Exam Review

Ranjay Krishna

Basic Exam Facts

Exam

2015.12.07

3:30pm-6:30pm

Mudd Chemistry Building LEC

Note: one single-sided 8.5x11" sheet of notes is allowed in the final.

Exam Layout

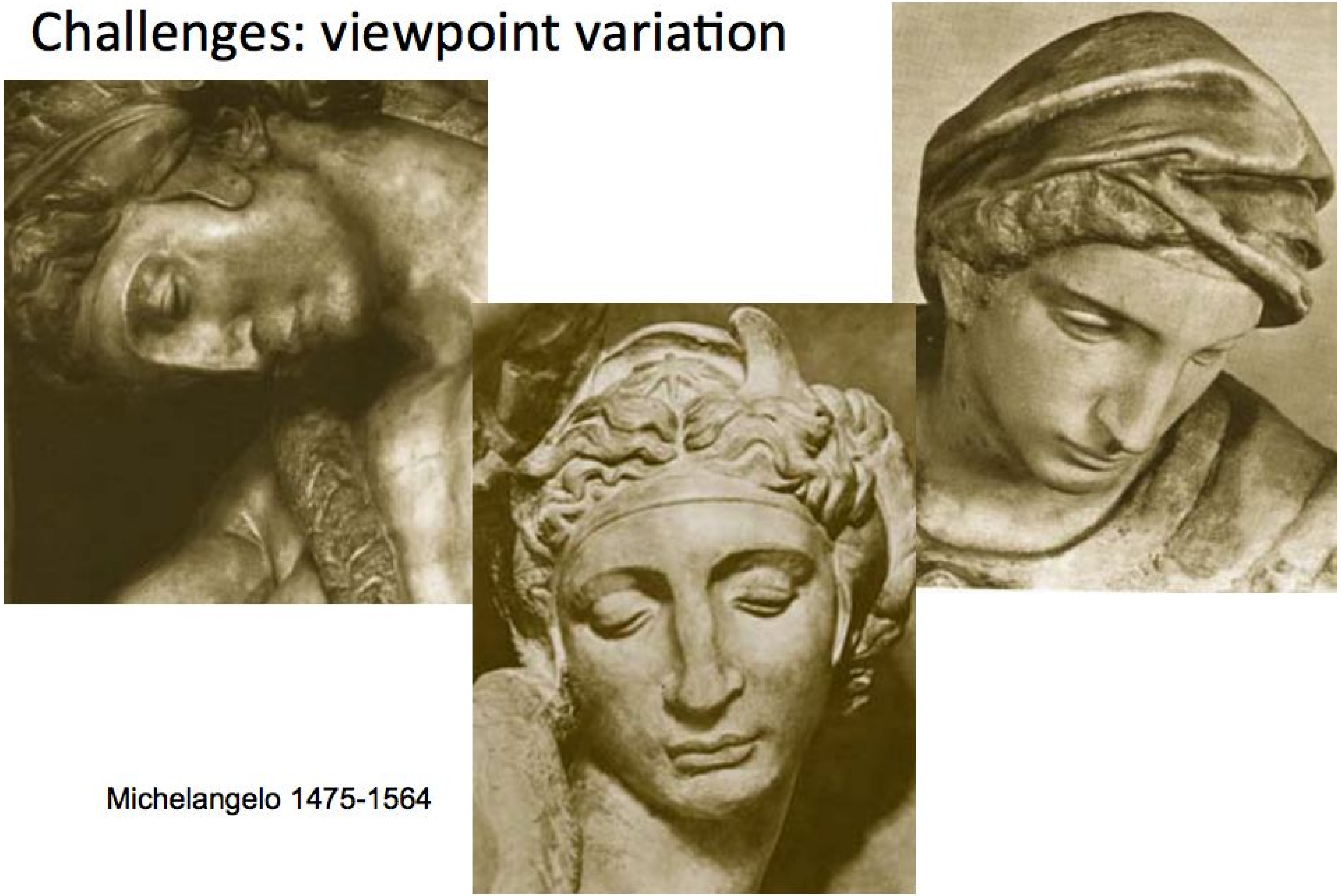
15 True False Questions (30 minutes)

15 Multiple Choice Questions (30 minutes)

4 Short Answer Questions with 4 sub parts each. (90 minutes)

Recognizing Faces and Objects

Challenges: viewpoint variation



Michelangelo 1475-1564

Challenges: illumination

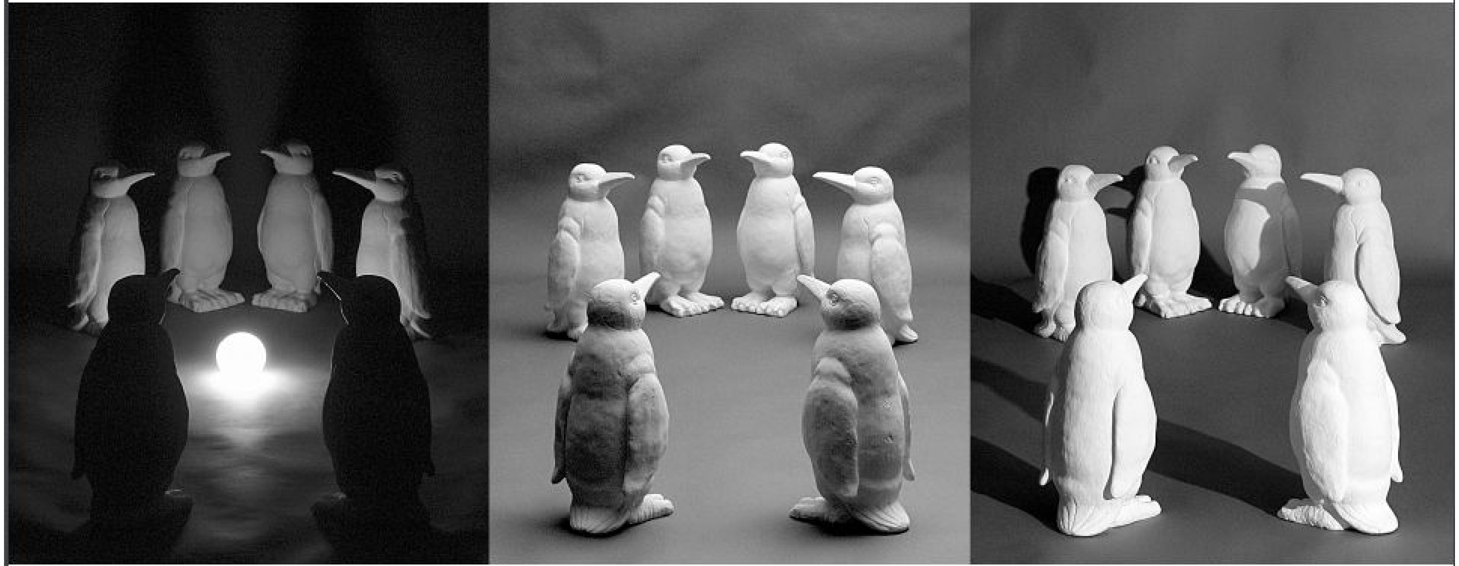


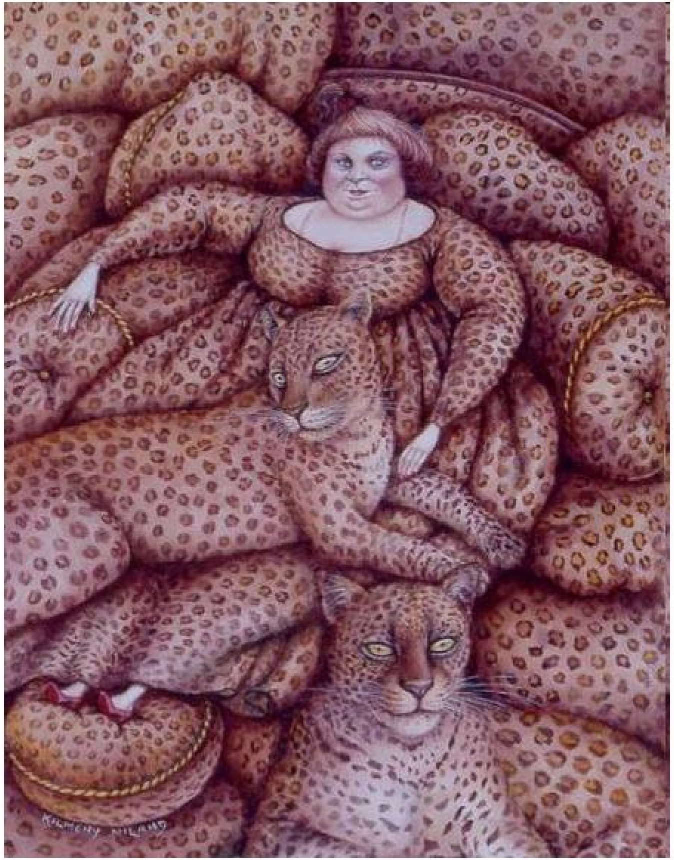
image credit: J. Koenderink

Challenges: scale

Challenges: occlusion

Magritte, 1957

Challenges: background clutter

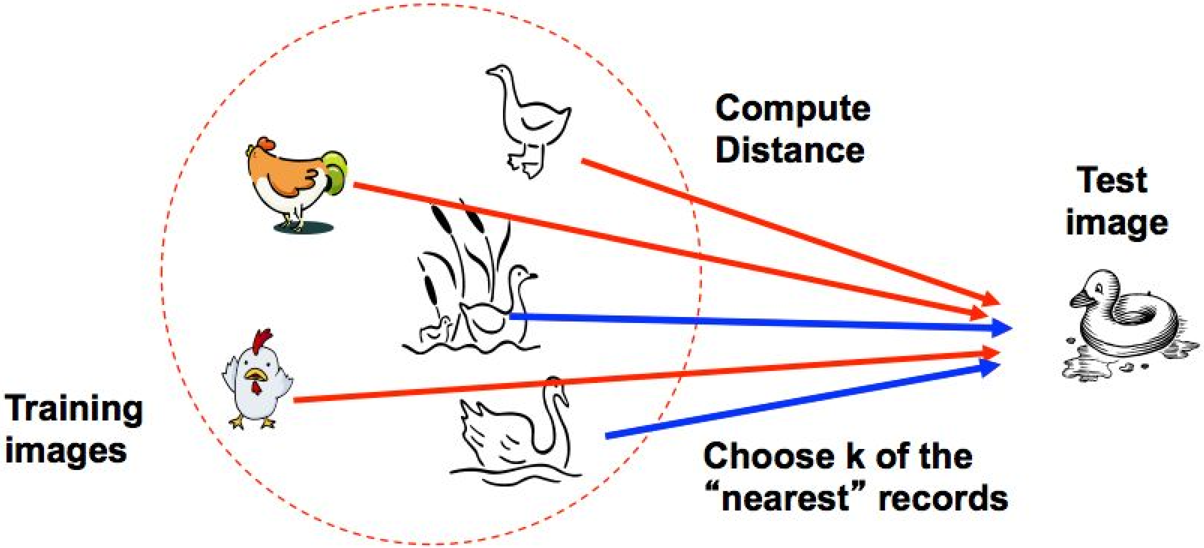


Kilmeny Niland. 1995

Challenges: intra-class variation

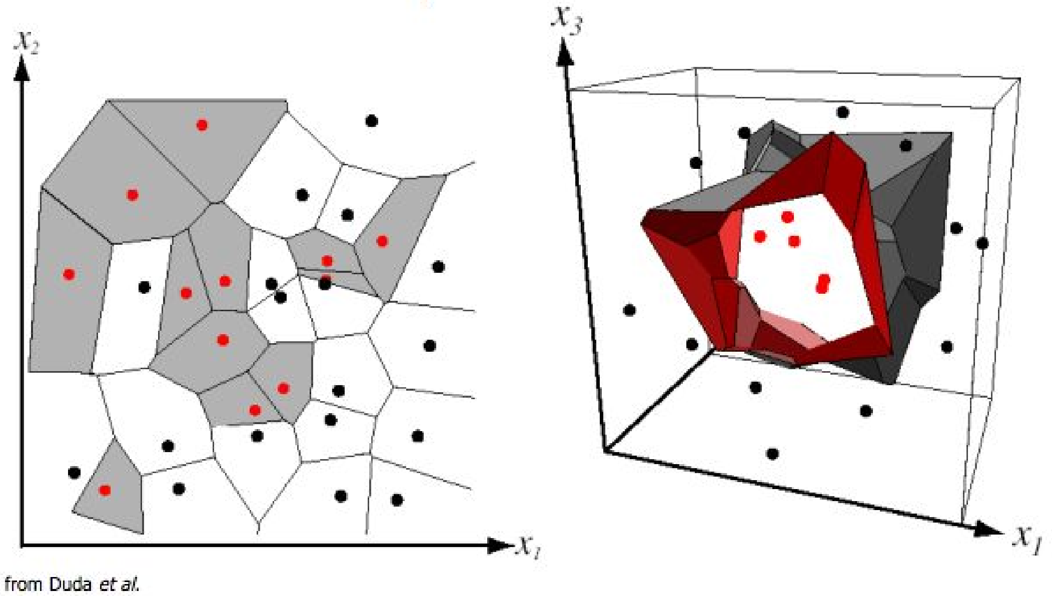
Nearest Neighbor Classifier

- Assign label of nearest training data point to each test data point



Source: N. Goyal

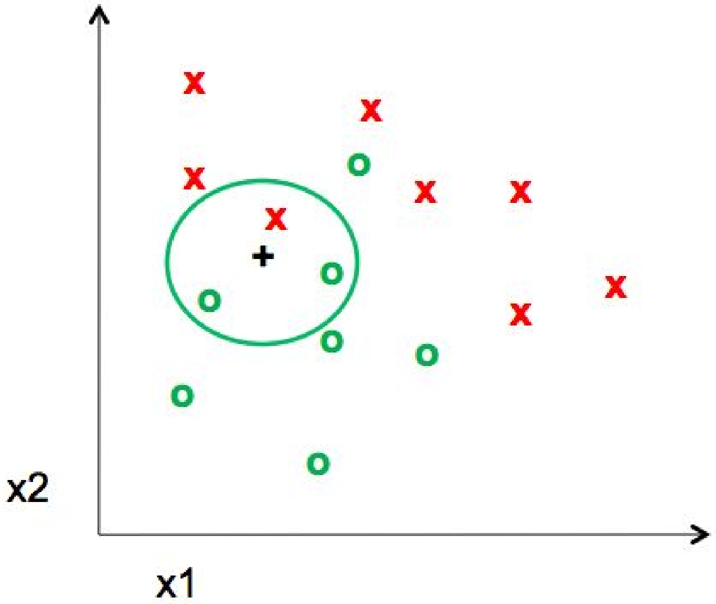
- Assign label of nearest training data point to each test data point



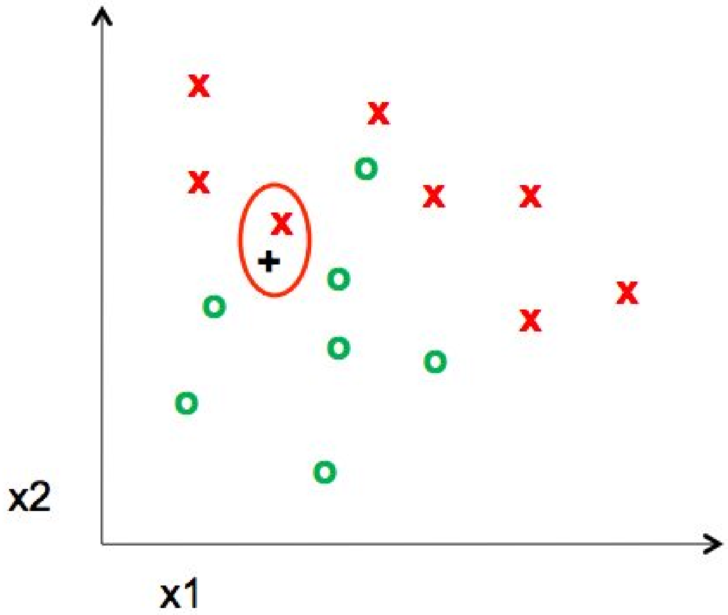
partitioning of feature space
for two-category 2D and 3D data

Source: D. Lowe

Overfitting vs. Underfitting

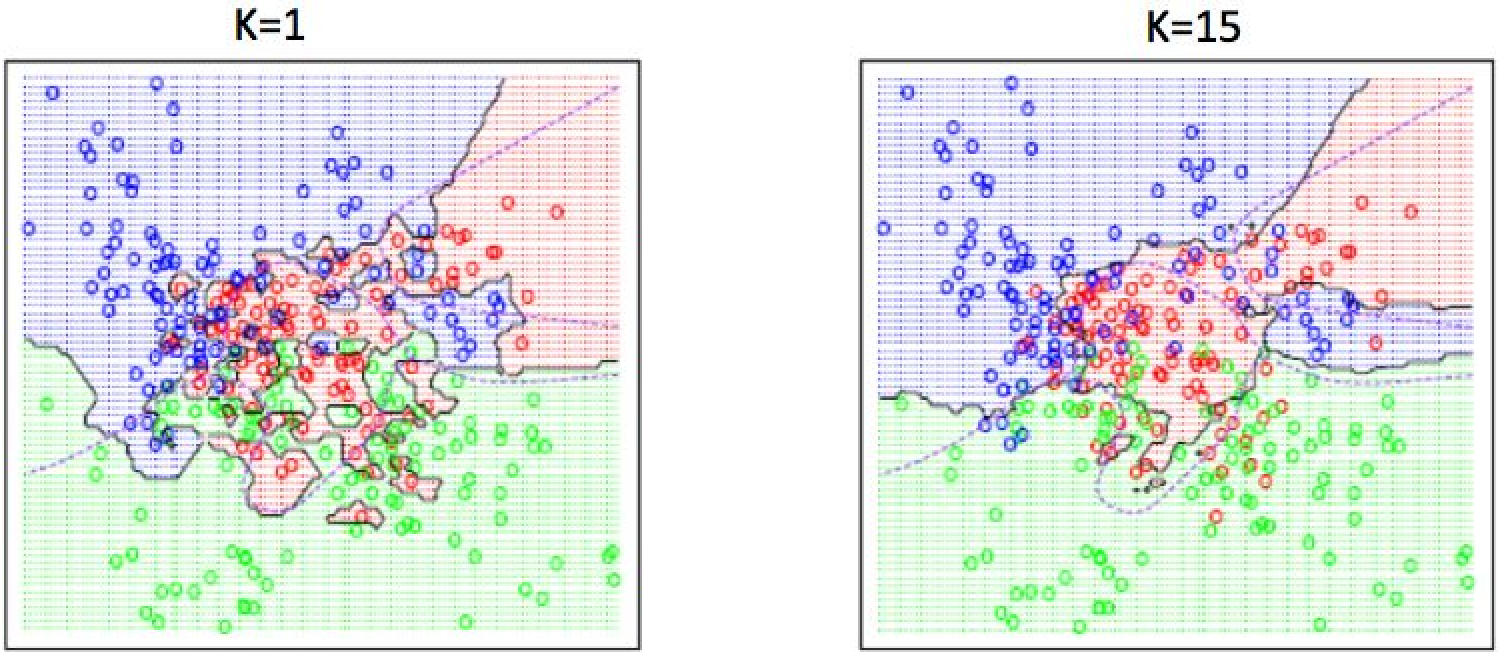


K = 1



K = 3

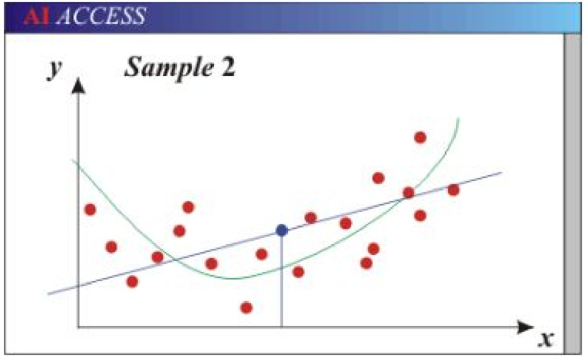
Decision Boundaries

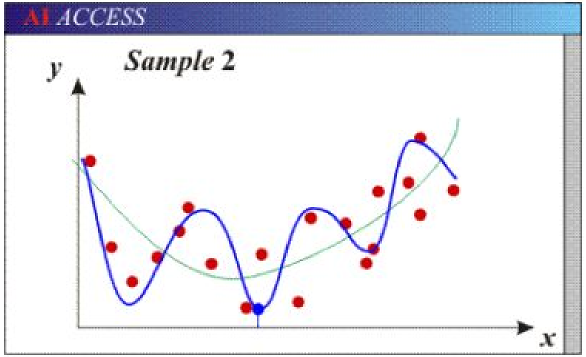


Other Issues

- Dimension of features
- Normalization?
- Size of training dataset

Bias-Variance Trade-off





- Models with too few parameters are inaccurate because of a large bias (not enough flexibility).
- Models with too many parameters are inaccurate because of a large variance (too much sensitivity to the sample).

Target Variance

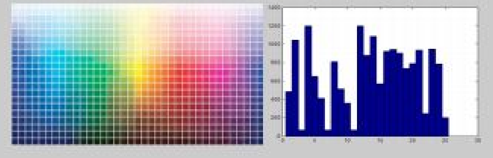
- More data
- Regularize

Target Bias??

Image features

Input image

Color: Quantize RGB values

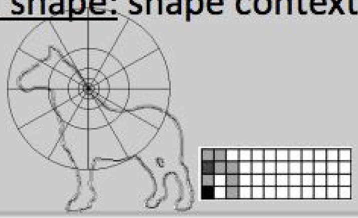


Invariance?
- Translation
- Scale
- Rotation
- Occlusion

Global shape: PCA space

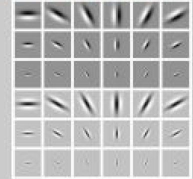
Invariance?
- Translation
- ? Scale
- Rotation
- Occlusion

Local shape: shape context



Invariance?
- Translation
- Scale
- ? Rotation (in-planar)
- Occlusion

Texture: Filter banks



Invariance?
- Translation
- ? Scale
- ? Rotation (in-planar)
- Occlusion

technical note

PCA Algorithm (training)

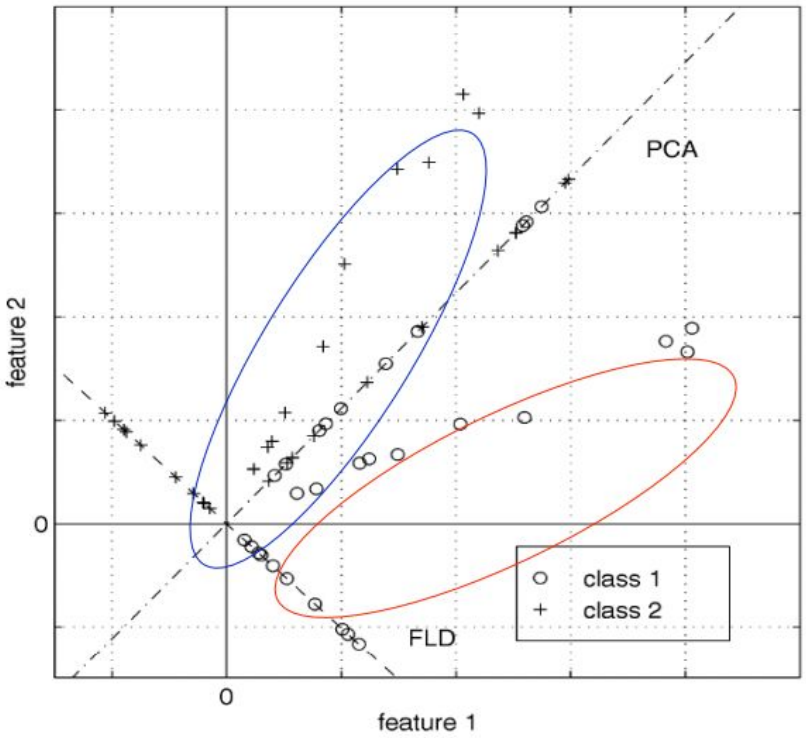
- Given sample $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}, \; x_i \in \mathcal{R}^d$
  - compute sample mean: $\hat{\mu} = \frac{1}{n}\sum_i(\mathbf{x}_i)$
  - compute sample covariance: $\hat{\Sigma} = \frac{1}{n}\sum_i(\mathbf{x}_i - \hat{\mu})(\mathbf{x}_i - \hat{\mu})^T$
  - compute eigenvalues and eigenvectors of $\hat{\Sigma}$

$$\hat{\Sigma} = \Phi\Lambda\Phi^T, \;\; \Lambda = diag(\sigma_1^2, \ldots, \sigma_n^2) \;\; \Phi^T\Phi = I$$

  - order eigenvalues $\sigma_1^2 > \ldots > \sigma_n^2$
  - if, for a certain k, $\sigma_k << \sigma_1$ eliminate the eigenvalues and eigenvectors above k.

Slide inspired by N. Vasconcelos

Basic intuition: PCA vs. LDA



Linear Discriminant Analysis (LDA)

- We have two classes such that

$$E_{X|Y}[X \mid Y=i] = \mu_i$$

$$E_{X|Y}\left[(X-\mu_i)(X-\mu_i)^T \mid Y=i\right] = \Sigma_i$$

- We want to find the line z that best separates them

$$z = w^T x$$

- One possibility would be to maximize

$$\left(E_{Z|Y}[Z \mid Y=1] - E_{Z|Y}[Z \mid Y=0]\right)^2 =$$

$$\left(E_{X|Y}\left[w^T x \mid Y=1\right] - E_{X|Y}\left[w^T x \mid Y=0\right]\right)^2 = \left(w^T[\mu_1 - \mu_0]\right)^2$$

technical note

Linear Discriminant Analysis (LDA)

- However, this difference

$$\left(w^T\left[\mu_1 - \mu_0\right]\right)^2$$

can be arbitrarily large by simply scaling w

- We are only interested in the direction, not the magnitude
- Need some type of normalization
- Fisher suggested

$$\boxed{\begin{aligned} &\max \frac{between\ \ class\ \ scatter}{within\ \ class\ \ scatter} = \\ &\max_w \frac{\left(E_{Z|Y}[Z \mid Y=1] - E_{Z|Y}[Z \mid Y=0]\right)^2}{\mathrm{var}[Z \mid Y=1] + \mathrm{var}[Z \mid Y=0]} \end{aligned}}$$

Slide inspired by N. Vasconcelos

Linear Discriminant Analysis (LDA)

- We have already seen that

$$\left(E_{Z|Y}[Z \mid Y=1] - E_{Z|Y}[Z \mid Y=0]\right)^2 = \left(w^T[\mu_1 - \mu_0]\right)^2$$
$$= w^T[\mu_1 - \mu_0][\mu_1 - \mu_0]^T w$$

- also

$$\operatorname{var}[Z \mid Y=i] = E_{Z|Y}\left\{\left(z - E_{Z|Y}[Z \mid Y=i]\right)^2 \mid Y=i\right\}$$
$$= E_{Z|Y}\left\{\left(w^T[x-\mu_i]\right)^2 \mid Y=i\right\}$$
$$= E_{Z|Y}\left\{w^T[x-\mu_i][x-\mu_i]^T w \mid Y=i\right\}$$
$$= w^T \Sigma_i w$$

Slide inspired by N. Vasconcelos

Linear Discriminant Analysis (LDA)

- And

$$J(w) = \frac{\left(E_{Z|Y}[Z \mid Y=1] - E_{Z|Y}[Z \mid Y=0]\right)^2}{\mathrm{var}[Z \mid Y=1] + \mathrm{var}[Z \mid Y=0]}$$

$$= \frac{w^T(\mu_1 - \mu_0)(\mu_1 - \mu_0)^T w}{w^T(\Sigma_1 + \Sigma_0)w}$$

- which can be written as

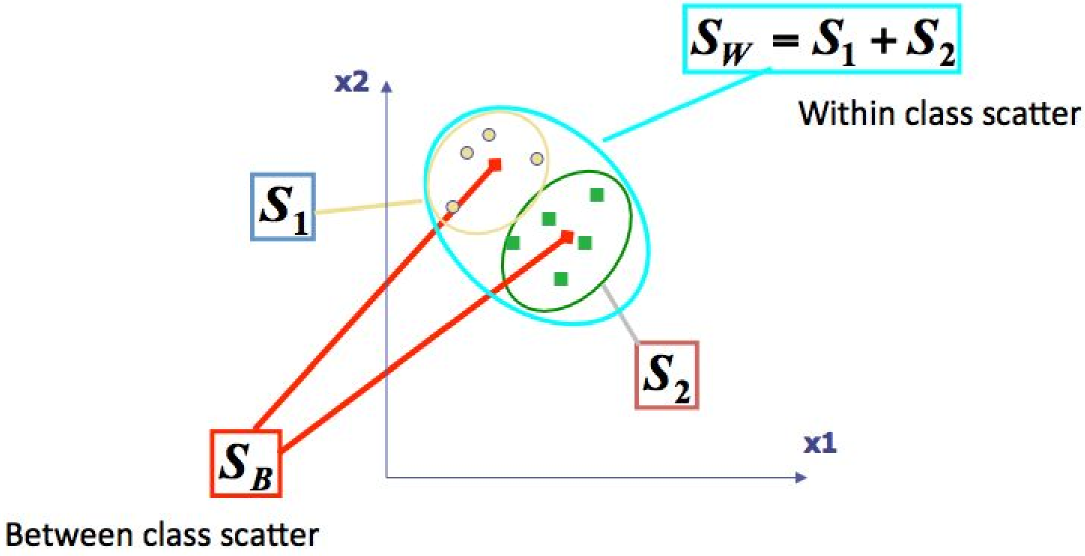
between class scatter

$$\boxed{J(w) = \frac{w^T S_B w}{w^T S_W w}} \qquad \boxed{\begin{aligned} S_B &= (\mu_1 - \mu_0)(\mu_1 - \mu_0)^T \\ S_W &= (\Sigma_1 + \Sigma_0) \end{aligned}}$$

within class scatter

Slide inspired by N. Vasconcelos

Visualization



technical note

Linear Discriminant Analysis (LDA)

- Maximizing the ratio

$$J(w) = \frac{w^T S_B w}{w^T S_W w}$$

  - Is equivalent to maximizing the numerator while keeping the denominator constant, i.e.

$$\boxed{\max_w w^T S_B w \quad \text{subject to} \quad w^T S_W w = K}$$

  - And can be accomplished using Lagrange multipliers, where we define the Lagrangian as

$$\boxed{L = w^T S_B w - \lambda\left(w^T S_W w - K\right)}$$

  - And maximize with respect to both w and λ

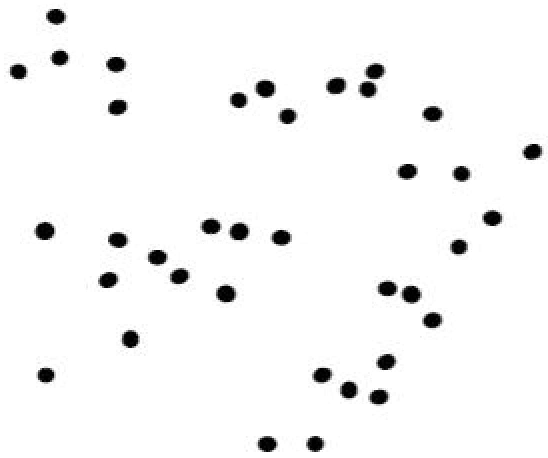
Slide inspired by N. Vasconcelos

Regions of Images, and Segmentation

Segmentation

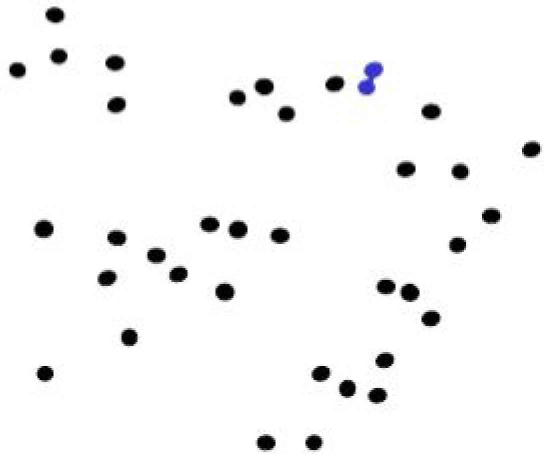
- Agglomerative clustering
  – Start with each point as its own cluster and iteratively merge the closest clusters
- K-means
  – Iteratively re-assign points to the nearest cluster center
- Mean-shift clustering
  – Estimate modes of pdf

Agglomerative clustering



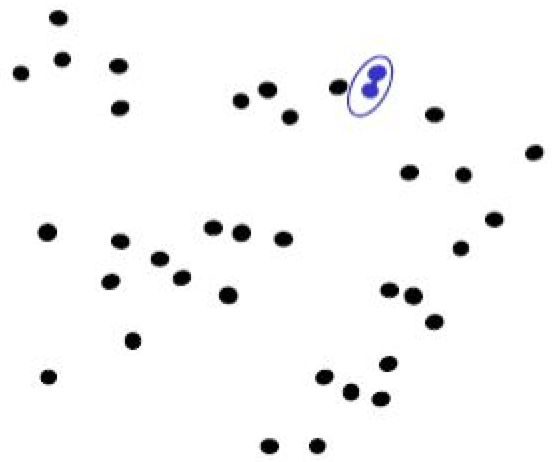
1. Say "Every point is its own cluster"

Agglomerative clustering



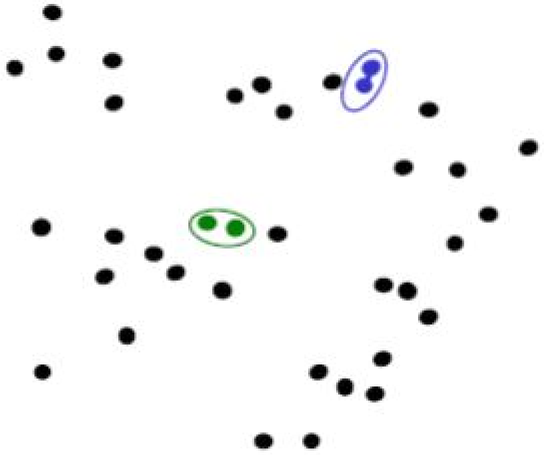
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters

Agglomerative clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster

Agglomerative clustering



1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

Good

- Simple to implement, widespread application
- Clusters have adaptive shapes
- Provides a hierarchy of clusters

Bad

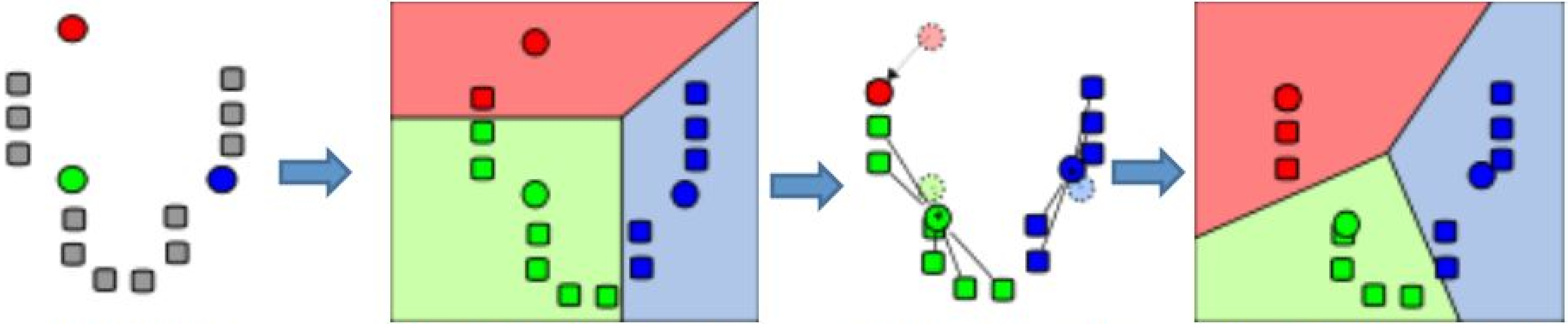
- May have imbalanced clusters
- Still have to choose number of clusters or threshold
- Need to use an "ultrametric" to get a meaningful hierarchy

You should be able to understand the clusters formed would be different with different definitions of what a cluster means.

How to define cluster similarity?

- Average distance between points,
- maximum distance
- minimum distance
- Distance between means or medoids

K-means clustering



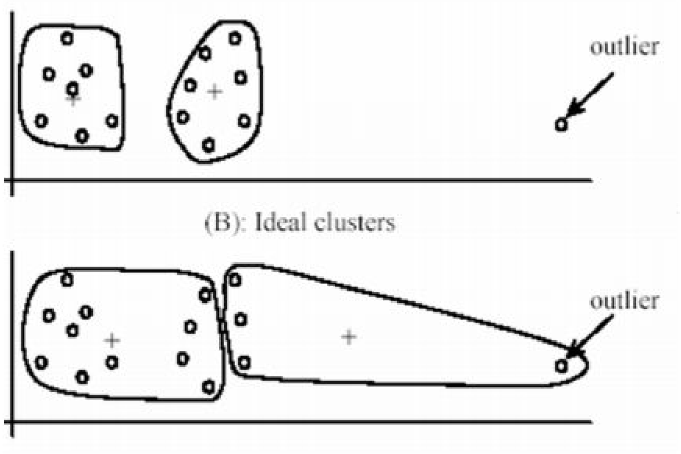
1. Initialize Cluster Centers | 2. Assign Points to Clusters | 3. Re-compute Means | Repeat (2) and (3)

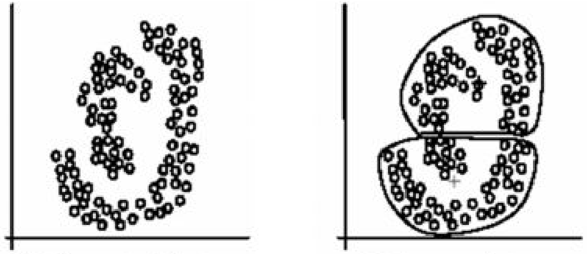
K-means Issues

- How do you pick the number of clusters?
- How do you prevent a bad local minima?
- How do you choose what features to use?
  Color or location or maybe something else?

K-Means pros and cons

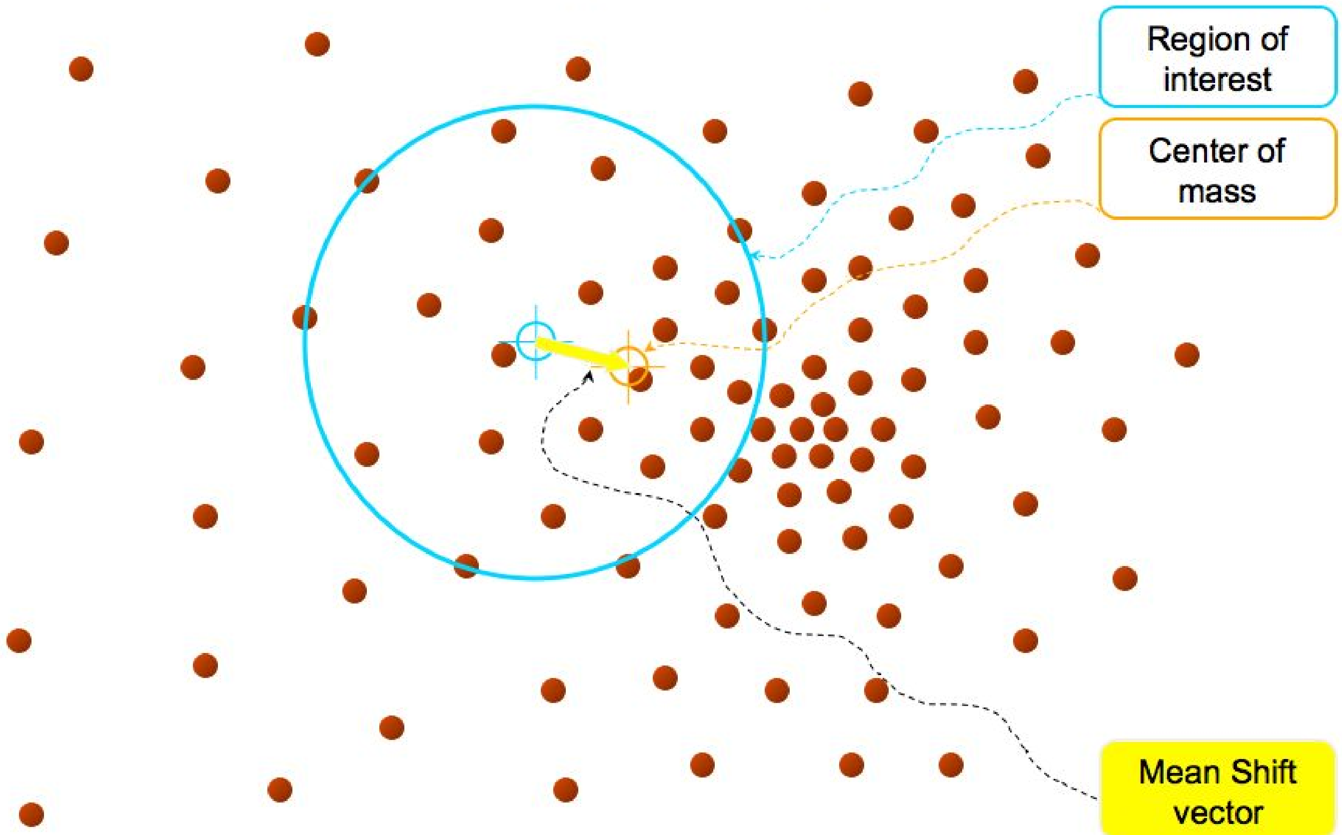
- Pros
  - Finds cluster centers that minimize conditional variance (good representation of data)
  - Simple and fast, Easy to implement
- Cons
  - Need to choose K
  - Sensitive to outliers
  - Prone to local minima
  - All clusters have the same parameters (e.g., distance measure is non-adaptive)
  - *Can be slow: each iteration is O(KNd) for N d-dimensional points
- Usage
  - Unsupervised clustering
  - Rarely used for pixel segmentation



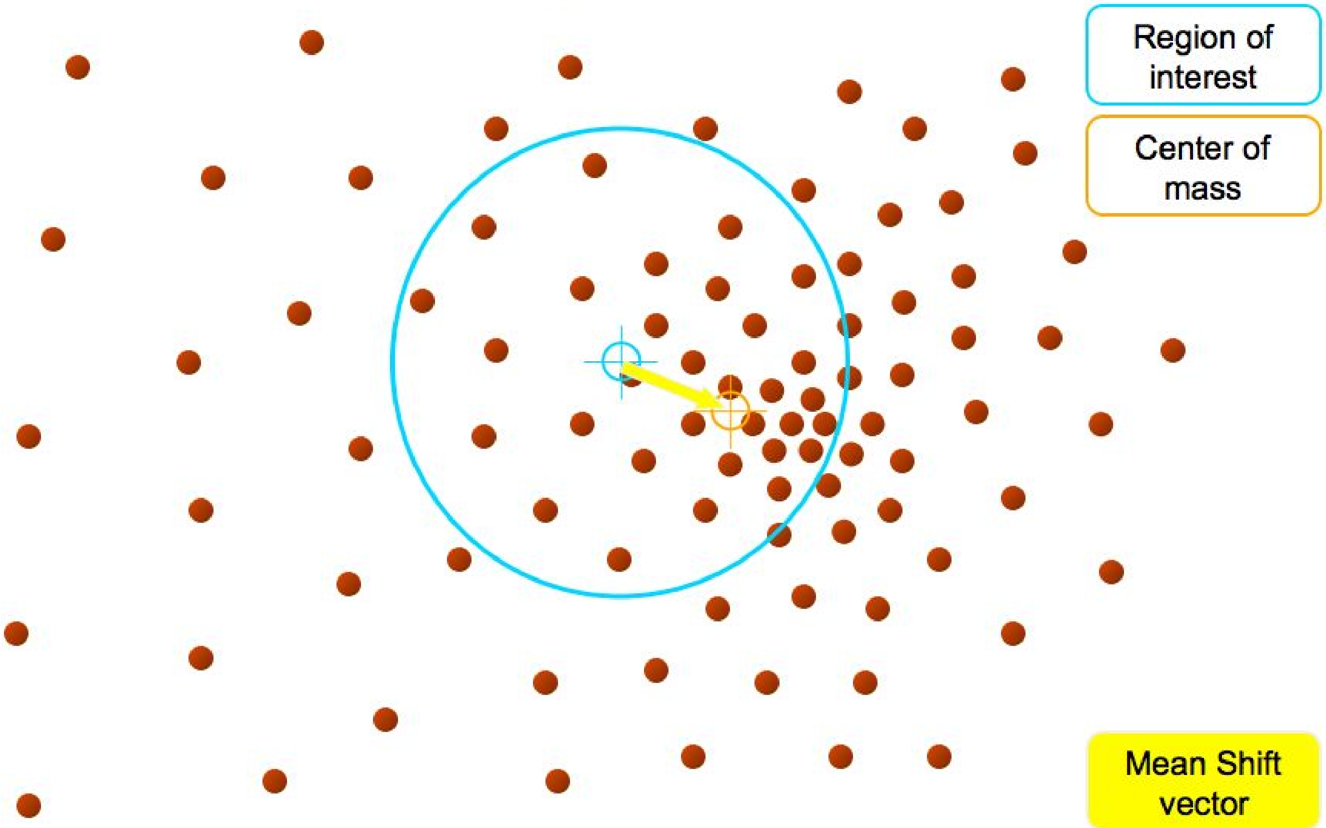


(A): Two natural clusters (B): k-means clusters

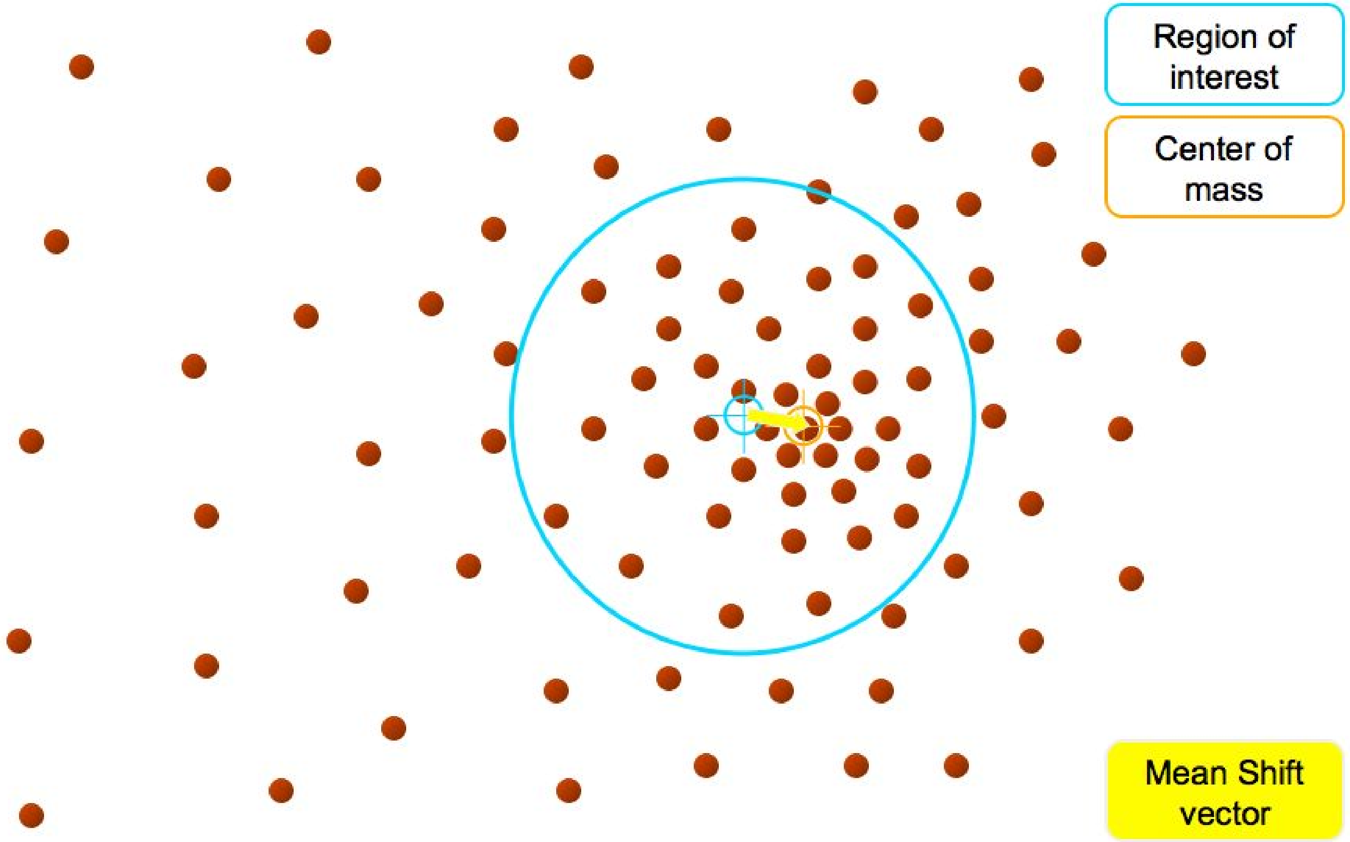
Mean-Shift



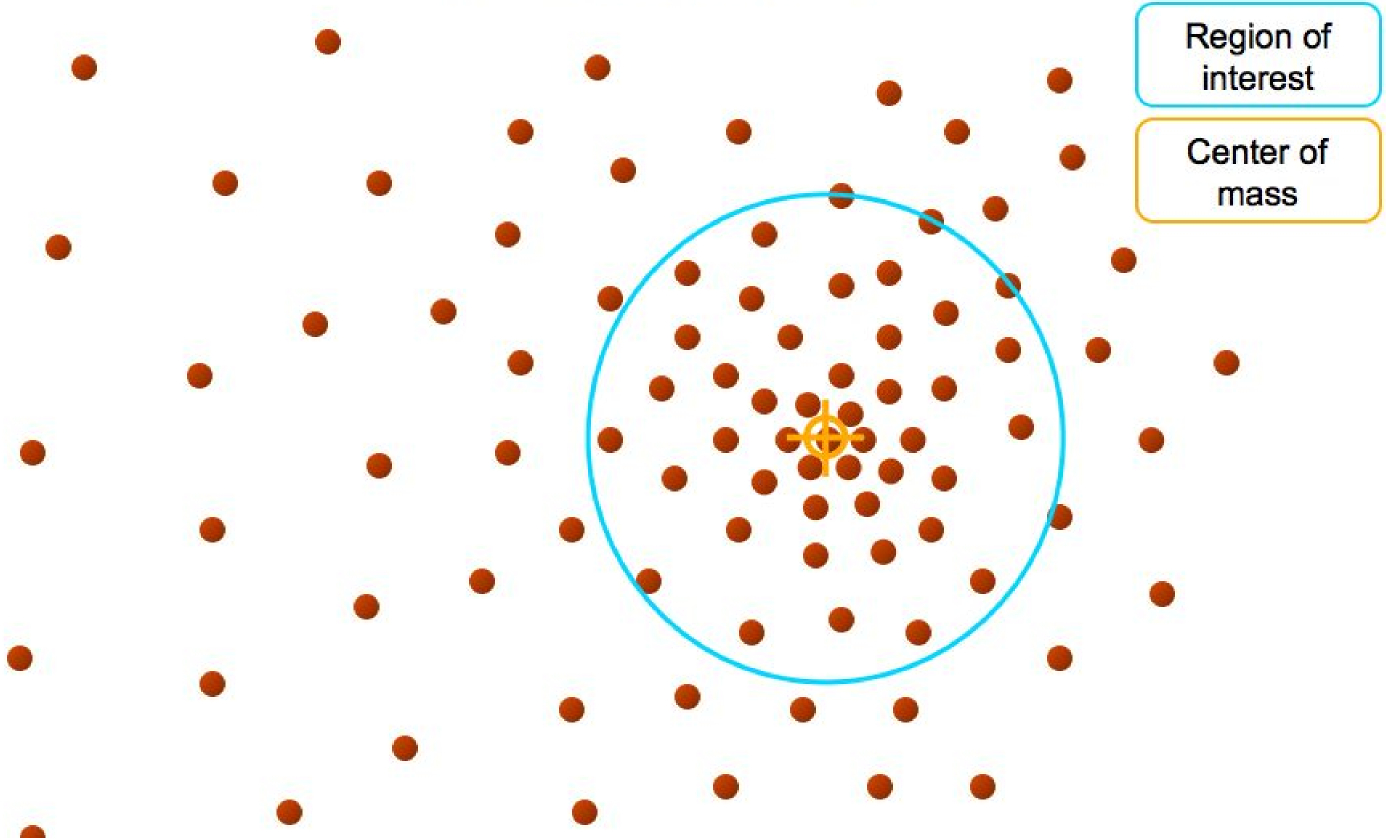
Mean-Shift



Mean-Shift



Mean-Shift

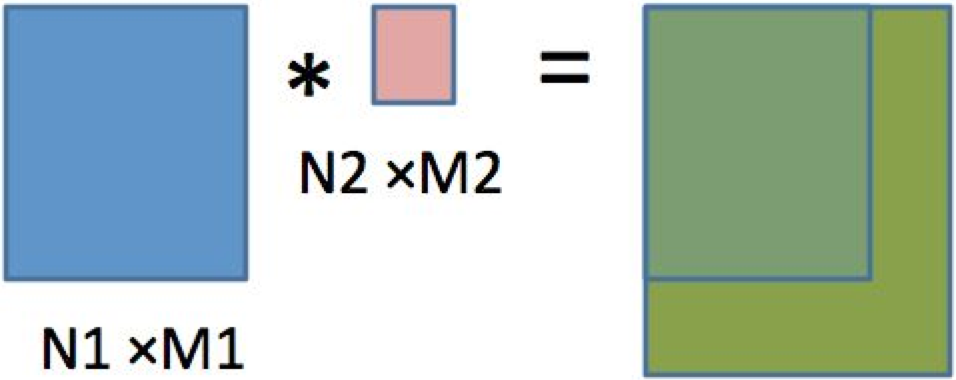


- Pros
  - General, application-independent tool
  - Model-free, does not assume any prior shape (spherical, elliptical, etc.) on data clusters
  - Just a single parameter (window size h)
    - h has a physical meaning (unlike k-means)
  - Finds variable number of modes
  - Robust to outliers
- Cons
  - Output depends on window size
  - Window size (bandwidth) selection is not trivial
  - Computationally (relatively) expensive (~2s/image)
  - Does not scale well with dimension of feature space

Pixels and Features

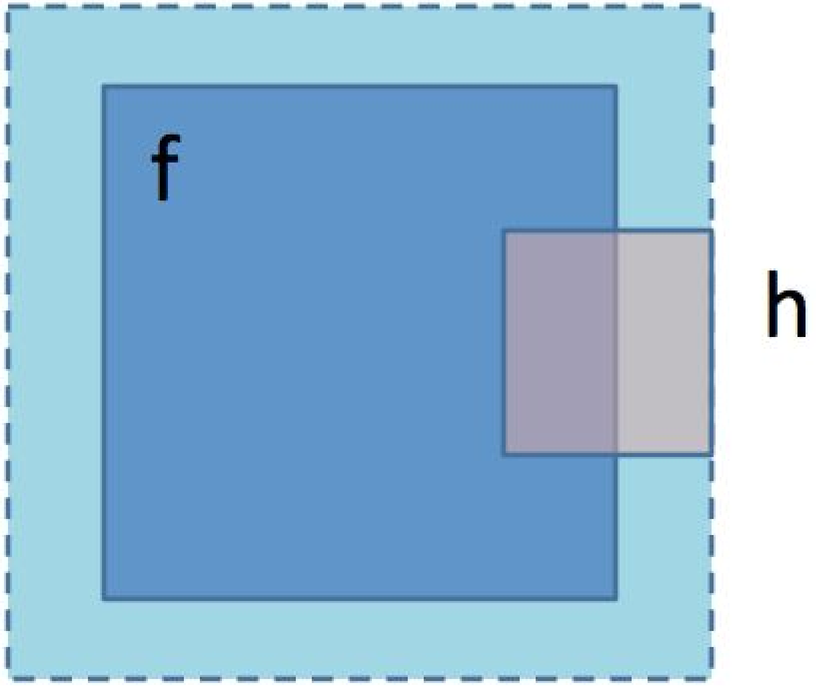
Convolution

$$(f * h)[m,n] = \sum_{k,l} f[k,l]\, h[m-k, n-l]$$



$(N1 + N2 - 1) \times (M1 + M2 - 1)$

Convolution



- zero "padding"
- edge value replication
- mirror extension
- more (beyond the scope of this class)

-> Matlab conv2 uses zero-padding

Moving Average

$$h[\cdot,\cdot]$$

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

Thresholding

$$g[n,m] = \begin{cases} 255, & f[n,m] > 100 \\ 0, & \text{otherwise.} \end{cases}$$

Linear Systems

S is a linear system (function) iff it *S satisfies*

$$\mathcal{S}[\alpha f_1 + \beta f_2] = \alpha\mathcal{S}[f_1] + \beta\mathcal{S}[f_2]$$

superposition property

LSI (linear *shift invariant*) systems

Impulse response

$$\delta_2[n, m] \rightarrow \boxed{\mathcal{S}} \rightarrow h[n, m]$$

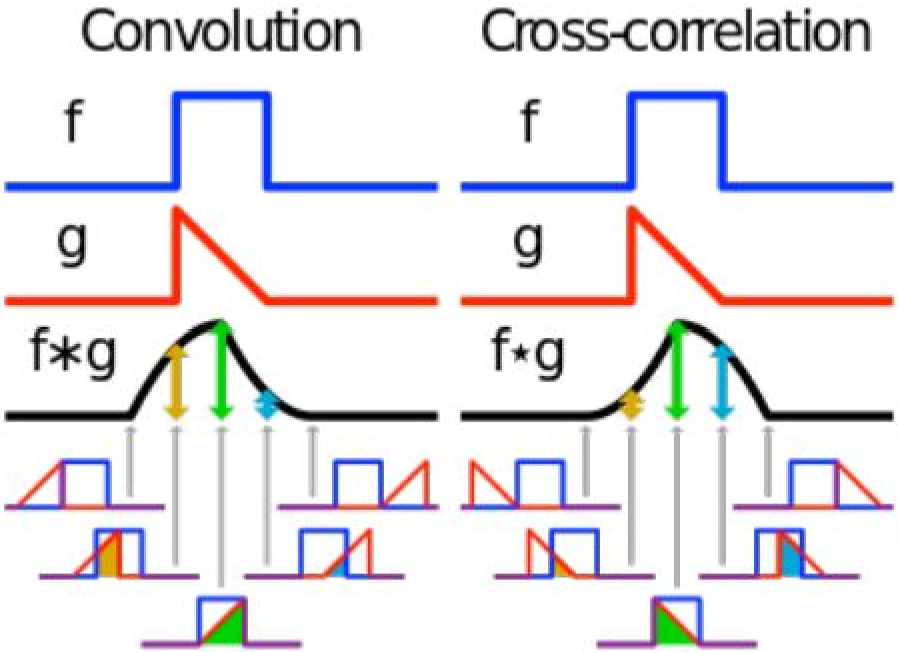
$$\delta_2[n-k, m-l] \rightarrow \boxed{\mathcal{S}\ (\mathrm{SI})} \rightarrow h[n-k, m-l]$$

Cross Correlation

$$r_{fg}[k,l] \triangleq \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} f[n,m]\, g^*[n-k, m-l]$$

$$= \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} f[n+k, m+l]\, g^*[n,m], \quad k,l \in \mathbb{Z}.$$

(k, l) is called the **lag**


Convolution
Cross-correlation
f
g
f∗g
f
g
f⋆g

properties

- **Commutative property:**

$$f ** h = h ** f$$

- **Associative property:**

$$(f ** h_1) ** h_2 = f ** (h_1 ** h_2)$$

- **Distributive property:**

$$f ** (h_1 + h_2) = (f ** h_1) + (f ** h_2)$$

The order doesn't matter! $h_1 ** h_2 = h_2 ** h_1$

- **Shift property:**

$$f[n, m] ** \delta_2[n - n_0, m - m_0] = f[n - n_0, m - m_0]$$

- **Shift-invariance:**

$$\begin{aligned} g[n, m] &= f[n, m] ** h[n, m] \\ \Longrightarrow & f[n - l_1, m - l_1] ** h[n - l_2, m - l_2] \\ &= g[n - l_1 - l_2, m - l_1 - l_2] \end{aligned}$$

Edge Detection

ow does Canny Edge Detector work?

RANSAC [Fischler & Bolles 1981]

RANSAC loop:

1. Randomly select a *seed group* of points on which to base transformation estimate (e.g., a group of matches)
2. Compute transformation from seed group
3. Find *inliers* to this transformation
4. If the number of inliers is sufficiently large, re-compute least-squares estimate of transformation on all of the inliers

- Keep the transformation with the largest number of inliers

RANSAC: How many samples?

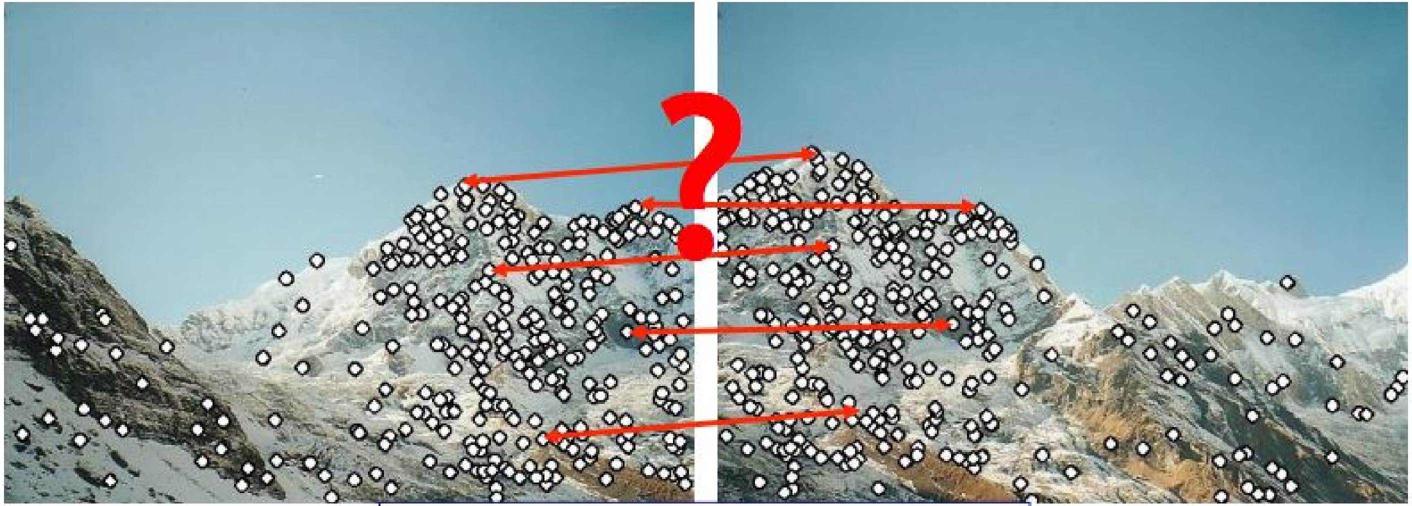
- How many samples are needed?
  - Suppose w is fraction of inliers (points from line).
  - n points needed to define hypothesis (2 for lines)
  - k samples chosen.
- Prob. that a single sample of n points is correct: w^n
- Prob. that all k samples fail is: $(1-w^n)^k$

$\Rightarrow$ Choose k high enough to keep this below desired failure rate.

RANSAC: Pros and Cons

- **Pros:**
  - General method suited for a wide range of model fitting problems
  - Easy to implement and easy to calculate its failure rate
- **Cons:**
  - Only handles a moderate percentage of outliers without cost blowing up
  - Many real problems have high rate of outliers (but sometimes selective choice of random subsets can help)

Image Matching



Point descriptor should be:

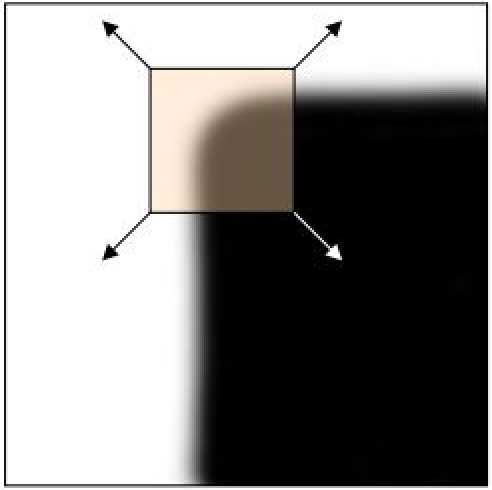
1. Invariant
2. Distinctive

Image Matching

- Region extraction needs to be repeatable and accurate
  - Invariant to translation, rotation, scale changes
  - Robust or covariant to out-of-plane (≈affine) transformations
  - Robust to lighting variations, noise, blur, quantization
- Locality: Features are local, therefore robust to occlusion and clutter.
- Quantity: We need a sufficient number of regions to cover the object.
- Distinctivenes : The regions should contain "interesting" structure.
- Efficiency: Close to real-time performance.

Harris Detector

- Translation invariance
- Rotation invariance
- Scale invariance?



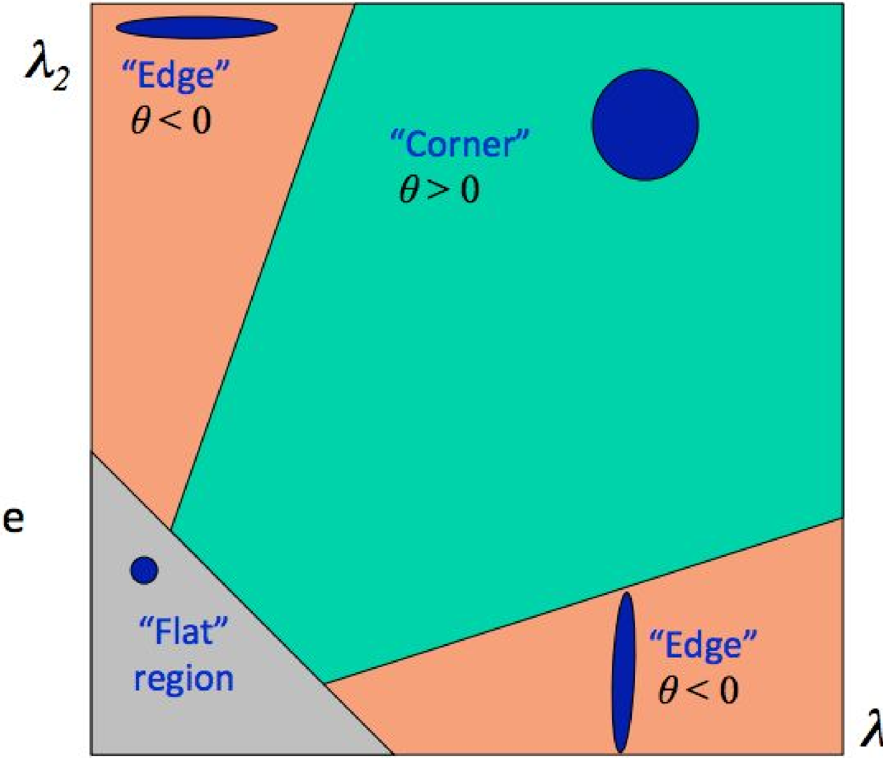
"corner":
significant change
in all directions

Harris Detector

Compute second moment matri
(autocorrelation matrix)

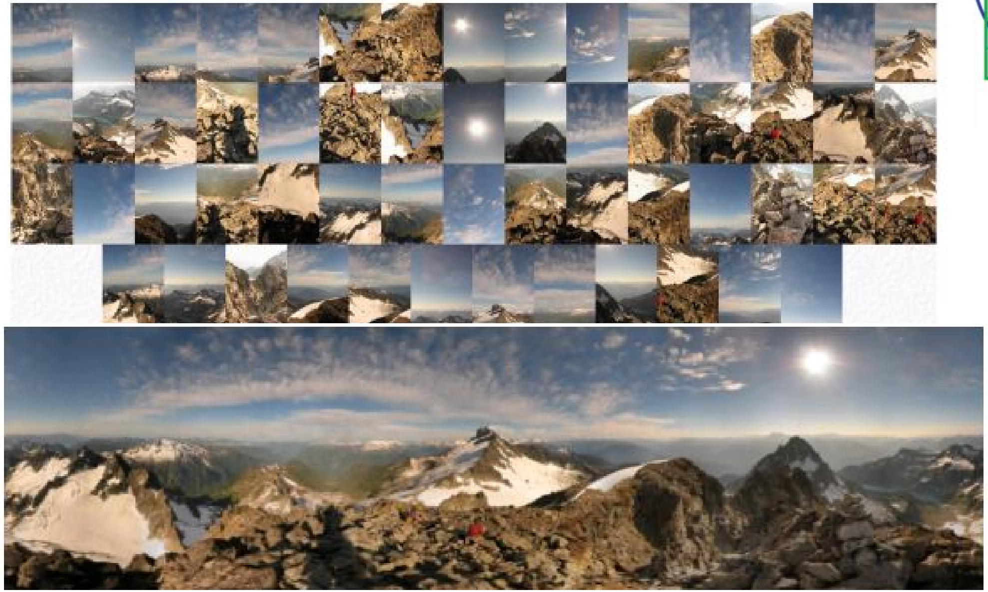
$$M(\sigma_I,\sigma_D)=g(\sigma_I)*\begin{bmatrix} I_x^2(\sigma_D) & I_xI_y(\sigma_D) \\ I_xI_y(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix}$$

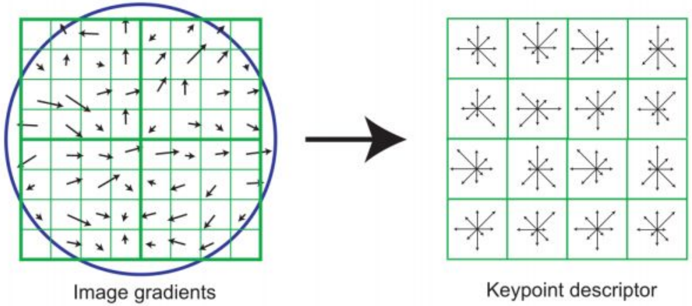
e



SIFT

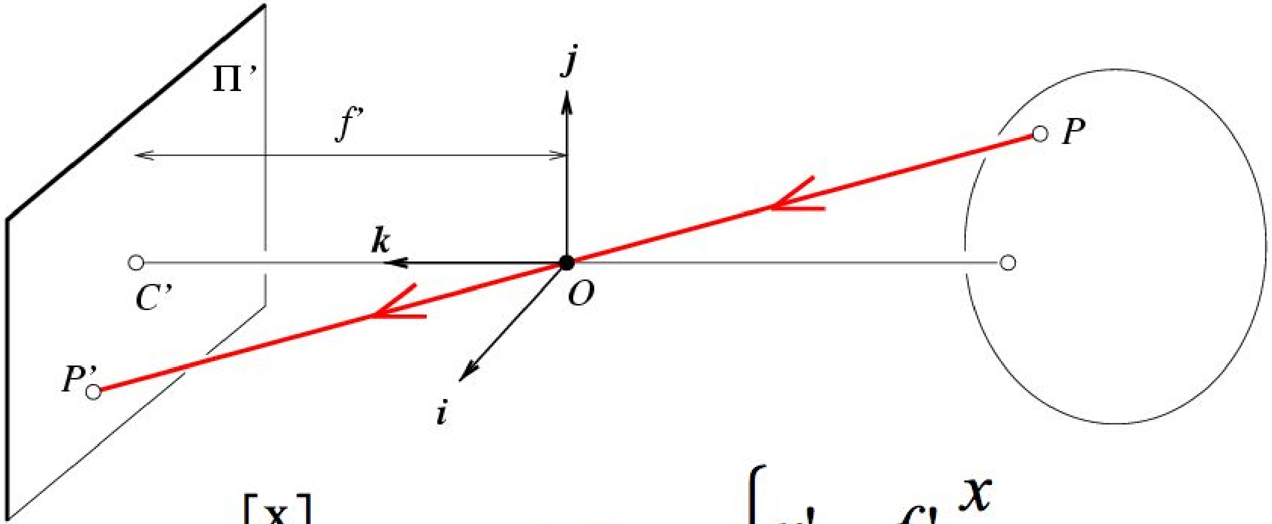
Scale Invariant





Camera

Pinhole camera



$$P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \rightarrow P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$\begin{cases} x' = f' \dfrac{x}{z} \\ y' = f' \dfrac{y}{z} \end{cases}$$

Note: z is always negative.

Derived using similar triangles

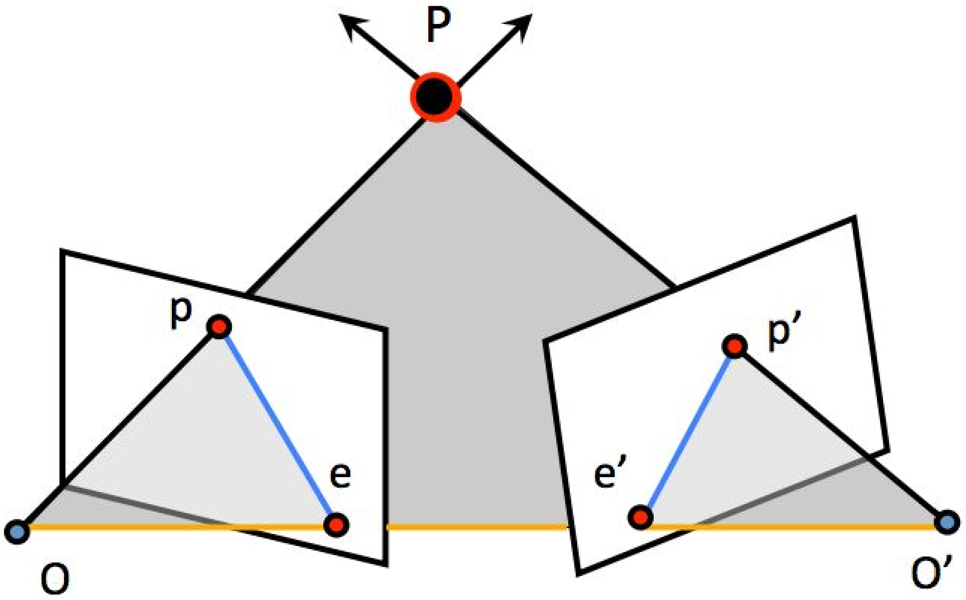
A generic projection matrix

Intrinsic Assumptions
- Optical center at (u_0, v_0)
- Rectangular pixels
- Small skew

Extrinsic Assumptions
- Allow rotation
- Camera at (t_x, t_y, t_z)

$$P' = K\begin{bmatrix} R & \vec{t} \end{bmatrix} P \Rightarrow w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Epipolar geometry



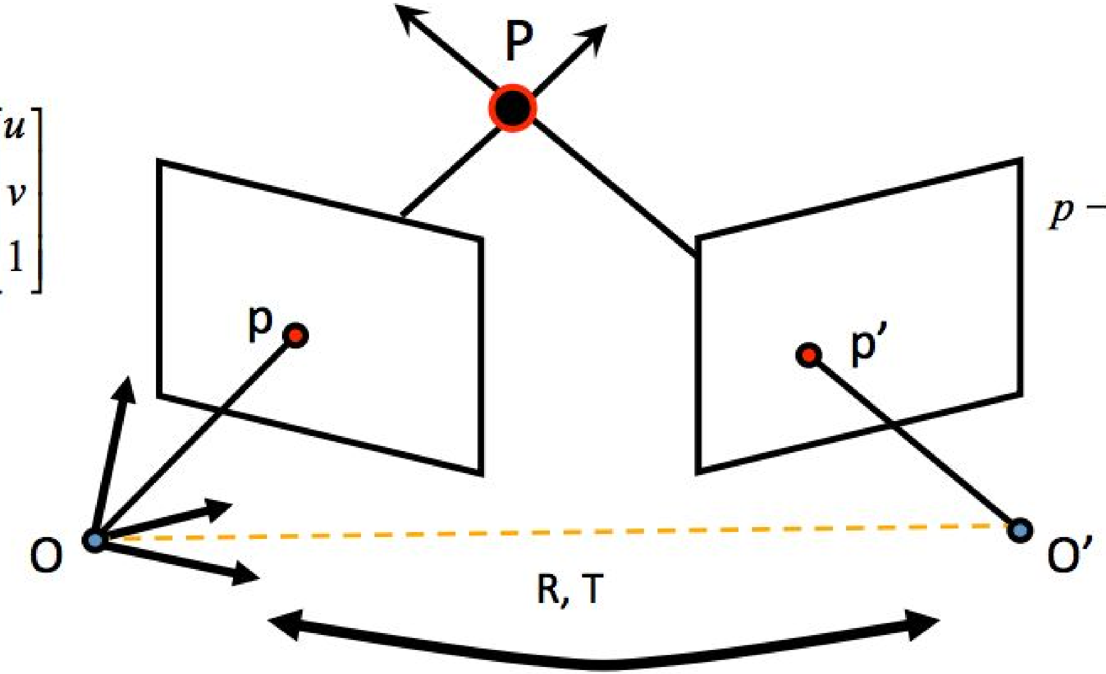
- Epipolar Plane
- Baseline
- Epipolar Lines

- Epipoles e, e'

  = intersections of baseline with image planes
  = projections of the other camera center
  = vanishing points of camera motion direction

Epipolar Constraint

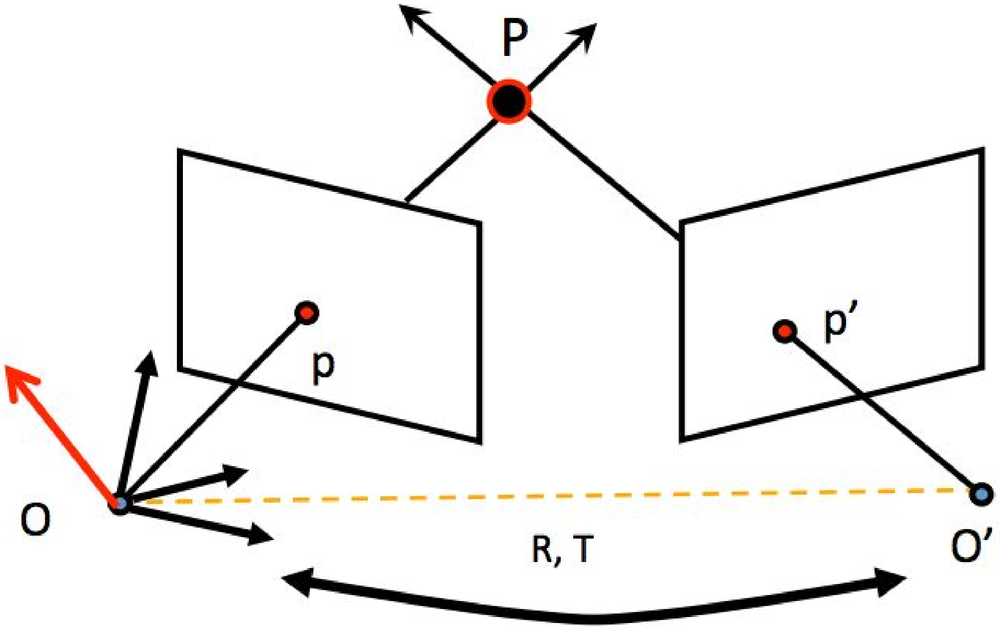
$$p \rightarrow M\ P = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$



$$p \rightarrow M'\ P = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix}$$

$$M = K\begin{bmatrix} I & 0 \end{bmatrix} \qquad M' = K'\begin{bmatrix} R & T \end{bmatrix}$$

Epipolar Constraint

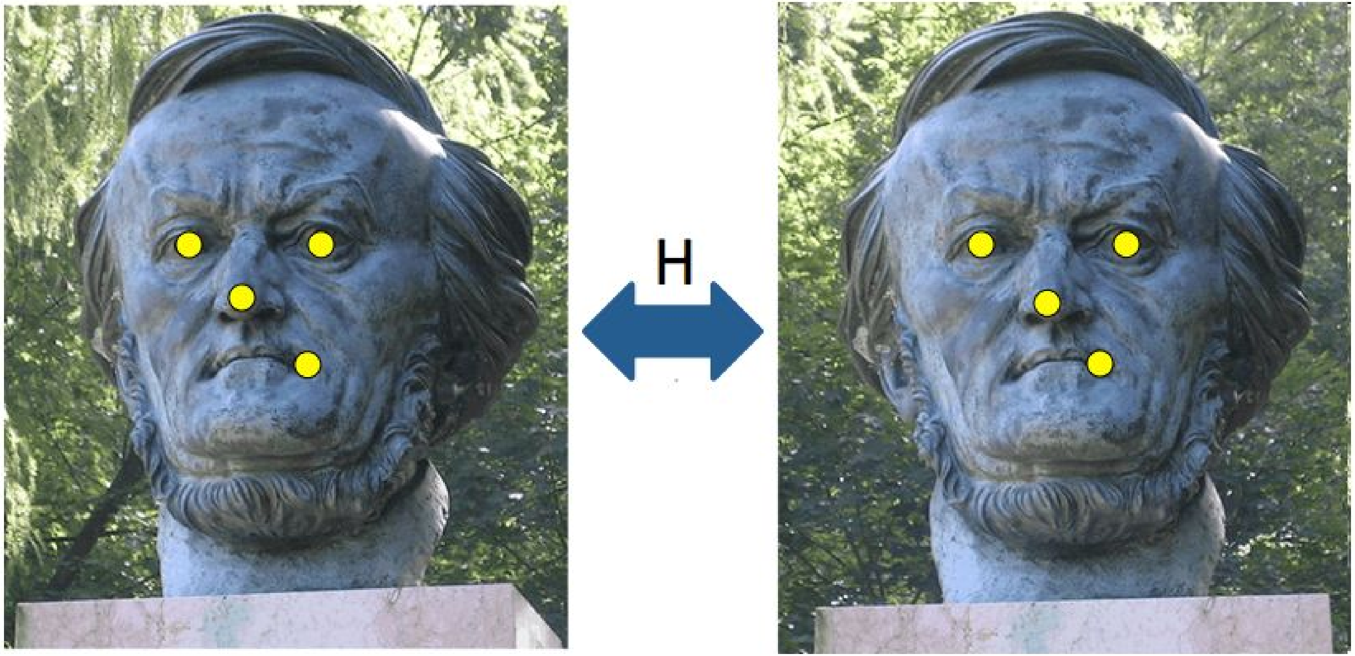


$$T \times (R\, p')$$

Perpendicular to epipolar plane

$$p^{T} \cdot [T \times (R\, p')] = 0$$

Goal: estimate the homographic transformation between two images



Assumption: Given a set of corresponding points.

DLT algorithm (direct Linear Transformation)

$$p_i' \times H \; p_i = 0 \longrightarrow \underbrace{\mathrm{A}_i}_{\substack{\text{Function of}\\ \text{measurements [2x9]}}} \overbrace{\mathbf{h}}^{\text{Unknown [9x1]}} = 0$$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \Rightarrow \mathbf{h} = \underbrace{\begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_9 \end{bmatrix}}_{9\text{x}1}$$

$\downarrow$

2 independent equations

Future Research

Self Driving Cars

Email Juan Carlos to get involved.