

Chapitre 08  
**Représentation des images**

Aissa Boulmerka  
2020-2021

# Représentation des images

# Introduction

Après la segmentation d'une image en régions, les agrégations de pixels doivent être représentées dans une forme appropriées pour faciliter leur traitement avec l'ordinateur.

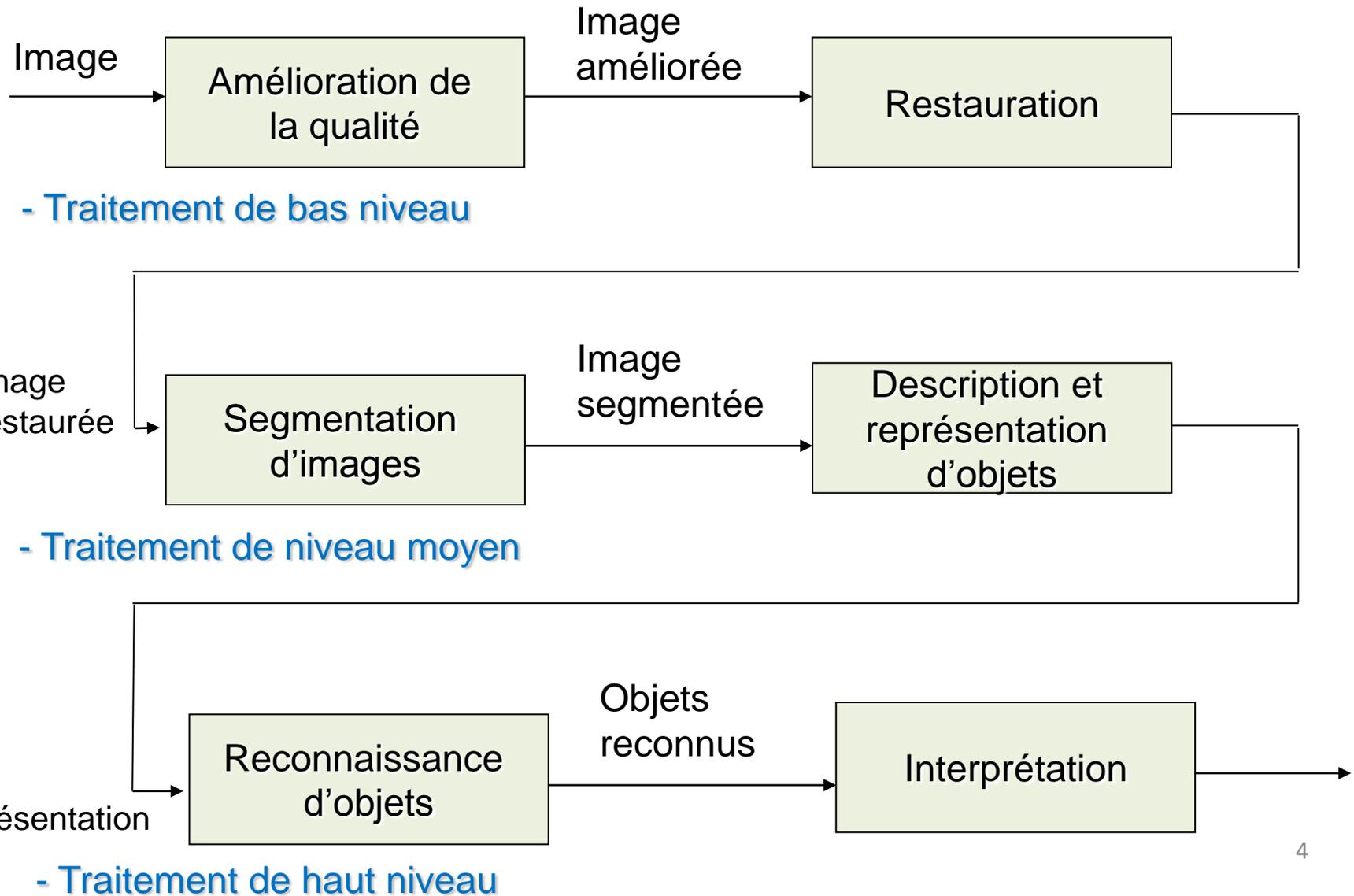
**La représentation d'une région** implique deux choix:

- On peut représenter la région avec **sa frontière**.
- On peut représenter la région avec **son contenu interne**.

Par exemple, la frontière d'une région peut être représentée par sa longueur, son orientation, etc. L'intérieur d'une région peut être représenté par la distribution de la couleur, de la texture, etc.

Les caractéristiques utilisées pour représenter les régions doivent être le moins sensible possible **au changement de taille, la translation et la rotation des objets**.

# Introduction



# **1- Représentation des frontières**

# Introduction

## Parcours de la frontière des régions

Plusieurs algorithmes supposent que les frontières des régions sont ordonnées dans le même sens, ou le sens contraire, des aiguilles d'une montre.

Dans la suite, on va considérer le sens positif de parcours comme le sens de mouvement des aiguilles d'une montre.

On suppose ici qu'on travaille sur **des images binaires** dans lesquelles les objets et le fond sont étiquetés par 1 et 0 respectivement.

En ayant une région ou sa frontière (en image binaire), l'algorithme de parcours de frontières de la régions est comme suit:

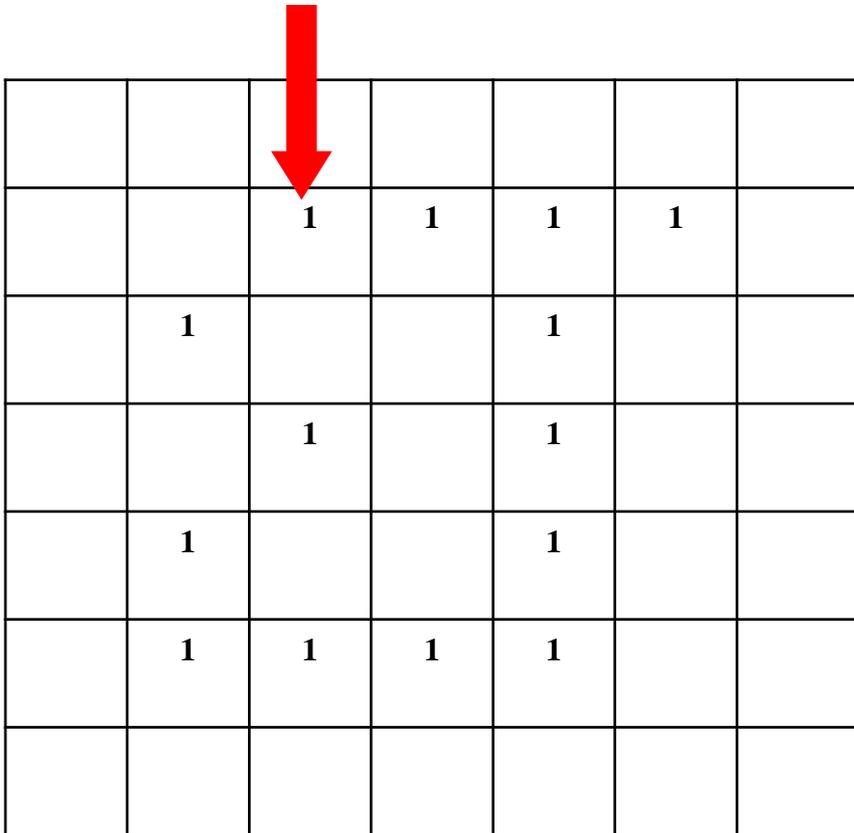
# Algorithme de parcours de frontières

1. Soit  $b_0$  le point de départ de l'objet et  $c_0$  le point à sa gauche.  
Examiner les 8 voisins de  $b_0$ , en commençant par  $c_0$  dans le sens positif. Soit  $b_1$  le premier point rencontré ayant la valeur 1.  
Soit  $c_1$  le point du fond précédent  $b_1$ .
2. Soit  $b = b_1$  et  $c = c_1$ .
3. Soit les 8 voisins de  $b$  commençant à partir de  $c : n_1, n_2, \dots, n_8$ .  
Trouver le premier point  $n_k$  ayant une étiquette 1.
4. Soit  $b = n_k$  et  $c = n_{k-1}$ .
5. Répéter les étapes 3 et 4 jusqu'à ce que  $b = b_0$  et le prochain point est  $b_1$ .

# Algorithme de parcours de frontières

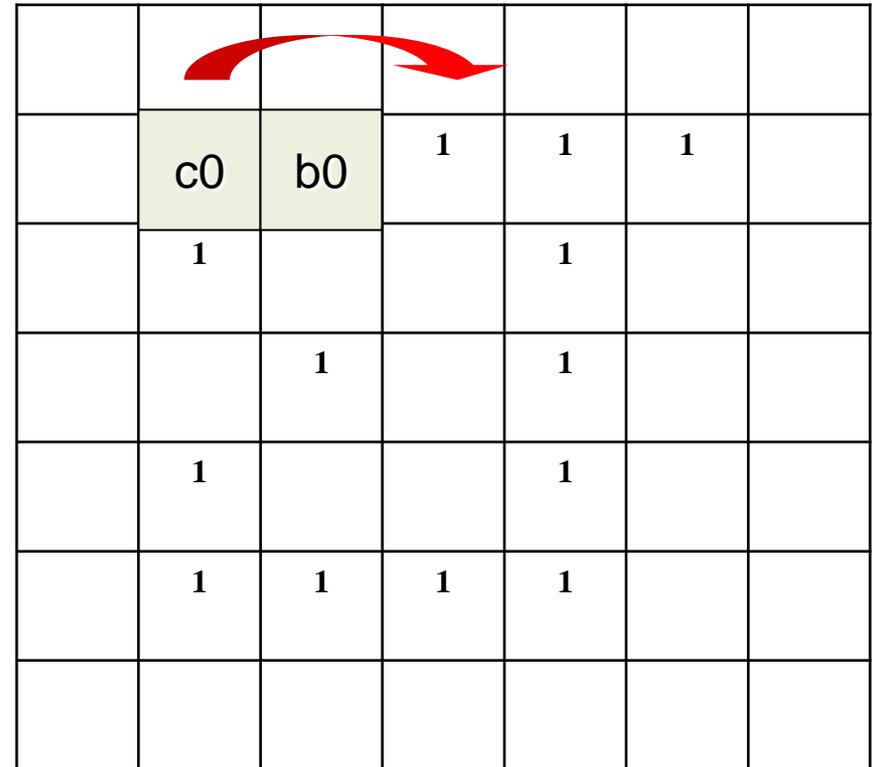
Exemple:

Premier point



		1	1	1	1	
	1			1		
		1		1		
	1			1		
	1	1	1	1		

sens



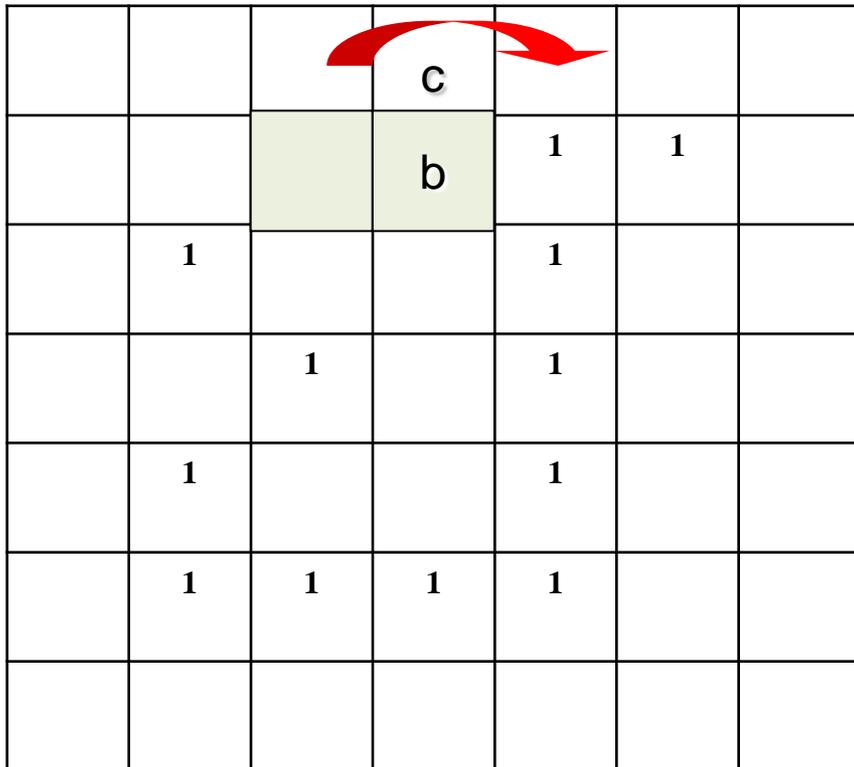
	c0	b0	1	1	1	
	1			1		
		1		1		
	1			1		
	1	1	1	1		

# Algorithme de parcours de frontières

Exemple:

## Principe de l'algorithme:

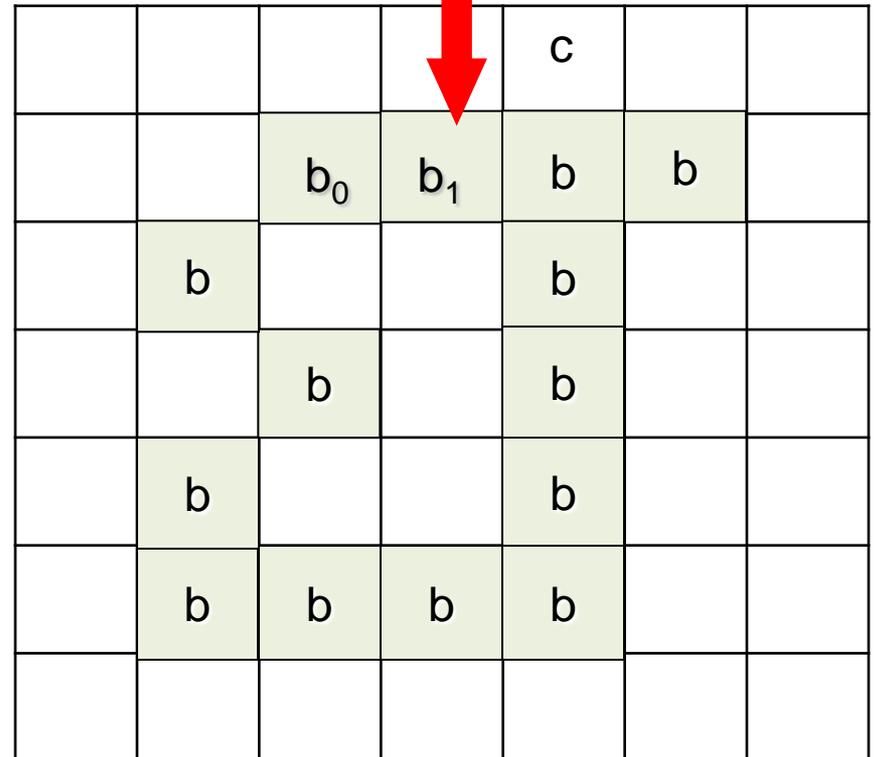
sens



A 7x7 grid with a boundary of '1's. The boundary starts at row 2, column 3 and goes right to row 2, column 5. From row 2, column 5, it goes down to row 6, column 5. From row 6, column 5, it goes left to row 6, column 3. From row 6, column 3, it goes up to row 2, column 3. A red arrow above the grid points from the cell (row 2, column 4) to (row 2, column 5), labeled 'sens'. The cell (row 2, column 4) contains 'c' and the cell (row 2, column 5) contains 'b'. The cells (row 2, column 3), (row 2, column 4), (row 2, column 5), (row 3, column 3), (row 3, column 4), (row 3, column 5), (row 4, column 3), (row 4, column 4), (row 4, column 5), (row 5, column 3), (row 5, column 4), (row 5, column 5), (row 6, column 3), (row 6, column 4), (row 6, column 5) are all '1's. The cells (row 2, column 3), (row 2, column 4), (row 2, column 5), (row 3, column 3), (row 3, column 4), (row 3, column 5), (row 4, column 3), (row 4, column 4), (row 4, column 5), (row 5, column 3), (row 5, column 4), (row 5, column 5), (row 6, column 3), (row 6, column 4), (row 6, column 5) are all '1's.

			c			
			b	1	1	
	1			1		
		1		1		
	1			1		
	1	1	1	1		

Premier point  $b_1$



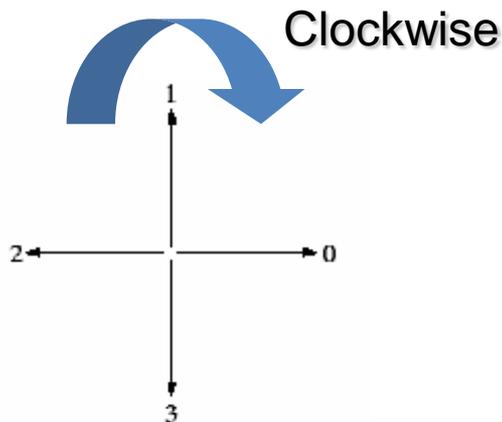
A 7x7 grid showing the boundary path. The path starts at row 2, column 5 (labeled 'c'), goes down to row 6, column 5, then left to row 6, column 3, then up to row 2, column 3. The cells (row 2, column 5), (row 3, column 5), (row 4, column 5), (row 5, column 5), (row 6, column 5), (row 6, column 4), (row 6, column 3), (row 5, column 3), (row 4, column 3), (row 3, column 3), (row 2, column 3) are all 'b'. The cell (row 2, column 4) is labeled 'b<sub>1</sub>' and is the first point of the path. A red arrow points down to this cell. The cell (row 3, column 4) is labeled 'b<sub>0</sub>'. The cells (row 2, column 3), (row 2, column 4), (row 2, column 5), (row 3, column 3), (row 3, column 4), (row 3, column 5), (row 4, column 3), (row 4, column 4), (row 4, column 5), (row 5, column 3), (row 5, column 4), (row 5, column 5), (row 6, column 3), (row 6, column 4), (row 6, column 5) are all '1's.

				c		
		b <sub>0</sub>	b <sub>1</sub>	b	b	
	b			b		
		b		b		
	b			b		
	b	b	b	b		

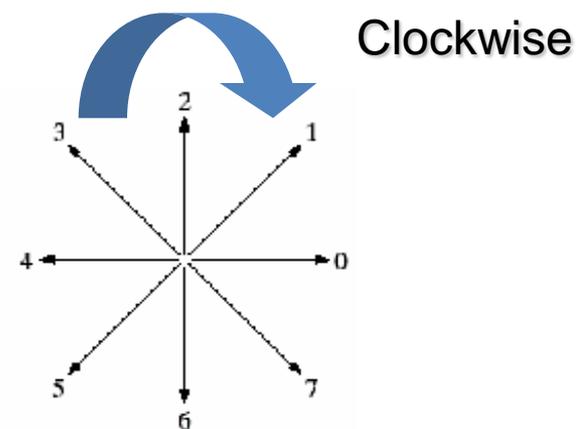
# Code chaine

Permet de représenter la frontière par une séquence connectée de ligne de segment droit. Chaque segment est spécifiée par sa longueur et son orientation.

Le code est basé soit sur le voisinage 4-connexe ou 8-connexes. La direction de chaque segment est codée utilisant le schéma de numérotation suivant.

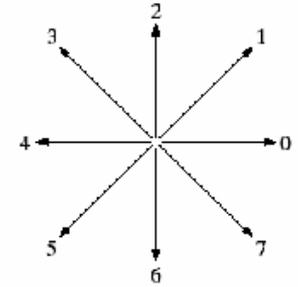


4-directional chain code



8-directional chain code

# Code chaine



Les images numériques sont souvent captées sous forme de grilles régulières dont les cellules sont les pixels. L'application du code chaine sur une image telle quelle a les inconvénients suivants:

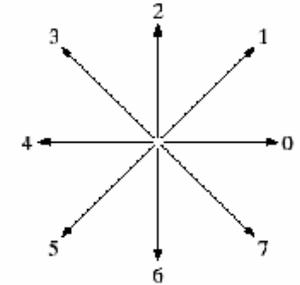
- 1- Le code sera trop long.
- 2- Les erreurs de segmentation et le bruit peuvent produire des perturbations dans le code.

## **Solution:**

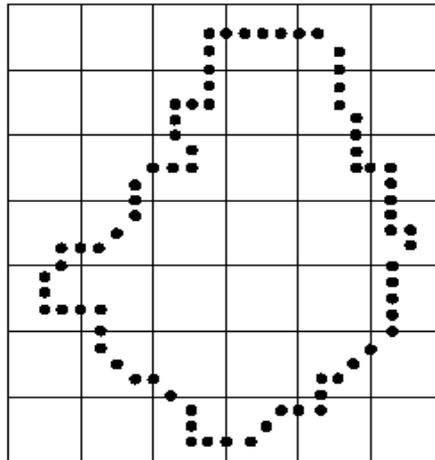
Ré-échantillonner les points de frontières des régions en utilisant des cellules plus grandes que les pixels.

La frontière obtenue par le ré-échantillonnage peut alors être représentée par une chaine à 4 ou 8 voisins.

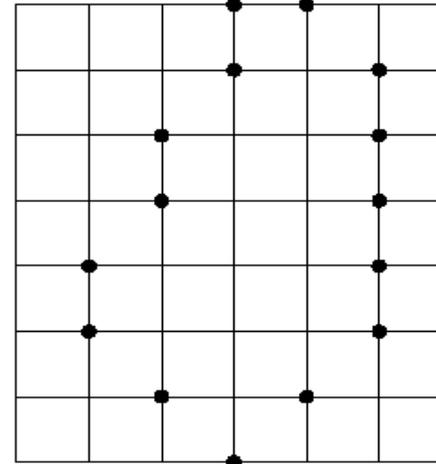
# Code chaine



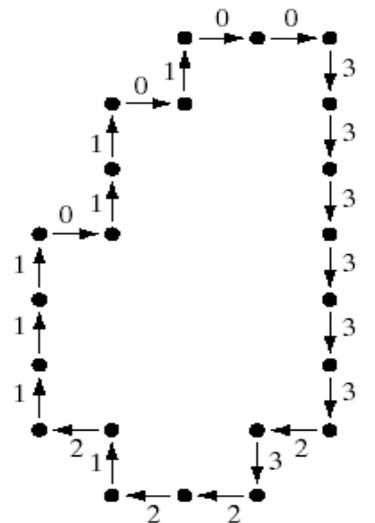
Frontière dans l'image originale



Frontière après ré-échantillonnage

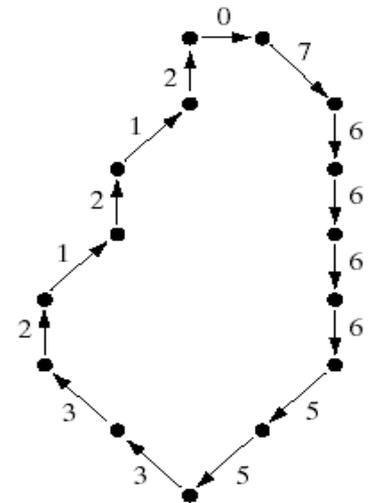


Code à 4 voisins



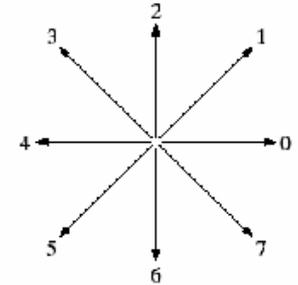
0033333323221211101101

Code à 8 voisins



076666553321212

# Code chaine



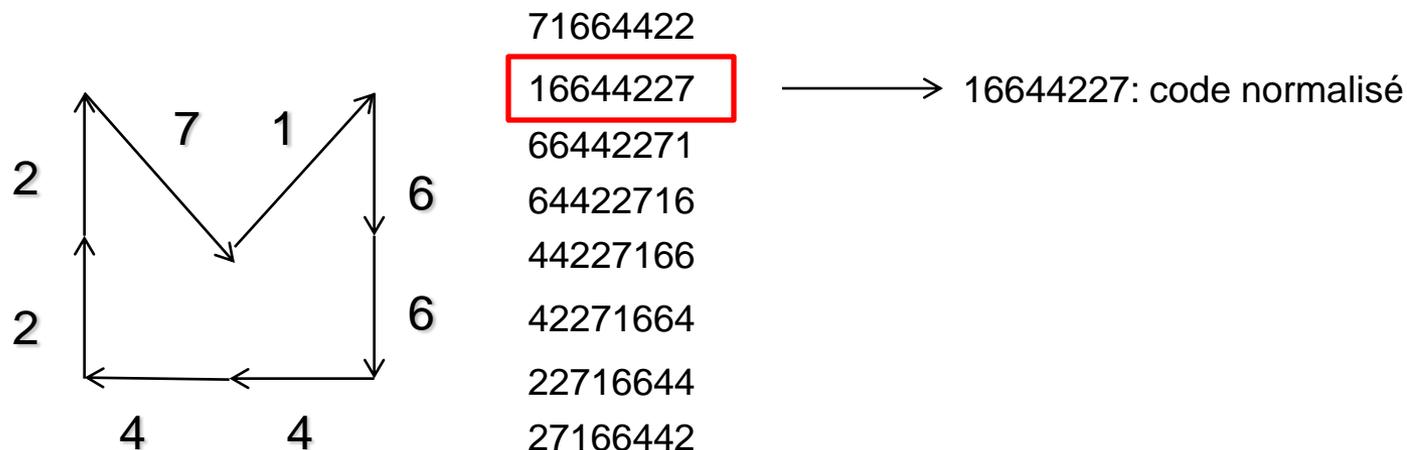
## Remarque1:

Le code d'une frontière dépendra du point de départ pris pour la frontière.

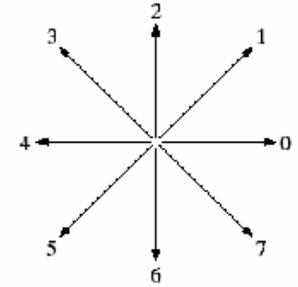
## Solution:

Le code peut être normalisé par rapport au point de départ. L'astuce utilisée est de considérer le code chaine comme **une séquence circulaire de nombres**.

On redéfinit alors le point de départ tel que la séquence résultante produit la **séquence de nombre dont l'amplitude est la plus petite**:



# Code chaine

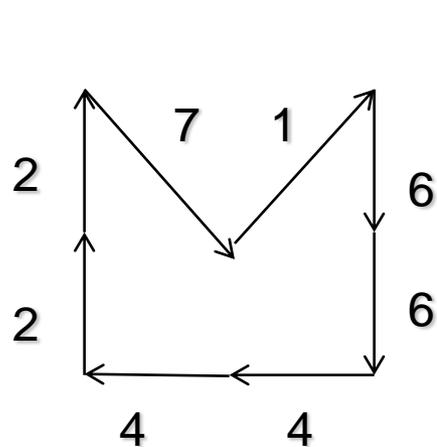


## Remarque2:

Le code d'une frontière est sensible à la rotation de l'objet. Autrement dit, si l'objet subit une rotation, son code va changer; ce qui n'est pas désirable.

## Solution:

Le code peut être normalisé par rapport à la rotation. En utilisant la différence entre nombre adjacents du code chaine au lieu d'utiliser la chaine elle même.



71664422

16644227

66442271

64422716

44227166

42271664

22716644

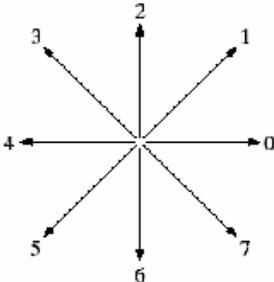
27166442

→ 16644227: code normalisé au point de départ

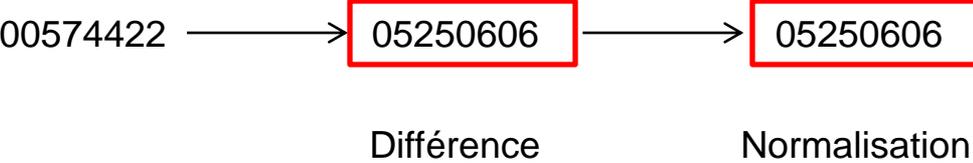
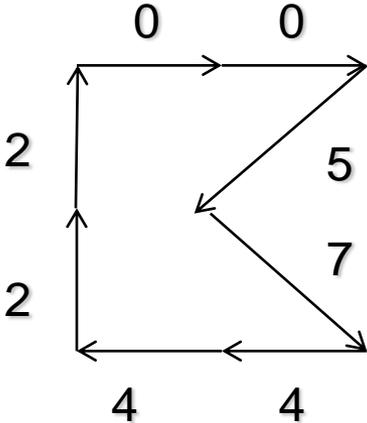
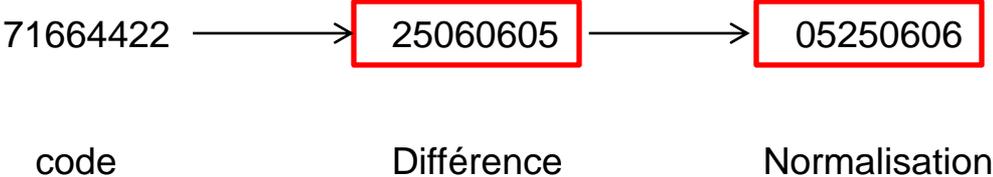
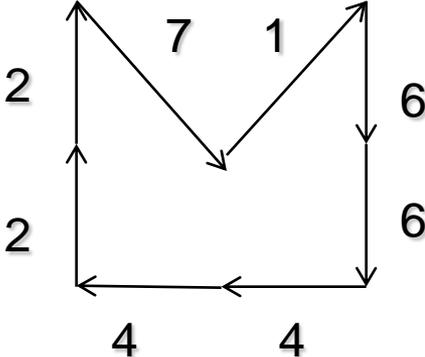


50606052: code normalisé pour la rotation

# Code chaine

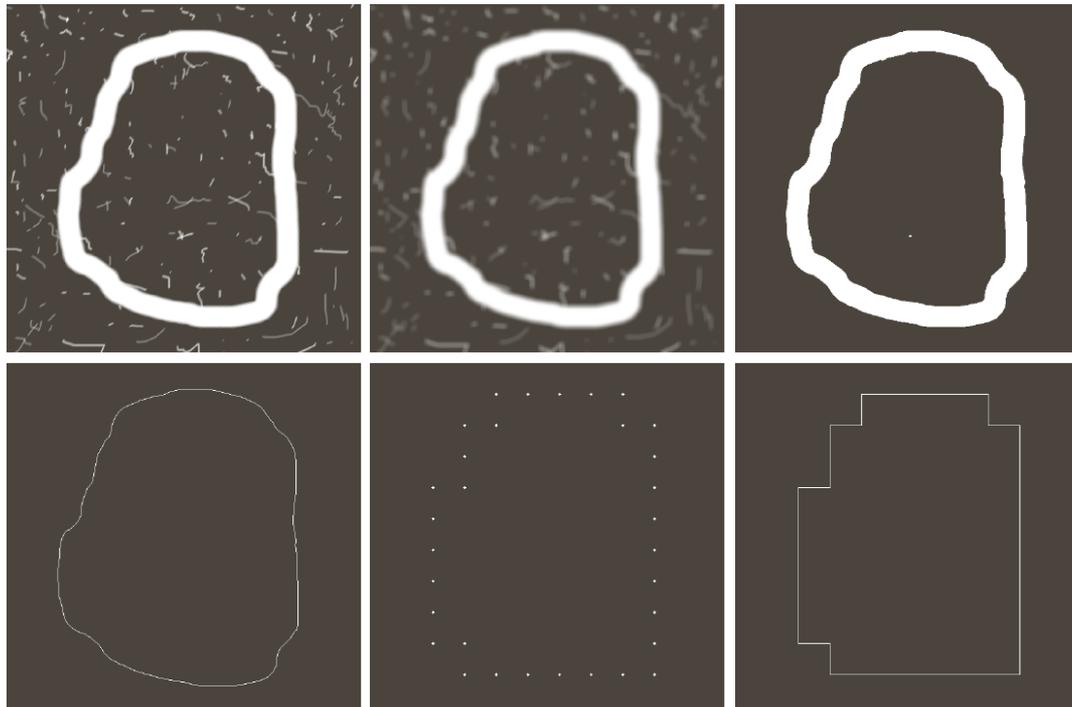
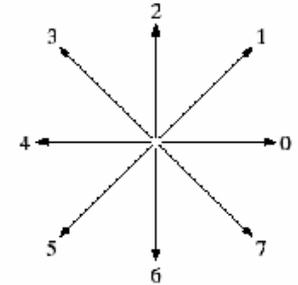


## Exemple 1:



# Code chaîne

## Exemple 2:

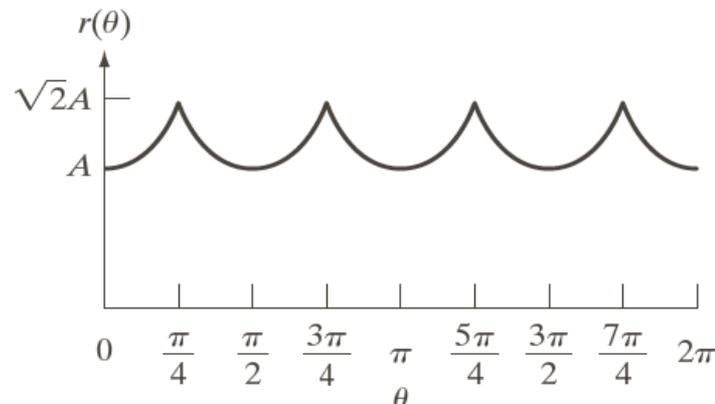
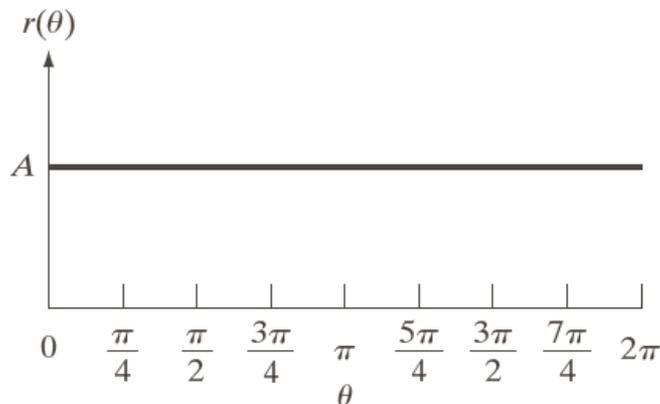
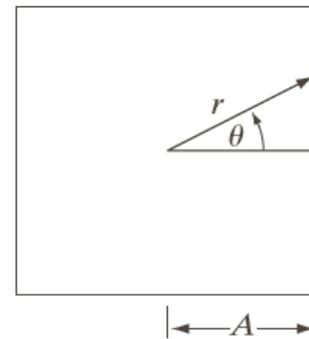
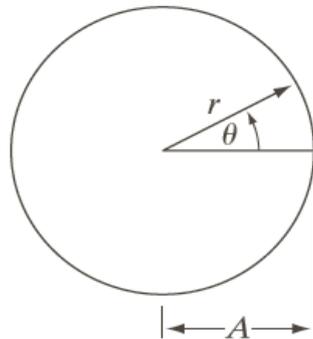


Code original:	00006066666666444444242222202202
Cod normalisé au point de départ:	00006066666666444444242222202202
Code normalisé à la rotation:	00062600000006000006260000620626

# Signatures d'objets

Une signature est une **représentation unidimensionnelle** de la frontière d'un objet. Elle peut être générée de différentes manières.

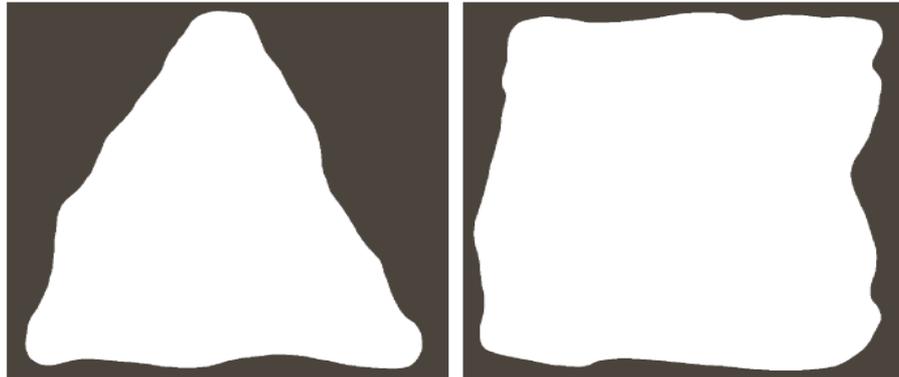
Un exemple simple de signature peut être généré par le graphe de la distance de la frontière au centre de l'objet, comme fonction de l'angle.



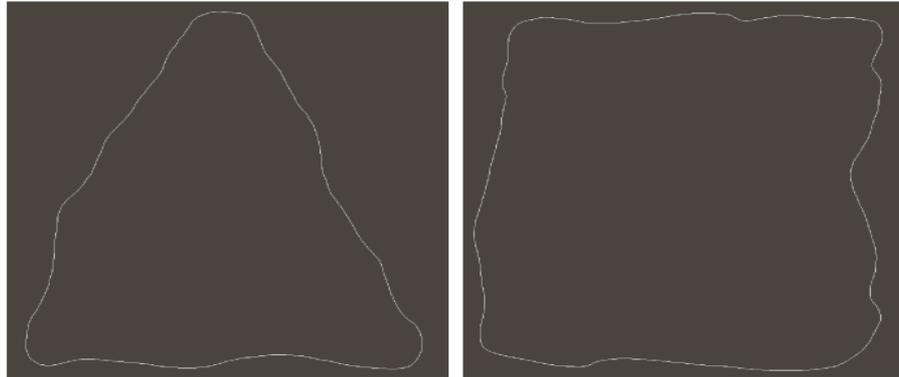
# Signatures d'objets

Exemple:

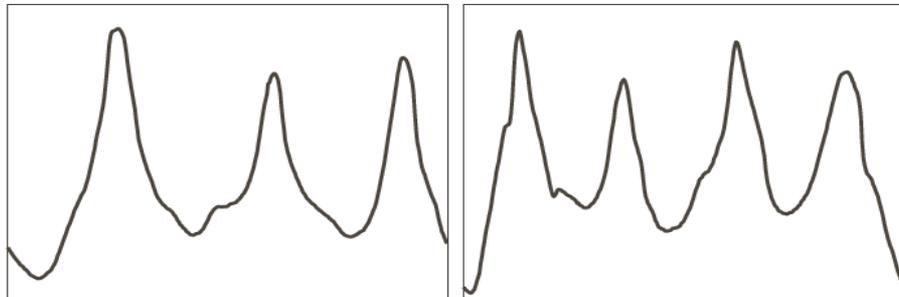
Image binaire  
d'un objet sur  
un fond



Frontière de  
l'objet.



Signature de  
l'objet.



# Signatures d'objets

## Problème:

Cette représentation est invariante à la **translation**, mais elle est sensible à la **rotation** et au **changement d'échelle** de l'objet.

## Solutions proposées:

Pour rendre la représentation invariante à la rotation, **on peut choisir le point de départ de la signature comme le point le plus loin** (le plus prêt) du centre (on supposant que le point est unique).

Pour rendre la représentation invariante au changement de taille des objets, **on peut faire en sorte que les distances calculées soient normalisées et prennent leurs valeurs dans l'intervalle [0,1]**.

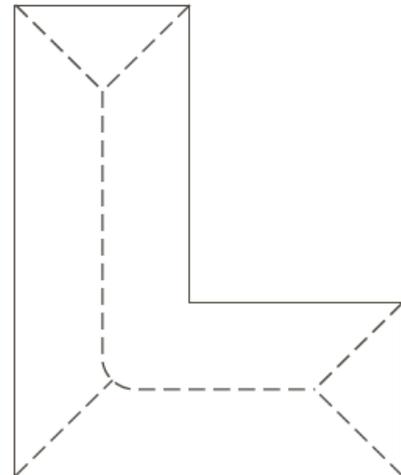
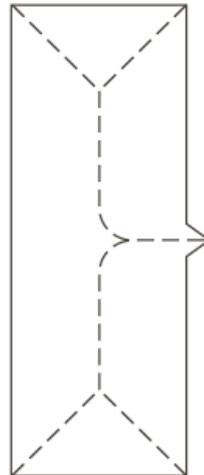
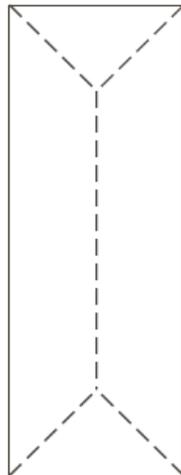
# Squelettes d'objets

Est une approche pour représenter **la forme structurelle** d'un objet sous forme de graphe. La réduction peut être obtenue en construisant le squelette de l'objet via un **algorithme de squelettisation**.

Le squelette d'un objet peut être construit par la méthode de **transformation en axe médian TAM** (*medial axes transformation MAT*). La TAM pour un objet  $R$  qui a une frontière  $B$  et définie comme suit:

Pour chaque point  $p$  à l'intérieur de l'objet, on cherche son point le plus proche sur la frontière  $B$ . S'il y a plus d'un point proche, alors  $p$  va appartenir à l'axe médian.

**Exemples:**



# Squelettes d'objets

La TAM d'un objet a une définition intuitive basée sur ce qu'on appelle «**concept de feu de prairie**». On considère que l'objet est une prairie qui contient de l'herbe sèche distribuée uniformément. On suppose qu'on allume un feu le long de toute la frontière de l'objet. Les fronts du feu vont avancer à la même vitesse.



La MAT est alors définie comme étant tous les points de la prairie qui seront touchés par plus d'un front du feu en même temps.

## Remarque:

L'implémentation directe de la squelettisation est très coûteuse en calcul car elle implique le calcul de la distance de chaque point intérieur de l'objet vers la frontière. On procède en général par des algorithmes plus efficaces.

# Algorithme de squelettisation

- \* On suppose que les pixels de l'objet sont des 1 et le fond des 0.
- \* Un point de frontière doit avoir une valeur 1 et au moins un voisin 0.
- \* On suppose le système des 8 - voisins est représenté par la figure suivante :

$p_9$	$p_2$	$p_3$
$p_8$	$p_1$	$p_4$
$p_7$	$p_6$	$p_5$

- \* L'algorithme se base sur deux étapes itératives principales :

# Algorithme de squelettisation (suite)

1. Étiqueter un point  $p_1$  de la frontière pour la suppression si les conditions suivantes sont remplies:

(a)  $2 \leq N(p_1) \leq 6$

(b)  $T(p_1) = 1$

(c)  $p_2 \cdot p_4 \cdot p_6 = 0$

(d)  $p_4 \cdot p_6 \cdot p_8 = 0$

$N(p_1)$ : est le nombre de point  $\neq 0$  du voisinage de  $p_1$ .

$$N(p_1) = p_2 + \dots + p_8 + p_9$$

$T(p_1)$ : est le nombre de transitions  $0 \rightarrow 1$  dans la séquence :

$$p_2 p_3 \dots p_9.$$

**Exemple:**

0	0	1
1	$p_1$	0
1	0	1

$$N(p_1) = 4.$$

$$T(p_1) = 3.$$

## Algorithme de squelettisation (suite)

Une fois qu'on passe par tous les points de la frontière, on supprime les points étiquetés pour la suppression.

2. Étiquetter un point  $p_1$  de la frontière pour la suppression si les conditions suivantes sont remplies:

(a) Idem que l'étape 1.

(b) Idem que l'étape 1.

$$(c') \quad p_2 \cdot p_4 \cdot p_8 = 0$$

$$(d') \quad p_2 \cdot p_6 \cdot p_8 = 0$$

Une fois qu'on passe par tous les points de la frontière, on supprime réellement les points étiquetés pour la suppression.

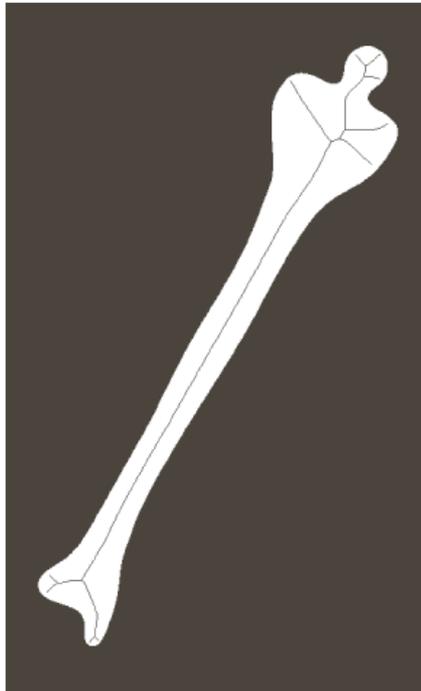
Répéter les étapes 1 et 2 jusqu'à ce qu'il n'y a aucun point étiqueté.

## Algorithme de squelettisation (suite)

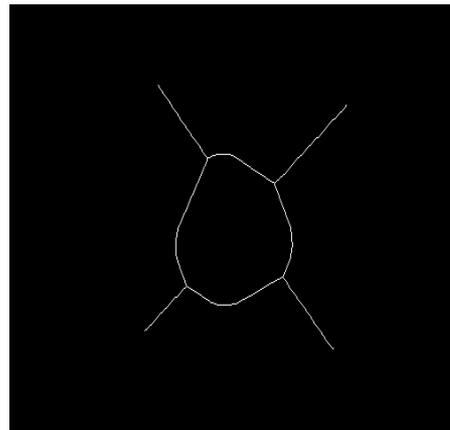
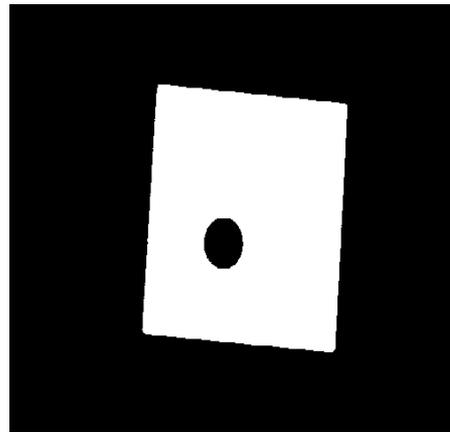
1. La condition (a) est violée quand  $p_1$  possède 1 ou 7 voisins de l'objet.  $p_1$  est alors une arête du squelette qu'on ne doit pas supprimer.
2. La condition (b) est violée quand  $p_1$  est sur une arête. Autrement dit, cette condition empêche la cassure des arêtes du squelette.
3. Les conditions (c) et (d) sont satisfaites par la condition minimale :  
 $(p_4 = 0 \text{ ou } p_6 = 0) \text{ ou } (p_2 = 0 \text{ et } p_8 = 0)$ .  
Si un point satisfait cette condition, ainsi que (a) et (b), alors le point est soit sur la frontière droite ou inférieure.
4. Les conditions (c') et (d') sont satisfaites par la condition minimale :  
 $(p_2 = 0 \text{ ou } p_8 = 0) \text{ ou } (p_4 = 0 \text{ et } p_6 = 0)$ .  
Si un point satisfait cette condition, ainsi que (a) et (b), alors le point est soit sur la frontière gauche ou supérieure.

# Algorithme de squelettisation (suite)

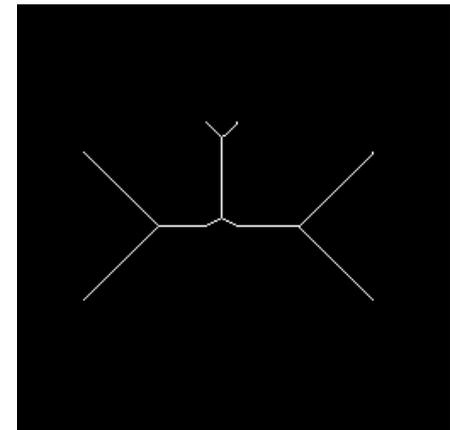
Exemples:



(1)



(2)



(3)

## **2- Représentation des régions**

## Descripteurs simples d'objets

**Aire:** est le nombre de pixels appartenant à l'objet.

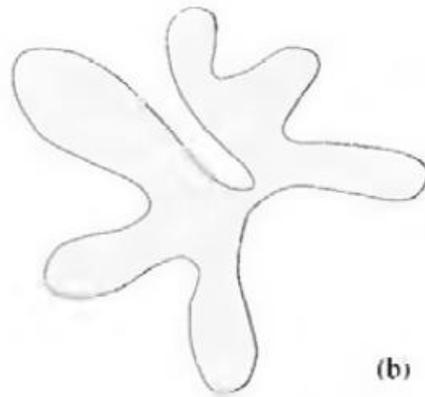
**Périmètre:** est le nombre de pixels appartenant à la frontière de l'objet.

**Compactness** = périmètre<sup>2</sup>/aire.



(a)

(a) compact



(b)

(b) non-compact

# Descripteurs simples d'objets



**Exemple d'application:**  
Calcul de la proportion des régions habitées en utilisant des images infrarouges captées la nuit.



Region no. (from top)	Ratio of lights per region to total lights
1	0.204
2	0.640
3	0.049
4	0.107

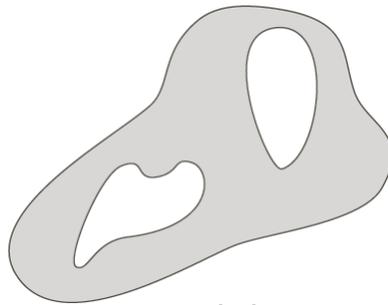


# Descripteurs topologiques

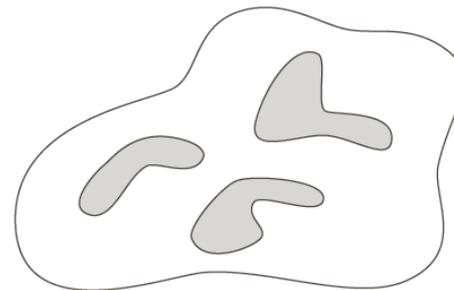
Servent à donner une description globale des propriétés géométriques d'un objet ( ex. connexité, nombre de trous, etc.).

Dans l'exemple (1), l'objet possède 1 composante connexe et 2 trous.

Dans l'exemple (2), l'objet possède 3 composantes connexes et 0 trou.



(1)



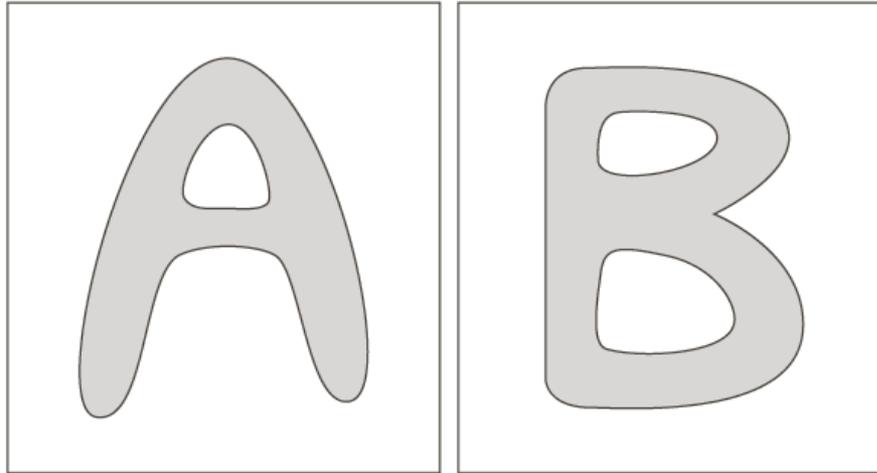
(2)

Les nombres de composantes connexes **C** et de trous **H** sont utilisés pour le **calcul du nombre d'Euler E**, qui est une propriété topologique, définie par la formule suivante:

$$E=C-H$$

# Descripteurs topologiques

Exemples:



(3)

(4)

Dans l'exemple (3),  $E=0$ .

Dans l'exemple (4),  $E=-1$ .

## **3- Représentation des textures**

# Descripteurs statistiques

Soit  $z$  la variable aléatoire qui représente l'intensité de l'image, et soit  $p(z_i)$  l'histogramme de l'image,  $z_i = 0, \dots, L-1$ . On définit le moment d'ordre  $n$  des niveaux de gris comme suit :

$$\mu_n = \sum_{i=0}^{L-1} (z_i - m)^n p(z_i)$$

où  $m$  est la moyenne des niveaux de gris, qui est donnée par :

$$m = \sum_{i=0}^{L-1} z_i p(z_i)$$

À noter que  $\mu_0 = 1$ ,  $\mu_1 = 0$  et  $\mu_2 = \sigma^2$ .

# Descripteurs statistiques

Le descripteur suivant est une mesure de contraste de la texture :

$$R = 1 - \frac{1}{1 + \sigma^2}$$

Si  $R \rightarrow 0$ , la région a une intensité constante.

Si  $R \rightarrow 1$ , la région a une grande variance d'intensité.

Le moment d'ordre 3:

$$\mu_3 = \sum_{i=0}^{L-1} (z_i - m)^3 p(z_i)$$

mesure le *skewness* de l'histogramme.

# Descripteurs statistiques

Finalement, on peut avoir des descripteurs comme :

$$U = \sum_{i=0}^{L-1} (p(z_i))^2$$

qui mesure l'*uniformité* de la texture, et :

$$E = - \sum_{i=0}^{L-1} p(z_i) \log(p(z_i))$$

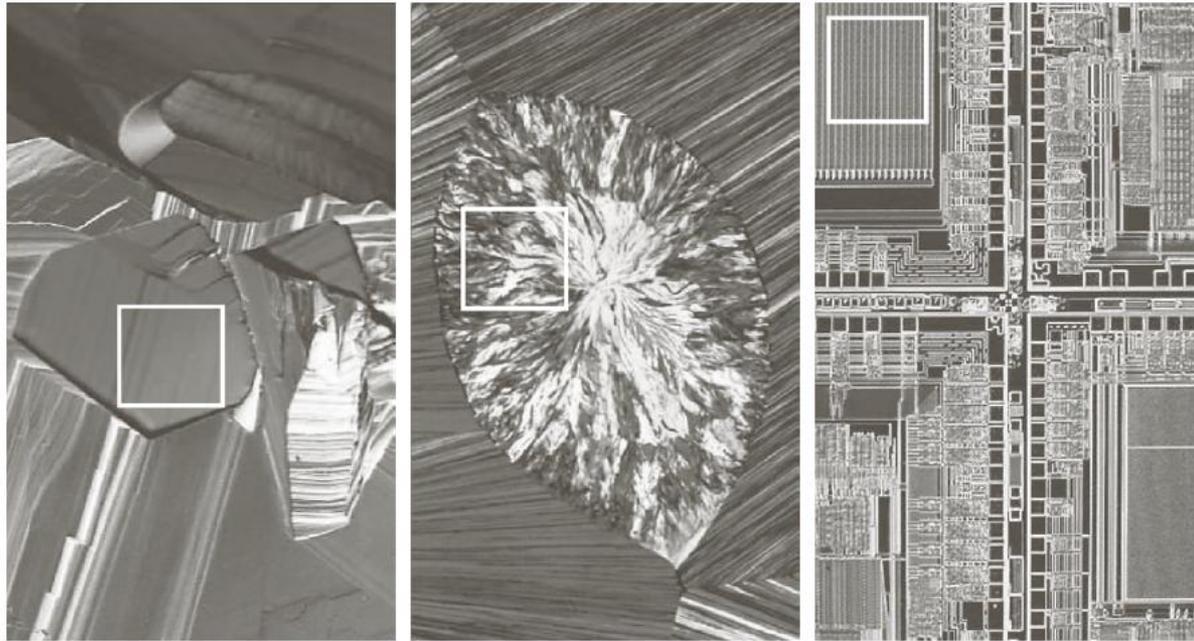
qui mesure l'*entropie* de la texture.

**Remarque:**

*L'uniformité est le contraire de l'entropie.*

# Descripteurs statistiques

Exemples:



Texture lisse

Texture grossière

Texture régulière

Texture	Moyenne	Écart-type	R	$\mu_3$	U	E
lisse	82.64	11.79	0.002	-0.105	0.026	5.434
grossière	143.56	74.63	0.079	-0.151	0.005	7.783
Régulière	99.72	33.73	0.017	0.750	0.013	6.674

# Matrice de cooccurrence

Soit  $Q$  un opérateur qui définit la position de deux pixels de l'image l'un par rapport à l'autre. Soit  $L$  le nombre de niveaux de gris.

Soit  $G$  une matrice dont chaque élément  $g_{ij}$  représente le nombre de fois que les pairs de pixels avec des intensités  $z_i$  et  $z_j$  apparaissent dans l'image suivant la position spécifiée par l'opérateur  $Q$ .

$G$  est appelée une matrice de *cooccurrence*.

**Exemple:** (L=8)

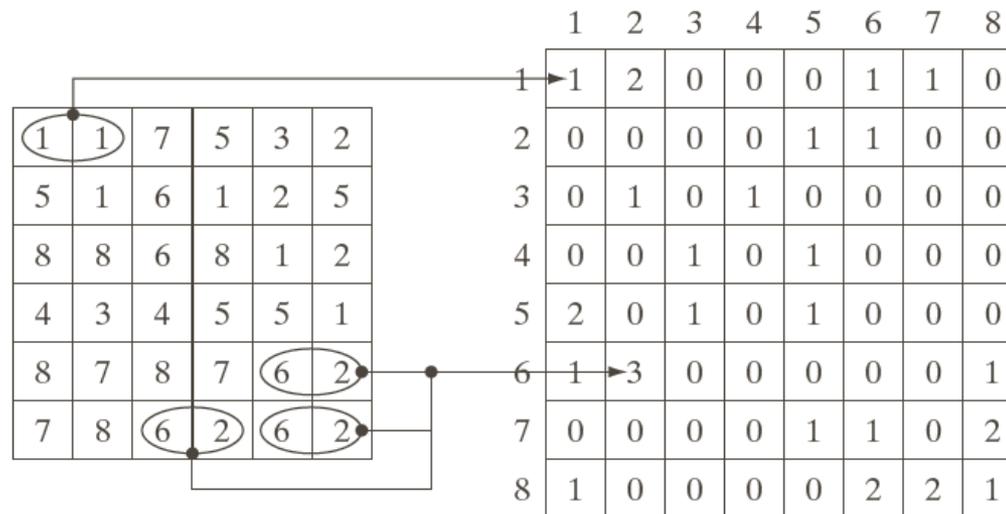


Image  $f$

Co-occurrence matrix  $G$

## Matrice de cooccurrence

Soit  $n$  la somme des éléments de la matrice  $G$  ( $n = 30$  dans l'exemple précédent). On définit alors la quantité :

$$p_{ij} = \frac{g_{ij}}{n}$$

qui est la probabilité que la paire de points satisfaisant l'opérateur  $Q$  ait la valeur  $(z_i, z_j)$ . Il est alors clair que  $p_{ij} \in [0,1]$  et que :

$$\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} p_{ij} = 1$$

On peut calculer plusieurs descripteurs à partir de la matrice de *cooccurrence*.

# Matrice de cooccurrence

**Contraste:**

$$\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} (i-j)^2 p_{ij}$$

Mesure le contraste d'intensité entre chaque pixel et son voisin à travers toute l'image :

**Homogénéité:**

$$\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \frac{p_{ij}}{1 + (i-j)^2}$$

Mesure l'homogénéité de la texture (contraire du contraste).

# Matrice de cooccurrence

**Contraste:**

$$\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} (i-j)^2 p_{ij} \in [0, (L-1)^2]$$

Mesure le contraste d'intensité entre chaque pixel et son voisin à travers toute l'image :

**Homogénéité:**

$$\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} \frac{p_{ij}}{1 + (i-j)^2} \in [0,1]$$

Mesure l'homogénéité de la texture (contraire du contraste). Autrement dit, elle mesure la proximité spatiale de la distribution des éléments de  $G$  sur sa diagonale.

# Matrice de cooccurrence

**Uniformité:**

$$\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} p_{ij}^2 \in [0,1]$$

Mesure l'uniformité de la texture. Elle est égale à 1 pour une image constante.

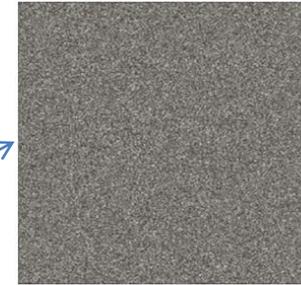
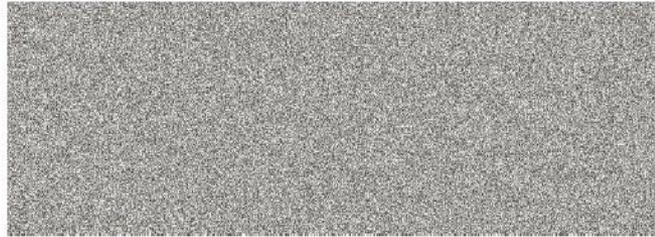
**Entropie:**

$$-\sum_{i=0}^{L-1} \sum_{j=0}^{L-1} p_{ij} \log_2(p_{ij}) \in [0, 2 \log_2 L]$$

Mesure le degré auquel les éléments de  $G$  sont aléatoires.  
(contraire de la texture régulière)

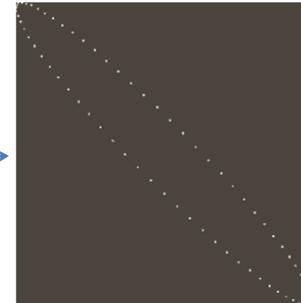
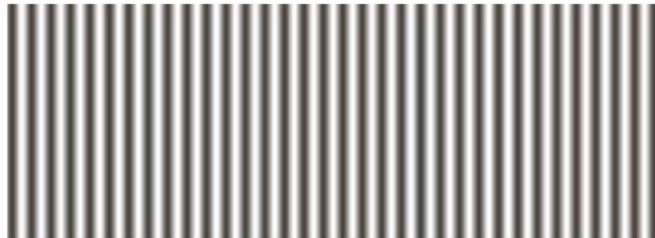
# Matrice de cooccurrence

Texture 1



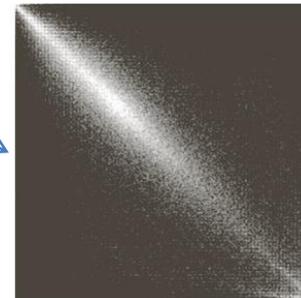
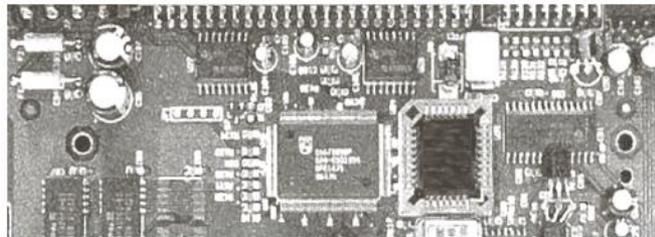
Matrice de cooccurrence

Texture 2



Matrice de cooccurrence

Texture 3



Matrice de cooccurrence

Texture	Contraste	Uniformité	Homogénéité	Entropie
Texture 1	10838	0.00002	0.002	15.75
Texture 2	570	0.01230	0.079	6.43
Texture 3	1356	0.00480	0.017	13.58

## **4- Aspects de la reconnaissance d'objets**

# Notion de «pattern»

Une forme (**pattern**) est arrangement de **descripteurs**. Le nom «**caractéristique**» (**feature**) est souvent utilisé pour désigner les descripteurs.

Une **classe de formes** est une famille de **formes** qui partagent des propriétés communes. Dans la suite, on dénotera les classes par:

$$w_1, w_2, \dots, w_m$$

où  $m$  est le nombre de classes.

Les formes sont généralement dénotées par des lettres  $\vec{x}$ ,  $\vec{y}$  ou  $\vec{z}$ .

Elles sont représentées par des vecteurs comme suit :

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

où chaque composante  $x_i$  représente le *ième* descripteur.

# Reconnaissance des formes

La **reconnaissance des formes (pattern recognition)** est une branche de **l'intelligence artificielle** qui vise à développer des techniques pour **assigner les formes aux classes** de manière automatique et avec le minimum d'intervention humaine.

À noter que les composantes de la forme (pattern)  $\vec{x}$  dépend de l'approche utilisée pour décrire la forme physique proprement dite.

## Exemple:

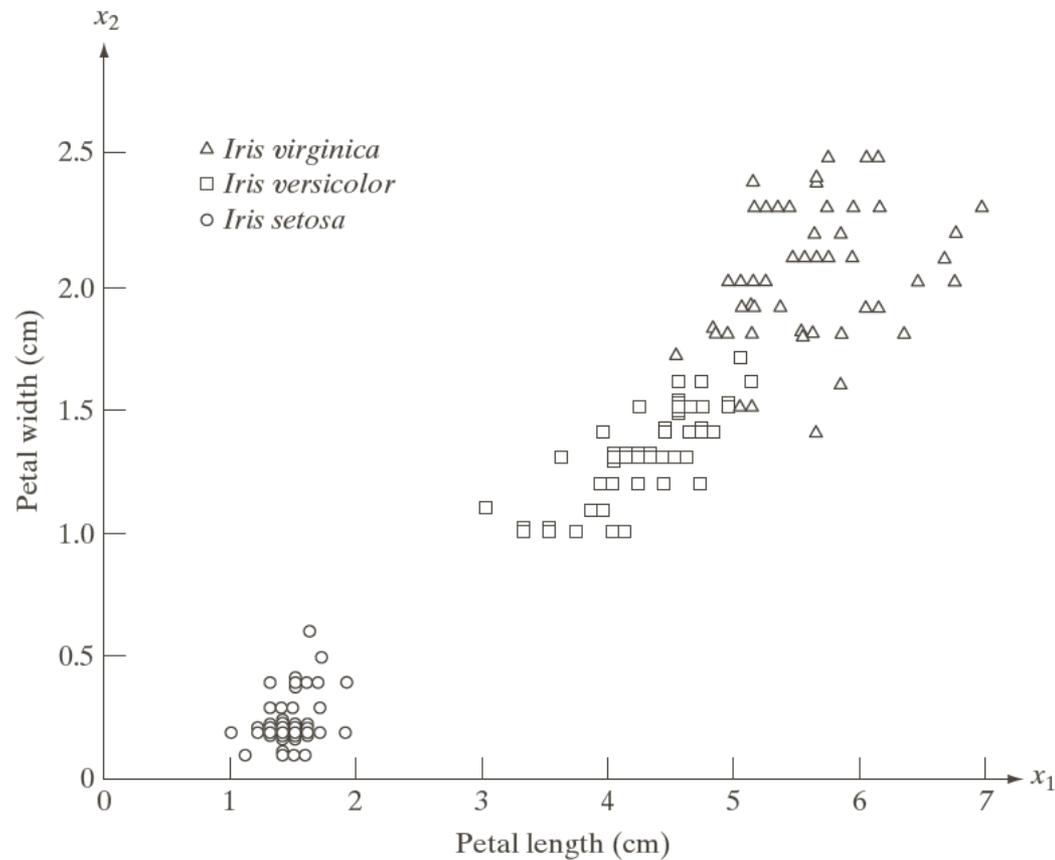
L'approche présentée a été utilisée par le statisticien Fisher pour classer trois types de fleurs Iris (*setosa*, *virginica* et *versicolor*). L'approche est appelée *l'analyse discriminante*.

# Reconnaissance des formes

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$x_1$  : La longueur des pétales.

$x_2$  : La largeur des pétales.



# Reconnaissance des formes

## D'autres applications en imagerie:

Reconnaissance des caractères.

Reconnaissance du sens de phrase (pour la traduction automatique).

Reconnaissance de visages sur les images et les vidéos.

Reconnaissance de piétons dans les séquences vidéos.

Reconnaissance d'objets sur les images et les vidéos.

Reconnaissance de mouvement, etc.

## **Références:**

1. M. S. Allili. *Eléments Avancés d'Analyse d'Images (Cours de 2e cycle)*. Université du Québec en Outaouais (UQO), Québec, Canada. Hivers 2014.
2. R. C. Gonzalez and R. E. Woods. *Digital image processing*. Pearson Education. 3<sup>rd</sup> Edition. 2008.
3. R. C. Gonzalez and R. E. Woods. *Digital image processing*. Pearson Education. 4<sup>th</sup> Edition. 2018.
4. R. C. Gonzalez, R. E. Woods, and S. L. Eddins. *Digital image processing using Matlab*. Gatesmark Publishing. 2nd Edition. 2009.