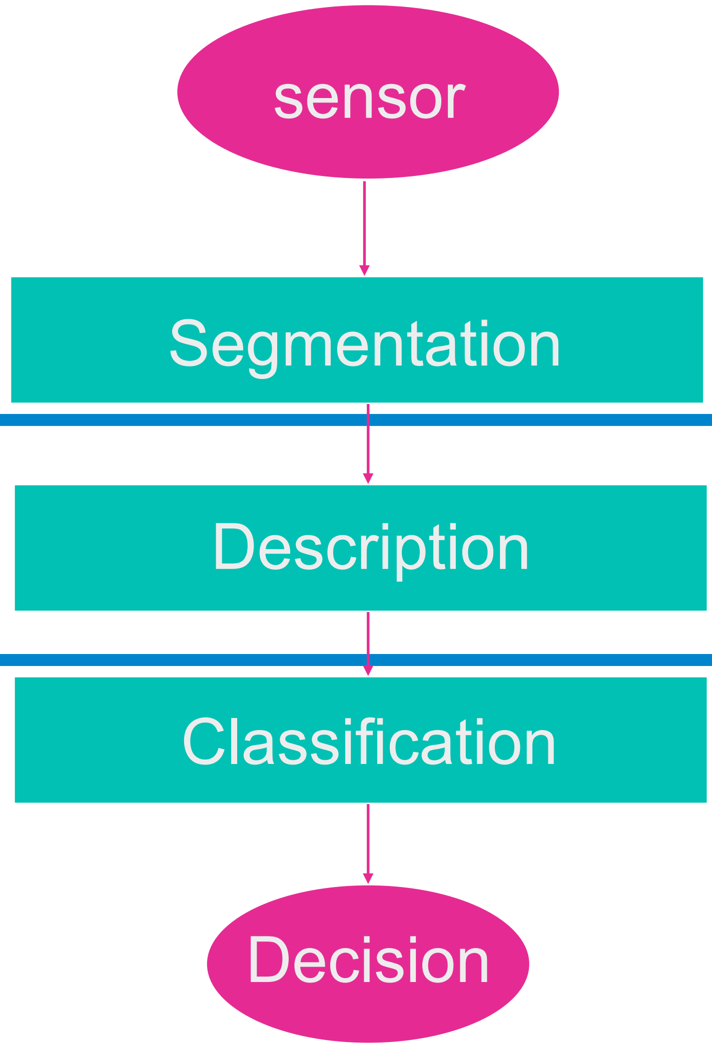
Image analysis and Pattern Recognition

Lecture 4 : object description

Prof. Jean-Philippe THIRAN
JP.Thiran@epfl.ch

The Pattern Recognition System



- Electricity, electronics, metrology
- Signal Processing
- Image Processing
- Geometry, topology, …
- Statistics
- AI
- Robotics, …

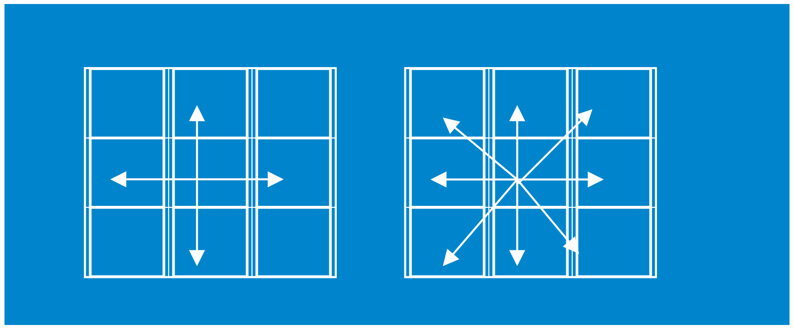
- Goal: present some of the main methods allowing describing a 2D shape by a set of adequate parameters
  - The 2D shape is provided as a binary image as the result of a segmentation task
  - There is a large variety of representations, ranging from “lossless” to very “elementary” ones.
  - The elements of a representation are called features
  - Reference : « Reconnaissance des formes et analyse de scènes », Murat KUNT, éditeur, chapitre 2.

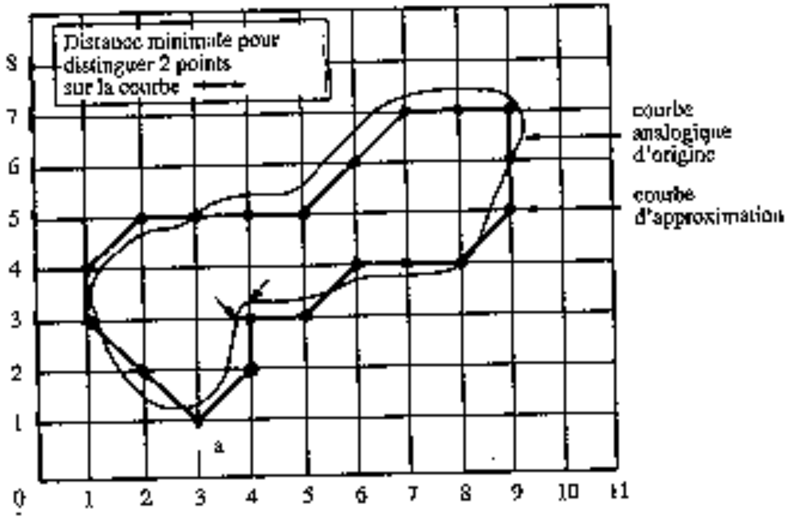
- Pattern recognition approach: object classification:
  - The features to extract should ideally respond to the following requirements:
    - *Small intra-class variance*
    - *Large inter-class variance*
    - *Small number of features*
    - *Independence in translation-rotation (and scaling some times)*

- Twp types of representations:
  - External : description based on the contours of the shape
  - Internal : description based on the inner part

Description based on the contours

- There are mainly two approaches in contour-based description:
  - Local methods: point-to-point description
  - Global methods: description by features calculated globally on the whole contour
- Notice that we treat contours on a discrete grid
  - 4 or 8 connectivity



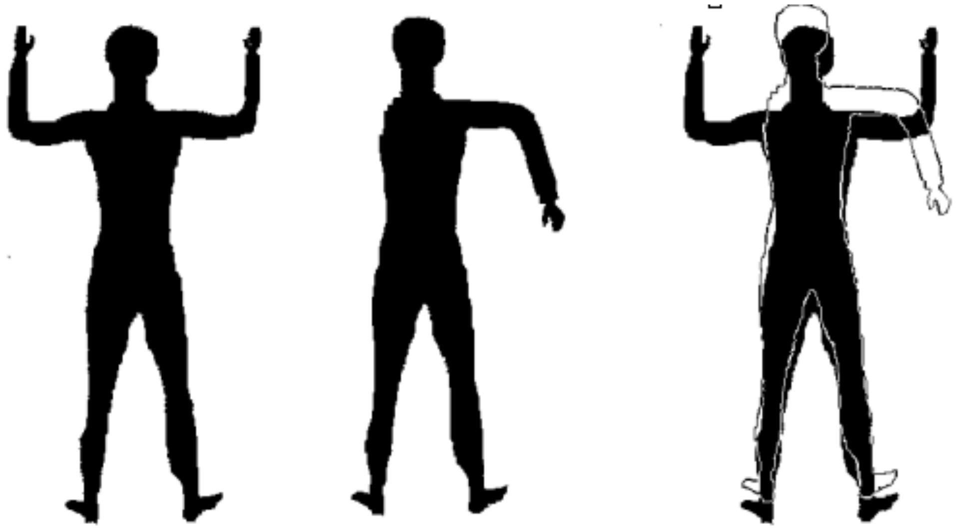


- The most simple feature is the contour itself: the set of border points
  - Not very compact, not easy to handle
  - Subject to noise => smoothing is needed (by mathematical morphology for instance)
- To classify a shape based on its contour, we need to define the similarity between contours, e.g. a notion of distance between a contour and a reference shape of different classes
  - The feature used for classification is then the distance between the shape and the reference shape of the different classes

- Distance between two contours = distance between two sets of points
  - Usual definition of the distance between two sets A and B : mean distance between each point of A and the closest point of B

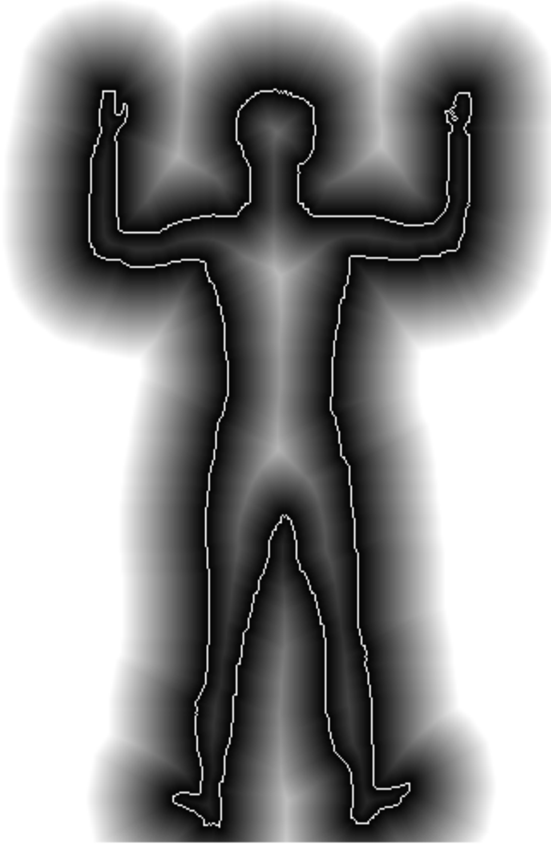
Problems :

- n*m operations
- Euclidean Distance : floating number



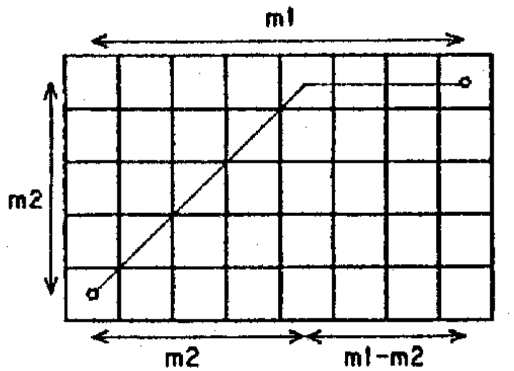
- Simplification 1 :
  distance map
  - To calculate the distance between two sets A and B, we can create a distance map of one of the two sets (say A)
  - v = distance map of A : image where each point has a value that is the distance between that point and the closest point in A
  - Distance between A and B :

$$d_{A,B} = \sqrt{\frac{1}{n} \sum_{(i,j)\in B} v_{i,j}^2}$$

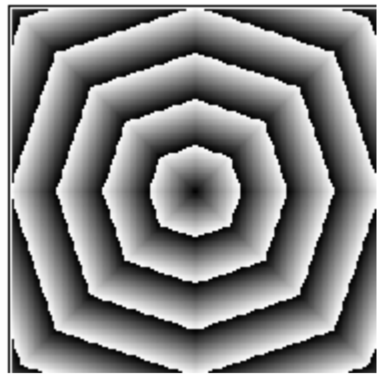


Contour-based description : distance between contours

- Simplification 2 : approximation of the Euclidean distance: *chamfer distance* (distance du chanfrein)
  - Approx of the Euclidean distance



4	3	4
3	0	3
4	3	4



  - Fast algorithm : two passes on the image:

Initialisation : $v(i,j) = 0$ for all the points in the object
$= \infty$ elsewhere

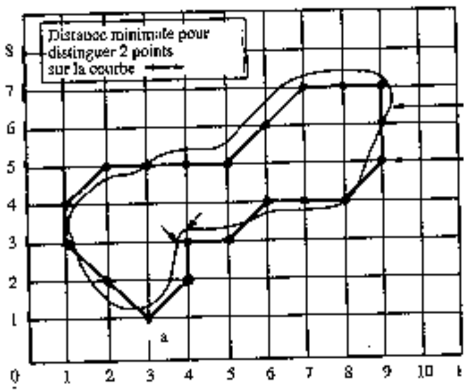
Direct : $v(i,j) = \min\{v(i-i,j-1)+4, v(i,-1j)+3, v(i-1,j+1)+4, v(i,j-1)+3, v(i,j)\}$,
for i=2…nl, j=2…nc

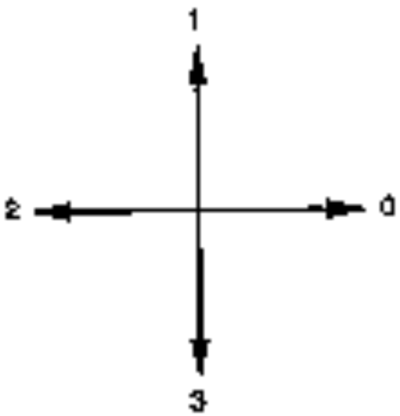
Inverse : $v(i,j) = \min\{v(i,j+1)+3, v(i+1,j-1)+4, v(i+1,j)+3, v(i+1,j+1)+4, v(i,j)\}$,
for i=nl-1…1, j=nc-1…1

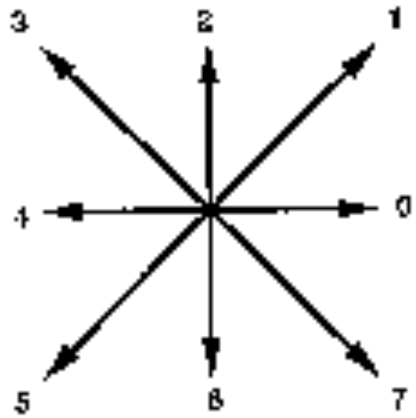
- Then, to calculate the similarity between two objects A and B, we can
  - Define the transformation we want to be invariant to (rotation, translation, scaling, perspective, …)
  - Search the optimal transformation of B that minimizes the distance between A and B
  - This minimal distance is the features used to classify the object B

Contour-based description: chain code

- Chain code or Freeman code
  - Let us define a starting point
    - *E.g. the top left point of the contour*
  - The position of the next point is defined with respect to the position of the previous point, in a given connectivity definition (4 or 8), using a code defined on 2 or 3 bits







  - Example : starting point : a
    - C_8 = *1201001224455444567 (i.e. 60 bits)*

- Interest : chain handling: known problem (in OCR)
- Cfr. Search similar words in a dictionary
  - Detection of sub-parts of an object that have a given shape = detection of sub-chains in a long chain of characters
  - Notion of distance between two chains : edition distance
  - Specification :
    - *Should reflect the difference in length between two chains*
    - *Should reflect the number of different characters between two chains appearing in corresponding positions*
    - *Thus, should reflect the minimum number of elementary operation (insert, suppress, substitute) necessary to transform one chain to the other*

- Example :
  - x = « ababa » and y = « abba »
    - *Replace the « a » in the middle by « b » and suppress the next « b » : 2 operations*
    - *Suppress the « a » in the middle : 1 operation*
    - *Thus the edition distance: $\delta(x,y) = 1$*

- Definition : The edition distance $\delta(x,y)$ two chains x and y is the minimum number of elementary operations (insertion, suppression or substitution) necessary to transform x into y

- There are variants of this definition, that penalizes more some operations:
  - Example : replace b by d or rn by m
- Computing of the edition distance:
  - Combinatory search: prohibitive cost
  - Algorithm of Fisher-Wagner : recursive
    - *Let $x(1\ldots m)$ and $y(1\ldots n)$ be the initial parts, of lengths m and n, of the chains x and y*
    - $D(m,n)=\delta(x(1\ldots m),y(1\ldots n))$

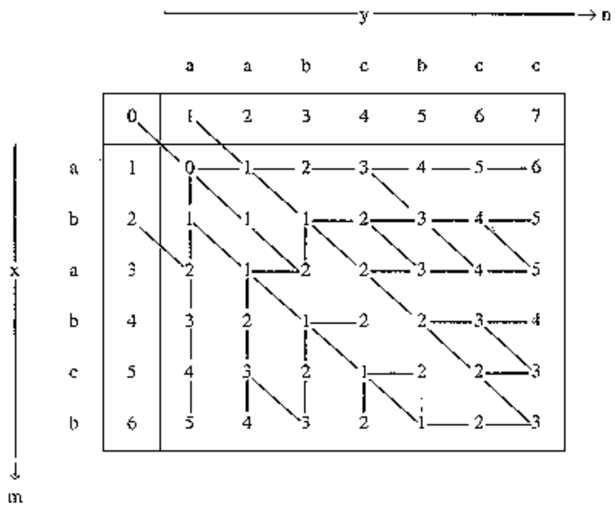
Contour-based description: edition distance

- Fisher-Wagner algorithm:

  $D(m,n) = \min\{\ D(m-1,n-1) + \delta(x(m),y(n)),\ \ D(m-1,n) + 1,$
  $D(m,n-1) + 1\ \}$

  With $\delta(x(m),y(n)) = 0$ if $x(m)= y(n)$ and
  $\delta(x(m),y(n)) = 1$ otherwise

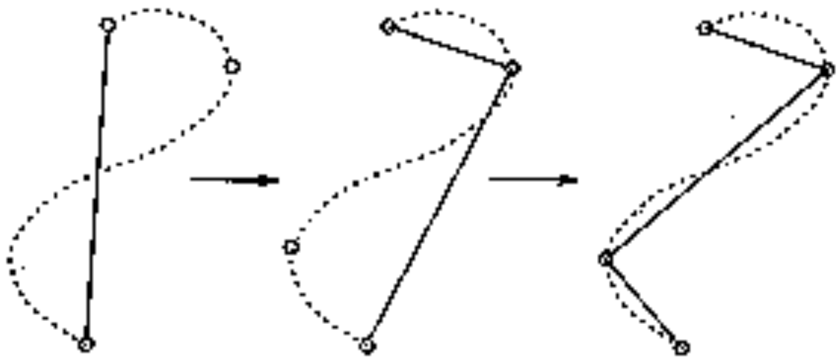
  Finally $\delta(x,y) = D(\text{length}(x),\text{length}(y))$



- Curve Polygonalisation
- Morphological skeleton
- Fourier descriptors

Contour-based description: polygonalisation

- Approximate the curve by a polygon
  - Recursive method by dichotomy
  - Choice of a starting point
  - Look for the most distant point of the curve
  - Look of the most distant point to that line
  - Etc.

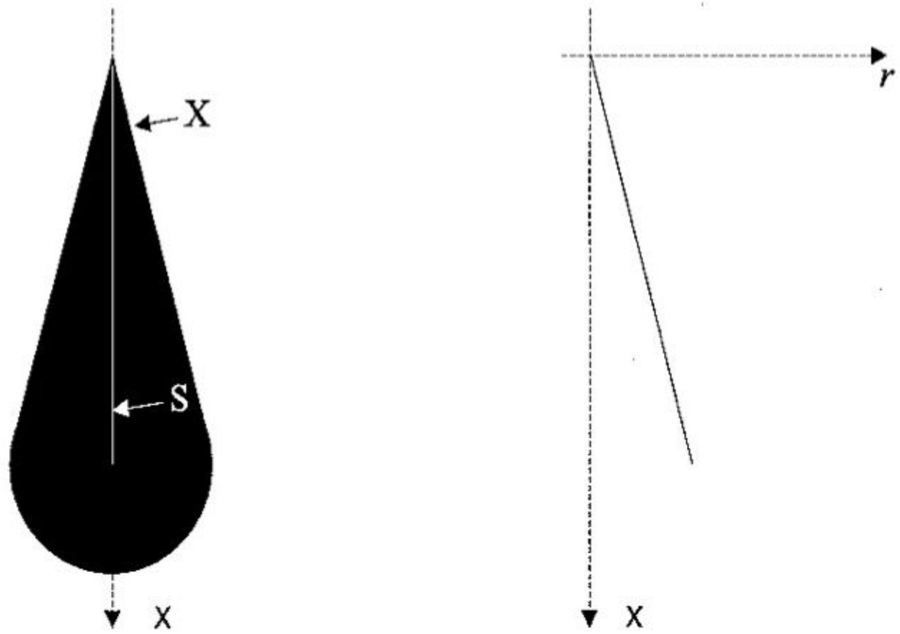


- Interest :
  - Progressive description (i.e with losses)
  - Few parameters (coord of the points, etc.)

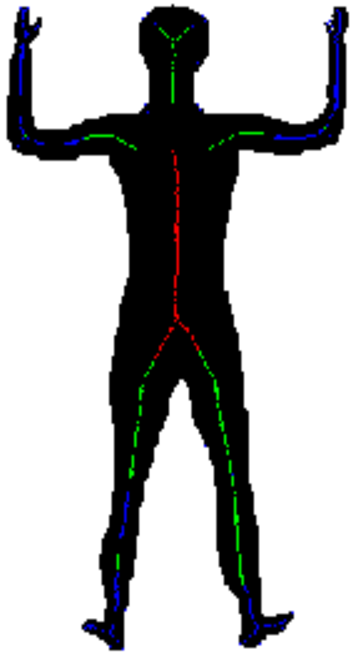
- The morphological skeleton

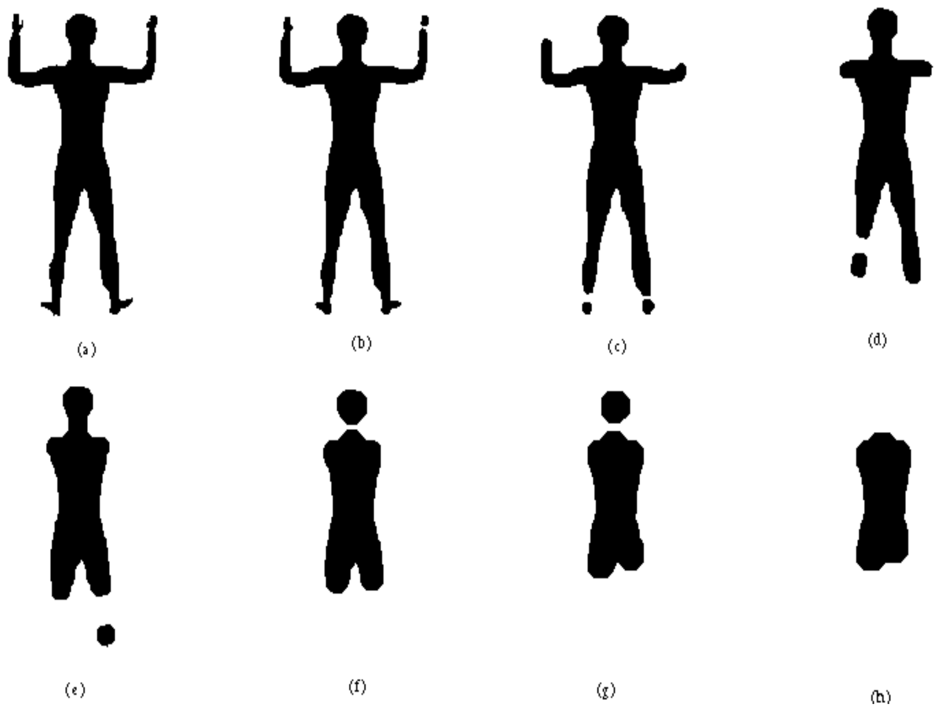
$$S(X)=\bigcup_{r=0}^{N} S_r(X) \quad \text{with} \quad S_r(X)=(X\ominus rB)-(X\ominus rB)\circ B$$

$$X=\bigcup_{r=0}^{N} S_r(X)\oplus rB$$



- Medial axis
- Multi-scale approach

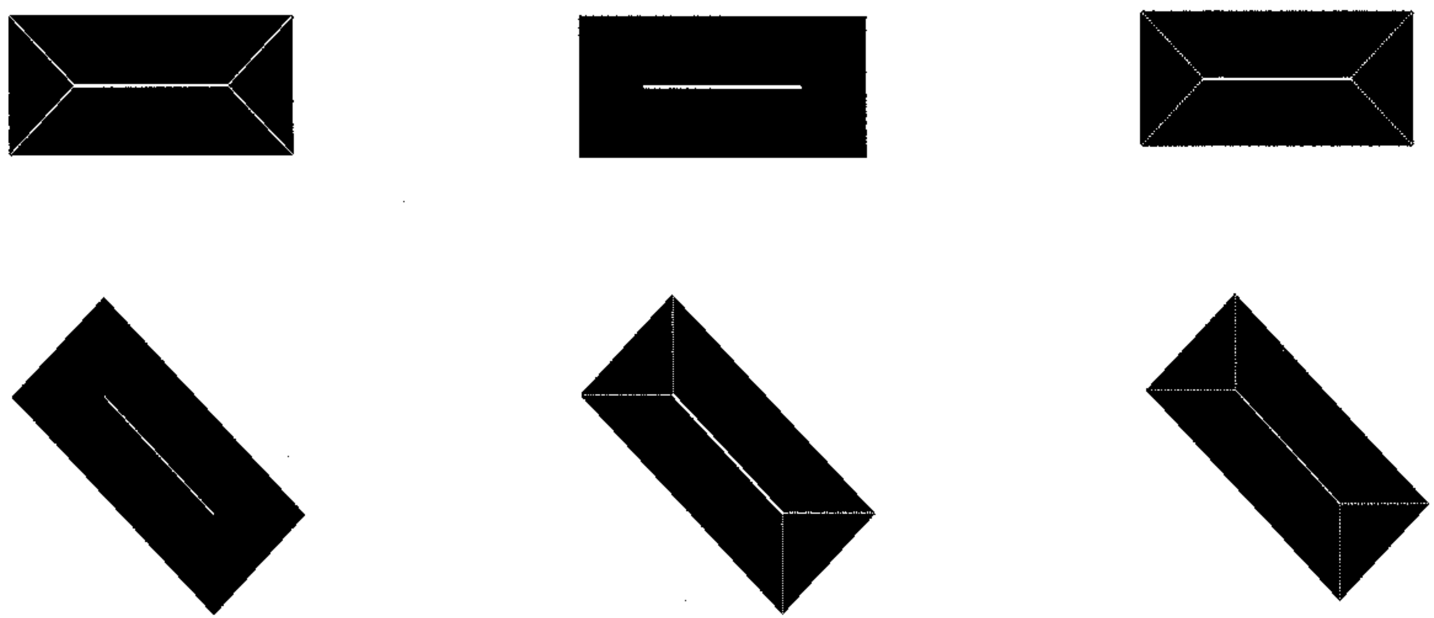




Some algorithmic details

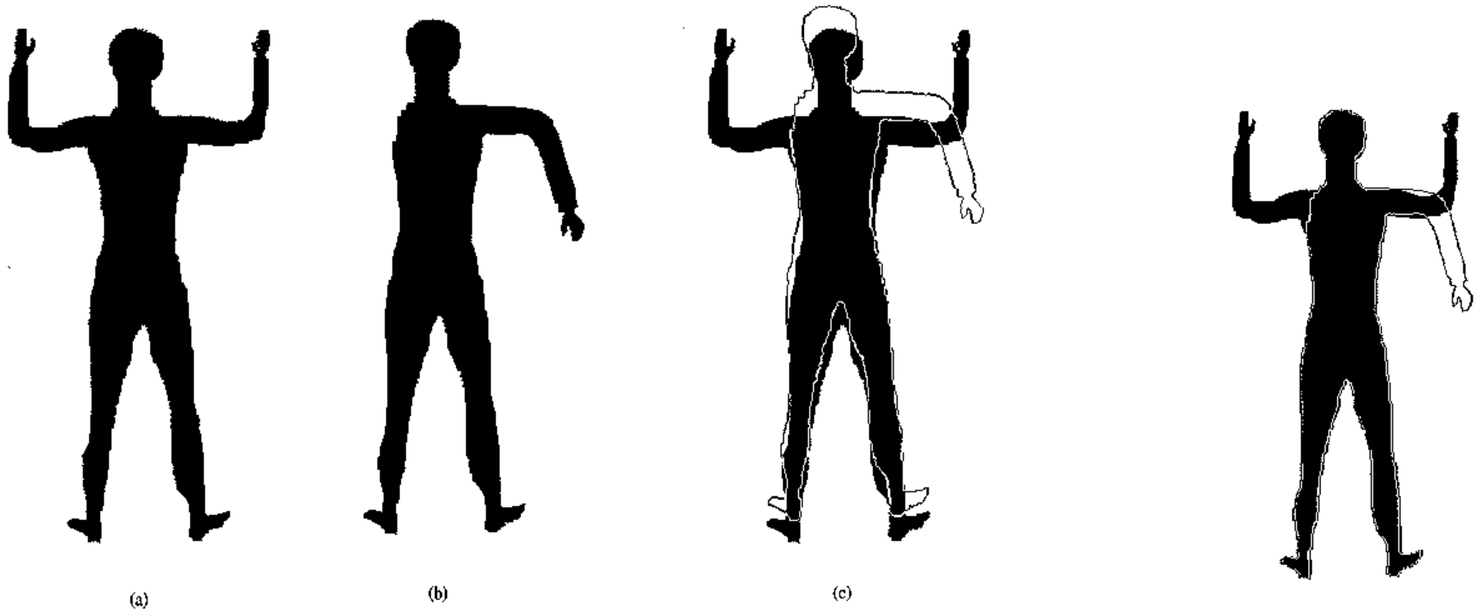
- Invariance in rotation
  - Based on the choice of the structuring element :

cross 3x3 | square 3x3 | combinaison



Shape analysis : morphological comparison

- Similar objects :
  - Main parts should be similar
  - What are the « main parts »??
  - Related to the scale of the object ➡ multi-scale description



- Idea : Fourier transform of the contours: decomposition in « frequencies » : fundamental frequency + high frequency details
- Different descriptions
  - Complex Definition
  - Angular Definition
- Let (x_k, y_k), $k=0...N-1$, be the coordinates of the N successive points of a contour. For each of those points, we define them as complex numbers:

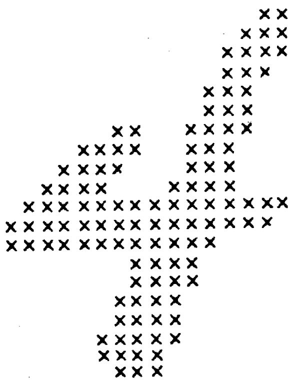
$$u_k = x_k + j\,y_k$$

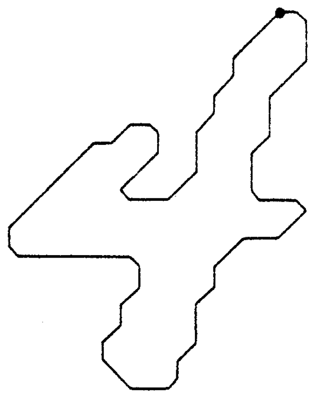
- With those N points u_k, we can calculate their DFT:

$$f_l = \sum_{k=0}^{N-1} u_k e^{-\frac{j2\pi kl}{N}}$$

- The f_l are the Fourier descriptors of the contour. They are complex numbers
- By inverse DFT, we can recover the original contour.

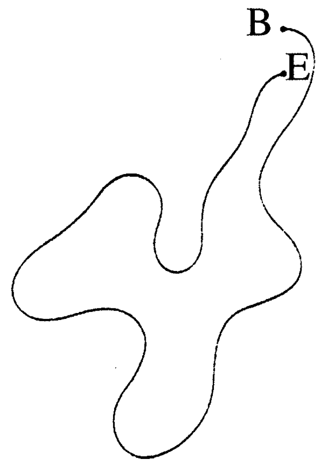
- The first descriptors contain the majority of the shape information of the object





i	A_i	α_i
1	.347	.698
2	.371	5.687
3	1.001	4.989
4	.384	3.279
5	.575	4.588
6	.423	1.084
7	.422	.004
8	.128	4.283
9	.240	4.084
10	.081	4.708

$\mu_0 = -0.75$



Effect of a translation, a rotation or a scaling

- Translation :

$$x'_k = x_k + \Delta x$$

$$y'_k = y_k + \Delta y \qquad \text{thus} \qquad u'_k = u_k + (\Delta x + j\Delta y) = u_k + \Delta u'$$

$$f'_l = \sum_{k=0}^{N-1} u'_k e^{-\frac{j2\pi kl}{N}} = f_l + \sum_{k=0}^{N-1} \Delta u \; e^{-\frac{j2\pi kl}{N}} = f_l + \Delta u \; N\delta(l)$$

  – A translation affect only f'_0

- Rotation :

$$u'_k = u_k e^{j\theta}$$

$$f'_l = \sum_{k=0}^{N-1} u'_k e^{-\frac{j2\pi kl}{N}} = e^{j\theta} \sum_{k=0}^{N-1} u_k \ e^{-\frac{j2\pi kl}{N}} = f_l \ e^{j\theta}$$

  – A rotation affect the phase of all the descriptors by the same amount, and does not modify their amplitude

- Scaling:

$$u'_k = a\, u_k$$

$$f'_l = \sum_{k=0}^{N-1} u'_k e^{-\frac{j2\pi kl}{N}} = a \sum_{k=0}^{N-1} u_k\, e^{-\frac{j2\pi kl}{N}} = a\, f_l$$

  – A scaling does not change the ratio $\dfrac{f_i}{f_j}$

- Choice of u_0 :

$$u'_k = u_{k-k_0}$$

$$f'_l = \sum_{k=0}^{N-1} u'_k e^{-\frac{j2\pi kl}{N}} = \sum_{k=0}^{N-1} u_{k-k_0} e^{-\frac{j2\pi kl}{N}}$$

$$= \sum_{m=0}^{N-1} u_m e^{-\frac{j2\pi(m+k_0)l}{N}} = f_l e^{-\frac{j2\pi k_0 l}{N}}$$

- – The choice of the starting point only affect the phase of the descriptors

- Example of normalization : choice of u_0

$$u'_k = u_{k-k_0} \quad \Rightarrow \quad f'_l = f_l e^{-\frac{j2\pi k_0 l}{N}}$$

thus

$$f'_1 = f_1 e^{-\frac{j2\pi k_0}{N}} \Rightarrow f'_1 = |f_1| e^{-j\phi_1} e^{-\frac{j2\pi k_0}{N}} = |f_1| e^{-j(\phi_1 + \frac{2\pi k_0}{N})} = |f_1| e^{-j(\phi'_1)}$$

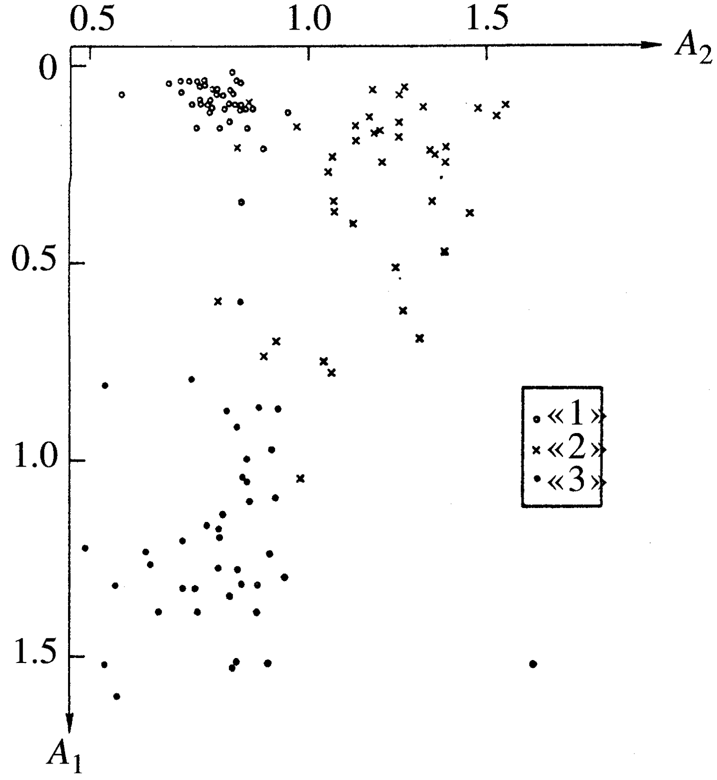
By defining the normalised descriptors by:

$$\hat{f}_l = f_l e^{jl\phi_1},$$

we get

$$\hat{f}'_l = f'_l e^{jl\phi'_1} = f'_l e^{j(l\phi_1 + \frac{2\pi k_0 l}{N})} = f_l e^{jl\phi_1} = \hat{f}_l$$

- By the fact the energy is concentrated in the low frequencies, the Fourier descriptors are powerful features for classification
  - Ex : amplitude of the two first Fourier descriptors

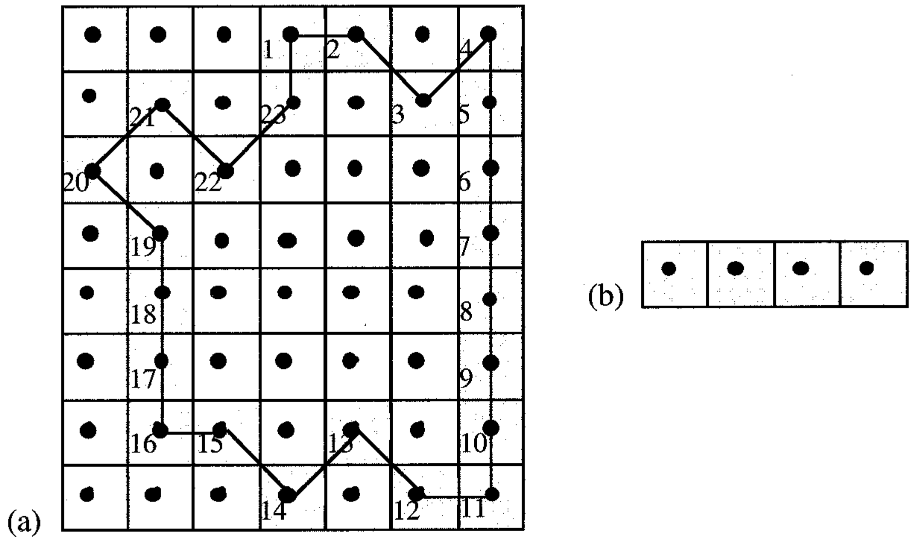


Region-based descriptors

- If we consider the object as a whole, there are many descriptors, more or less complete. Let us describe some of them.
- The surface *A*
  - $A_1 = n.\varepsilon^{2,}$
    - *n is the number of points of the object*
    - *ε is the size of a pixel*
  - Does not vary much w.r.t ε and of the orientation of the shape, because of the cancellation effect of the error

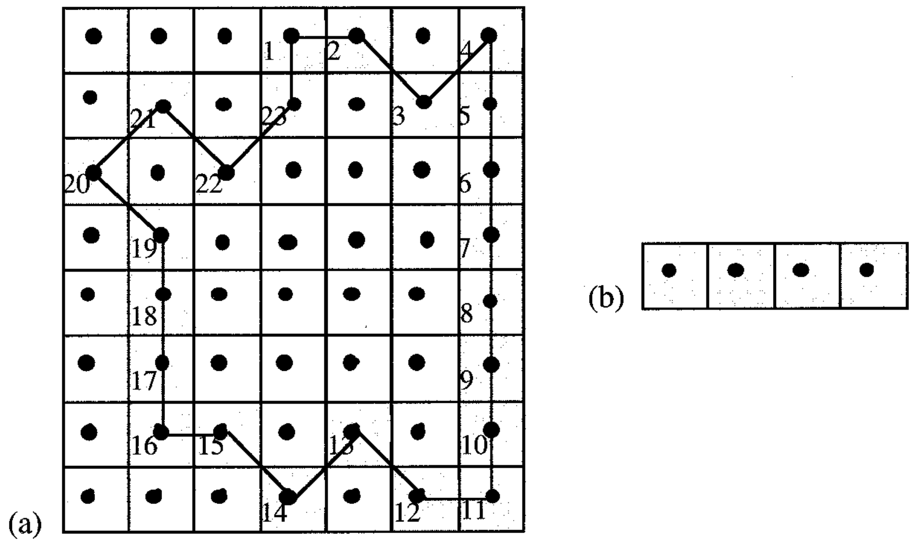
Shape (ε=0.2cm)	**Area**	**A1**	**erreur**
Circle (d=5.44cm)	23.24	23.63 and 23.65	+1.75% et +1.75%
Circle (d=2.50cm)	4.91	4.85 and 4.76	-1.2% et –3%
Square (a=2.54cm)	6.45	6.58 and 6.40	+2% et –0.8%
Rectangle (a=5.08 cm and b = 2.62 cm)	38.71	39.89	+3%
Triangle (5.1 cm, 2.07 cm et 5.1 cm)	13.01	12.98	-0.2%

- Other possibility: area in the polygon connecting the contour points
  - $A_2 = \varepsilon^2\ (b/2+i-1)$
    - *b : number of contour points*
    - *i : number of points inside the contour*

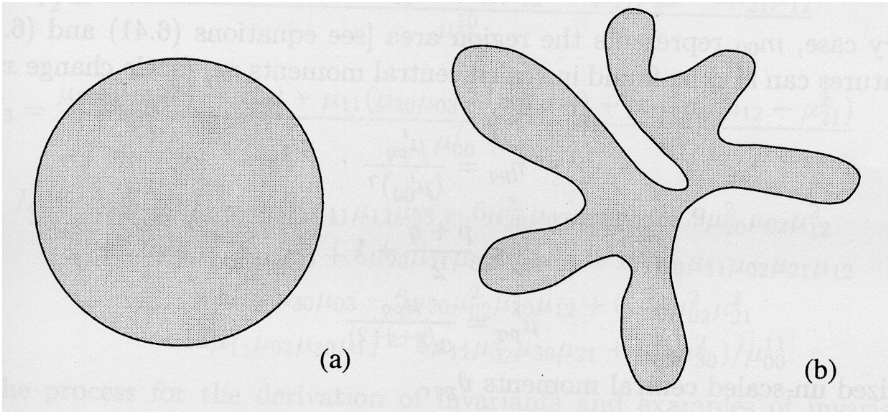


Region-based descriptors: simple ones

- Perimeter
  - Very variable w.r.t. the size of the pixel and the orientation!
  - P_1: surface of the contour points
    - *Ex : (a) P_1=23 and (b) P_1=4*
  - P_2 : follow the contour with +1 and +1.414
    - *Ex : (a) P_2=26.2 and (b) P_2=6*
  - P_3 : length of the polygon connecting the contour points
    - *Ex : (a) P_3=26.2 and (b) P_3=3*

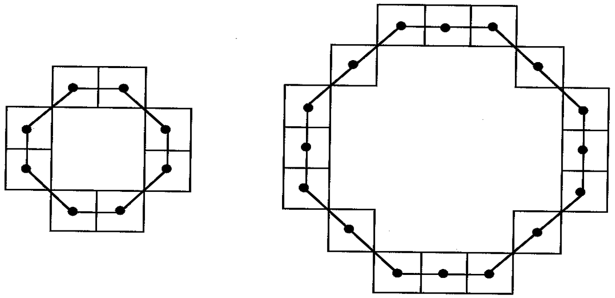


- Compacity $$C = \frac{P^2}{A} \geq 4\pi$$



  – Invariant in translation, rotation and scaling

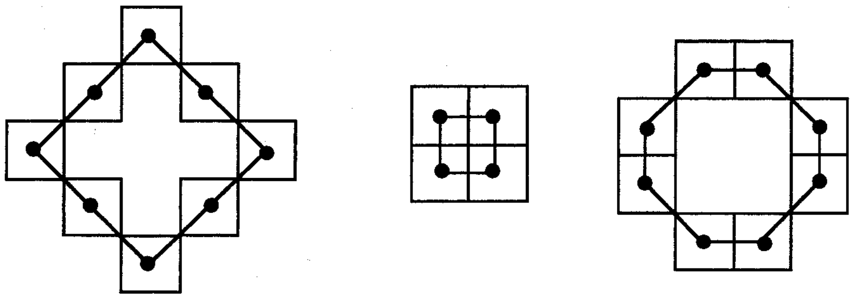
- But sensitive to the definition of A and P



(a) (b)

$4\pi=12.56$. with A_1 and P_3 : (a) : $C=7.77$, (b) : $C=11.64$

  – However with A_2 and P_3



(a) (b) (c)

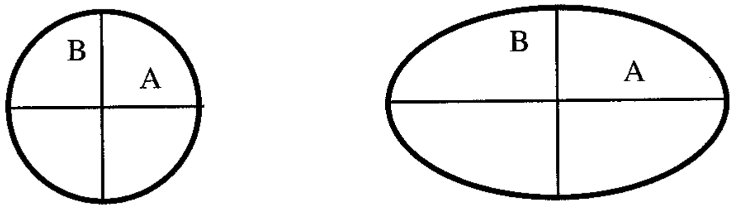
(a) $C=16$, (b) $C=16$, (c) $C=13.3$

- Rectangularity : ratio between the surface of the object and that of the circumscript rectangle

$$\text{rectangularity} = \max_k \left(\frac{A}{Arect_k} \right)$$

$Arect_k$: area of the circumscript rectangle of orientation k

- Elongation : ratio between the maximum diameter of the object and its minimum diameter perpendicular to it.



(a) (b)

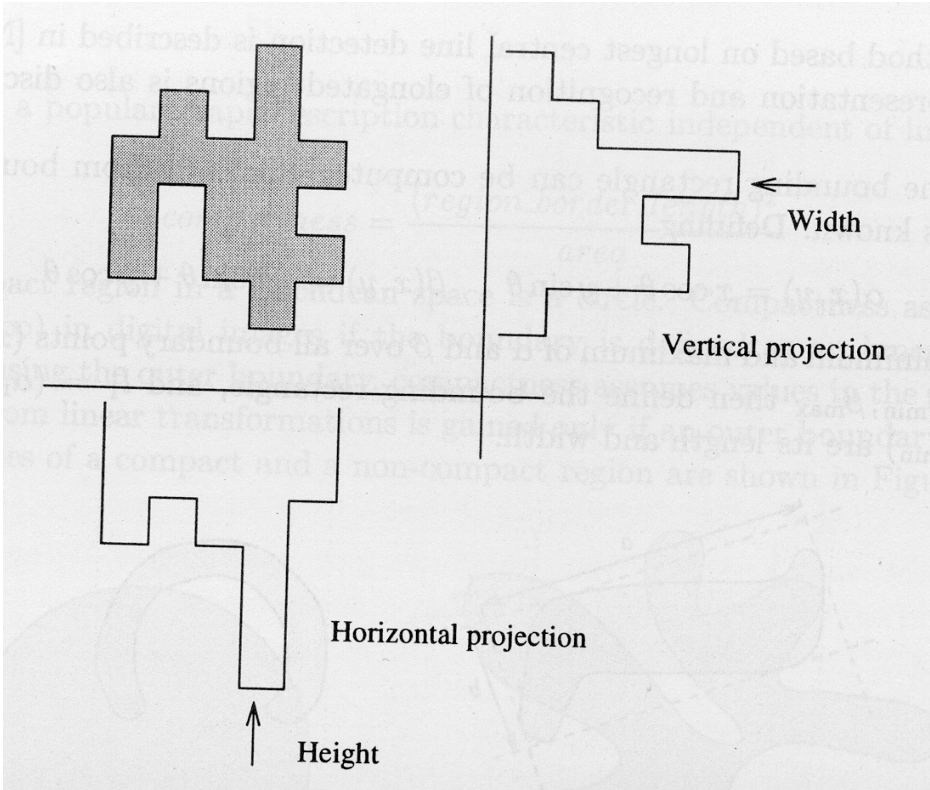
  - Can also be defined as the square root of the ratio of the eigenvalues of the matrix of inertia (see later)

- Projections :

$$p_h(i) = \sum_j f(i,j)$$

$$p_v(j) = \sum_i f(i,j)$$

  - Good basis for a classification
    - *Width and length*
    - *Number of maxima/ minima*



Region-based descriptors: moments

- Moments: used in physics to describe the mass repartition of a body
- For an image :
  - Repartition of the grey levels $f(k,l)$
  - Binary image ($f(k,l)$ = 0 or 1): description of the shape
- Moments : $$M_{i,j} = \sum_k \sum_l k^i l^j f(k,l)$$
- Centre of gravity :

$$\bar{k} = \frac{M_{1,0}}{M_{0,0}} \qquad \bar{l} = \frac{M_{0,1}}{M_{0,0}}$$

- In order to make them invariant to translation, we can choose the center of gravity as origin, and define centered moments:

$$\mu_{i,j} = \sum_k \sum_l (k-\bar{k})^i (l-\bar{l})^j f(k,l)$$

$$\mu_{0,0} = M_{0,0}$$

$$\mu_{1,0} = \mu_{0,1} = 0$$

$$\mu_{1,1} = M_{1,1} - (M_{0,0}.\bar{k}.\bar{l})$$

- In order to make them invariant to rotation, we can define :

$$M_1 = \mu_{2,0} + \mu_{0,2}$$

$$M_2 = (\mu_{2,0} - \mu_{0,2})^2 + 4\mu_{1,1}^2$$

$$M_3 = (\mu_{3,0} - 3\mu_{1,2})^2 + (3\mu_{2,1} - \mu_{0,3})^2$$

$$M_4 = (\mu_{3,0} + \mu_{1,2})^2 + (\mu_{2,1} + \mu_{0,3})^2$$

etc. (see page 43 in the book of M. Kunt)

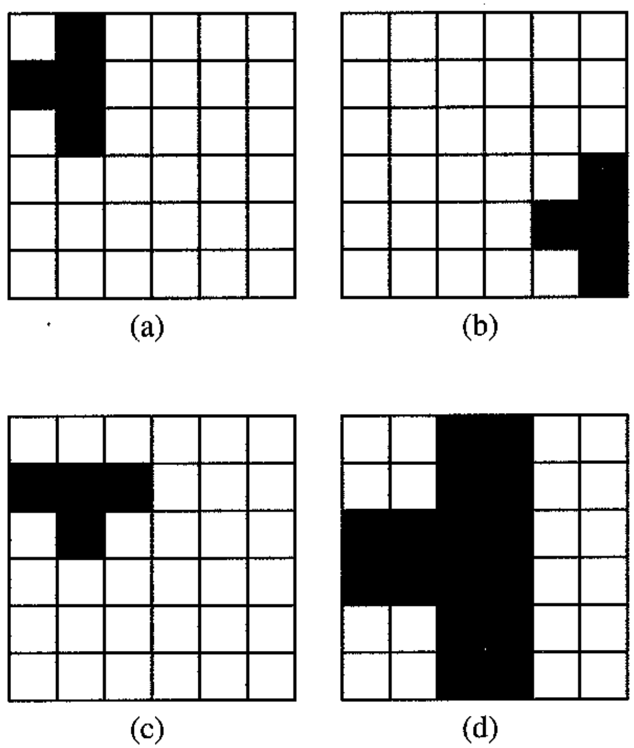
- In order to make them invariant to scaling, we define standard centered moments:

$$\eta_{i,j} = \frac{\mu_{i,j}}{(\mu_{0,0})^{\gamma}}$$

$$avec \quad \gamma = \left\lfloor \frac{i+j}{2} \right\rfloor + 1$$

- By using the standard centered moments with the definition of M_i, we obtain moments M_i' invariant in translation, rotation et scaling

- Example : see page 45 in the book of M. Kunt



- Axes of inertia: an system of axes that minimizes the variance of the shape projected on the axes
- The variance of a shape S w.r.t. an axis **u** is given by

$$
\begin{aligned}
V(S,\mathbf{u}) &= \sum_{x\in S}[\mathbf{u}^T(\mathbf{x}-\bar{\mathbf{x}})]^2 \\
&= \sum_{x\in S}[\mathbf{u}^T(\mathbf{x}-\bar{\mathbf{x}})][(\mathbf{x}-\bar{\mathbf{x}})^T\mathbf{u}] \\
&= \mathbf{u}^T\sum_{x\in S}[(\mathbf{x}-\bar{\mathbf{x}})(\mathbf{x}-\bar{\mathbf{x}})^T]\mathbf{u}
\end{aligned}
$$

- By developing the central term, we get

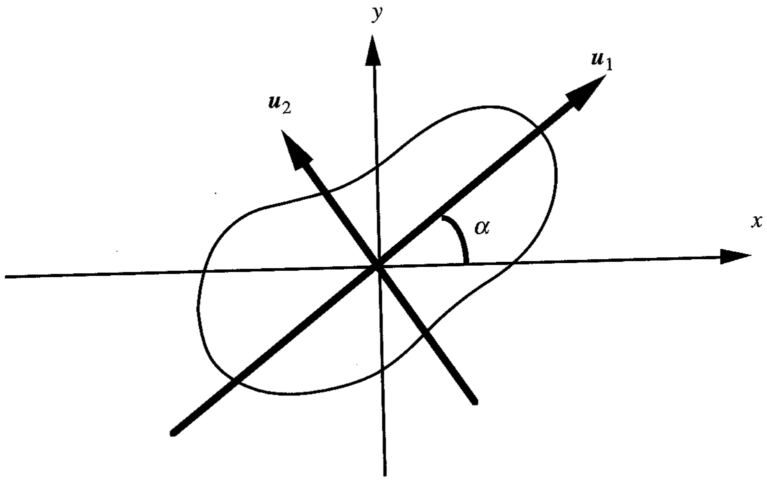
$$T = \begin{pmatrix} x_1 - \overline{x} & x_2 - \overline{x} & \dots & x_N - \overline{x} \\ y_1 - \overline{y} & y_2 - \overline{y} & \dots & y_N - \overline{y} \end{pmatrix} \begin{pmatrix} x_1 - \overline{x} & y_1 - \overline{y} \\ x_2 - \overline{x} & y_2 - \overline{y} \\ \dots & \dots \\ x_N - \overline{x} & y_N - \overline{y} \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{i=1}^{N} (x_i - \overline{x})^2 & \sum_{i=1}^{N} (x_i - \overline{x})(y_i - \overline{y}) \\ \sum_{i=1}^{N} (x_i - \overline{x})(y_i - \overline{y}) & \sum_{i=1}^{N} (y_i - \overline{y})^2 \end{pmatrix} = \begin{pmatrix} \mu_{2,0} & \mu_{1,1} \\ \mu_{1,1} & \mu_{0,2} \end{pmatrix}$$

- This is the Covariance Matrix of the object

Region-based descriptors: axes of inertia

- We have thus $V(S,\mathbf{u}) = \mathbf{u}^T T \mathbf{u}$
- The axes of inertia are the eigenvectors of T
- The eigenvalues express the variance of the shape projected on the axes of inertia



and

$$\alpha = \frac{1}{2} arctg \frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}}$$

- Objects can be described
  - By their contours
    - *Perimeter*
    - *Freeman Codes*
    - *Polygon approximating the contour*
    - *Distance to an object of reference*
  - By morphological skeletons
    - *Multi-scale approach*
  - By region descriptor
    - *Surface, capacity, etc.*
    - *Moments et aces of inertia*

- Pattern recognition consists in choosing good descriptors for a given application and in using a classification algorithm for recognizing the object.