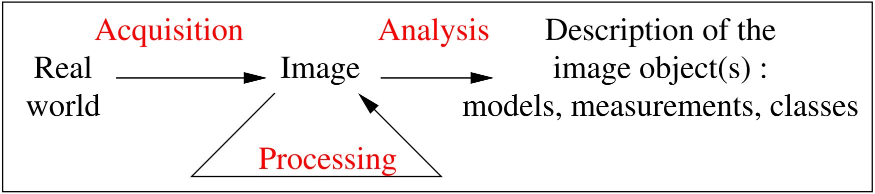
Part 1 : Image Segmentation

Anne Vialard

LaBRI, Université de Bordeaux

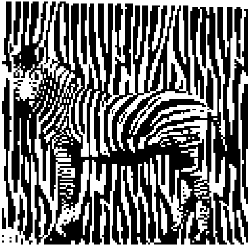
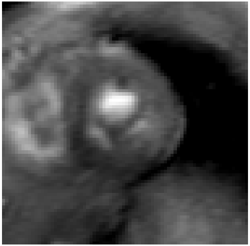
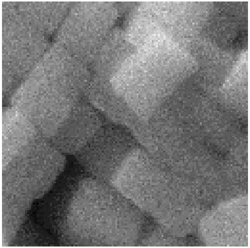
Contents

Issues



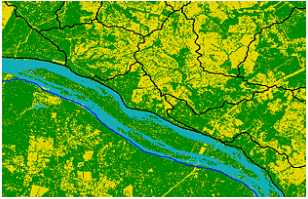
- imitate the human vision system
- a priori knowledge is important for high-level processes
- no generic solution but a set of solutions for specific problems

Difficult cases...

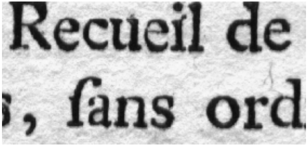
  

Examples

- Satellite imagery



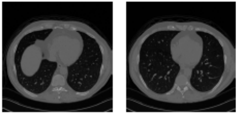
- Document image analysis

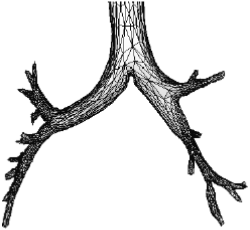


Recueil de
s, fans ord

- Content based image retrieval

- Aid in medical diagnosis





- Nondestructive test
  Control of a trajectory

Image analysis subfields I

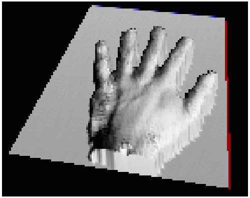
- Segmentation / Reconstruction
- Pattern recognition :
  Assign a category to (a part of) an image

$\boxed{\text{preprocessing}} \rightarrow \boxed{\text{features extraction}} \rightarrow \boxed{\text{classifying}}$

- 3D Vision

  Shape from shading : Retrieve a 3D shape from a 2D image (depth from color variations)

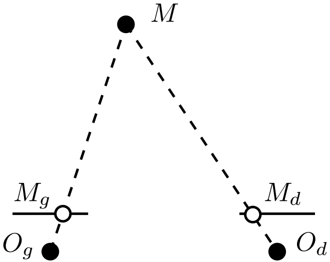
  

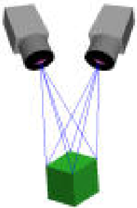
  

  Stereovision (Shape from stereo) : Retrieve the points depth from two projections of the scene (different points of view)

Image analysis subfields II

- image matching
- reconstruction





- Motion analysis

Course outline I

1. Image segmentation
    - region based approach : split, merge, split and merge
    - contour based approach
    - other approaches : watersheds, Mumford Shah, deformable models, level sets
2. Digital geometry applied to image analysis
    - curves and surfaces / regions
    - algorithms for tracing region boundaries
    - representing a 2D/3D partition
    - digital lines and planes, recognition algorithms
    - digital distances, distance transform, skeletonization

Course outline II

3. Geometric and topologic characterization
   - region features : geometric moments, convexity, topologic features
   - geometric features of a curve/surface : length/area, normal and curvature at a point, salient points

Bibliography I

- Digital Image Processing - RC Gonzalez, RE Woods - Prentice Hall
- Computer Vision : A Modern Approach - DA Forsyth, J Ponce - Pearson
- Digital Geometry : Geometric Methods for Digital Image Analysis - R. Klette, A. Rosenfeld - Morgan Kaufmann
- Image Processing, Analysis and Machine Vision - M. Sonka, V. Hlavac, R. Boyle - Cengage Learning

Contents

Image segmentation methods I

- segmentation : cut an image into meaningful regions, the image "objects" $\Longrightarrow$ pixels/voxels labeling.

  easy for humans : prior knowledge, global point of view on an image, inference (e.g. hidden boundaries)
- Methods :
  - region-based : gather similar pixels/voxels $\Rightarrow$homogeneous regions
  - edge-based : look for dissimilar pixels/voxels $\Rightarrow$interface between different zones

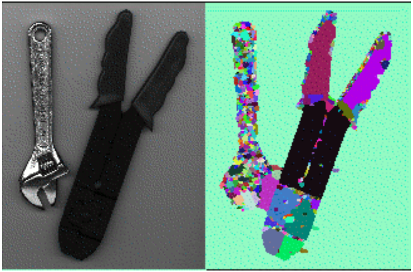


Image segmentation : example 1

Labeling into 4 components
(Region based approach : Markov field)

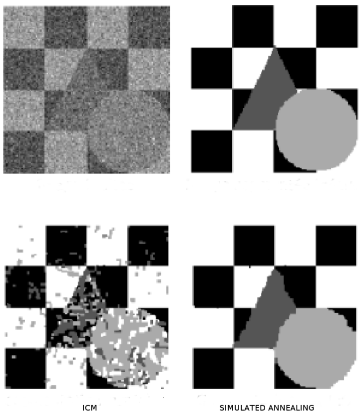
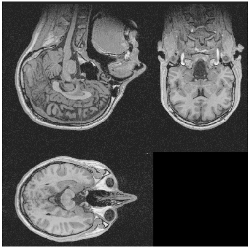
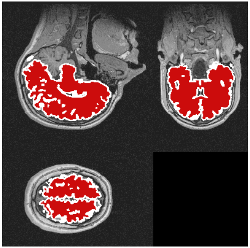


Image segmentation : example 2

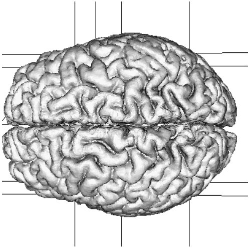
3D reconstruction of the cerebral cortex
(edge based approach : deformable model)



3D Images (3 slices)



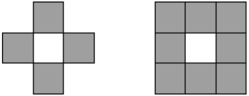
Segmentation



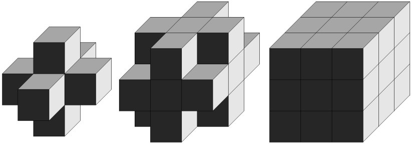
3D Reconstruction

Definitions - Notations I

- pixel/voxel : belongs to $X \subset \mathbb{Z}^2$ / $\mathbb{Z}^3$
- 2D connectivities : 4-connectivity / 8-connectivity



- 3D connectivities : 6-connectivity / 18-connectivity / 26-connectivity



- region R : connected subset of X
- size $|R|$ of R = number of pixels/voxels in R

Definitions - Notations II

- border of a region R : δR = interpixel/intervoxel boundary of R (set of pixel edges / voxel faces between R and its complement) $|\delta R|$ = size of the border (length / area)
- image : function I from X to a set $E =$
  1. $\{0, 1\}$: binary image
  2. $\{0, \ldots, 255\}$: graylevel image (8 bits)
  3. $\{0, \ldots, 2^{16} - 1\}$: graylevel image (16 bits)
  4. $\{0, \ldots, 255\}^3$: color image (RGB color space)
  5. $\ldots$
- histogram h : function from E to $\mathbb{Z}^+$ of the occurences of each value of I
- mean value of a region R : $\mu_R = \frac{1}{|R|} \sum_{p \in R} I(p)$.
- variance of a region R : $\sigma_R^2 = \frac{1}{|R|} \sum_{p \in R} (I(p) - \mu_R)^2$.

Formalising the segmentation problem

Definition *[Horowitz75]*

X : domain of the image I

P : predicate defined on the set of X subsets, depends on I

segmentation of X : $(S_i)_{i=1..n}$, subsets of X such that

1. $X = \cup_{i=1}^{n} S_i$ and $\forall i, j \in 1..n, i \neq j, S_i \cap S_j = \emptyset$
2. $\forall i \in 1..n$, S_i is connected et $P(S_i) = \mathit{true}$
3. $\forall i, j \in 1..n$, S_i adjacent to S_j and $i \neq j \Rightarrow P(S_i \cup S_j) = \mathit{false}$

Examples of homogeneity predicate :

- $P(R) = \text{true} \Leftrightarrow \sigma_R < 5$
- $P(R) = \text{true} \Leftrightarrow \forall p \in R, |I(p) - \mu_R| < 10$

Exercise

$$\text{Image } I = \begin{array}{cc} 100 & 45 \\ 55 & 0 \end{array}$$

Segmentation of I ?
$P(R) = \text{true} \Leftrightarrow \forall p \in R, |I(p) - \mu_R| < 30$

Difficulties

Unicity ? Stability ? Calculability ?

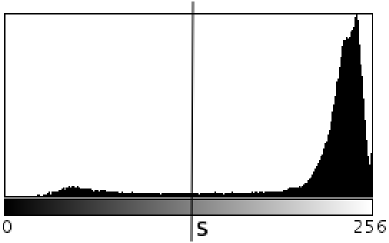
Contents

3 - Segmentation : region based approaches

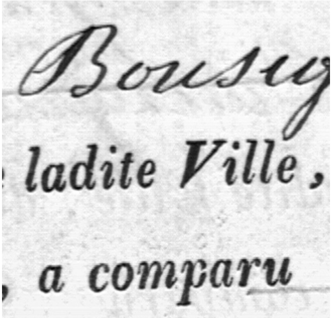
Thresholding / Classification
Split and merge methods
Grouping pixels

Thresholding

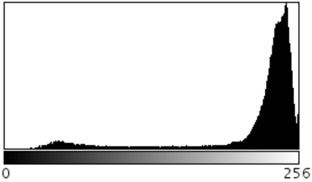
- Goal : assign a class to each pixel of a gray-level image.
  class $=$ gray-level range
- Idea :
  - compute thresholds from the histogram (image/region)
  - a pixel p is classified by comparing $I(p)$ to the thresholds



Thresholding and histogram I



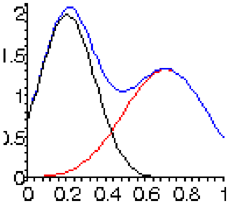
- histogram of I in $R \subset X \approx$ distribution of the values in R.
- Example : uniform object $\Rightarrow$ histogram $\approx$ gaussian curve of low variance
- bimodal histogram : two uniform objects with different means



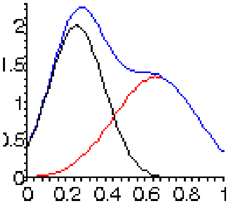
- Thresholding : find the threshold(s) which best separate the two (or more) objects.

Thresholding and histogram II

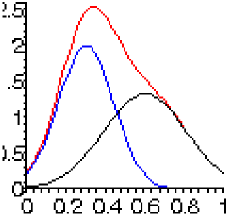
difficult when the means get closer.



$(\mu_1 = 0.2, \sigma_1 = 0.2)$
$(\mu_2 = 0.7, \sigma_2 = 0.3)$



$(\mu_1 = 0.25, \sigma_1 = 0.2)$
$(\mu_2 = 0.65, \sigma_2 = 0.3)$



$(\mu_1 = 0.3, \sigma_1 = 0.2)$
$(\mu_2 = 0.6, \sigma_2 = 0.3)$

Segmentation by thresholding

pixel : (x, y),
gray level : $I(x, y)$
local property : $P(x, y)$
threshold used to classify pixel (x, y) : $T(x, y)$

3 types of thresholding methods :

- *global* thresholding : $T(x, y) \stackrel{\text{def}}{=} T(I(x, y))$
- *local* thresholding : $T(x, y) \stackrel{\text{def}}{=} T(I(x, y), P(x, y))$
- *dynamic* thresholding : $T(x, y) \stackrel{\text{def}}{=} T(I(x, y), P(x, y), x, y)$

If 2 classes, binarization.

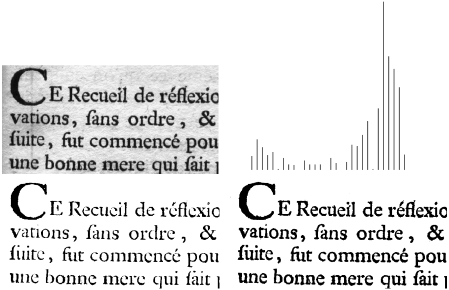
Example of global thresholding 1/3
Binarization *[Otsu79]*

- the histogram h is split so as to minimize the partition error
- Idea : minimize the variance in each class (C_1 and C_2).
  $p(n)$ probability of gray level n, t threshold
  $p(C_1) = \sum_0^t p(n)$, $\mu_{C_1} = \frac{\sum_0^t np(n)}{p(C_1)}$, $\sigma^2_{C_1} = \frac{\sum_0^t (n-\mu_{C_1})^2 p(n)}{p(C_1)}$
  $p(C_2) = \sum_{t+1}^N p(n)$, $\mu_{C_2} = \frac{\sum_{t+1}^N np(n)}{p(C_2)}$, $\sigma^2_{C_2} = \frac{\sum_{t+1}^N (n-\mu_{C_2})^2 p(n)}{p(C_2)}$

$$\begin{aligned}\sigma^2_{intra} &= p(C_1)\sigma^2_{C_1} + p(C_2)\sigma^2_{C_2} \\ \sigma^2_{inter} &= p(C_1)p(C_2)(\mu_{C_1} - \mu_{C_2})^2\end{aligned}$$

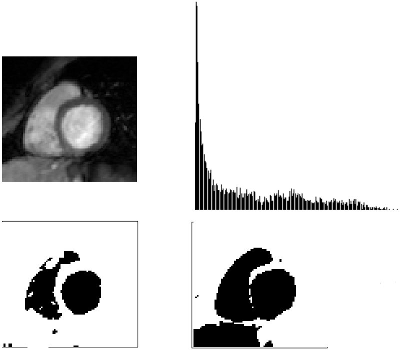
To minimize σ^2_{intra} is equivalent to maximize σ^2_{inter}

Example of global thresholding 2/3



T= 50 T= 128

Example of global thresholding 3/3



T = 130 T= 82

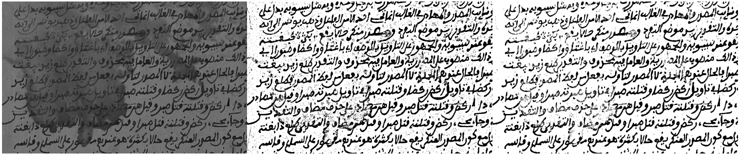
Example of local thresholding 1/2

Binarization *[Sauvola00]*

Idea : the threshold is adapted to the local contrast.

$T(x, y) = \mu(x, y)(1 + k(\frac{\sigma(x,y)}{R} - 1)$

- $\mu(x, y)$ mean of the gray levels in the neighborhood of (x, y)
- $\sigma(x, y)$ standard deviation of the gray levels in the neighborhood of (x, y)
- standard values of the parameters $k = 0.5$, $R = 128$



k=0.1, k=0.2, radius of the neighborhood (square) = 15

Example of local thresholding 1/2

Binarization *[Bhanu01]*

Idea : if a point belongs to C_i, most of its neighbors (8-connected neighborhood) also belong to C_i.

- $P(p) = (P_{C_1}(p), P_{C_2}(p))$ probability vector associated to pixel p
  $P_{C_i}(p)$: probability that pixel p belongs to C_i
  $P_{C_1}(p) + P_{C_2}(p) = 1$
- $Q(p) = (Q_{C_1}(p), Q_{C_2}(p))$ compatibility vector associated to pixel p
  $Q_{C_i}(p) = \frac{1}{8} \sum_{q \in N_8(p)} P_{C_i}(q)$
- To maximize : $\sum_{image}(P_{C_1}(p)Q_{C_1}(p) + P_{C_2}(p)Q_{C_2}(p))$

Example of local thresholding 2/2

Binarization *[Bhanu01]*

Iterative algorithm : μ initial threshold.

$$P^0_{C_1}(p) = \begin{cases} \frac{I(p)-\mu}{MaxGl} + 0.5 & \text{if } I(p) > \mu \\ \eta\frac{I(p)-\mu}{MaxGl} + 0.5 & \text{else } (\eta \text{ coef between 0.5 and 1}) \end{cases}$$

$$P^{n+1}_{C_1}(p) = \begin{cases} (1-\alpha_1)P^n_{C_1}(p) + \alpha_1 & \text{if } Q^n_{C_1}(p) > 0.5 \\ (1-\alpha_2)P^n_{C_1}(p) & \text{else} \end{cases}$$

- α_1, α_2 coefs between 0 and 1.

$\Rightarrow$ increases the probability that p belongs to C_1 if the probability that its neighborhood belongs to C_1 is high.

- Iterations until 90% of the pixels are well labeled ($P_{C_1}(p) > 0.9$ or $P_{C_2}(p) > 0.9$)

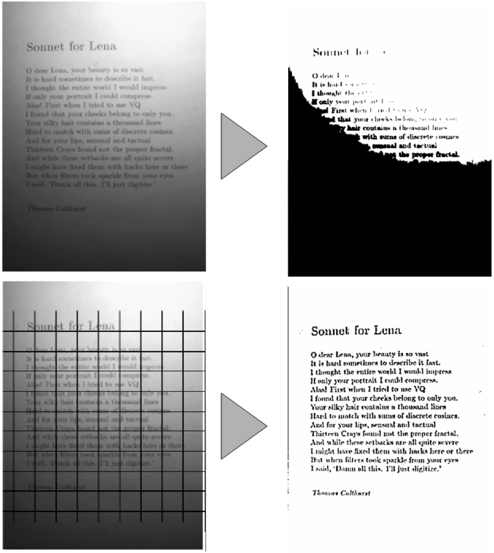
Example of dynamic thresholding I

- The image is split into regular blocks
- A threshold is computed for each block and assigned to the block center :
  Is the block histogram bimodal ?
  - If it is, the threshold is computer from the histogram
  - If not, the threshold is defined as the mean of the thresholds of the neighboring blocks.
- The threshold of a pixel is computed by linear interpolation from the thresholds of the neighboring blocks.

Potential problem : truncated regions

Example of dynamic thresholding II

Binarization *[Chow,Kaneko72]*



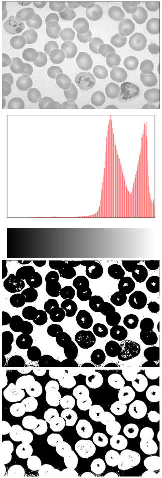
Classification

Idea :

- histogram mode $\simeq$ image component.
- split the histogram into k classes.
- each pixel is labeled with the number of its class

How to find k classes from the histogram :

- User
- Finding "valleys" of the histogram
- Gaussian mixture
- "k-means" algorithm ...



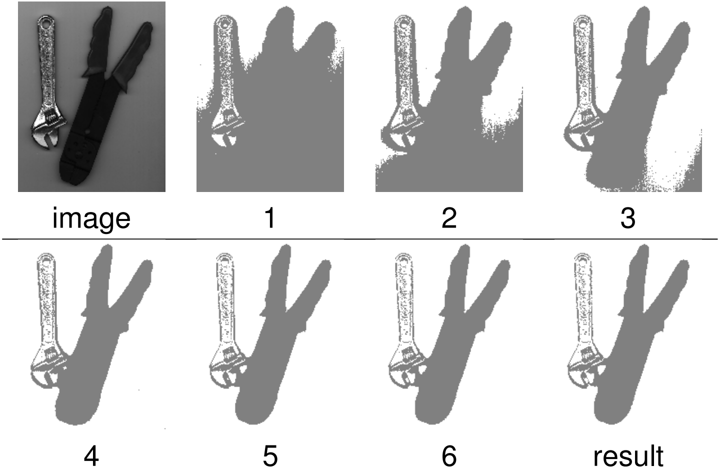
Classification example : k-means algorithm

Iterative method to split the histogram into k classes (k is fixed).
Algorithm :

- Arbitrary choose k values $\{c_1, \ldots, c_k\}$ in the histogram
- Until c_i are modified in the loop, do
    - For each histogram value find the nearest c_i value
    - The class C_i is the set of values that are closer to c_i than to any other c_j.
    - Replace c_i by the mean of its class C_i
- resulting classes = C_i.

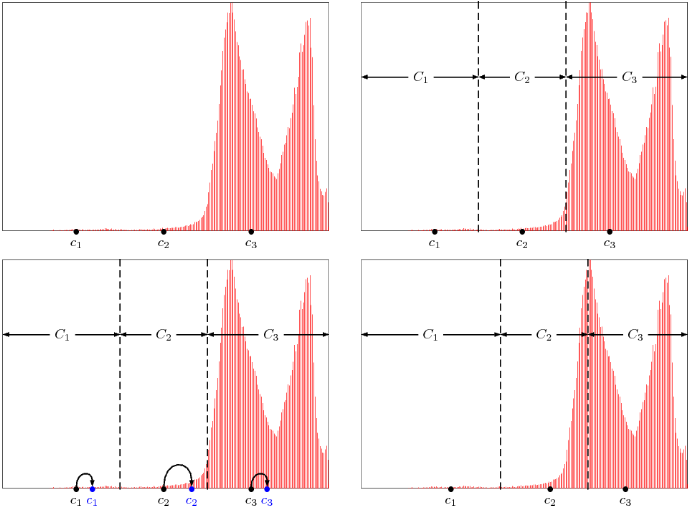
k-means algorithm : example

2 classes



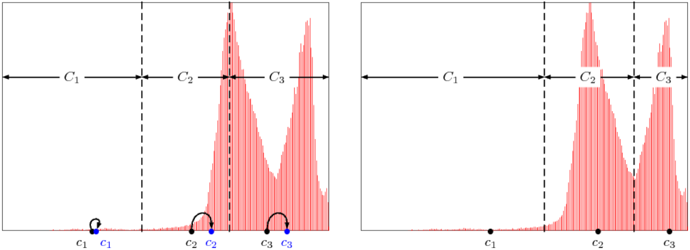
k-means algorithm : example I

3 classes



k-means algorithm : example II

3 classes



Thresholding / classification (conclusion)

- fast labeling based on the image histogram
- often used as the initialization of a higher level segmentation algorithm with take into account the *location* of the pixels values
- relaxation filtering of the labels : The new label of a pixel p is the most frequent label in a neighborhood of p.
- fuzzy classification : each value is given a probability to belong to a class

3 - Segmentation : region based approaches

Thresholding / Classification
Split and merge methods
Grouping pixels

Splitting

Top-down approach

Focusing attention : from a coarse scale to small details.

Idea

Initialization = under-segmentation (all the image, a region)
Split non homogeneous regions

Recursive splitting

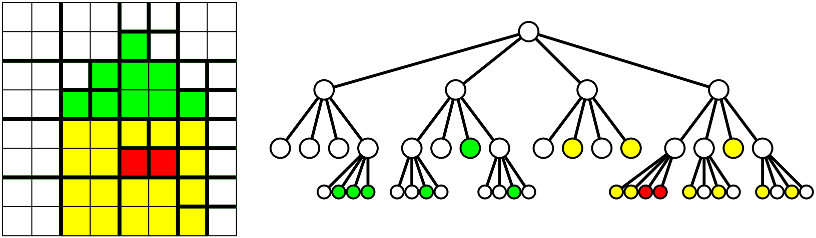
1. Partition the region, for example from its histogram.
2. For each resulting region, if possible (and necessary) go back to 1.

Problem : no undo

Quadtree 1/2

The image is coded as a tree - recursive definition :

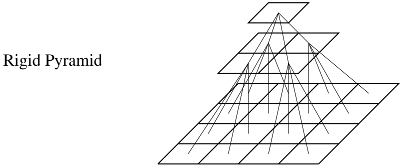
- root : whole image
- if not homogeneous, the part of the image corresponding to a node is split into 4 parts ⇒ each node has 4 children.



Splitting segmentation method based on a uniformity criterion (and not on the histogram).

Quadtree 2/2

- a quadtree is a particular case of pyramid (sequence of graphs representing an image at different resolution levels)



- drawback : regular decomposition
- can be used as an initial partition for a merging method

Idea

Initialization = over segmentation into uniform regions
Merge each pair of adjacent regions that verify a homogeneity criterion.

$\Rightarrow$ to define a predicate $Merge(R_i, R_j)$ where R_i and R_j are adjacent regions.

features of a region R_i :

- μ_i : mean of the gray values
- σ_i : standard deviation of the gray values
- $|R_i|$: number of pixels/voxels
- δR_i contour and its size $|\delta R_i|$
- $|\delta R_i \cap \delta R_j|$ size of the shared boundary between R_i and R_j

Homogeneity criteria

Simple evaluation of the homogeneity of $R = R_i \cup R_j$:

- σ of R is inferior to a threshold
- the amount of pixels of R which gray-level is outside $[\mu_R - \sigma_R, \mu_R + \sigma_R]$ is inferior to a threshold
- ?

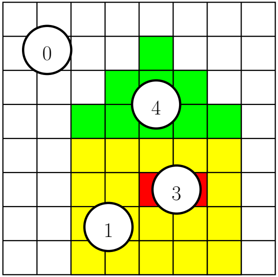
Beveridge criterion

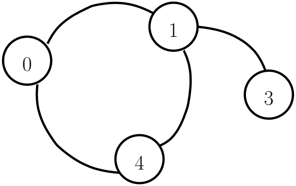
$$f(R_i, R_j) = f_{sim}(R_i, R_j)\sqrt{f_{size}(R_i, R_j)}f_{cont}(R_i, R_j)$$

- Similarity criterion :
  $f_{sim}(R_i, R_j) = \frac{|\mu_i - \mu_j|}{max(1, \sigma_i + \sigma_j)}$
- Size criterion :
  $f_{size}(R_i, R_j) = min(2, \frac{min(|R_i|, |R_j|)}{T_{opt}})$, T_{opt} : fixed according to the image size
- Shared boundary criterion :
  $$f_{cont}(R_i, R_j) = \begin{cases} C(R_i, R_j) & \text{if } \frac{1}{2} \leq C(R_i, R_j) \leq 2 \\ \frac{1}{2} & \text{if } C(R_i, R_j) < \frac{1}{2} \\ 2 & \text{else} \end{cases}$$
  $C(R_i, R_j) = \frac{min(|\delta R_i|, |\delta R_j|)}{4|\delta R_i \cap \delta R_j|)}$

Data structure for merging : RAG

Region Adjacency Graph : non oriented graph where the nodes correspond to the image regions
∃ an edge between 2 nodes if and only if the 2 corresponding regions are adjacent.
Additional need : geometric description of the regions





Merging adjacent regions = shrinking an edge + (deleting multiple edges)

Merging with adaptive pyramids 1/3

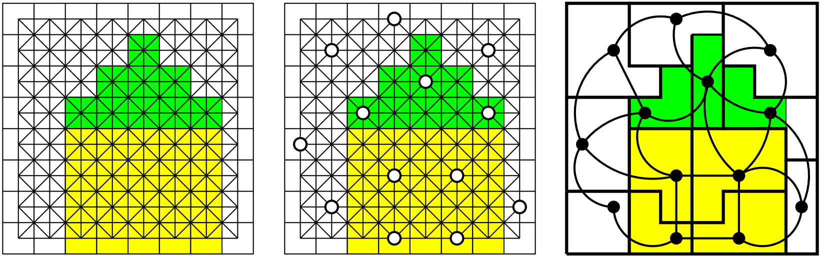
- hierarchical data structure for a specific merging algorithm : merging groups of regions instead of regions pairs
- pyramid = linked graphs , each graph represents a partition of the image (RAG).

Algorithm :

- Base of the pyramid : initial image, 8-connectivity.
- How to compute the next level :
  1. Remaining nodes computation
  2. Merging

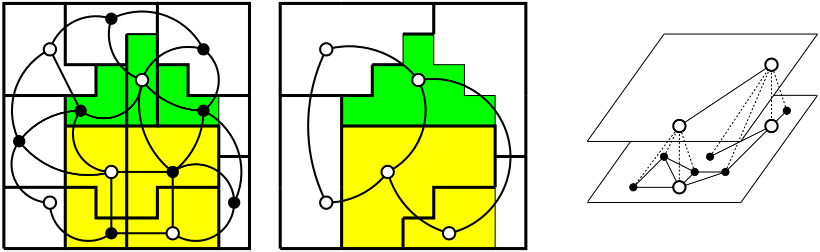
Merging with adaptive pyramids 2/3

1. **Remaining nodes computation.** Two remaining nodes can't be adjacent, a non remaining node is adjacent to at least one remaining node
   Selection criterion (example) : local minimum of variance + testing the rules
2. **Merging.** Each non remaining node is merged with a remaining node (the most similar one)



Merging with adaptive pyramids 3/3

- iterations...



+ : fast reduction of the graph, the result is independent from the way of traversing the image

Splitting and merging

- Any initial segmentation (for example fixed size blocks)
- Subdivide non uniform regions
  $\forall i \in 1..n,\ P(R_i)$ false $\Rightarrow$ split R_i
- Merge non maximal regions
  R_i *adj* R_j and $Merge(R_i, R_j)$ true $\Rightarrow$ merge R_i and R_j

Better results are obtained by alternating splitting and merging

Splitting and merging : example 1

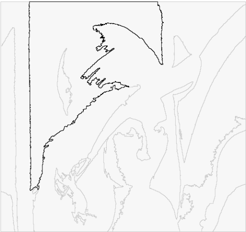
Lenna

After splitting

After merging

Splitting and merging : example 1

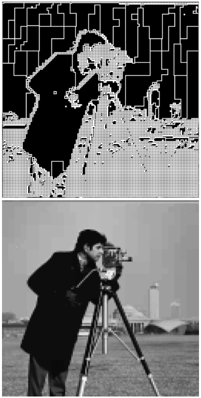
Lenna



Region selection

Splitting

Splitting and merging : example 2



after splitting

after merging

3 - Segmentation : region based approaches

Region growing

- Seeds selection (seed = set of connected pixels in a homogeneous part of the image)
- Seeds growing by adding similar connected pixels

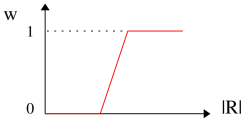

Region growing

```
growSeed(seed)
  R.init()
  R.add(seed)
  while R.hasNeighbor()
        p = R.getNextNeighbor()
        if pred(p, R)
           R.add(p)
```

Example of $pred(p, R)$: $\frac{|I(p)-\mu_R|}{\sigma_R} <= T$

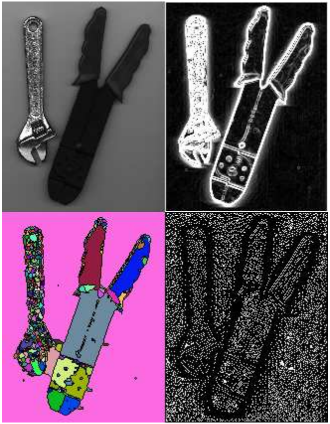
Possible to adapt the criterion to the region size :

$$\frac{w(|R|)}{T_1}\frac{|I(p)-\mu_R|}{\sigma_R} + \frac{(1-w(|R|)}{T_2}\sigma_{R\cup p} <= 1$$



Others : geometric criterion, simultaneous region growing

Detection of the "catchment areas" on the norm of the gradient

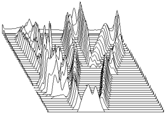


Watershed

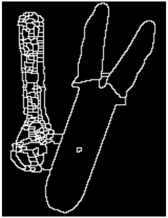
Example

image

norm of the gradient



elevation map

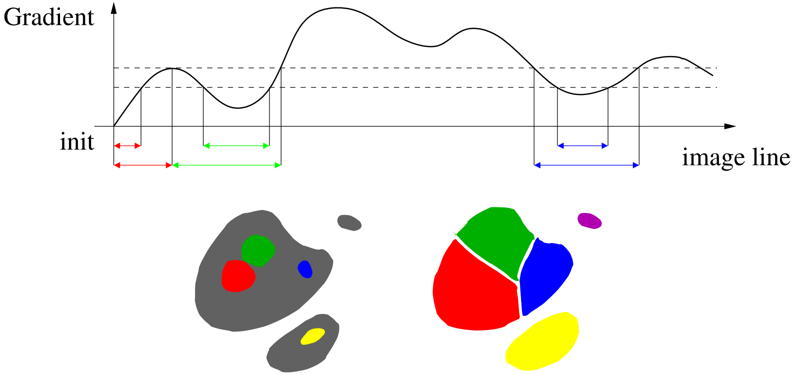


basins

Watershed

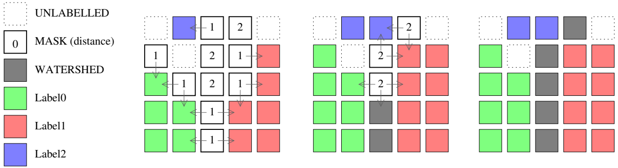
Computation by immersion

- seeds : pixels with a low gradient value
- ith step : level $\nearrow \Rightarrow$ new pixels (p_i) of higher gradient
  If p_i is adjacent to an existing basin, add it to the basin
  Else p_i is a new seed (basin)

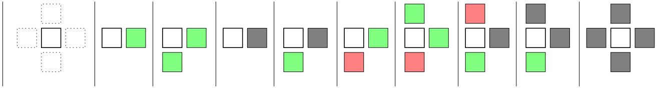


Watershed

Algorithm *[Vincent,Soille91]*



- Sort the pixels by increasing altitude
- Group pixels (p_i) of same altitude
  - processing order according to the distance to an existing basin
  - the label of each p_i depends on its neighborhood



  - If p_i has no label (new basin), it takes a new label which is spread to its non labeled neighbors many minima $\Rightarrow$many small regions (smooth the norm of the gradient value to reduce this problem).

Wateshed

Markers

Avoiding over-segmentation with markers (user interaction)

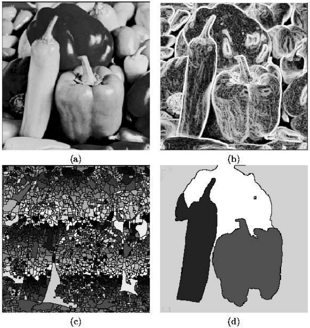


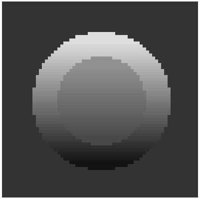
Figure 5.51: *Watershed segmentation: (a) original; (b) gradient image, 3×3 Sobel edge detection, histogram equalized; (c) raw watershed segmentation; (d) watershed segmentation using region markers to control oversegmentation. Courtesy W. Higgins, Penn State University.*

Watershed computed from the Laplacian value

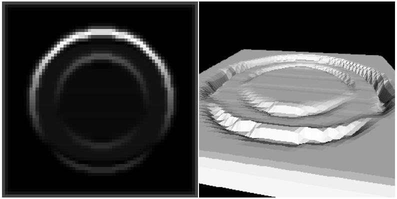
Contents

Edge-based segmentation

Edge detection from the gradient value

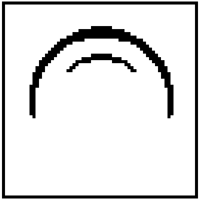


Image



Gradient norm

Thresholded gradient norm



Gradient (2D image)

Gray-level at pixel (x, y) : $I(x, y)$

Gradient : $\nabla I(x, y) = (G_x, G_y) = (\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y})$

Gradient Norm : $G = \sqrt{G_x^2 + G_y^2}$

Gradient Orientation : $\theta = arctan(\frac{G_y}{G_x})$

Simple computation : Sobel filter

$y \downarrow \rightarrow x$

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

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

Gradient (3D image)

Gray-level at pixel (x, y, z) : $I(x, y, z)$

Gradient :

$\nabla I(x, y, z) = (G_x, G_y, G_z) = (\frac{\partial I(x,y,z)}{\partial x}, \frac{\partial I(x,y,z)}{\partial y}, \frac{\partial I(x,y,z)}{\partial z})$

Gradient Norm : $G = \sqrt{G_x^2 + G_y^2 + G_z^2}$

Gradient Orientation : $\theta = arctan(\frac{G_y}{G_x}) \phi = arctan(\frac{G_x}{G_z})$

Sobel mask for computing G_y (slices orthogonal to z)

$y \downarrow$

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

-2	-4	-2
0	0	0
2	4	2

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

- Simple filters : Prewitt, Sobel
- Canny filter : $\frac{\partial I}{\partial x}$ computed by convolution with $x \mapsto Axe^{-\frac{x^2}{2\sigma^2}}$ (Gaussian derivative). Similar for y.
- Deriche filter : $\frac{\partial I}{\partial x}$ computed by convolution with $x \mapsto Axe^{-\alpha|x|}$. Similar for y.
- Others : Shen-Castan filter...

NB : very similar in practice, direct extension to 3D

Edge image

Idea

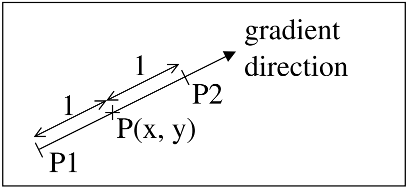
local max of the gradient norm $\Rightarrow$ edge point

How to compute an edge image :

1. gradient estimation at each image point
2. extraction of the local maxima of the gradient norm in the gradient direction
3. selection of the significant local maxima
4. edge closing by following paths on a ridge line in the gradient norm image

Edge image

Extraction of the local maxima of the gradient norm



Gr : gradient norm at P
Gr_1 : gradient norm at P_1
Gr_2 : gradient norm at P_2

local maximum : $Gr > Gr_1$ and $Gr > Gr_2$

Gr_1 and Gr_2 are computed by linear interpolation

Edge image

Hysteresis thresholding of local maxima

Goal : limit the contours fragmentation

2 thresholds : $T_h > T_l$

Are kept :

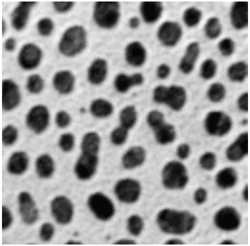
- local maxima with a value greater than T_h
- local maxima with a value greater than T_l belonging to a connected component of local maxima ($\geq T_l$) containing at least one value $\geq T_h$

Result : a binary image (edge image)

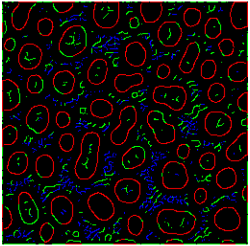
Edge image

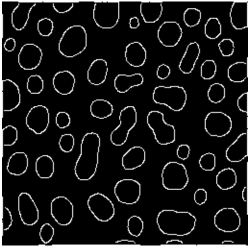
Hysteresis thresholding : example 1

initial image



local maxima of the gradient

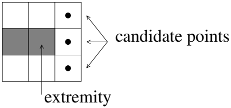




Edge image
Edge closing (2D)

Idea : follow a ridge line in the image of the gradient norm from each edge extremity

1. Find the extremity points



2. Choice between candidate points to extend the edge : weighted paths exploration

Contents

5 - Other approaches for segmenting an image

Deformables models

Level-set method

Segmentation as energy minimization

Deformable models

Idea : variational approach

- set of possible shapes
- an energy (real number) is associated to each shape
- minimum search

$$\boxed{E(C) = E_{internal}(C) + E_{external}(C)}$$

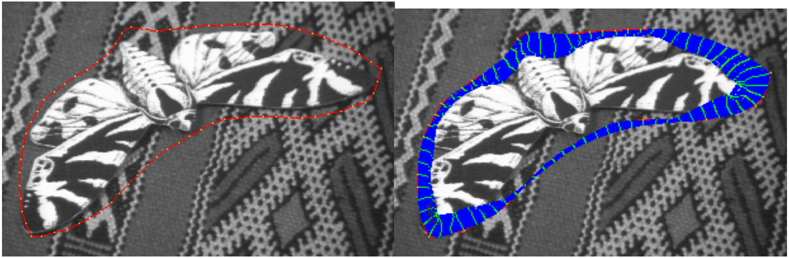
- External energies : correspond to edges in the image, user interaction, a priori knowledge about the shape, ...
- Internal energies : stretching and bending, keep the model smooth during deformation, ease the extraction of a shape with fuzzy or fragmented contours by filling the missing information

very generic framework : segmentation, stereo-vision, object tracking in videos, ...

Active contours : 2D example

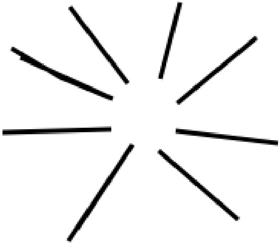
Active contour or Snake : iterative optimization

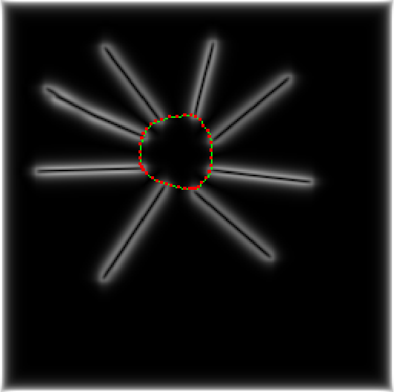
- initialization : a curve near the contour to extract
- iterations : deformations of the active contour until it reaches a location with minimum energy.



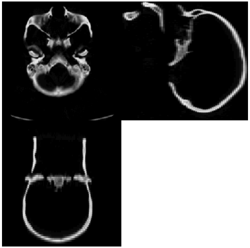
Active contours : 2D example

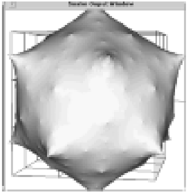
"conceptual" shape extraction

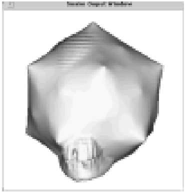


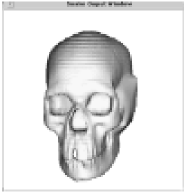


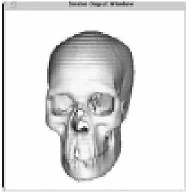
Active contours : 3D example











(a) (b) (c) (d)

Active contours : 2D formulation [Kass et al. 87]

Parametric representation of the active contour :
$C = \{v(s) = (x(s), y(s)); s \in [0, 1]\}$

$$\boxed{E_{internal}(C) = \int_0^1 \alpha \left|\frac{\partial v(s)}{\partial s}\right|^2 + \beta \left|\frac{\partial^2 v(s)}{\partial s^2}\right|^2 ds}$$

$\Rightarrow$ the active contour has a low internal energy when it is not "too" stretched and not "too" bent.

$$\boxed{E_{image}(C) = \lambda \int_0^1 -|\nabla I(v(s))|^2 ds}$$

$\Rightarrow$ the active contour has a low image energy when it is located on a contour of the image.
$\Longrightarrow$ find the contour that minimizes the sum.

Active surfaces : 3D formulation [Terzopoulos et al. 91]

Parametric representation of the active surface :
$S = \{v(r,s) = (x(r,s), y(r,s), z(r,s)); (r,s) \in [0,1]^2\}$

$$E_{internal}(S) = \int_0^1 \int_0^1 \alpha_r \left|\frac{\partial v(r,s)}{\partial r}\right|^2 + \alpha_s \left|\frac{\partial v(r,s)}{\partial s}\right|^2 + \beta_{rs}\left|\frac{\partial^2 v(r,s)}{\partial s \partial r}\right|^2 + \beta_{rr}\left|\frac{\partial^2 v(r,s)}{\partial r^2}\right|^2 + \beta_{ss}\left|\frac{\partial^2 v(r,s)}{\partial s^2}\right|^2 drds$$

$$E_{image}(S) = \lambda \int_0^1 \int_0^1 -|\nabla I(v(r,s))|^2 drds$$

Active contours : how to compute the 2D evolution

1- The contour is digitized in N points :
$i = 0..N-1, X[i] = x(\frac{i}{N})$ or $X(ih)$ with $h = \frac{1}{N}$.

$$X = \begin{bmatrix} \dots \\ x_i \\ \dots \end{bmatrix} \qquad Y = \begin{bmatrix} \dots \\ y_i \\ \dots \end{bmatrix}$$

External energy : forces deriving from the energies

$$f_x(X,Y) = \begin{bmatrix} \dots \\ \frac{\partial |\nabla I|^2}{\partial x}(x_i, y_i) \\ \dots \end{bmatrix} \quad f_y(X,Y) = \begin{bmatrix} \dots \\ \frac{\partial |\nabla I|^2}{\partial y}(x_i, y_i) \\ \dots \end{bmatrix}$$

Active contours : how to compute the 2D evolution

Minimizing the energy is equivalent to solve (Euler-Lagrange) :

$$-\alpha x''(s) + \beta x^{(4)}(s) = \frac{\partial |\nabla I(v)|^2}{\partial x}$$
$$-\alpha y''(s) + \beta y^{(4)}(s) = \frac{\partial |\nabla I(v)|^2}{\partial y}$$

2- Approximation with finite differences :

$$\boxed{\begin{array}{l} AX = f_x(X,Y) \\ AY = f_y(X,Y) \end{array}}$$

$$A = \frac{1}{h^2}\begin{bmatrix} 6\frac{\beta}{h^2}+2\alpha & -4\frac{\beta}{h^2}-\alpha & \frac{\beta}{h^2} & 0 & \cdots \\ -4\frac{\beta}{h^2}-\alpha & 6\frac{\beta}{h^2}+2\alpha & -4\frac{\beta}{h^2}-\alpha & \ddots & \ddots \\ \frac{\beta}{h^2} & -4\frac{\beta}{h^2}-\alpha & 6\frac{\beta}{h^2}+2\alpha & \ddots & \ddots \\ 0 & \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}.$$

Active contours : how to compute the 2D evolution

3- Iterative resolution : successive locations of the contour

the parameter γ is the inverse of the time step but can be interpreted as a friction coefficient.

From the initial location (X_0, Y_0). Iterations :

$$\boxed{\begin{array}{l} X_t = (A+\gamma I)^{-1}(\gamma X_{t-1} + f_x(X_{t-1}, Y_{t-1})) \\ Y_t = (A+\gamma I)^{-1}(\gamma Y_{t-1} + f_y(X_{t-1}, Y_{t-1})) \end{array}}$$

The contour is deformed until it reaches an energy minimum.

5 - Other approaches for segmenting an image

Level-set method : tracking interfaces

- **Origin.** In physics, modeling of front propagation. Ex : grass fire
- **Model.** The evolution of the interface is deduced from the evolution of the entire environment [Osher and Sethian 88].

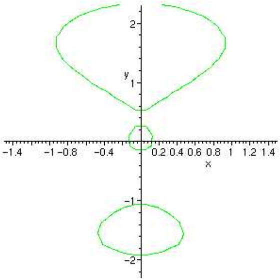
  $\Rightarrow$ interface = isopotential in a potential field.
- **Segmentation.** matching model/image [Malladi *et al.* 93], [Caselles *et al.* 93]
- **Aim.** avoiding topological problems, easy *nD* extension

Evolving environment : a moving hyper-surface (I)

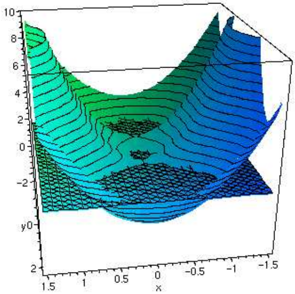
- modeling the evolution of a curve $C(t)$ in the plane
- let $f(t, \mathbf{x}) \;: [0, \infty[\times \mathbb{R}^2 \to \mathbb{R}$ be a scalar function in the plane, such that $f(t, \mathbf{x}) = \pm d$, with d distance from $\mathbf{x}$ to $C(t)$.
  - $\Rightarrow$ f is a sort of signed distance map to $C(t)$.
- $S(t) = \{(\mathbf{x}, f(t, \mathbf{x}))\}$ is an (hyper-)surface of $\mathbb{R}^3$
  $f(t, \cdot)$ is the elevation map at time t.
- $S(t)$ is cut by the plane $z = 0$ to obtain $C(t)$.

Evolving environment : a moving hyper-surface (II)

- $C(t) = 0$ level of the surface $S(t)$



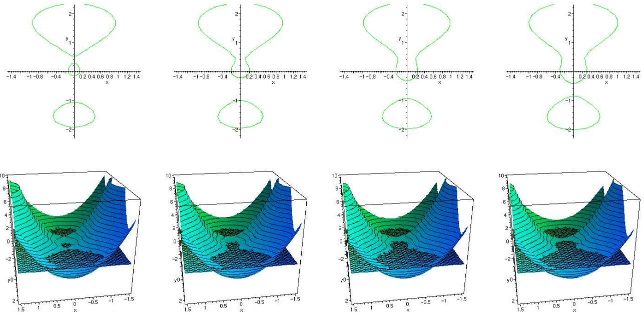
$C(t)$



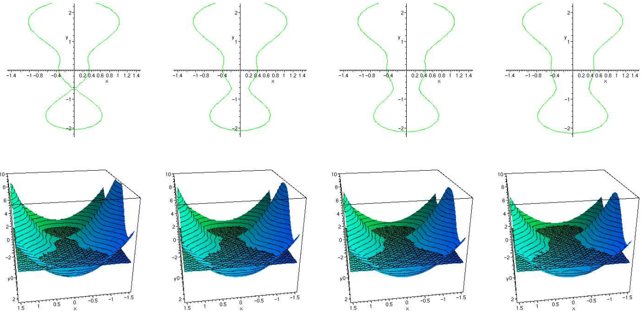
$S(t)$

Evolving environment : a moving hyper-surface (III)

- The hyper-surface $S(t)$ *never* changes topology, $C(t)$ can change topology.

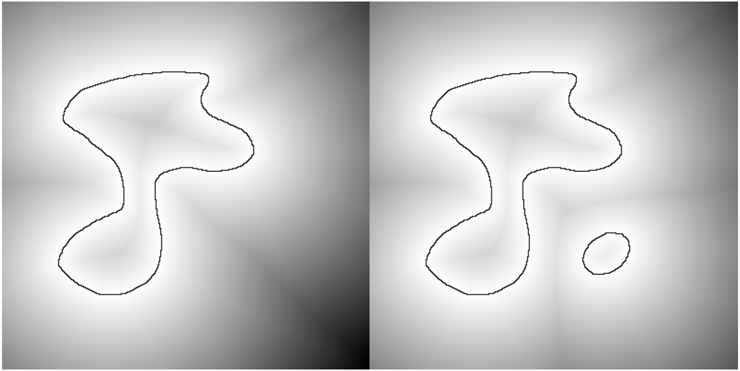


Evolving environment : a moving hyper-surface (IV)



$\Rightarrow$ Deform S (and thus f) instead of C.

hyper-surface = distance map to the contour



- $f(t, \mathbf{x})$: altitude at point $\mathbf{x}$ according to the time t

= signed distance to the contour $C(t)$

- process where : evolution of $C \Leftrightarrow$ evolution of f

Evolution : principles

- The front inflates or deflates in the direction of its normal vector.
- The hyper-surface $S(t)$ is deformed similarly to the front $C(t)$.
  - $\Rightarrow$ Each level of the function f moves similarly to the 0 level $(C(t))$.
- The front $C(t)$ tends to fill the holes
  - $\Rightarrow$ the front evolves faster when its curvature is very negative
  - $\Rightarrow$ the front is smoothed
- The front slows down when it reached strong edges in the image
  - $\Rightarrow$ speed of the front $=$ mix of local curvature and local image edges

Evolution : equation

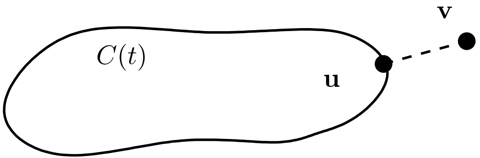
$$\boxed{\frac{\partial C}{\partial t} = A(t)\mathbf{n}(t) \Leftrightarrow \frac{\partial f}{\partial t}_{|\mathbf{x}(t)} = -\underbrace{A(t)}_{\hat{A}_{|\mathbf{x}(t)}} |\nabla f_{|\mathbf{x}(t)}|}$$

- Let $\mathbf{x}(t)$ be a front/contour point moving at speed $A(t)$.
- propagation in the direction the front normal vector $\mathbf{n}(t)$.
- the point remains on the front : $\forall t,\ f(t, \mathbf{x}(t)) = 0$
- By differentiating : $\frac{df}{dt} = \frac{\partial f}{\partial t} + \mathbf{x}' \cdot \frac{\partial f}{\partial \mathbf{x}}$
- We have $\mathbf{x}'(t) = A(t)\mathbf{n}(t)$ and ∇f aligned with $\mathbf{n}$ at point $\mathbf{x}(t)$.
- Thus $\frac{df}{dt} = 0 = \frac{\partial f}{\partial t} + A|\nabla f|$ at any point $\mathbf{x}(t)$.
- Valid equation on all the contour $C(t)$.

$$\frac{\partial f}{\partial t}_{|\mathbf{x}(t)} = -A(t)|\nabla f_{|\mathbf{x}(t)}|$$

Solving the evolution equation

- A is known along all the contour (see below).
- A is extended at any point of the plane : $\hat{A}$.
  - $\hat{A}(\mathbf{x}) = A(\mathbf{u})$ where $\mathbf{u}$ is the point of $C(t)$ which is closest to $\mathbf{x}$.

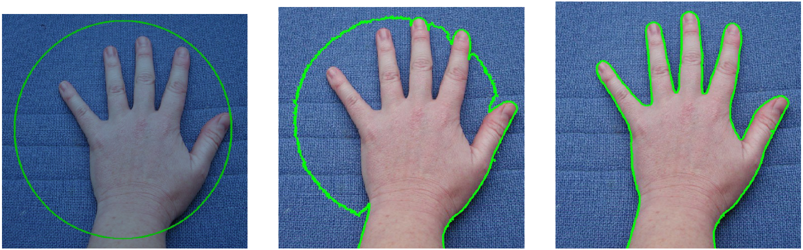


- Finite differences
  - regular grid of nodes ij separated by a distance h.
  - f_{ij}^n approaches the solution $f(n\Delta t, ih, jh)$, Δt time step.
  - We can write : $\frac{f_{ij}^{n+1} - f_{ij}^n}{\Delta t} = -\hat{A}_{ij} \left| \nabla_{ij} f_{ij}^n \right|$.
  - ∇_{ij} : gradient operator
- The distance function f has to be frequently reinitialized (the contour is extracted and the distances recomputed).

Front speed

- $A(\mathbf{x}) = -g(\mathbf{x})(A_0 + A_1\kappa(\mathbf{x}))$
  - A_0 : constant term (to propagate the front in the environment)
  - $A_1\kappa(\mathbf{x})$: removes the locations of high curvature. We have : $\kappa(\mathbf{x}) = \mathrm{div}\frac{\nabla f(\mathbf{x})}{|\nabla f(\mathbf{x})|} = \frac{f_{xx}f_y^2 - 2f_xf_yf_{xy} + f_{yy}fx^2}{(f_x^2+f_y^2)^{\frac{3}{2}}}$
  - $g(\mathbf{x})$: slows the front on the image contours. Examples : $g(\mathbf{x}) = \frac{1}{1+|\nabla I|^2}$, $g(\mathbf{x}) = \frac{1}{1+|\nabla G_\sigma \star I|^2}$

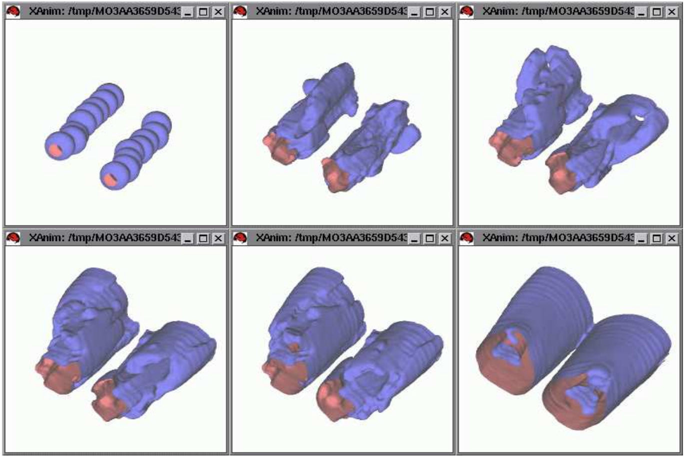
Example : 2D segmentation



images from D Lingrand (http ://www.polytech.unice.fr/ lingrand)

Example : 3D segmentation

- Thighs segmentation in a RMI (Malladi,Sethian)



Conclusion

- Implicit model
- Easy topology changes
- Link with classical deformable models (contour smoothing, use of the image edges)
- interesting for 3D images.
- many variants.

Limits

- Costly computation (distance from the whole space to the interface)
  - $\Rightarrow$ computation window (narrow band)
- Constant inflation or constant deflation
- End of the evolution not determined
- Not easy to add new constraints (user interaction)

5 - Other approaches for segmenting an image

Deformables models
Level-set method
Segmentation as energy minimization

Segmentation as energy minimization

Segmentation = minimizing the energy of a partition of the image

(R_i) set of regions, partition of X

$$E(\cup R_i) = \sum_i \underbrace{E_{\text{intra}}(R_i)}_{\searrow \text{ with homogeneity}} + \sum_{i,j/R_i \text{ adj. } R_j} \underbrace{E_{\text{inter}}(R_i, R_j)}_{\searrow \text{ with heterogeneity}} \tag{1}$$

- This problem (maximizing intra regions homogeneity and inter regions heterogeneity) is ill posed.

$\Rightarrow$ Problem regularization by adding constraints : length of the contours between regions, curvatures along contours,...

Mumford-Shah Model[89] (1/2)

- Idea : approximate the image I by a "smooth" function u.

$$E(u,\Gamma) = \underbrace{\mu^2 \iint_R (u-I)^2}_{\text{matching to image data}} + \underbrace{\iint_{R-\Gamma} \|\nabla u\|^2 + \nu|\Gamma|}_{\text{regularization}}. \tag{2}$$

- R : regions, Γ : contours between regions.
- μ, ν tune the terms contributions.
- Simplification : u is piece-wise constant.

$$E(u,\Gamma) = \sum_i \mu^2 \sum_{R_i} (\underbrace{u}_{=\mu_i} - I)^2 + \nu|\Gamma| \tag{3}$$

Mumford-Shah Model[89] (2/2)

- Finding the best contour Γ is difficult.
- Many heuristic approaches. Examples :
  - by splitting and merging in quadtrees *[Ackahmiezan93]*
  - by a level-set approach *[Chan01]*

Segmentation in a graph

- Image domain seen as a graph (V, A), (usually the grid)
- segmentation = labeling of the graph vertices by minimizing an energy value E
- E = sum of energies on vertices and edges

$$\boxed{E(\lambda) = \underbrace{\sum_{p \in V} U_p(\lambda_p)}_{\text{matching to image data}} + \underbrace{\sum_{\{p,q\} \in A} U_{p,q}(\lambda_p, \lambda_q)}_{\text{regularization}} . \qquad (4)}$$

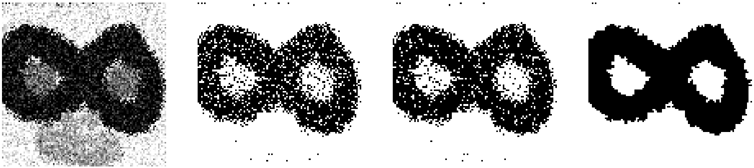
with λ a labeling of the graph vertices.

Examples of energies definitions I

- Image binarization : 2 labels (0 and 255) and

$$
\begin{aligned}
U_p(\lambda_p) &= |I(p) - \lambda_p| \\
U_{p,q}(\lambda_p, \lambda_q) &= -\beta \text{ if } \lambda_p = \lambda_q \\
&= \beta \text{ else}
\end{aligned}
$$

$\beta \nearrow$ = increasing regularization



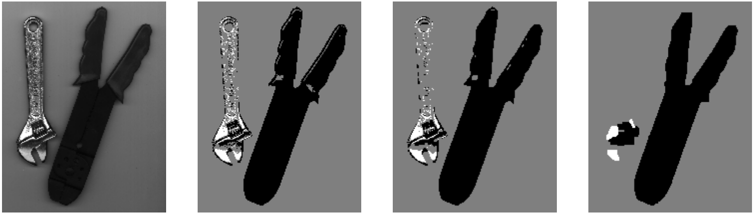
$\beta = 0$ $\beta = 5$ $\beta = 50$

Examples of energies definitions II

- Segmenting into k known classes (μ_i, σ_i)

$$
\begin{aligned}
U_p(\lambda_p) &= \frac{(I(p) - \mu_i)^2}{2\sigma_i^2} \\
U_{p,q}(\lambda_p, \lambda_q) &= -\beta \text{ if } \lambda_p = \lambda_q \\
&= \beta \text{ else}
\end{aligned}
$$

Example : 3 classes, $(\mu_i, \sigma_i) = ((0, 20), (160, 8), (250, 10))$



$\beta = 0$ $\beta = 3$ $\beta = 30$

Energy optimization

Finding the minimum of $E(\lambda)$

- Heuristics : graph cuts *[Boykov01]*
- Stochastic approaches and link with Markov fields : algorithms that compute the global minimum (theoretically)

Conclusion

- Application domains : binarization, segmentation, restoration, video tracking,...
- Applicable to any graphs and not only to adjacency grids of image points
- Intuitive energy formulation
- But the tuning of energies and parameters for a given application is difficult.

Contents

A data structure for image segmentation

How to represent an image partition

Image partition $\Rightarrow$ 2 types of data

- Geometry : region shape
  - set of pixels/voxels
  - region containing a given pixel/voxel
  - boundaries / boundary of a region
- Topology : neighborhood, inclusions
  - set of the regions that are adjacent to a given region
  - region included in a given region
  - including region

Representation of a 2D partition

Inter-pixels boundaries and combinatorial maps

- Partition geometry
  - *segment* : maximum path between 2 regions (shared boundary).
  - *node* : intersection of segments

  The outside of the image is considered as a region.
- Partition topology : planar graph with a matching node/vertex, segment/edge, region/face

Reminder : a permutation is a bijection from a set E to E

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 1 & 2 & 4 \end{pmatrix} \quad \text{cycles} : (1,3)(2,5,4)$$

2D combinatorial map

The partition topology is represented by a combinatorial map : 2 permutations (σ, α)

Remark : each edge of the graph is decomposed into two half-edges (dart)

Permutation σ represents the vertices (a vertex is encoded as the sequence of darts encountered when turning around it in the positive orientation)

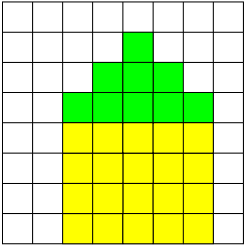
Permutation α represents the edges (= adjacency relation between faces). Let *b* be a dart, $\alpha(b) = -b$. Each dart belongs to only one face (the one at its right).

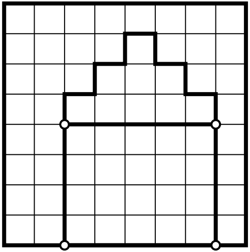
The cycles of permutation $\phi = \sigma \circ \alpha$ correspond to the faces. A function λ labeling the faces is obtained by associating a constant to each cycle of ϕ.

$\lambda(b) =$ face to which belongs *b*.

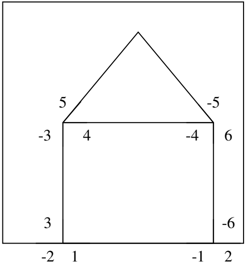
$\lambda^{-1}(face) =$ a dart belonging to the face.

Example of 2D combinatorial map





Géométrie



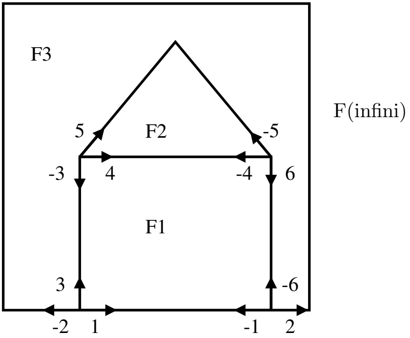
Topologie

$$\sigma = (3, -2, 1)(-6, -1, 2)(-5, -4, 6)(5, -3, 4)$$

$$\sigma = \begin{pmatrix} -6 & -5 & -4 & -3 & -2 & -1 & 1 & 2 & 3 & 4 & 5 & 6 \\ -1 & -4 & 6 & 4 & 1 & 2 & 3 & -6 & -2 & 5 & -3 & -5 \end{pmatrix}$$

$$\alpha = (1, -1)(2, -2)(3, -3)...$$

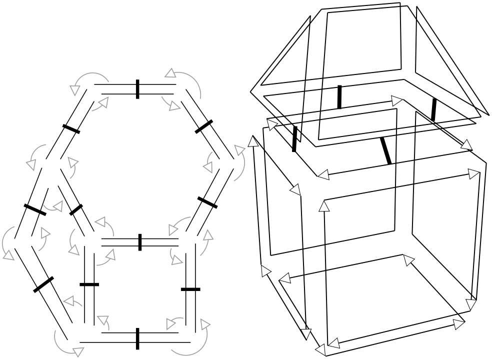
Example of 2D combinatorial map



$$\phi = \begin{pmatrix} -6 & -5 & -4 & -3 & -2 & -1 & 1 & 2 & 3 & 4 & 5 & 6 \\ -5 & -3 & 5 & -2 & -6 & 3 & 2 & 1 & 4 & 6 & -4 & -1 \end{pmatrix}$$

$$\phi = \underbrace{(-4, 5)}_{f_2}\underbrace{(4, 6, -1, 3)}_{f_1}\underbrace{(1, 2)}_{f_\infty}\underbrace{(-5, -3, -2, -6)}_{f_3}$$

From 2D to 3D



additional involution : adjacency relation between volumes