Introduction à l'imagerie numérique

3I022-fev2018

Segmentation des images

Licence d'informatique

Février 2018

Définition de la segmentation

- Subdiviser l'image en structures :
  - des objets,
  - des régions.
- Objets : une forme particulière, par exemple :
  - des points,
  - des segments de droites,
  - des contours,
  - des formes plus complexes.

Avertissement

La plupart des exemples donnés dans ce cours sont reproductibles à partir des images disponibles dans le dépôt de l'UE et de la séquence de commandes Shell/Inrimage indiquée ou du matériel donné et développé en TME.

Définition de la segmentation

- Segmentation : partition de l'image en composantes connexes. Une composante connexe = une région.
- Régions : pas d'hypothèse particulière sur la forme mais des propriétés sur la distribution des niveaux de gris supposés vérifiées en tout point de la région, par exemple :
  - homogénéité en niveau de gris,
  - texture particulière.
- des régions qui vérifient à la fois des critères de forme et de texture.
- le Graal du traitement de l'image : un sujet difficile, une méthode (imparfaite) traite un cas particulier.

Détection de régions

Plan du cours

1. Notion d'homogénéité et analyse d'histogramme
2. Seuillage global
3. Choix optimal du seuil
4. Autre seuil : local, adaptatif
5. Algorithme de Découpage/Fusion
6. Croissance de régions
7. Ligne de partage des eaux

Avertissement

La plupart des exemples donnés dans ce cours sont reproductibles à partir des images disponibles dans le dépôt de l'UE et de la séquence de commandes Shell/Inrimage indiquée ou du matériel donné et développé en TME.

Détection de régions par seuillage

Caractériser l'homogénéité des régions

- Lorsque les régions sont très homogènes dans leur distribution des niveaux de gris, on peut se contenter d'un seuillage pour segmenter une image :

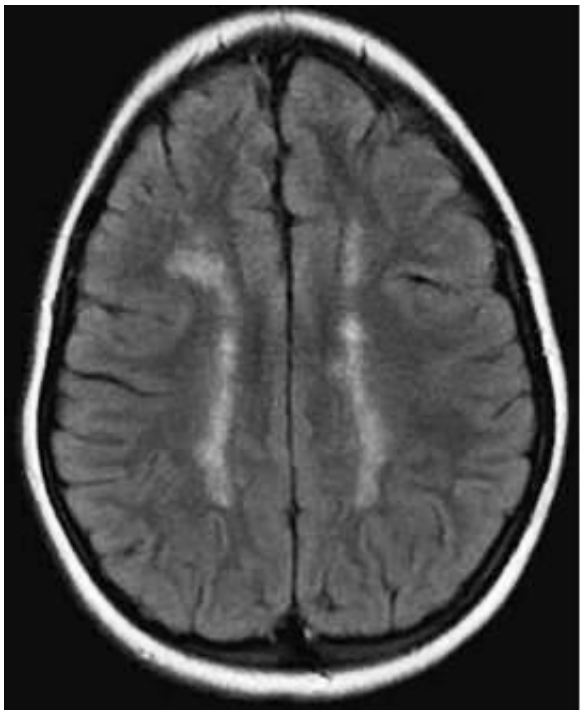


FIGURE – Une image IRM

- On visite les valeurs des niveaux de gris pour chaque classe (avec `xvis` maintenir le bouton gauche enfoncé : la valeur est affichée en haut de la fenêtre).
- os : $[180, 255]$
- matière blanche : $[120, 180[$
- matière grise : $[60, 120[$
- liquide céphalo-rachidien : $[20, 60[$

Seuillage

Homogénéité

```
vb -n `carflo 180` 0 IRM.png | xvis
```

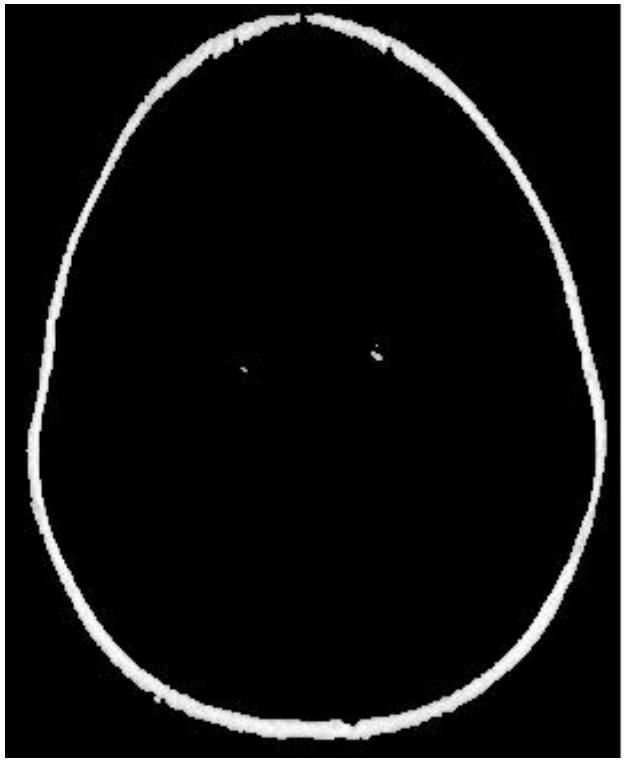


FIGURE – Os

Rappel : manuel de `vb`, voir `man arit1`.

Seuillage

Homogénéité

```
vb -n `carflo 120` 0 IRM.png | vh -n `carflo 180` 0 | xvis
```

FIGURE – Matière grise

Seuillage

Homogénéité

```
vb -n `carflo 60` 0 IRM.png | vh -n `carflo 120` | xvis
```

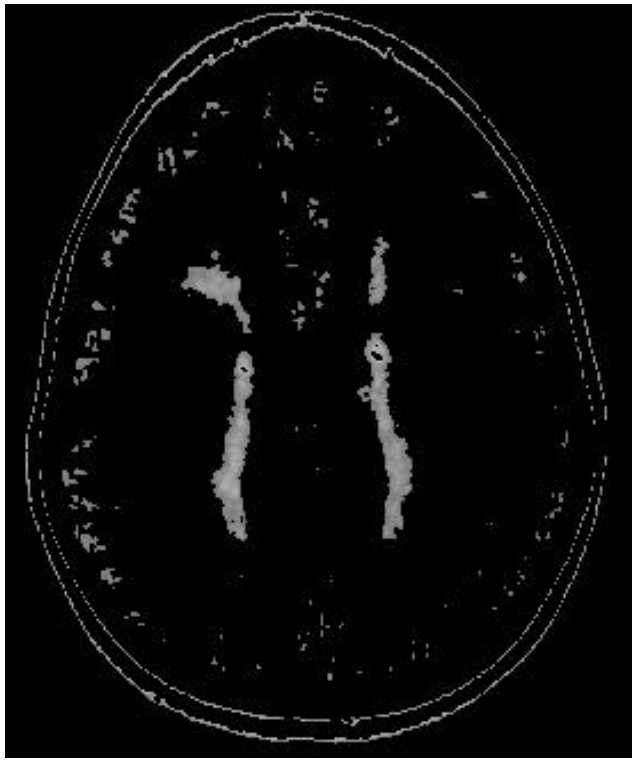


FIGURE – Matière grise

Seuillage

Homogénéité

```
vb -n `carflo 20` 0 IRM.png | vh -n `carflo 60` 0 | xvis
```

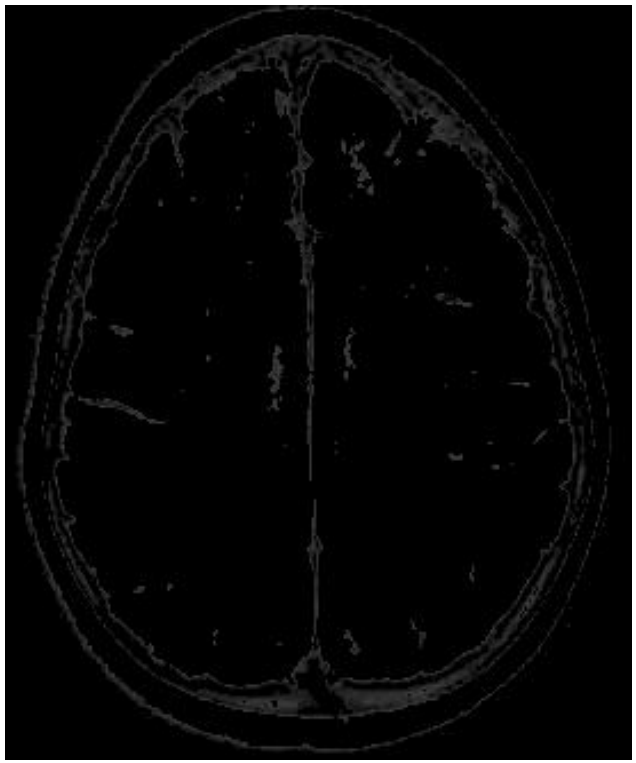
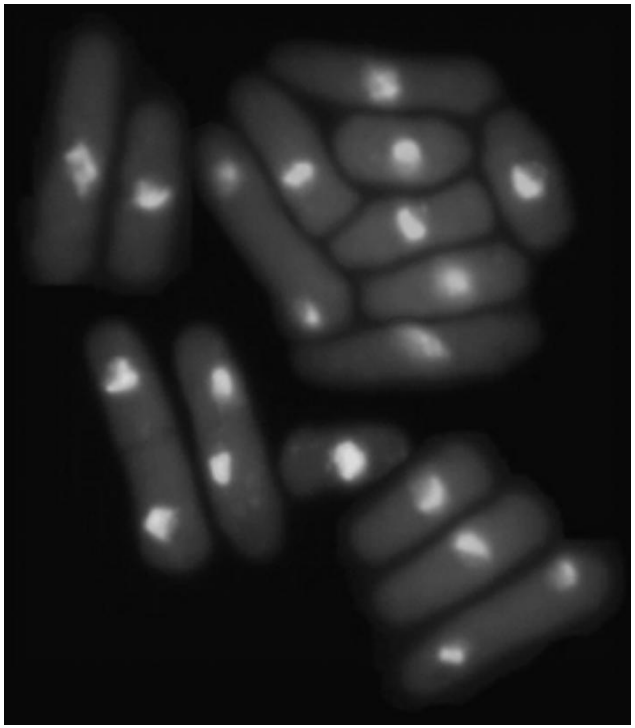


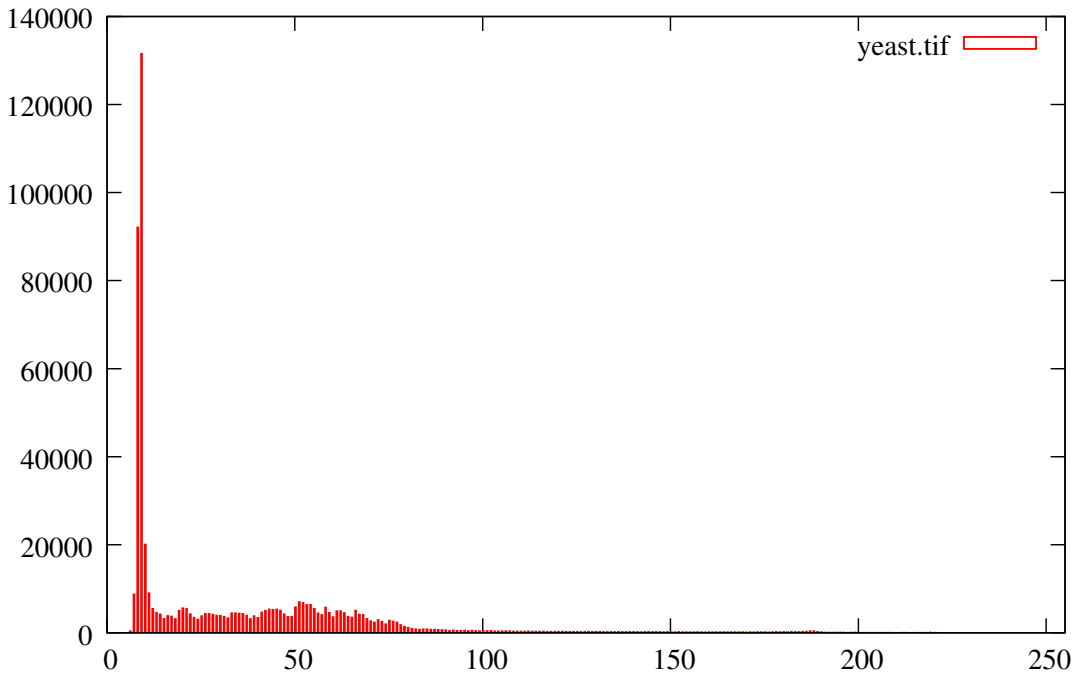
FIGURE – CFR

Seuillage

Homogénéité

- Comment choisir le(s) seuil(s) ?
- Un élément de réponse : comment se caractérise la distribution des niveaux de gris (c'est-à-dire l'histogramme) dans une région dite "homogène" ?

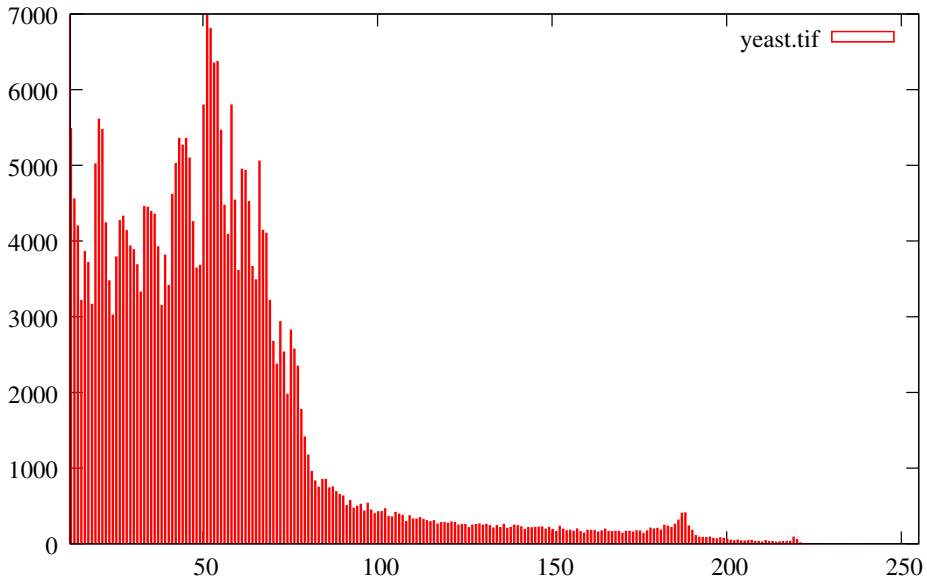




Seuillage

Homogénéité

- L'étude de l'histogramme permet de trouver facilement le seuil qui distingue le fond des cellules : ~ 12.
- Réaffichons l'histogramme en ne prenant pas en compte les valeurs en dessous de 12 :

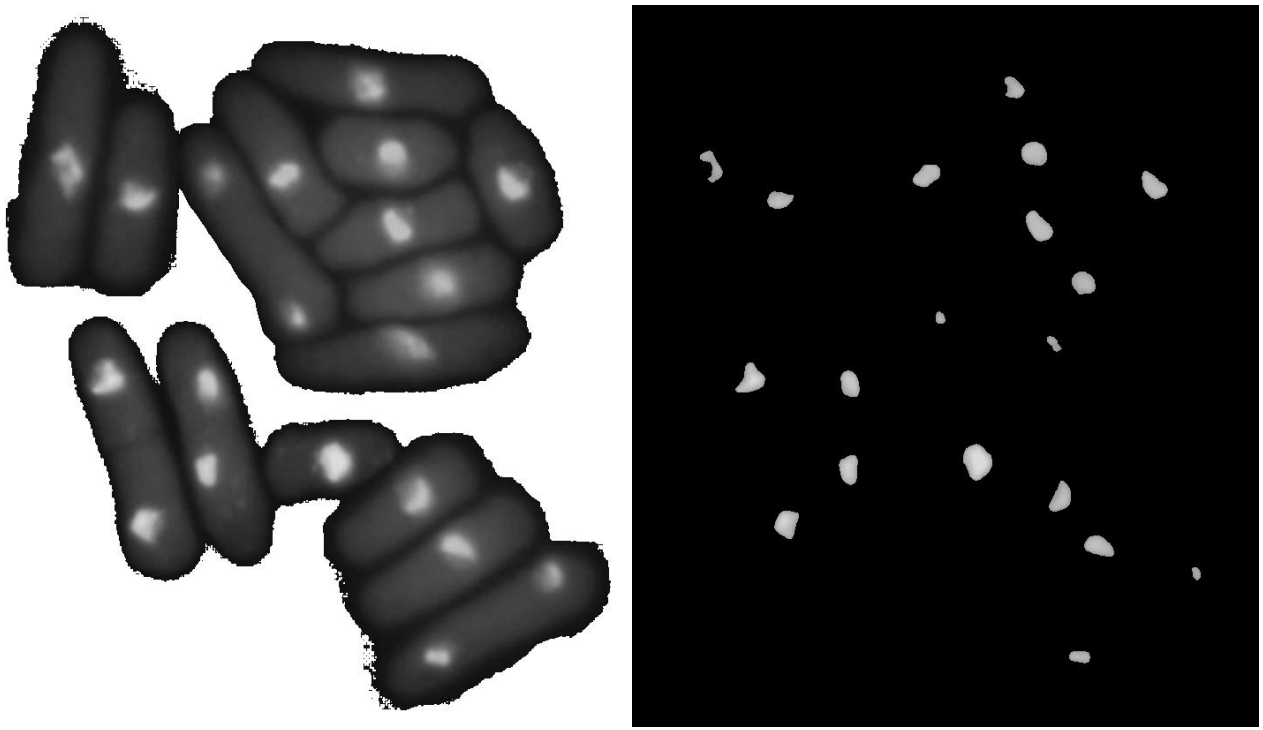


- grande variance les ndg des cellules 12-90, les niveaux de gris au delà de 150 correspondent aux noyaux.

Seuillage

Homogénéité

```
vb -n `carflo 15` 1 yeast.inr | xvis -n
vb -n `carflo 150` 0 yeast.inr | xvis -n
```

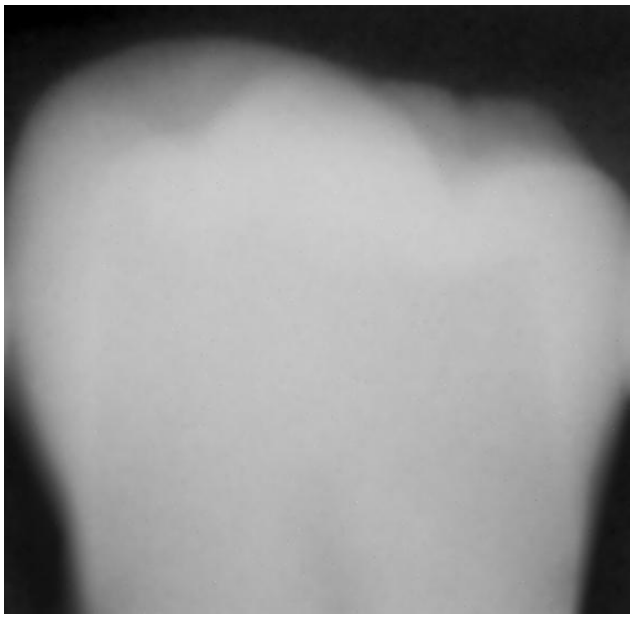


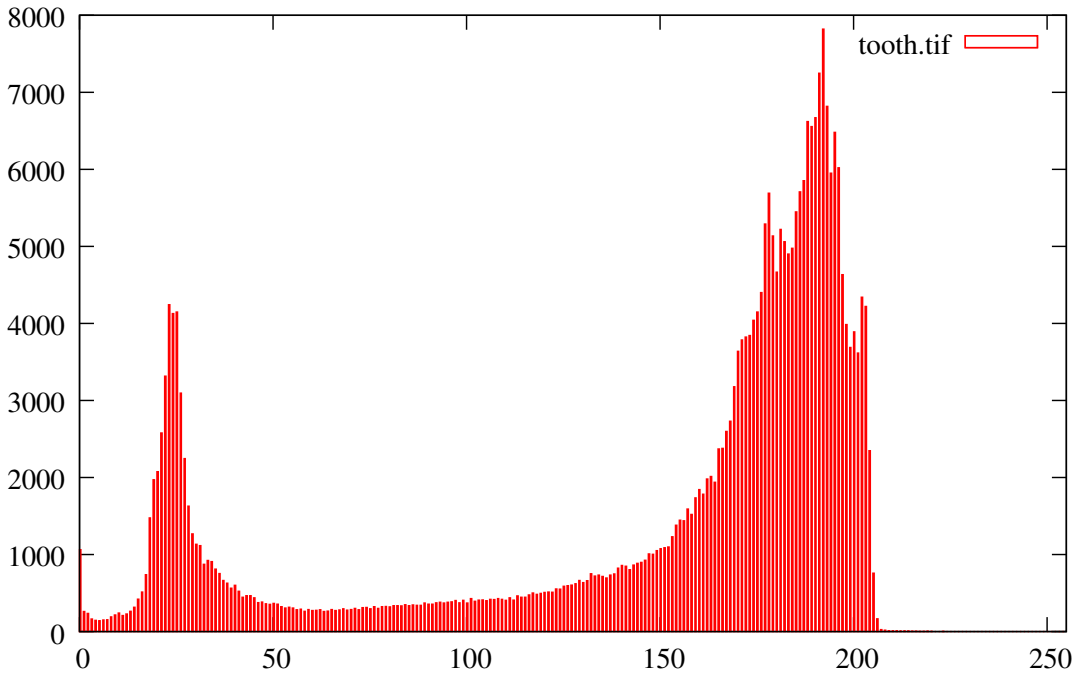
(a) Seuillage à 15 (b) Seuillage à 150

Seuillage

Homogénéité

- Exemple de région homogène ayant une large variance.
- Le choix du seuil devient difficile (ou plutôt subjectif).





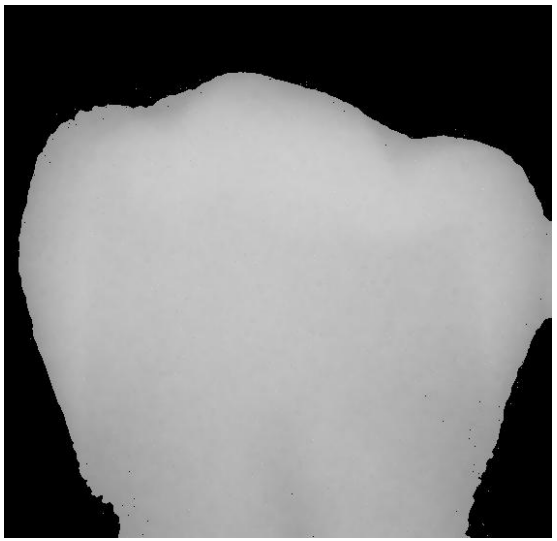
Seuillage

Homogénéité

```
vb -n `carflo 50` 0 tooth.inr | xvis -n
vb -n `carflo 100` 0 tooth.inr | xvis -n
vb -n `carflo 150` 0 tooth.inr | xvis -n
```

(c) seuillage à 50

(d) seuillage à 100



(e) seuillage à 150

Seuillage

Homogénéité

- Donc une région homogène se traduit par un pic dans son histogramme plus ou moins large (c'est-à-dire de variance plus ou moins grande).
- Les distributions de type "gaussienne" représentent assez bien ce type d'histogramme.

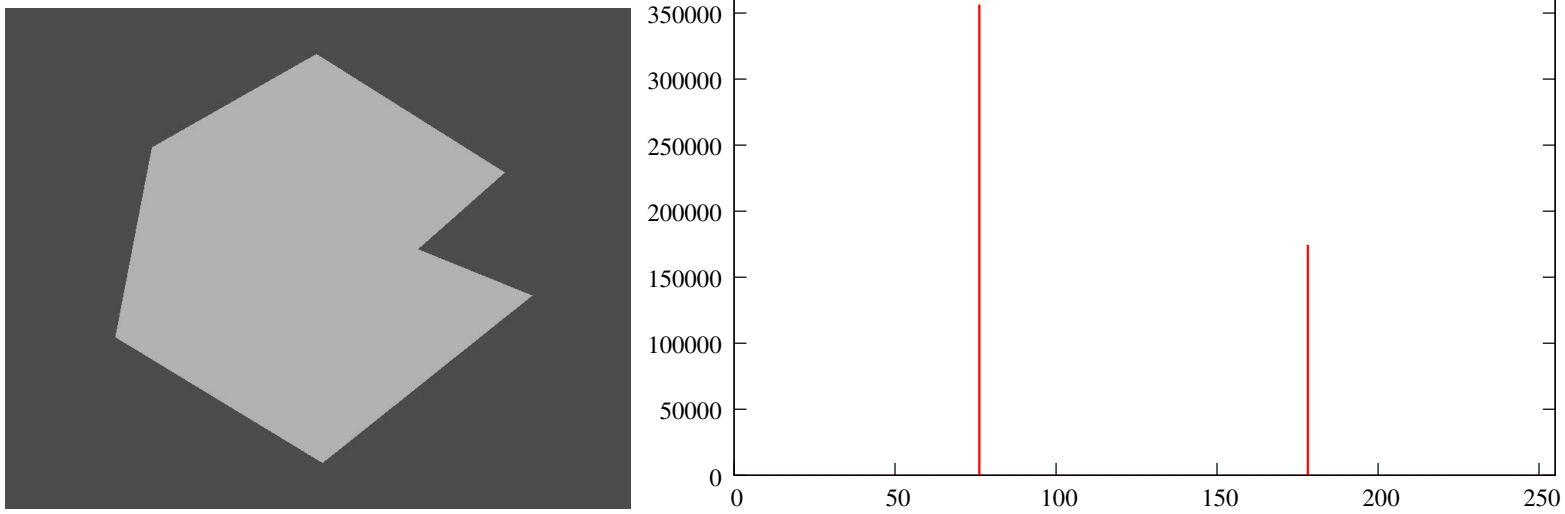


FIGURE – Régions idéalement homogènes : variance nulle

Seuillage

Homogénéité

- Ajoutons un bruit gaussien ($\mu = 0, \sigma = 10$) :

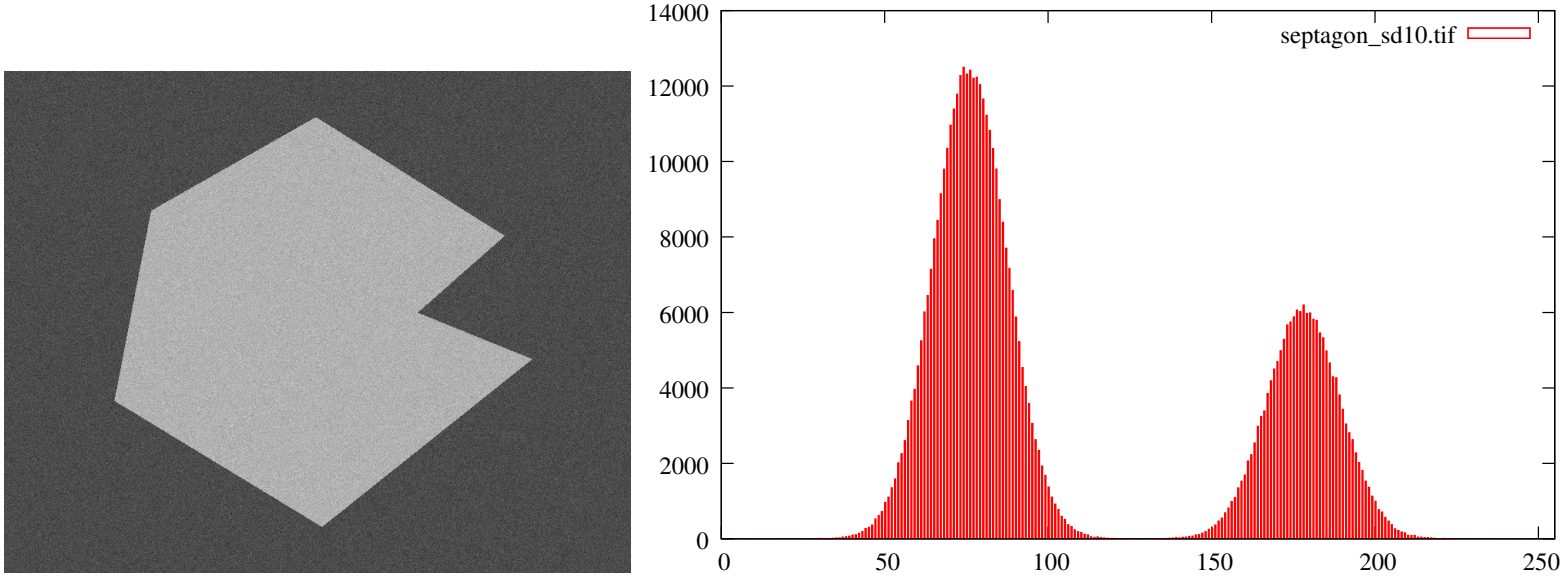


FIGURE – Régions relativement homogènes : répartition gaussienne.

Seuillage

Choix du type de seuillage et du seuil

- Le choix du seuil : on peut donc le déduire d'une analyse visuelle de l'histogramme.
- Le nombre de seuils est lui dicté par le nombre de classes que l'on veut obtenir.
- On parle bien de classes : l'ensemble des pixels ayant un niveau de gris supérieur (inférieur, similaire) à tant ; on n'a aucune assurance de connexité dans le résultat du seuillage (pas de propriétés spatiales).

Seuillage optimal

Méthode d'Otsu

- On étudie ici le cas le plus simple : un objet homogène sur un fond homogène et on cherche un seuil k partageant l'image I en deux classes :
  - le fond : l'ensemble $B = (n|I(n) < k)$
  - l'objet : l'ensemble $O = (n|I(n) \geq k)$
- Calculons la moyenne des pixels dont les niveaux de gris sont strictement inférieurs à un seuil k :

$$\mu_B(k) = \frac{1}{N(k)} \sum_{n=0}^{N-1} I(n) \mathbb{1}_{I(n)<k}$$

$$N(k) = \sum_{n=0}^{N-1} \mathbb{1}_{I(n)<k}$$

et N est le nombre de pixels dans I.

Seuillage optimal

Méthode d'Otsu

- Il est facile de voir que :

$$\mu_B(k) = \frac{1}{\alpha(k)} \sum_{l=0}^{k-1} lg(l)$$

g est l'histogramme normalisé et $\alpha(k)$ tel que :

$$\alpha(k) = \sum_{l=0}^{k-1} g(l) = \frac{N(k)}{N}$$

- Intérêt de la seconde formule : étant donné l'histogramme de l'image, le calcul des moyennes est beaucoup plus rapide, en particulier, indépendante de la taille de l'image.

Seuillage optimal

Méthode d'Otsu

- De la même façon, on calcule la moyenne des pixels dont les niveaux de gris sont supérieurs au seuil k :

$$\mu_O(k) = \frac{1}{N - N(k)} \sum_{n=0}^{N-1} I(n) \mathbb{1}_{I(n) \geq k}$$

$$\mu_O(k) = \frac{1}{1 - \alpha(k)} \sum_{l=k}^{L-1} lg(l)$$

Méthode d'Otsu

Variance intraclasse

Definition (Variance intraclasse)

Soit une image segmentée en *n* régions, on appelle la variance intraclasse la moyenne pondérée (par le taux des pixels appartenant à la classe concernée) des variances de chaque classe.

- Dans notre cas (deux classes) la variance intraclasse est :

$$\sigma^2_{intra} = \alpha\sigma^2_B + (1-\alpha)\sigma^2_O$$

- La variance intraclasse est d'autant plus petite que les variances de chaque classe le sont (en proportion de leur taille) et une classe de variance faible indique une classe homogène en distribution de n.d.g.

Méthode d'Otsu

Variance intraclasse

- Otsu (1979) propose donc pour valeur de seuil k celle qui minimise la variance intraclasse. En effet :
  - Minimiser la variance intraclasse revient à partager l'image en deux régions, chacune ayant une variance minimale (en proportion de leur taille).
  - Partager l'image en deux régions de variance minimale signifie donc de partager l'image en deux régions les plus homogènes possible.
- La recherche du minimum est assez coûteuse (surtout si on a un grand nombre de niveaux de gris). On peut essayer de simplifier la formule.

Méthode d'Otsu

Variance interclasse

Definition (variance interclasse)

On appelle variance interclasse la quantité :

$$\sigma^2_{inter} = \sigma^2 - \sigma^2_{intra}$$

- Donc minimiser la variance intraclasse est équivalent à maximiser la variance interclasse.
- On peut montrer que la variance interclasse est égale à :

$$\begin{aligned} \sigma_{inter}(k) &= \alpha(k)(\mu - \mu_B(k))^2 + (1 - \alpha(k))(\mu - \mu_O(k))^2 \\ &= \alpha(k)(1 - \alpha(k))(\mu - \mu_B(k))^2 \end{aligned}$$

Méthode d'Otsu

Variance intra/interclasse

- Évolution des variances de chaque classe, de la variance interclasse en fonction du seuil.

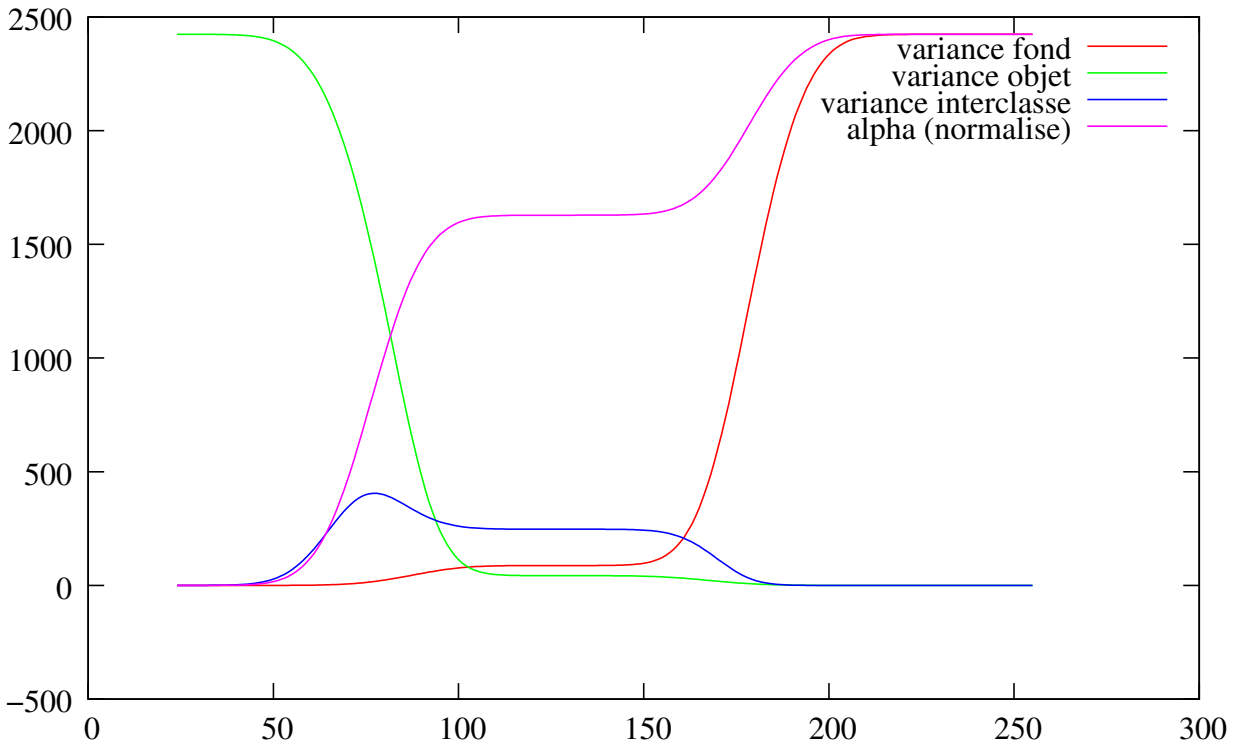


FIGURE – Image du septagon bruité

Méthode d'Otsu

Variance intra/interclasse

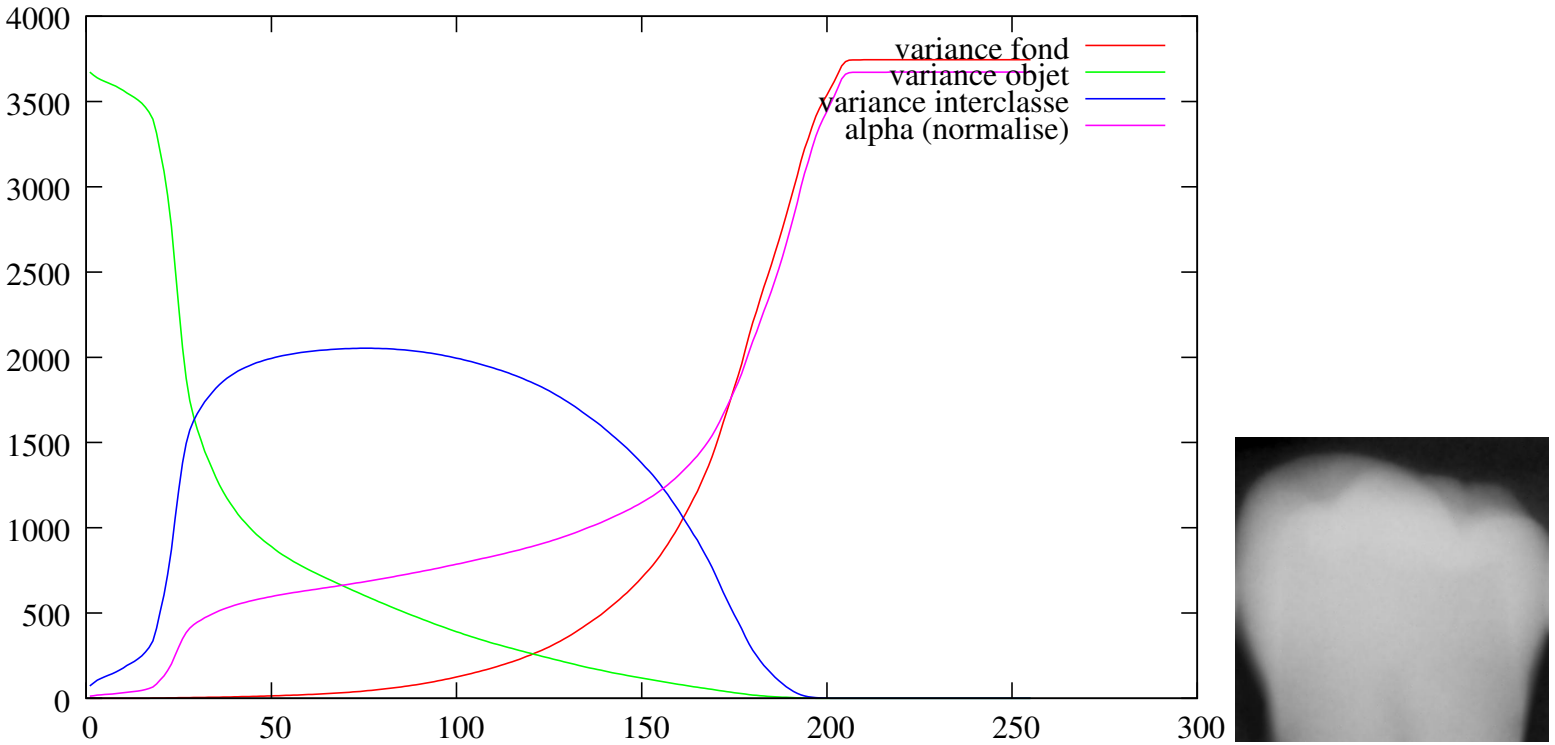


FIGURE – Image `tooth.inr`

Méthode d'Ostu

Algorithme

```
algo Ostu(g:histogramme normalise): seuil s
  max := 0; s := 0;
  mu := calcul_mu(g);
  pour l variant entre 0 et L-1 faire
    mu1 := calcul_muB(g,l);
    alpha := calcul_alpha(g,l);
    sigma_inter := alpha*(1-alpha)(mu-muB)^2;
    si sigma_inter > max alors
      max = sigma_inter;
      s := l;
    fin si
  fin pour
fin algo
```

- L'algorithme Otsu calcule donc les variances interclasse pour toutes les valeurs de seuils.

Méthode d'Ostu

Généralisation

- Cet algorithme est facilement généralisable à c classes correspondantes à $c-1$ seuils $C_i = \{n | k_i \leq I(n) < k_{i+1}\}$ avec $k_0 = 0$ et $k_{c+1} = L$.
- La variance intraclasse est par définition :

$$\sigma^2_{intra} = \sum_{i=0}^{c-1} \alpha_i (\mu - \mu_i)^2$$

avec :

  - μ_i la moyenne des pixels de la classe C_i
  - α_i la proportion de la classe C_i

- On notera que le coût algorithme explose avec le nombre de seuils dans la recherche du maximum de la variance intraclasse.

Méthode de Redi *et al* (1984)

- Pour éviter la recherche du maximum, on peut résoudre l'équation :

$$\frac{\partial \sigma^2_{inter}(k)}{\partial k} = 0$$

- On montre que la solution de l'équation précédente vérifie :

$$\mu_B(k) + \mu_O(k) = 2k$$

- Pas de solution explicite, on cherche une solution approximative par la méthode du point fixe :
  La suite définie par :

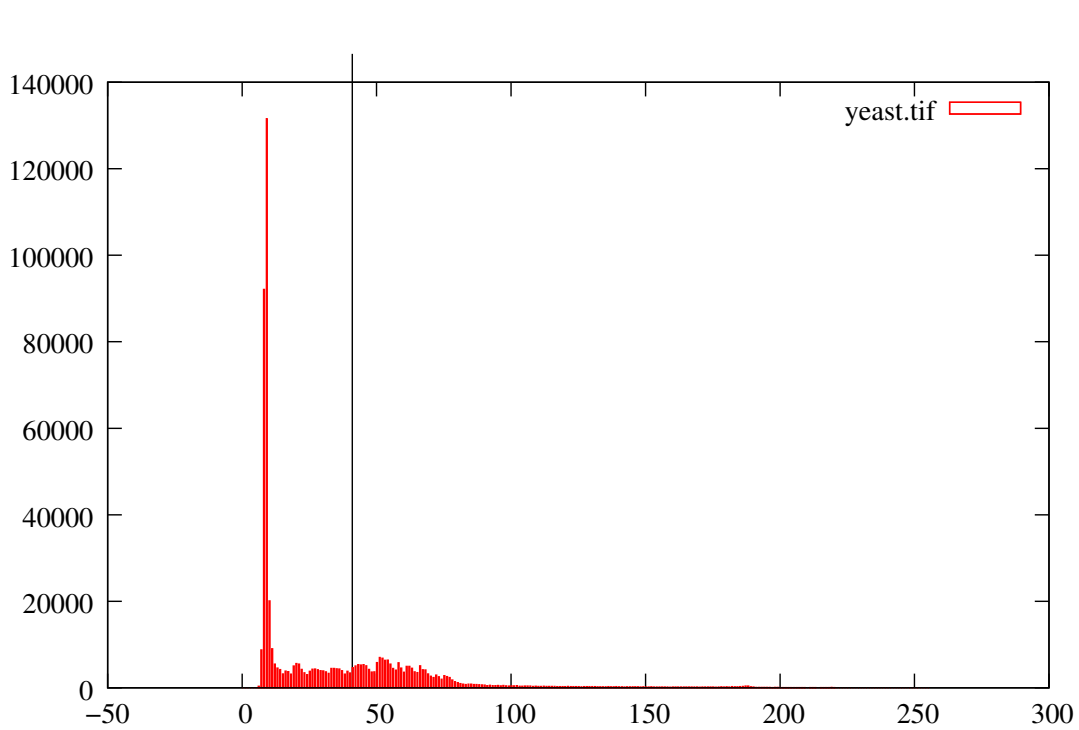
$$\begin{aligned} k_0 &= \mu \\ k_{n+1} &= \frac{1}{2}(\mu_B(k_n) + \mu_O(k_n)) \end{aligned}$$

  converge vers la solution de l'équation précédente.
- Arrêt des itérations lorsque $k_{n+1} \sim k_n$. En pratique, l'algorithme converge rapidement en une dizaine d'itérations.

Seuillage optimal

Exemples

- Algorithme `otsu` appliqué aux images précédemment vues.



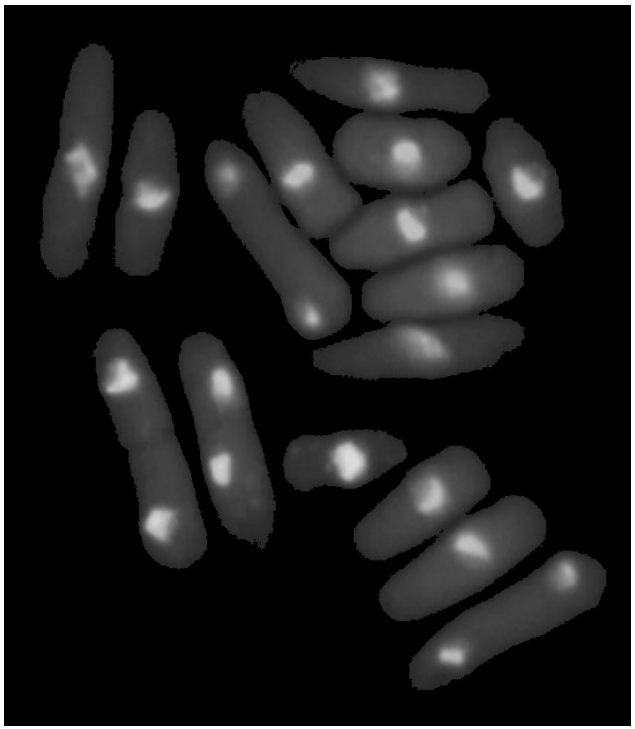


FIGURE – Seuil à 41

Seuillage optimal

Exemples

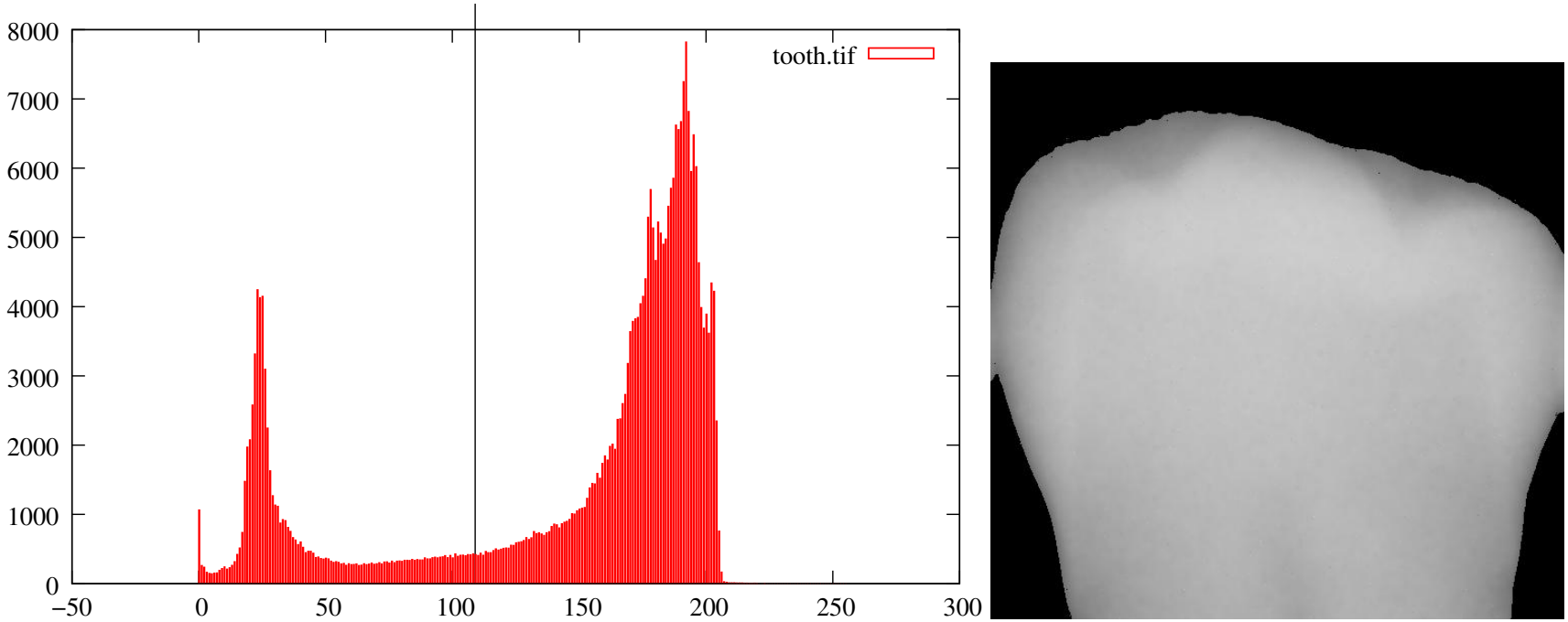
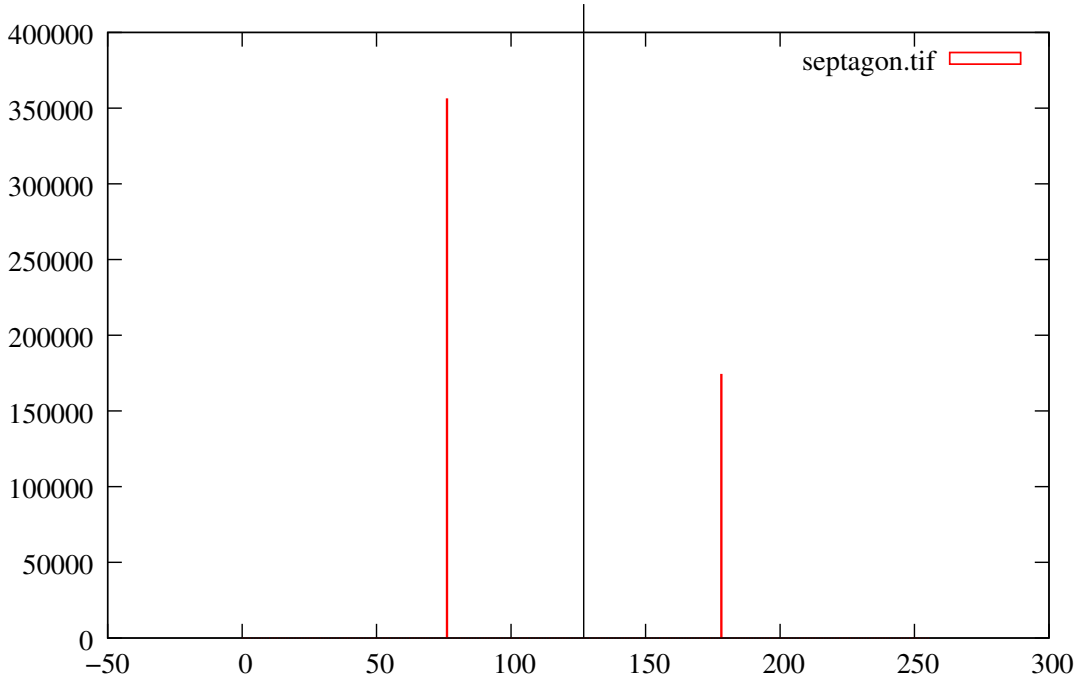


FIGURE – Seuil à 109

Seuillage optimal

Exemples



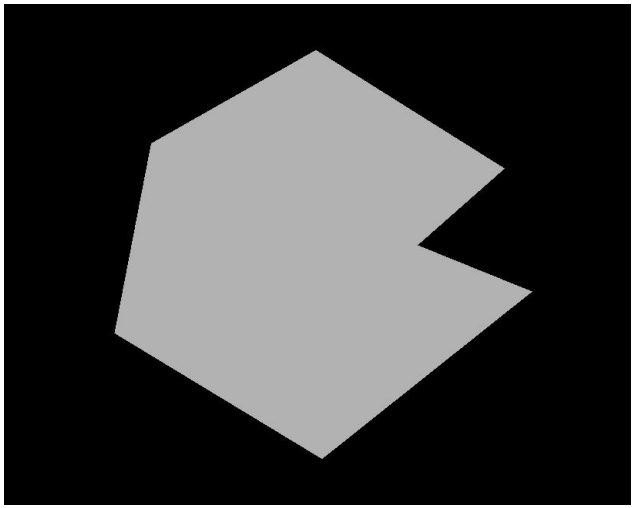
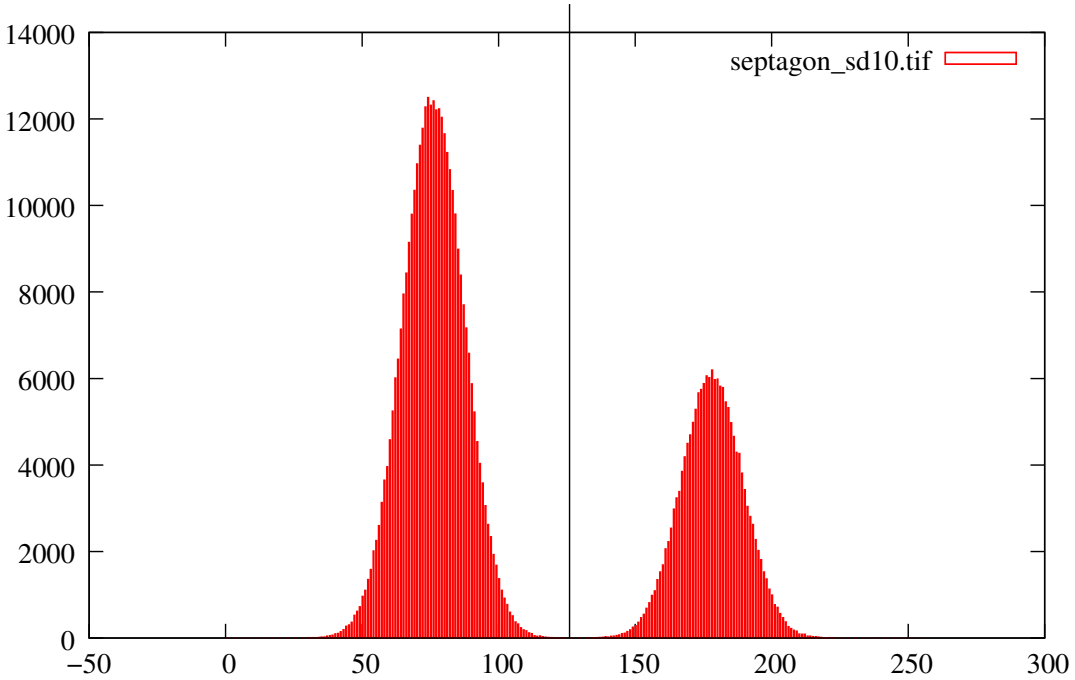


FIGURE – Seuil à 127

Seuillage optimal

Exemples



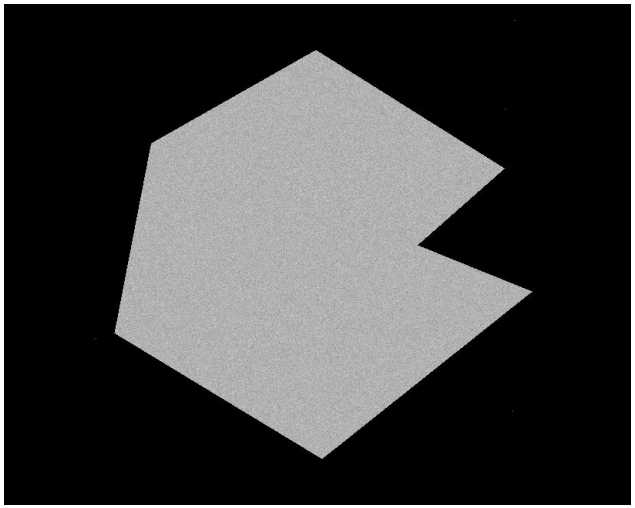
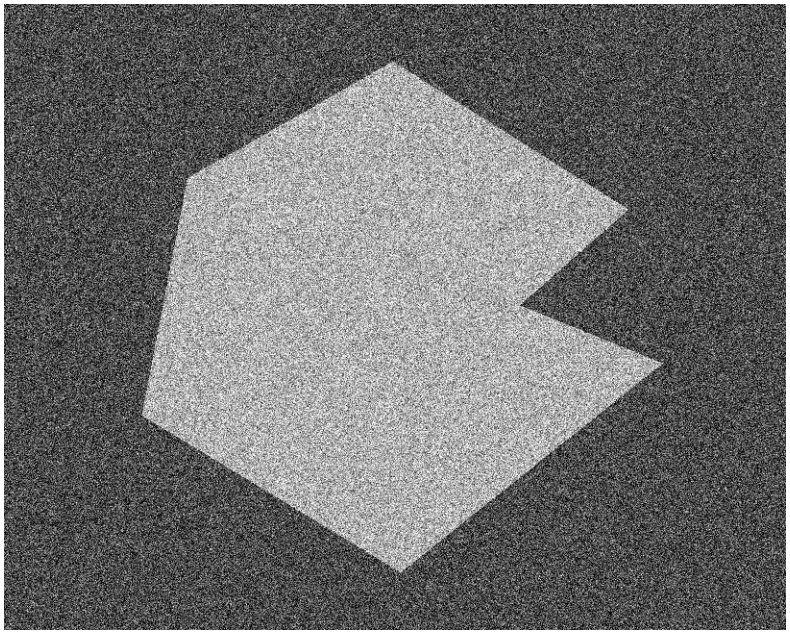


FIGURE – Seuil à 126

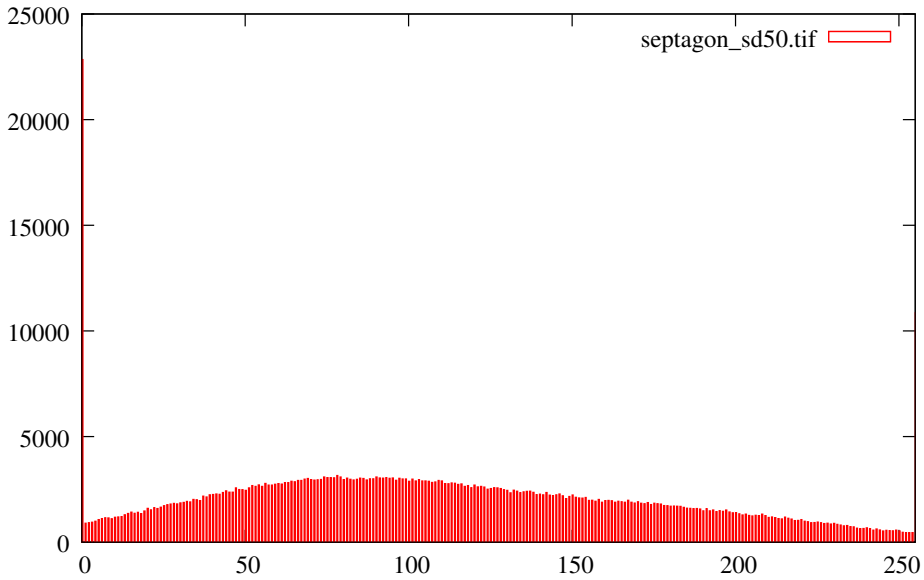
Seuillage

Les limites

- Trop de bruit ...



(a) bruit additif gaussien $\sigma = 50$

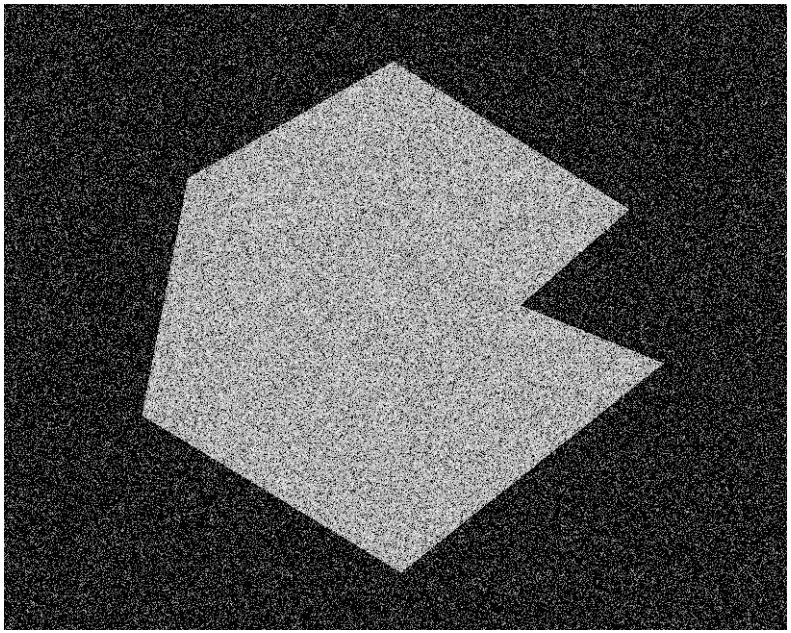


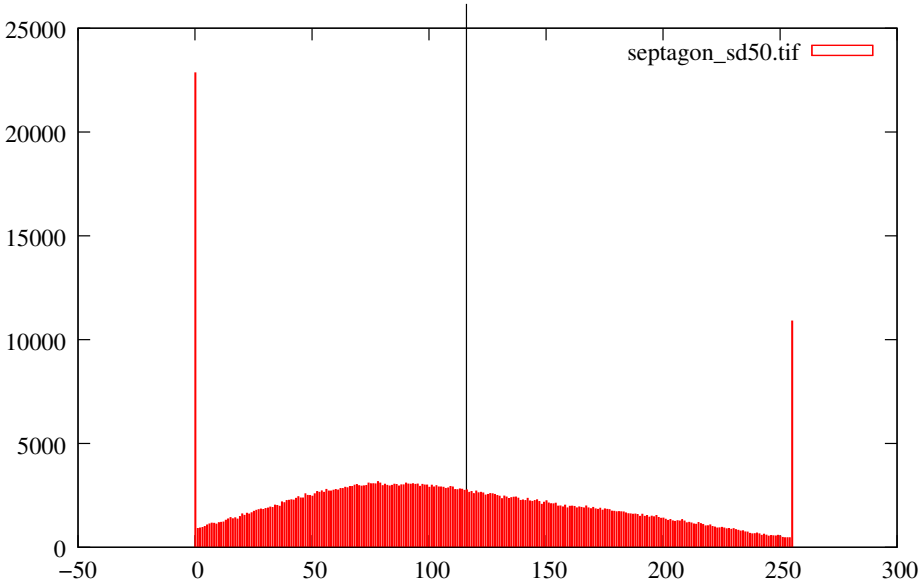
(b) histogramme : où seuiller ?

Seuillage

Les limites (bruit)

- Otsu trouve 116



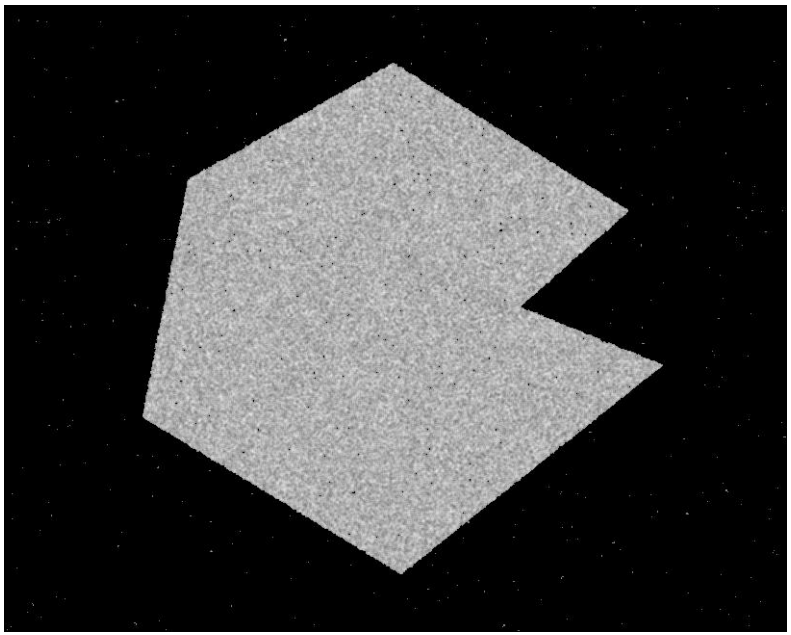


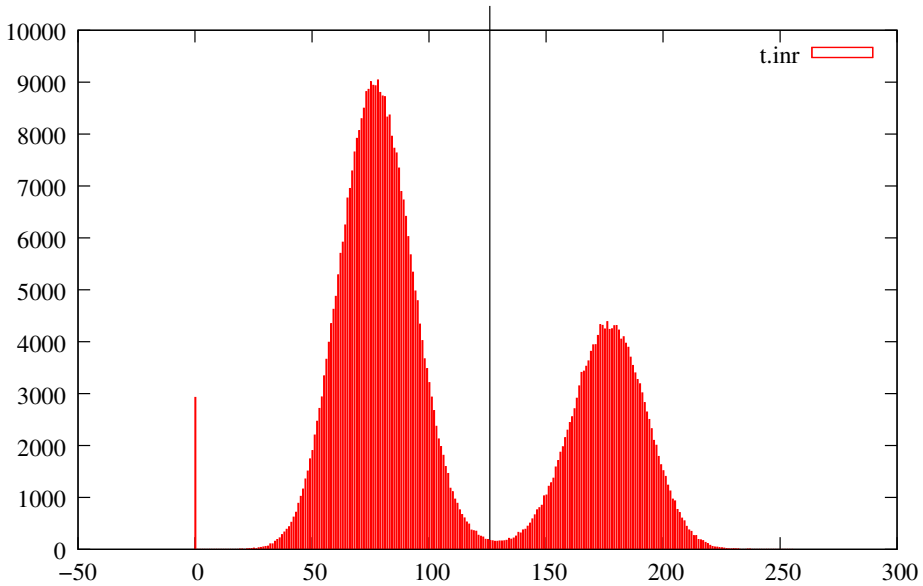
Seuillage

Les limites (bruit)

- En lissant (filtre moyenneur ou gaussien), on peut débruiter et seuiller convenablement :

```
fmoy septagon_sd50 | vb -n `fmoy septagon_sd50 | otsu` 0 | xvis
```



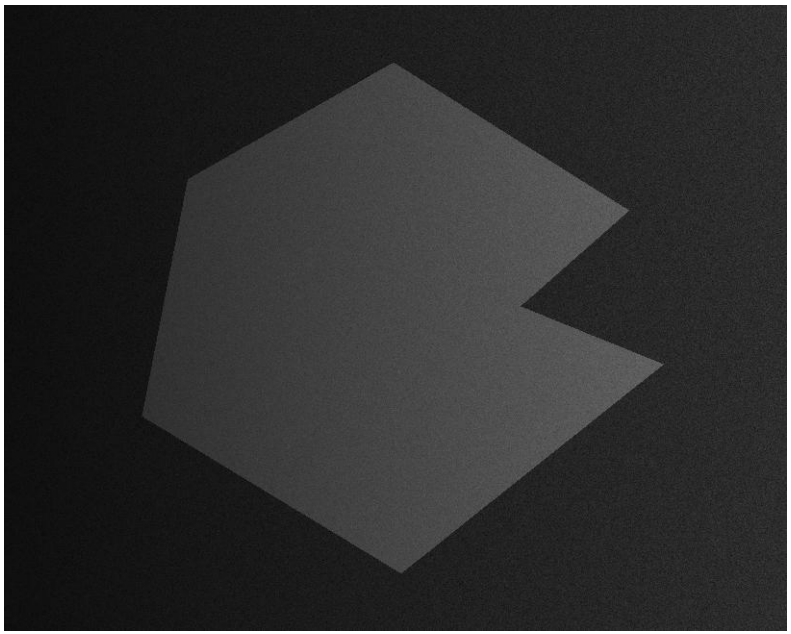


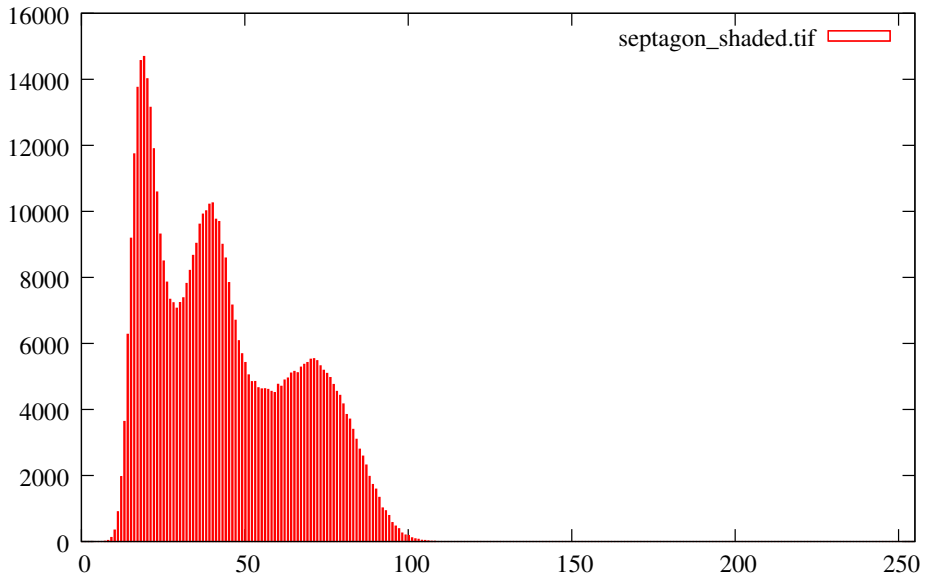
- à vous de coder la commande `otsu` !

Seuillage

Les limites

- Dégradé (non homogénéité)

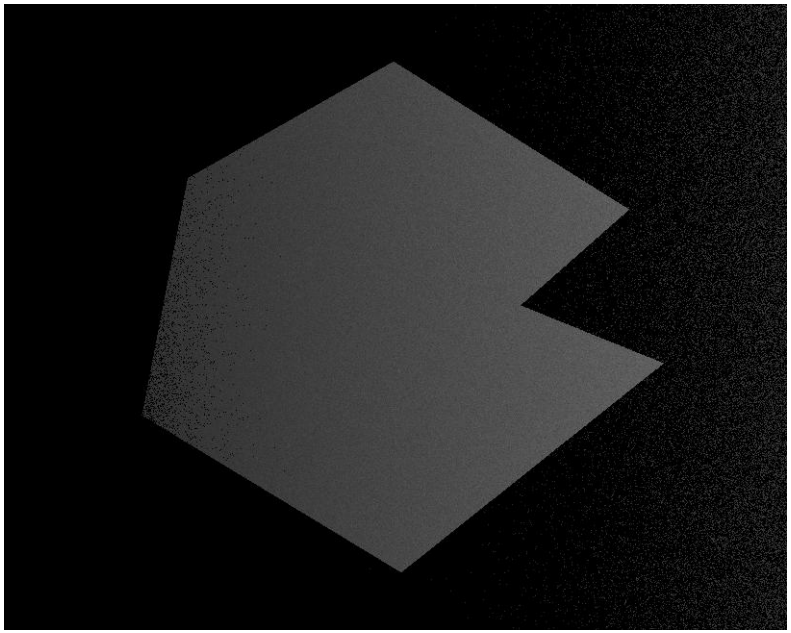


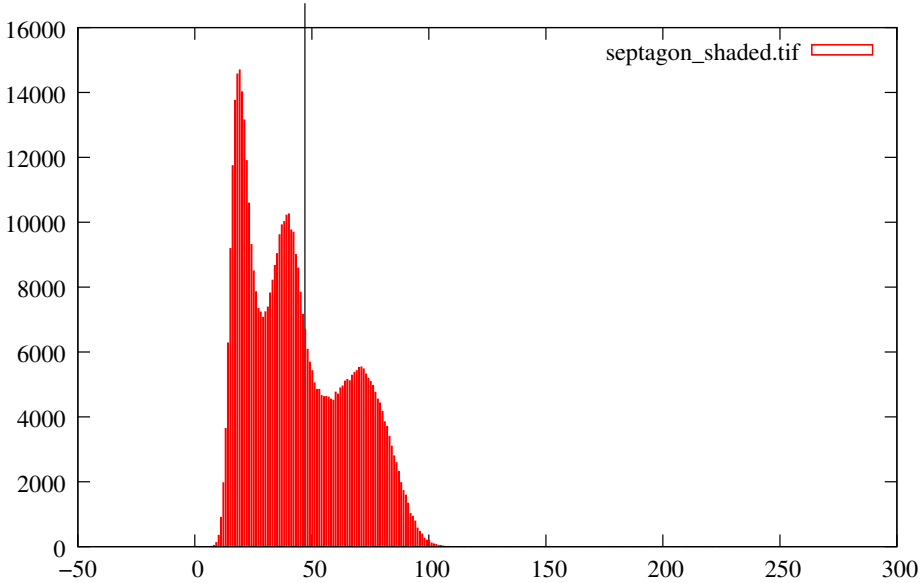


Seuillage

Les limites (inhomogénéité)

- Otsu trouve 47

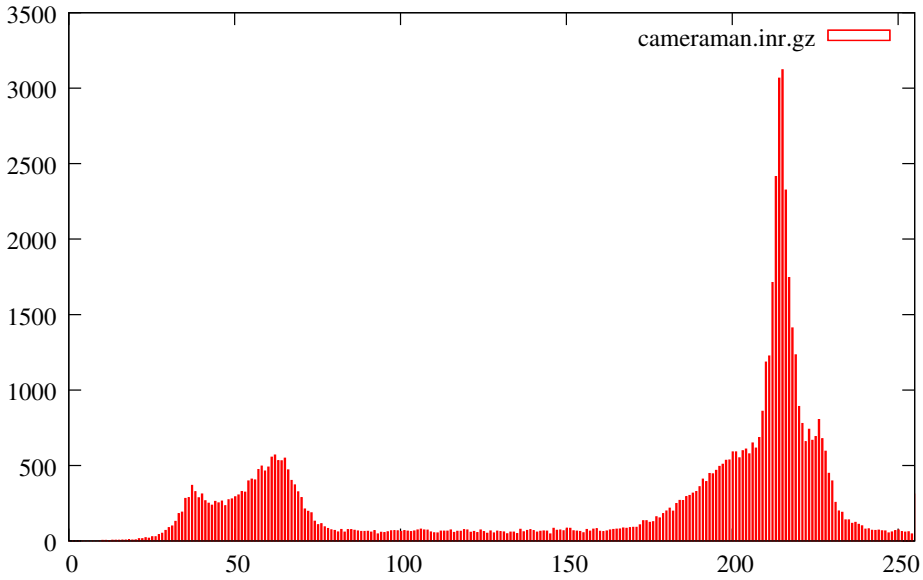




Seuillage

Les limites

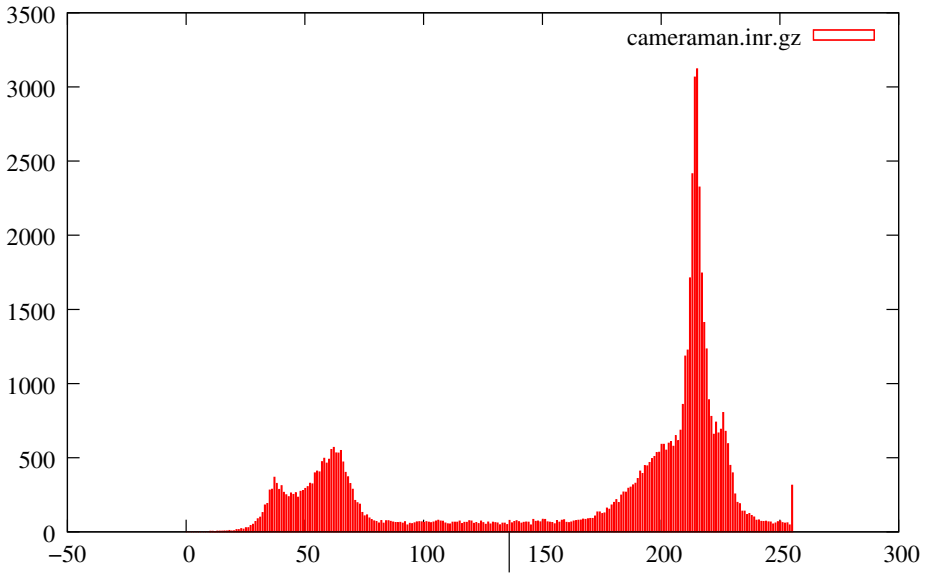
- Trop de classes



Seuillage

Les limites (trop de classes)

- Otsu trouve 136



Segmentation par découpage

Principe

- Construire une représentation en *Quadtree* de l'image.

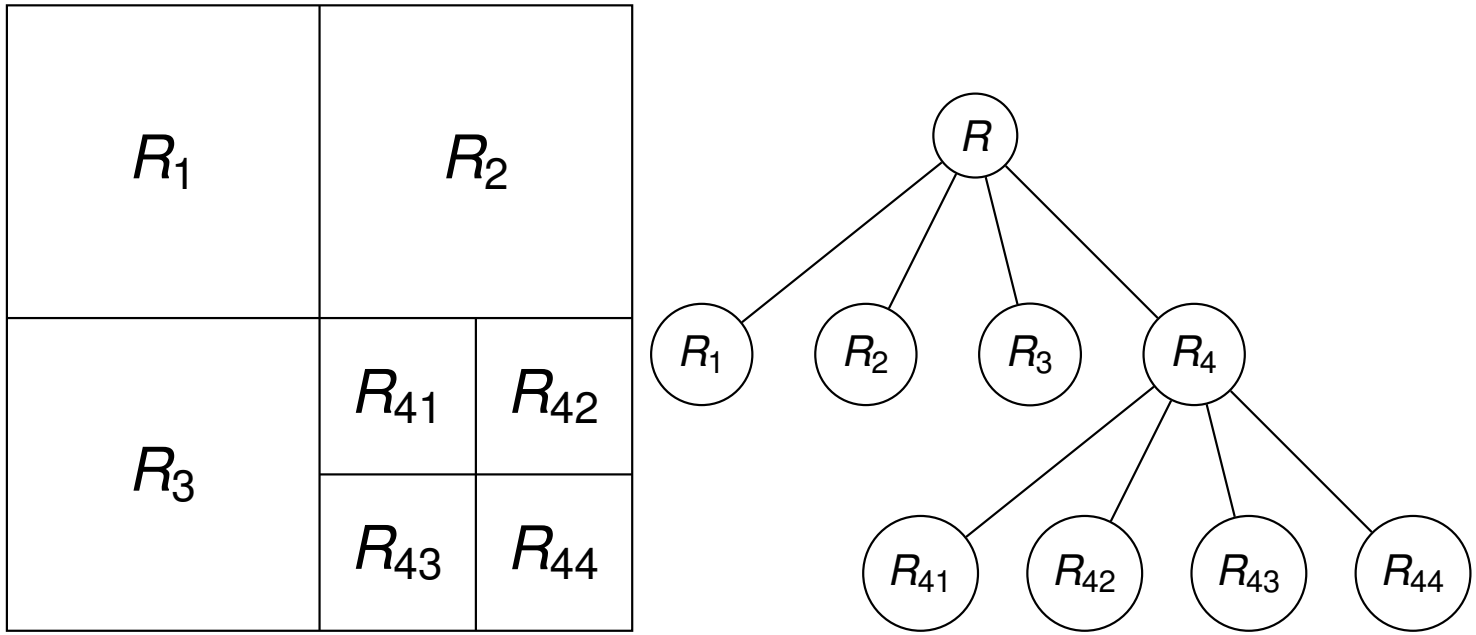


FIGURE – Décomposition en *Quadtree*

- Dans chaque région, un critère d'homogénéité de la répartition des niveaux de gris (ou des couleurs) est respectée.

Segmentation par découpage

Principe

- Pour décider du découpage d'une région R en 4 sous-régions, on se donne un critère (à préciser ultérieurement), appelé aussi prédicat, qui décide si une région est homogène ou non.
- Soit un algorithme

```
algo pred(R:region): booleen;
```

qui retourne `VRAI` lorsque le critère d'homogénéité est vérifié pour la région R, et `FAUX` sinon.

- On cherche à découper l'image en régions R_i telles qu'à la fin de l'algorithme on ait :
  1. $I = \bigcup_i R_i$ et $R_i \bigcap R_j = \emptyset$
  2. `Pred(`R_i`) = VRAI`

Segmentation par découpage

Algorithme

- Un tel algorithme est facilement implémenté récursivement :

```
algo split_rec( I:image, R:region, qt: arbre 4-aire)
   si pred(R) = FAUX et taille(R) > rmin alors
     qt.no := creer_noeud();
     qt.ne := creer_noeud();
     qt.so := creer_noeud();
     qt.se := creer_noeud();
     split_rec(I,R.nord_ouest,qt.no);
     split_rec(I,R.nord_est,qt.ne);
     split_rec(I,R.sud_ouest,qt.so);
     split_rec(I,R.sud_est,qt.se);
   fin si
fin algo
```

Segmentation par découpage

Algorithme

```
algo split( I:image): arbre 4-aire
    racine := creer_noeud();
    split_rec( I, dims(I), racine);
    split := racine;
fin algo
```

- En pratique, chaque noeud `qt` de l'arbre doit contenir les informations suivantes :
  - coordonnées de la région (un rectangle),
  - statistiques diverses sur la région image, utile pour calculer le critère de découpage.

Segmentation par découpage

Critère de découpage

- Dépendra des propriétés que l'on souhaite voir vérifiées par chaque région après segmentation.
- Homogénéité des niveaux de gris :
  - après segmentation chaque région est homogène dans sa distribution des valeurs de niveaux de gris : la variance de la région est plus petite qu'un seuil à fixer.
  - Remarque : une région réduite à un pixel est de variance nulle.
- Homogénéité en texture/couleurs : non abordée dans ce cours d'initiation.
- Taux de contours réduit : appliquer un détecteur de contours dans la région R et découper si :

$$\tau = \frac{|\text{points de contours}|}{|R| - |\text{points de contours}|} > \text{seuil}$$

- Critère mixte (homogénéité, taux de contours).

Segmentation par découpage

Exemples

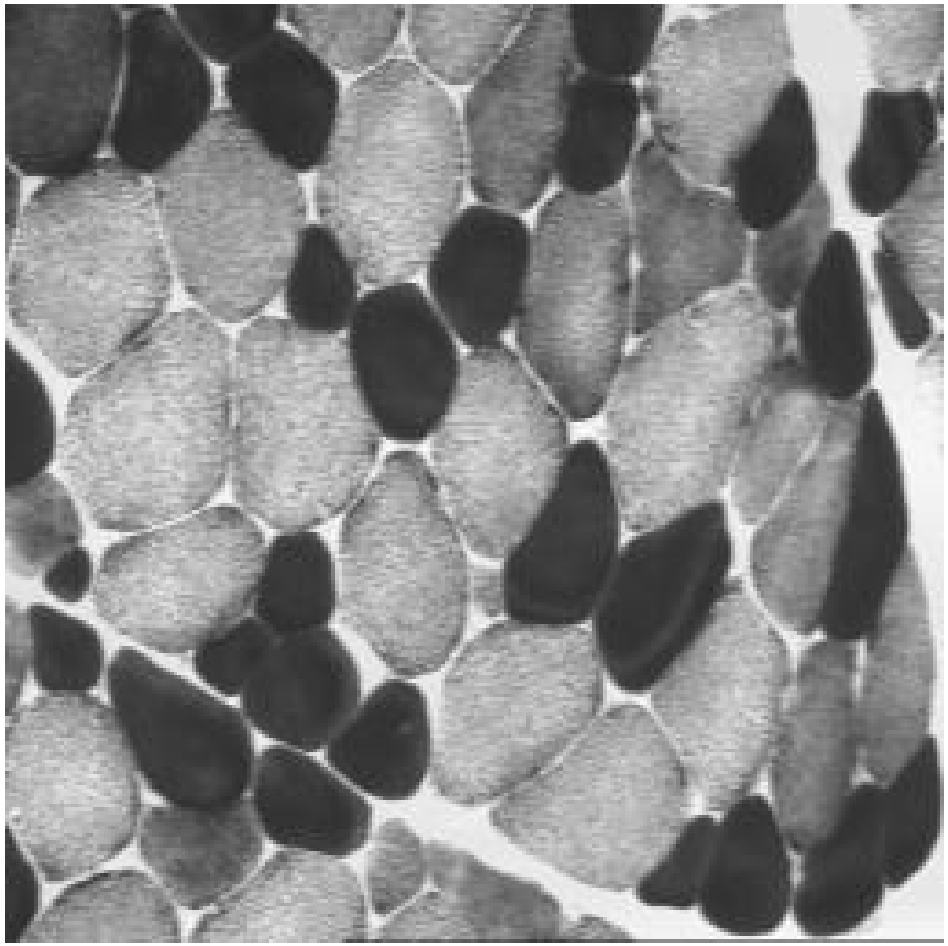


FIGURE – Variance de cette image : 3210, taille : 256×256

Segmentation par découpage

Exemples

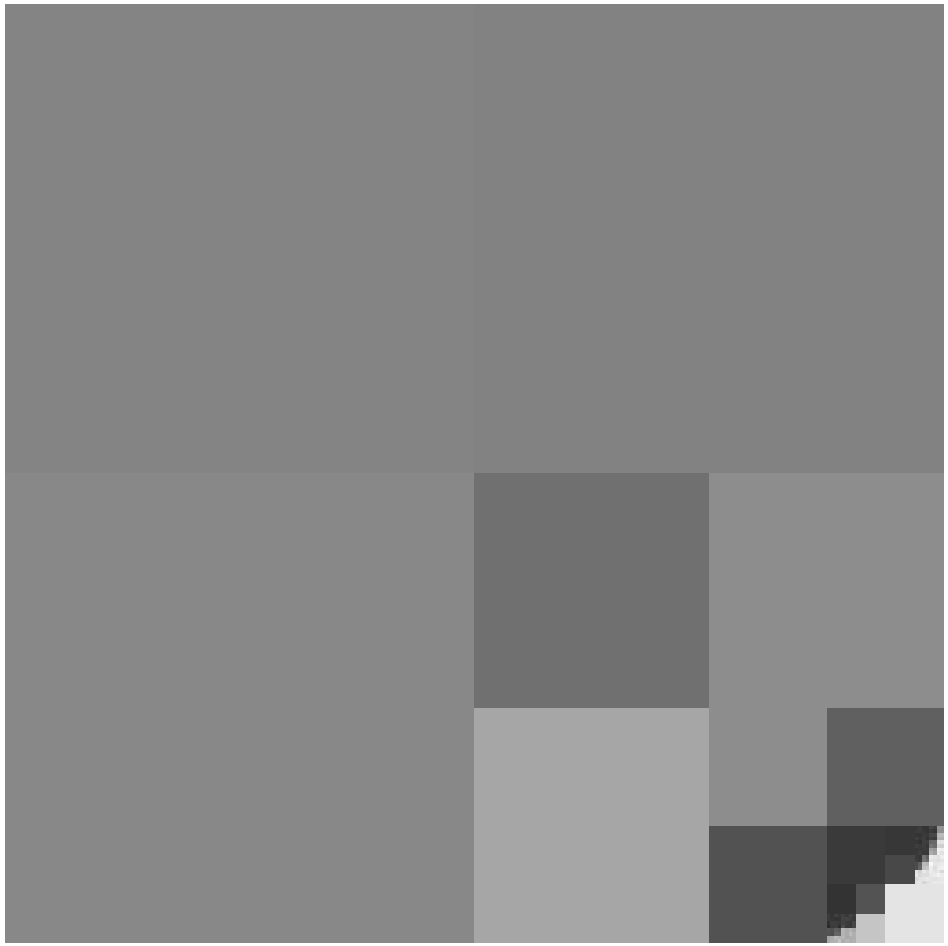


FIGURE – Seuil du découpage à 3210, 40 régions

Segmentation par découpage

Exemples

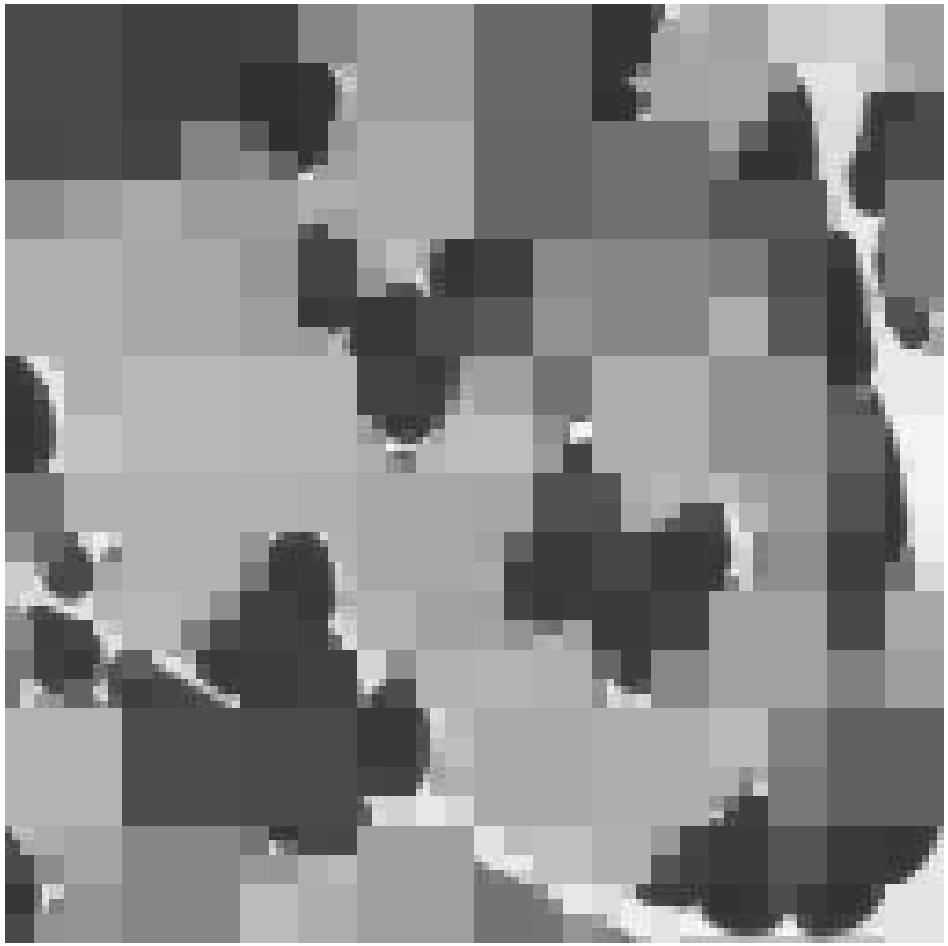


FIGURE – Seuil du découpage à 1605, 1144 régions

Segmentation par découpage

Exemples

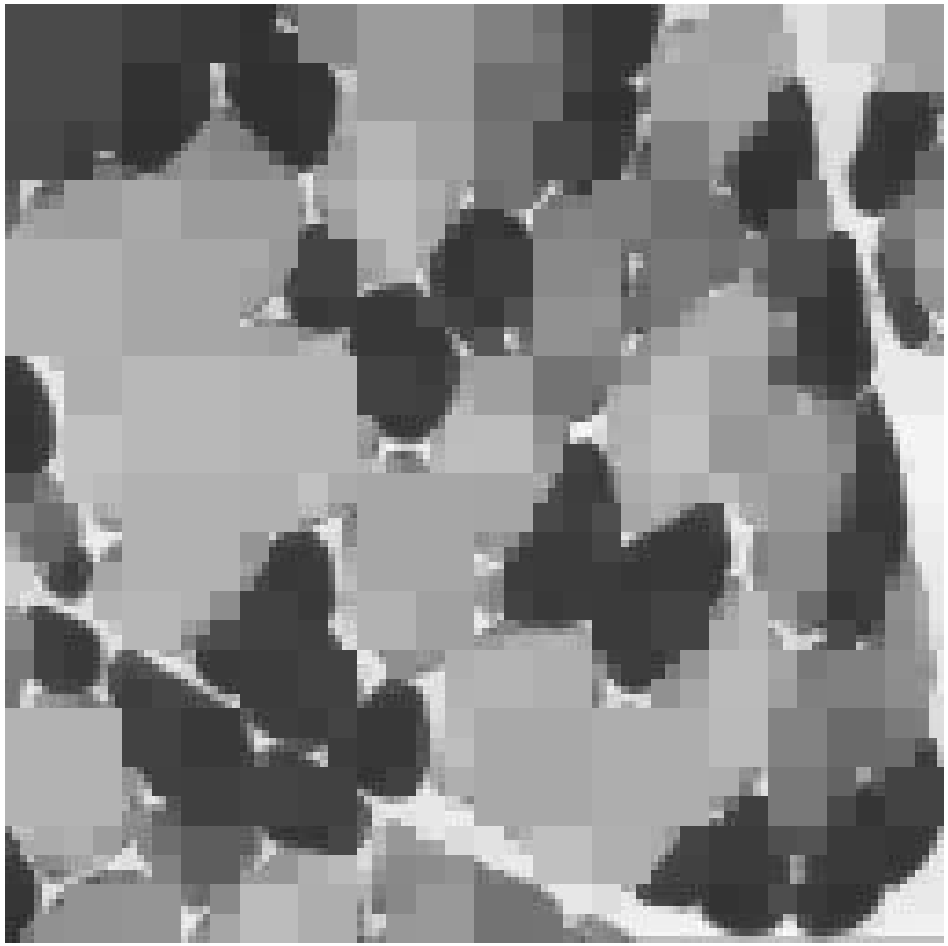


FIGURE – Seuil du découpage à 802, 2500 régions

Segmentation par découpage

Exemples

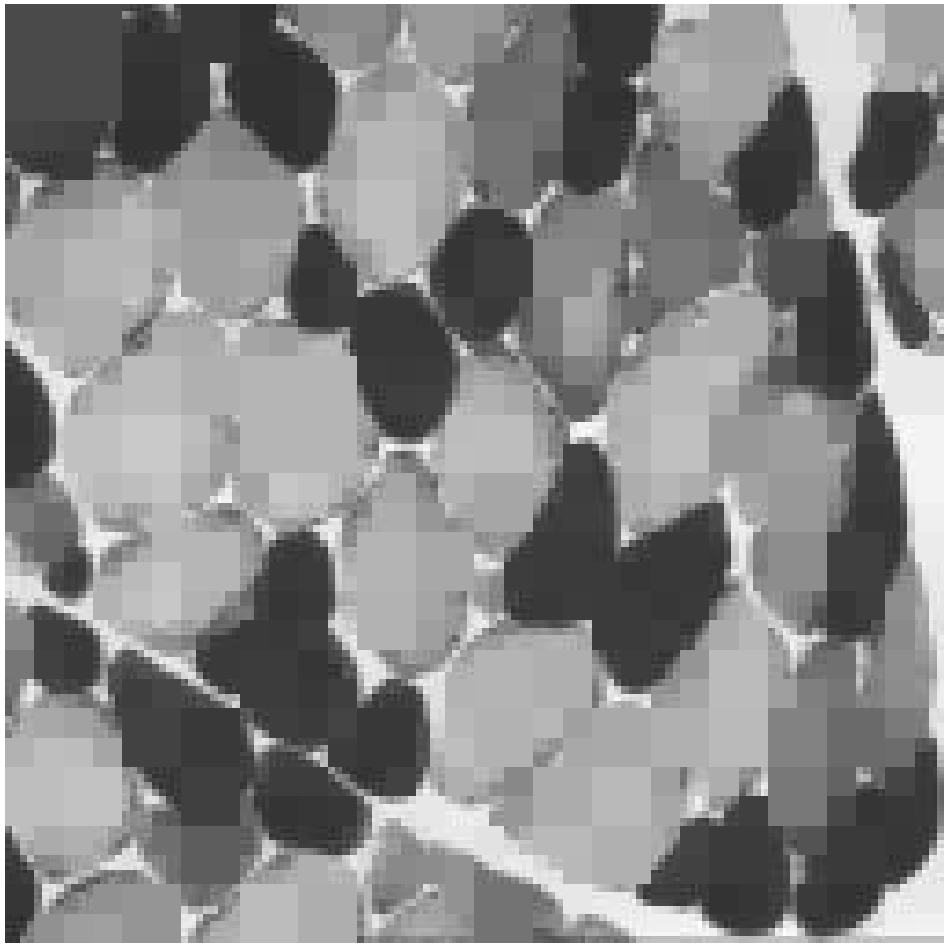


FIGURE – Seuil du découpage à 401, 4288 régions

Segmentation par découpage

Exemples

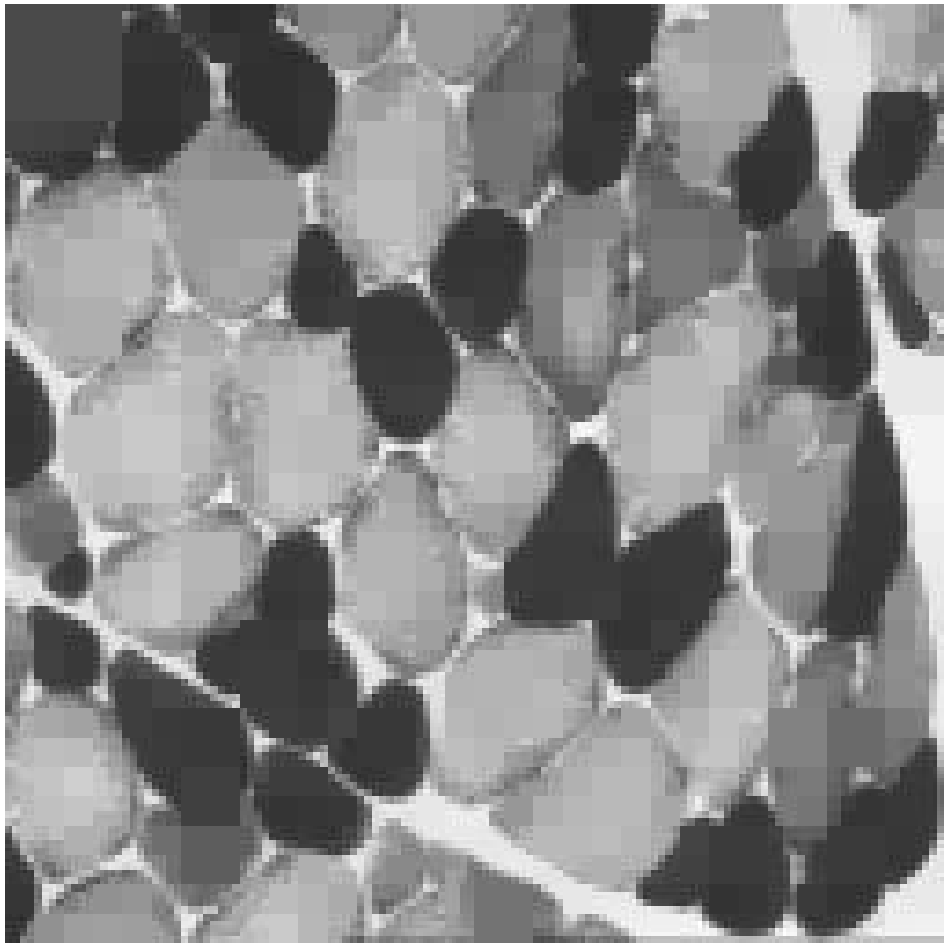


FIGURE – Seuil du découpage à 200, 6196 régions (10 pixels par région)

Segmentation par découpage

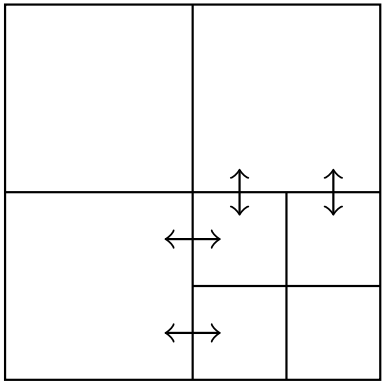
Conclusion sur l'exemple

- Un découpage au seuil de 200 signifie que toutes les régions ont une variance inférieure à 200.
- C'est-à-dire un écart-type inférieur à 14, c'est-à-dire à l'intérieur de chaque région, les pixels ont une valeur comprise entre $\mu - 14$ et $\mu + 14$ où μ est la moyenne de la région considérée.
- On voit également qu'en moyenne les régions possèdent 10 pixels ($\frac{65535 \text{ pixels}}{6196 \text{ régions}}$).
- Pour obtenir une bonne uniformité des régions (variance inférieure à 200), il a fallu découper en petites régions !

Segmentation par découpage

Conclusion sur la méthode

- La segmentation par découpage conduit donc à une sur-segmentation.
- À cause du découpage en *Quadtree*, on trouve des régions non voisines dans l'arbre, adjacentes dans l'image (par exemple R_3 et R_{41}) et qui vérifient peut-être le critère d'homogénéité.



- En particulier on aimerait ajouter aux deux conditions de la page 41, celle-ci :
  - `split_criterion(`$R_i \bigcup R_j$`) = FAUX` pour toutes régions i et j adjacentes.

Segmentation par découpage/fusion

Principe

- Cette nouvelle condition impose donc une phase dite de *fusion*, pendant laquelle on examine chaque couple de régions adjacentes possible et on vérifie si le critère de fusion est vérifiée (troisième condition).
- Chaque fusion conduit à une nouvelle structure de régions et nécessite un réexamen des fusions !
- Il n'est pas interdit d'utiliser un critère de fusion différent du critère de découpage :
  - par exemple plutôt que de vérifier que la variance des régions fusionnées est en dessous du seuil de découpage, vérifier si la variance décroît :

$$\sigma(R_i \bigcup R_j) < \max(\sigma(R_i), \sigma(R_j))$$

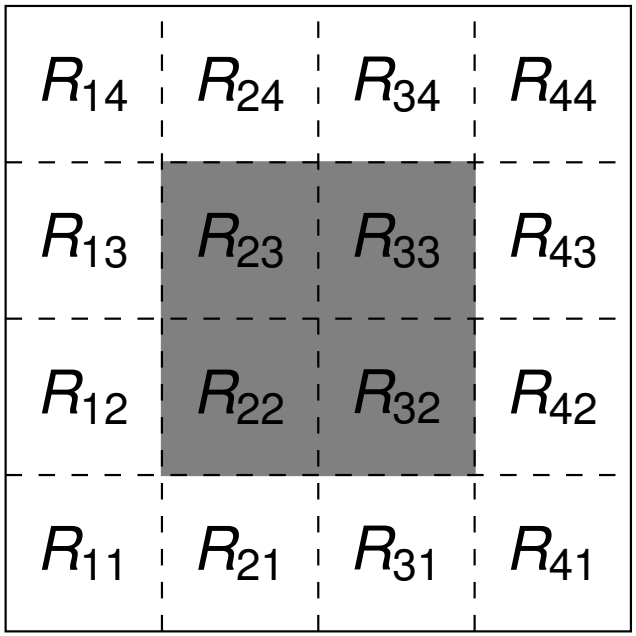
- Algorithme de découpage/fusion (*Split and merge*).

Segmentation par découpage/fusion

```
algo merge(qt: arbre 4-aire): liste region
   K:= liste_feuilles(qt); L:=creer_liste();
   tantque K non vide faire
     b := K[1]; R:=creer_region();
     retire(b,K); ajoute(b,R);
     N := voisins(b);
     pour chaque bn dans N faire
       si (bn appartient a K) et
            critere_fusion(bn,R) = VRAI alors
         retire(bn,K); ajoute(bn,R);
         ajoute(voisins(bn), N);
       fin si
     fin pour
     L := [L,R];
   fin pour
   merge := L;  fin algo
```

Segmentation par découpage/fusion

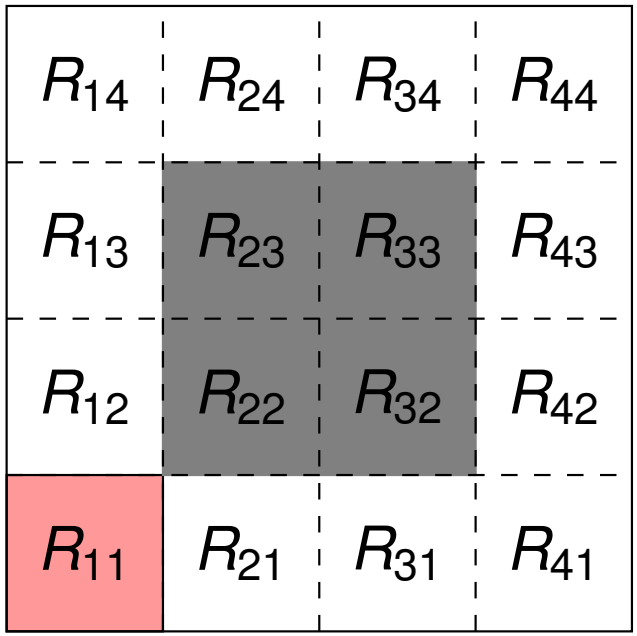
Fonctionnement de l'algorithme



- $K = \{R_{11}, R_{12}, \cdots, R_{44}\}$
- $L = \{\}$
- $R = \{\}$
- $N = \{\}$

Segmentation par découpage/fusion

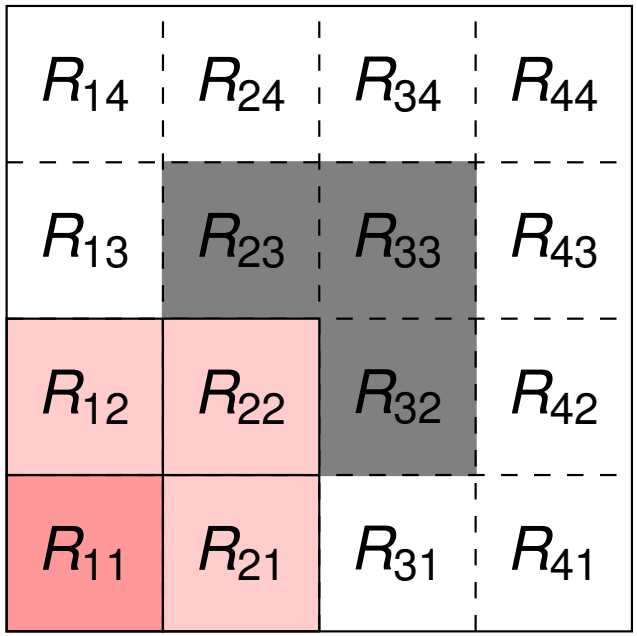
Fonctionnement de l'algorithme



- $K = \{R_{12}, R_{13}, \cdots, R_{44}\}$
- $L = \{\}$
- $R = \{R_{11}\}$
- $N = \{\}$

Segmentation par découpage/fusion

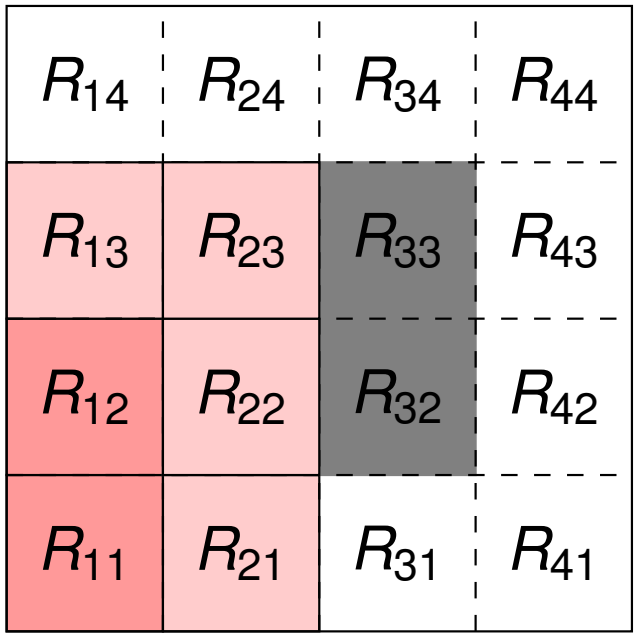
Fonctionnement de l'algorithme



- $K = \{R_{12}, R_{13}, \cdots, R_{44}\}$
- $L = \{\}$
- $R = \{R_{11}\}$
- $N = \{R_{12}, R_{22}, R_{21}\}$

Segmentation par découpage/fusion

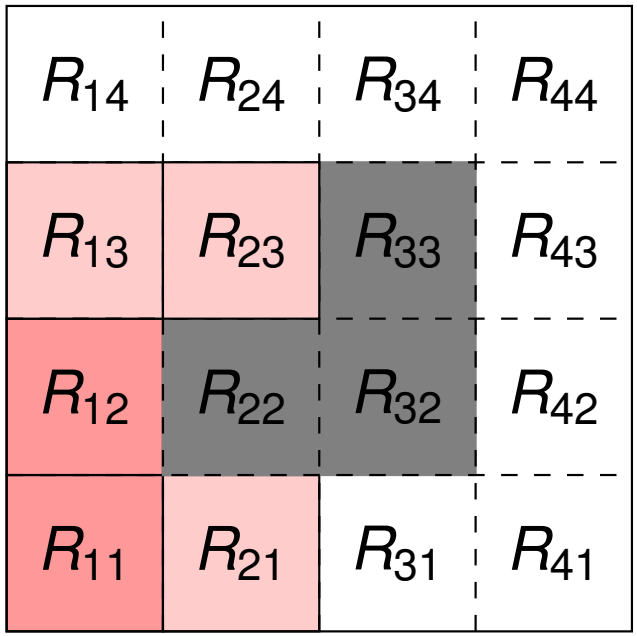
Fonctionnement de l'algorithme



- $K = \{R_{13}, R_{14}, \cdots, R_{44}\}$
- $L = \{\}$
- $R = \{R_{11}, R_{12}\}$
- $N = \{R_{22}, R_{21}, R_{13}, R_{23}\}$

Segmentation par découpage/fusion

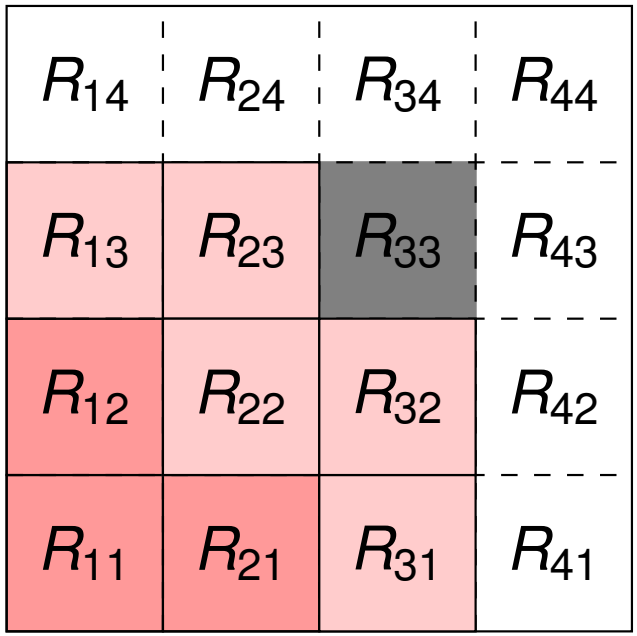
Fonctionnement de l'algorithme



- $K = \{R_{13}, R_{14}, \cdots, R_{44}\}$
- $L = \{\}$
- $R = \{R_{11}, R_{12}\}$
- $N = \{R_{21}, R_{13}, R_{23}\}$

Segmentation par découpage/fusion

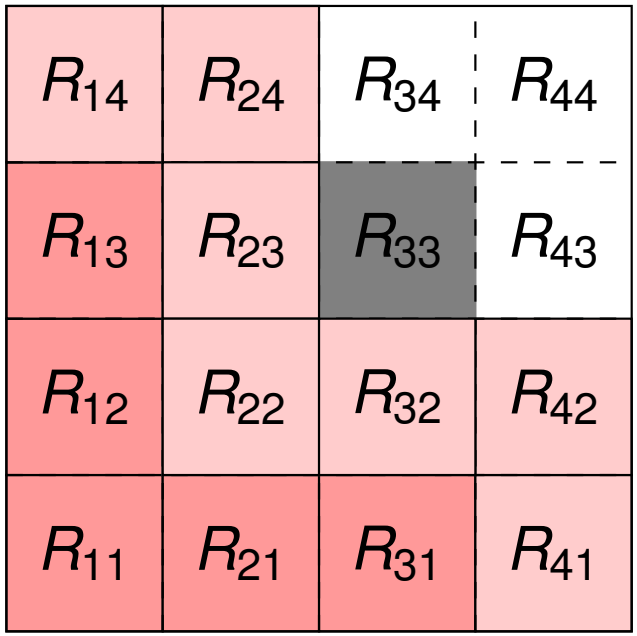
Fonctionnement de l'algorithme



- $K = \{R_{13}, R_{14}, \cdots, R_{44}\}$
- $L = \{\}$
- $R = \{R_{11}, R_{12}, R_{21}\}$
- $N = \{R_{13}, R_{23}, R_{22}, R_{32}, R_{31}\}$

Segmentation par découpage/fusion

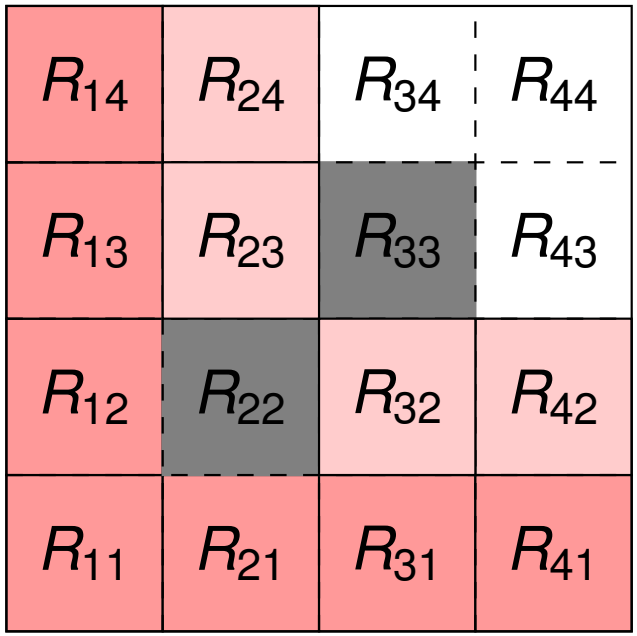
Fonctionnement de l'algorithme



- $K = \{R_{14}, R_{22}, \cdots, R_{44}\}$
- $L = \{\}$
- $R = \{R_{11}, R_{12}, R_{21}, R_{13}, R_{31}\}$
- $N = \{R_{14}, R_{24}, R_{23}, R_{22}, R_{32}, R_{42}, R_{41}\}$

Segmentation par découpage/fusion

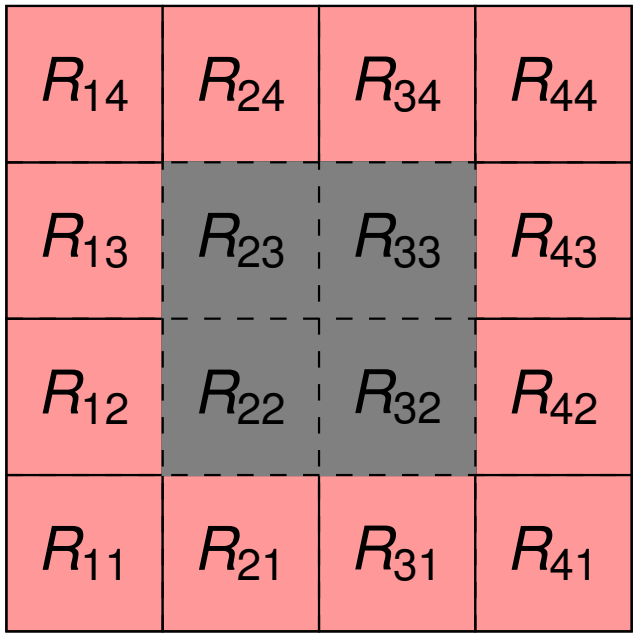
Fonctionnement de l'algorithme



- $K = \{R_{22}, R_{23}, \cdots, R_{44}\}$
- $L = \{\}$
- $R = \{R_{11}, R_{12}, R_{21}, R_{13}, R_{31}, R_{14}, R_{41}\}$
- $N = \{R_{24}, R_{23}, R_{32}, R_{42}\}$

Segmentation par découpage/fusion

Fonctionnement de l'algorithme



- $K = \{R_{22}, R_{23}, R_{32}, R_{33}\}$
- $L = \{R\}$
- $R = \{R_{11}, R_{12}, R_{21}, R_{13}, R_{31}, R_{14}, \cdots, R_{41}, R_{24}, R_{42}, R_{34}, R_{43}, R_{44}\}$
- $N = \{\}$

Segmentation par découpage/fusion

Exemples

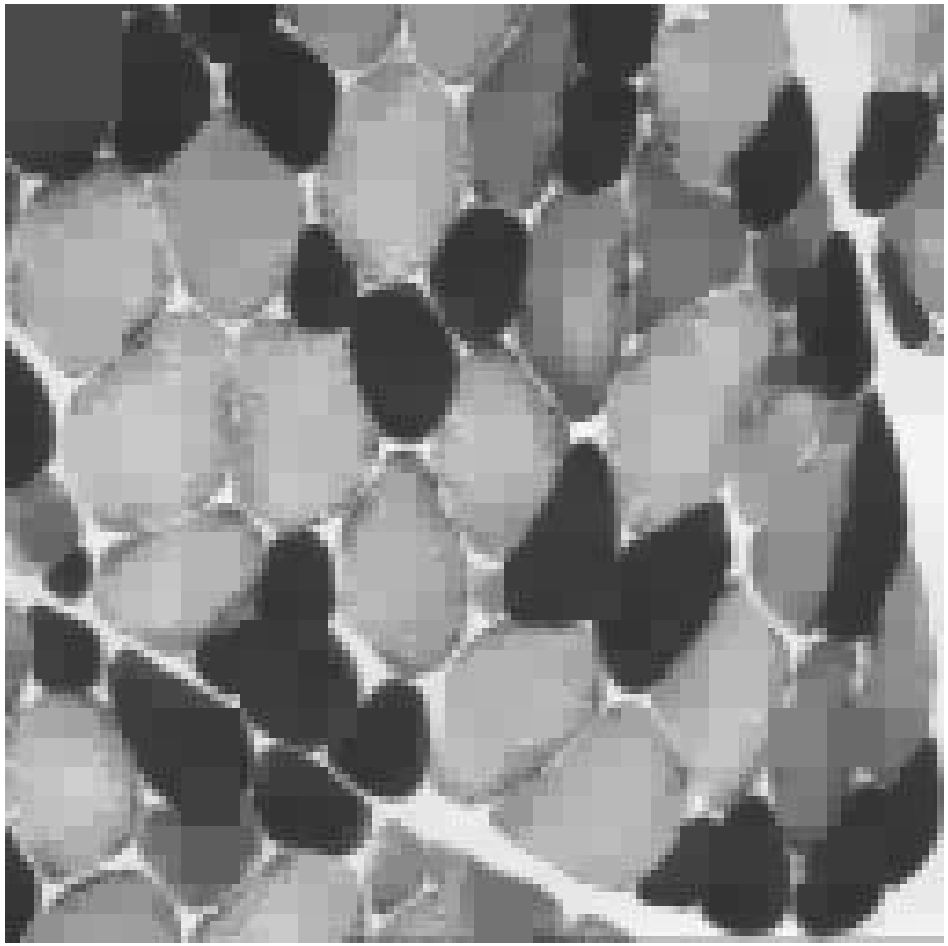


FIGURE – Découpage à $\sigma^2 = 200$ (6196 régions)

Segmentation par découpage/fusion

Exemples

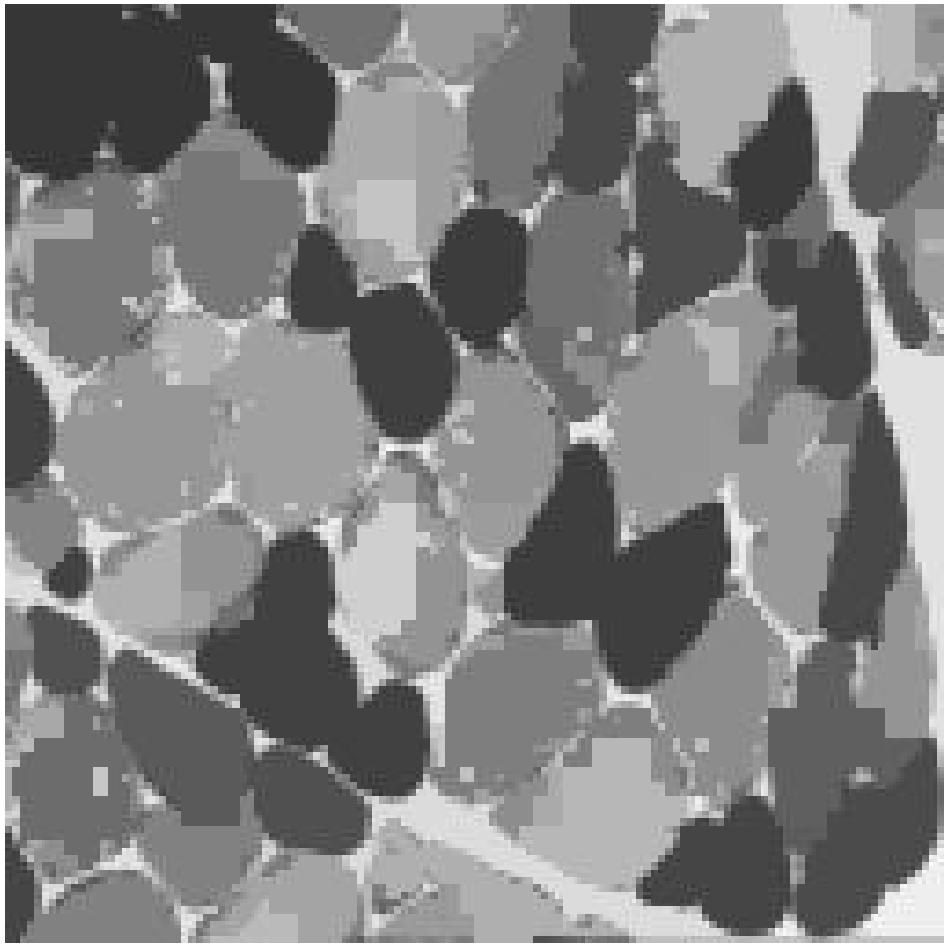


FIGURE – Fusion à $\sigma^2 = 200$ (2558 régions)

Segmentation par découpage/fusion

Exemples

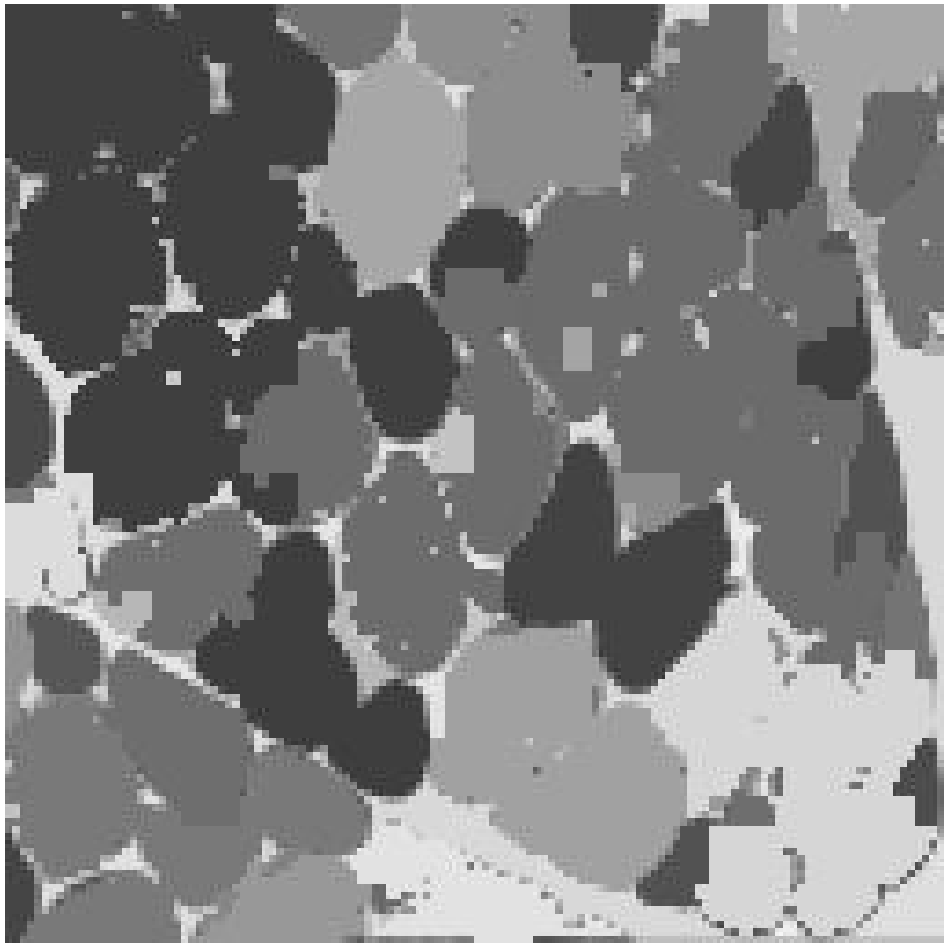


FIGURE – Fusion à $\sigma^2 = 400$ (1440 régions)

Segmentation par découpage/fusion

Exemples

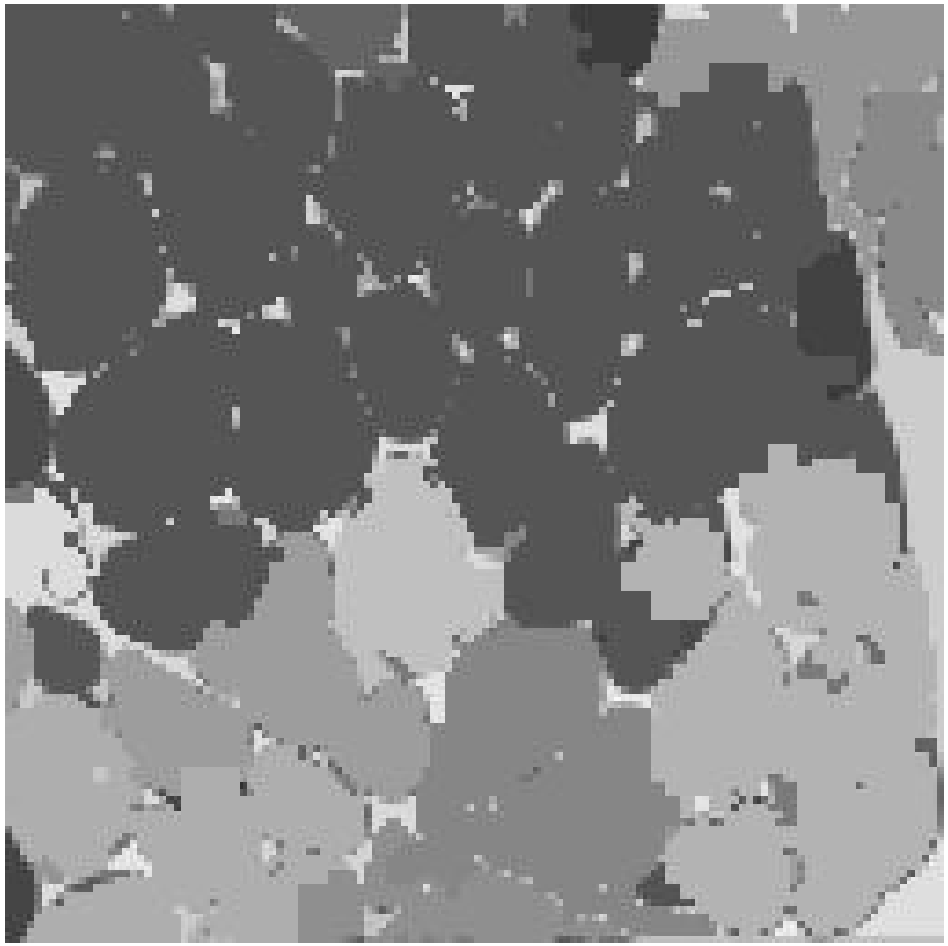


FIGURE – Fusion à $\sigma^2 = 600$ (1023 régions)

Segmentation par découpage/fusion

Exemples

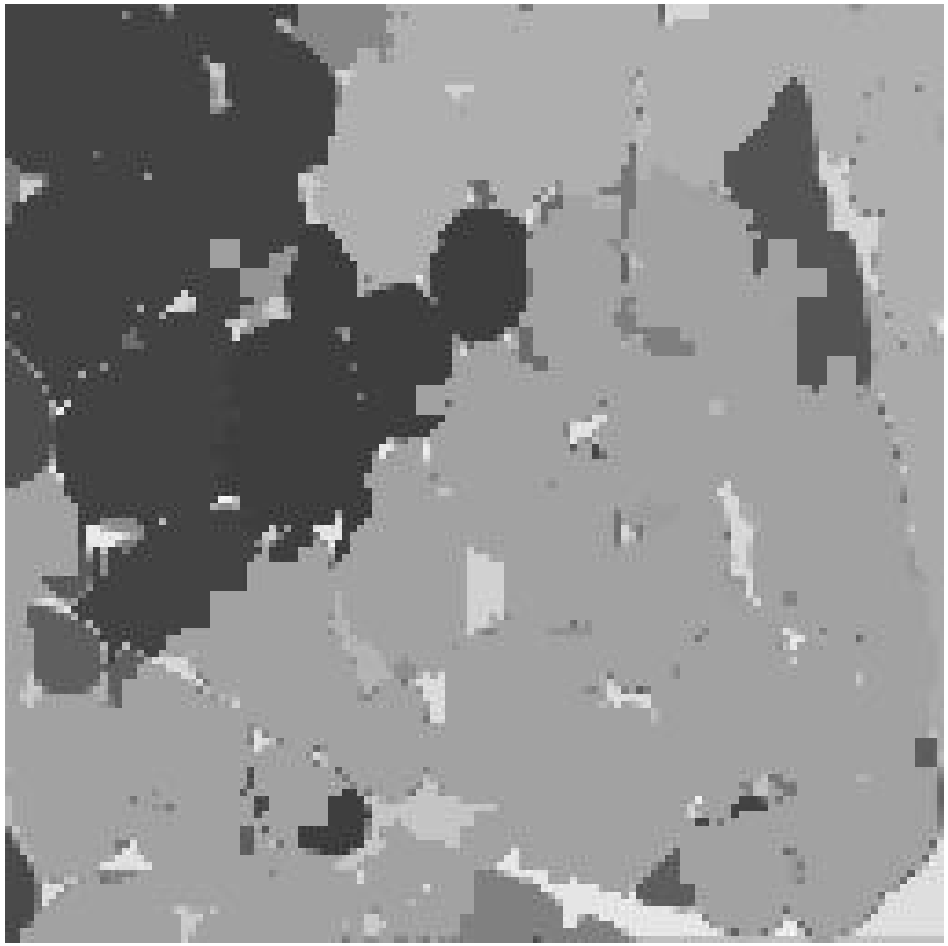
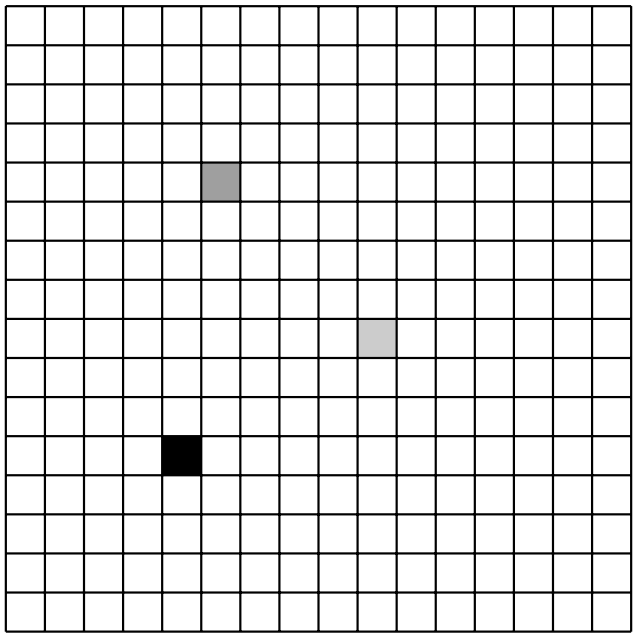


FIGURE – Fusion à $\sigma^2 = 800$ (721 régions)

Croissance de régions (*Growing regions*)

Principe

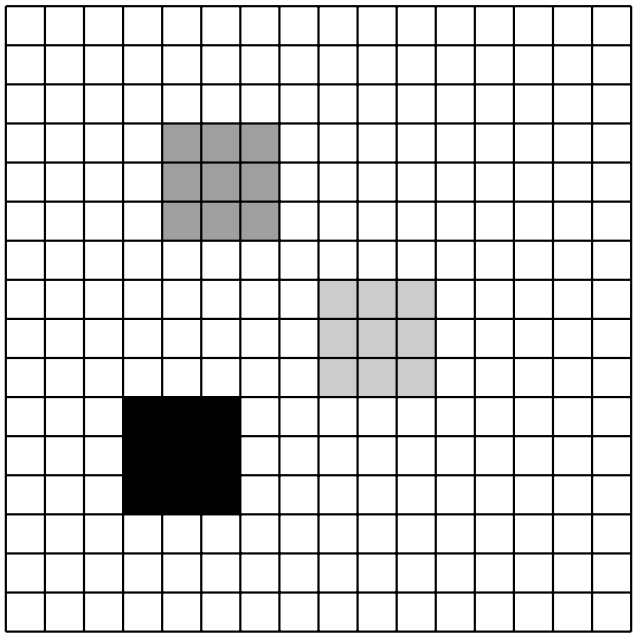
- Choisir des points initiaux (graines).
- À partir de ces graines (régions initiales), on agglomère les pixels voisins si un critère d'agglomération est vérifié.
- Le procédé est itéré jusqu'à rencontrer des pixels déjà segmentés ou si les régions ne grossissent plus.



Croissance de régions (*Growing regions*)

Principe

- Choisir des points initiaux (graines).
- À partir de ces graines (régions initiales), on agglomère les pixels voisins si un critère d'agglomération est vérifié.
- Le procédé est itéré jusqu'à rencontrer des pixels déjà segmentés ou si les régions ne grossissent plus.



Croissance de régions (Algorithme)

```
algo  growing(I:image, S: points, LR: regions)
  pour chaque point p dans S faire
    si p n'est pas dans LR alors
      R:=[p]; // nouvelle region
      N:=voisins(p);
      pour chaque point pn dans N faire
        si pn n'est pas dans LR et
            est_homogene(R,pn) alors
              R:=[R,pn];
              N:=[N,voisins(pn)];
        fin si
      fin pour
    fin si
    LR:=[LR,R];
  fin pour
fin algo
```

Croissance de régions

Paramètres

- Le critère de croissance : il est semblable à celui des algorithmes de découpage / fusion.
  - critère basé sur la variance ;
  - critère basé sur les contours ;
  - critère mixte.
- L'algorithme dépend fortement des graines !
- Cette dernière fixe par exemple le nombre de régions !
- Choix supervisé.
- Choix non supervisé :
  - algorithme de découpage (en *quad tree*),
  - le centre des régions découpées fournissent les graines.

Croissance de régions

Exemples

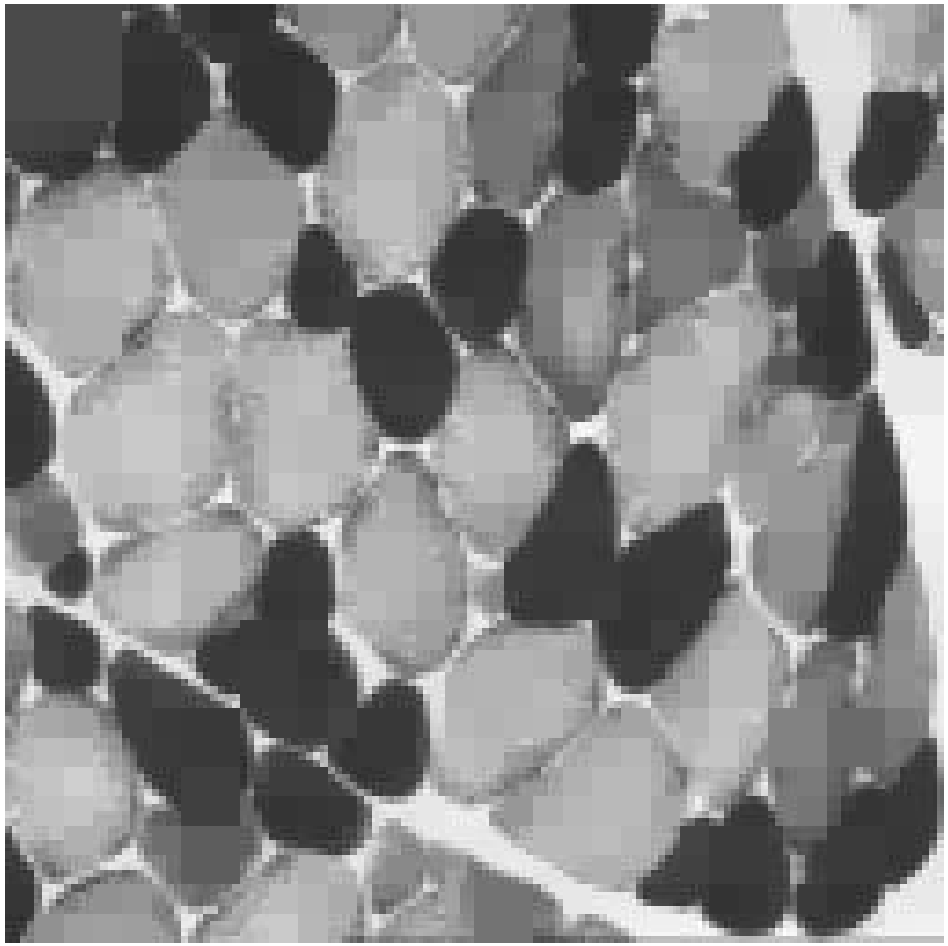


FIGURE – Initialisation : découpage à $\sigma = 200$

Croissance de régions

Exemples

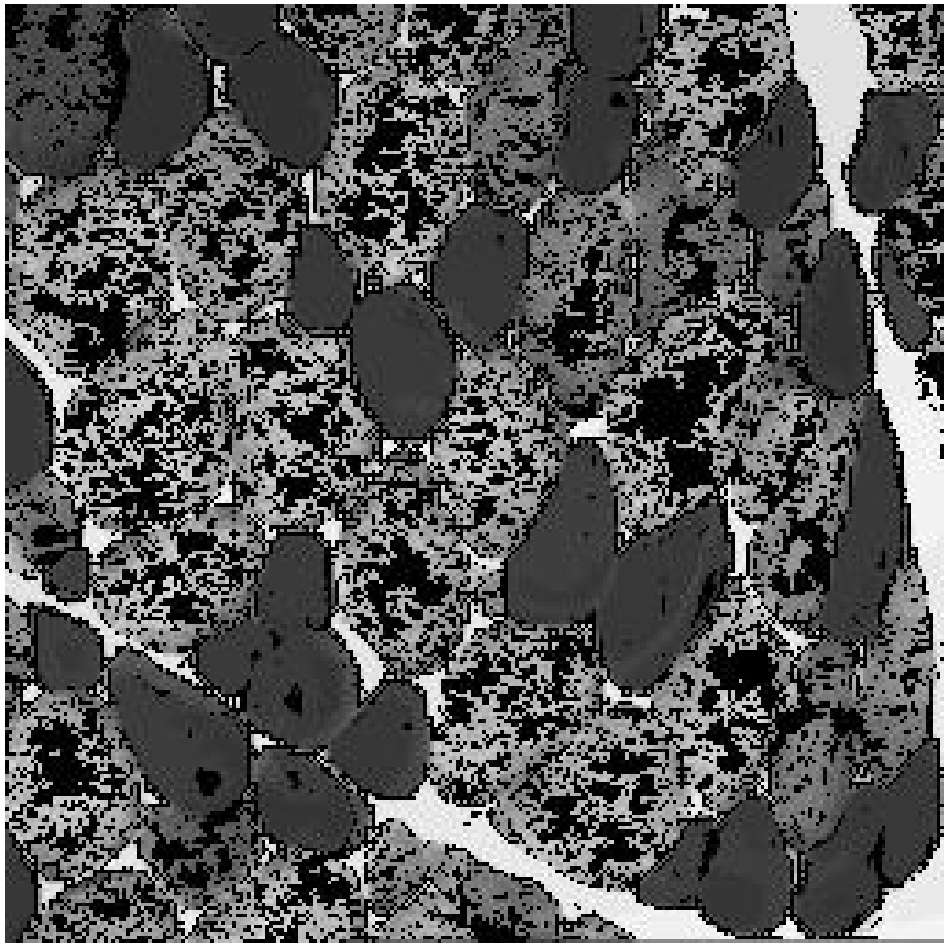


FIGURE – Seuil croissance à 10 (21504 pixels non segmentés)

Croissance de régions

Exemples

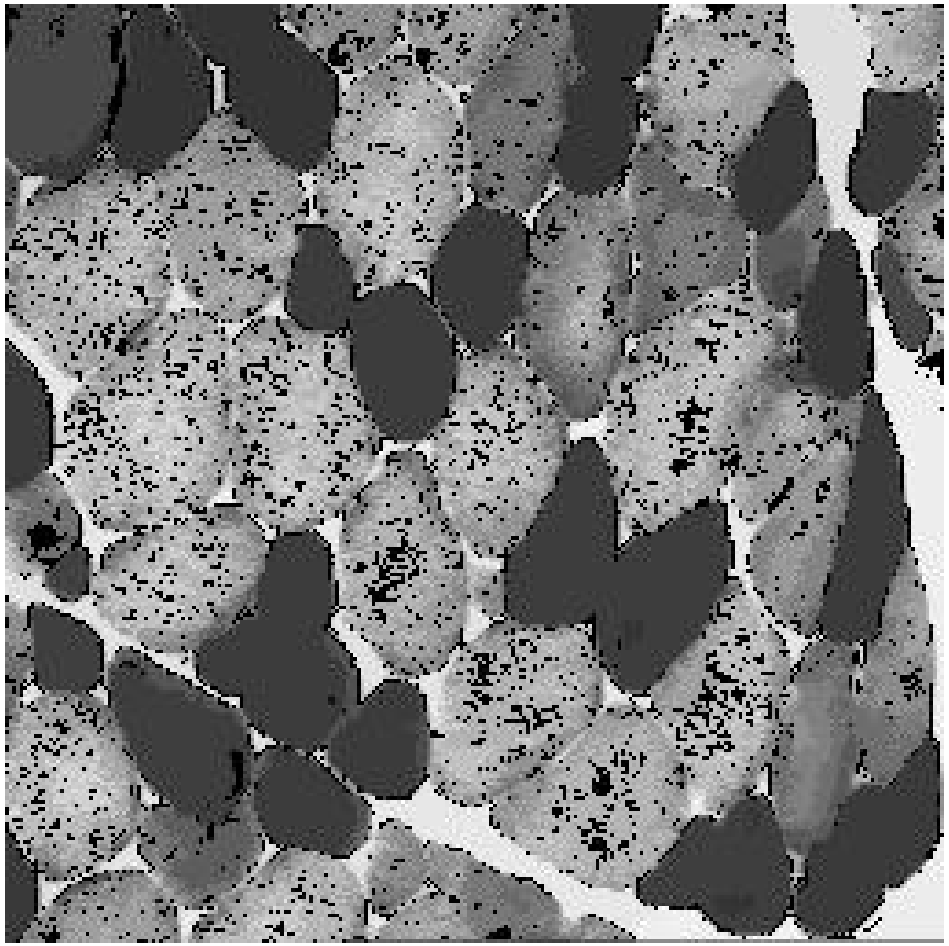


FIGURE – Seuil croissance à 50 (5966 pixels non segmentés)

Croissance de régions

Exemples

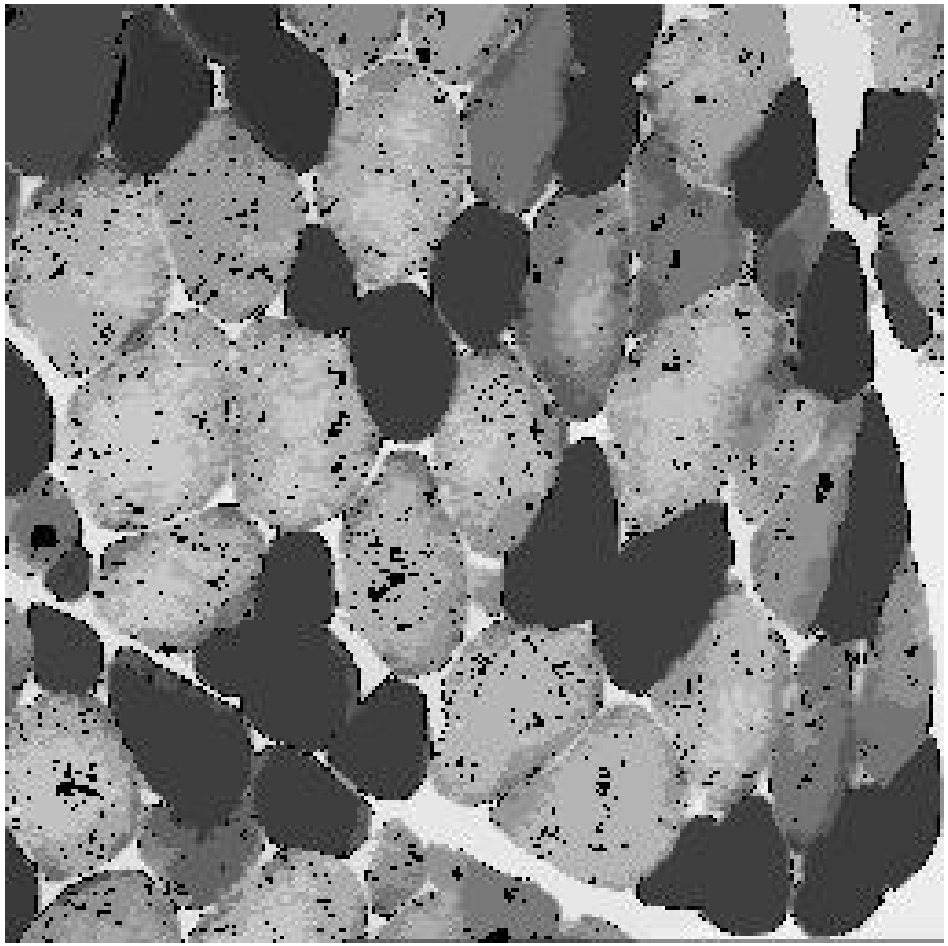


FIGURE – Seuil croissance à 100 (2885 pixels non segmentés)

Croissance de régions

Exemples

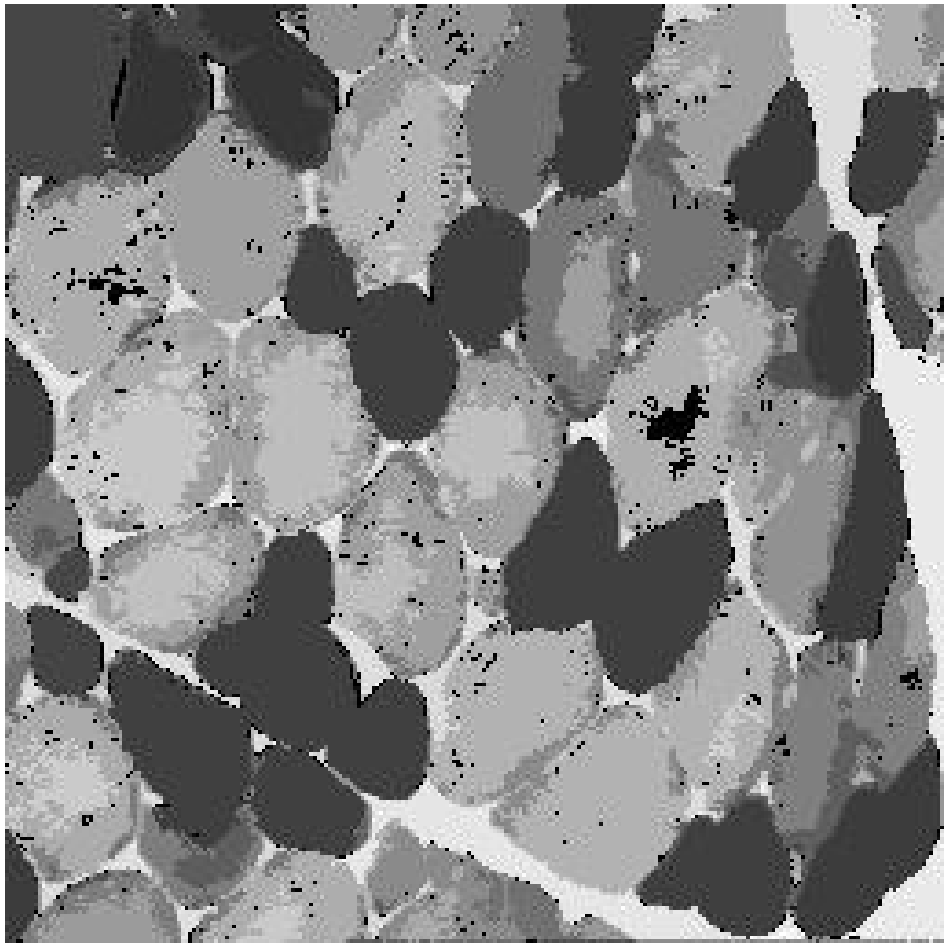


FIGURE – Seuil croissance à 200 (1381 pixels non segmentés)

Croissance de régions

Exemples

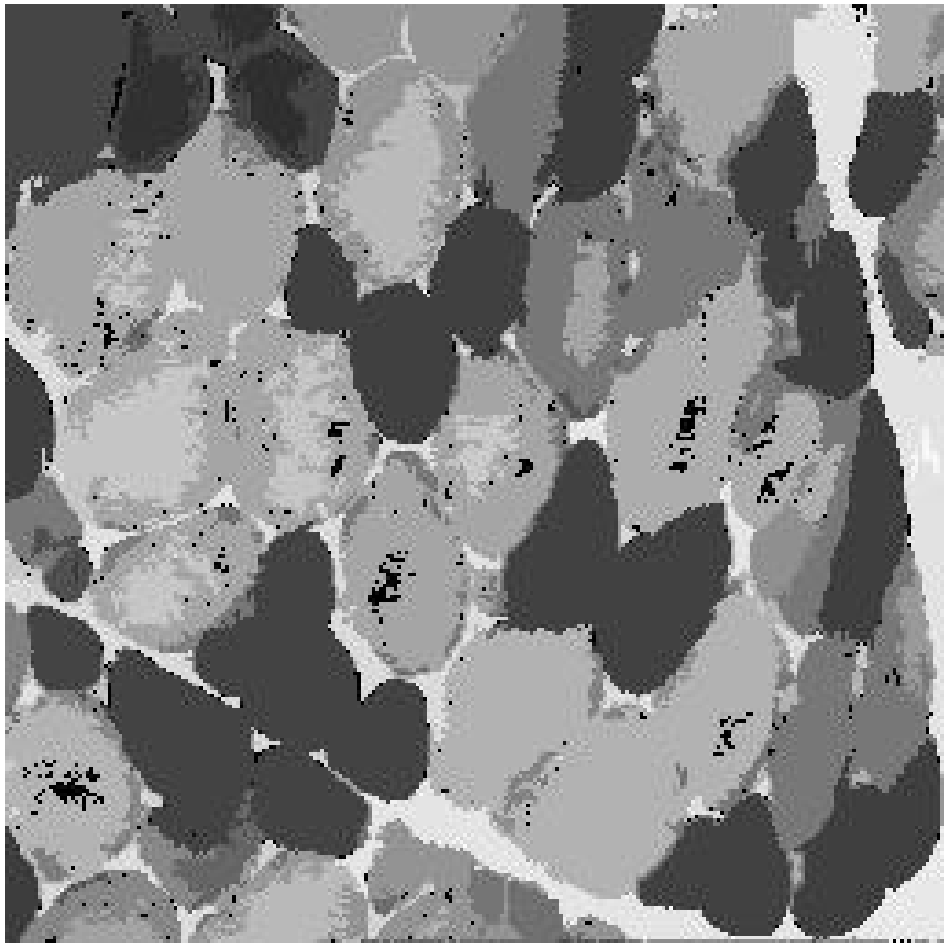


FIGURE – Seuil croissance à 300 (897 pixels non segmentés)

Croissance de régions

Exemples

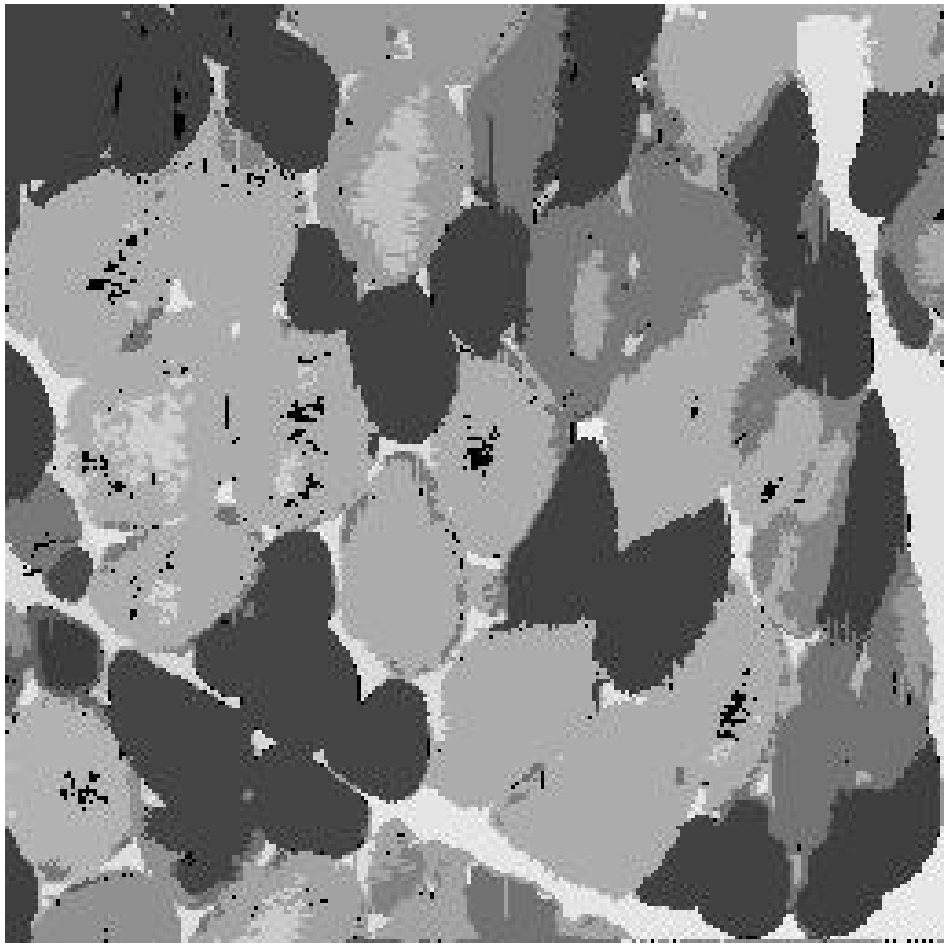


FIGURE – Seuil croissance à 400 (759 pixels non segmentés)

Croissance de régions

Exemples

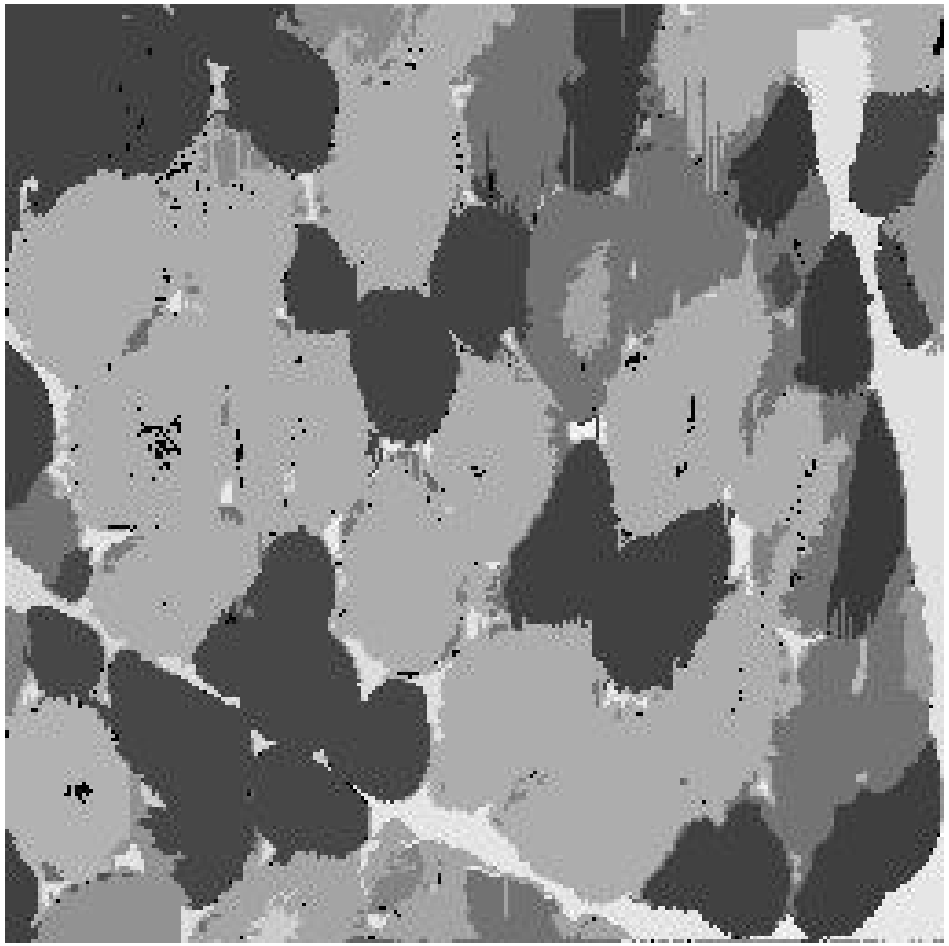


FIGURE – Seuil croissance à 500 (401 pixels non segmentés)

Croissance de régions

Exemples

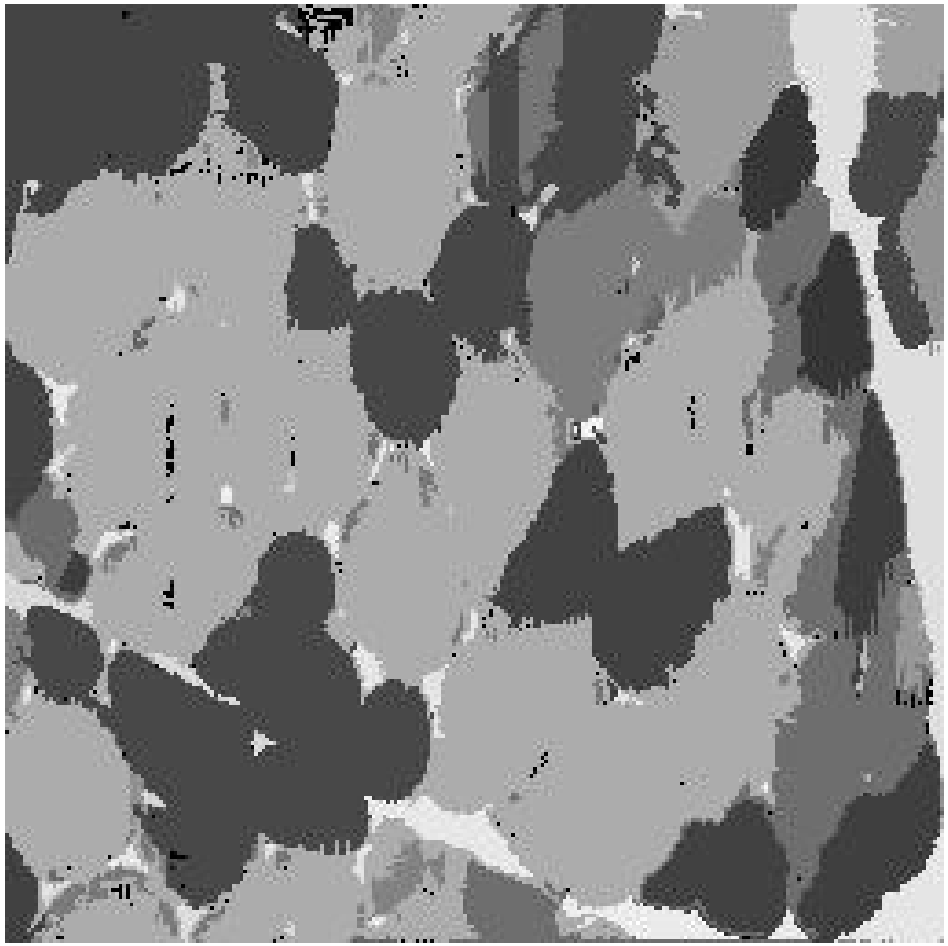


FIGURE – Seuil croissance à 600 (354 pixels non segmentés)

Croissance de régions

Exemples

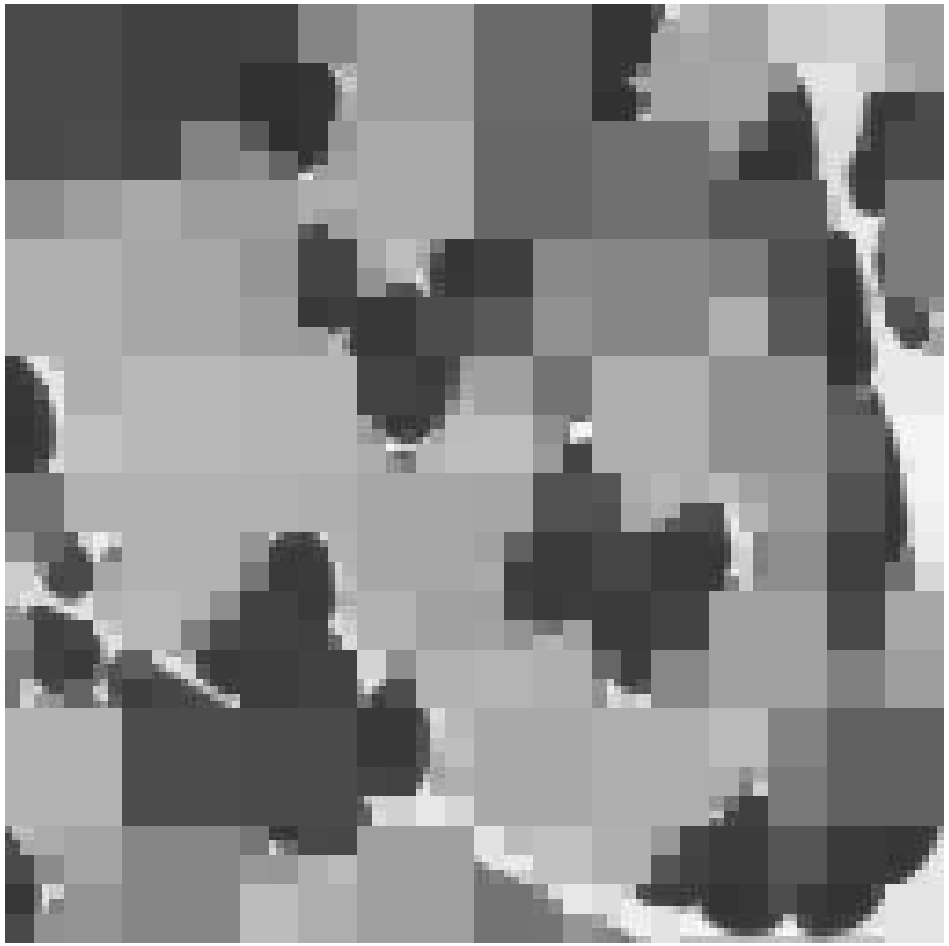


FIGURE – Initialisation : découpage à $\sigma = 1600$

Croissance de régions

Exemples

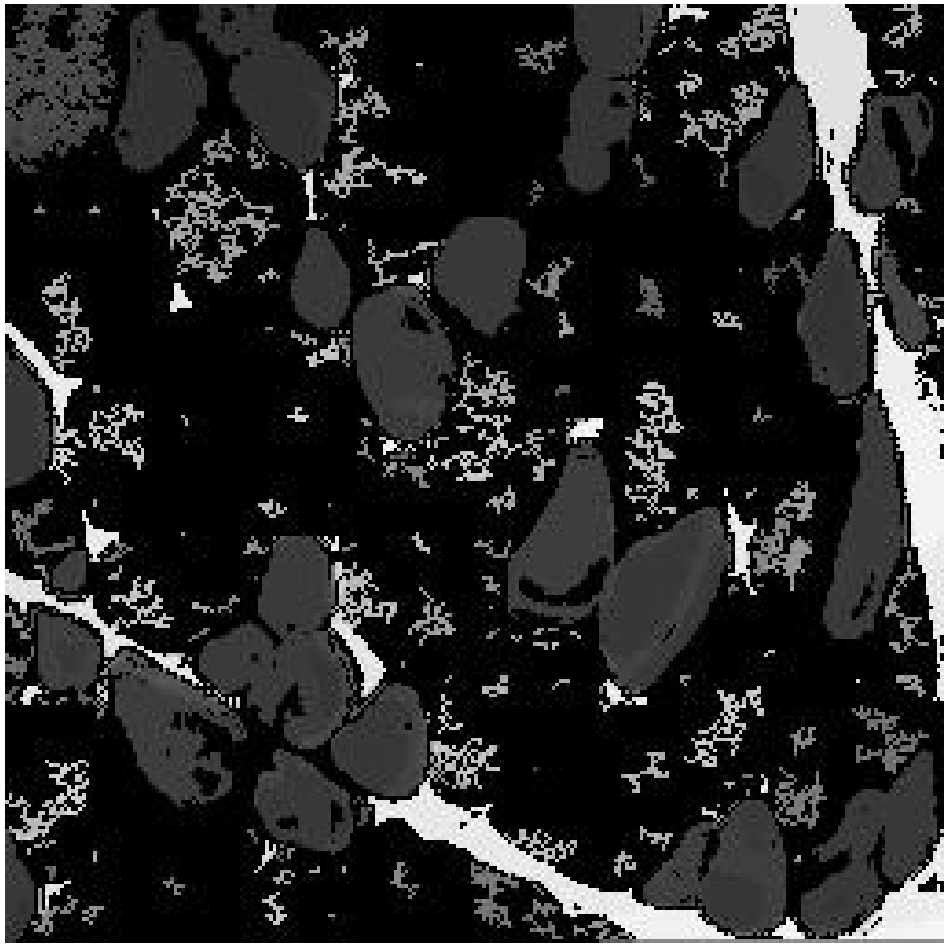


FIGURE – Seuil croissance à 10 (42301 pixels non segmentés)

Croissance de régions

Exemples

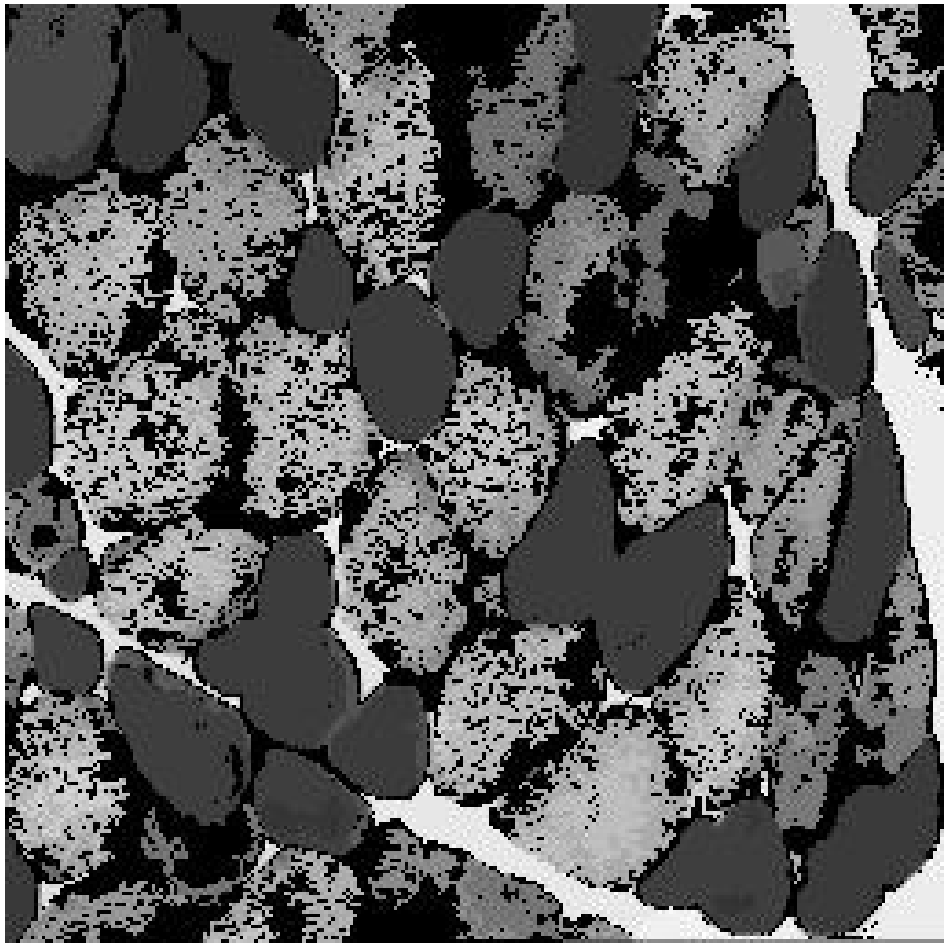


FIGURE – Seuil croissance à 50 (18940 pixels non segmentés)

Croissance de régions

Exemples

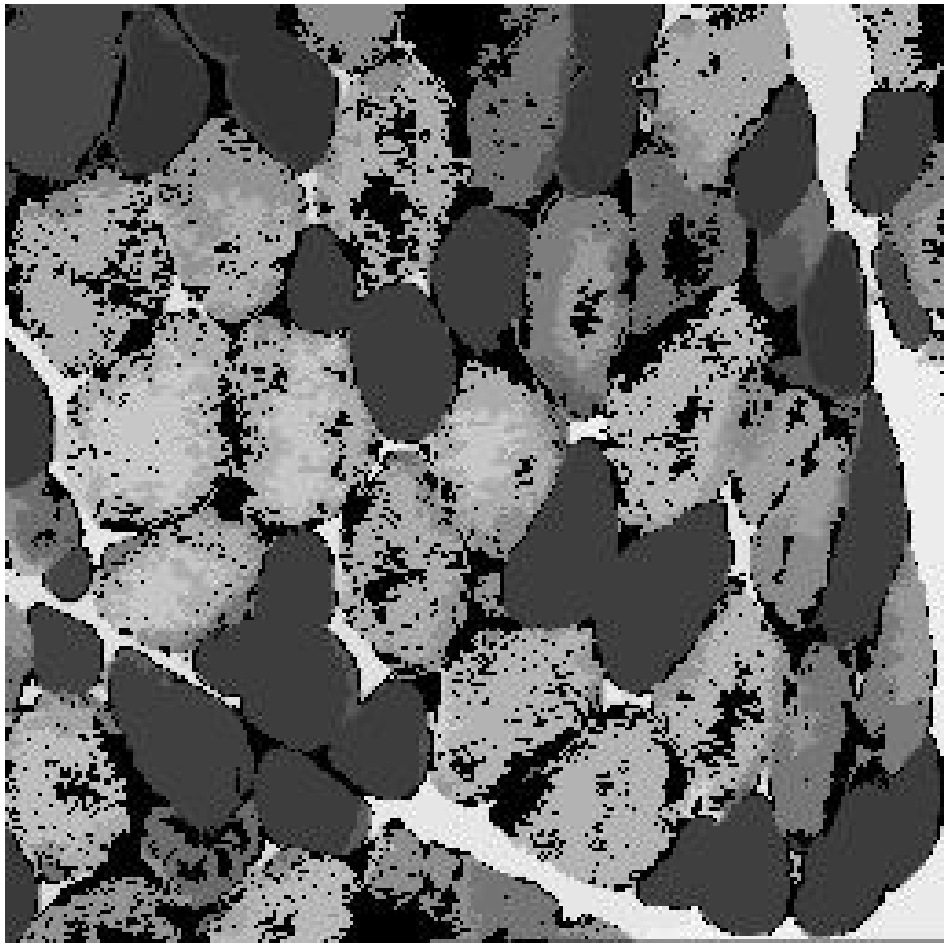


FIGURE – Seuil croissance à 100 (11654 pixels non segmentés)

Croissance de régions

Exemples

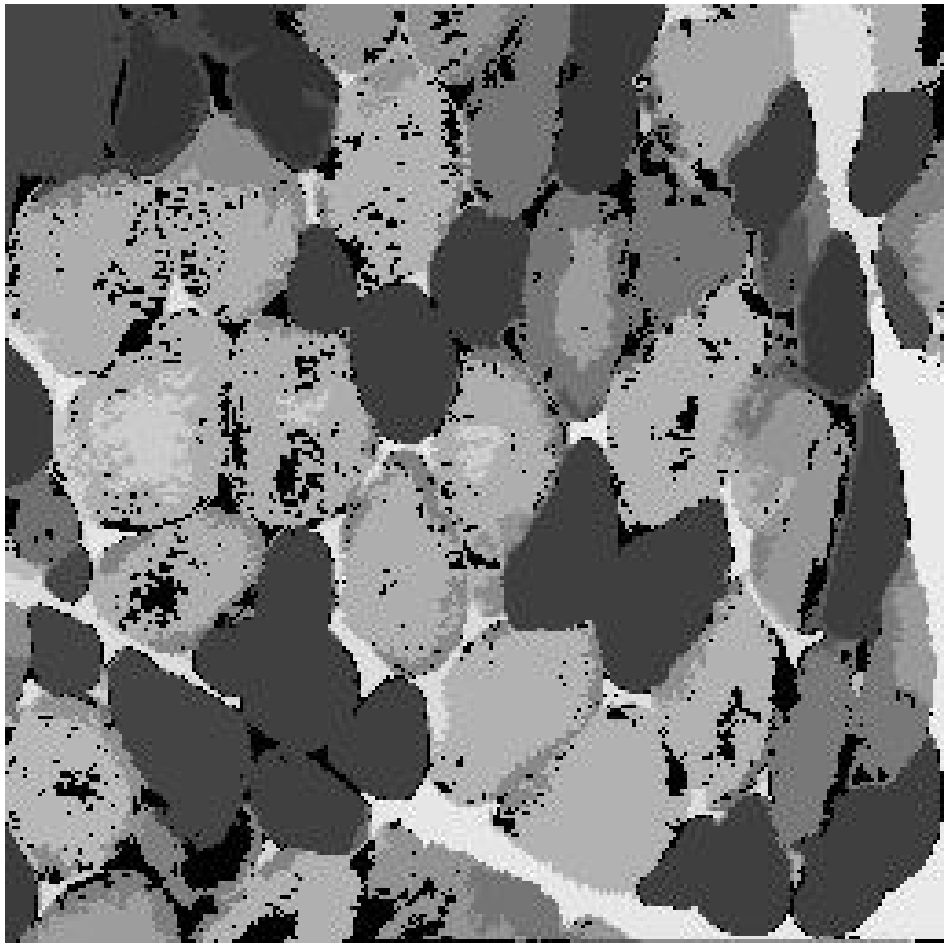


FIGURE – Seuil croissance à 200 (5592 pixels non segmentés)

Croissance de régions

Exemples

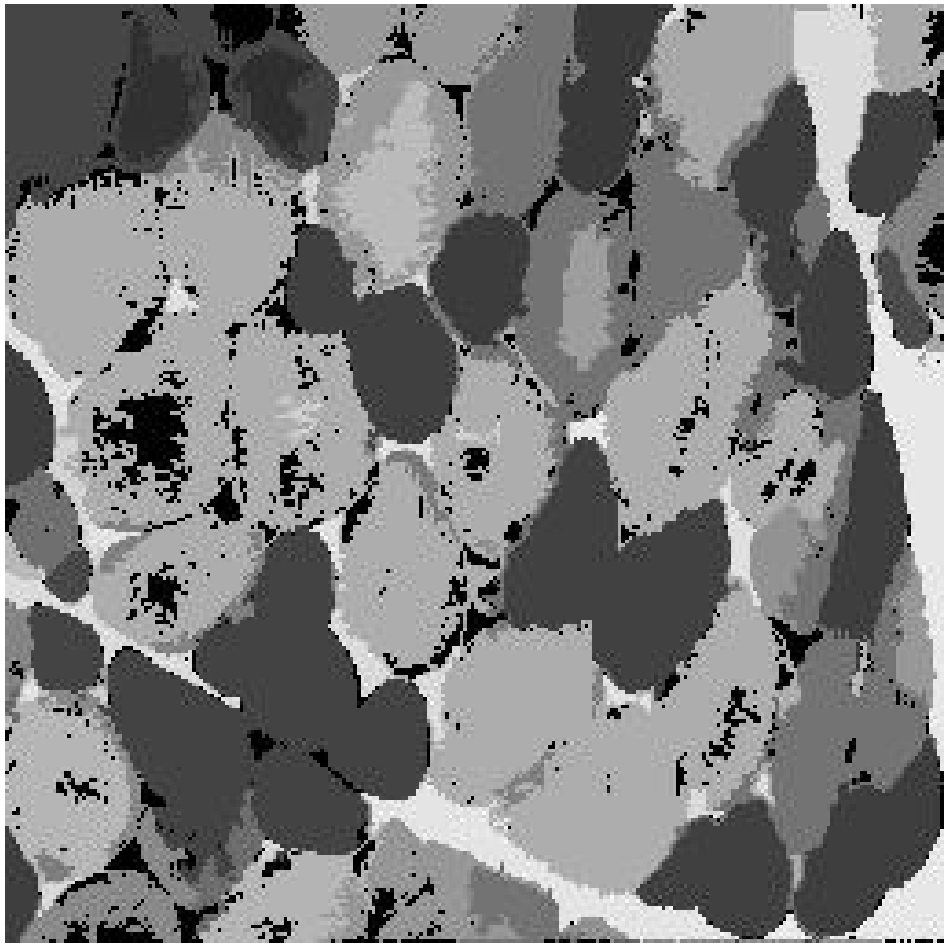


FIGURE – Seuil croissance à 300 (4071 pixels non segmentés)

Croissance de régions

Exemples

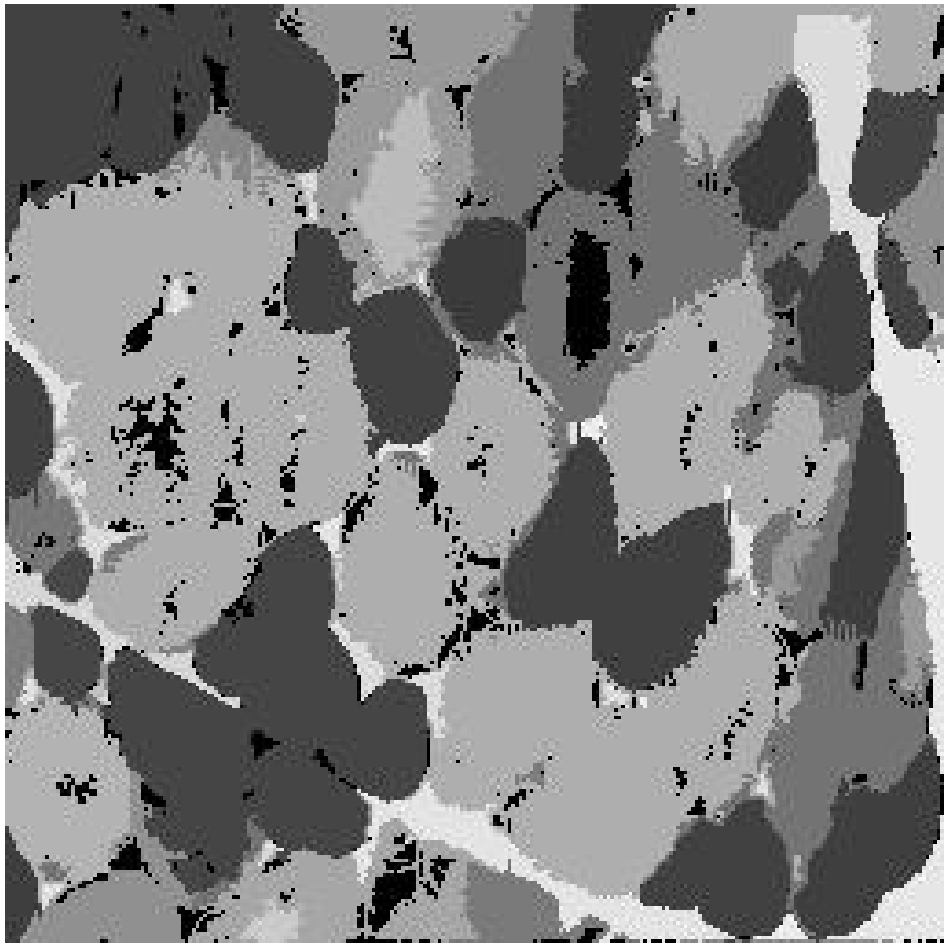


FIGURE – Seuil croissance à 400 (2868 pixels non segmentés)

Croissance de régions

Exemples

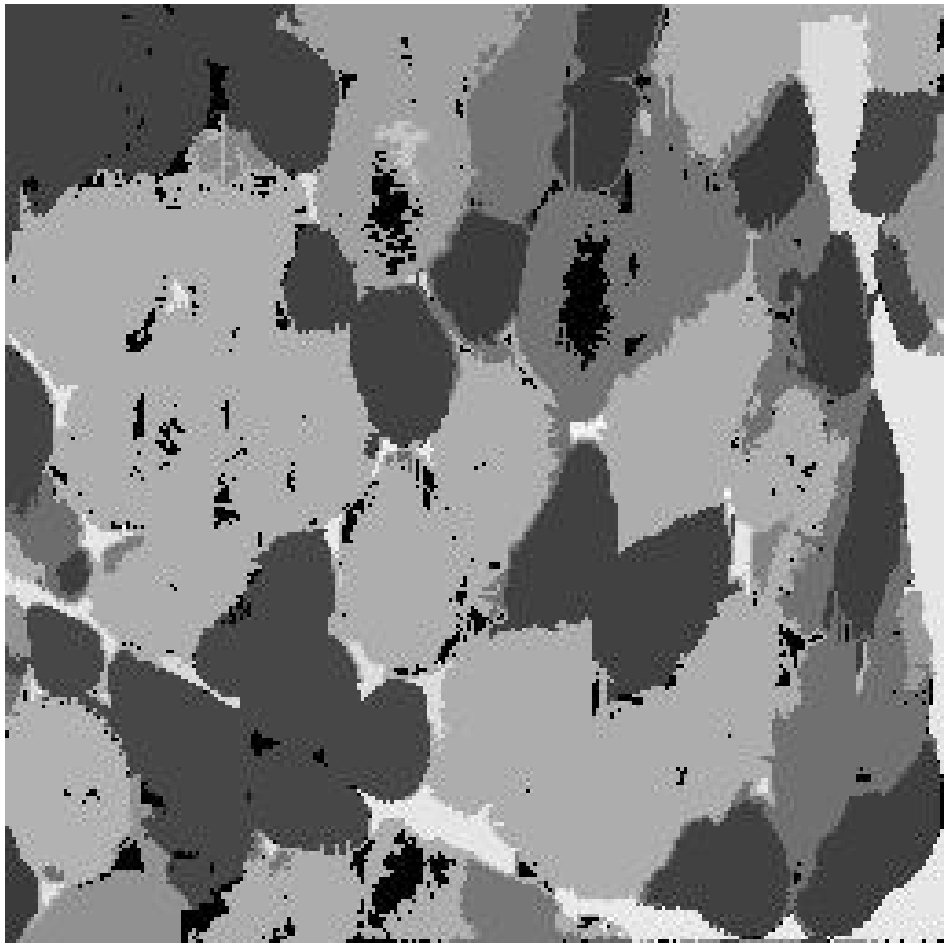


FIGURE – Seuil croissance à 500 (2289 pixels non segmentés)

Croissance de régions

Exemples

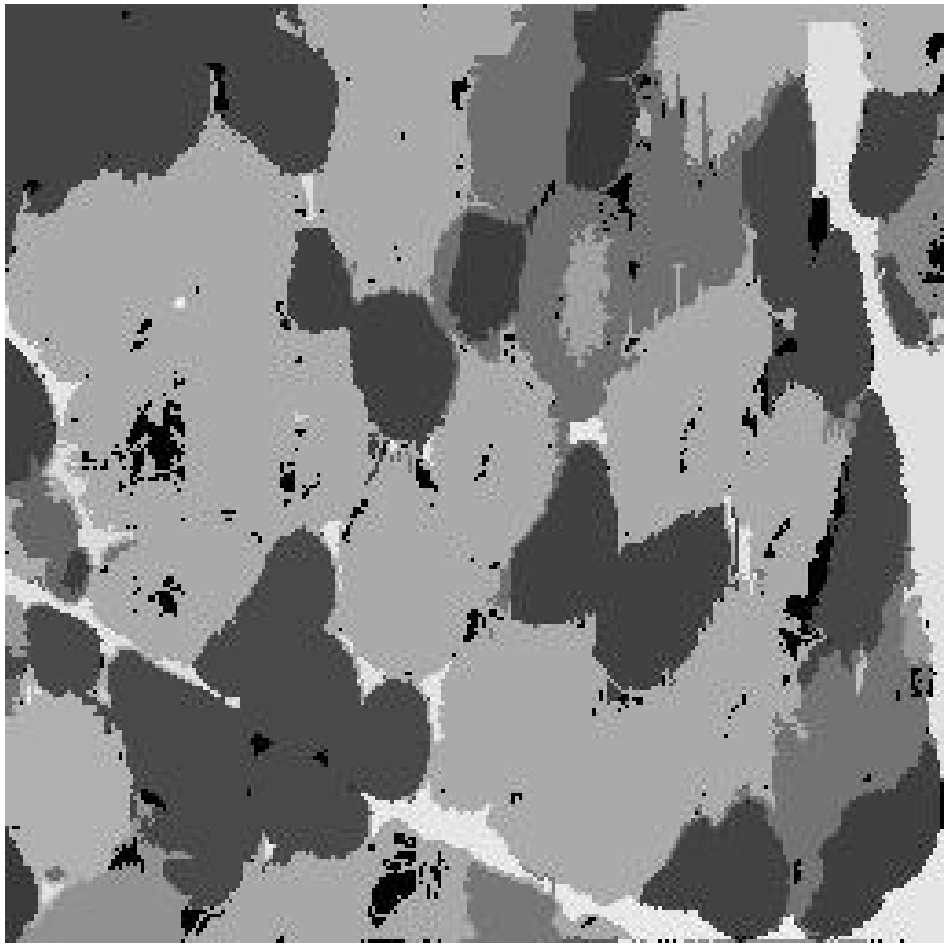
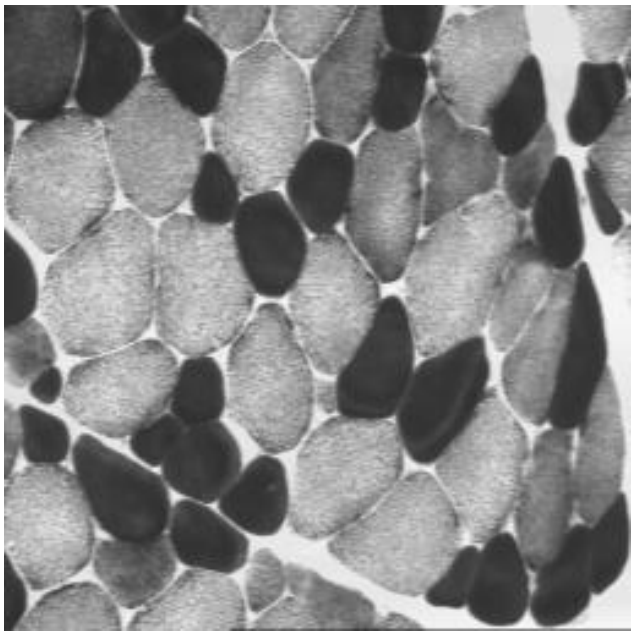
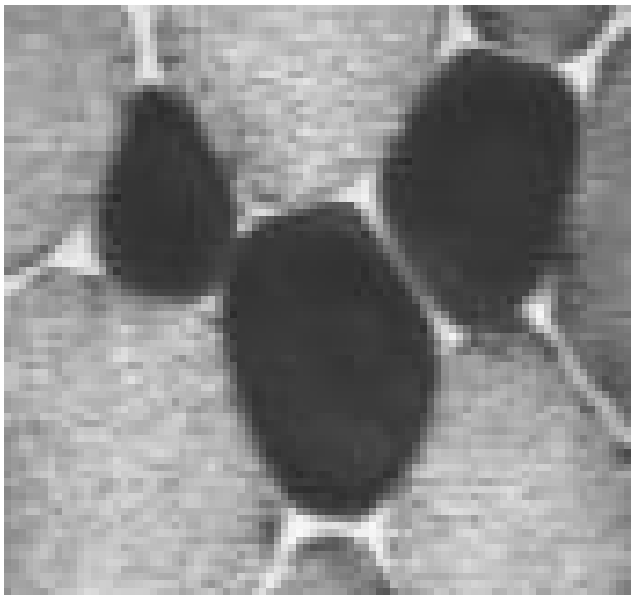


FIGURE – Seuil croissance à 600 (1661 pixels non segmentés)

Ligne de partage des eaux

- On peut représenter une image I sous la forme du graphe de la fonction $(x, y) \mapsto I(x, y)$.
- Un niveau de gris = une altitude.

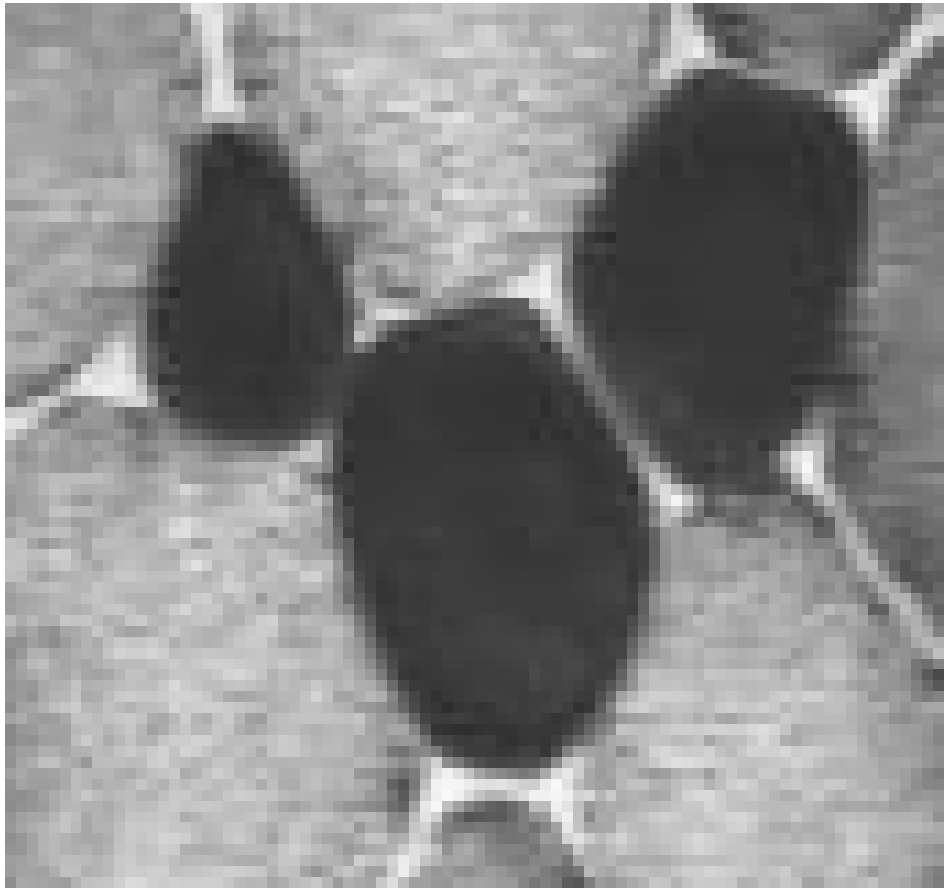




...

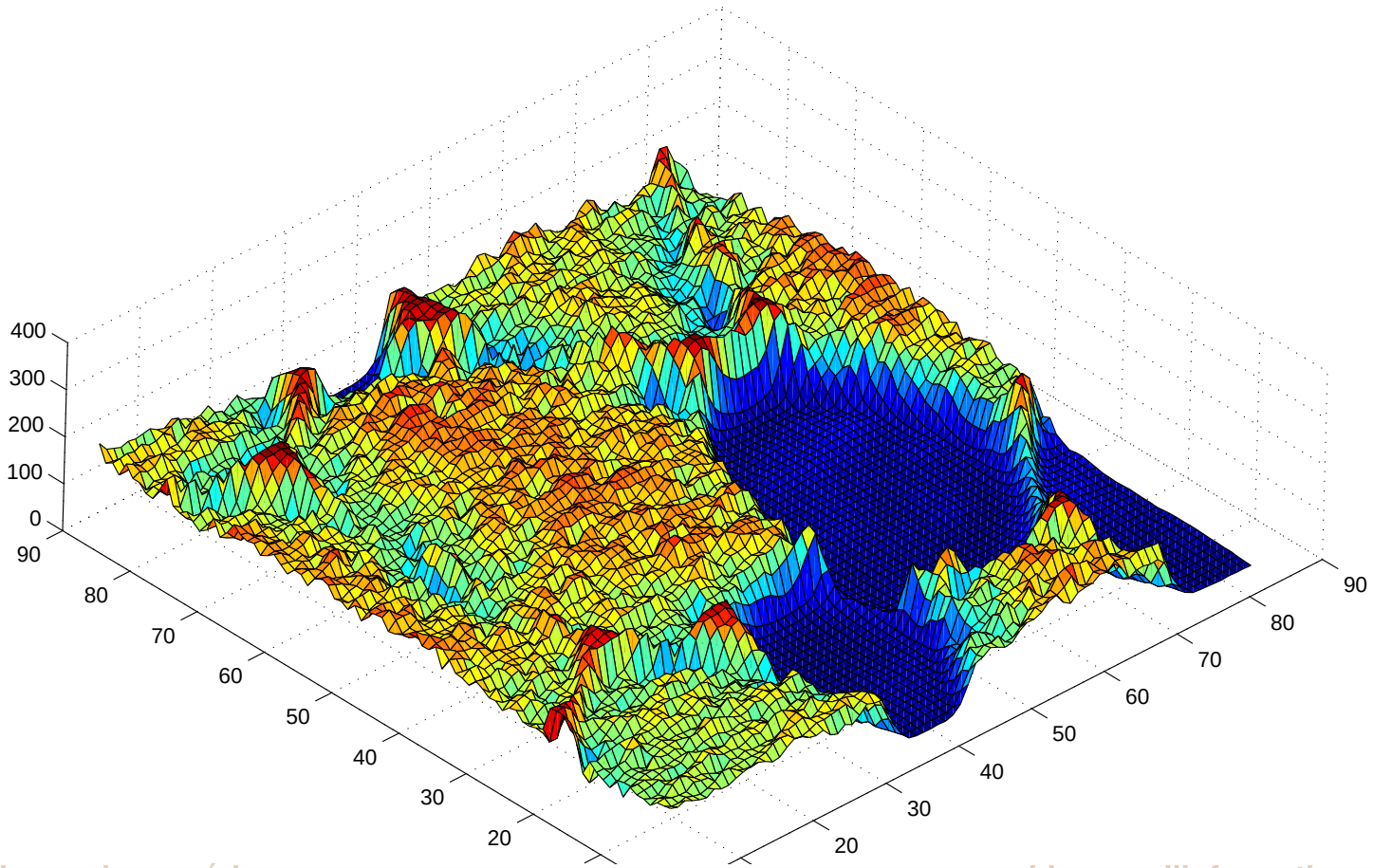
Ligne de partage des eaux

- On peut représenter une image I sous la forme du graphe de la fonction $(x, y) \mapsto I(x, y)$.
- Un niveau de gris = une altitude.



Ligne de partage des eaux

- On peut représenter une image I sous la forme du graphe de la fonction $(x, y) \mapsto I(x, y)$.
- Un niveau de gris = une altitude.



Ligne de partage des eaux

- Analogie avec la géographie : les zones homogènes correspondent à des vallées ou à des plateaux.
- Les régions sont bordées de contours : si on travaille sur des images de norme de gradient, les zones homogènes sont des vallées et les contours des lignes de crêtes.

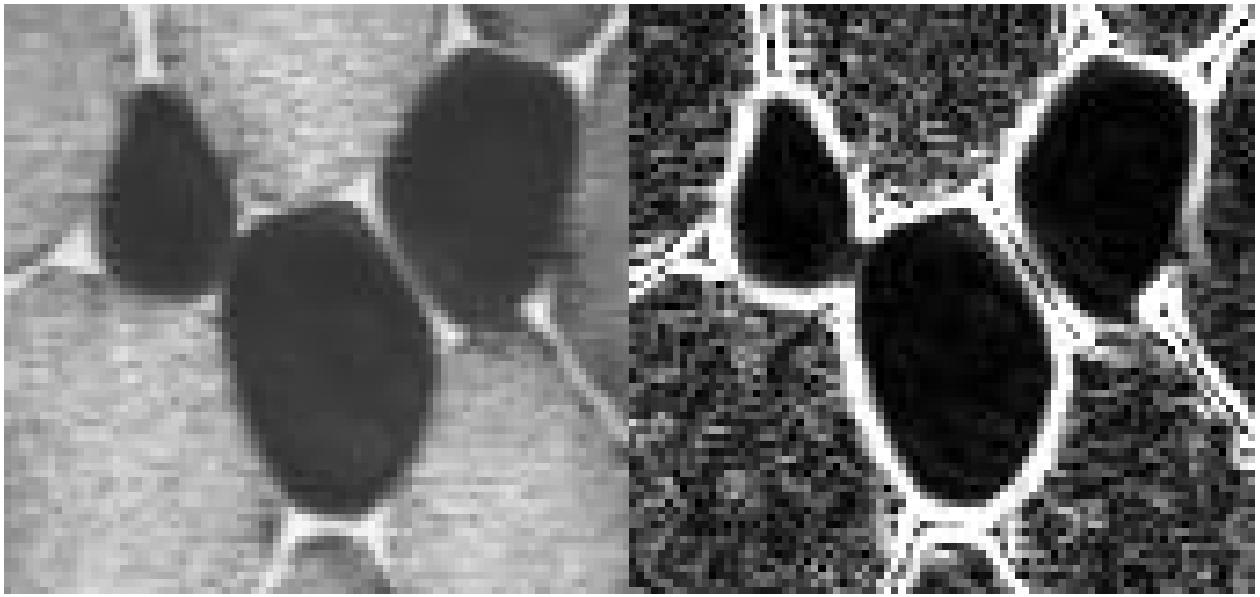
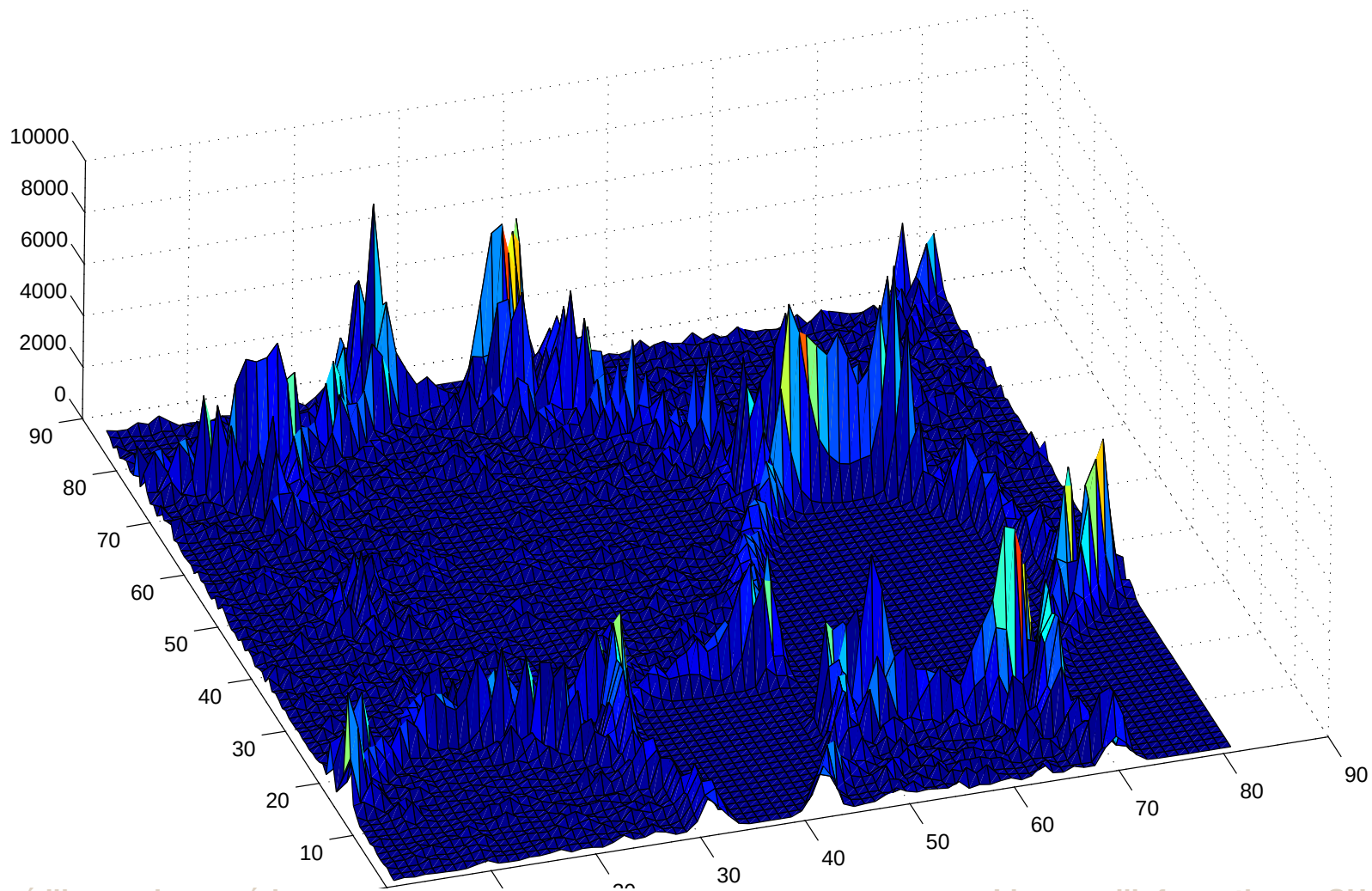


FIGURE – Zone d'intérêt - Image de la norme du gradient

Ligne de partage des eaux

Vue 3D de la norme du gradient



Ligne de partage des eaux

Principe

- Le principe est donc simple : segmenter les régions situées de part et d'autre d'une ligne de crête.
- Ces régions s'appellent des bassins versants.
- Une ligne de crête s'appelle également une ligne de partage des eaux (*watershed*) car c'est elle qui décide dans quel bassin versant ira la pluie.
- En imaginant un déluge, les bassins versants se remplissent et forment les régions segmentées.
- Comment simuler ce remplissage ? Suivre le trajet de l'eau le long des lignes de gradients n'est pas réalisable car c'est coûteux et le calcul des gradients est trop imprécis et sensible au bruit, il génère des trajectoires trop imprécises.

Ligne de partage des eaux

Algorithme de l'immersion (Soille *et al* 1991)

1. L'idée est de plonger notre image 3D dans l'eau : les pixels les plus bas seront les premiers immergés et correspondent aux futures régions (*flooding*).
2. Au fur et à mesure qu'on immerge l'image :
   - les régions s'étendent,
   - de nouvelles régions apparaissent.
3. Lorsque des régions se rejoignent : les points de réunion marquent la ligne de partage des eaux et aussi la frontière entre régions adjacentes : ces frontières n'évoluent plus avec l'immersion.
4. L'algorithme s'arrête lorsque l'immersion est totale.

Ligne de partage des eaux

vue comme une croissance de régions

- L'algorithme de l'immersion est donc très similaire à la croissance de régions :
  - il s'applique sur des images pour lesquelles le niveau de gris peut être assimilé de façon pertinente à une altitude (norme de gradient, carte de distance aux contours).
  - il n'y a pas de critère de croissance : une région s'arrête de croître lorsqu'elle rencontre une autre région.

Ligne de partage des eaux

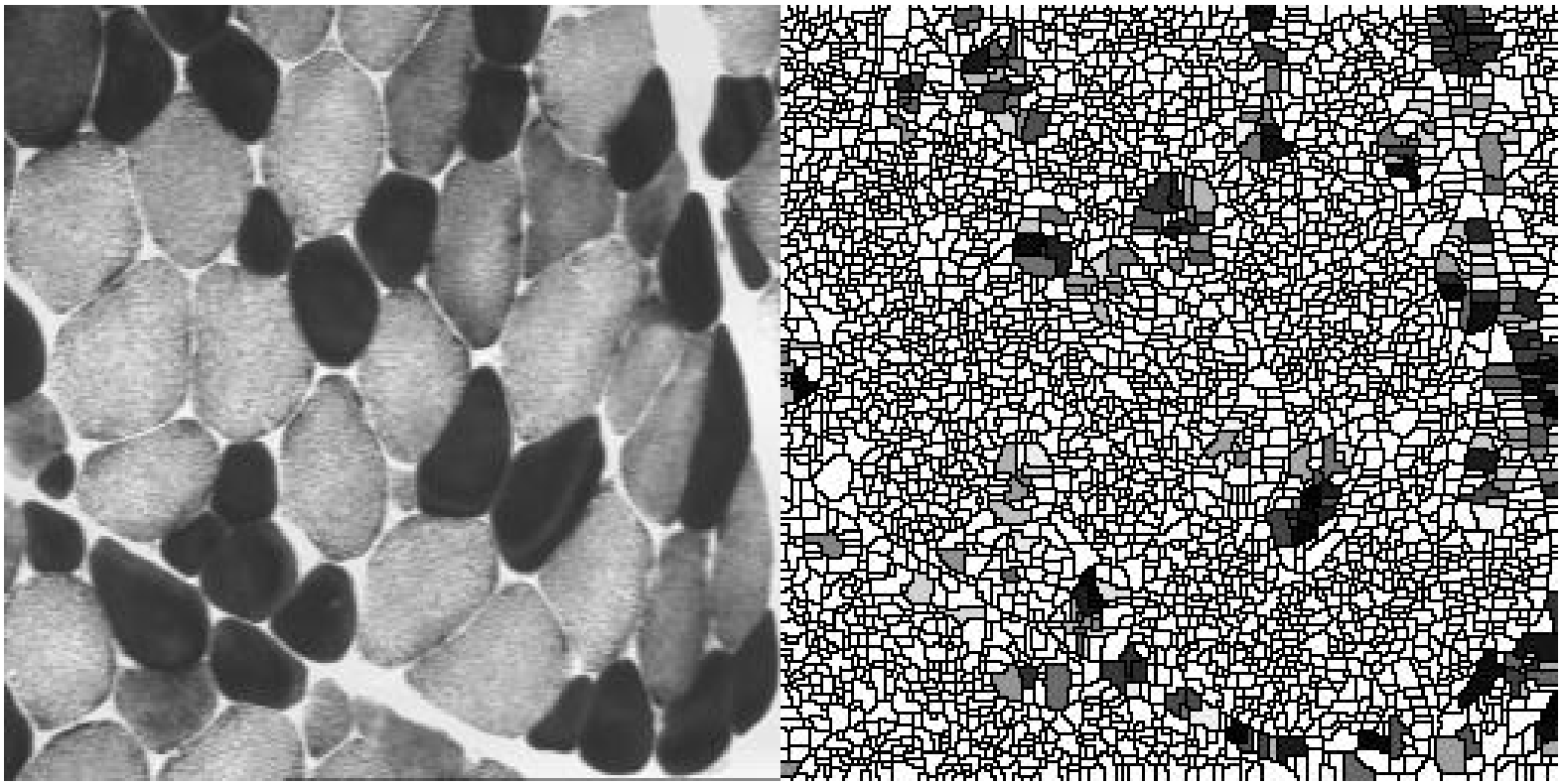
Algorithme

```
algo watershed( G:image gradient, LR:liste regions)
    LR :=[];
    pour niveau variant de 0 a max faire
      pour R variant dans LR faire
        faire croitre R jusqu'a l'altitude niveau;
      fin pour
      pour chaque pixel p tq G(p) = niveau faire
        si p n'est pas dans LR alors
          R := [p];
          LR:=[LR,R]
          faire croitre R jusqu'a l'altitude niveau;
        fin si
      fin pour
    fin pour
fin algo
```

Ligne de partage des eaux

Exemples

- L'algorithme est non supervisé : on fixe seulement un système de voisinage.
- Pas de commande Inrimage, on a utilisé Matlab, (on peut aussi utiliser OpenCV) ...



Ligne de partage des eaux

Sur-segmentation

- Le principal inconvénient : conduit à une sur-segmentation.
- L'algorithme est très sensible au bruit :
  - un bruit provoque induit des extrema locaux dans l'image donc des minima locaux dans la norme du gradient
  - autant de régions initiales apparaissent lors de l'immersion : on peut le constater sur l'image 3D des gradients (voir page 95).
- Remèdes :
  - Bien choisir les points d'immersion (ne doivent plus être des minima locaux) : on parle de marqueurs. Le procédé devient supervisé.

Ligne de partage des eaux

Sur-segmentation

- Le principal inconvénient : conduit à une sur-segmentation.
- L'algorithme est très sensible au bruit :
  - un bruit provoque induit des extrema locaux dans l'image donc des minima locaux dans la norme du gradient
  - autant de régions initiales apparaissent lors de l'immersion : on peut le constater sur l'image 3D des gradients (voir page 95).
- Remèdes :
  - Bien choisir les points d'immersion (ne doivent plus être des minima locaux) : on parle de marqueurs. Le procédé devient supervisé.
  - Éliminer le bruit en ...

Ligne de partage des eaux

Sur-segmentation

- Le principal inconvénient : conduit à une sur-segmentation.
- L'algorithme est très sensible au bruit :
  - un bruit provoque induit des extrema locaux dans l'image donc des minima locaux dans la norme du gradient
  - autant de régions initiales apparaissent lors de l'immersion : on peut le constater sur l'image 3D des gradients (voir page 95).
- Remèdes :
  - Bien choisir les points d'immersion (ne doivent plus être des minima locaux) : on parle de marqueurs. Le procédé devient supervisé.
  - Éliminer le bruit en ... lissant l'image !

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

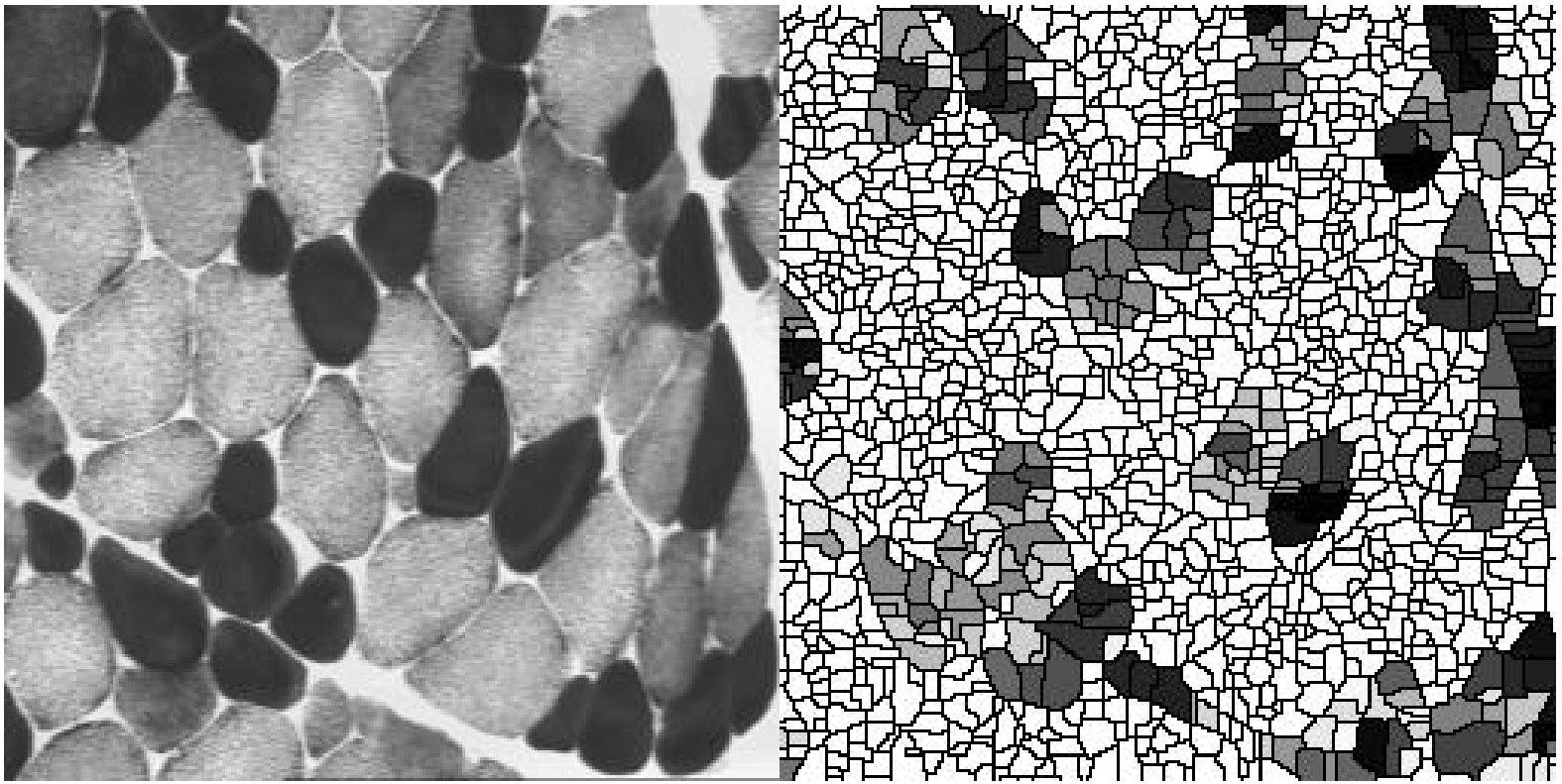


FIGURE – $\sigma = 1$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

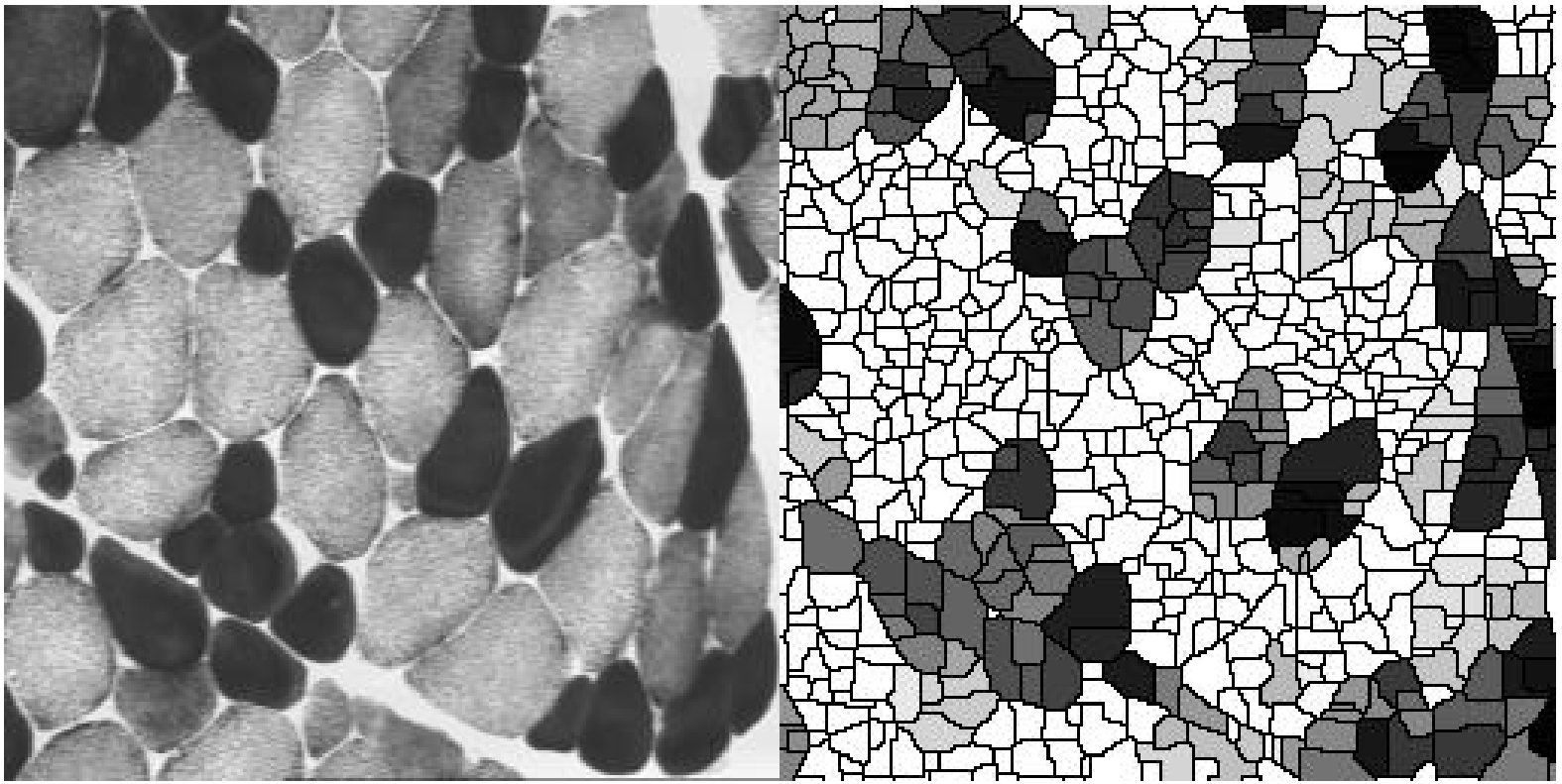


FIGURE – $\sigma = 1.5$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

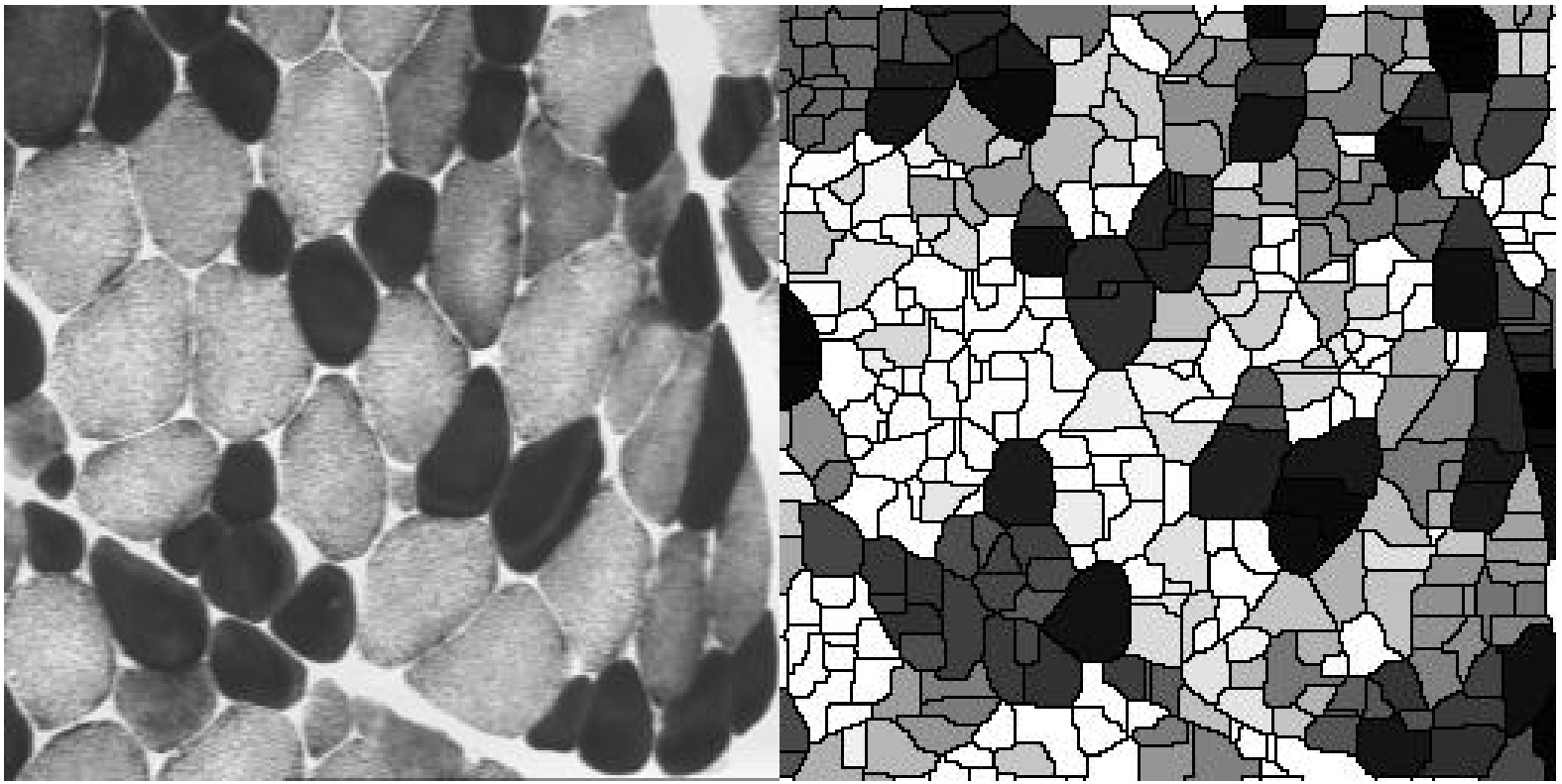


FIGURE – $\sigma = 2$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

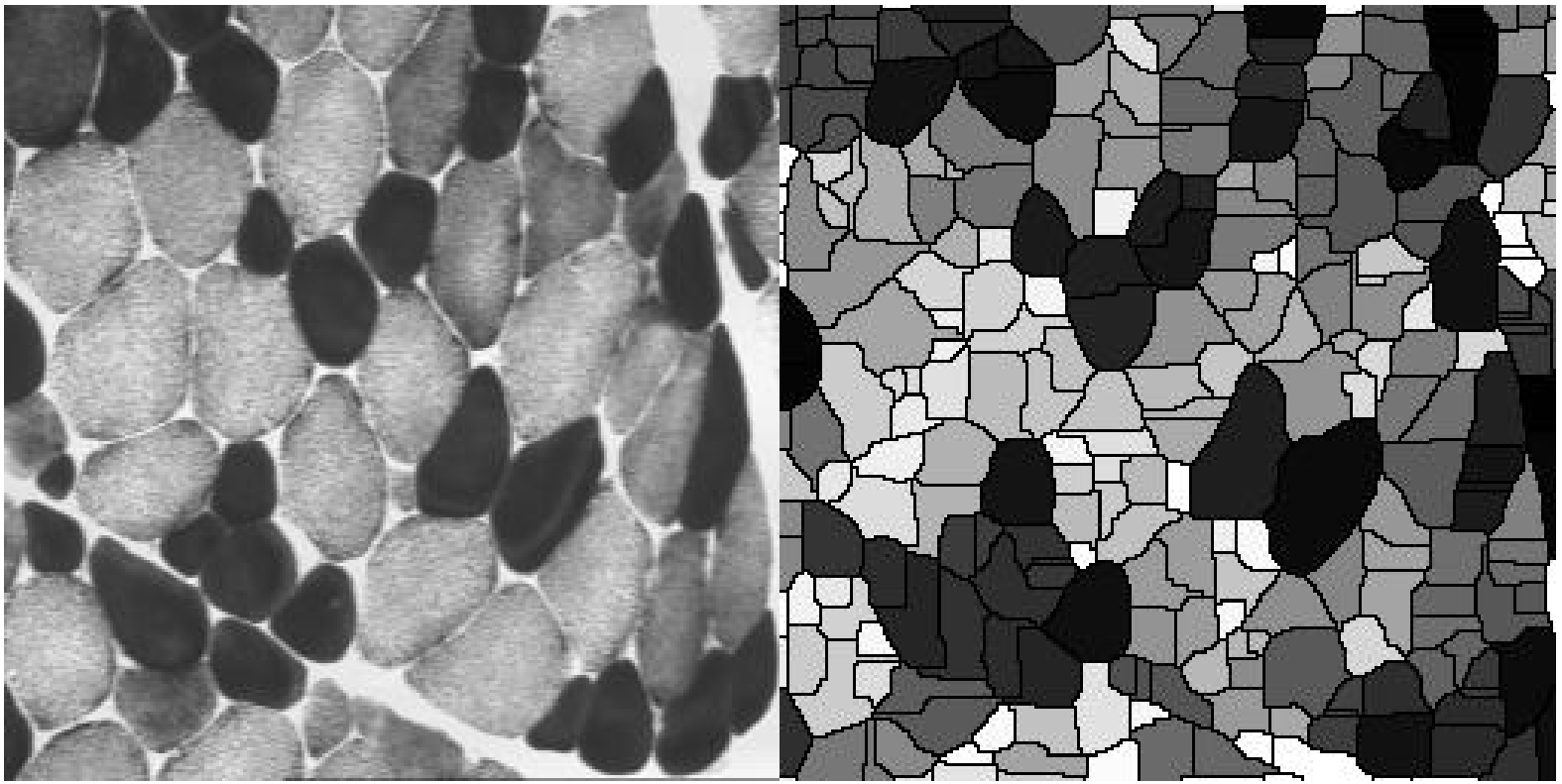


FIGURE – $\sigma = 2.5$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

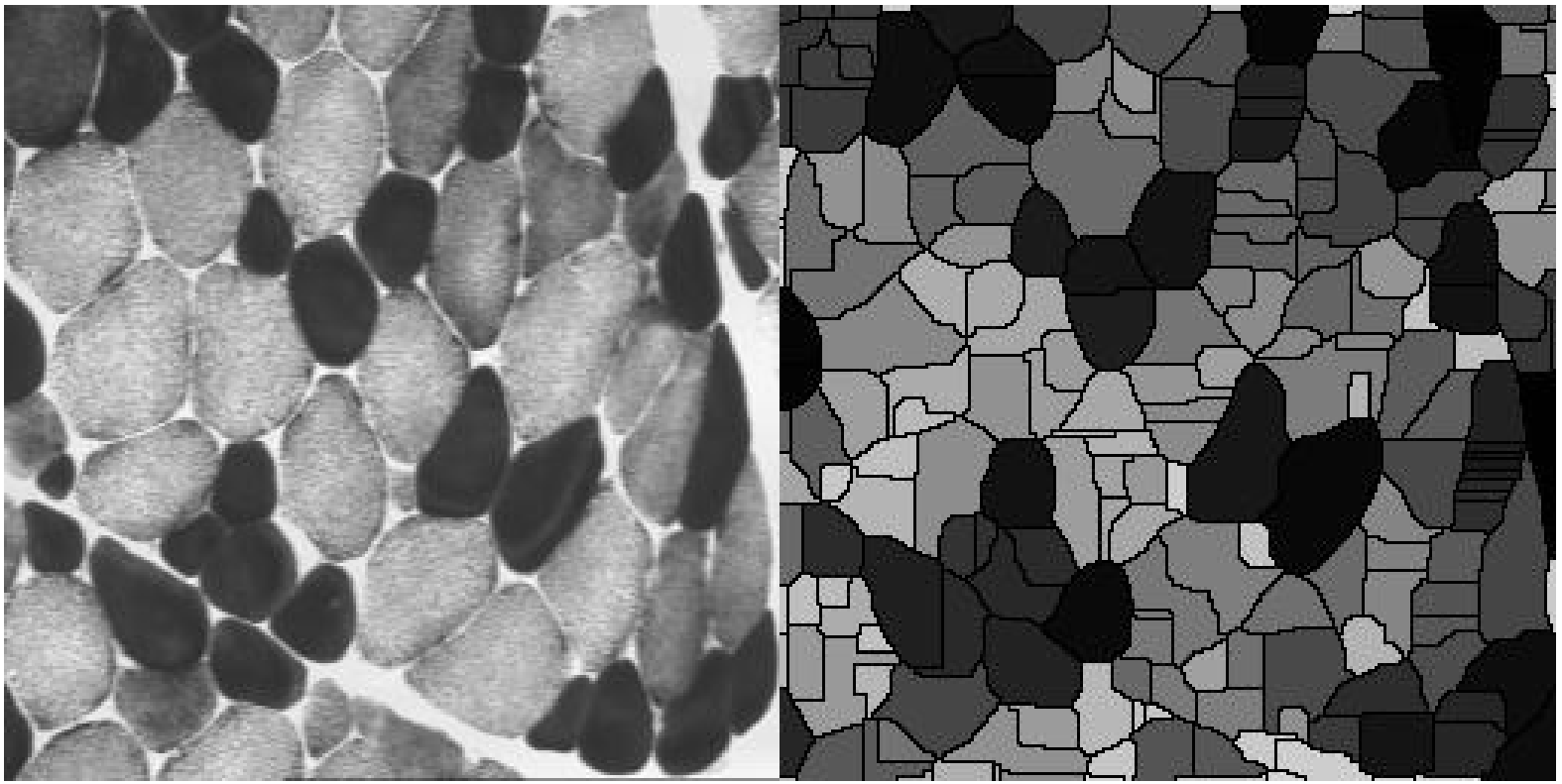


FIGURE – $\sigma = 3$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

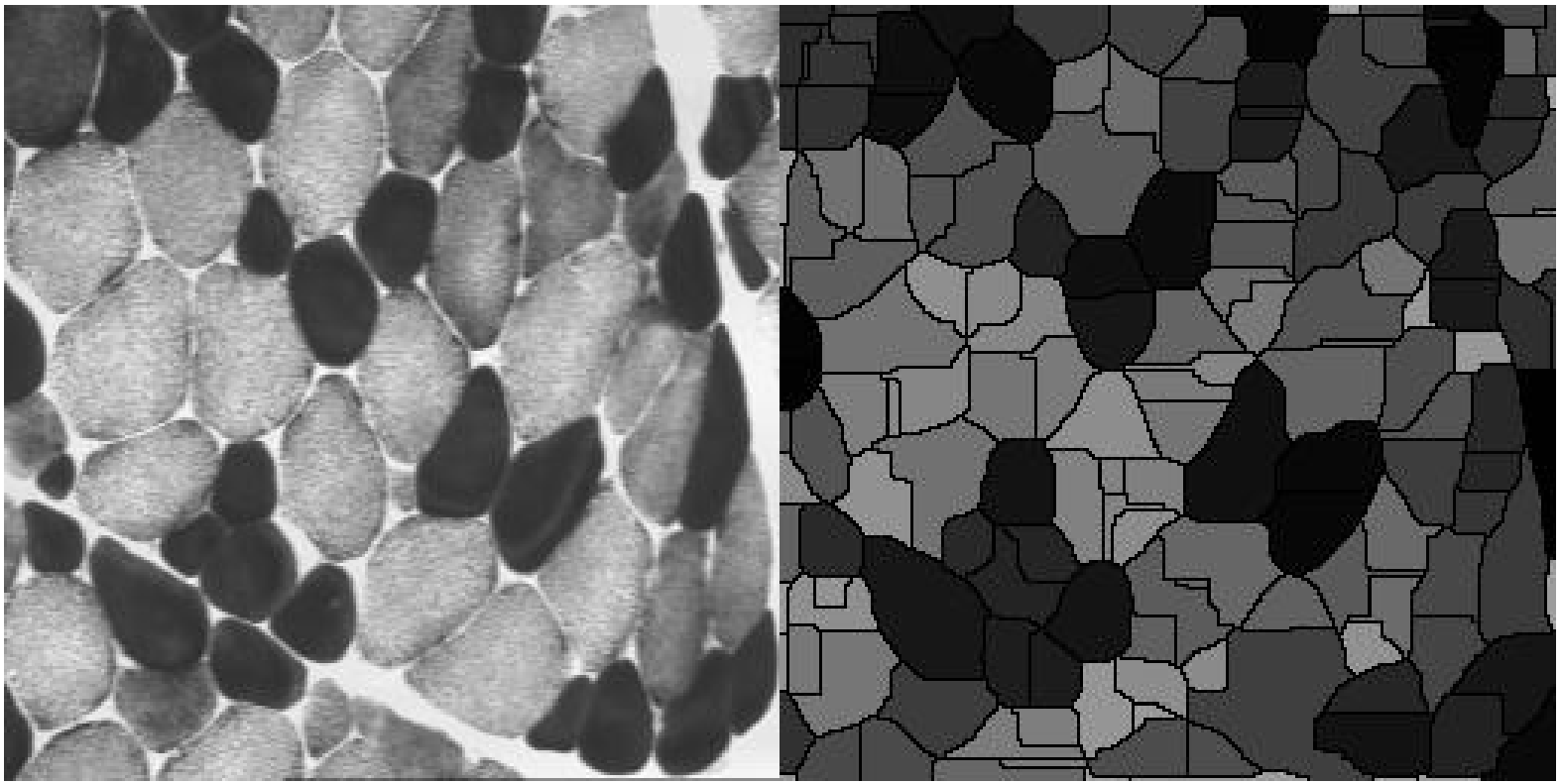


FIGURE – $\sigma = 3.5$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

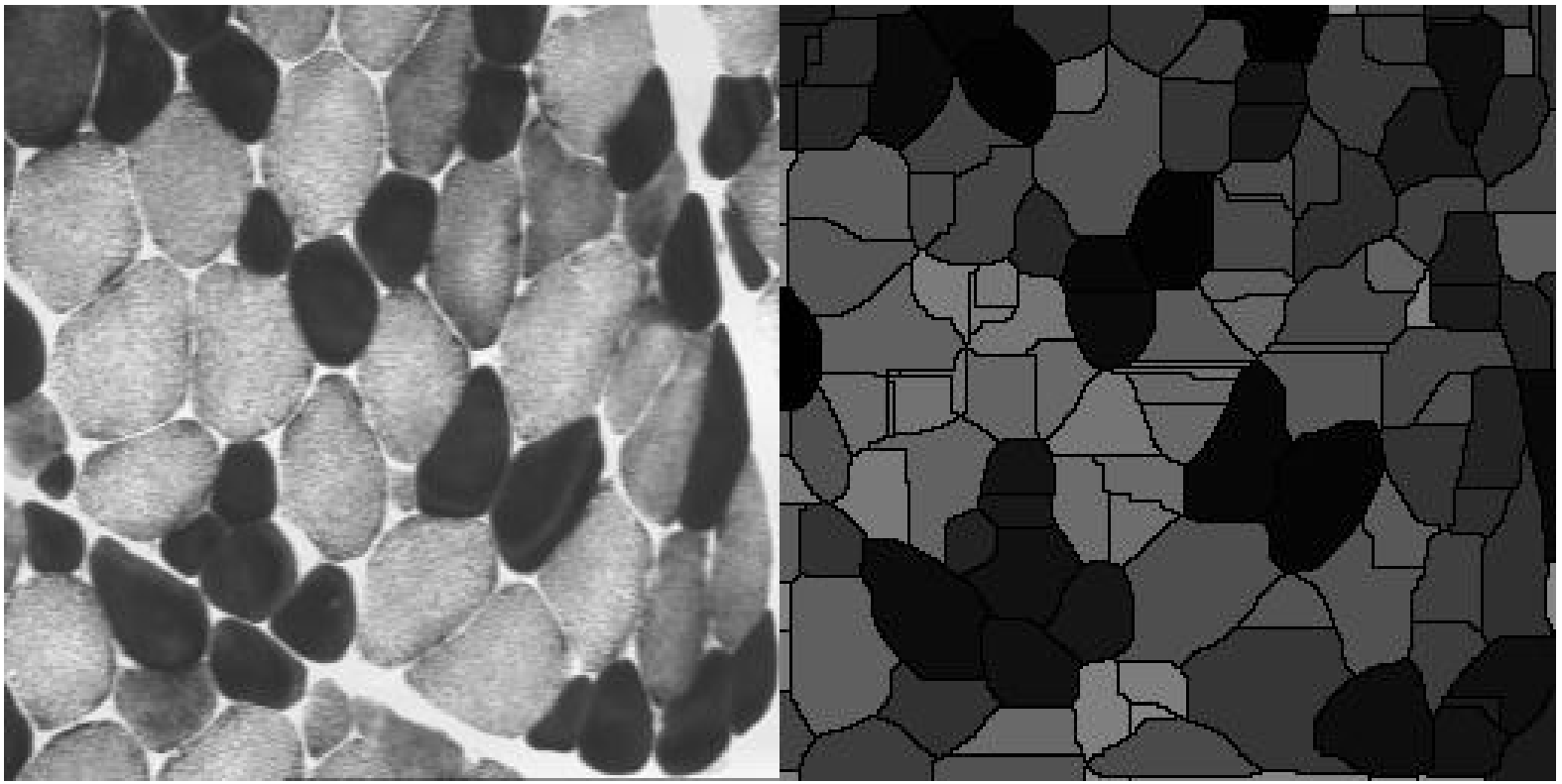


FIGURE – $\sigma = 4$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

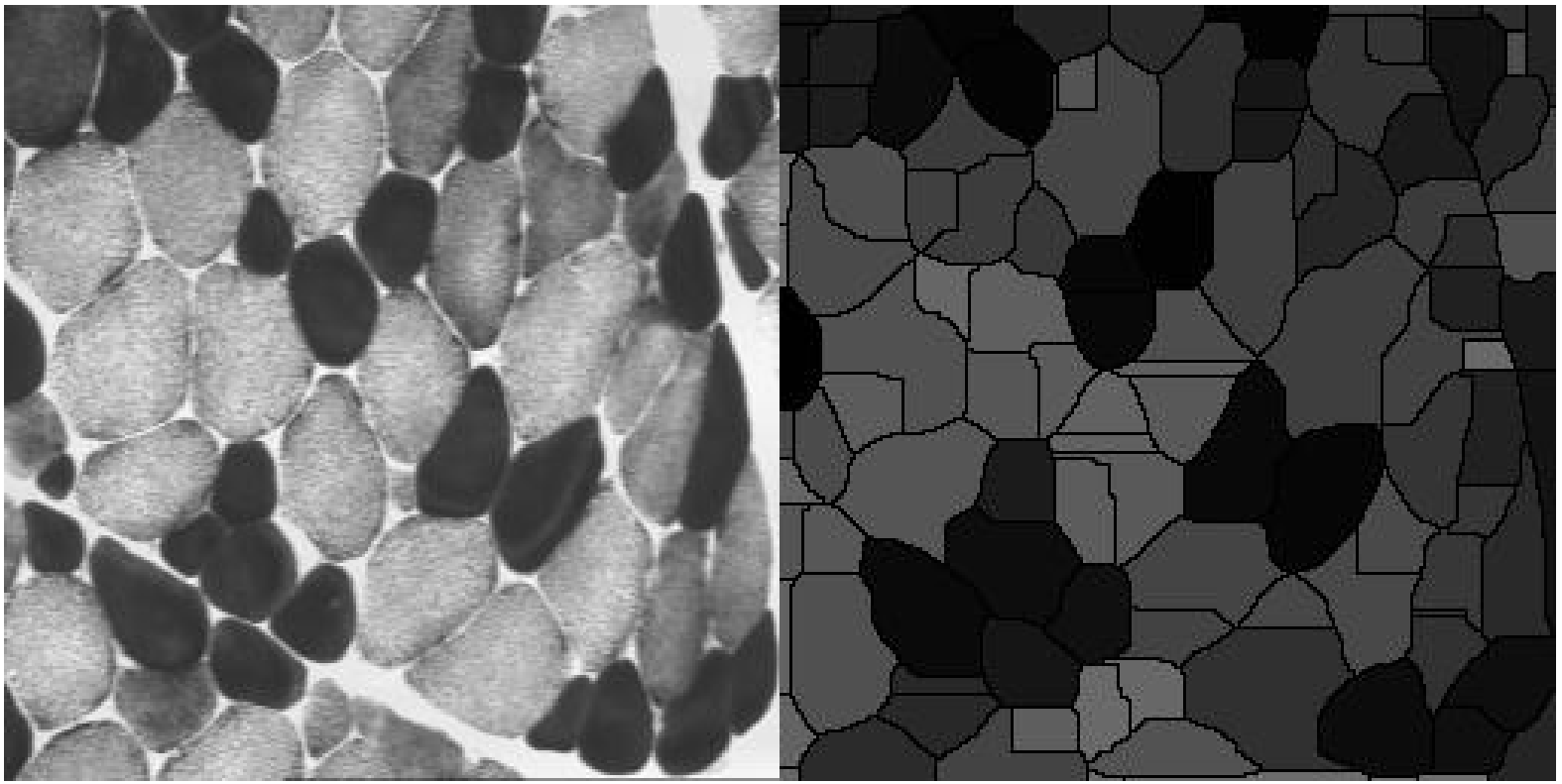


FIGURE – $\sigma = 4.5$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

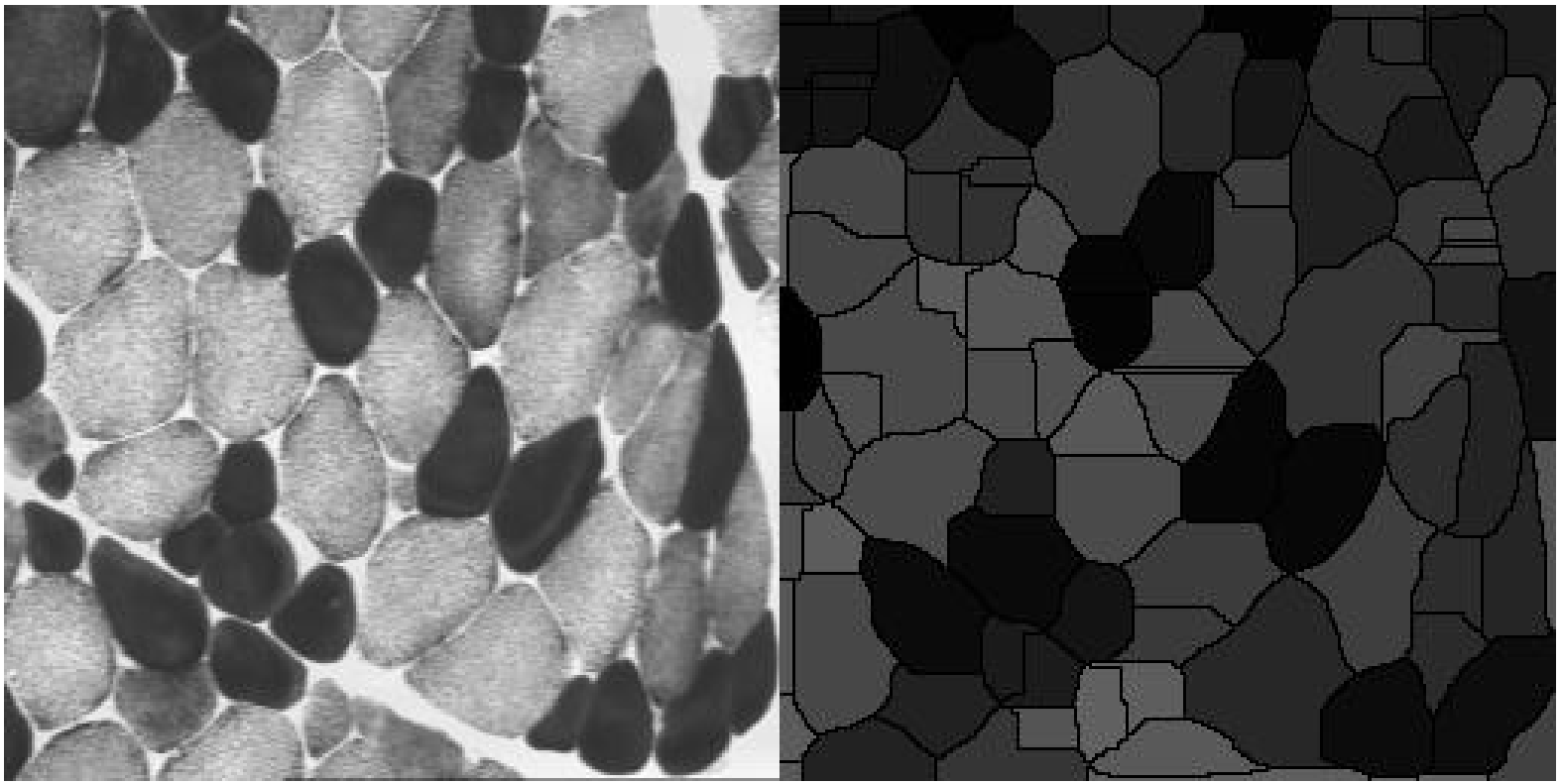


FIGURE – $\sigma = 5$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

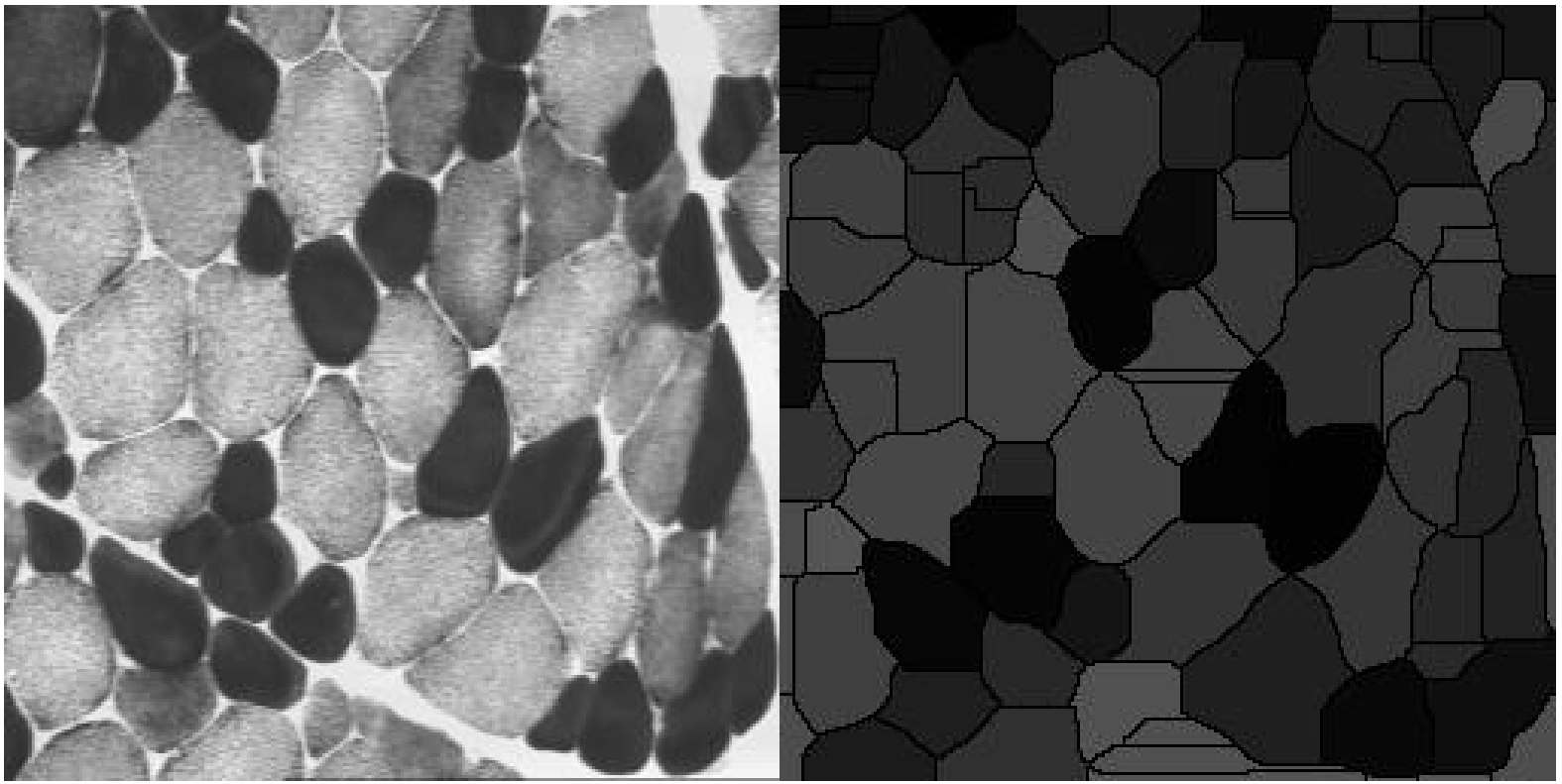


FIGURE – $\sigma = 5.5$

Ligne de partage des eaux

Exemple

- Convolution de l'image de la norme du gradient par un filtre gaussien puis segmentation.

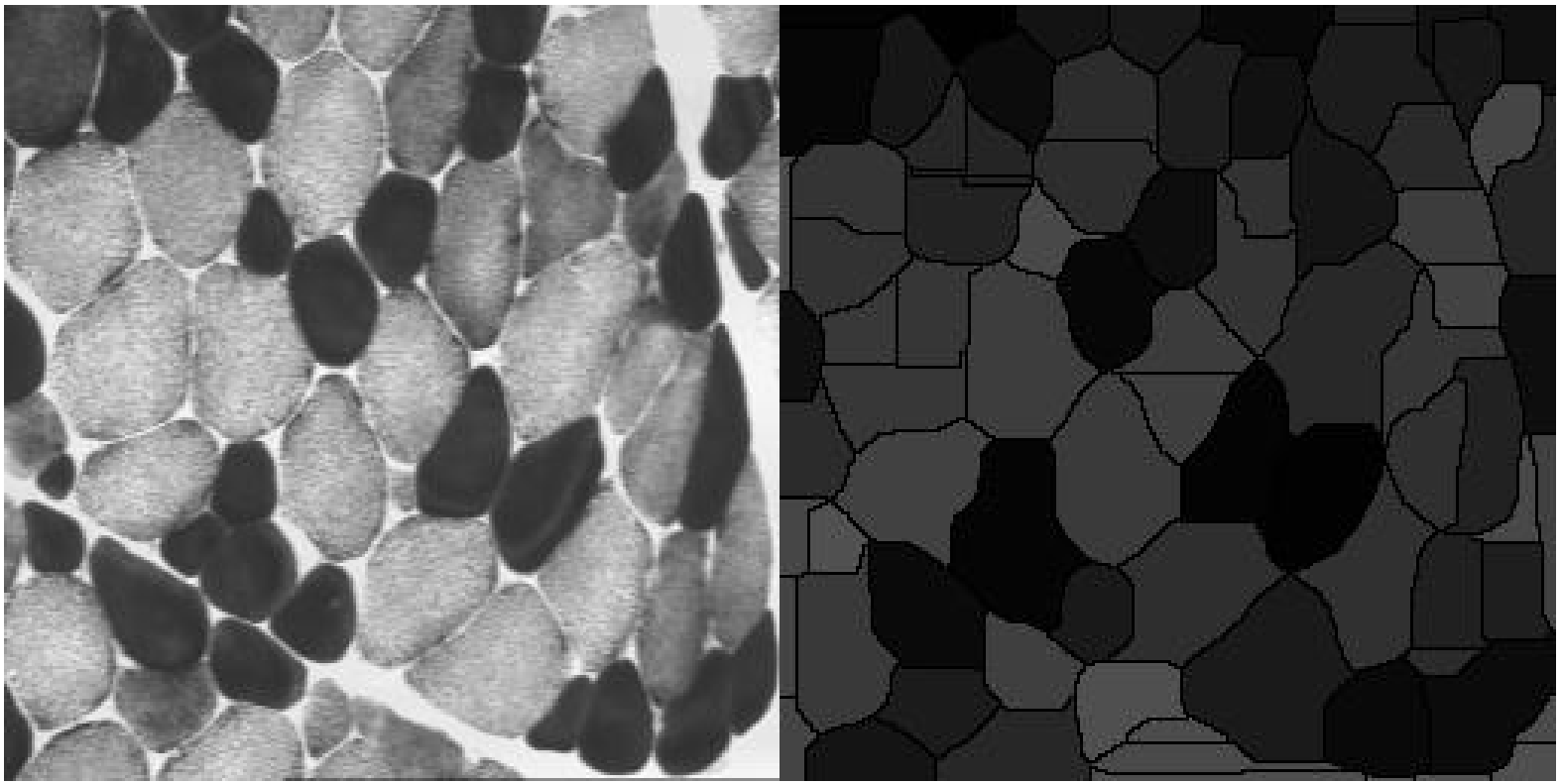


FIGURE – $\sigma = 6$