UniGe | MaLGa

Image Segmentation

Computational Vision

Francesca Odone - francesca.odone@unige.it

Segmentation is a difficult task to solve and to formulate

From pixels to....

images can also be divided in somewhat uniform image regions

uniformity is based on a quality (e.g, brightness, color or motion)

- image segmentation (regions that can be further analyzed)
- superpixels computation (~ features)

UniGe | MaLGa

Image segmentation

Image segmentation methods overview

Unsupervised: segmentation is based on pixels and appearance information

Supervised: segmentation is guided by some semantic concept

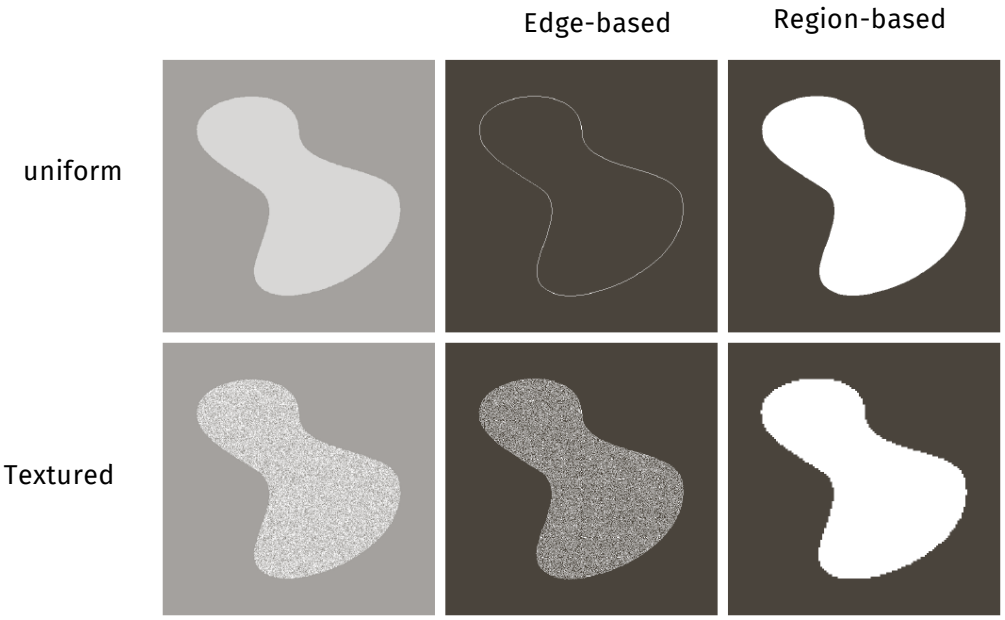
Unsupervised image segmentation methods overview

Edge-based methods

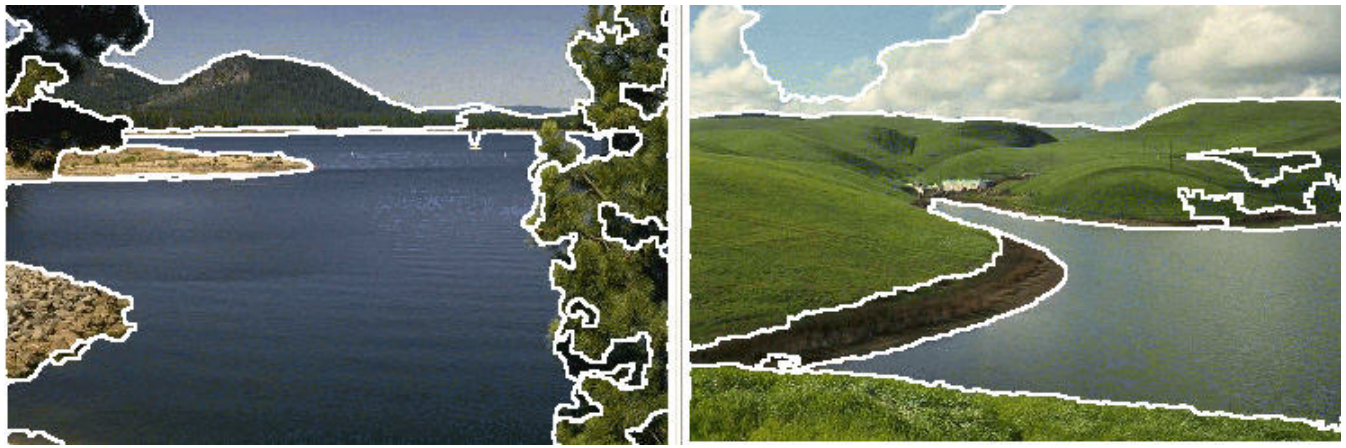
- look for discontinuities

Region-based methods

- look for similarity regions according to some criteria (it's a more general formulation)



Unsupervised image segmentation methods overview



Region based methods

Thresholding (followed by connected components computation)

Region growing

Morphological watersheds

Clustering methods (K-means, mean shift, ...)

May be applied on more complex higher dimensional feature spaces

Graph based energy minimization methods (graph cuts, ...)

Image segmentation: definition

Let R be the spatial region occupied by an image

Image segmentation may be defined as the process of partitioning *R* into n subregions

$$\cup_{i=1}^{n} R_i = R$$
$$R_i \text{ is a connected set } i = 1, \ldots, n$$
$$R_i \cap R_j = \emptyset \text{ for all } i \text{ and } j, i \neq j$$
$$Q(R_i) = \text{ TRUE } i = 1, \ldots, n$$
$$Q(R_i \cup R_j) = \text{ FALSE } \text{ for any adjacent region } R_i \text{ and } R_j$$

Q is a logical predicate defined over the points in a set, for instance: Q(A)=TRUE if all pixels in A have the same intensity level

A thresholding method we already know: motion segmentation

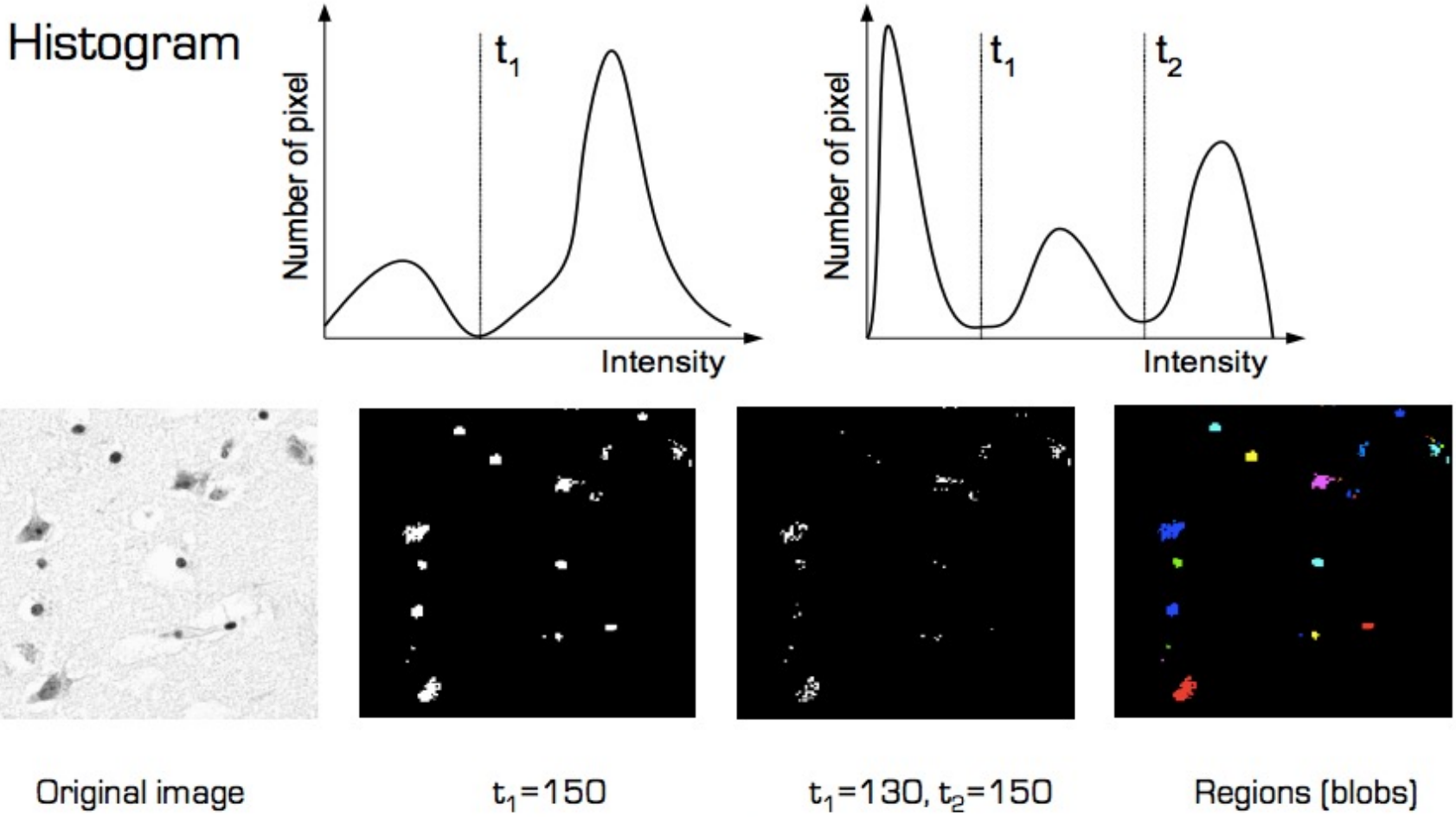
We identify segments of the image of moving pixels

It’s binary segmentation as we are dealing with 2 classes only (moving - not moving)

Image segmentation by thresholding + connected components

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{otherwise} \end{cases}$$

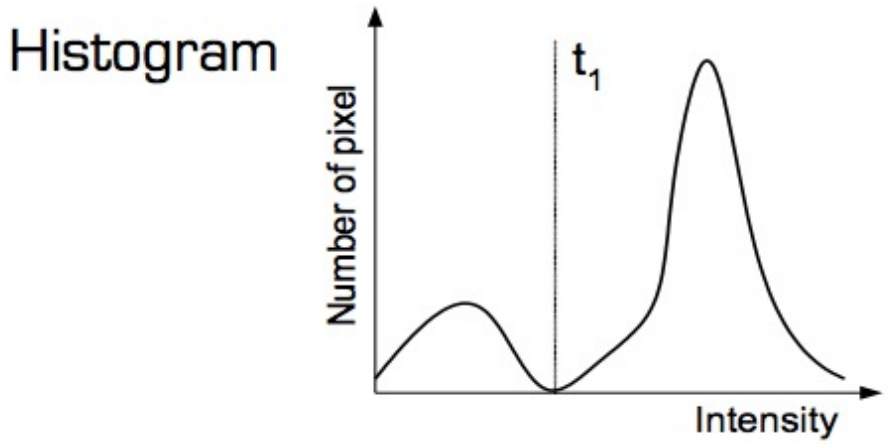
$$g(x,y) = \begin{cases} a & \text{if } f(x,y) > T_2 \\ b & \text{if } T_1 < f(x,y) \le T_2 \\ c & \text{otherwise} \end{cases}$$



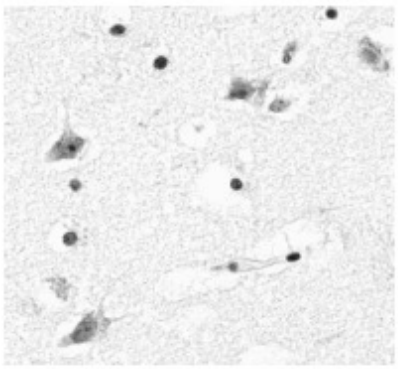
Credit: Markus Vincze, Technische Universität Wien

Image segmentation
the classical binary case

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{otherwise} \end{cases}$$



Foreground / Background separation



Original image

t_1=150

Automatic choice of a global threshold

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{otherwise} \end{cases}$$

How to choose T automatically?

1. **Select an initial estimate of *T***
2. **Segment the image using *T***
   - *G1 are pixels so that f(x,y) > T*
   - *G2 the others*
3. **Compute the mean intensity values in G1 and G2, let them be m1 and m2**
4. **Compute a new threshold**

$$T = \frac{1}{2}(m_1 + m_2)$$

5. **Repeat steps 2 to 4 until T does not change any more (or the change is small)**

It works well to find automatically "clear" valley in the histogram between background and a foreground

OTSU'S METHOD FOR GLOBAL THRESHOLDING

- **Objective: segment foreground/background by maximizing the *between class variance***

- **All computations are performed on the normalized histogram of the image**
  - The value of each bin is pi

- **Let us consider a threshold T(*k*) *0 < k < L-1* (L is the maximum value in the range, eg 255) and use it to segment image pixels in two classes:**
  - C1 pixels in the range *[0,k]*
  - C2 pixels in the range *[k+1,L-1]*

A very well known algorithm grounded on statistical discriminant analysis

OTSU'S METHOD FOR GLOBAL THRESHOLDING

- The probability that a pixel is in C_1 is $P_1(k) = \sum_{i=0}^{k} p_i$
- Similarly, the probability that a pixel is in C_2 is

$$P_2(k) = \sum_{i=k+1}^{L-1} p_i = 1 - P_1(k)$$

- The mean intensity value of pixels assigned to class C_1 is

$$m_1(k) = \frac{1}{P_1(k)} \sum_{i=0}^{k} i p_i$$

- Similarly

$$m_2(k) = \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} i p_i$$

OTSU'S METHOD FOR GLOBAL THRESHOLDING

$$m_G = \text{ global mean } = \sum_{i=0}^{L-1} ip_i$$

– Verify by substitution that

$$P_1 m_1 + P_2 m_2 = m_G$$
$$P_1 + P_2 = 1$$

OTSU'S METHOD FOR GLOBAL THRESHOLDING

– The optimal threshold can be found as follows

$$k^* = \arg \max_{0 \le k \le L-1} \eta(k)$$

with

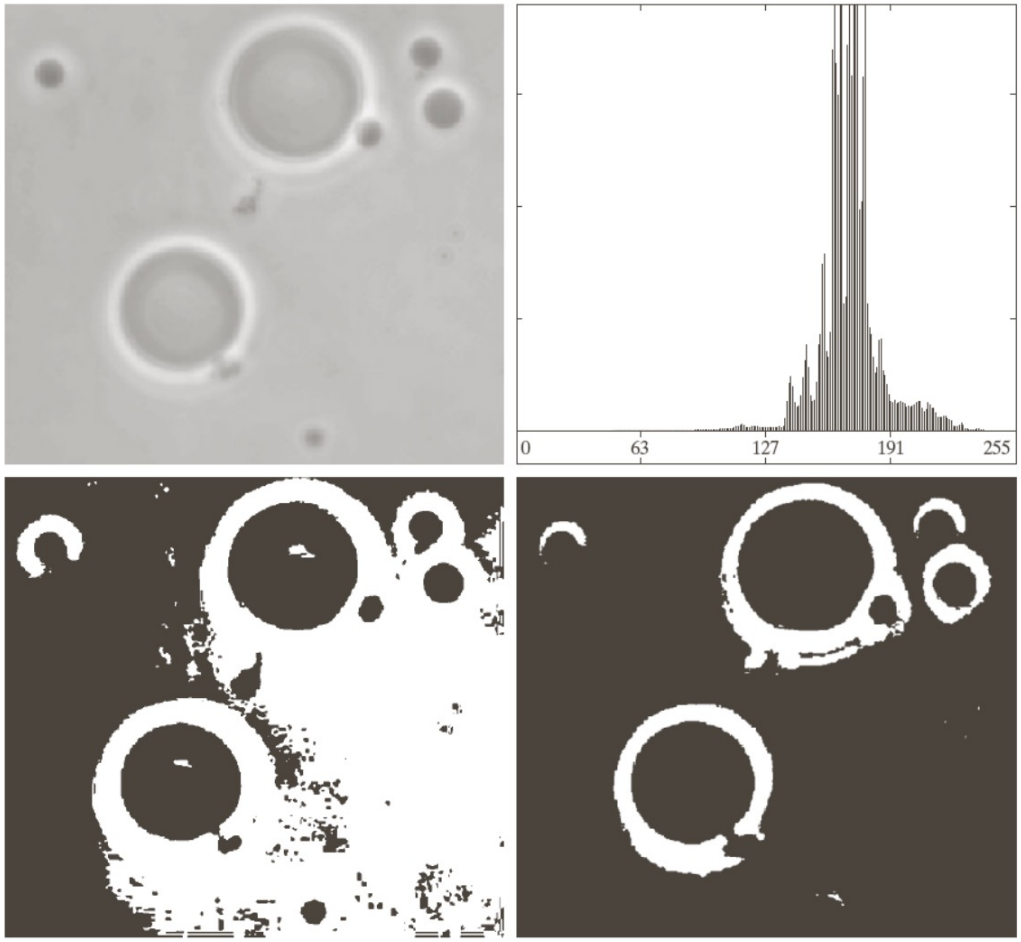
$$\eta(k) = \frac{\sigma_B^2(k)}{\sigma_G^2}$$

where

$$\sigma_G^2 \text{ (global variance)} = \sum_{i=0}^{L-1} (i - m_G) p_i$$

$$\sigma_B^2 \text{ (between class variance)} = P_1(m_1 - m_G) + P_2(m_2 - m_G)$$

OTSU'S METHOD FOR GLOBAL THRESHOLDING



a	b
c	d

(a) original image;

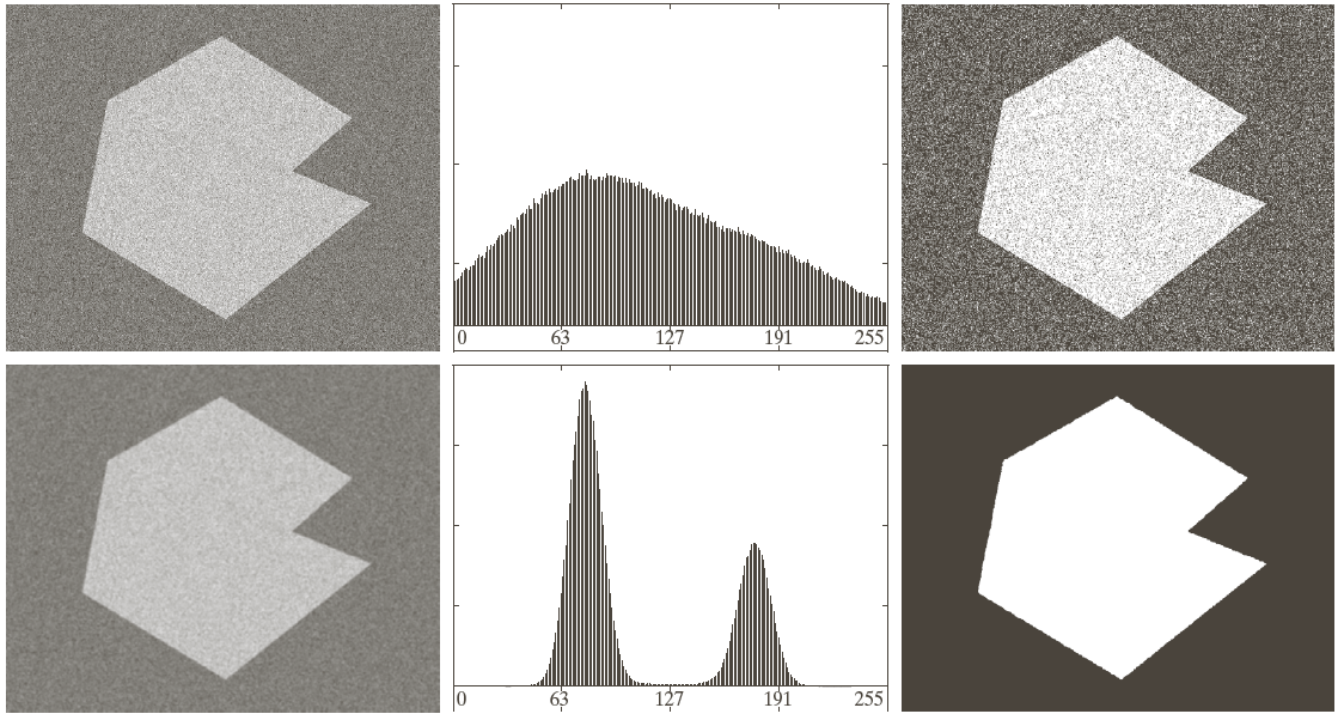
(b) histogram of (a);

(c) global threshold: $T = 169$, $\eta = 0.467$;

(d) Otsu's method: $T = 181$, $\eta = 0.944$.

IMAGE SMOOTHING TO IMPROVE GLOBAL THRESHOLDING

Original (noisy)

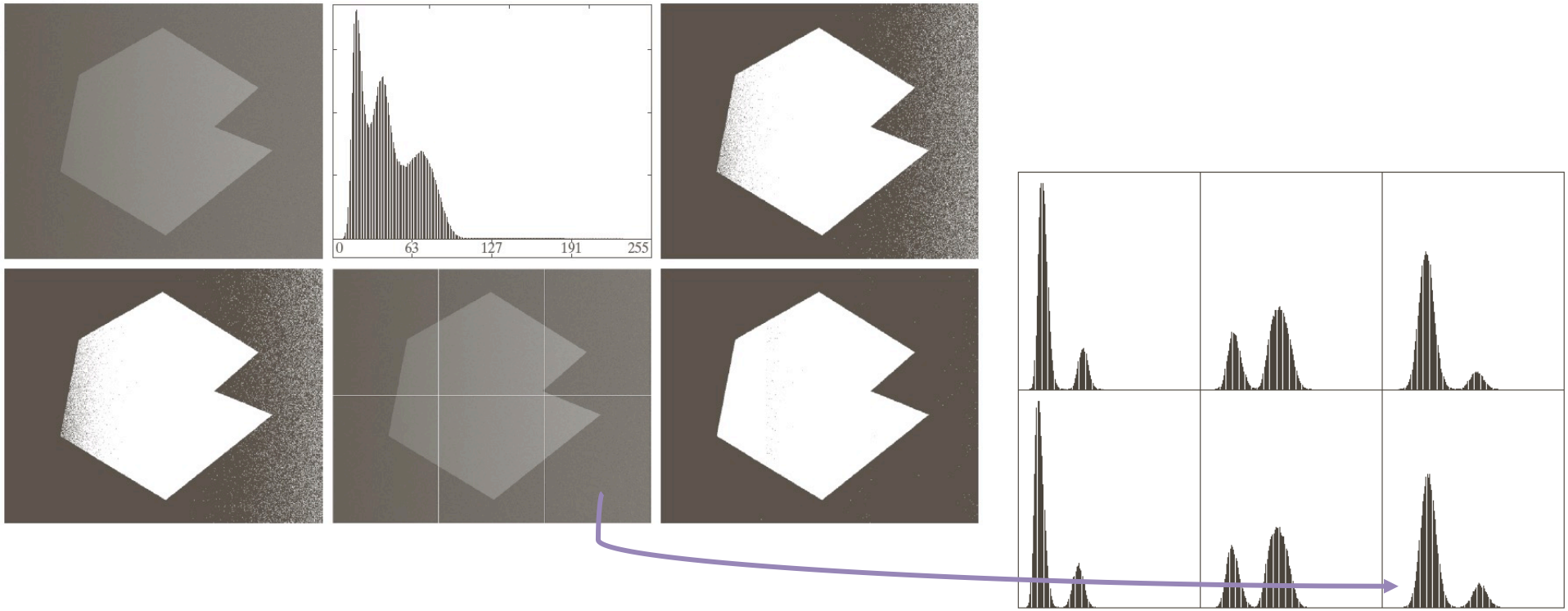


Gaussian smoothing

VARIABLE THRESHOLDING

– Objective: choosing different thresholds for different image parts

– Simple idea: subdivide an image into non overlapping rectangles



VARIABLE THRESHOLDING FROM LOCAL IMAGE PROPERTIES

- Compute a threshold T_{xy} in each point (x,y) based on its neighbourhood properties
- A simple example of this approach is the following

$$T_{xy} = a\ \sigma_{xy} + b\ m_{xy} \qquad a, b \geq 0$$

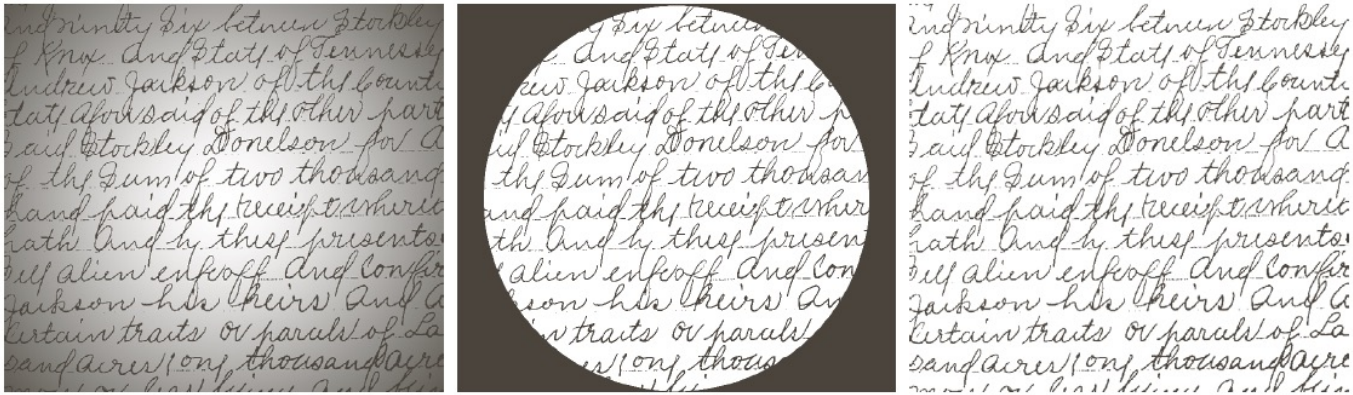
VARIABLE THRESHOLDING BY MOVING AVERAGE

- **This approach has been widely adopted for text segmentation**
  - There is a uniform distribution of text and background over space

$$m_{k+1} = \frac{1}{n} \sum_{i=k+2-n}^{k+1} z_i = m(k) + \frac{1}{n}(z_{k+1} - z_{k-n})$$

with $T_{xy} = b\ m_{xy}$

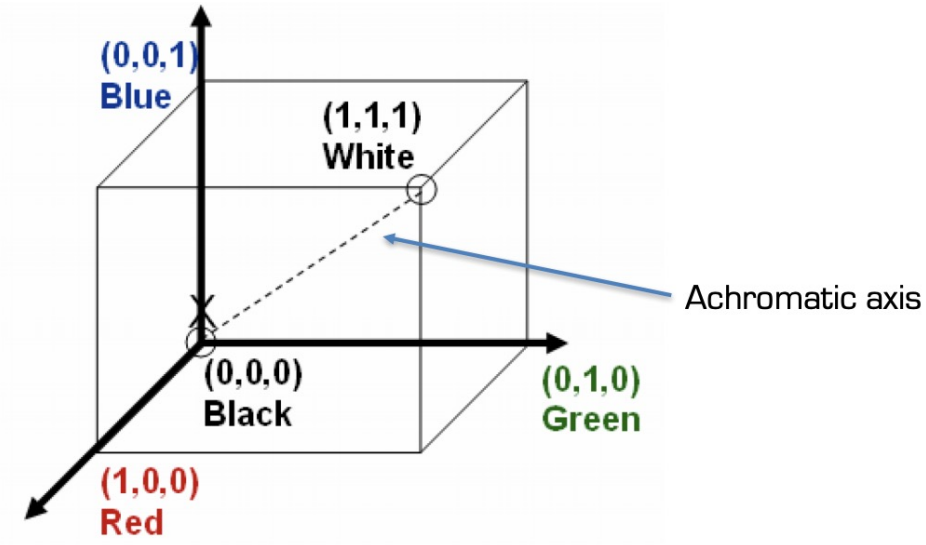
Only the last n visited pixels are used to estimate the statistics



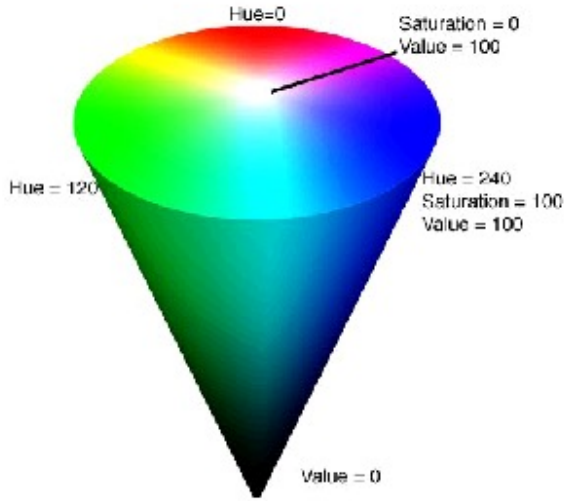
global local

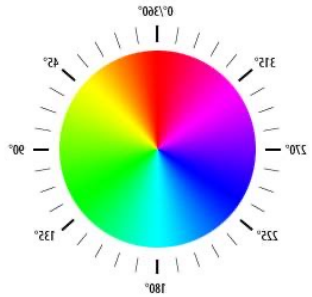
Color spaces: quick review

RGB model



HSV and other "perceptual spaces"





Color segmentation

NOTICE: Here the input space is higher dimensional!

Color-based thresholding: one possibility

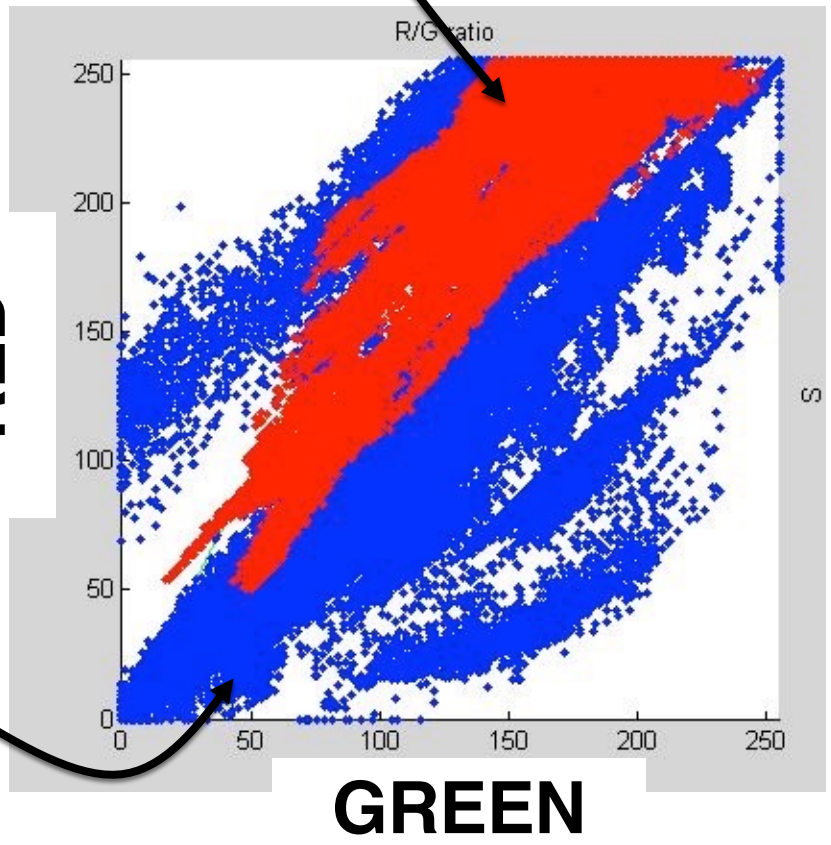
1. Smooth the image
2. Transform to HSV space
3. (Discard V)
4. Choose thresholds for H and S (notice they have different ranges!) appropriate for the task you are trying to address (eg detecting red objects)

Color thresholding - example

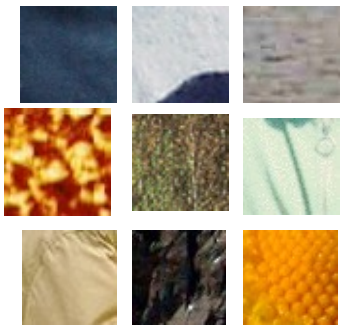
SKIN SEGMENTATION

Training / automatic
Threshold detection phase

positive (skin) examples



negative (non skin) examples



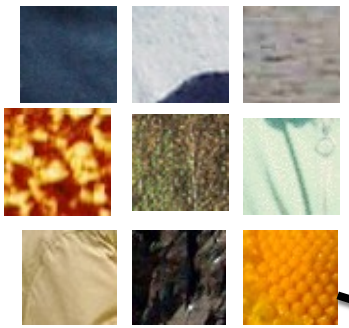
Color thresholding - example

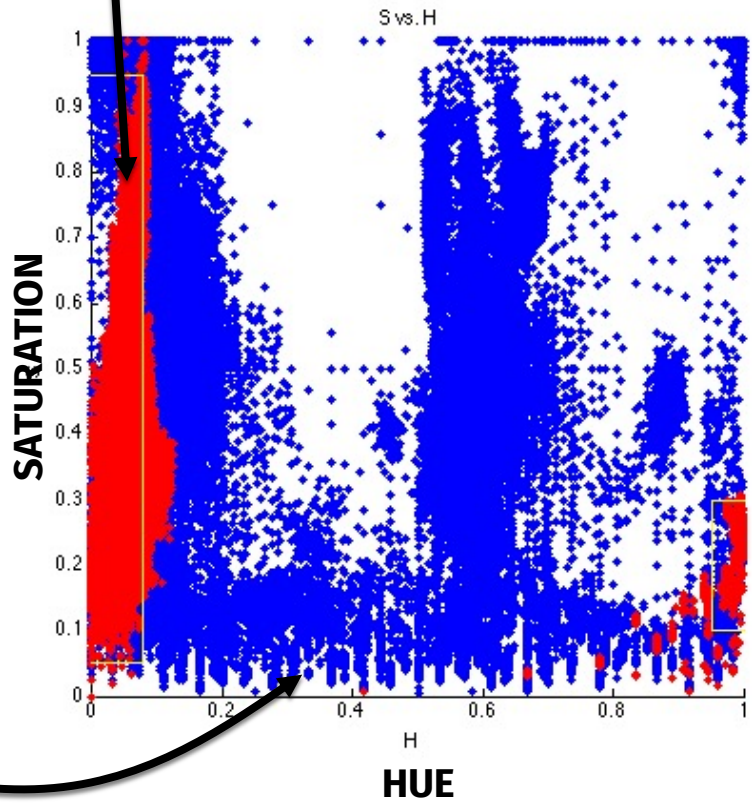
SKIN SEGMENTATION

Training phase

positive (skin) examples

negative (non skin) examples

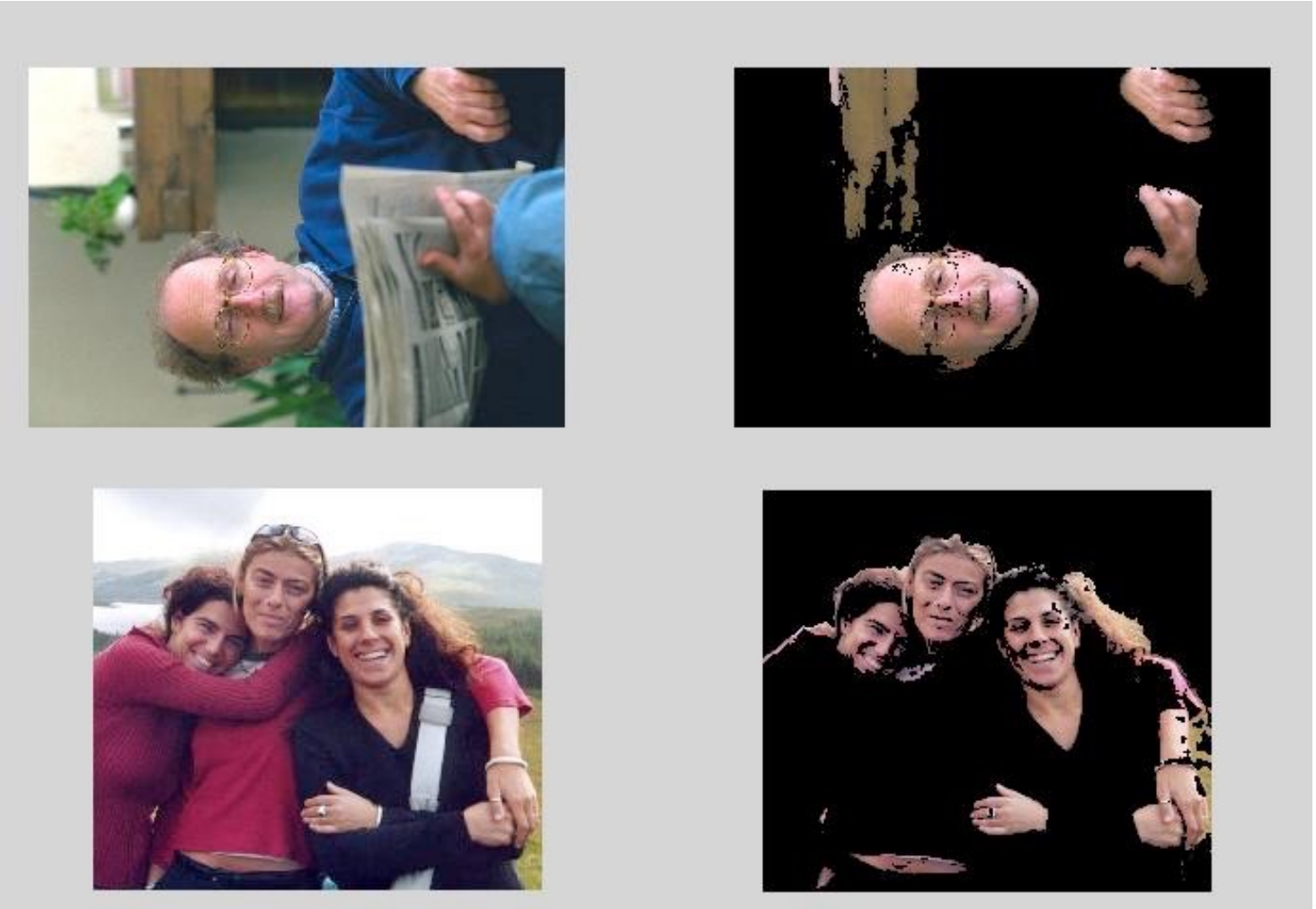




Color thresholding - example

SKIN SEGMENTATION

Test



Clustering methods

What if … you don't have a specific color in mind?

For more generic color segmentation: image pixels can be grouped in clusters of elements with similar colors

A classical choice is K-means, you already met several times

K-means clustering reminder

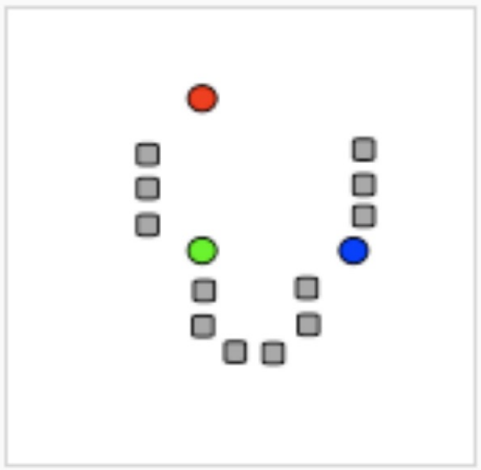
Given a set of data (x_1, … , x_n) the goal is to partition it into K sets

$S=(S_1, \ldots S_K)$ so to minimize the within-cluster variance:

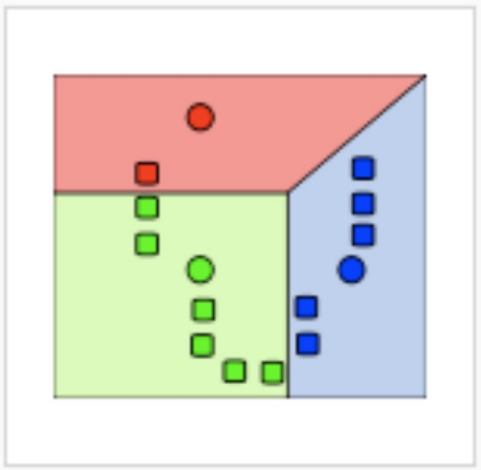
$$\arg\min_{\mathbf{S}} \sum_{i=1}^{K} \sum_{\mathbf{x} \in S_i} ||\mathbf{x} - \mu_i||^2$$

Iterative technique

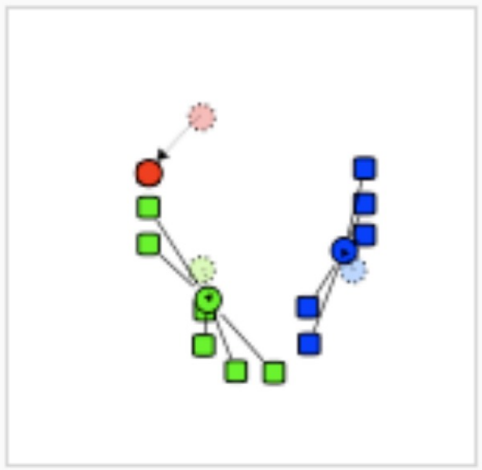
Input: set of unlabeled data appropriately represented, number of clusters K



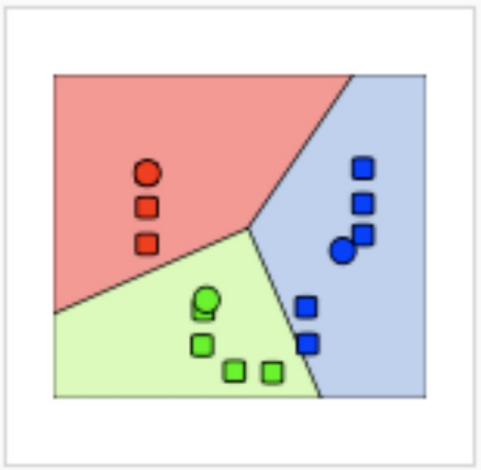
1. Random initialization



2. Points association

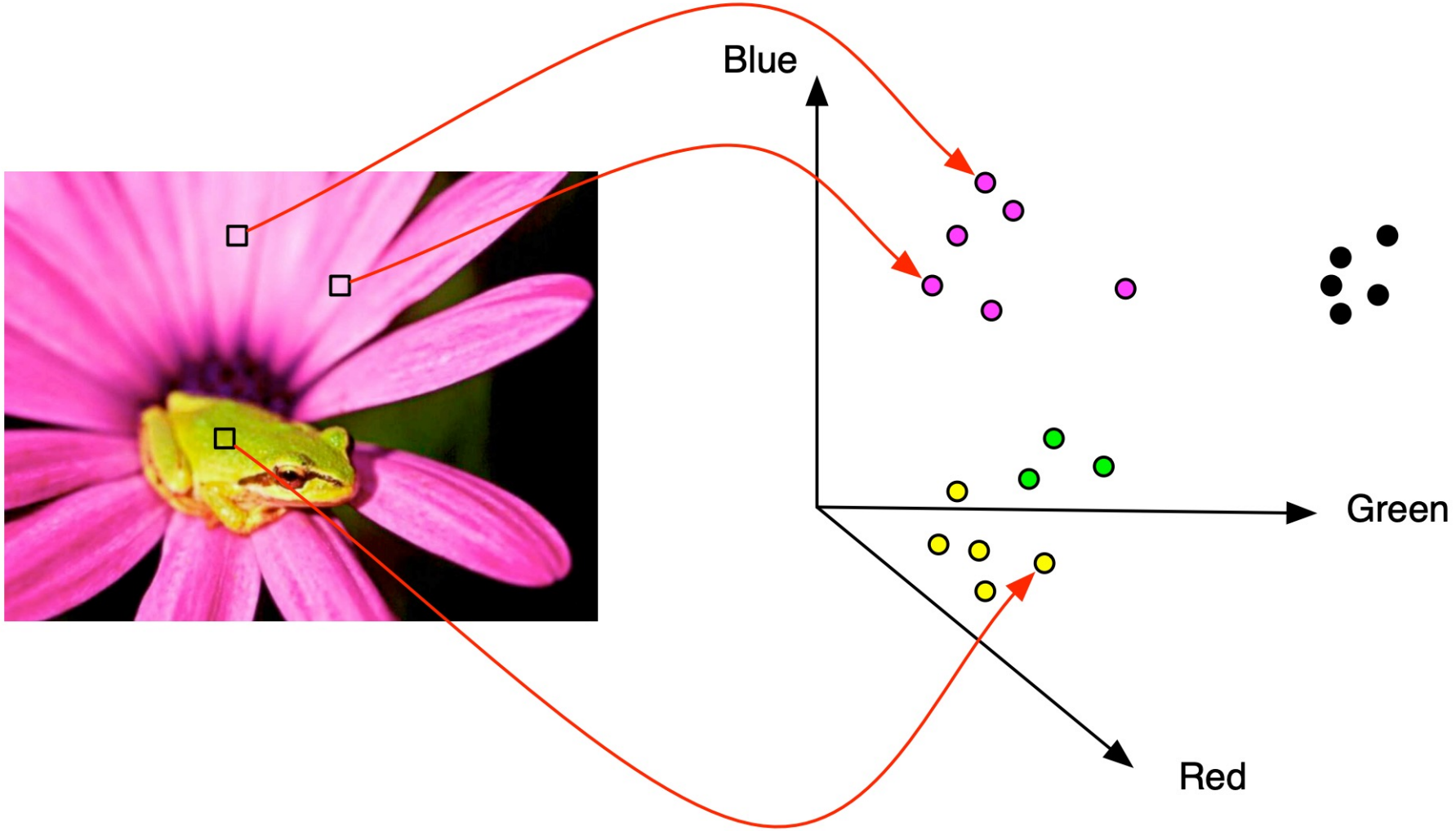


3. Centroids update



Steps 2 and 3 repeated until stability is reached

K-means clustering for image segmentation



If you want to consider also location information you may select a higher dimensional space (RGBXY)

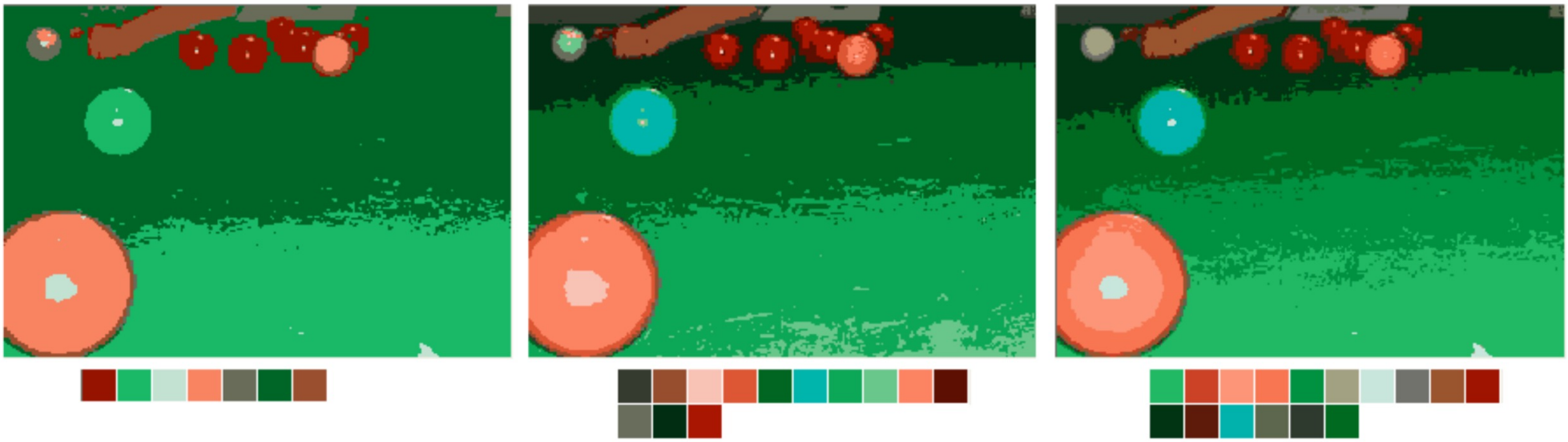
unsupervised image segmentation

The challenge is: how to choose K?

Try multiple K and search the ones that give the highest confidence by using a cluster quality indicator

eg Davies Bouldin index

This approach does not take into account the cluster size



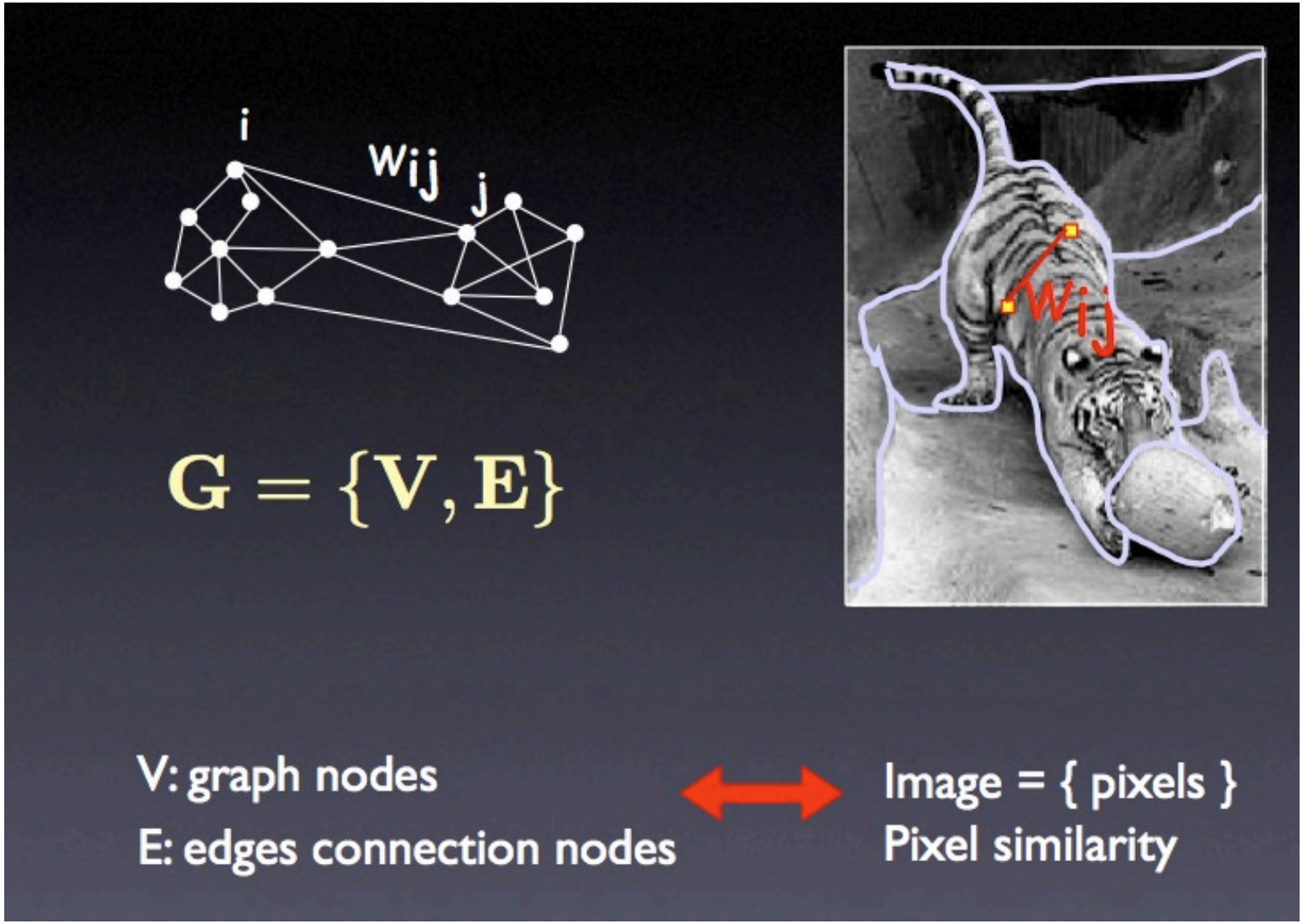
Kmeans image segmentation

Pros and cons

Pros – simplicity mainly

Cons – if we don’t have priors on the choice of K we may obtain non deterministic results. The method does not prevent the formation of unbalanced clusters

GRAPH BASED IMAGE SEGMENTATION



GRAPH BASED IMAGE SEGMENTATION

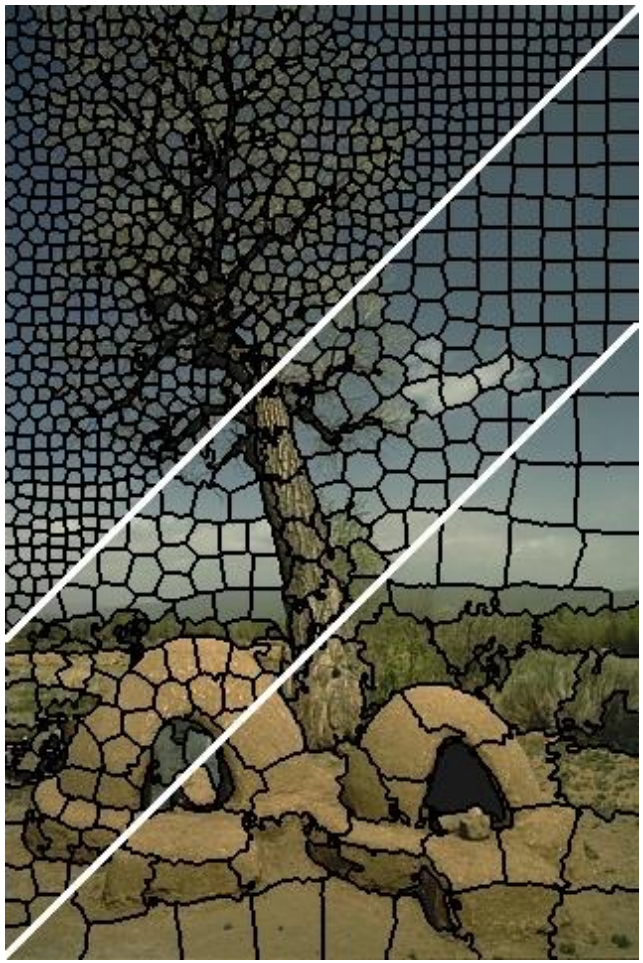
- **The goal of this approach is to devise a segmentation method that *"extracts a global impression of an image"* (Shi-Malik 2000)**
- **The segmentation process here becomes a graph partitioning problem**
- **This involved the following questions**
  - What is a precise criterion for a good partition? (min-cut, normalized-cut, …)
  - How can we make it computationally efficient? (the original normalized cuts problem definition resulted in a NP-hard task)
- **We will not proceed along this direction but if you're interested you find pointers in the Szeliski book (section 7.5)**

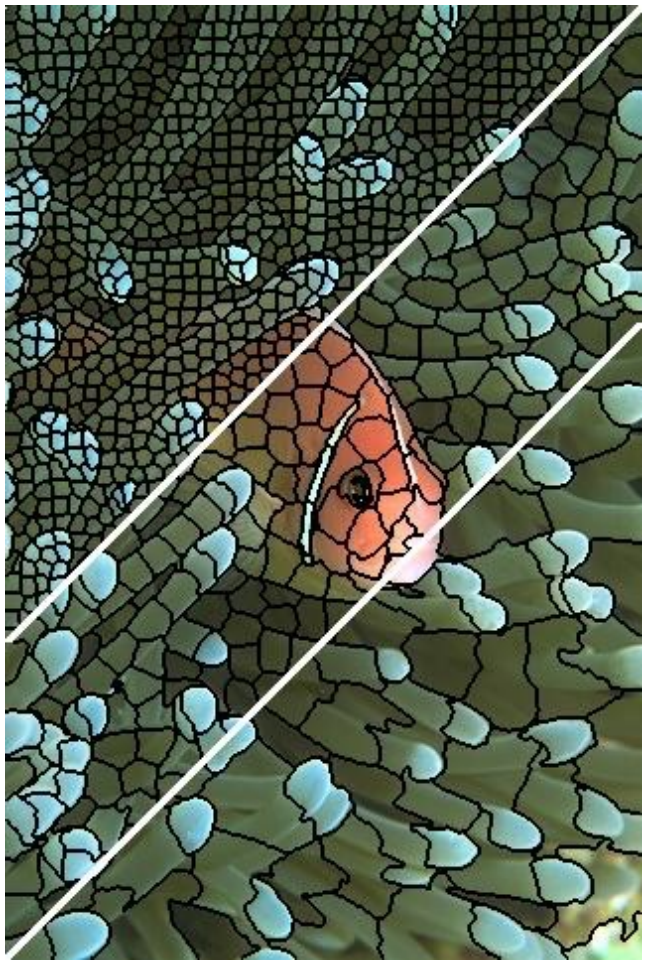
UniGe | MaLGa

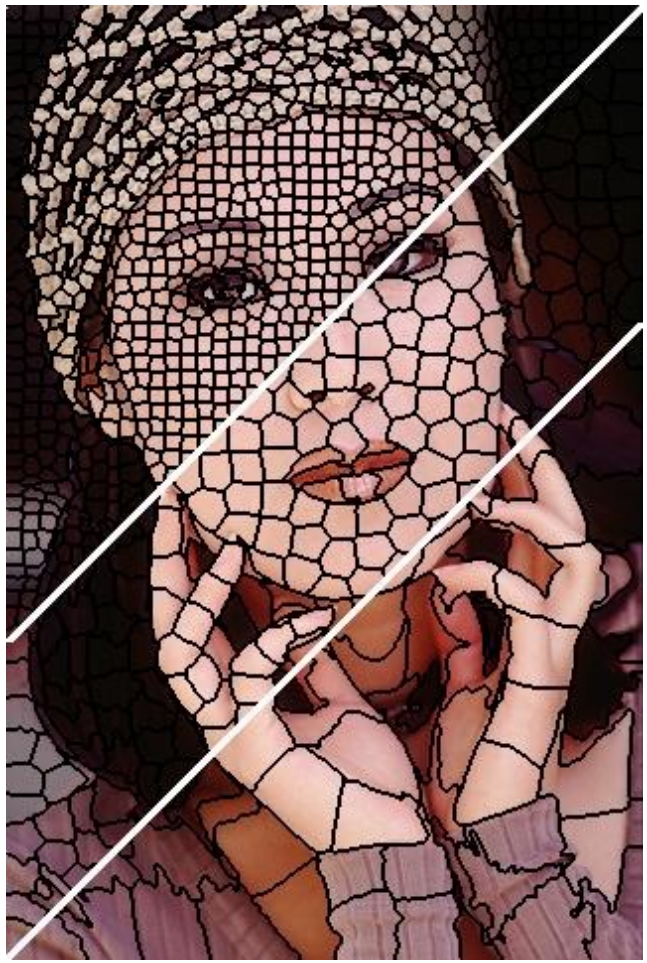
Superpixels computation

superpixels

small image parts characterized by internal uniformity



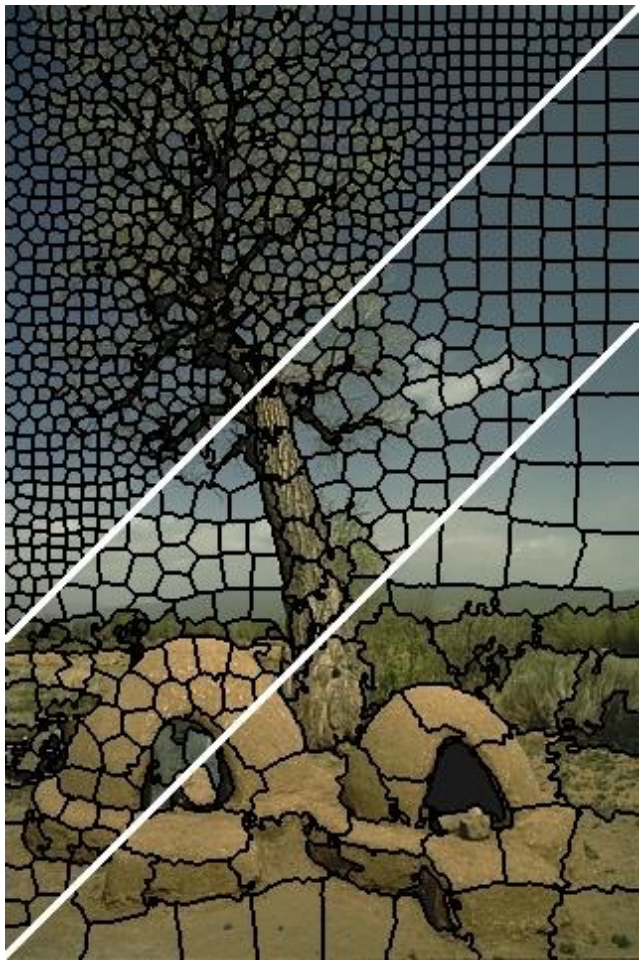


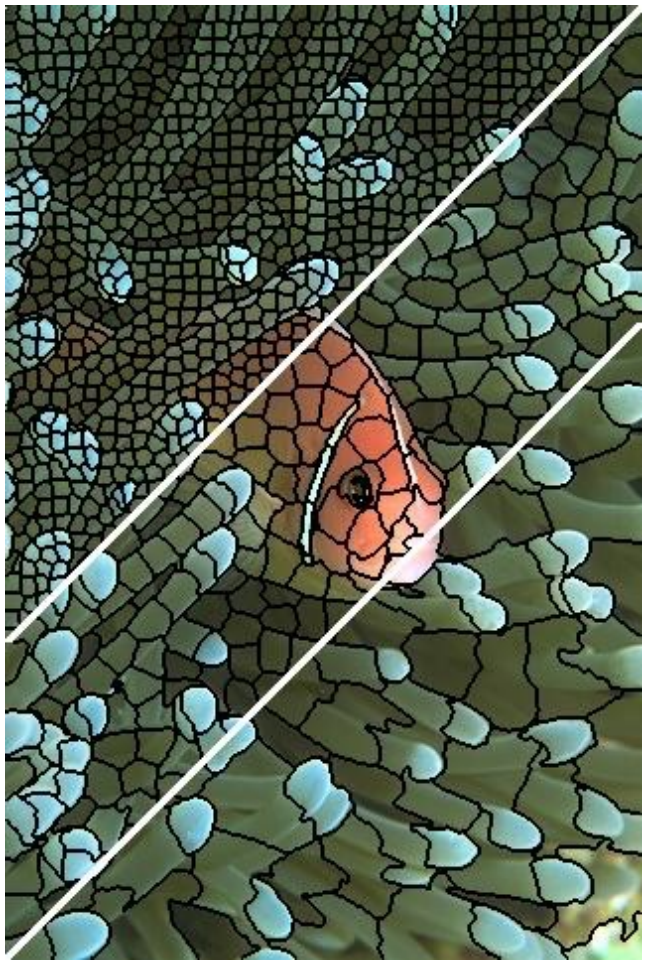


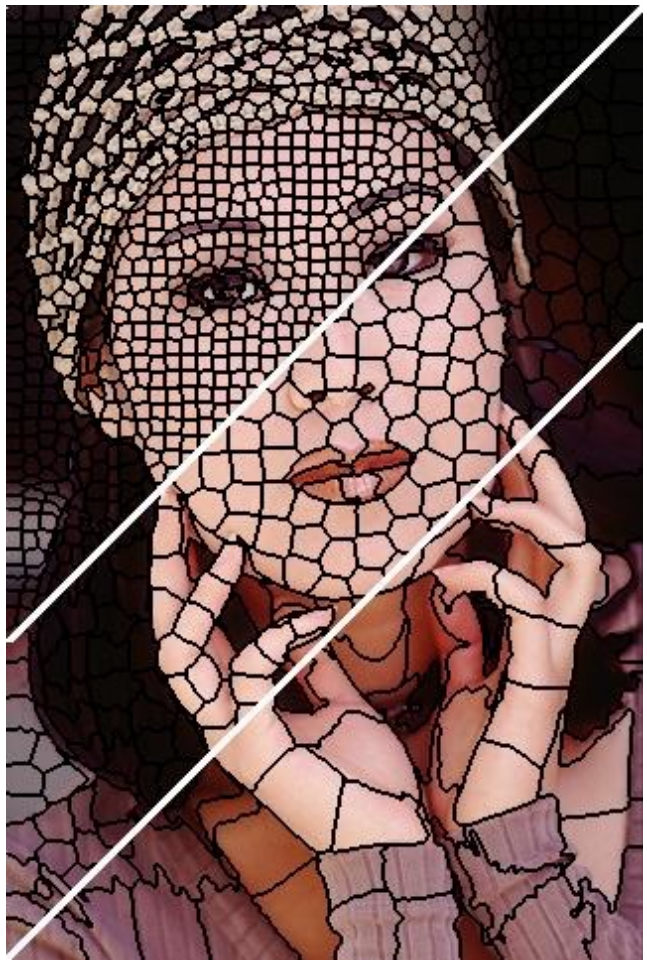
superpixels

they should be evenly distributed and should follow very well object boundaries

applications: a pre-processing step for higher level analysis (segmentation, classification).







SLIC - Simple linear iterative clustering

Simple and fast, based on Kmeans

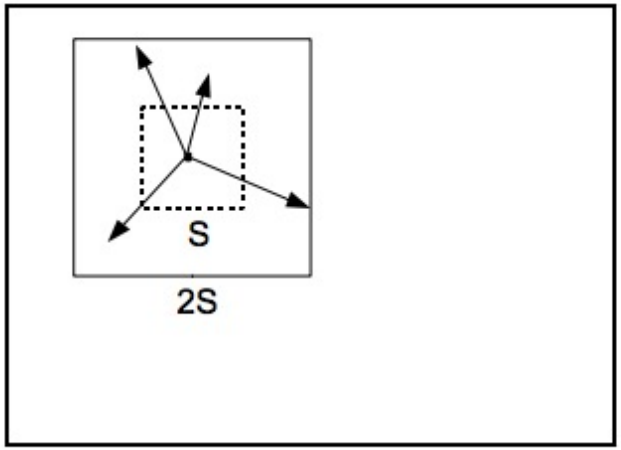
- search space limited to a region proportional to the desired super pixel size

- a weighted distance measure combines **color** and **spatial proximity**

Variants available, also for 3D (super-voxels)

SLIC SUPERPIXEL ALGORITHM

the only input parameter is the number of super pixels k



(b) SLIC searches a limited region

Algorithm 1 SLIC superpixel segmentation

/ Initialization */*

Initialize cluster centers $C_k = [l_k, a_k, b_k, x_k, y_k]^T$ by sampling pixels at regular grid steps S.

Move cluster centers to the lowest gradient position in a 3×3 neighborhood.

Set label $l(i) = -1$ for each pixel i.

Set distance $d(i) = \infty$ for each pixel i.

repeat

  / Assignment */*

  for each cluster center C_k **do**

    for each pixel i in a $2S \times 2S$ region around C_k **do**

      Compute the distance D between C_k and i.

      if $D < d(i)$ **then**

        set $d(i) = D$

        set $l(i) = k$

      end if

    end for

  end for

  / Update */*

  Compute new cluster centers.

  Compute residual error E.

until $E \leq$ threshold

UniGe