Segmentation en régions

- Partition
- Croissance de région
- Partition par quad-tree
- Montée des eaux
- Nuées dynamiques

1. Partition d'une image

Ou segmentation en régions d'une image

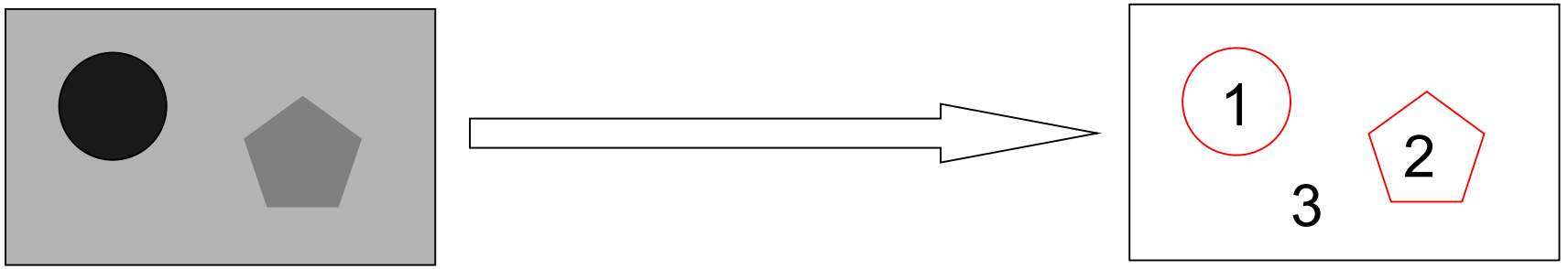


Image en niveaux de gris

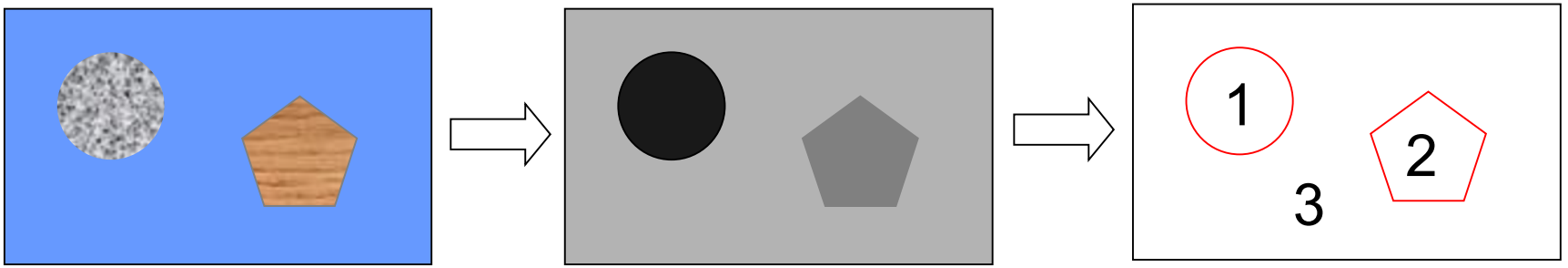
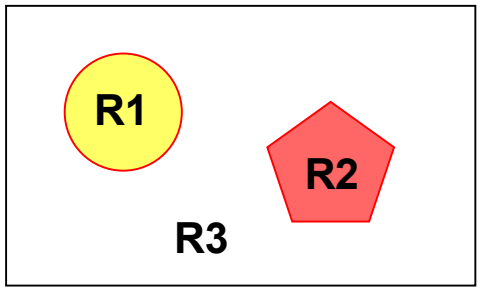


Image texturée

1.1 Définition d'une partition

- Soit I une image
- Soit P un prédicat d'homogénéité
- $S = R_1, R_2, \ldots, R_n$ est une partition de I selon P :
  - $I = \cup R_i, \ i \in [1, n]$
  - R_i est connexe, $\forall\ i \in [1, n]$
  - $P(R_i)$ est vrai, $\forall\ i \in [1, n]$
  - $P(R_i \cup R_j)$ est faux, pour tout couple (R_i , R_j) de régions connexes



1.2 Obtenir une partition

Les axiomes d'une partition définissent une segmentation unique ? ***non*** *!*

Il faut en plus :

- fonction Q caractérisant la qualité d'une segmentation
- contrainte d'optimisation de la fonction Q

S* est **la partition** recherchée sur I si :

- $I = \cup R_i$, $i \in [1, n]$
- R_i est connexe, $\forall\ i \in [1, n]$
- $P(R_i)$ est vrai, $\forall\ i \in [1, n]$
- $P(R_i \cup R_j)$ est faux, pour tout couple (R_i , R_j) de régions connexes
- **$\forall$ une partition S de I, $Q(S) \leq Q(S^*)$**

Complexité trop forte !

➔ Algorithmes de segmentation sous-optimaux

2. Croissance de région

2.1 Principe

Regrouper de manière itérative des régions

- connexes
- dont l'union respecte une propriété « d'homogénéité »

Il faut définir :

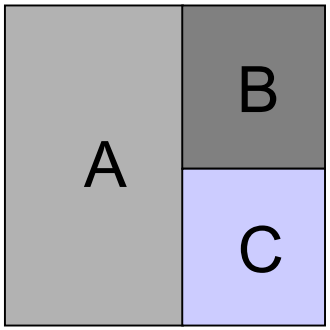
- une partition initiale de l'image
- un ensemble de critères de regroupement de 2 régions voisines
- une heuristique pour maîtriser la complexité

2.2 Stratégie de segmentation

Exemple :

- critère de fusion : *écart des luminances* $< \varepsilon$
- partition initiale : *parcelles 3x3*
- fusionner deux régions adjacentes **si** l'union respecte le critère
- arrêt du processus : aucun couple ne vérifie les conditions de fusion

Remarques : la partition obtenue dépend de l'ordre de traitement des couples



Supposons :

$max\text{-}min(A \cup B) < \varepsilon$

$max\text{-}min(A \cup C) < \varepsilon$

$max\text{-}min(A \cup B \cup C) > \varepsilon$

Quelle fusion faut-il faire en premier ?

- *un choix à l'instant t conditionne le résultat final (comme au jeu d'échec …)*
- *impossible d'évaluer toutes les partitions pour choisir la meilleure*

Stratégie pour une solution sous-optimale guidée par les données :

- définir une fonction de qualité Q pour une région (ex : *-variance(R)*)
- construire la table des couples (R_i, R_j) de régions adjacentes
- trier la table suivant $Q(R_i \cup R_j)$ décroissant
- fusionner le couple (R_k, R_l) en tête de liste → *nouvelle région* R_k
- mettre à jour la table des couples en respectant l'ordre induit par Q
- réitérer tant qu'il existe un couple (R_i , R_j) fusionnable

Amélioration :

Définir plusieurs critères de fusion :

1. max-min d'une région $< \varepsilon$
2. variance $< \varepsilon$
3. gradient aux frontières $< \varepsilon$

Hiérarchiser les critères dans le processus de segmentation :

- Croissance de région avec le critère **1** jusqu'à stabilité
- Puis, croissance de région avec le critère **2** jusqu'à stabilité
- Puis, Croissance de région avec le critère **3** jusqu'à stabilité

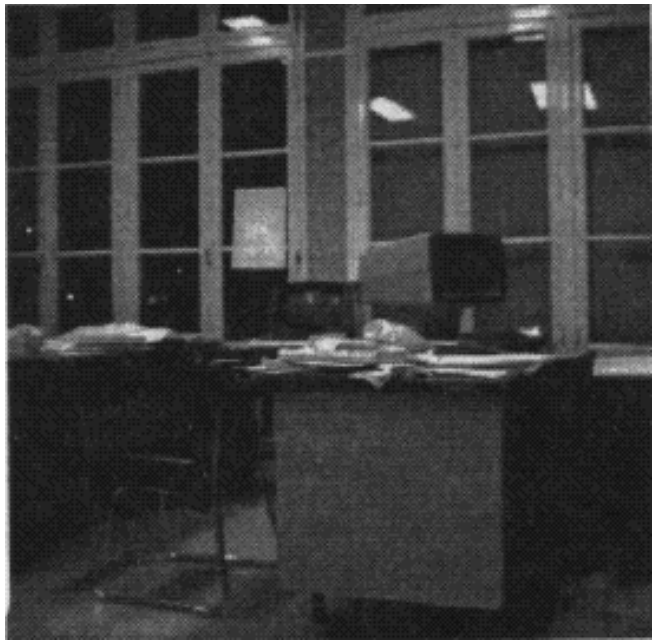
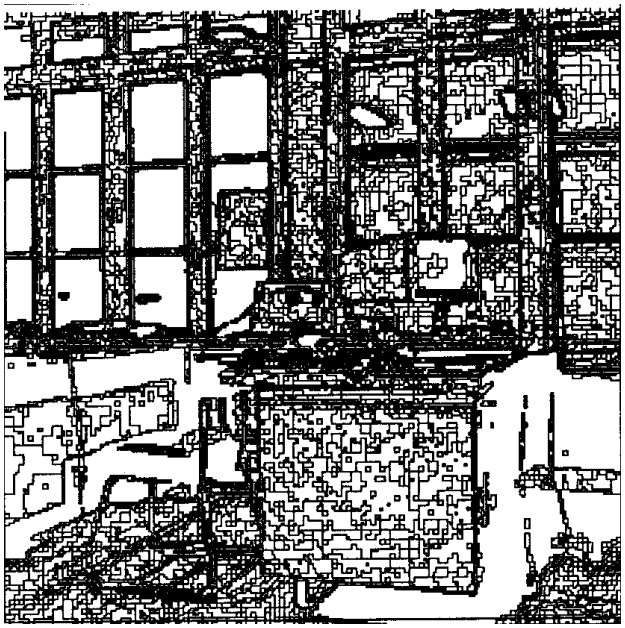
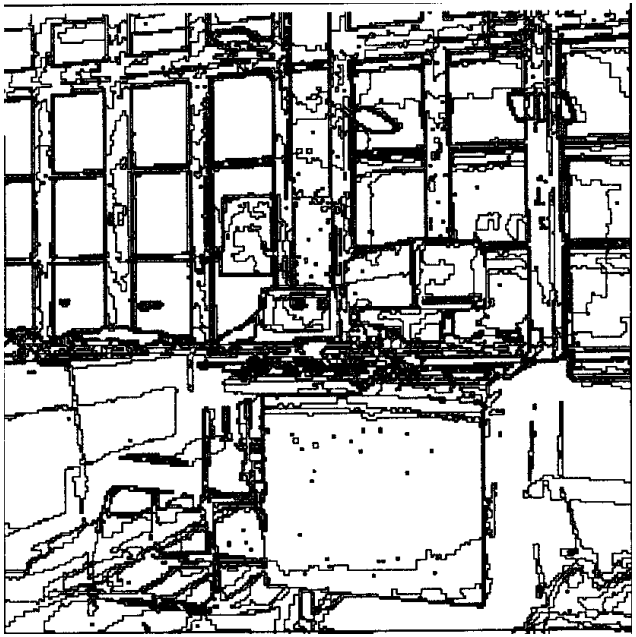


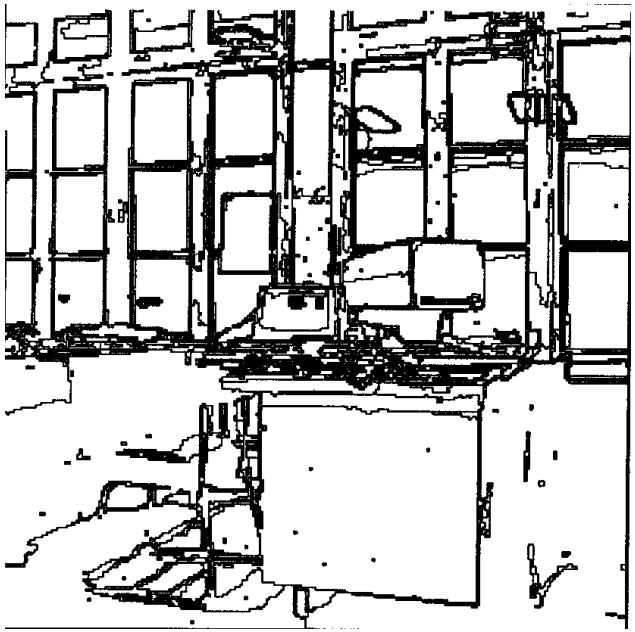
Image initiale



Critère max-min



Critère variance



Critère gradient à la frontière

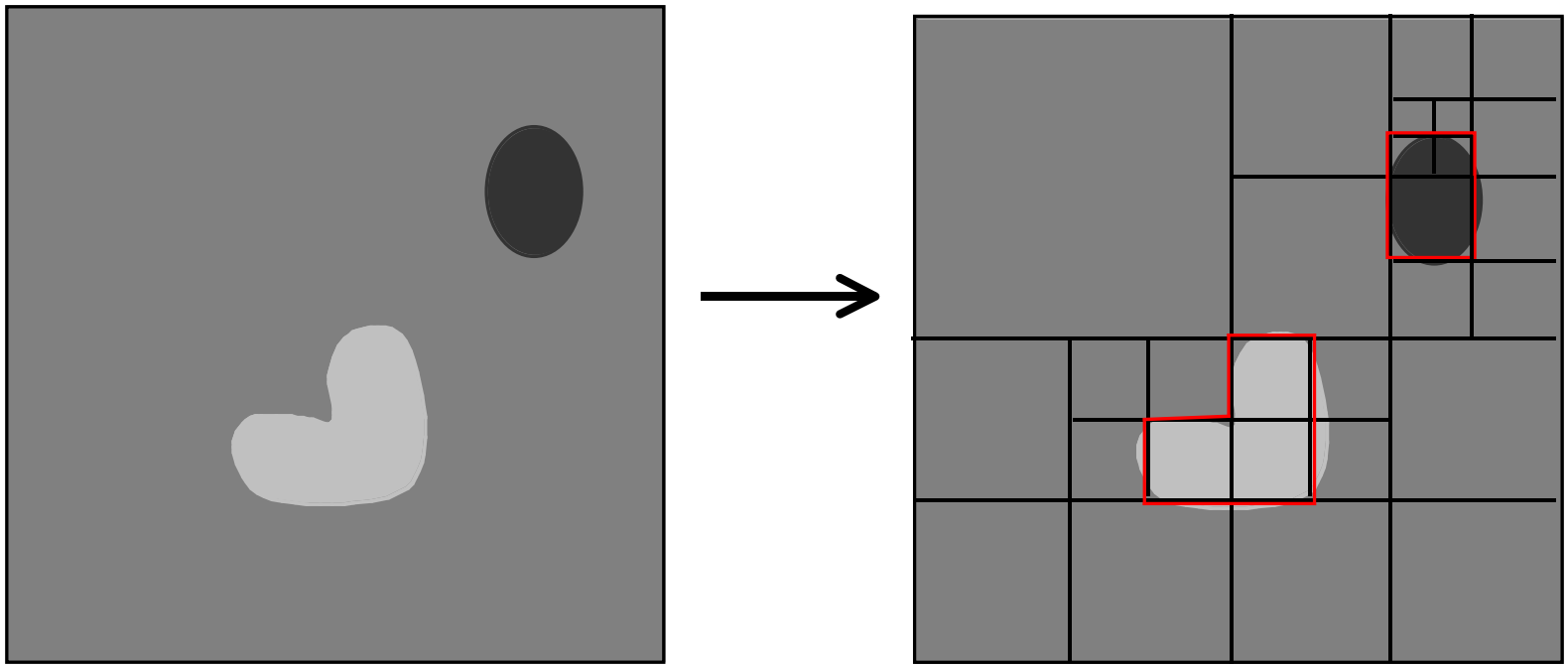
3. Partition par quad-tree

3.1 Principe

Spleet and merge ou *éclatement-regroupement*

Deux phases :

- **Eclater** l'image de manière récursive en **parcelles** (rectangles) « homogènes »
- **Regrouper** les **parcelles** voisines en **régions** « homogènes »



3.2 Eclatement

- définir une taille minimum de parcelle
- définir un critère d'homogénéité (ex : $max\text{-}min < \varepsilon$)
- découpage récursif en parcelles (rectangles)

```
Action eclater(parcelle)
   si surface(parcelle) < seuil alors
       mémoriser(parcelle)
   sinon si homogène(parcelle) alors
       mémoriser(parcelle)
   sinon
       découper en 4 sous-parcelles
       eclater chaque sous-parcelle
   fin si
fin action
```

Soient (x1, y1) et (x2, y2) les coins inf-gauche et sup-droit d'une parcelle

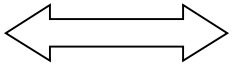
```
action eclater(x1, y1, x2, y2)
   si (x2-x1)*(y2-y1)< surface minimum  alors
       creerParcelle(x1,y1,x2,y2) ;
   sinon si ecartMinMax(x1, y1, x2, y2)< seuil alors
       creerParcelle(x1,y1,x2,y2) ;
   sinon
       x3 = (x1+x2)/2 ;
       y3 = (y1+y2)/2 ;
       eclater(x1,y1,x3,y3) ;
       eclater(x3,y1,x2,y3) ;
       eclater(x1,y3,x3,y2) ;
       eclater(x3,y3,x2,y2) ;
fin action
```

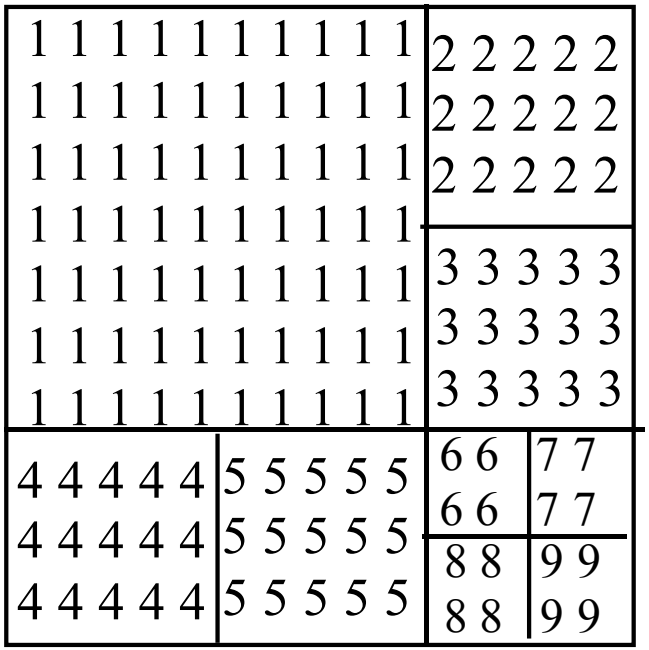
Représentation d'une parcelle :

- table des parcelles :
  - un numéro de parcelle
  - un numéro de région (initialisé par le numéro de parcelle)
- liens entre l'image et une parcelle :
  - coordonnées de 2 coins opposés d'une parcelle
  - chaque pixel est étiqueté avec le numéro de la parcelle le contenant

	Rég.	*position*	*infos*	*suiv*
1	1			
2	2			
3	3			
4	4			
5	5			
6				
7				

Table des parcelles





Carte des parcelles

3.2 Regroupement

Région : Liste de parcelles connexes et homogènes

➔ construire la table des couples de parcelles voisines pour gérer les fusions

```
pour toutes les parcelles i
   parcourir la périphérie de la parcelle i
   si la parcelle voisine j est nouvelle alors
       si i<j alors
             empiler (i,j)
       fin si
   fin si
fin pour
```

1	1	3	3	2
5	6	6	6	7
5	6	6	6	7
5	8	8	8	9

Carte des parcelles

7	9
6	7
6	8
6	9
5	6

Table des voisins

```
action fusion

  création des régions initiales (n°région = n°parcelle)
  création de la table des régions voisines
  pour tous les couples (i, j) de régions voisines :
      si homogénéité(i ∪ j) < ε alors
            concaténer les deux listes de parcelles
            choisir i comme représentant
            anciennes parcelles de j <- numéro de i
            mettre à jour la table des voisins de j
      fin si
  fin pour
```

Stratégie pour guider la fusion :

Privilégier la fusion des régions donnant la meilleur homogénéité.

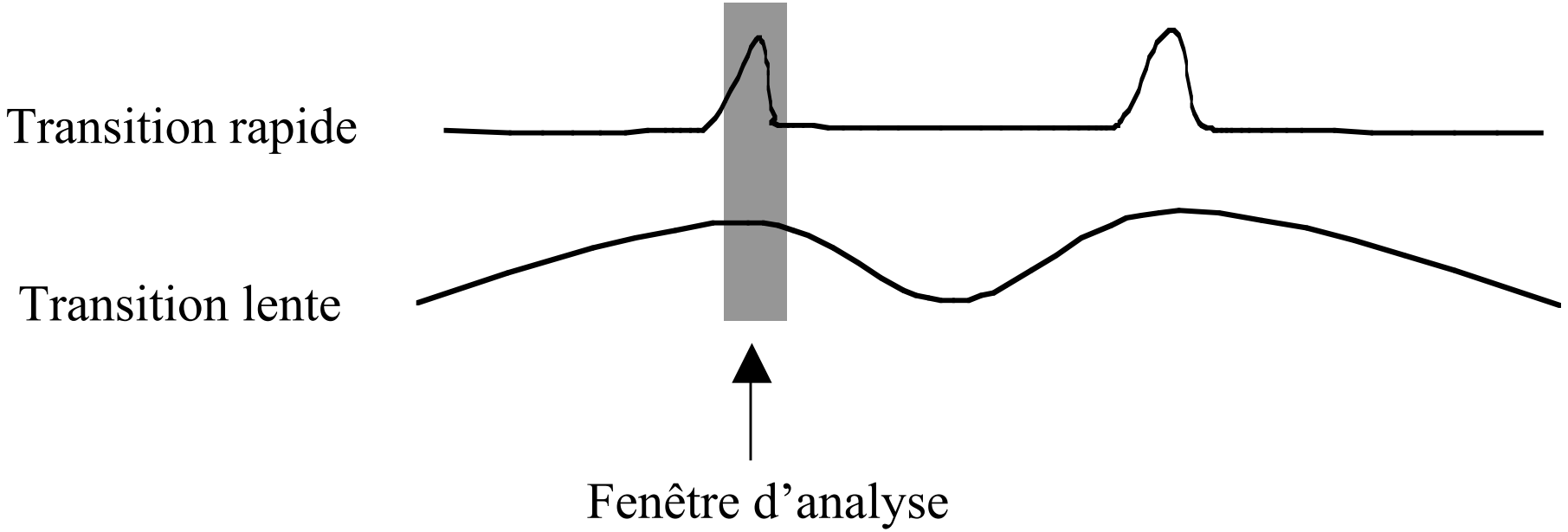
- créer une liste des régions voisines
- ordonner cette liste suivant l'homogénéité décroissante des couples
- tant que homogène(premier couple)
  - dépiler le premier couple (r1, r2) de régions
  - fusionner les deux régions *(par exemple : r1 « absorbe » r2)*
  - mettre à jour la liste des régions voisines :
    - mise à jour des voisins de la région absorbée ((r2, r3) devient (r1, r3))
    - repositionner les couples modifiés (car l'homogénéité s'est dégradée)

4. Processus de « montée des eaux »

4.1 Remarque

Les frontières entre régions peu contrastées

- sont difficiles à détecter
- sont « floues » (position peu précise)



4.2 La « ligne de partage des eaux »

Méthode efficace pour délimiter des régions peu contrastées.

Principe :

- image en niveau de gris : un relief
- sélection des minima locaux dans l'image : « cuvettes »
- montée progressive de l'eau à partir des cuvettes
- Au contact de deux régions, élévation de « digues » pour éviter que « les eaux ne se mélangent ».

➔ Les digues : ligne de partage des eaux !

a) Sélection des minima locaux

Difficulté principale : limitation du nombre de minima locaux.

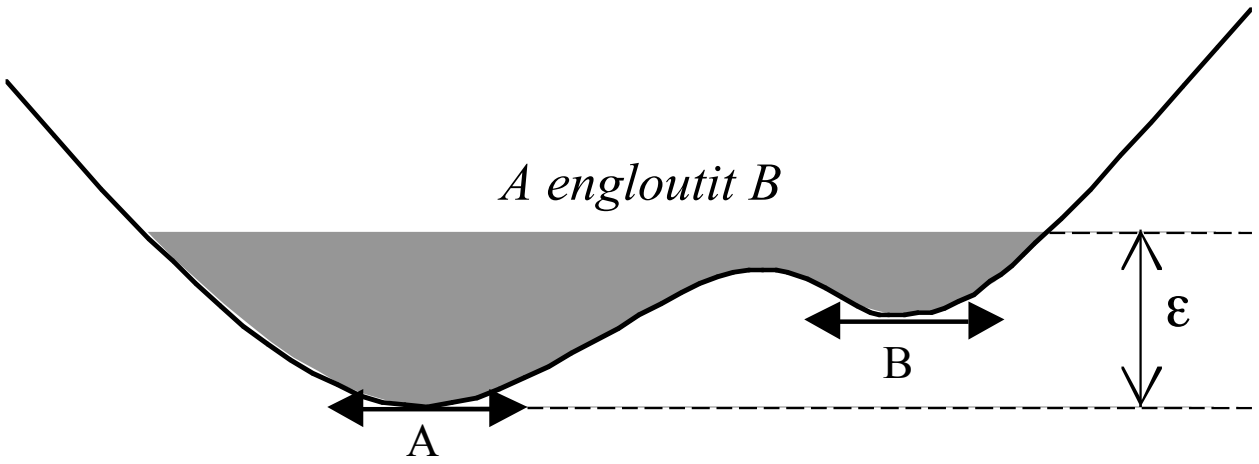
Une méthode de sélection :

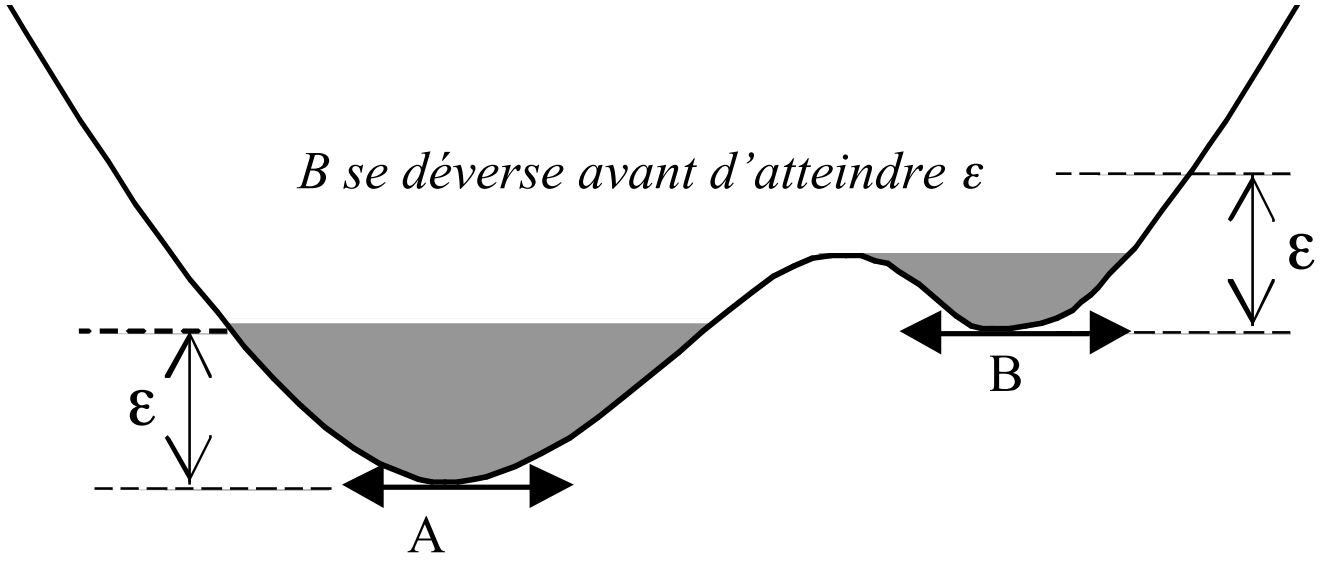
- lisser fortement l'image
- rechercher des minima locaux (< aux 8 voisins)
- éliminer des minima locaux dont la profondeur relative est < ε

Pour ce dernier point,

- trier les minima dans l'ordre croissant des altitudes,
- pour chaque minimum dans cet ordre :
  - croissance de région jusqu'à atteindre (min + ε)
  - neutraliser les éventuels minima voisins qui ont été submergés (< min + ε)
  - abandonner ce minimum si rencontre d'un point plus bas

Minima non significatifs :





b) Montée des eaux

On construira ainsi une carte des cuvettes initiales (étiquettes spécifiques) qui servira de point de départ pour le processus de montée des eaux.

Deux approches similaires :

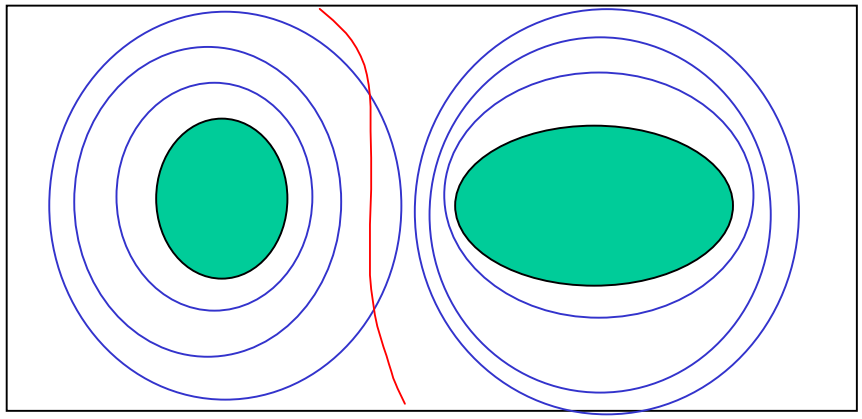
➔ processus progressif de **dilatation contrôlée de régions** (sous condition)

- altitude < ligne de niveau fixée
- jusqu'à « contact » avec une autre région

➔ **amincissement homotopique conditionnel**

- du complémentaire des cuvettes initiales
- sous condition (altitude < ligne de niveau)
- le squelette constitue la ligne de partage des eaux.

La croissance des cuvettes est contrôlée par des lignes de niveaux



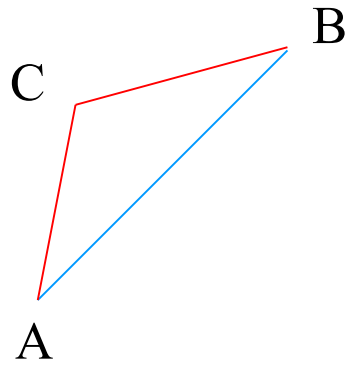
```
dH : Gain d'altitude entre deux lignes de niveau
carte1 et carte2 initialisées à 0
marquer les cuvettes dans carte2
H <-  minimum de l'image
tant que H < maximum de l'image
  H <- H+dH ;
  répéter
      carte1 <- carte2 ;
      croissanceLimitée(carte1, H, carte2) ;
  jusqu'à stabilité (carte1 = carte2)
fin tant que
```

Processus de « dilatation » soumis à
- une limitation d'altitude
- détection de contact entre 2 régions (les digues…)

```
action croissanceLimitée(carte1, H, carte2)
  pour tous les pixels P
    si carte1(P)≠0
      carte2(P)= carte1(P)
    sinon
      si hauteur(P)<H
        si P est voisin d'une région i dans carte1
            carte2(P) <- i;
        sinon
            carte2(P) <- 0;
        fin si
      fin si
    fin si
  fin pour
fin action
```

5. Nuées dynamiques

5.1 Préliminaires

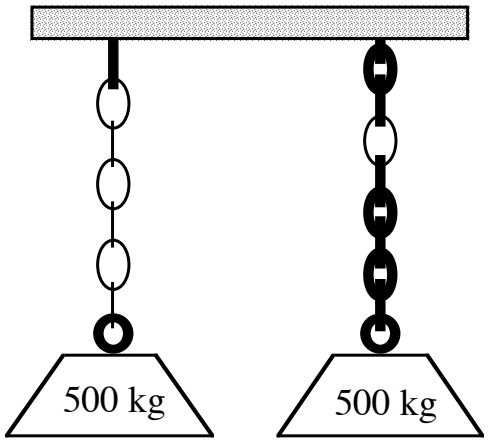


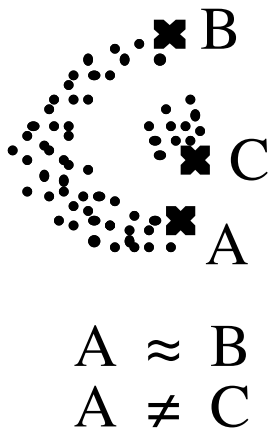
Axiomes définissant une distance : Soit d : ExE -> R^+

- d(x, x) = 0
- d(x, y) = d(y,x)
- d(x, y) ≤ d(x, z) + d(z, y)

Axiomes définissant une ultra-métrique : Soit d : ExE -> R^+

- d(x, x) = 0
- d(x, y) = d(y,x)
- d(x, y) ≤ max(d(x, z), d(z, y))





5.2 Partition adaptative (k-means)

Soit d une métrique ou une ultra-métrique sur un ensemble fini E,
soit $P = \{C_1 , C_2 , ..., C_k \}$ une partition de E en k classes
soient k éléments de E choisis respectivement dans chaque classe :

$a_1 , a_2 , ..., a_k$. a_i est dit "descripteur" de C_i

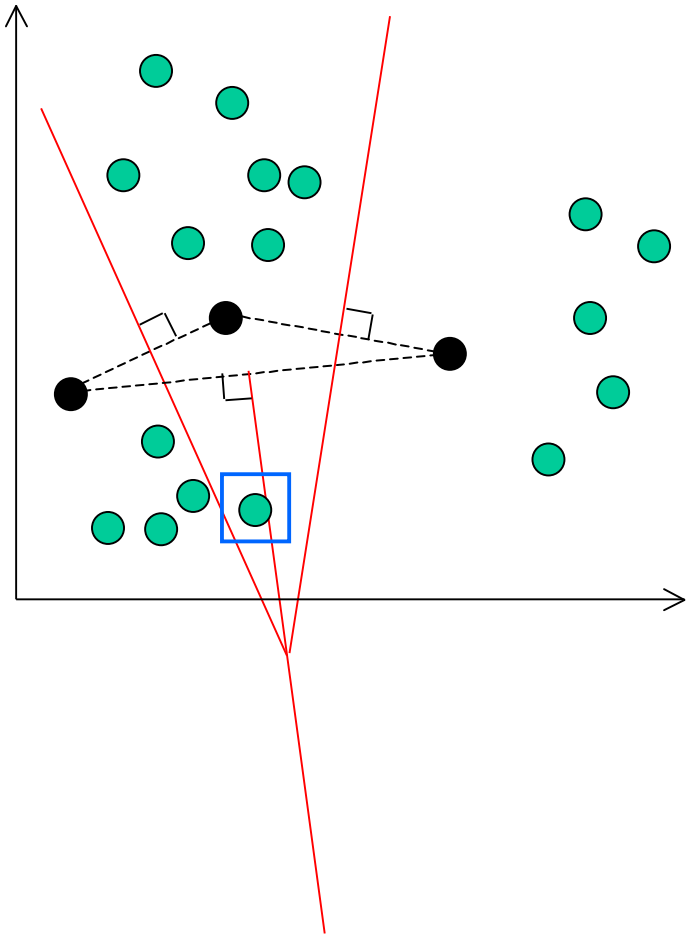
Algorithme α

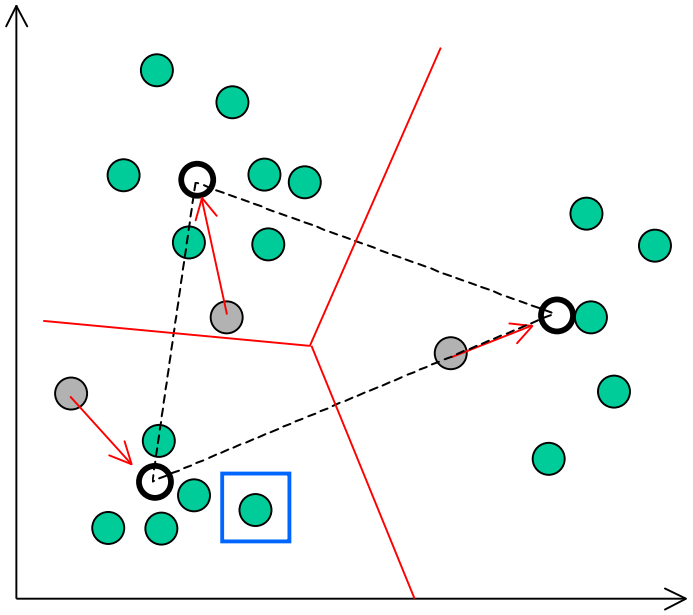
- initialisation : choisir (aléatoirement par ex) k a_i dans E
- groupement : associer chaque élément de E au centroïde le plus proche

  $\forall x \in E, x \in C_i$ si $\forall j \neq i$, $d(a_i, x) < d(a_j, x)$
- adaptation : évaluation d'un nouveau centroïde a_i optimum pour chaque C_i

  (barycentre de C_i par ex)
- retourner au groupement si un a_i a changé.

Partition en 3 classes





Algorithme β

- soit e un nouvel élément de E
- affecter e à une classe C_j :

  $e \in C_j$ si $\forall\, i \neq j,\ d(a_j, e) < d(a_i, e)$
- adaptation : évaluation d'un a_i optimum pour chaque C_i
- groupement : réévaluer les classes C_i
- arrêter lorsque tous les éléments de E sont affectés à une classe .

Remarques sur l'algorithme β :

- évolution des descripteurs après chaque affectation d'un élément à E
- adapté à une arrivée séquentielle de nouvelles données en grande quantité

Inconvénients :

- obligation de fixer à priori le nombre de classes
- dépendance au choix des centres initiaux
- une distance ne détecte bien que les formes convexes (surtout sphériques de même taille).

Pour contourner ce dernier point

- utiliser une ultra-métrique
- changer d'espace de représentation

Nuées dynamiques : Généralisation de l'algorithme α

- les descripteurs a_i sont des n-uplets (plusieurs centroïdes pour une classe),
- on utilise un indice de dissemblance $D(x, a_i)$ pour effectuer le regroupement
- on utilise un indice de ressemblance $R(a_i, C_i)$ pour l'adaptation

Image segmentée en trois classes par k-means

Image segmentée en six classes par k-means