Image Segmentation

Javier González Jiménez

Reference Books:

- *Computer Vision: Algorithms and Applications*. Richard Szeliski. Springer. 2010.

http://szeliski.org/Book

Content

- Introduction
- Contour-based techniques
- Thresholding
- Region-based techniques
  - Region growing
  - K-Means
  - Expectation-Maximixation
  - Mean-Shift (not included)
- Semantic segmentation

1. Introduction

Segmentation …

- divides an image in regions whose pixels have similar properties (intensity, color, texture, location in the image, …).

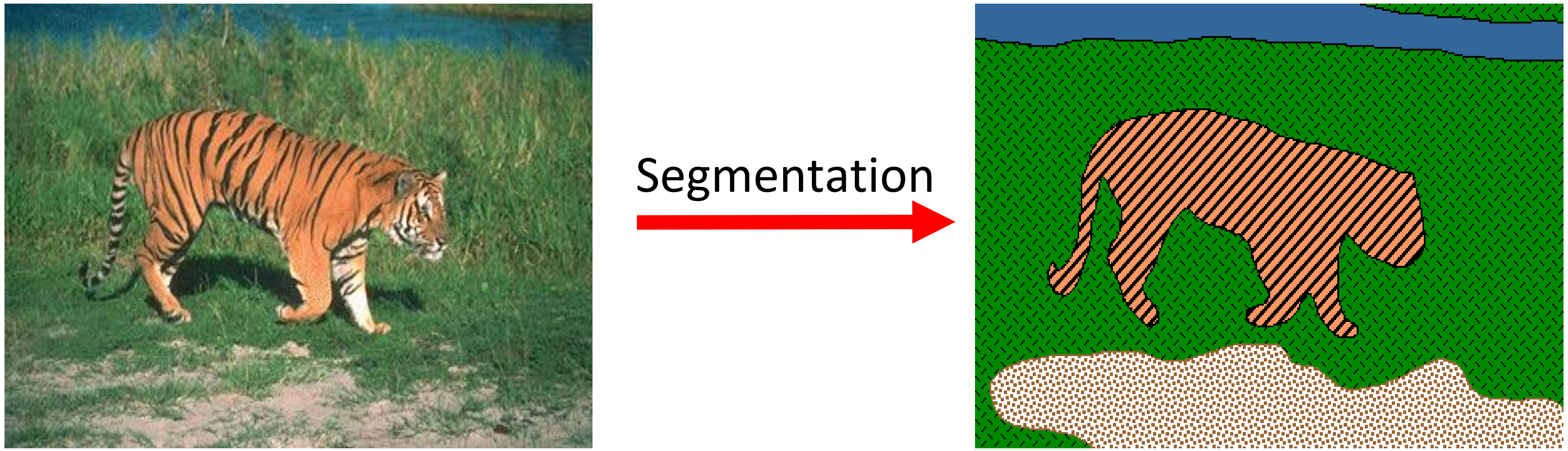


Image segmented in 4 regions

1. Introduction

Conceptually, two approaches exist:

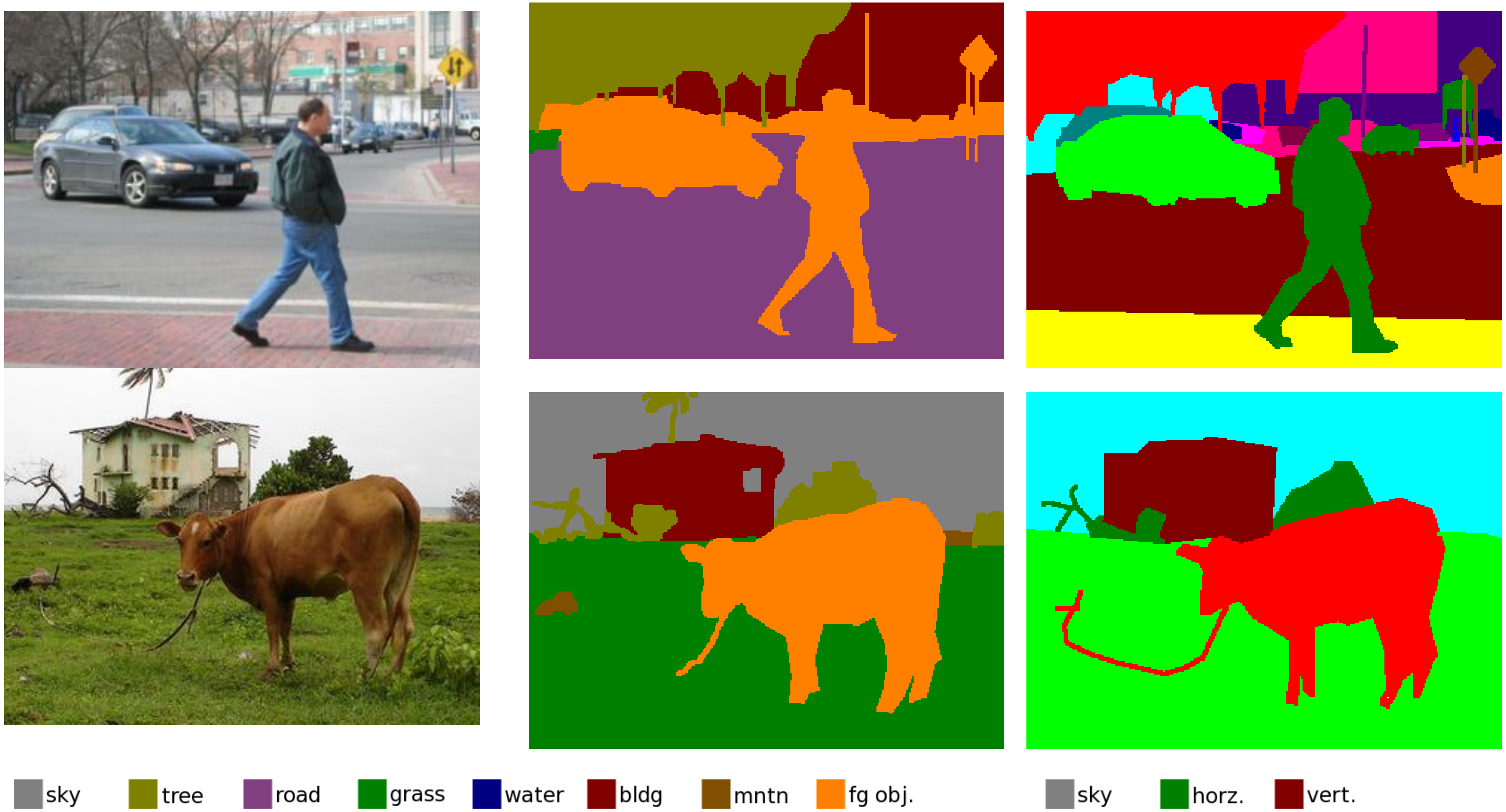
- *Top down (Semantic segmentation)*: Pixels from the same scene object should be in the same region

The zebras are one object!

- *Bottom up (pixel-driven segmentation)*: Similar pixels must be in the same region

Each black/white band is an object!

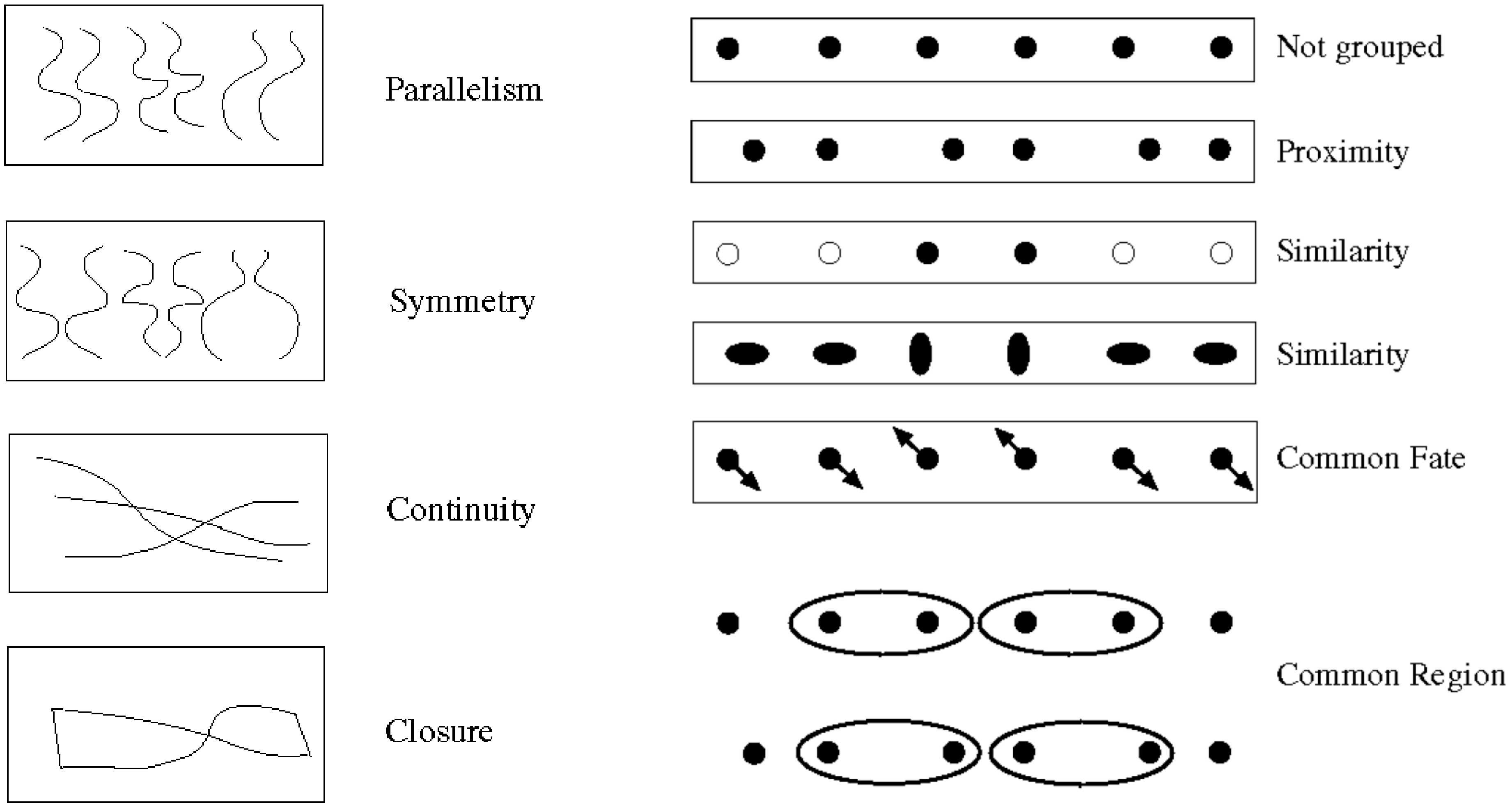
Different results depending on the properties/knowledge employed



http://dags.stanford.edu/projects/scenedataset.html

1. Introduction

The Gestalt theory stablishes properties/rules that humans employ to group entities of the images:



Examples of human segmentation

Using similarity, proximity

Using knowledge of the world

If we don’t know about the features of a dalmatian dog, no way to segment this image!

2. Contour-based techniques

Atempt to indentify the image regions by detecting the their contours

Image contours: edge pixels that enclose a region of similar intensities

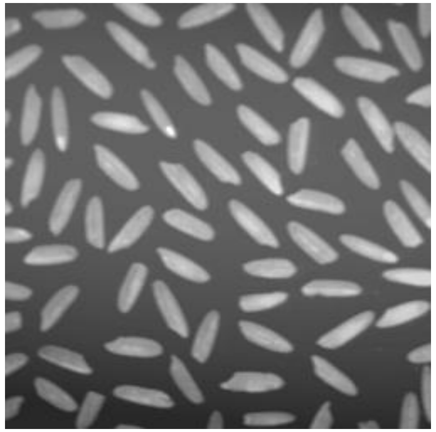
Two main approaches:

Local techniques → { LoG + zero crossing; Edge following → Canny operator }

Global techniques → Hough transform

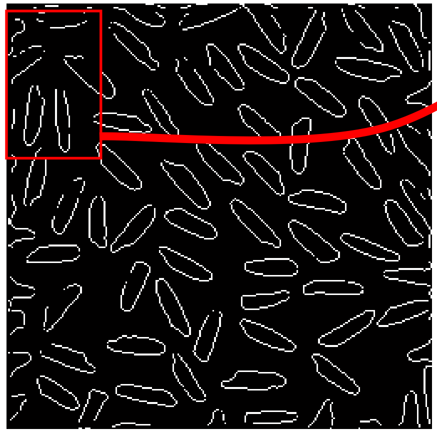
2. Contour-based techniques

LoG + Zero crossing



MATLAB

```
I = imread('rice.tif');
%BW=edge(I,'log',thresh,sigma)
log = edge(I,'log',0.001,2);
figure, imshow(log)
```

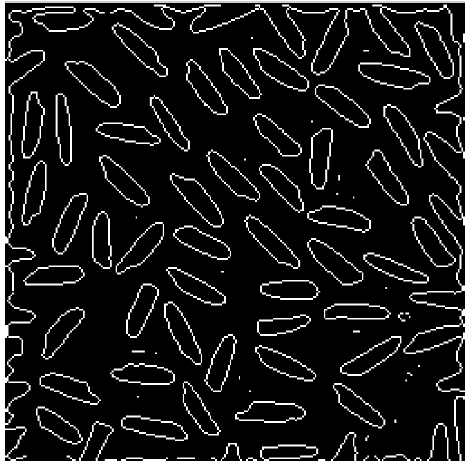


Open contours

σ=2
th=0.007

σ=2

th=0.001

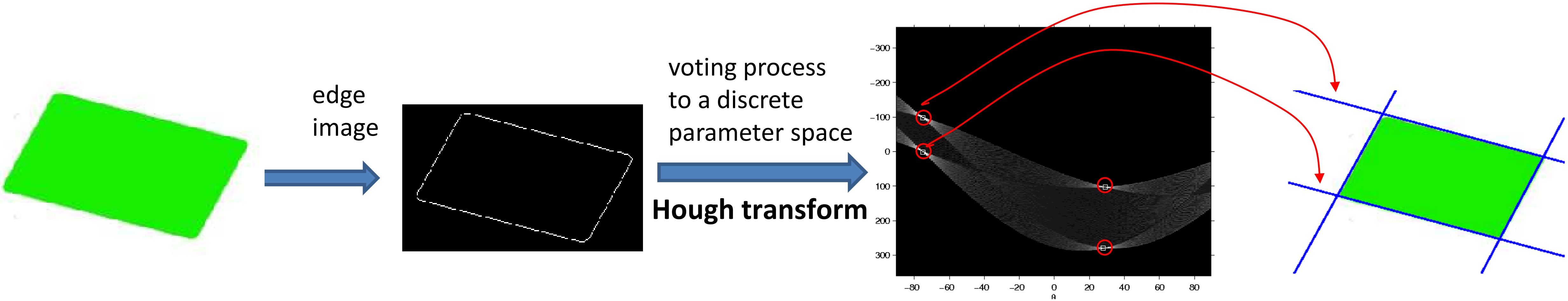


It’s an interesting method for region segmentation since closed contours provide regions of **similar pixel intensities**.

2. Contour-based techniques

Hough transform

- Algorithm patented by Paul Hough to recognize lines in photographs (Hough, 1962)
- Can detect any shape in the image
  - **Analytical forms** (classic Hough): typically line, circle, ellipse.
  - **Numerically described forms** (generalized Hough): shape given by a table
- Based on a voting scheme: each point *(x,y)* of an edge image votes for a parameter set that defines the shape in the parameter space



Hough transform: Line detection

Basic form: $y = ax + b$

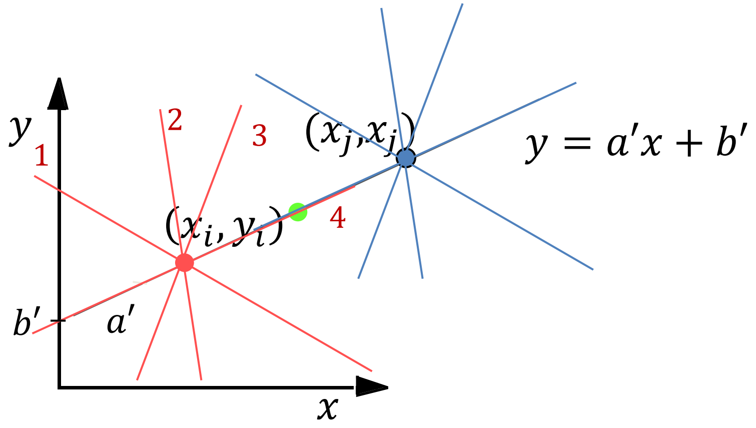
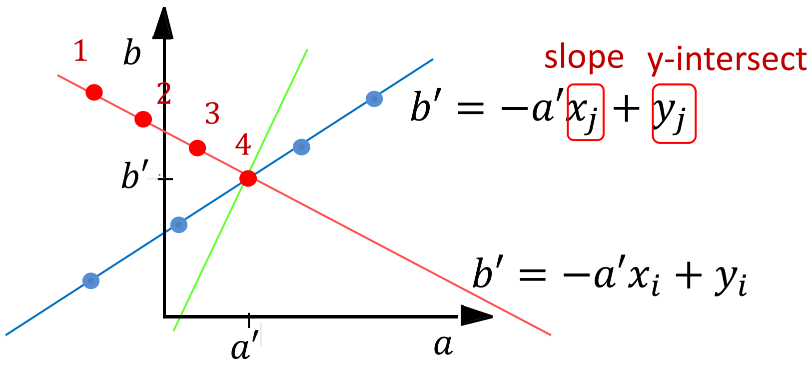


Image space



Parameter space: a, b

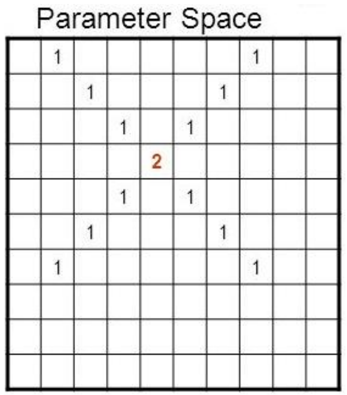
- Each point (x_i, y_i) transforms to a line in the (a, b)-parameter space

$$(x_i, y_i) \rightarrow b = -ax_i + y_i$$

- Lines through points (x_i, y_i) transform to points in the (a, b)-parameter space

Implemented as a discrete accumulator

$A(a, b)$



Algorithm

- Quantize parameter space (a, b)
- Create Accumulator array $A(a, b)$
- Set $A(a, b) = 0 \quad \forall a, b$
- For each (x_i, y_i) in the edge image

  For $a_k = 1$ *to* N

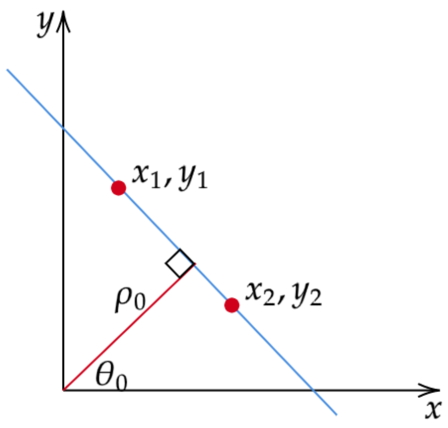
  $b_k = -a_k x_i + y_i$

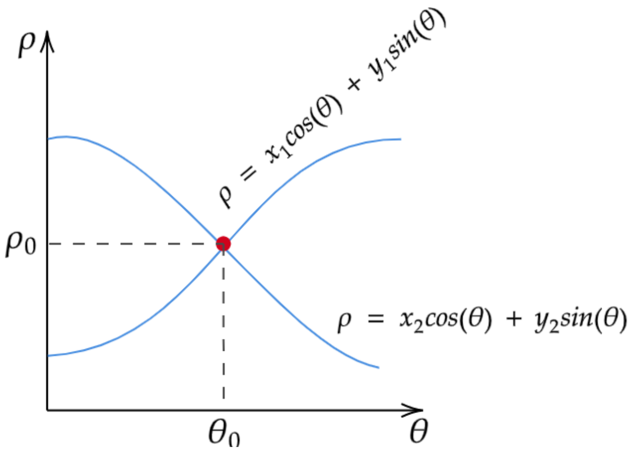
  $A(a_k, b_k) = A(a_k, b_k) + 1$
- Find local máxima in $A(a, b)$

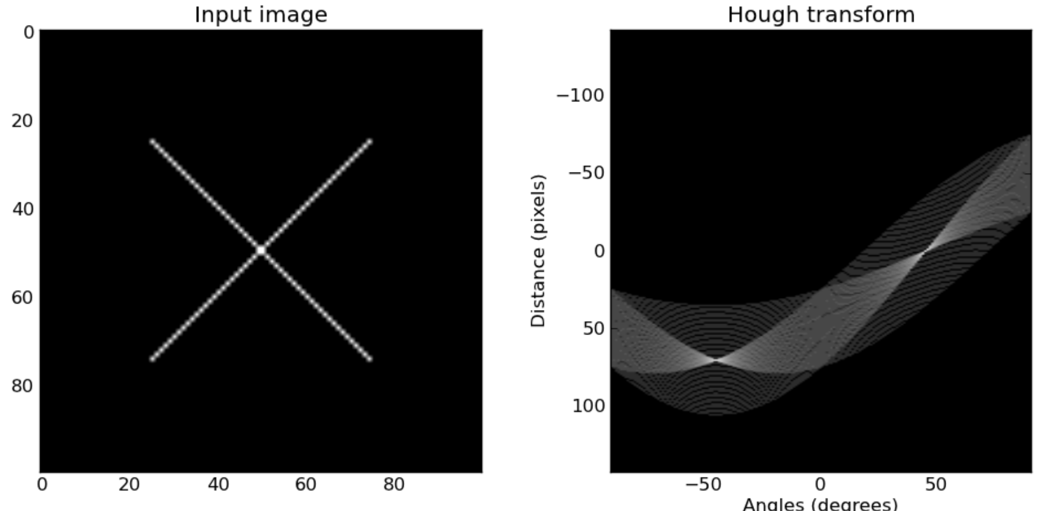
Hough transform: Line detection

Normal form: $\rho = x cos\theta + y sin\theta$

This line description does not have problems with vertical lines ($a = \infty$)



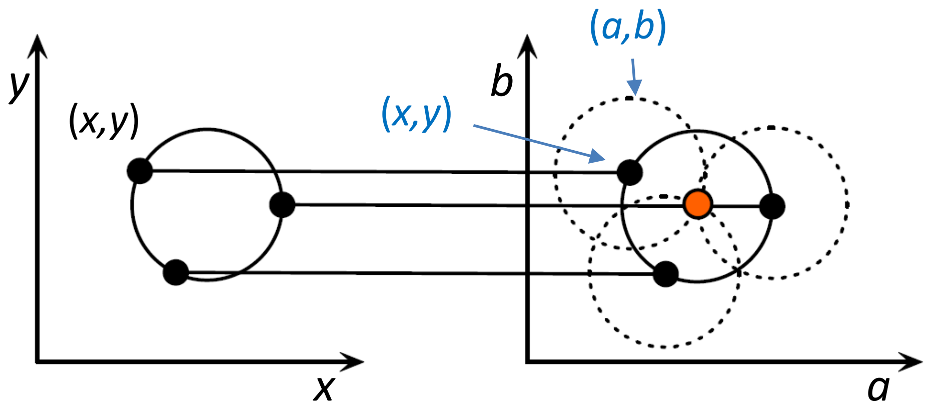




Hough transform

Example: Detecting circles

A circle of known radius *r* has two parameters: the coordinates of the center *(a,b)*

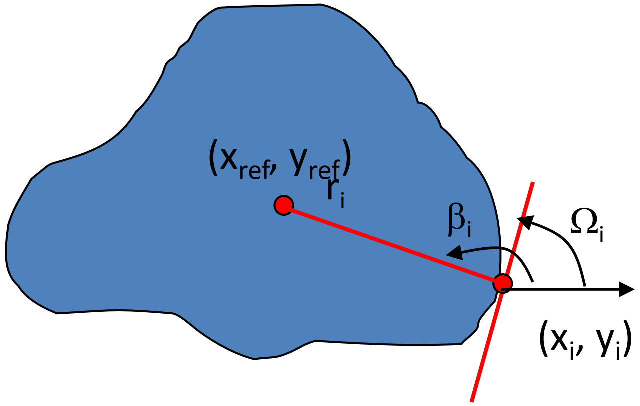


$$r^2 = (x - a)^2 + (y - b)^2$$

- Each point (*x,y*) in image space (left) votes for a circle (dashed) of candidate centers (*a,b*) in the parameter space (right).
- Ideally, all the generated circles in parameter space intersect at a certain (*a,b*) that is the unknown center of the circle in the image.

Hough transform

For numerically described shapes (generalized Hough): shape given by a table

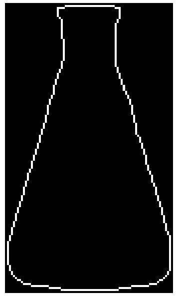


Ω_1	
Ω_2	
…	
Ω_i	$(r_1, \beta_1), (r_2, \beta_2)$
…	

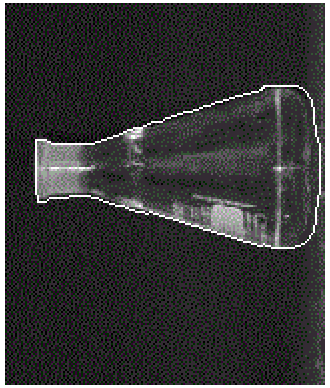
- For the tangent angle Ω_i at each contour point (x_i, y_i) we store the pair <distance (r_i), angle (β_i)> to the reference point (x_{ref}, y_{ref})
- This define the shape regardless its orientation.

Example:

Input image



Prototype shape

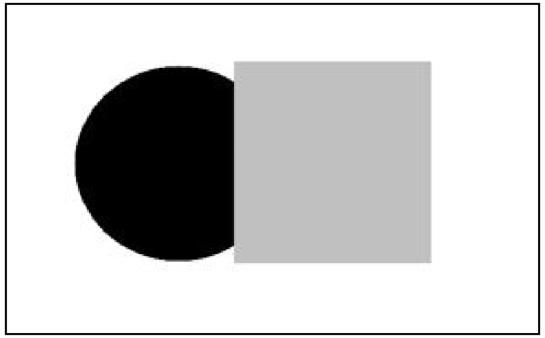


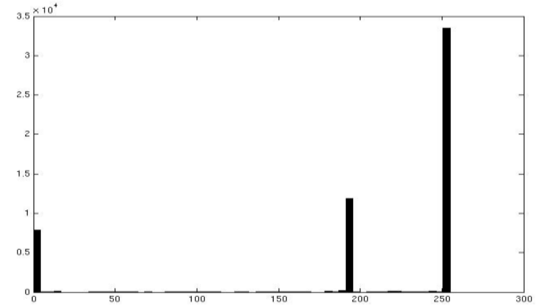
Found object

3. Thresholding

Assumption: different objects present different intensities

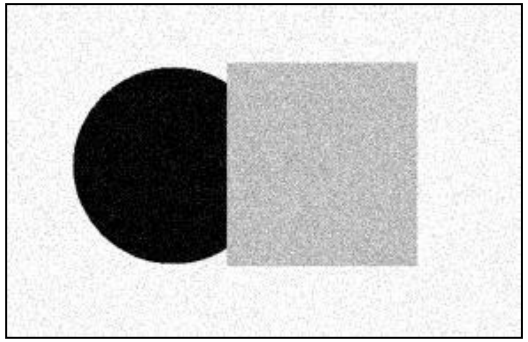
Ideal image
(3 objects)

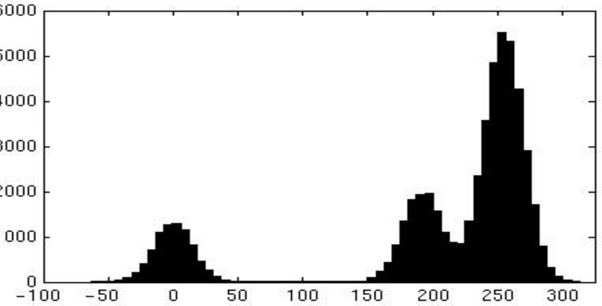




Noisy image

(the assumption does not hold)



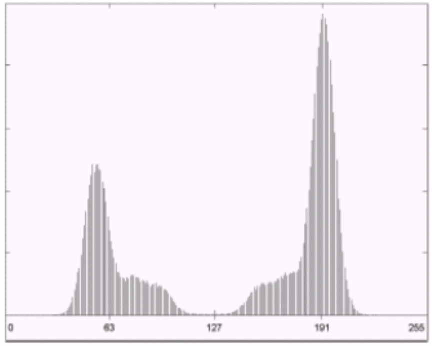


Image

Histogram

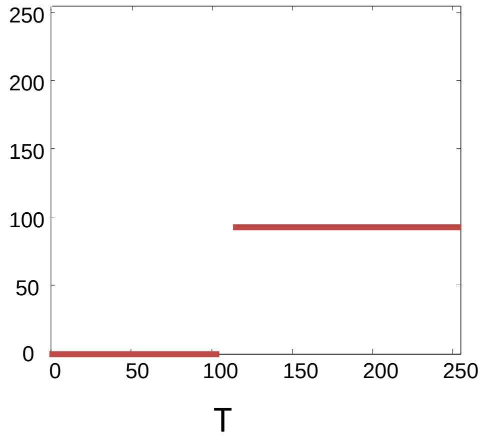
3. Thresholding

Example:



<u>Objetive</u>: find the threshold T that best separates pixels of similar intensities

Applying a binarization LUT



Segmented image

3. Thresholding

The threshold may depend on a number of variables:

$$T = T[\underbrace{x, y}_{\text{Pixel coordinates}}, p(x,y), f(x,y)]$$

Local property at pixel (x,y)

Intensity

$T = T[\, f(x,y)]$ Global threshold (only depends on the intensities)

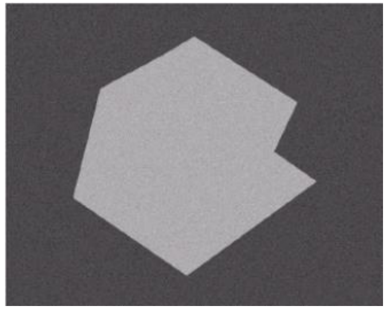
$T = T[p(x,y), f(x,y)]$ Dynamic threshold

$T = T[\, x,y, f(x,y)]$ Local threshold

Sometimes, only one threshold is required (i.e. one-color object on a one-color background) → BINARIZATION

3. Thresholding

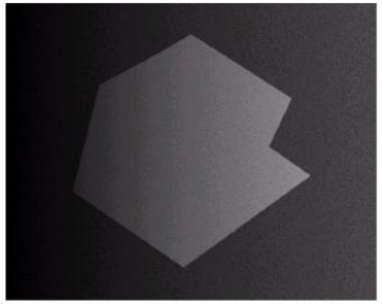
Effect of illumination

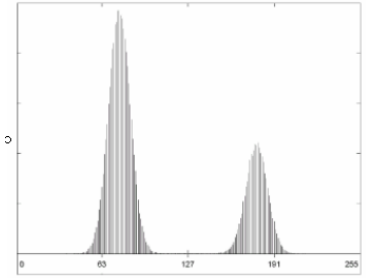


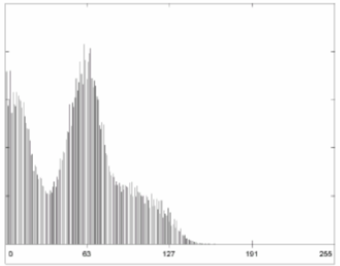
+

Non-uniform illumination

=



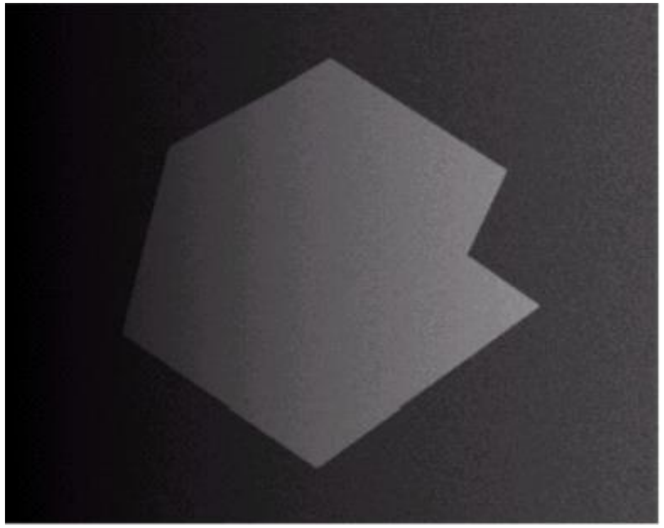




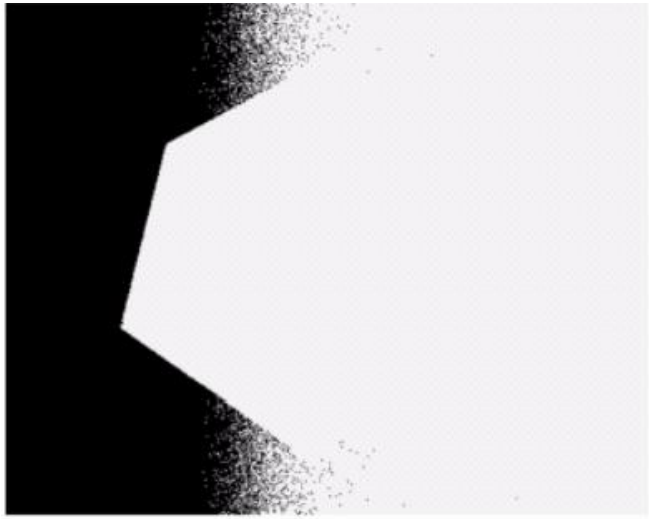
This image can be effectively segmented in two regions by binarization

None digitalization will produce good results

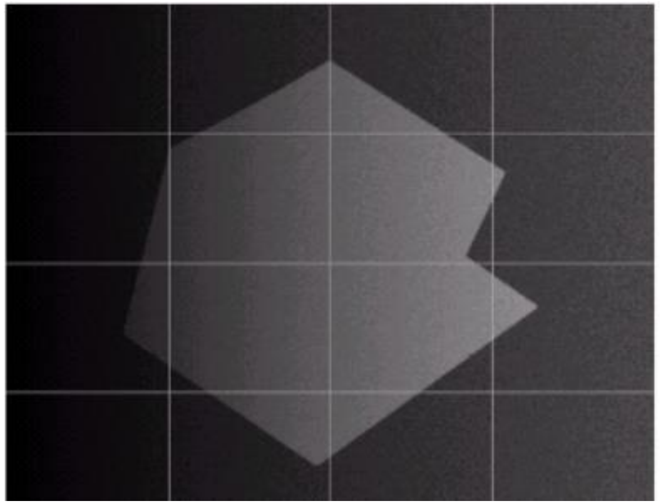
3. Thresholding



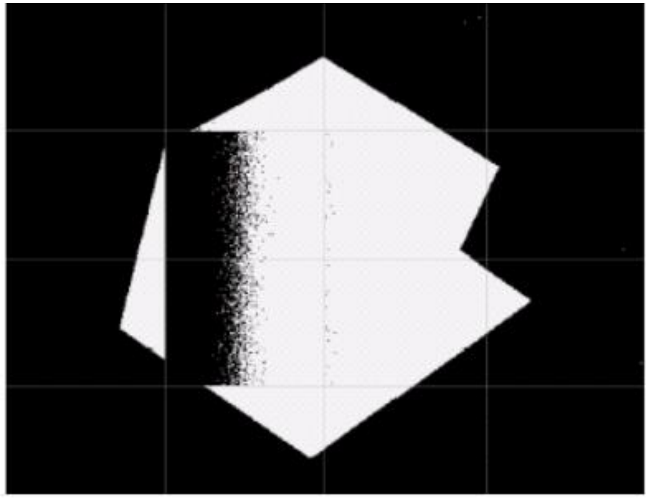
Original Image



Global threshold (one for the whole image)



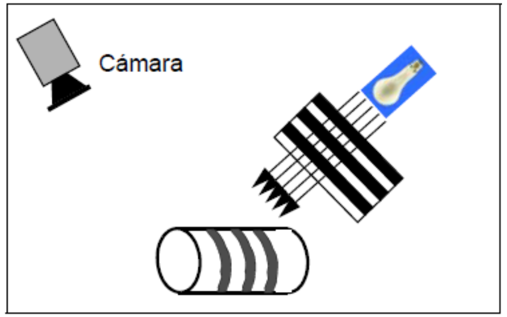
Regular partitioning of the image



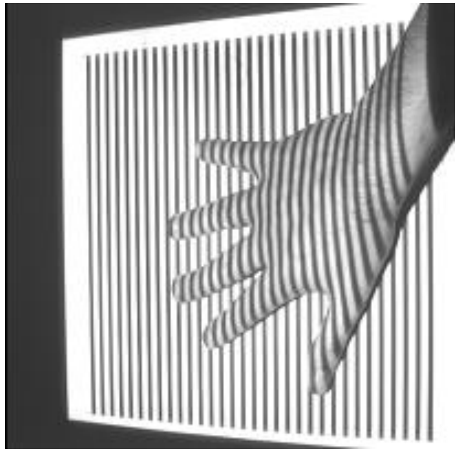
Local threshold (12 different thresholds)

3. Thresholding

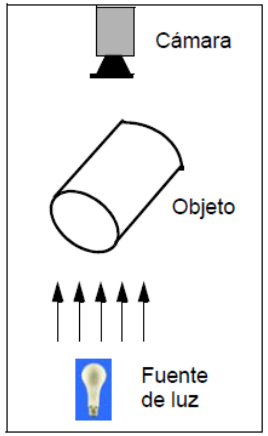
Types of illumination



Structural illumination



Used for 3D reconstruction

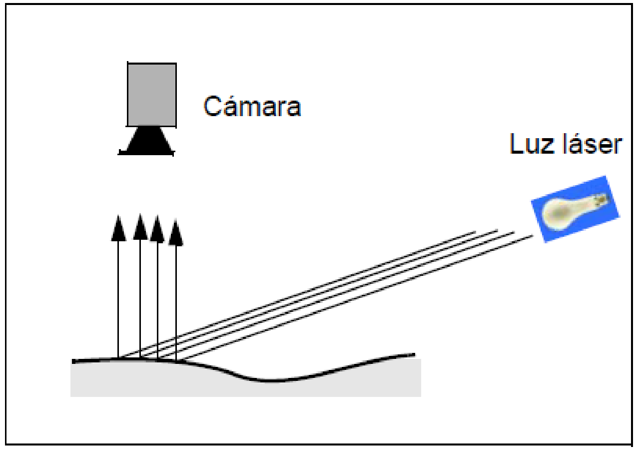


Back-projection

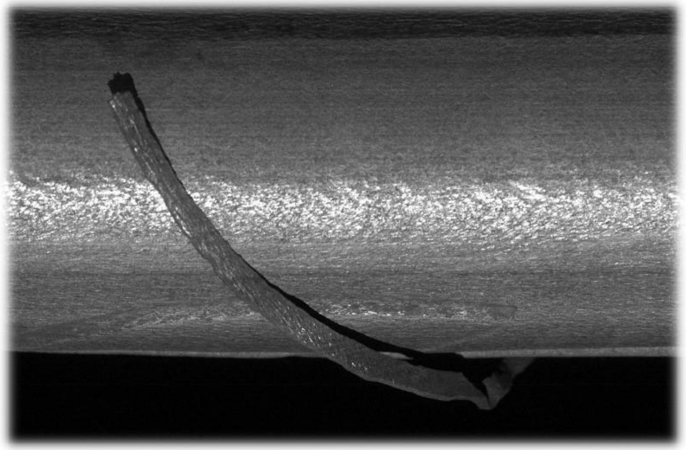
Used for binarization

3. Thresholding

Types of illumination

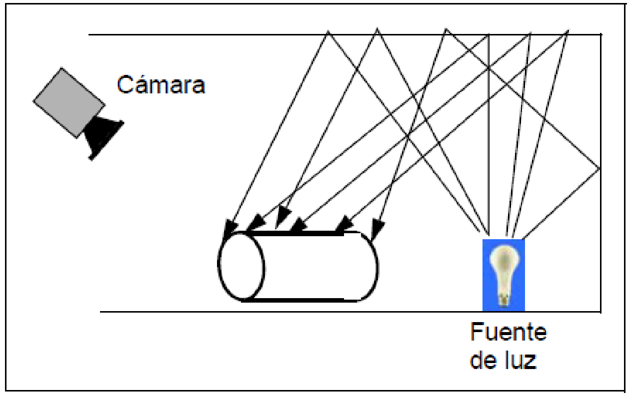


Directional illumination



© National Instruments

Used for surface inspection



Diffuse illumination

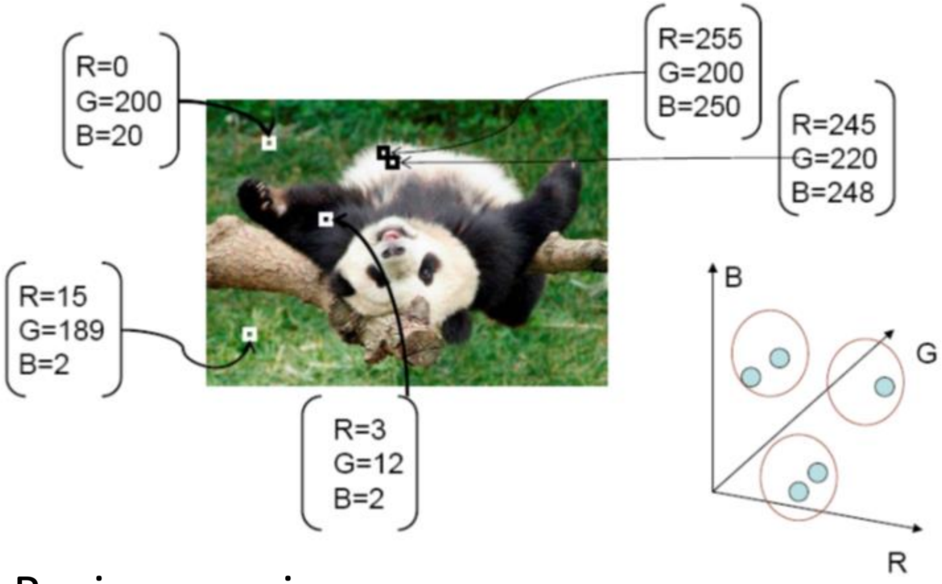
Used for color segmentation

4. Region-based techniques

Idea: Group together pixels that are similar according to some properties (*Clustering problem)*

Properties to decide on similarity: intensity, texture, color, pixel location, etc.

<u>Example</u>: property: color (RGB)



Each pixel is a point in the RGB space

Similar pixels are close to each other in this space

Techniques:

- Region growing
- K-Means
- Expectation-Maximixation
- Mean-Shift

} Probabilistic approaches

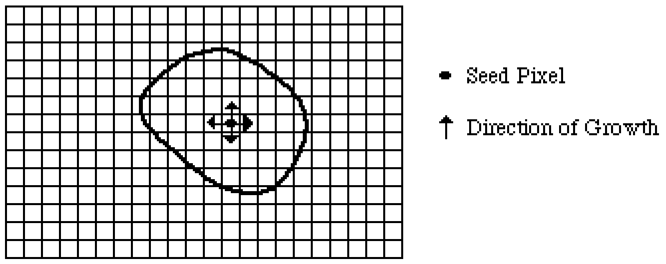
Source: K. Grauman

4. Region-based techniques

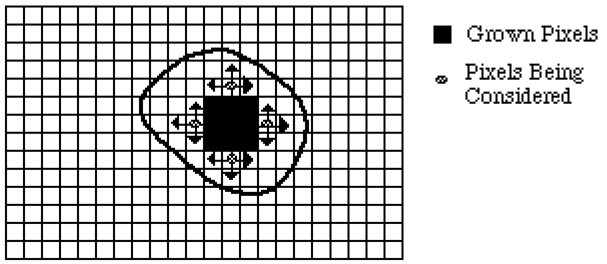
Region growing

- Start from a set of seed-pixels which are recursively grown with neighbouring pixels that show similar properties

- If *n* seed-pixels are used, the algorithm ends up with *n* regions, at most (some can be merged)



(a) Start of Growing a Region



(b) Growing Process After a Few Iterations

Region growing. Example

0	0	5	6	7
1	1	5	8	7
0	1	6	7	7
2	0	7	6	6
0	1	5	6	5

Similarity criterion: intensity difference less than 3

0	0	5	6	7
1	1	5	8	7
0	1	6	7	7
2	0	7	6	6
0	1	5	6	5

0	0	5	6	7
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
0	1	5	6	5

a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b

Similarity criterion: intensity difference less than 8

0	0	5	6	7
1	1	5	8	7
0	1	6	7	7
2	0	7	6	6
0	1	5	6	5

0	0	5	6	7
a	a	a	b	b
a	a	a	b	b
a	a	a	b	b
0	1	5	6	5

a	a	a	a	b
a	a	a	b	b
a	a	a	b	b
a	a	a	b	b
a	a	a	a	b

Result may depend on the implementation:
if the growing process starts from seed 7

0	0	5	6	7
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
0	1	5	6	5

a	b	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	b	b	b	b

4. Region-based techniques

Region growing

Key decisions:

1. How many seed pixels
   - Typically, the number of objects we are looking for.
   - Merging of connected regions during the process is possible

2. Where to place the seed pixels
   - we must apply any knowledge of the expected regions, if none, at random

3. How to select the similarity criterion to add new pixels

- ***Texture***: Images of good resolution are needed.
- ***Color/Intensity***: We can take into account the (dynamic) mean and standard deviation of the current region.
- ***Expected Shape***: We can weight particular directions that more likely fit the expected object shape

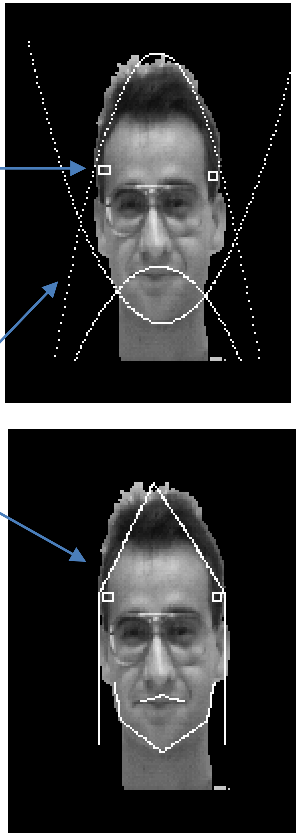
4. Region-based techniques

Region growing

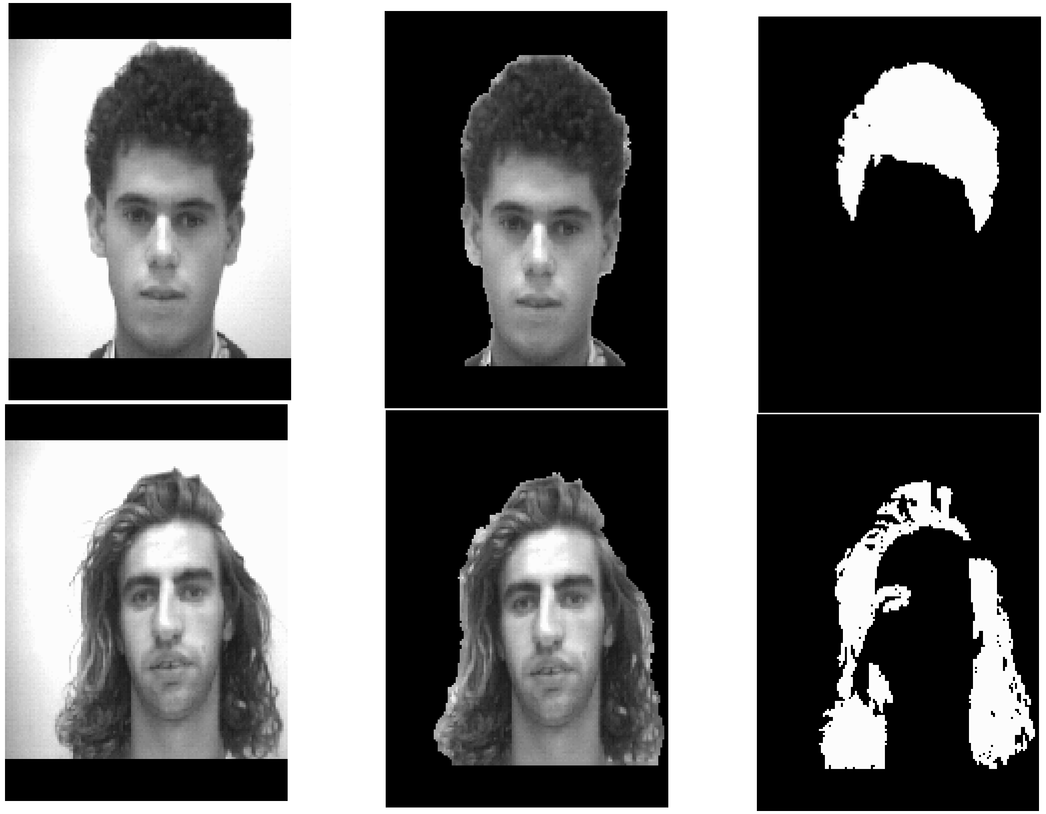
Example: hair segmentation on a previously segmented head

Seeds placed at specific pixels (detected by a previous algorithm)

Expected shapes of the hair used as property



Original image | Segmentad head (given) | Segmented hair



Property used here: hair pixels must be close to these axes and darker than the face pixels

4. Region-based techniques

K-means

A clustering technique: Given a set of elements, make K clusters out of them

<u>How</u>: minimize the sum of squared Euclidean distances between points x_i and their **nearest cluster centers** m_k

$$\arg\min_M D(X,M) = \sum_{\text{cluster } k} \; \sum_{\substack{\text{point } i \text{ in} \\ \text{cluster } k}} (x_i - m_k)^2$$

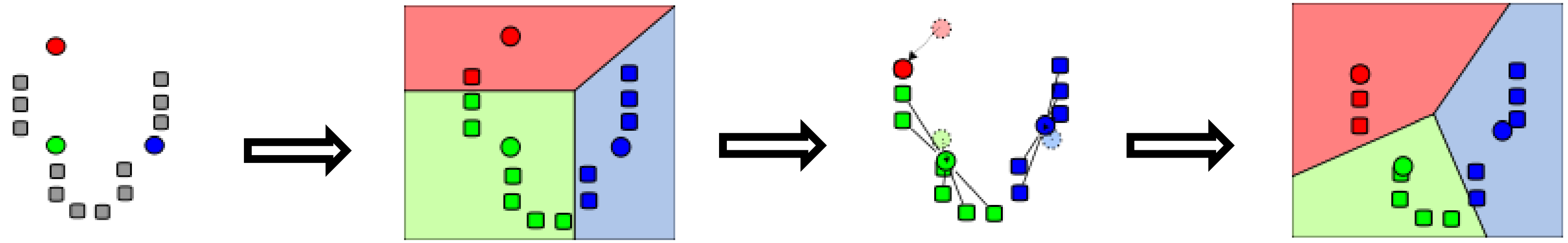
General Algorithm:

1. Pick K **initial means** representing the clusters (not necessary elements of the set)
2. **Assign** each element **to the closest mean**, so creating new clusters
3. Compute the new means of the clusters
4. Repeat steps 2 and 3 until convergence

Example K=3

Assignment 1

Assignment 2



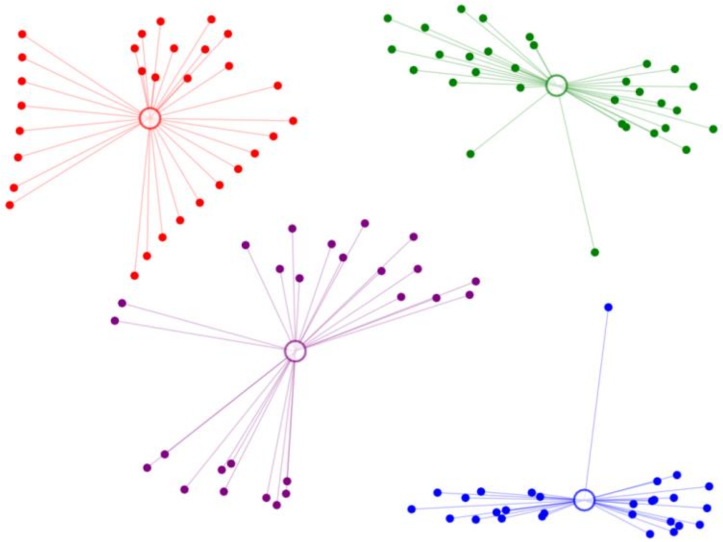
(Wikipedia)

4. Region-based techniques

K-means

Similar to region growing, but …

- All the pixels in the image are classified (assigned to a region) at each step, not just the neighbours
- The algorithm stops when the centers of the regions do not move (in region growing the stop condition is “there are no pixels to add”)



Interactive illustration:
https://user.ceng.metu.edu.tr/~akifakkus/courses/ceng574/k-means/

4. Region-based techniques

K-means for segmentation

- Each **pixel** of the image is represented by a **feature vector** (e.g., color, intensity, texture, etc.)
- Each **region** is represented by the **mean of the feature vector** of the pixels in it
- In the feature space we need to define a **distance between vectors** (e.g. Euclidean distance)

Algorithm

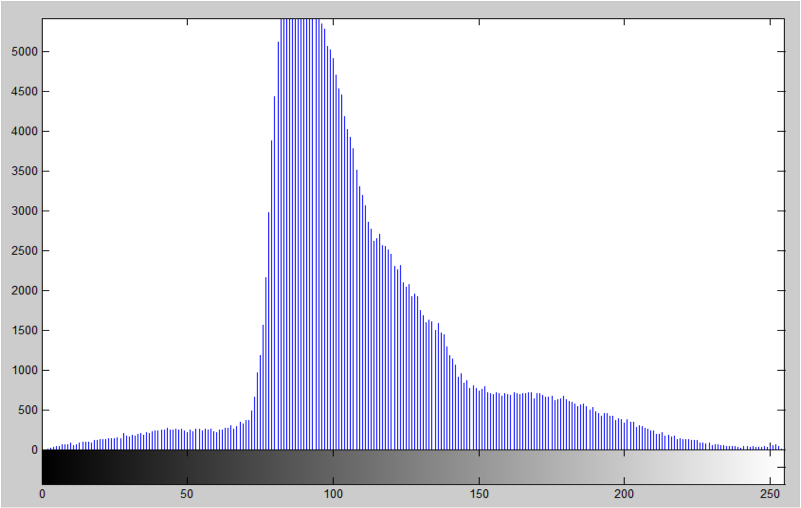
1. Select K pixels in the image (manually, at random, with some heuristic, ...) which will represent K regions
2. Assign each pixel in the image to the more similar region (the closest one according to the adopted distance)
3. Update the feature vector of the new region
4. Repeat 2 and 3 until no image pixel changes from one region to another (i.e. region centers do not change) → CONVERGENCE
5. Merge connected regions if similar

K-means: Matlab example using intensity as feature vector (1x1)

```
im = imread('torre_monica.jpg'); if (size(im,3)==3) im = rgb2gray(im); end
figure,subplot(2,2,1),imshow(im),title('Original Image')
subplot(2,2,3),imhist(im),title('Histogram')
nPixels = prod(size(im)); k=2; % Two classes
data = reshape(im, nPixels,1 ); %Image as a row vector (one feature)
idx = kmeans(double(data), k); % cluster indices of each pixel (1 or 2)
clust=reshape(idx, size(im)); %Vector-Image back to a matrix
im_clust=uint8(255*clust/max(max(clust)));
subplot(2,2,2), imshow(im_clust), title('Segmented Image')
subplot(2,2,4),imhist(im_clust),title('Histogram')
```

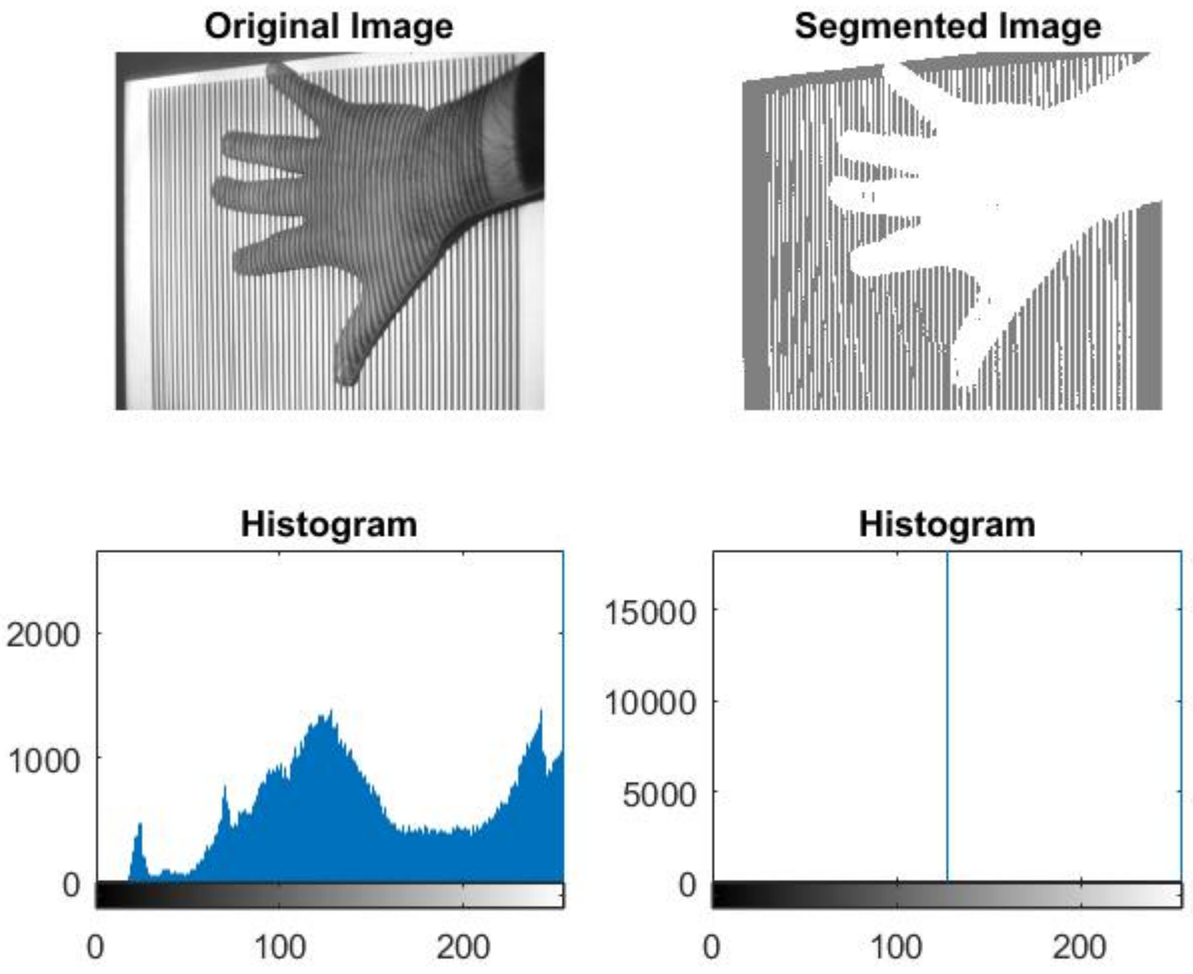
Gray image

Segmentation from gray



Equivalent to thresholding with the threshold at the middle of the two means (reason why!)

Another example of K-means using intensity as feature vector (dimension 1x1)



4. Region-based techniques

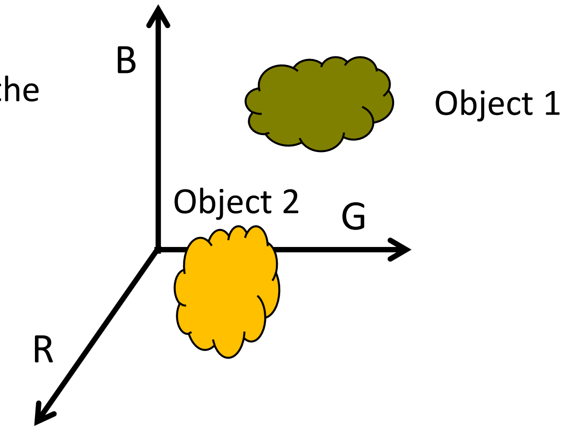
K-means. Example using color R,G,B as feature vector

P pixels in the image

$$\texttt{data} = \begin{bmatrix} R_1 & R_2 & \cdots & R_p \\ G_1 & G_2 & \cdots & G_p \\ B_1 & B_2 & \cdots & B_p \end{bmatrix}$$

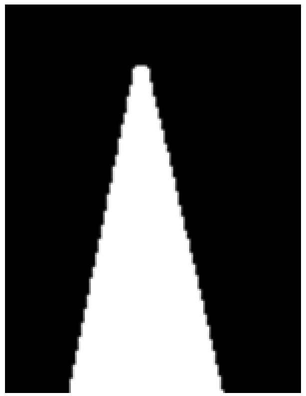
Feature vectors: R,G,B

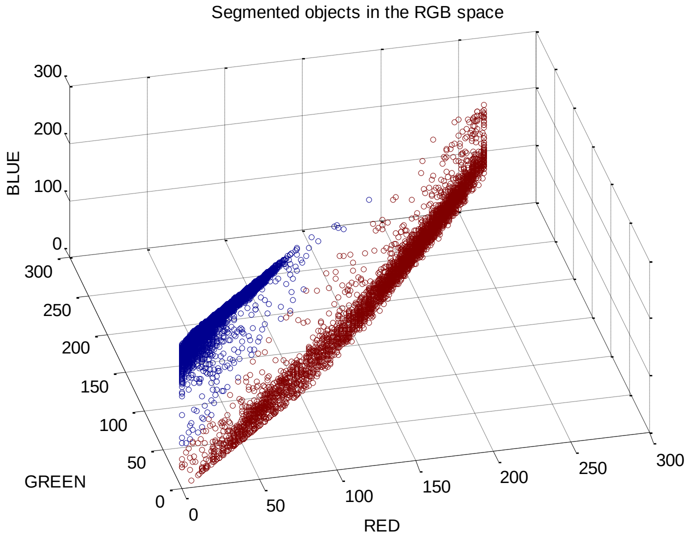
2 clusters in the RGB space



Color image

Segmentation from color





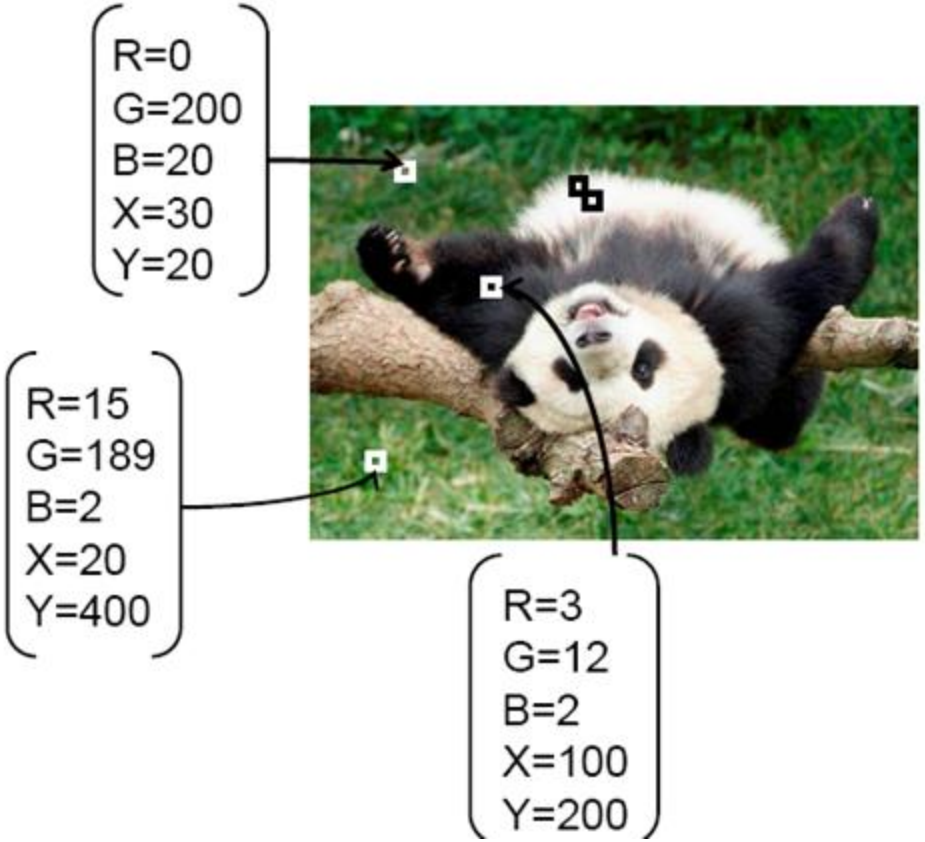
Notice: If the two initial pixels are from the sky the result will not be correct.

4. Region-based techniques

K-means

Usually, color RGB is not enough for a good segmentation ⇨ We need some localization feature p.e. (r,g,b,x,y)

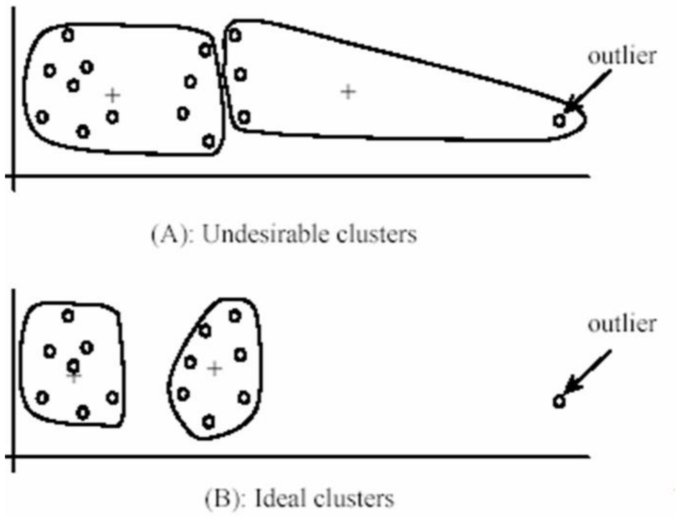
Pixel position



4. Region-based techniques

K-means

- Pros
  - Simple
  - Convergence to a local minima (but no guarantee to reach the global minima)
- Cons
  - High use of memory
  - Fixed K
  - Sensible to the selection of the initialization
  - Sensible to outliers
  - Circular clusters in the feature space are assumed (because the use of the Euclidean distance)



(A): Undesirable clusters

(B): Ideal clusters

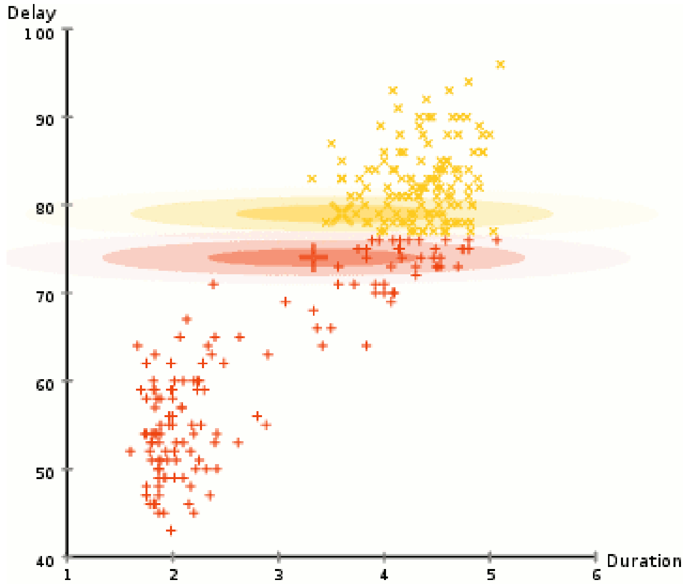
4. Region-based techniques

Expectation – Maximization (EM)

- Is the generalization of the K-means method
- It's a *soft* clustering since it does not give "hard" clusters but the probability that an element (feature vector x of a pixel) belongs to each cluster C_j: $p(x|C_j)$
- Probabilities of the clusters are assumed to be Gaussians: $p(x|C_j) \sim N(\mu_j, \Sigma_j)$
- At each iteration not only the mean is refined (as in K-means), but also the covariance matrix of the cluster

Example for a two-dimensional feature vector.

Each point is the feature vector of a image pixel



What is a posterior probability?

Prior probability

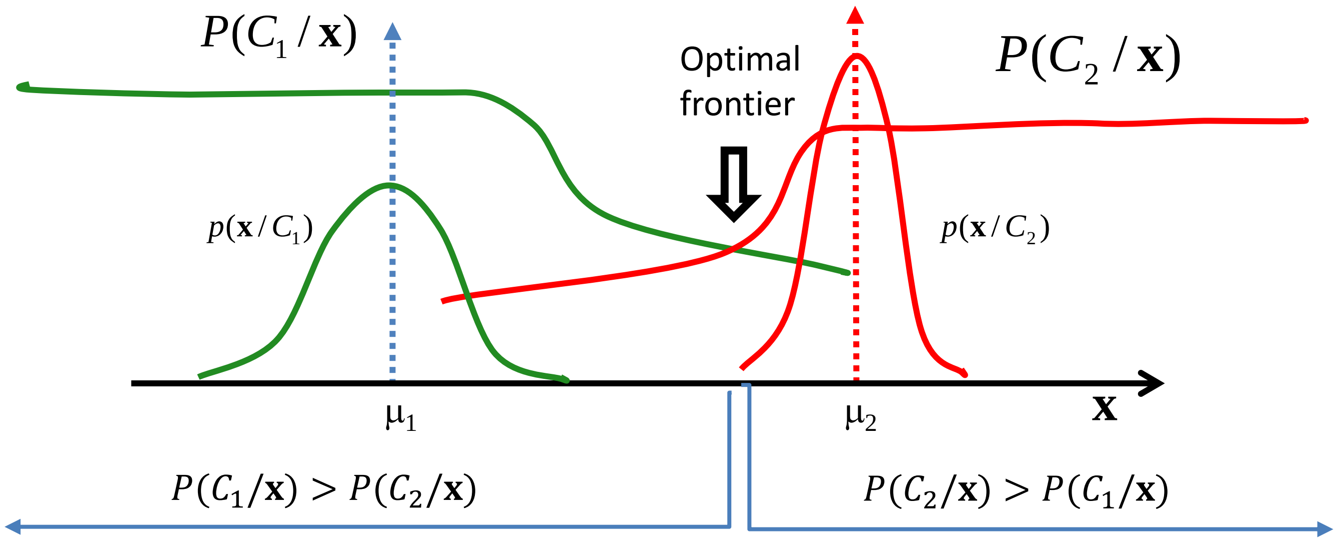
Bayes rule: $$P(C_j/\mathbf{x}_i) = \frac{p(\mathbf{x}_i/C_j)P(C_j)}{p(\mathbf{x}_i)} = Kp(\mathbf{x}_i/C_j)P(C_j)$$

Posterior probability

Total probability (does not depend on C_j)

Conditional pdf (o *Likelihood of* C_j) $p(\mathbf{x}_i/C_j)$: Probability function that a pixel from the cluster C_j has the feature vector $\mathbf{x}_i$

Example of two clusters :



Pdf: probability density function

For this $\mathbf{x}$ it is more probable that it belongs to cluster C_1

For this $\mathbf{x}$ it is more probable that it belongs to cluster C_2

4. Region-based techniques

Expectation – Maximization

Normal (Gaussian) distribution: Features $\mathbf{x}_i$ are continuous random variables that follow the pdf:

$$p(\mathbf{x}_i/C_j) = \frac{1}{(2\pi)^{n/2}\left|\Sigma_j\right|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}_i-\boldsymbol{\mu}_j)^T \Sigma_j^{-1}(\mathbf{x}_i-\boldsymbol{\mu}_j)}$$

Feature vector of dimension *n*: $\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_f \quad \cdots \quad x_n]^T$

Mean vector: $\boldsymbol{\mu} = [\mu_1 \quad \mu_2 \quad \cdots \quad \mu_f \quad \cdots \quad \mu_n]^T$

Covariance matrix

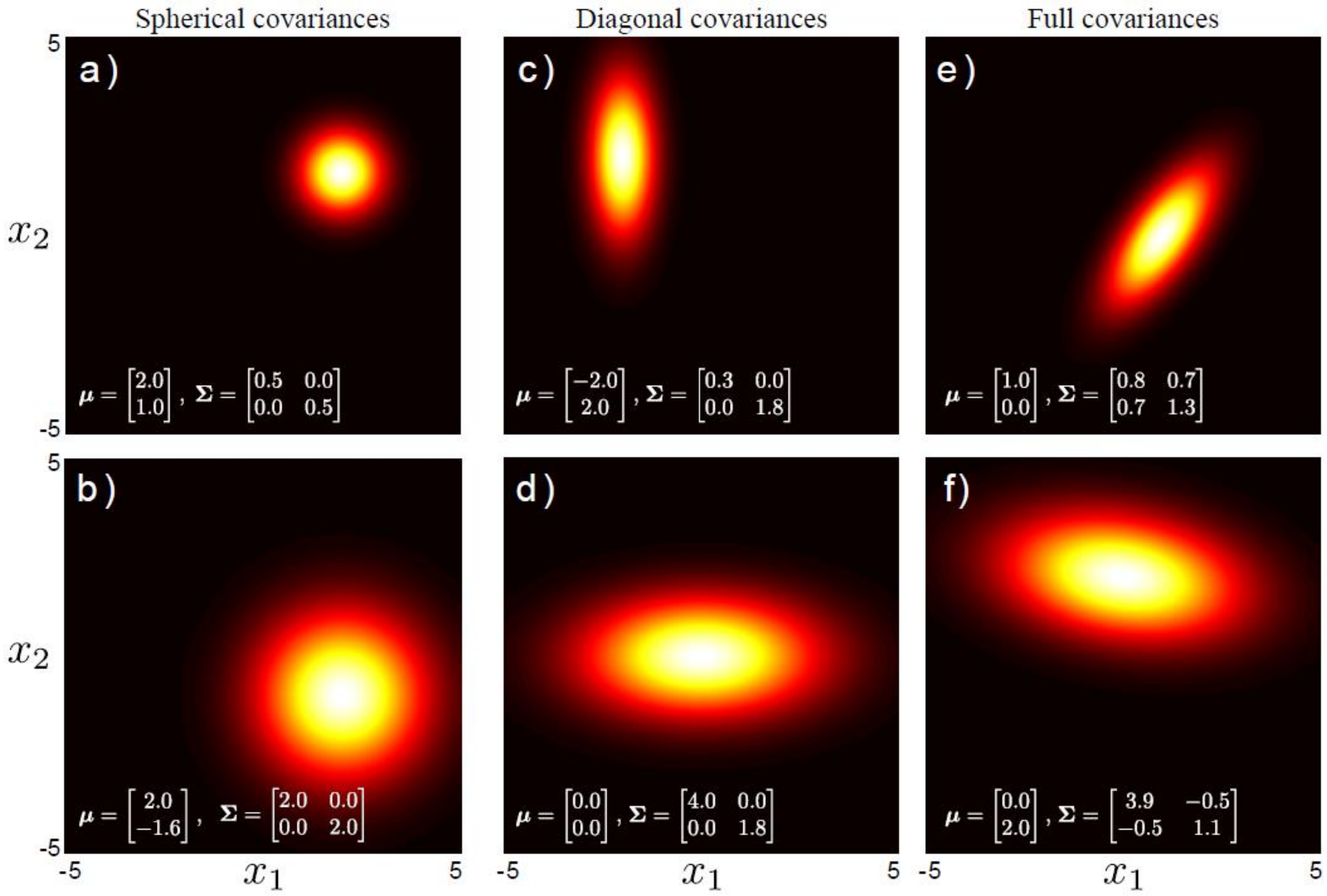
$$\Sigma = \mathrm{E}[(\mathbf{x}-\boldsymbol{\mu})\cdot(\mathbf{x}-\boldsymbol{\mu})^T] = \begin{bmatrix} \sigma_{11} & \cdots & \sigma_{1f} & \cdots & \sigma_{1n} \\ \vdots & & \vdots & & \vdots \\ \sigma_{n1} & \cdots & \sigma_{nf} & \cdots & \sigma_{nn} \end{bmatrix} = \begin{bmatrix} \sigma_{11} & \cdots & 0 & \cdots & 0 \\ \vdots & & \vdots & & \vdots \\ 0 & \cdots & 0 & \cdots & \sigma_{nn} \end{bmatrix}$$

If the features are independents

4. Region-based techniques

Expectation – Maximization

A Gaussian pdf for different μ and Σ



Expectation – Maximization

Each *cluster* C_j has a different and unknown Gaussian pdf $p(x|\mathrm{C}_j) \sim N(\mu_j, \Sigma_j)$

Estimated from data

Given

Computed (with Bayes)

$$\left.\begin{array}{l} \mathrm{C}_1 \rightarrow p(x|\mathrm{C}_1) \sim N(\mu_1, \Sigma_1),\ P(\mathrm{C}_1) \\ \mathrm{C}_2 \rightarrow p(x|\mathrm{C}_2) \sim N(\mu_2, \Sigma_2),\ P(\mathrm{C}_2) \\ \qquad\qquad \ldots \\ \mathrm{C}_k \rightarrow p(x|\mathrm{C}_k) \sim N(\mu_k, \Sigma_k),\ P(\mathrm{C}_k) \end{array}\right\} \xrightarrow{\text{Expectation}} \begin{array}{c} p(\mathrm{C}_1|x) \\ p(\mathrm{C}_j|x_2) \\ \ldots \\ p(\mathrm{C}_k|x_k) \end{array}$$

Likelihood

Prior

Posterior

(Likelihood) Maximization

If all the $P(C_j)$ are assumed to be identical for all the clusters and $\Sigma_j = \sigma^2 I$ (identical isotropic gaussians), **EM is equivalent to K-means**

EM Algorithm:

- **Initialize K clusters:**

  Pick at random (μ_j, Σ_j) and $P(C_j)$ for each cluster j

We need to define an initial $p(x_i|C_j)$ $\sim N(\mu_j, \Sigma_j)$ for each cluster C_j

Prior

- **Iteration Steps**:
  - Using Bayes, compute the cluster C_j to which each data point x_i belongs

    $p(C_j|x_i)$ ⇒ Expectation

  - Re-estimate the cluster parameters (MLE: Maximum Likelihood estimate)

    $(\mu_j, \Sigma_j), p(C_j)$ For each cluster j ⇒ Maximization

EM Algorithm:

Iteration steps (until convergence):

Expectation Step: With the current $P(x_i|C_j)$ and $P(C_j)$ compute (via Bayes) the **expected probabilities** $P(C_j|x_i)$ (that x_i belongs to cluster C_j)

$$P(C_j \mid x_i) = \frac{p(x_i \mid C_j)\cdot p(C_j)}{p(x_i)} = \frac{p(x_i \mid C_j)\cdot p(C_j)}{\sum_i p(x_i \mid C_j)\cdot p(C_j)}$$

Assign x_i to the cluster C_j with the highest probabilities $P(C_j|x_i)$

Maximization Step: Estimate $(\mu_j, \Sigma_j), p(C_j)$ that **maximize the Likelihood** $p(x|C_1)$

(Maximun Likelihood Estimation - MLE)

$$\mu_j = \frac{\sum_i p(C_j \mid x_i)\cdot x_i}{\sum_i p(C_j \mid x_i)} \qquad \Sigma_j = \frac{\sum_i p(C_j \mid x_i)\cdot(x_i-\mu_j)\cdot(x_i-\mu_j)^T}{\sum_i p(C_j \mid x_i)}$$

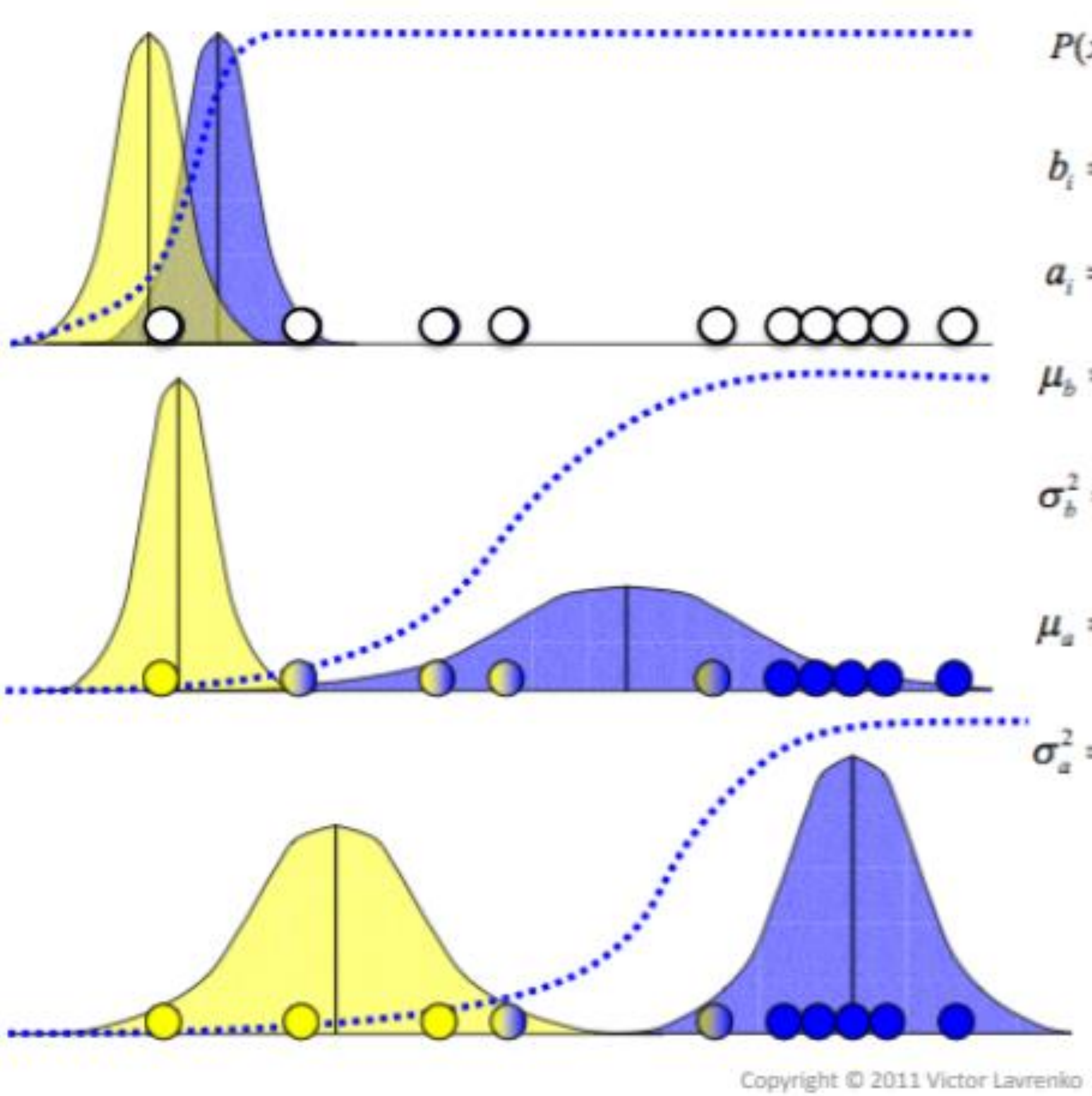
ML estimation of the mean and covariance

$$p(C_j) = \sum_i p(C_j \mid x_i)p(x_i) = \frac{\sum_i p(C_j \mid x_i)}{N}$$

If no other information is available, $p(x_i)$ are considered equal-probable

Convergence: when no change ocurrs in a complete iteration

EM: 1-d example



$$P(x_i \mid b) = \frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left(-\frac{(x_i-\mu_b)^2}{2\sigma_b^2}\right)$$

$$b_i = P(b \mid x_i) = \frac{P(x_i \mid b)P(b)}{P(x_i \mid b)P(b) + P(x_i \mid a)P(a)}$$

$$a_i = P(a \mid x_i) = 1 - b_i$$

$$\mu_b = \frac{b_1x_1 + b_2x_2 + \ldots + b_nx_{n_b}}{b_1 + b_2 + \ldots + b_n}$$

$$\sigma_b^2 = \frac{b_1(x_1-\mu_1)^2 + \ldots + b_n(x_n-\mu_n)^2}{b_1 + b_2 + \ldots + b_n}$$

$$\mu_a = \frac{a_1x_1 + a_2x_2 + \ldots + a_nx_{n_b}}{a_1 + a_2 + \ldots + a_n}$$

$$\sigma_a^2 = \frac{a_1(x_1-\mu_1)^2 + \ldots + a_n(x_n-\mu_n)^2}{a_1 + a_2 + \ldots + a_n}$$

could also estimate priors:

$P(b) = (b_1 + b_2 + \ldots b_n) / n$

$P(a) = 1 - P(b)$

K-means *vs.* EM

	K-means	**EM**
Cluster Representation	Mean	Mean, variance
Cluster Initialization	Randomly select K means	Initialize K Gaussian distributions (μ_j, Σ_j) and $P(C_j)$
Expectation: Estimate the cluster of each data	Assign each point to the closest mean	Compute $P(C_j \| x_i)$
Maximization: Re-estimate the cluster parameters	Compute means of current clusters	Compute new (μ_j, Σ_j), $P(C_j)$ for each cluster j