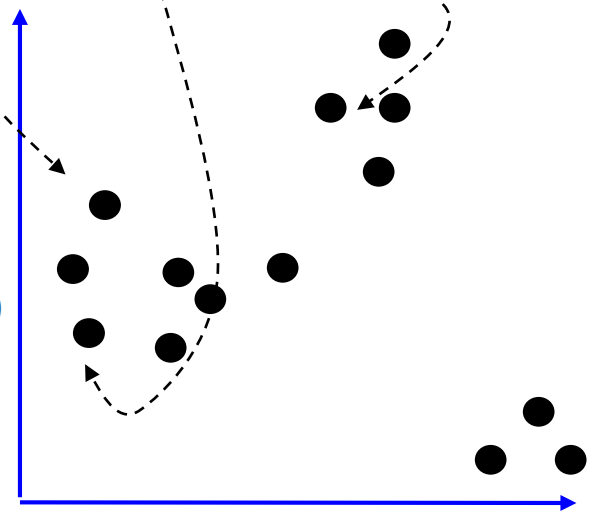
Bag-of-Words for Image Classification

- Training

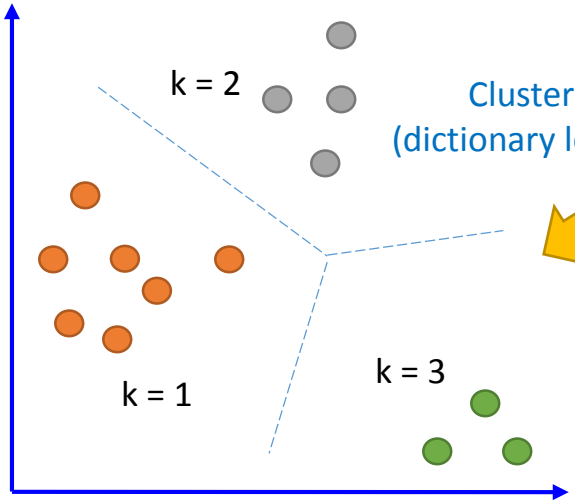
Training images

Interest point detection



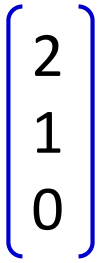


Clustering (dictionary learning)



Quantization (w/ normalization)

Feature Encoding

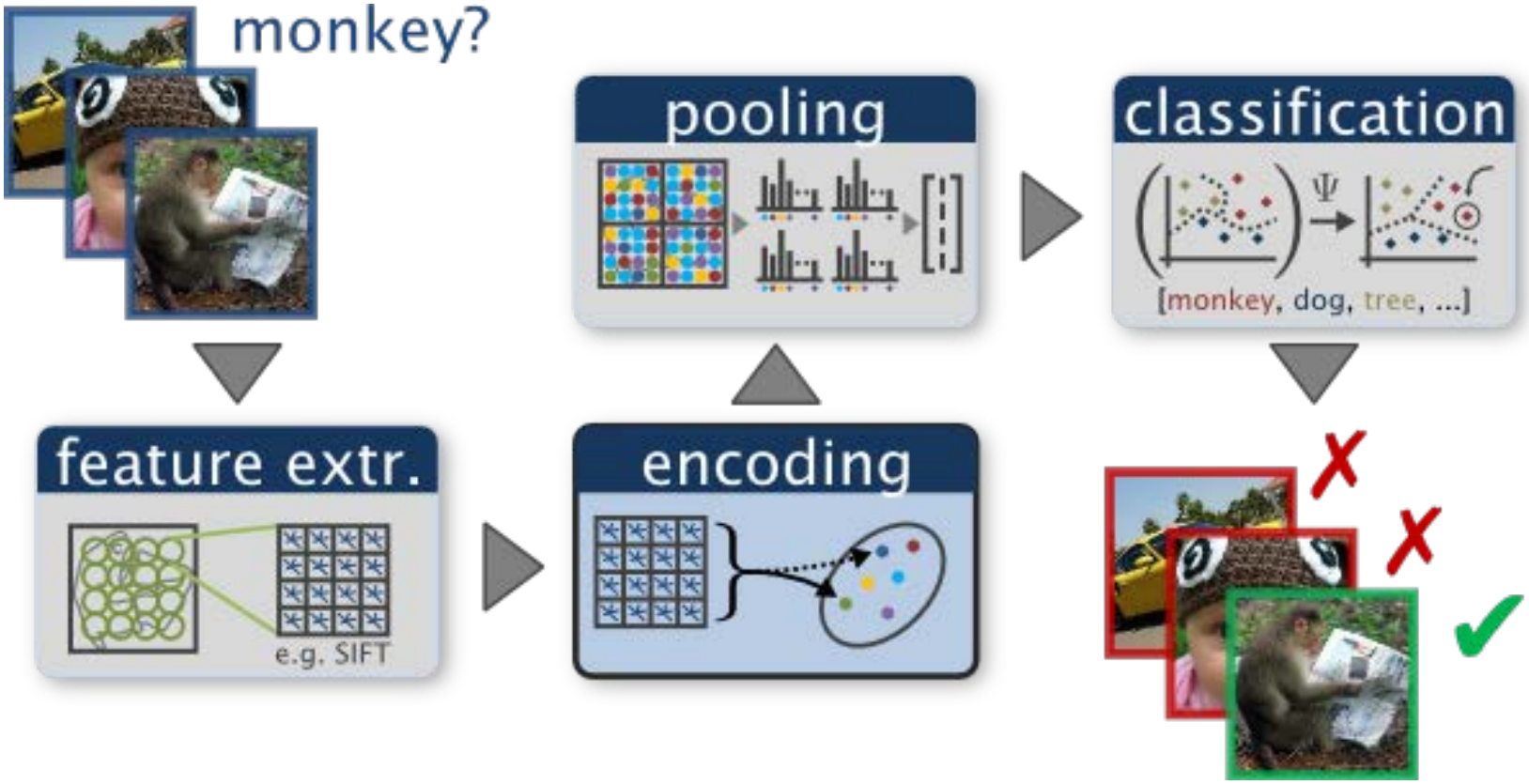


Bag-of-Words for Image Classification

- Testing

Bag-of-Words for Image Classification

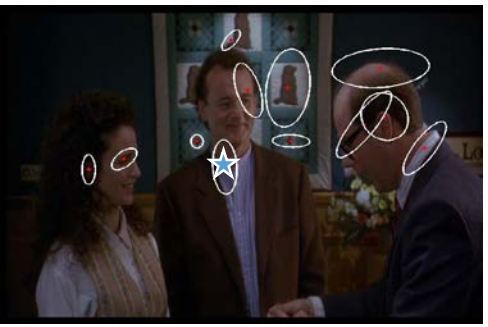
- Overview

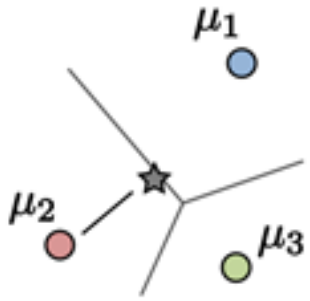


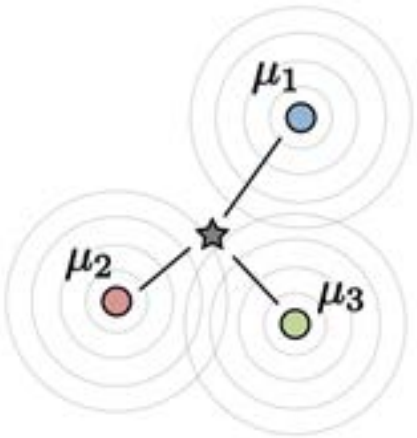
[Chatfieldet al. BMVC 2011]

About Feature Encoding for Bag-of-Words

- Hard vs. soft assignments to clusters

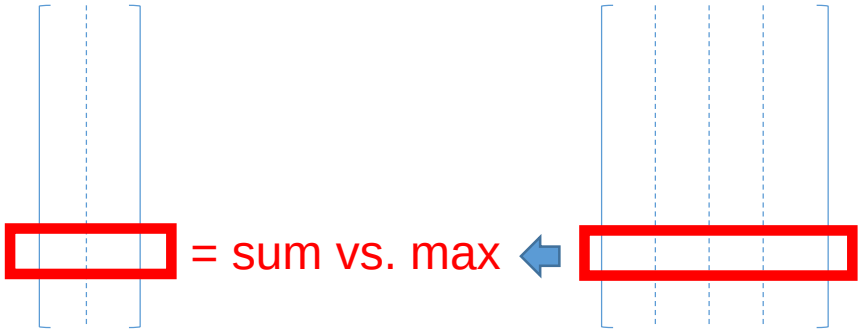


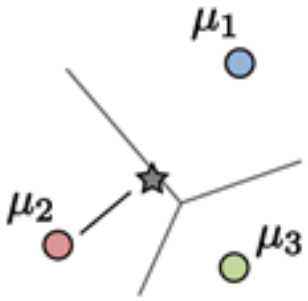


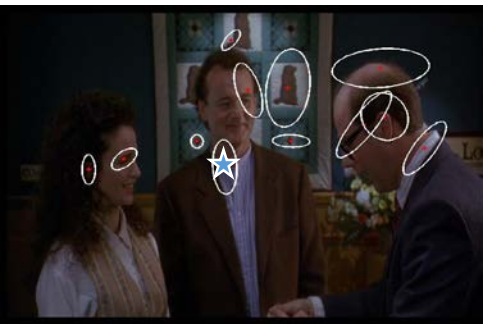


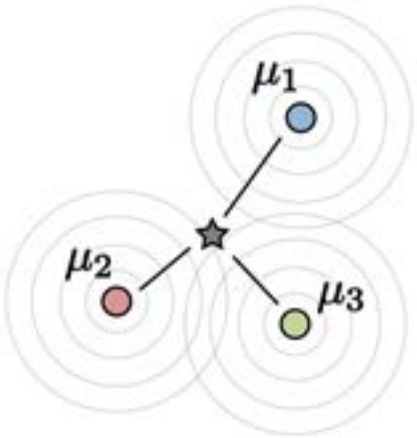
About Feature Encoding for Bag-of-Words

- Sum vs. max pooling



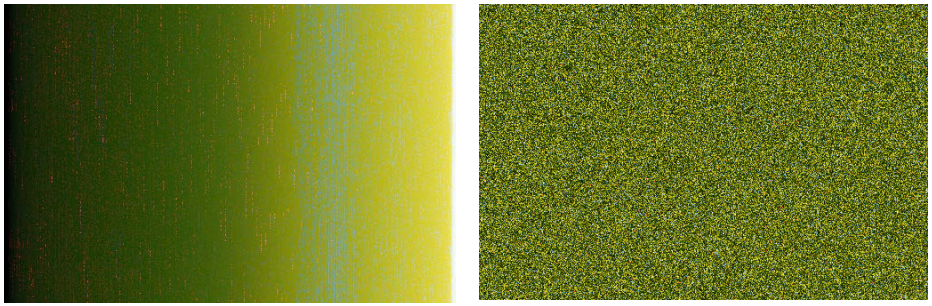


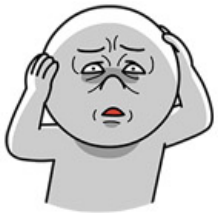


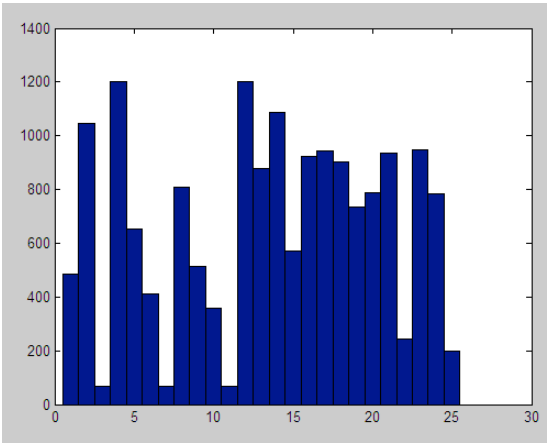


Final Remarks on BoW

- What's the limitation?
  - Loss of…
  - What's the possible solution?

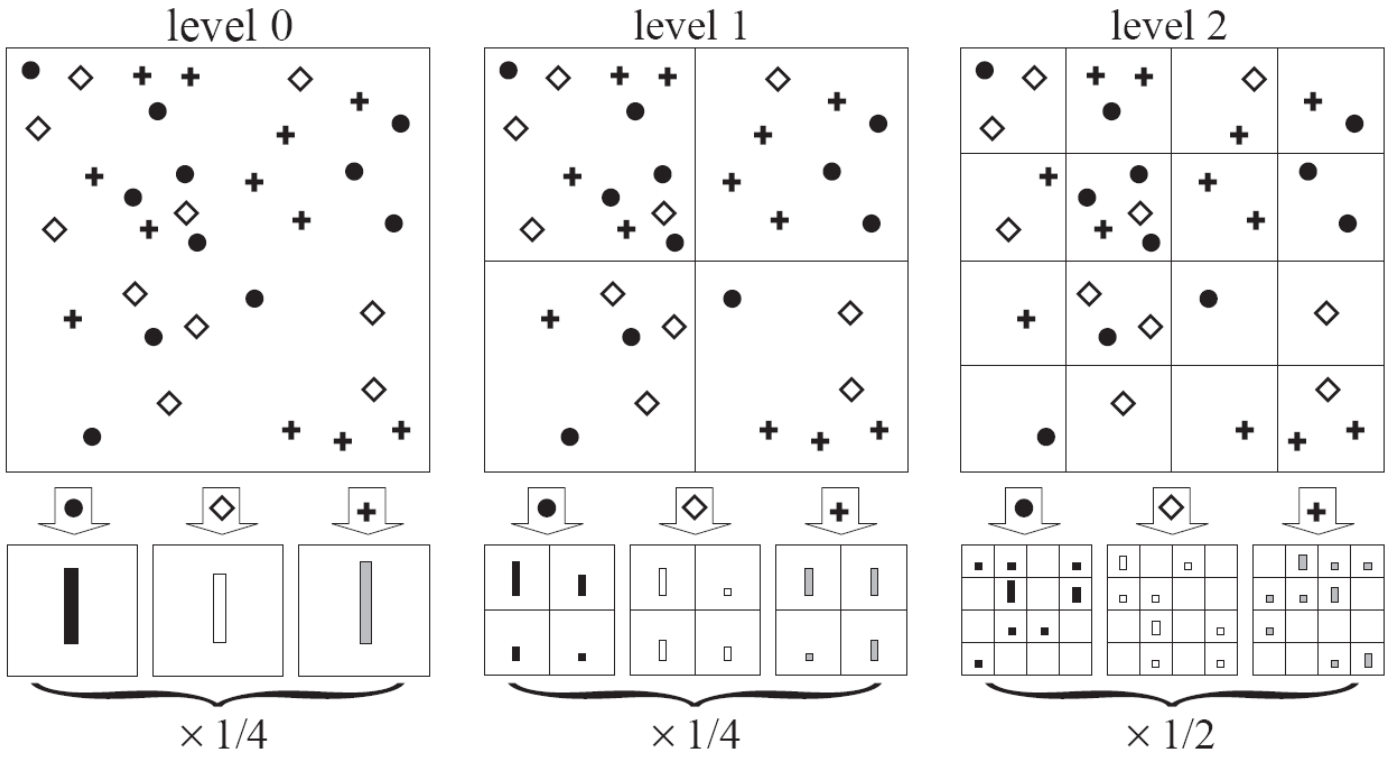






Final Remarks on BoW

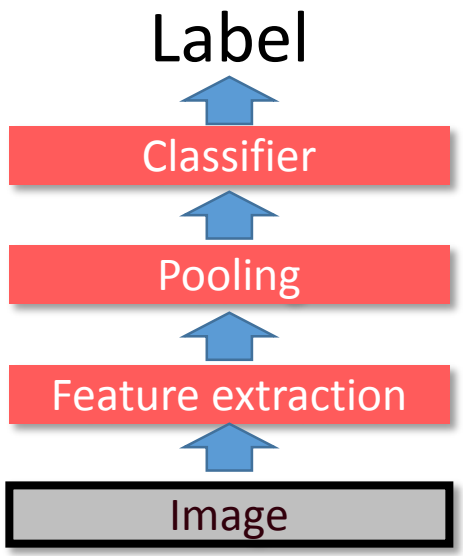
- Spatial pyramid
  - Compute BoW in each spatial grid + concatenation

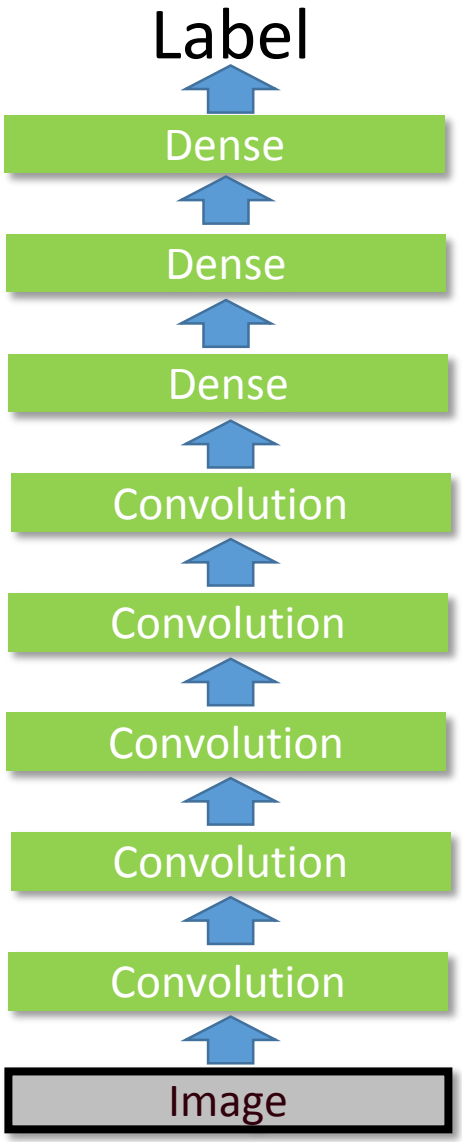


[Lazebnik et al. CVPR 2006]

Shallow vs. Deep Learning for Image Classification

- Engineered vs. deeply learned features
  - Lots of data + GPU





What We Learned Today…

- Unsupervised vs. Supervised Learning
  - Dimension Reduction, Clustering
  - Linear Classification
- Image Representation
  - Color Space
  - Image Filtering
  - Image Pyramid
- General Framework for Image Classification
- And, HW #1 is out and due in 2 weeks! No late submission for HW #1!!