Digital Image Processing

1 February 2007

Dr. ir. Aleksandra Pizurica
Prof. Dr. Ir. Wilfried Philips

Aleksandra.Pizurica @telin.UGent.be
Tel: 09/264.3415

UNIVERSITEIT GENT
Telecommunicatie en Informatieverwerking

INSTITUUT VOOR PERMANENTE VORMING

Image analysis

Introduction

Image analysis often begins with

•edge detection - detecting object boundaries

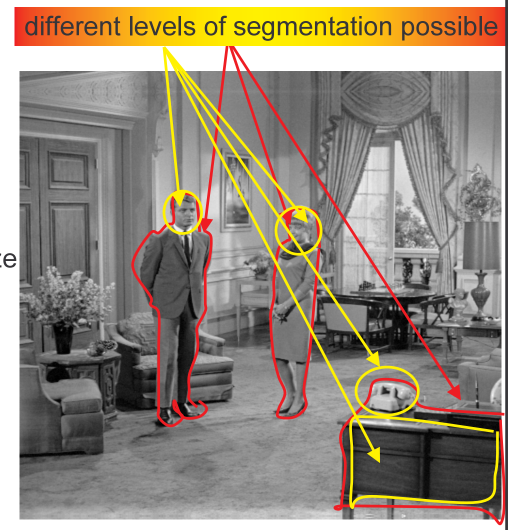
•image segmentation (grouping pixels that belong to the same object)

When image is segmented into regions, we can extract and analyze some characteristic features of the different regions, such as texture, shape, size,…

Some problems

•Objects are not always well-defined. How do we want to segment a given image? Depends on the applications.

•It is not always easy to recognize objects in an image



Adaptive thresholding

Thresholding

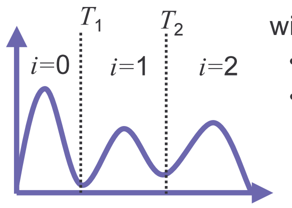
Homogeneity criterion:

- Divide gray values in a number of intervals $[T_i, T_{i+1}[$
- Homogeneous ⇔ all pixels values are within the same interval

Choice of the threshold values T_i

- If the histogram consists of different "modes": manually chosen thresholds between the modes
- better: minimizing mean intra group variance

$$\sigma_w^2 = \sum_{i=1}^{N} N_i \sigma_i^2$$

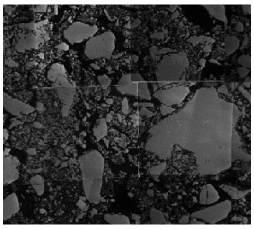


with

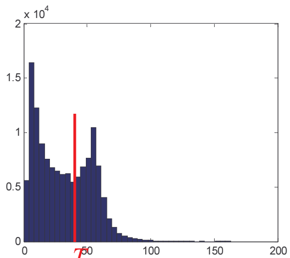
- N_i = number of pixels with gray values in $[T_i, T_{i+1}[$
- σ_i = deviation of the gray values of these pixels

Thresholding works only if the histograms of different object (classes) do not strongly overlap.

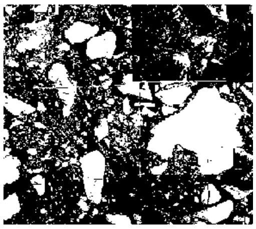
Global thresholding



input image



T_{opt}



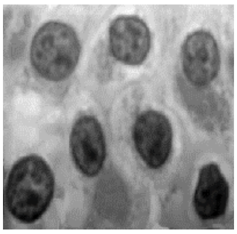
thresholded with T_{opt}

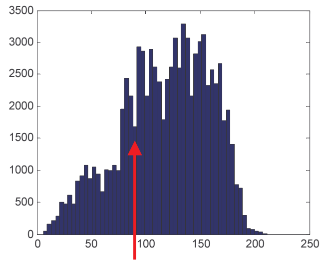
- Global thresholding is the simplest segmentation approach.
- Classifies image pixels into foreground (object) and background by comparing their value with a single threshold that is constant for the whole image.
- This simple approach works for some simple images that have a bi-modal histogram. The optimal threshold T_{opt} is in the valley between the two peaks.
- In reality this approach rarely works well alone and usually it is used as a part of a more complex and adaptive thresholding method

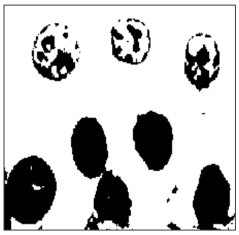
Where global thresholding does not work…

Global thresholding works only when histograms of the objects that need to be classified into different classes do not (strongly) overlap

In reality, histograms of the objects and the background overlap due to textures, noise and non-uniform illumination

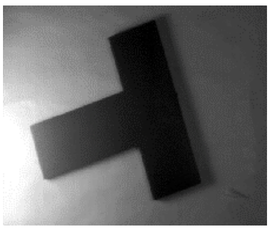


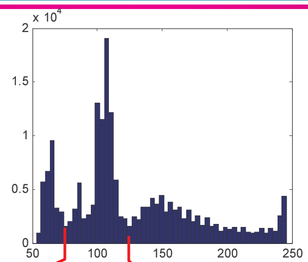


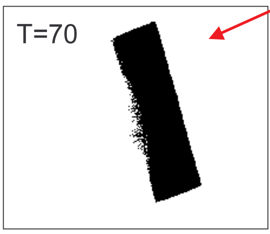


thresholded with T=90

…Where global thresholding does not work



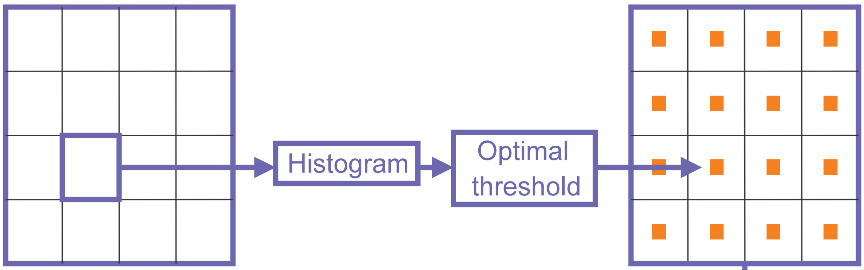


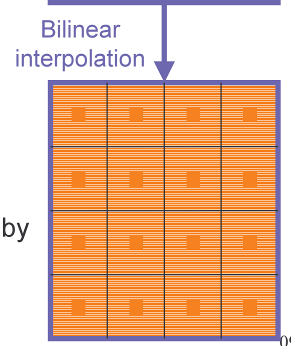


Object is still not extracted completely and the background is already included in the object → increasing threshold further is of no use

In this case of a strong non-uniform illumination we cannot find a single threshold that works well for the whole image

Spatially-adaptive thresholding





Principle: spatially-varying thresholds $T_i(x,y)$

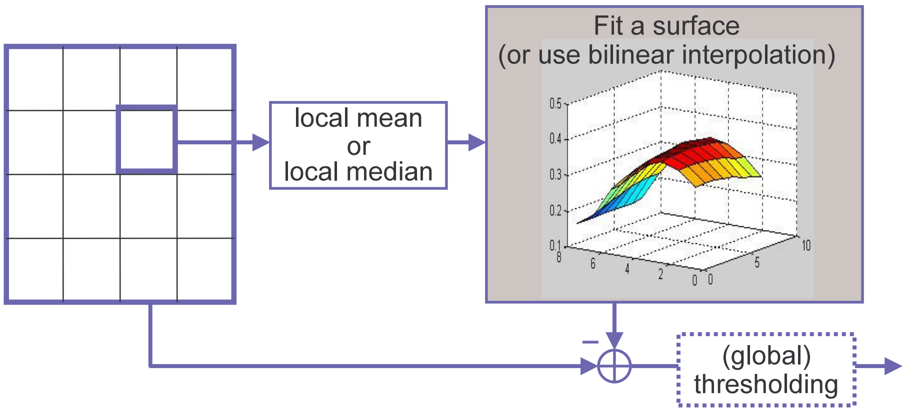
- Compute optimal thresholds $T_{i,k}$ for a number of image blocks centered around (x_k, y_k)
- Use these for central pixels:

$$T_i(x_k, y_k) = T_{i,k}$$

- Determine for other pixels (x,y) thresholds $T_i(x,y)$ by interpolation

In this way we take into account spatially varying image properties

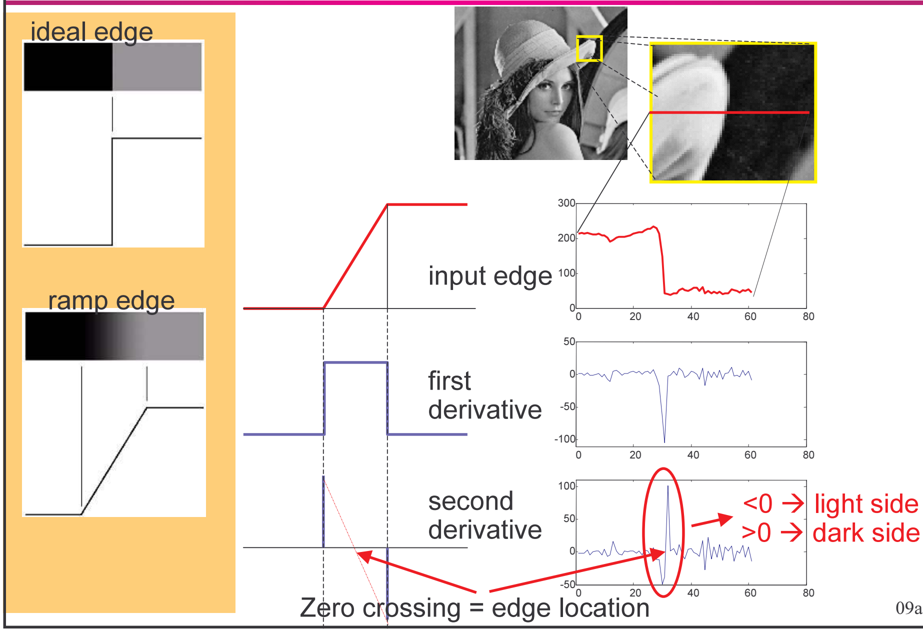
Removing nonuniform illumination



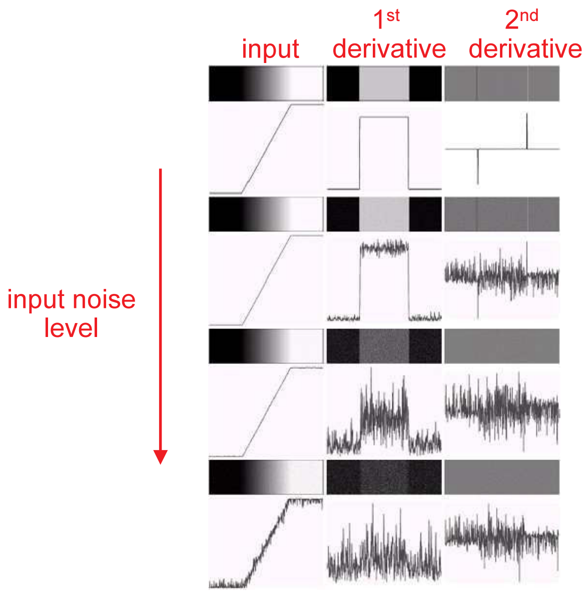
Spatially varying illumination can be effectively estimated by computing local mean (or median) gray value and fitting a surface through the computed points. This gives estimates of the illumination at each pixel. By subtracting the estimated surface (intensity profile) from the original image nonuniform illumination is removed.

Edge detection

Edge detection



Noise sensitivity of the derivatives



- The derivatives are highly sensitive to noise.
- Second-order derivative becomes unusable already at very small noise levels
- A possible solution is to “pre-filter” (smooth) the image first. A Gaussian smoothing kernel is often used for this

Gradient operators…

The gradient of an image $f(x,y)$ at position (x,y) is defined as the vector

$$\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

At image edges the gray value $f(x,y)$ strongly varies as a function of x and/or y $\Rightarrow$big partial derivatives $\Rightarrow$ big gradient magnitude

$$\|\nabla \mathbf{f}\| = \sqrt{G_x^2 + G_y^2}$$

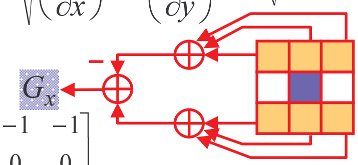
The direction of edge is perpendicular to the direction of the gradient vector

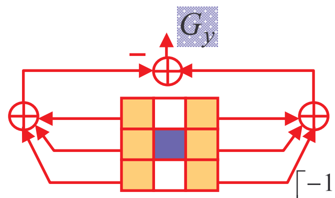
$$\alpha(x,y) = \tan^{-1}\left(\frac{G_x}{G_y}\right)$$

…Gradient operators…

Computation is based on digital approximations of the partial derivatives G_x, G_y

$$\|\nabla \mathbf{f}\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} = \sqrt{G_x^2 + G_y^2}$$





$$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

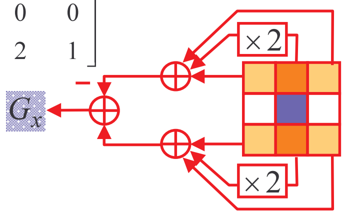
Prewitt operator

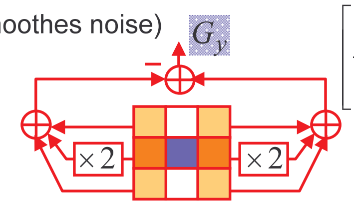
$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Sobel operator (smoothes noise)

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$





…Gradient operators…

- The Prewitt and Sobel masks are among the most frequently used. Sobel is somewhat more complex but less sensitive to noise
- True gradient magnitude is isotropic. The Prewitt and Sobel outputs not.
- Sometimes masks that give highest response for diagonal edges are used:

PREWITT

$$\begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

SOBEL

$$\begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

- The exact expression for the gradient magnitude

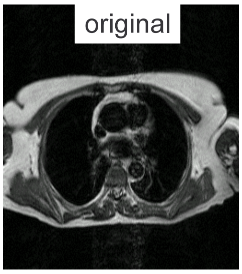
$$\|\nabla \mathbf{f}\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} = \sqrt{G_x^2 + G_y^2}$$

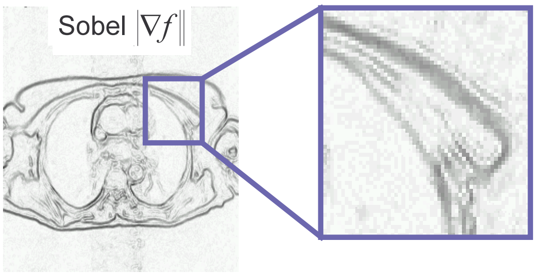
is in practice often approximated by

$$\|\nabla \mathbf{f}\| \approx |G_x| + |G_x|$$

which is easier to implement and preserves relative changes in gray level

Sobel-operator: Example 1

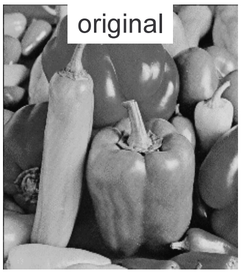


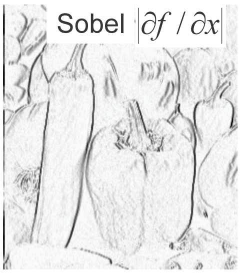


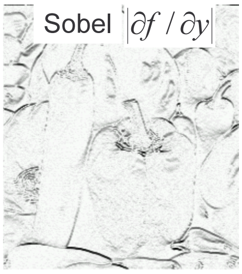
Typical situation: different edges of the same object have different strength (different gradient magnitude)

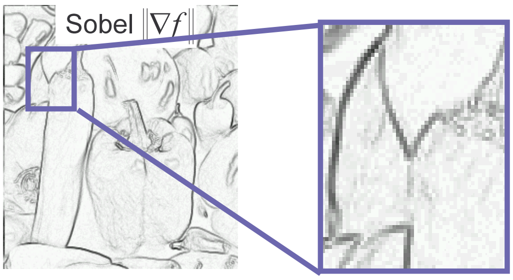
$\Rightarrow$ "leakage" between objects

Sobel-operator: Example 2









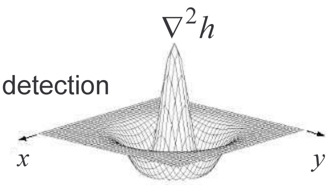
Typical situation: different edges of the same object have different strength (different gradient magnitude)

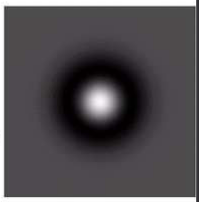
$\Rightarrow$ "leakage" between objects

Laplacian of Gaussian (LoG) edge detector

The Laplacian of a 2-D function f(x,y) is a second-order derivative operator

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

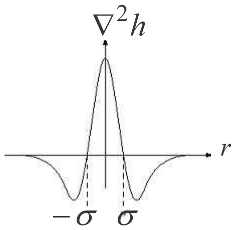




Laplacian is rarely used for edge detection
- too sensitive to noise
- produces double edges
- unable to detect edge direction

Its use (in combination with other edge detectors) consists mainly in exploiting zero-crossing for edge location

For edge location, Laplacian is often combined with a Gaussian smoothing function (to reduce noise sensitivity)



0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

A Gaussian (with negative sign):

$$h(r) = -e^{-\frac{r^2}{2\sigma^2}}$$ → Apply Laplacian → LoG: $$\nabla^2 h(r) = -\left[\frac{r^2-\sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}}$$

Canny edge detector

A well known edge detector developed by John F. Canny in 1986

input image

input parameters
standard deviation of the Gaussian kernel
two thresholds

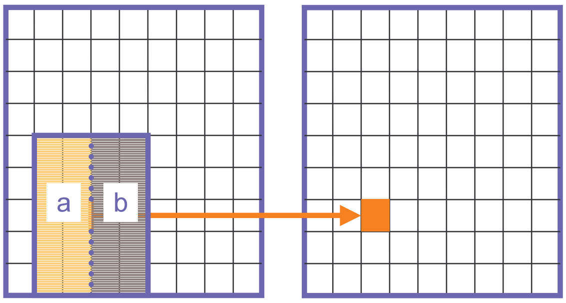
Noise suppression
Gaussian smoothing

Finding intensity gradients
Find Max and corresponding direction

Tracing edges
Detect strong edges using a high threshold

Trace weak edges (that are connected to a strong one) using a low threshold and edge direction information

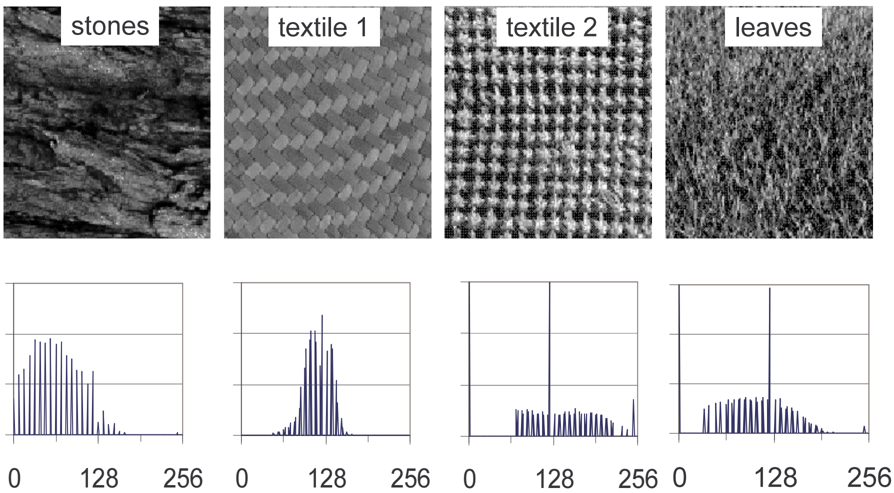
Nonlinear edge detectors



Now we detect edges between pixels instead of at pixels

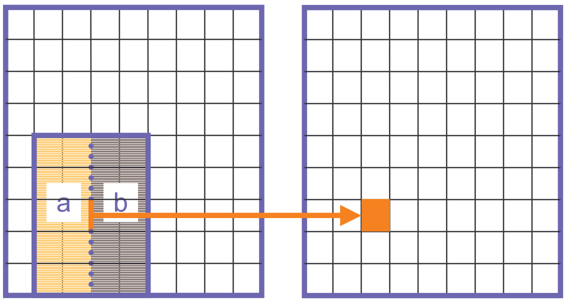
Non linear detectors perform a statistical test for how likely its is that the (most) pixels in a en the (most) pixels in b belonfg to different objects
⇒the resulting edge plausability takes over the role of a partial derivative
This test has to be repeated for different orientations

Edge detection based on histograms...



Different textures often have histograms of different form

… Edge detection based on histograms



$$G = \frac{1}{N}\sum_{i=0}^{n-1}\left|h_b(i) - h_a(i)\right|$$

N =number of pixels in a
= number of pixels in b

We try to detect a transition between two by comparing histograms in regions a and b

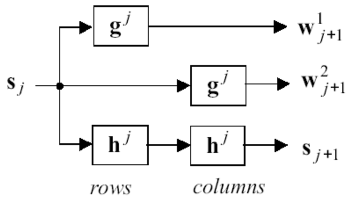
De edge plausibility $G \approx 0$ if the histograms do not (strogly) differ

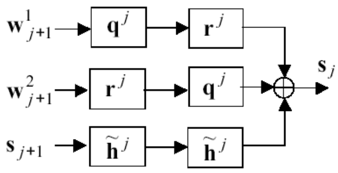
Remark: the grey values first have to be quantized to a sufficiently small number of levels n ($n << N$, otherwise G will always be ≈1 due to noise)

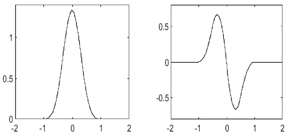
Disadvantage: fairly big regions needed to obtain a reliable result

Wavelet based edge detection

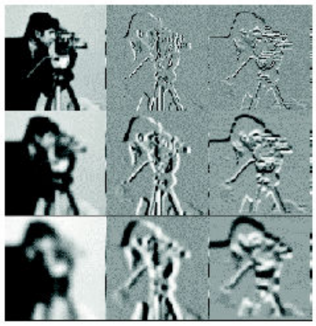
Edge detection wavelets







Quadratic spline wavelet

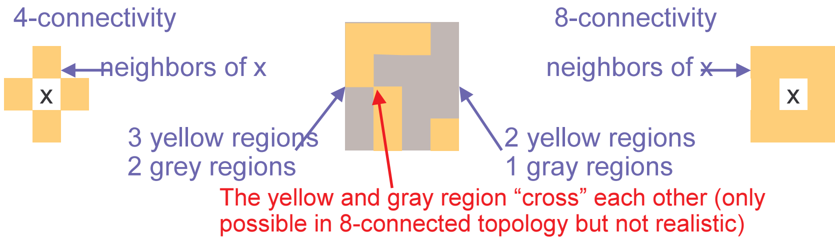


Segmentation

Segmentation

Goal: divide image into as big as possible spatially connected regions of pixels with similar properties

- A region is connected if any two pixels within the region can be linked by a trajectory of neighboring pixels within the region.

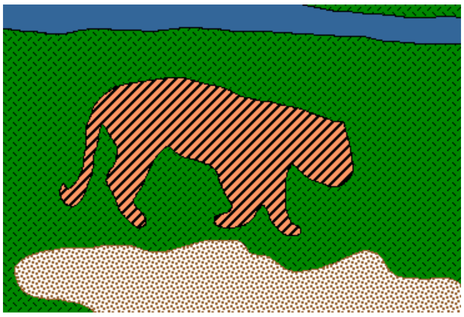


- Homogeneity criterion (similarity criterion)
  - Standard deviation of gray values within the region smaller than σ_m,
  - Difference in biggest and smallest gray value smaller than m,...
- The union of two regions does not satisfy the homogeneity criterion
- Solution is not necessarily unique!

Image Segmentation

Input image

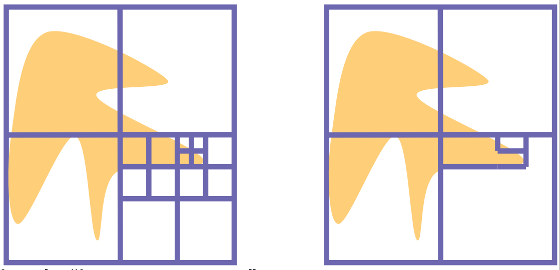
Desired segmentation



The homogeneity criterion is sometimes very complex
- Black lines on the brown tiger have to be described as homogeneous

The problem is not uniquely defined
- Where does the sand terrain end?
- Shadow of the tiger: a separate object or not?`

"Quadtree Split-and-merge" segmentation



"Splitting" step:

- Divide the image into 4 equal squares
- Divide each "not sufficiently homogeneous" square in 4
- Stop if all squares are sufficiently "homogeneous"
- The homogeneity criterion can be, e.g., that the variance of the gray values is smaller than a threshold

"Merging" step:

- Join all the neighboring regions if their union is sufficiently homogeneous

Disadvantages

- Not natural edges (only horizontal and vertical segments)
- The result of merging depends on the order in which the neighboring regions were searched for possibility to merge

A remark on homogeneity criteria

Criterion: "a region is homogeneous ⇔ the difference between the smallest and the biggest gray value does not exceed 2

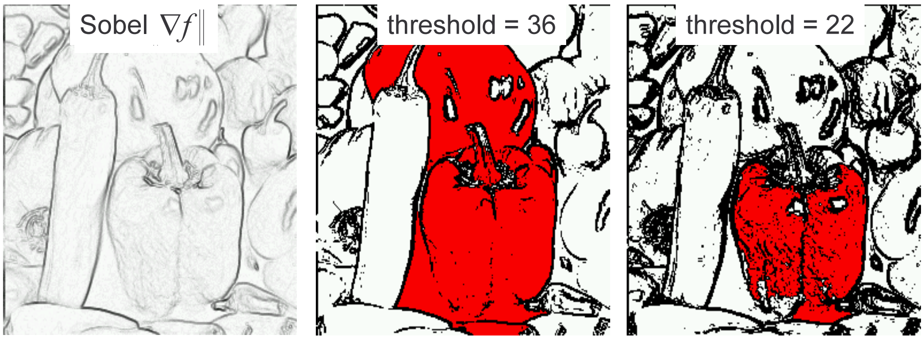
1	3	4	7
1	3	4	7
1	3	4	7
1	3	4	7

1	3	4	7
1	3	4	7
1	3	4	7
1	3	4	7

Segmentation is not unique!

The result of merging depends on the order in which the neighboring regions were searched for possibility to merge

From gradients to segments



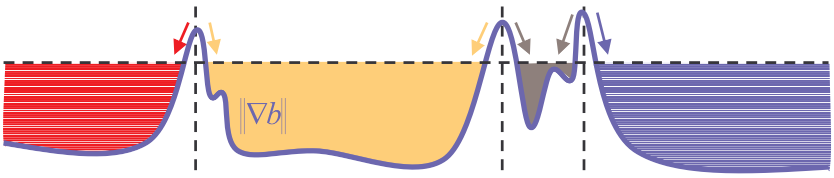
Principle

- Pixels with gradient above a threshold are denoted as edges
- Non-edge pixels belong to the same region if they can be linked by a path that does not contain any edge pixels

Disadvantages

- The edges are typically broader than 1 pixel
- Weak contours become broken for a too high threshold
- If the threshold is too low noise regions typically appear

Contour thinning



Contour thinning: edges that are broader than 1 pixel are mad thinner by assigning the edge pixels to the neighboring segments

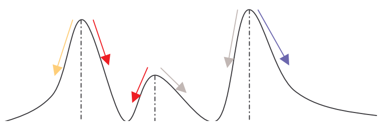
A possible approach: “watershed segmentation”

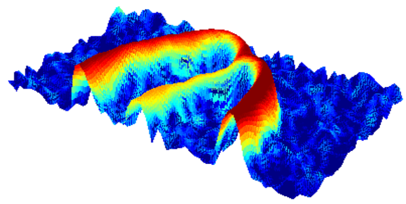
- Each edge pixel is associated with the segment that is reached by following the steepest descending trajectory in the gradient image

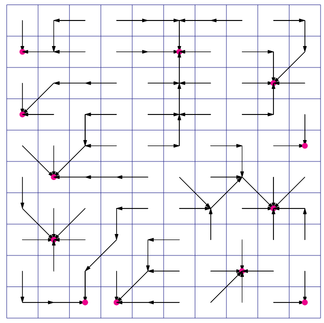
Interpretation: if we describe gradient image as a terrain height

- Then every segment corresponds to a valley and
- On each edge pixel we let a water drop fall and we assign the edge pixel to the valley where the drop flows

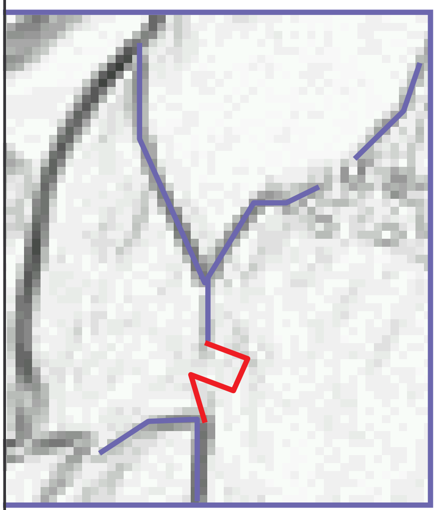
Watershed segmentation







Closing contours



Due to weak gradients gaps appear in the contours

We try to link the end points of an open contour with a neighboring contour

For this we investigate all possible trajectories and choose the one that minimizes a cost criterion

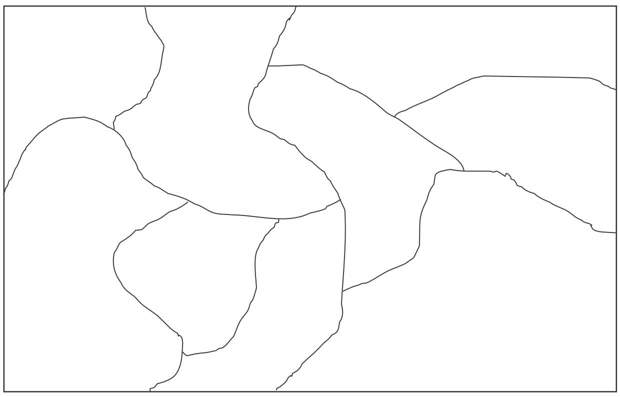
Possible cost criterion:

$$C = \lambda N - \sum_{i=1}^{N} \left\| \nabla b(x_i, y_i) \right\|$$

The criterion can also take into account smoothness of the connecting trajectories

Hierarchical segmentation

Hierarchical segmentation…

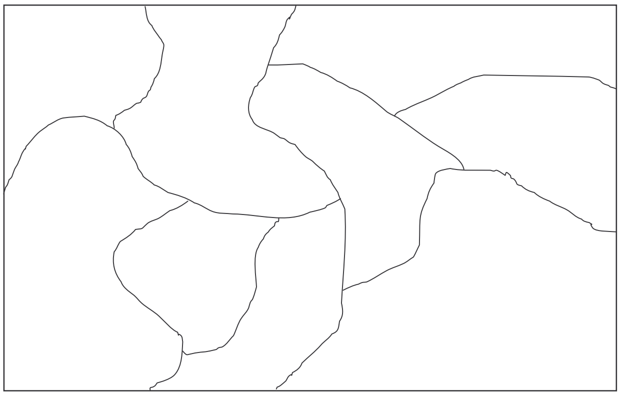


Hierarchical segmentation via “split” techniques

- •The image is first divided into a small number of big regions of arbitrary form
- •The regions are subsequently split

⇒a hierarchy of (potential) objects is created

… Hierarchical segmentation



In general “merge” techniques

- start from an image that is divided into a big number of small regions of arbitrary form
- The regions are joined into larger regions

⇒a hierarchy of (potential) objects is created

Use of a hierarchy of segmentations

Object recognition often needs features from different scales

- detect first a car
- Search then the number plate
- Recognize the letters on the number plate

Common problems in segmentation

- Over segmentation: one object divided into (many) regions
- Onder segmentation: different objects are wrongly grouped into one region

For an optimal compromise some regions will be oversegmented and others onder segmented ⇒ possible solution

- Compute a hierarchical segmentation
- Apply the segmentation interactively

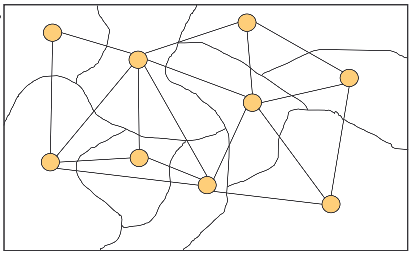
⇒ advantages of having a hierarchical segmentation:

- User has to decide only on: split or join
- No need for contours, … drawing ⇒ faster

Graph based representation

The segmentation on a given level of hierarchy can be described as a "region adjacency graph"

- The nodes are the segments
- The edges connect neighboring segments



"Costs" are asigned to graph edges

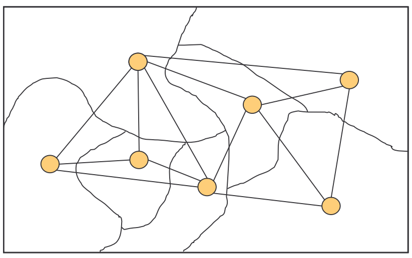
- The cost is low if two neighboring segments match well
  - similar colors, textures, … in their inside regions
  - and/or in the are of their joint edge
- or if joining the corresponding segments yields a"better" segmentation
  - less small regions
  - regions with a more regular shape
  - …

Graph based hierarchical segmentation

Region merging:

- Start from "microsegments" (very small regions that surely do not contain more than one object)
- Iteratively join two best matching segments

⇔ Deleting the edge with lower cost
and replacing two nodes at the end of that edge by one node



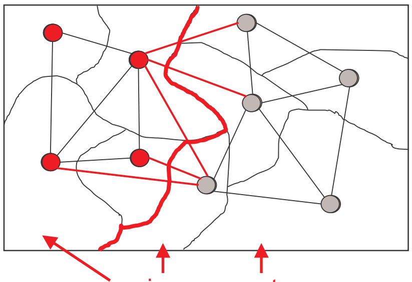
Graph based hierarchical segmentation

Graph partitioning

- Starting from microsegments" (very small regions that surely do not contain more than one object)
- Divide graph in two sub graphs by deleting edges
- And this such that splitting costs of the cutted edges is as small as possible (note: splitting cost instead of the joining cost)

⇔ determinig an optimal "cut" in the graph (e.g. "normalized cut")

Result: 2 segments that will be further split in the next step



Remark: the obtained macro segments in each phase correspond with a sub graph and not with a node

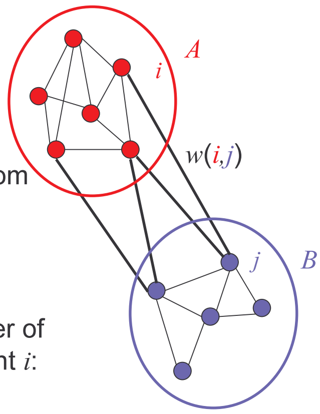
The metod of minimum cost cut…

Notation:

- V is de set of all nodes
- A is a set of nodes and $B=V\backslash A$ is the set of other nodes
- $C(X,Y)$: the sum of all costs of all edges that connect one node from set X and one node from set Y

Minimum cut method

- Search A and $B=V\backslash A$ zodat $C(A, B)$ is minimal
- Typical weighting factors, where $\mathbf{x}_i$ is the center of gravity and I_i the mean intensity of the segment i:



$$w(i,j) = e^{-\lambda_I \|I_i - I_j\|^2} e^{-\lambda_s \|\mathbf{x}_i - \mathbf{x}_j\|^2}$$

$$C(A,B) = \sum_{i\in A, j\in B} w(i,j)$$

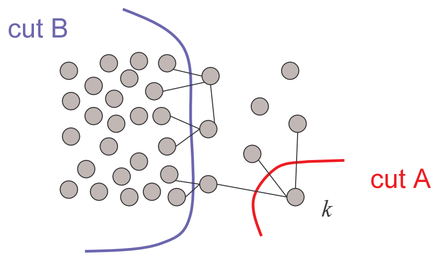
low split cost for microsegments with strongly different gray value

Low splitting cost for vmicrosegments that are far apart from each other

…The metod of minimum cost cut

Disadvantage: de “minimum-cost cut” method favors splitting off individual big micro segments → unbalanced hierarchy

Example: $w(i,j) = e^{-\lambda_s \|\mathbf{x}_i - \mathbf{x}_j\|^2}$

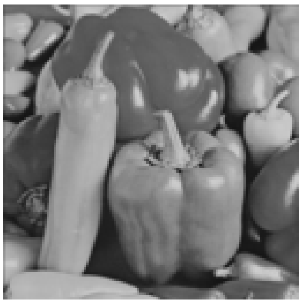


Segment k is far from all other segments $\Rightarrow$ cut A has much lower cost than cut B

Cut B is however better from the point of view of hierarchical splitting

- •a region left with a lot of small microsegments
- •a region right with only big microsegments

Examples: Normalized cut…


Original image


Micro segments


First cut

Remarks over this kind of “split” techniques

- •very slow
- •can be combined with merge techniques

… Examples: Normalized cut…

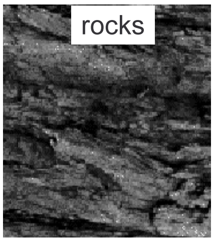
… Examples: Normalized cut

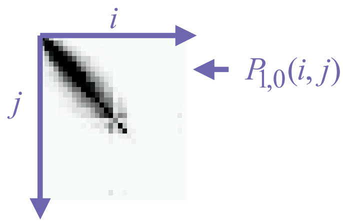
Original image | Micro segments | After several cut-steps

Segmentation based on textural features

Co-occurrence matrix

Segmentatie m.b.v. 2D histogrammen...





Two dimensional histogram ("co-occurrence matrix") measures the second order statistics of an image

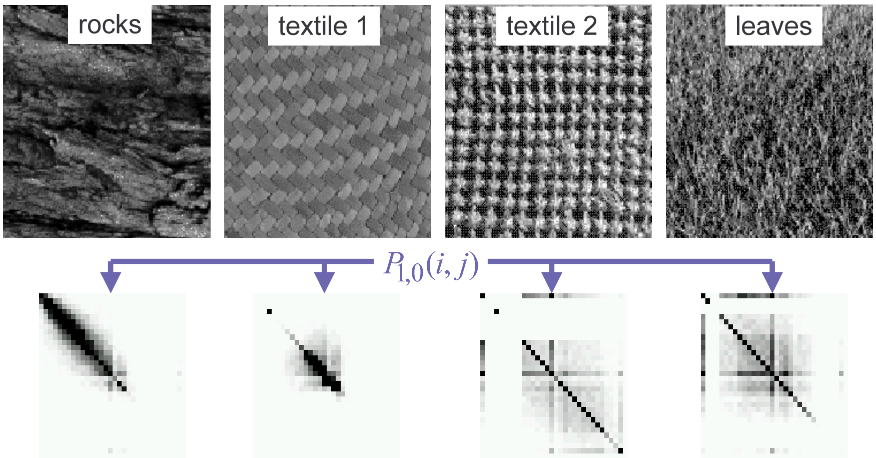
$$P_{s,t}(i,j) = \left(b(x,y) = i; b(x+s, y+t) = j\right)$$

Commonly used are $P_{1,0}(i,j)$, $P_{0,1}(i,j)$, en $P_{1,1}(i,j)$

Usually we do not work directly with these histograms but with a small number of parameters that are computed from these histograms

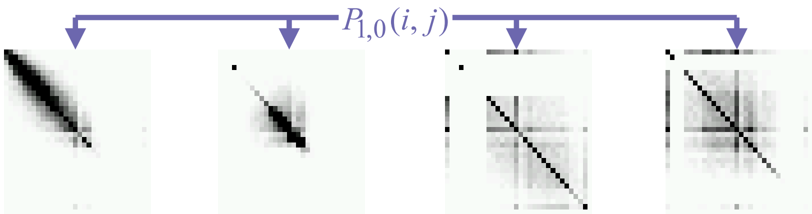
With these parameters we define homogeneity criteria (for segmentation) or edge strength measures (for edge detection)

Examples



The two dimensional histograms ("co-occurrence matrices) strongly differ depending on the spatial properies of the texture

... Segmentation based on 2D histograms



rocks textile 1 textile 2 leaves

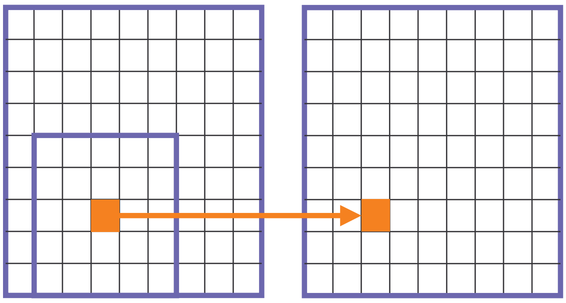
Possible parameters:

entropy: $-\sum_{i,j} P_{s,t}(i,j)\log_2 P_{s,t}(i,j)$ How random the textuur is?

inertia: $-\sum_{i,j} (i-j)^2 P_{s,t}(i,j)$ Amount of high-frequencies?

(how strongly differ neighboring pixels)

Segmentation by texture classification



For each pixel we calculate the parameters of the local 2D histogram

We compare these parameters with those of example textures
The pixel is assigned to the class with best matching example texture

The classification can be done based on other parameters
- (mean) RGB-values in color images
- Markov-parameters, ...

Bibliography

Edge detection
- Canny, J., *A Computational Approach To Edge Detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

Segmentation
- J. Shi en J. Malik. Normalized Cuts and Image Segmentation. IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, augustus 2000, pp. 888-905.