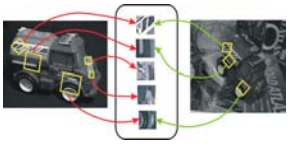
# EE 6882
# Visual Search Engine

Prof. Shih-Fu Chang, Feb. 13th 2012
Lecture #4

- Local Feature Matching
- Bag of Word image representation: coding and pooling



(Many slides from A. Efors, W. Freeman, C. Kambhamettu, L. Xie, and likely others)
(Slides preparation assisted by Rong-Rong Ji)

## Corner Detection

- Types of local image windows
  - *Flat*: Little or no brightness change
  - *Edge*: Strong brightness change in single direction
  - *Flow*: Parallel stripes
  - *Corner/spot*: Strong brightness changes in orthogonal directions
- Basic idea
  - Find points where two edges meet
  - Look at the gradient behavior over a small window

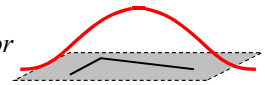
(Slide of A. Efros)

## Harris Detector: Mathematics

*Change of intensity for the shift [u,v]:*

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

*Window function* — *Shifted intensity* — *Intensity*

*Window function* $w(x,y)$ =  *or* 

*1 in window, 0 outside* *Gaussian*

## Harris Detector: Mathematics

*Taylor's Expansion: For small shifts* $[u,v]$ *we have a bilinear approximation:*

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \quad M \quad \begin{bmatrix} u \\ v \end{bmatrix}$$

*where M is a 2×2 matrix computed from image derivatives:*

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

## Harris Detector: Mathematics

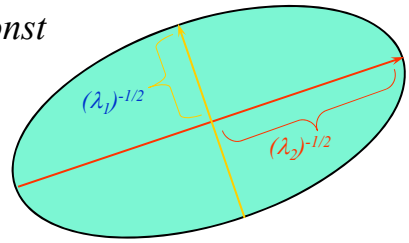
*Intensity change in shifting window: eigenvalue analysis*

$$E(u,v) \cong \begin{bmatrix} u, v \end{bmatrix} \quad M \quad \begin{bmatrix} u \\ v \end{bmatrix}$$

$\lambda_1 > \lambda_2$ – *eigenvalues of* $M$

*If we try every possible shift, the direction of fastest change is* $\lambda_1$

*Ellipse E(u,v) = const*




(Slide of K. Efros)

## Harris Detector: Mathematics

*Measure of corner response:*

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$R = \frac{\det M}{\operatorname{Trace} M}$$

*Or*

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

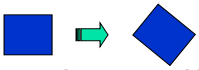
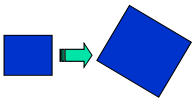
$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

*(k – empirical constant, k = 0.04-0.06)*

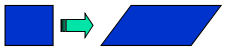
## Harris Detector

- The Algorithm:
  - Find points with large corner response function $R$ ($R$ > threshold)
  - Take the points of local maxima of $R$

## Models of Image Change

- Geometry
  - Rotation 
  - Similarity (rotation + uniform scale) 
  - Affine (scale dependent on direction)
    valid for: orthographic camera, locally planar object

$$p' = \mathbf{H}p$$

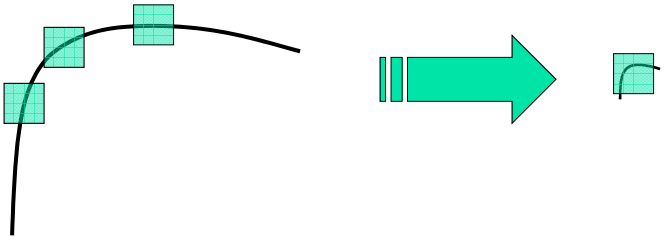


- Photometry
  - Affine intensity change ($I \rightarrow a\,I + b$) 

(Slide of C. Kambhamettu)

## Harris Detector: Some Properties

- But: non-invariant to *image scale*!



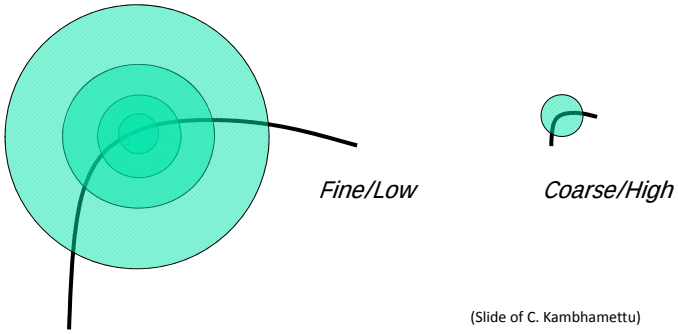
*All points will be classified as edges*

*Corner !*

(Slide of C. Kambhamettu)

## Scale Invariant Detection

- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes (at different scales) will look the same in both images



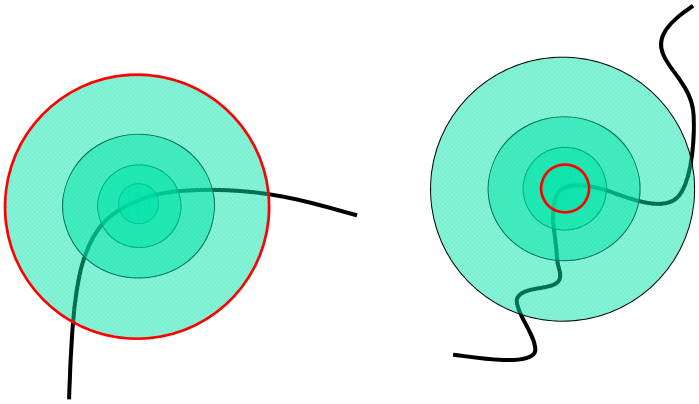
*Fine/Low*

*Coarse/High*

(Slide of C. Kambhamettu)

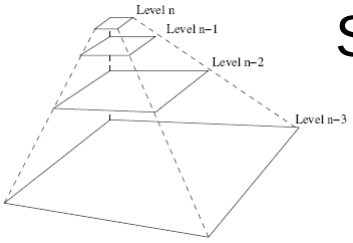
## Scale Invariant Detection

- The problem: how do we choose corresponding circles ***independently*** in each image?

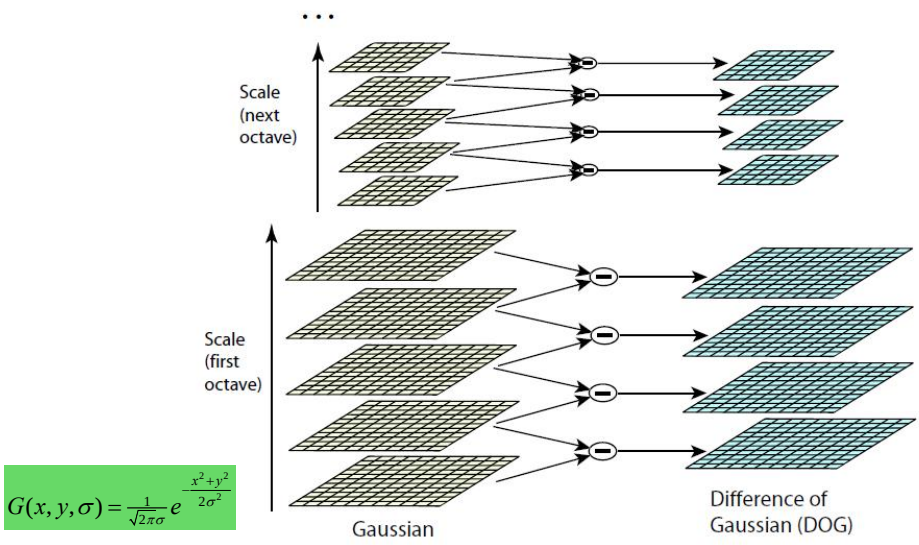


(Slide of C. Kambhamettu)

## Scale-Space Pyrimad




# Scale Space: Difference of Guassian




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

# Scale Invariant Detection

- Functions for determining scale

$$f = \text{Kernel} * \text{Image}$$

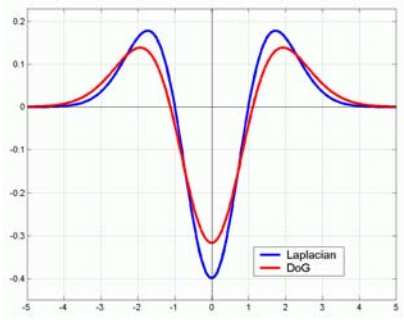
*Kernels:*

$$DoG = G(x,y,k\sigma) - G(x,y,\sigma)$$

*(Difference of Gaussians)*

$$L = \sigma^2\left(G_{xx}(x,y,\sigma) + G_{yy}(x,y,\sigma)\right)$$

*(Laplacian)*




*where Gaussian*

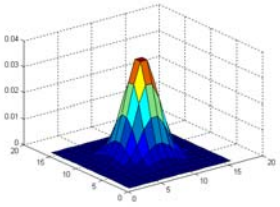
$$G(x,y,\sigma)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

*Note: both kernels are invariant to scale and rotation*

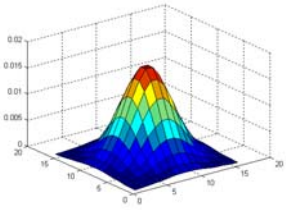
(Slide of C. Kambhamettu)

# Gausian Kernel, DOG

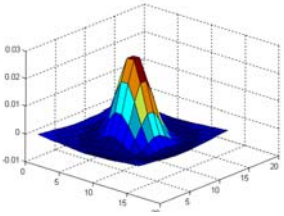
Sigma 2



Sigma 4



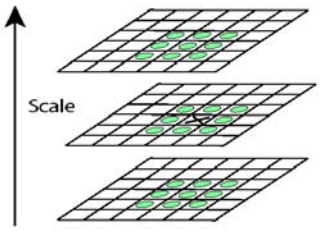
Diff Sigma2-Sigma4



# Difference of Gaussian, DOG

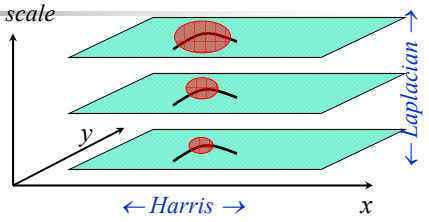
## Key Point Localization

- Detect maxima and minima of difference-of-Gaussian in scale space

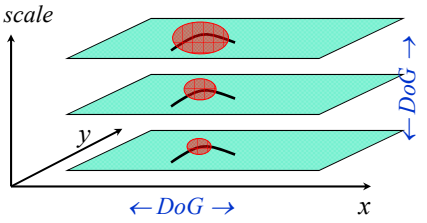



## Scale Invariant Interest Point Detectors

- Harris-Laplacian[1]
  *Find local maximum of:*
  - Harris corner detector in space (image coordinates)
  - Laplacian in scale




- *SIFT (Lowe)*[2]
  *Find local maximum of:*
  - *Difference of Gaussians in space and scale*




(Slide of C. Kambhamettu)

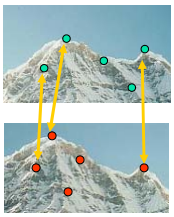
[1] K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001
[2] D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004

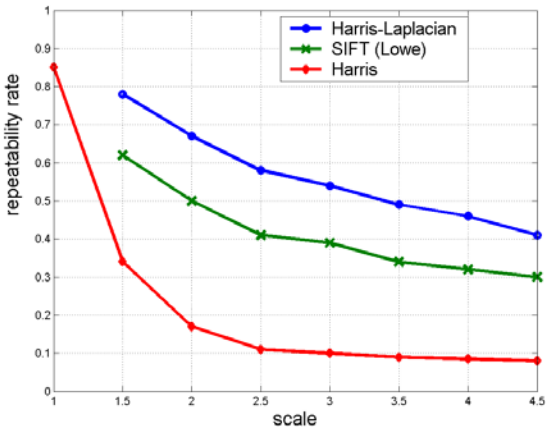
## Scale Invariant Detectors

- Experimental evaluation of detectors w.r.t. scale change

*Repeatability rate:*

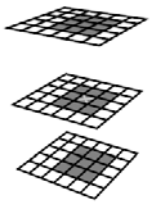
$$\frac{\#\ \text{correct correspondences}}{\text{avg}\ \#\ \text{detected points}}$$

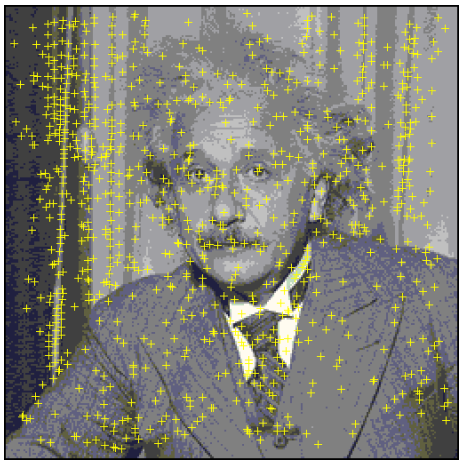




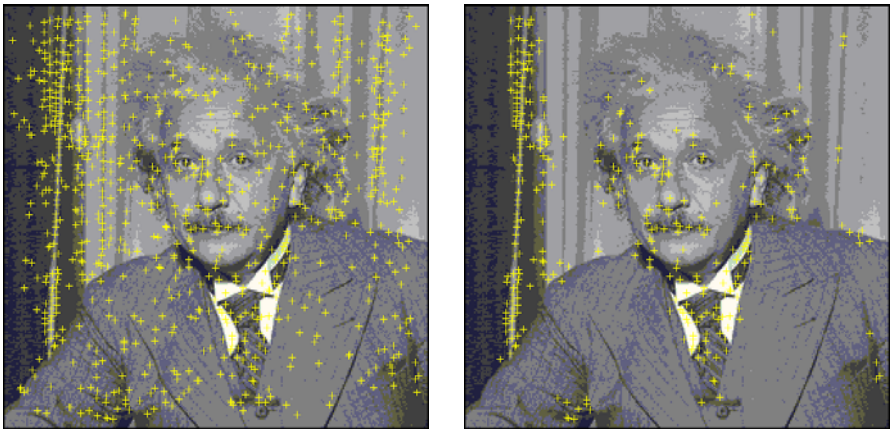

*K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001*

## SIFT keypoints

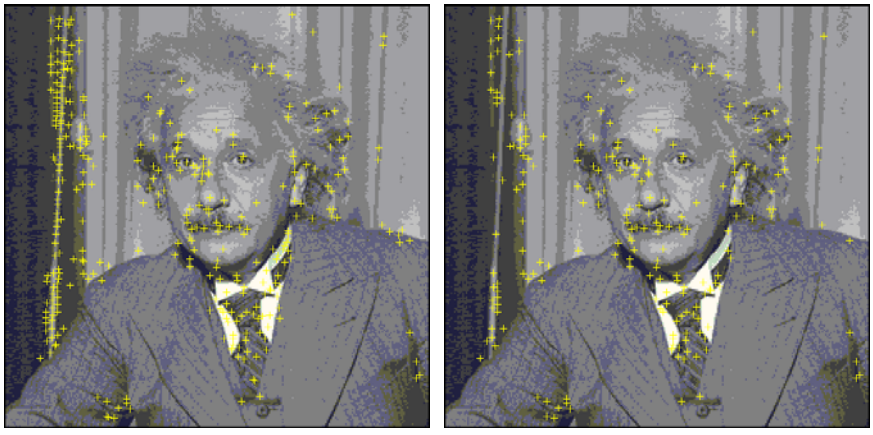




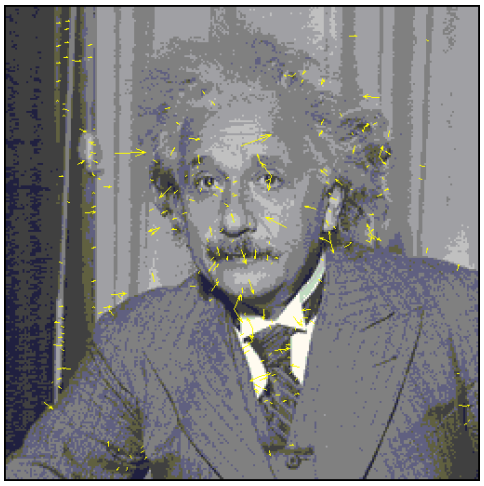
## After extrema detection



## After curvature, edge responses



# Keypoints orientation and scale

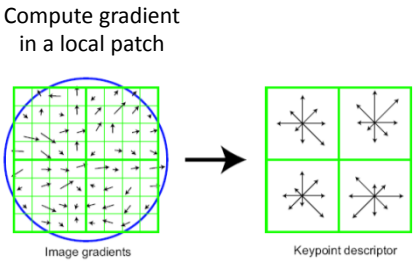


# SIFT Invariant Descriptors

- Extract image patches relative to local orientation

*Dominant direction of gradient*

# Local Appearance Descriptor (SIFT)




Histogram of oriented gradients over local grids

- e.g., 4x4 grids and 8 directions
  -> 4x4x8=128 dimensions
- Scale invariant

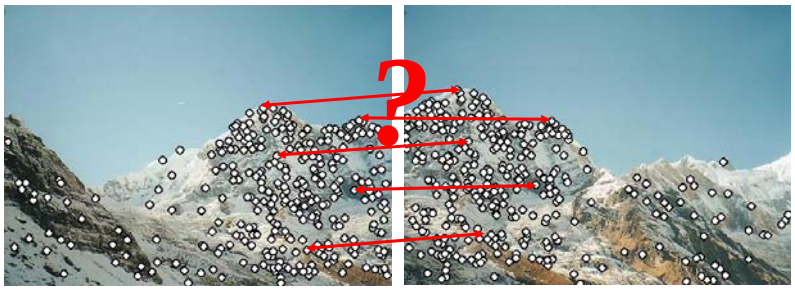


[Lowe, ICCV 1999]

## Point Descriptors

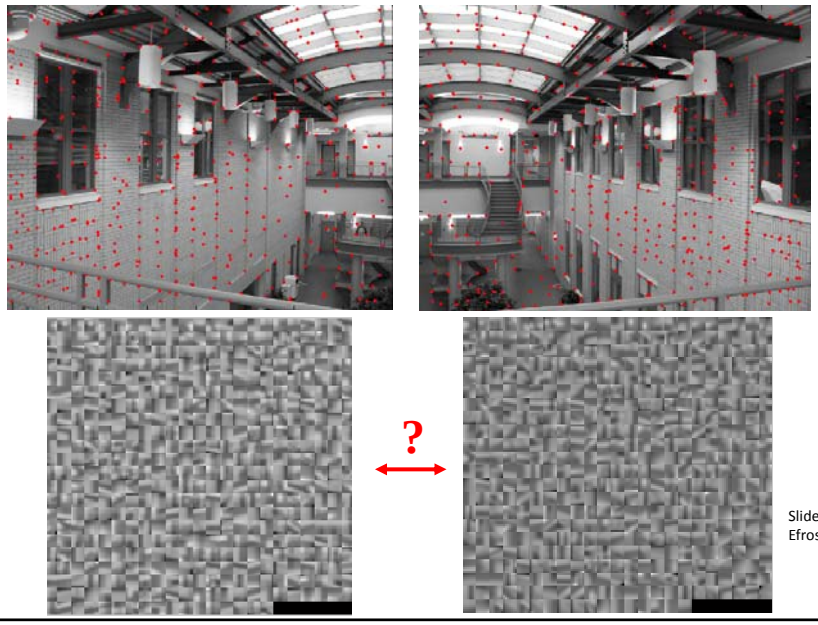
- We know how to detect points
- Next question:

**How to match them?**




- Point descriptor should be:
  - Invariant
  - Distinctive

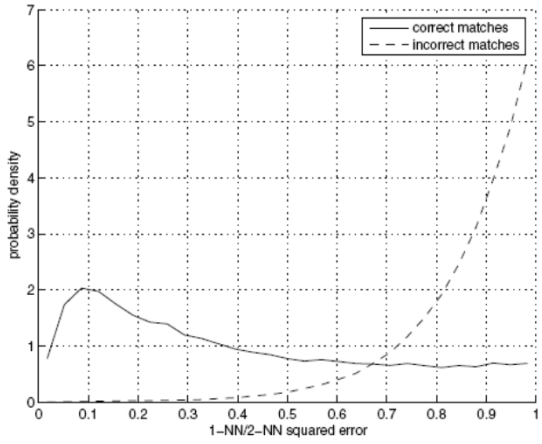
## Feature matching




Slide of A. Efros

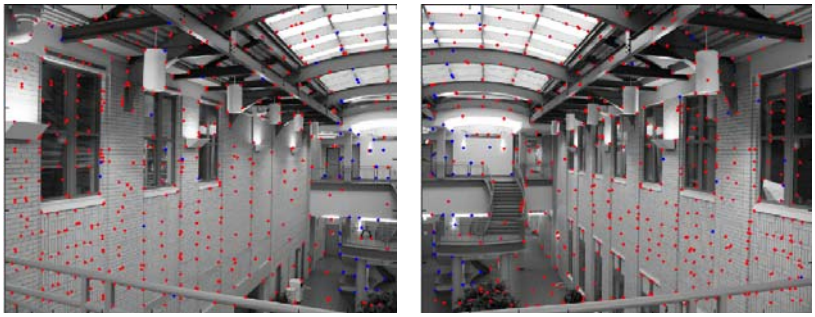
## Feature-space outlier rejection [Lowe, 1999]:

- 1-NN: SSD of the closest match
- 2-NN: SSD of the second-closest match
- Look at how much the best match (1-NN) is than the 2nd best match (2-NN), e.g. 1-NN/2-NN




Slide of A. Efros

## Feature-space outliner rejection



Can we now compute H from the blue points?

- No! Still too many outliers…
- What can we do?

Slide of A. Efros

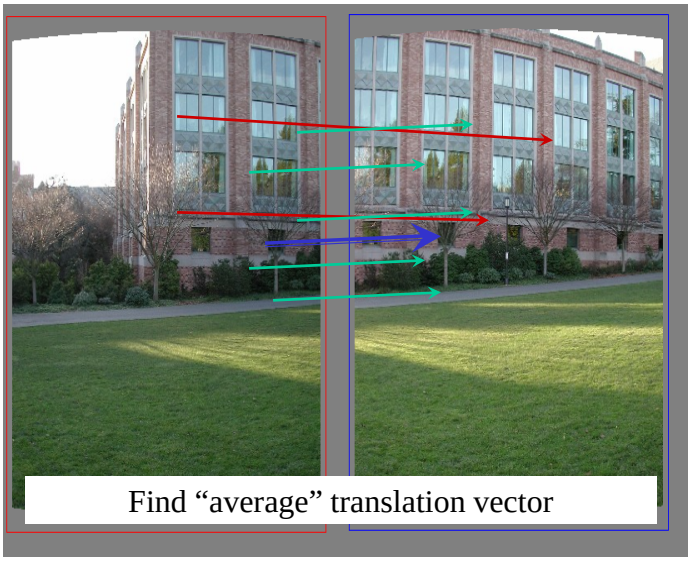
## RANSAC for estimating homography

RANSAC loop:

1. Select four feature pairs (at random)
2. Compute homography H (exact)
3. Compute *inliers* where $SSD(p_i', \boldsymbol{H}\, p_{i)} < \varepsilon$
4. Keep largest set of inliers
5. Re-compute least-squares H estimate on all of the inliers

Slide of A. Efros

## Least squares fit

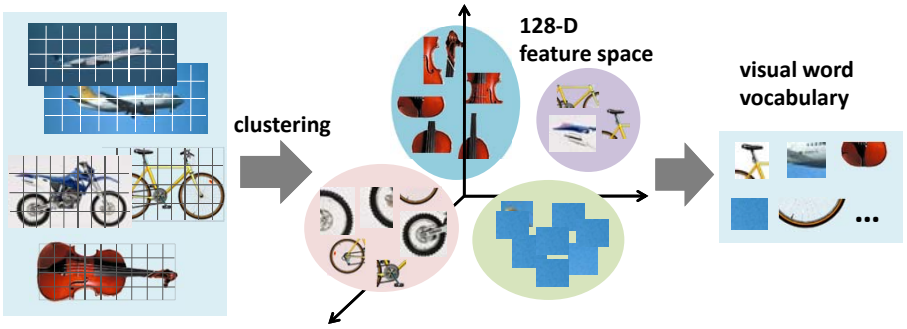



Slide of A. Efros

## RANSAC

Slide of A. Efros

# From local features to *Visual Words*

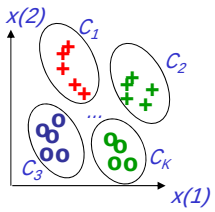



# K-Mean Clustering

- Training data

$$\{x_i\} + \{label(i)\ ?\ \}$$

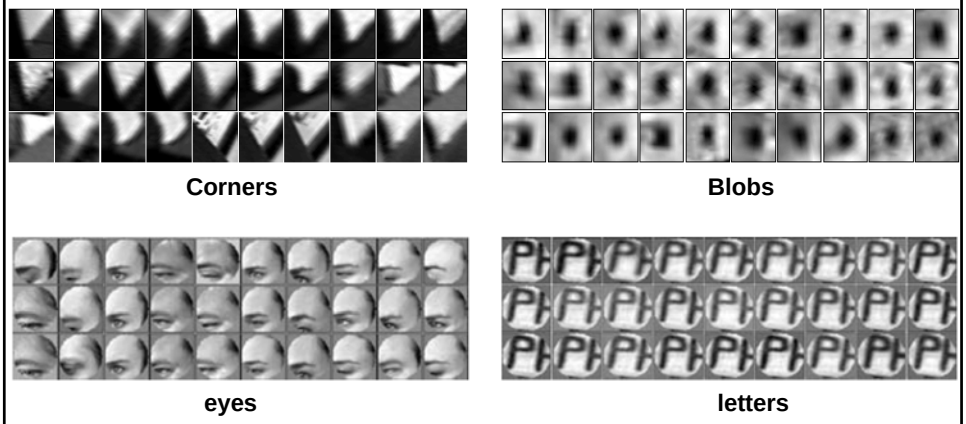
- Unsupervised learning
- K-mean clustering
  - Fix K value
  - Initialize the representative of each cluster
  - Map samples to closest cluster
  - Re-compute the centers




$$x_1, x_2, ..., x_N \ \ samples$$
$$for\ i=1,2,...,N,$$
$$x_i \rightarrow C_k, if\ \ Dist(x_i, C_k) < Dist(x_i, C_{k'}), k \neq k'$$
$$end$$

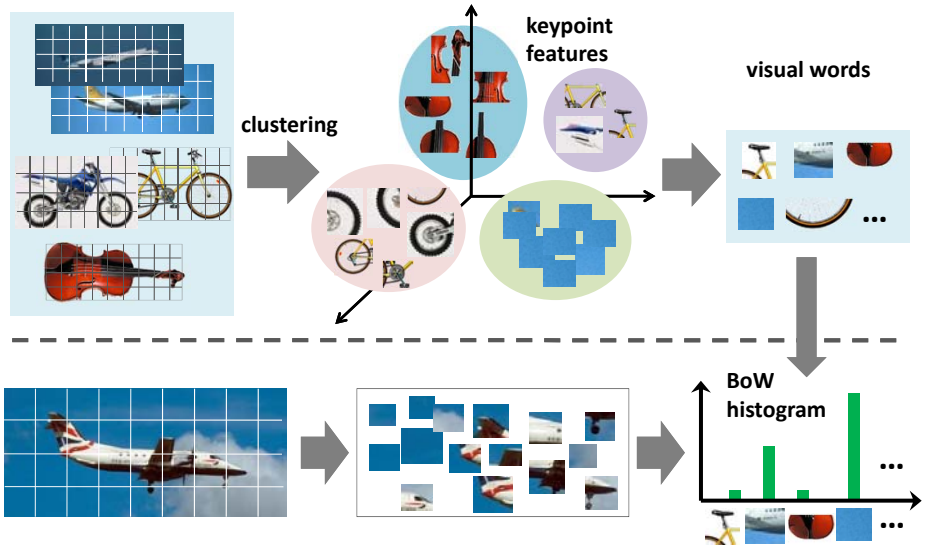
- Can be used to initialize other clustering methods

# Visual Words: Image Patch Patterns




Sivic and Zisserman, "Video Google", 2006

# Represent Image as Bag of Words



# Pooling Binary Features

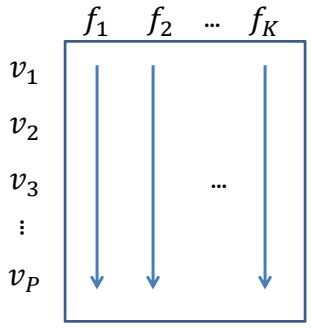
*Boureau, Jean Ponce, Yann LeCun, A Theoretical Analysis of Feature Pooling in Visual Recognition, ICML 2010*

$$\text{average pooling } f_a(\boldsymbol{v}) = \frac{1}{P}\sum_{i=1}^{P} \boldsymbol{v}_i$$

$$\text{max pooling } f_m(\boldsymbol{v}) = \max_i \boldsymbol{v}_i.$$

*Consider PxK matrix*
*P: # of features, K: # of codewords*

*To begin with simple model, assume vi are iid.*




# Distribution Separability

Given two classes $C_1$ and $C_2$, we examine the separation of conditional distributions $p(f_m|C_1)$ and $p(f_m|C_2)$, and $p(f_a|C_1)$ and $p(f_a|C_2)$.

*Better separability achieved by*

1. *increasing the distance between the means of the two class-conditional distributions*
2. *reducing their standard deviations.*

# Distribution Separability

Average pooling:

$$\text{mean } \mu_a = \alpha, \text{ and variance } \sigma_a^2 = \alpha(1-\alpha)/P.$$

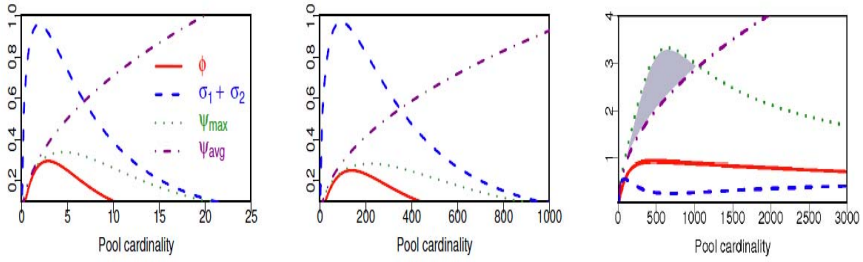
Max pooling:

$$\mu_m = 1 - (1-\alpha)^P$$
$$\sigma_m^2 = (1-(1-\alpha)^P)(1-\alpha)^P.$$

*Class separability*

$$\psi_{avg} = |\alpha_1 - \alpha_2|.\sqrt{P}/(\sqrt{\alpha_1.(1-\alpha_1)} + \sqrt{\alpha_2.(1-\alpha_2)})$$

$$\psi_{max} = \phi/(\sigma_1 + \sigma_2) \qquad \phi(P) = |(1-\alpha_1)^P - (1-\alpha_2)^P|$$




(a) $\alpha_1 = 0.4, \alpha_2 = 0.2$ (b) $\alpha_1 = 1.10^{-2}, \alpha_2 = 5.10^{-3}$ (c) $\alpha_1 = 1.10^{-2}, \alpha_2 = 1.10^{-4}$

*Figure 1.* $\phi(P) = |(1-\alpha_1)^P - (1-\alpha_2)^P|$, $\sigma_1$ and $\sigma_2$ denote the distance between the expectations of the max-pooled features of mean activation $\alpha_1$ and $\alpha_2$, and their standard deviations, respectively. $\psi_{max} = \phi/(\sigma_1+\sigma_2)$ and $\psi_{avg} = |\alpha_1 - \alpha_2|.\sqrt{P}/(\sqrt{\alpha_1.(1-\alpha_1)} + \sqrt{\alpha_2.(1-\alpha_2)})$ give a measure of separability for max and average pooling. $\phi$ reaches its peak at smaller cardinalities than $\psi_{max}$. (a) When features have relatively large activations, the peak of separability is obtained for small cardinalities (b) With coarser feature activations, the range of the peak is much larger (note the change of scale in the x-axis). (c) When

*For binary features:*

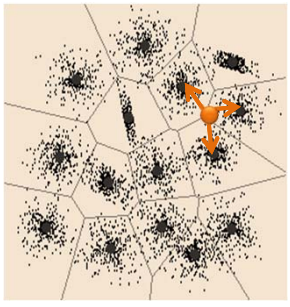
- Max pooling is particularly well suited to the separation of features that are very sparse (i.e., have a very low probability of being active)
- Using all available samples to perform the pooling may not be optimal
- The optimal pooling cardinality should increase with dictionary size

*For continuous features:*

- Modeling will be more complex and the conclusions are slightly different

## Soft Coding

$$\alpha_{i,j} = \frac{\exp\left(-\beta \|\boldsymbol{x}_i - \mathbf{d}_j\|_2^2\right)}{\sum_{k=1}^{K} \exp\left(-\beta \|\boldsymbol{x}_i - \mathbf{d}_k\|_2^2\right)}$$



*-- Assign a feature to multiple visual words*

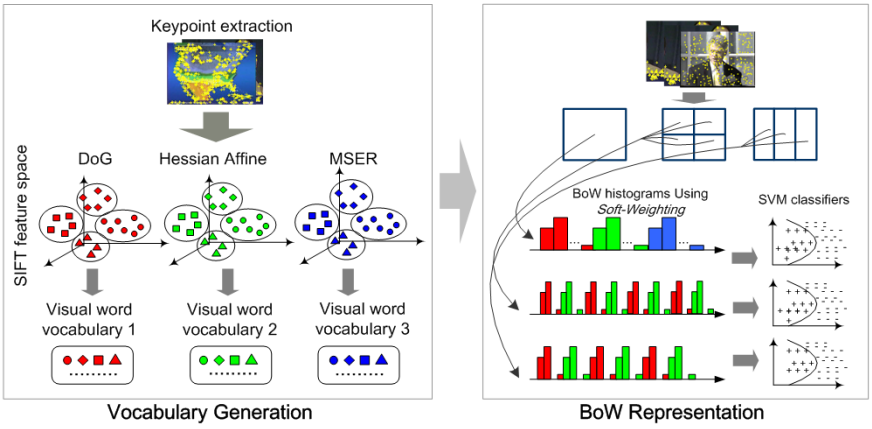
*-- weights are determined by feature-to-word similarity*

Details in: Jiang, Ngo and Yang, ACM CIVR 2007.

42

*Image source: http://www.cs.joensuu.fi/pages/franti/vq/lkm15.gif*

# Multi-BoW Spatial Pyramid Kernel




*S. Lazebnik, et al, CVPR 2006*

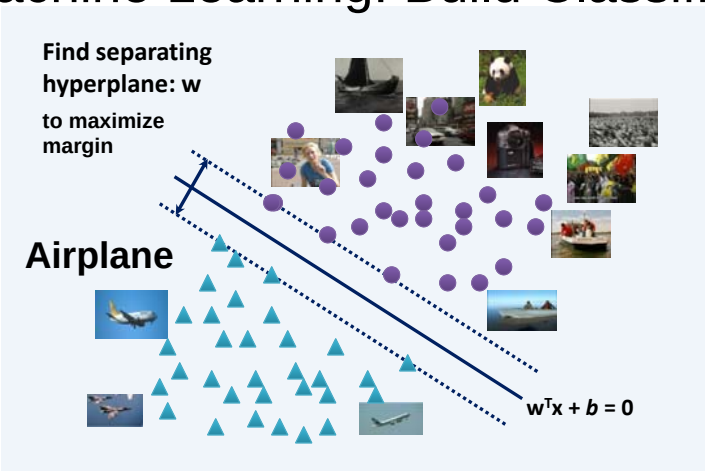
43

# Classifiers

- K-Nearest Neighbors + Voting
- Linear Discriminative Model (SVM)

44

# Machine Learning: Build Classifier




Decision function: $f(x) = \text{sign}(w^T x + b)$

$w^T x_i + b > 0$ if label $y_i = +1$

$w^T x_i + b < 0$ if label $y_i = -1$

# Support Vector Machine (tutorial by Burges '98)

- **Look for separation plane with the highest margin**

  *Decision boundary*

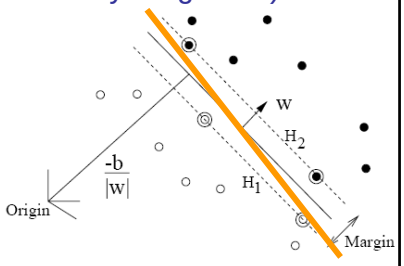
  $H_0: \mathbf{w}^t \mathbf{x} + b = 0$

- **Linearly separable**

  $w^T x_i + b > 1$ if label $y_i = +1$

  $w^T x_i + b < -1$ if label $y_i = -1$

  $y_i (w^T x_i + b) > 1$ for all $x_i$




- **Two parallel hyperplanes defining the margin**

  hyperplane $H_1 (H_+): \mathbf{w}^t \mathbf{x}_i + b = +1$

  hyperplane $H_2 (H_-): \mathbf{w}^t \mathbf{x}_i + b = -1$

- **Margin: sum of distances of the closest points to the separation plane**

  $\text{margin} = 2 / \|\mathbf{w}\|$

- **Best plane defined by w and b**

## Max Margin Solution for separable case

$$\frac{\partial}{\partial w_\nu} L_P = w_\nu - \sum_i \alpha_i y_i x_{i\nu} = 0 \quad \nu = 1, \cdots, d \longrightarrow W^* = \sum \alpha_i y_i x_i$$

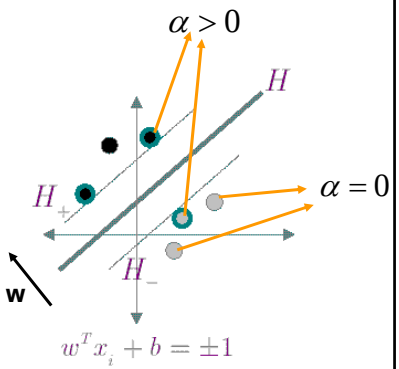
$$\frac{\partial}{\partial b} L_P = -\sum_i \alpha_i y_i = 0$$

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0 \quad i = 1, \cdots, l$$

$$\alpha_i \geq 0 \quad \forall i$$

$$\alpha_i(y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1) = 0 \quad \forall i$$

- **Weight sum from positive class = Weight sum from negative class**
- **Direction of w: roughly from negative support vectors to positive ones**




if $a_i > 0$, $\mathbf{x}_i$ is on $H_+$ or $H_-$ and is a support vector

- **How to compute w and b?**
- **How to classify new data?**

## Non-separable

- **Add slack variables** $x_i$

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 - \xi_i \quad \text{for } y_i = +1$$

$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 + \xi_i \quad \text{for } y_i = -1$$

$$\xi_i \geq 0 \ \forall i.$$

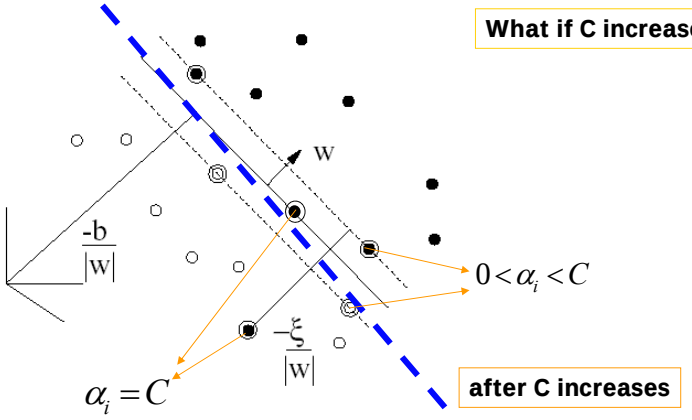
if $\xi_i > 1$, then $\mathbf{x}_i$ is misclassified (i.e. training error)

Lagrange multiplier: minimize

$$L_P = \underbrace{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i}_{\text{New objective function}} - \sum_i \alpha_i\{y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i\} - \sum_i \mu_i \xi_i$$

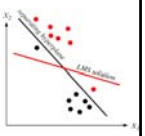
Ensure positivity

- **All the points located in the margin gap or the wrong side will get** $\alpha_i = C$




- **When C increases, samples with errors get more weights**
  - **better training accuracy, but smaller margin**
  - **less generalization performance**

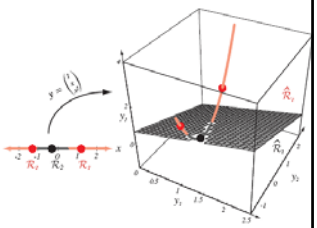
## Generalized Linear Discriminant Functions



- **Include more than just the linear terms**

$$g(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d}\sum_{j=1}^{d} w_{ij} x_i x_j = w_0 + \mathbf{w}^t\mathbf{x} + \mathbf{x}^t\mathbf{W}\mathbf{x}$$

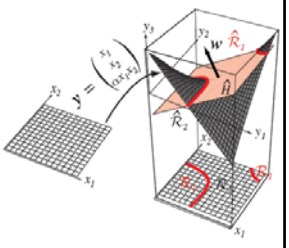
- **Shape of decision boundary**
  - **ellipsoid, hyperhyperboloid, lines etc**
- **In general** $g(\mathbf{x}) = \sum_{i=1}^{\hat{d}} a_i y_i(\mathbf{x}) = \mathbf{a}^t\mathbf{y}$



- **Example**

$$g(x) = a_1 + a_2 x + a_3 x^2 = [a_1 \quad a_2 \quad a_3]\begin{bmatrix}1\\ x\\ x^2\end{bmatrix}$$

$$g(x) = a_1 x_1 + a_2 x_2 + a_3 x_1 x_2 = [a_1 \quad a_2 \quad a_3]\begin{bmatrix}1\\ x_1\\ x_1 x_2\end{bmatrix}$$



- **Data become separable in higher-dimensional space**
  - **learning parameters in high dimension is hard (curse of dim.)**
  - **instead, try to maximize margins → SVM**

Figure from Duda, Hart, and Stork

# Non-Linear Space

$\Phi : \mathbb{R}^d \mapsto \mathcal{H}$. Map to a high dimensional space, to make the data separable $\Phi(\mathbf{x}) = \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix}$

- **Find the SVM in the high-dim space (embedding space)**

$$g(\mathbf{x}) = \sum_{i=1}^{N_s} a_i y_i \Phi(\mathbf{s}_i) \times \Phi(\mathbf{x}) + b$$

$\sum_{i=1}^{N_s} a_i y_i \Phi(\mathbf{s}_i)$ = **w**

- **Luckily, we don't have to find** $\Phi(\mathbf{s}_i)$ nor $\sum_{i=1}^{l} a_i y_i \Phi(\mathbf{s}_i)$
- **Instead, we define kernel** $K(\mathbf{s}_i, \mathbf{x}) = \Phi(\mathbf{s}_i) \times \Phi(\mathbf{x})$

$$\text{Þ } g(\mathbf{x}) = \sum_{i=1}^{N_s} a_i y_i K(\mathbf{s}_i, \mathbf{x}) + b$$

- **We can use the same method to maximize $L_D$ to find** $a_i$

$$L_D = \sum_{i=1}^{l} a_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} a_i a_j y_i y_j \Phi(\mathbf{x}_i) \times \Phi(\mathbf{x}_j)$$

$$= \sum_{i=1}^{l} a_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} a_i a_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

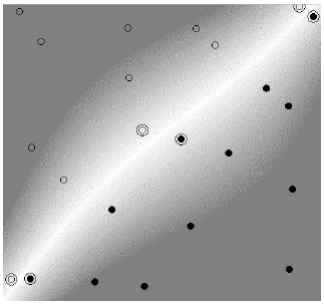
# Some popular kernels

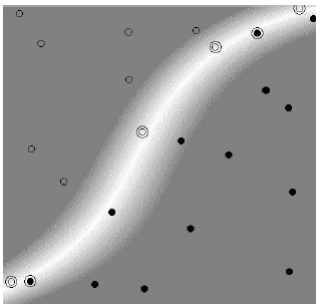
$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p$ **polynomial**

$K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2}$ **Gaussian Radial Basis Function (RBF)**

$K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x} \cdot \mathbf{y} - \delta)$ **sigmoidal neural network**

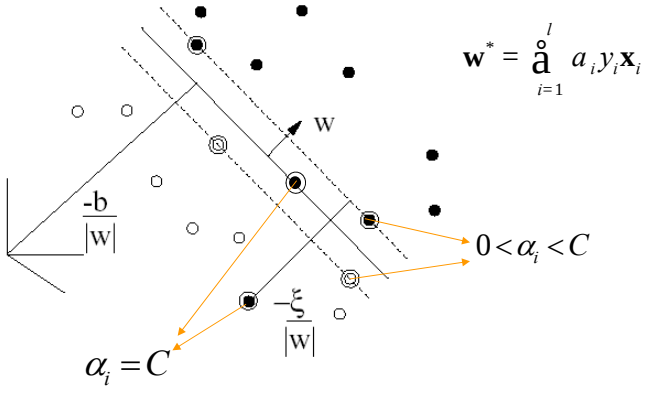
separable — Cubic polynomial — non-separable





- **SVM Classifier is completely determined by the training samples that are on the hyperplanes or within the margin**

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) <= 1$$




# Reading List


- Lazebnik, S., C. Schmid, and J. Ponce. *Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. in IEEE CVPR. 2006.*
- Jiang, Y., C. Ngo, and J. Yang. *Towards optimal bag-of-features for object categorization and semantic video retrieval. in ACM CIVR. 2007.*
- Chang, S., et al. *Columbia University/VIREO-CityU/IRIT TRECVID2008 high-level feature extraction and interactive video search. in NIST TRECVID Workshop. 2008.*
- Jiang, Y., et al. *Columbia-UCF TRECVID2010 Multimedia Event Detection: Combining Multiple Modalities, Contextual Concepts, and Temporal Matching. in TRECVID Workshop. 2010.*
- *Pattern Classification*, 2nd ed., Richard O. Duda, Peter E. Hart, and David G. Stork ISBN: 0-471-05669-3, 2000, Wiley
- Viola, P. and M. Jones. *Rapid Object Detection using a Boosted Cascade of Simple Features. in Proceedings IEEE Conf. on Computer Vision and Pattern Recognition. 2001.*
- Yan, R., J. Yang, and A.G. Hauptmann. *Learning Query-Class Dependent Weights in Automatic Video Retrieval. in ACM Multimedia. 2004. New York.*