APPLICATIONS

MICROMORPH®

TABLE DES MATIERES

Introduction 4

Concepts et notions élémentaires 6
- Notations 6
- Définitions 7
- Quelques relations 8
- Définitions diverses 9
- Exercices 9

Erosions et dilatations 12
- Erosions, dilatations, rappel 12
- Gradients 13
- Exercices 14
- Résumé 28

Ouvertures, fermetures 30
- Ouvertures, fermetures, définitions 30
- Chapeau haut-de-forme 30
- Exercices 31
- Résumé 45

Filtrage morphologique 46
- Filtres, rappel 46
- Contrastes, rappel 46
- Exercices 47
- Résumé 55

Géodésie et connexité 56
- Rappels, cas binaire 56
- Cas numérique 56
- Exercices 57
- Résumé 62

Applications de la géodésie 64
- Exercices 64
- Résumé 72

Squelettes 74
- Exercices 74
- Résumé 79

Amincissements, épaississements 80
- Introduction 80

Amincissements binaires, rappel 80
Homotopie 80
Amincissements numériques 81
Exercices 81
Résumé 103

Ligne de partage des eaux 106
Zone d'influence 106
Ligne de partage des eaux 106
Exercices 107
Résumé 112

Segmentation 114
Exercices 114

Mesures 126
Rappels 126
Mesures et transformations morphologiques 126
Suggestions 126
Exercices 127
Résumé 136

INTRODUCTION

Ce manuel pratique de Morphologie Mathématique (en abrégé MM) est essentiellement un manuel d'exercices destiné à introduire étape par étape les principales notions de MM. Les exercices décrits doivent être résolus à l'aide du logiciel MICROMORPH pour Windows, programme didactique pour PC développé au Centre de Morphologie Mathématique. Le logiciel permet à l'utilisateur d'élaborer progressivement un dictionnaire de transformations morphologiques, et lui apprend donc à manipuler des transformées et des algorithmes de plus en plus complexes.

Il est préférable d'effectuer les exercices dans l'ordre où ils sont présentés. En effet, chacun d'eux introduit le plus souvent une transformation nouvelle, transformation qu'il vous faudra définir et ajouter au dictionnaire, afin de pouvoir résoudre les exercices suivants.

Chaque chapitre de ce manuel est consacré à une ou plusieurs notions fondamentales de MM. Après un bref rappel des notions et des définitions utilisées, on trouvera l'énoncé des exercices. Du fait de la structure même du langage MICROMORPH, beaucoup d'entre eux sont d'inspiration algorithmique. Cependant, pour ne pas donner à l'utilisateur l'illusion que la MM n'est qu'une suite de transformations, on a ajouté, chaque fois que nous l'avons jugé bon, quelques exercices ne faisant pas appel à MICROMORPH, et plutôt destinés à montrer soit des résultats et des théorèmes importants, soit l'usage qui peut être fait, et les renseignements que l'on peut tirer des transformations effectuées. En particulier, certains exercices constituent une application complète de la MM à la résolution d'un problème d'analyse d'image.

Certains exercices doivent être considérés comme le prolongement de l'apprentissage des notions théoriques. Il n'est en effet pas possible de donner une vision exhaustive de la MM, faute de temps, et surtout pour éviter d'embrouiller l'apprenti morphologue qui ne saurait plus alors distinguer ce qui est essentiel de ce qui est accessoire. C'est ainsi que des notions comme la covariance, les granulométries linéaires, les ensembles aléatoires, etc., font partie du "bagage culturel" élémentaire de tout morphologue, et, à ce titre, doivent donc être connues.

Les exercices de ce manuel font appel à des images qui vous sont fournies avec le logiciel MICROMORPH. Bien que la résolution de ces exercices en utilisant l'image suggérée dans l'énoncé soit indispensable, rien ne vous interdit en revanche de tester vos algorithmes sur les autres images fournies ou sur toute image obtenue par vos propres moyens.

Les transformations élaborées au cours des exercices sont regroupées dans des fichiers avec l'extension .mic (erosion.mic, etc..). Certaines transformations dont l'élaboration n'est pas présentée lors des exercices mais qui font partie de la famille des transformations introduites sont également présentes dans ces fichiers Ces transformations sont généralement commentées. De plus on trouvera dans certains fichiers (comme utility.mic, display.mic) des transformations qui ne sont pas des opérateurs morphologiques mais dont l'usage est fréquent dans l'élaboration des transformations. Le lecteur est donc invité à consulter ces fichiers toutes les fois où il rencontre dans ce manuel une procédure non explicitée.

Chapitre 1

CONCEPTS ET NOTIONS ELEMENTAIRES

Ce chapitre vous rappellera les diverses notations utilisées, ainsi que quelques définitions élémentaires. On n'abordera pas le fonctionnement du logiciel MICROMORPH. Pour ce faire, reportez-vous au manuel de référence.

1.1. Notations

La notion mathématique d'ensemble est la représentation associée aux images binaires, tandis que celle de fonction numérique (de $\mathbb{R}^2$ dans $\mathbb{R}$) est la représentation associée aux images à teintes de gris.

1.1.1. Ensembles

Les ensembles sont généralement désignés par les lettres majuscules X, Y, Z. X_i désigne la i-ème composante connexe d'un ensemble X.

1.1.2. Eléments structurants

Les éléments structurants sont désignés à l'aide des lettres majuscules B, H, L, M, T, etc. . T_1 , T_2 représentent les deux composantes d'un élément structurant T biphasé: $T=(T_1,T_2)$.

1.1.3. Fonctions

Les fonctions sont généralement désignées à l'aide des lettres minuscules f, g, h, etc. .

1.1.4. Points, vecteurs

Les lettres minuscules désignent indifféremment les points ou les vecteurs de l'espace. La distinction entre ces deux notations dépend en fait du contexte et ne pose pas de problème particulier. Ainsi :

$x \in X$ (dans ce cas, x désigne un point de l'ensemble X).
$x + y = z$ (x, y et z sont trois vecteurs ayant même origine O).

Dans ce dernier cas, on aurait pu écrire $Ox + Oy = Oz$.

1.1.5. Transformations d'ensembles et de fonctions

On utilisera les lettres grecques Ψ, Φ, etc., pour désigner les transformations. $\Phi(X)$ (resp. $\Phi(f)$) désigne le résultat de la transformée F appliquée à l'ensemble X (resp. à la fonction f).

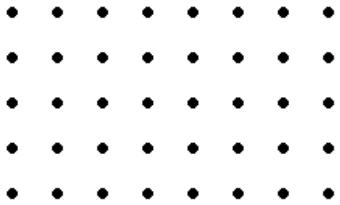
1.1.6. Sous-graphe

Le sous-graphe ou l'ombre d'une fonction f de $\mathbb{R}^n$ dans $\mathbb{R}$ est désigné par U(f) et représente l'ensemble des points (x,y) de $\mathbb{R}^n x \mathbb{R}$ tels que $y \leq f(x)$.

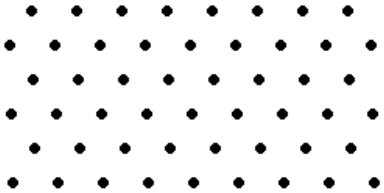
1.2. Définitions

1.2.1. Grille

On dénote classiquement par $\mathbb{Z}$ l'ensemble des entiers rationnels (i.e. positifs, négatifs ou nuls), par $\mathbb{Z}^2$ l'ensemble des couples ordonnés de deux entiers rationnels. Une grille de points est l'image d'une application injective de $\mathbb{Z}^2$ dans $\mathbb{R}^2$. En d'autres termes, $\mathbb{Z}^2$ est une classe d'équivalence des grilles, dont les propriétés sont liées à la structure mathématique de module de $\mathbb{Z}^2$.



Grille carrée

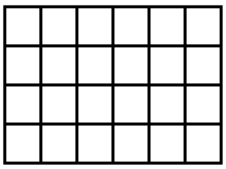


Grille hexagonale

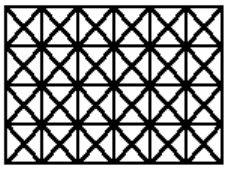
1.2.2. Trame, graphe de voisinage

Un graphe est défini par un ensemble de points, appelés les sommets du graphe, et un ensemble de couples de points, pris parmi les sommets, et appelés les arêtes du graphe.

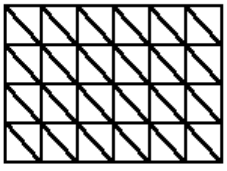
Une trame est un graphe planaire construit sur les sommets d'une grille et invariant pour le groupe des translations de $\mathbb{Z}^2$, ou un sous-graphe de celui-ci.



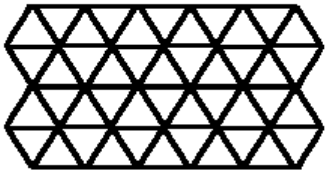
Graphe carré



Graphe octogonal



Graphe hexagonal



Graphe hexagonal

A quelques exceptions près, tous les exercices se feront sur la trame hexagonale.

1.2.3. Complémentation

1.2.3.1. Images binaires

On désigne par X^c l'ensemble complémentaire de X, c'est-à-dire l'ensemble constitué de tous les points n'appartenant pas à X.

1.2.3.2. Images numériques

La complémentation d'une image est définie par :

$$f = MAX - f$$

où f est l'image originale et MAX est le niveau de gris maximum que l'on peut coder dans le système utilisé (jusqu'à 32767 avec MICROMORPH pour Windows).

1.2.4. Union, intersection

1.2.4.1. Images binaires

$X \cup Y$ désigne l'union de X et de Y, c'est-à-dire l'ensemble des points appartenant à l'un ou à l'autre de ces ensembles. $X \cap Y$ désigne l'intersection des deux ensembles X et Y, c'est-à-dire l'ensemble des points appartenant à la fois à X et à Y.

1.2.4.2. Images numériques, Sup, inf

Sup(f,g) noté aussi $f \vee g$ désigne le sup de deux fonctions f et g : en tout point x, $(f \vee g)(x)$ est la plus grande des deux valeurs f(x) et g(x).

Inf(f,g) noté aussi $f \wedge g$ désigne l'inf de deux fonctions f et g : en tout point x $(f \wedge g)(x)$ est la plus petite des deux valeurs f(x) et g(x).

1.3. Quelques relations

1.3.1. Formules de De Morgan

Ces formules expriment la dualité de l'union et de l'intersection :

$$(X \cup Y)^c = X^c \cap Y^c \text{ et } (X \cap Y)^c = X^c \cup Y^c$$

De la même façon :

$$(f \vee g)^c = f^c \wedge g^c \text{ et } (f \wedge g)^c = f^c \vee g^c$$

1.3.2. Différence, différence symétrique

X/Y désigne la différence ensembliste de X et de Y. C'est l'ensemble constitué des points appartenant à X et n'appartenant pas à Y.

$$X/Y = X \cap Y^c$$

X*Y désigne la différence ensembliste symétrique. C'est l'ensemble des points qui n'appartiennent qu'à un et un seul des deux ensembles X et Y.

$$X*Y = (X \cap Y^c) \cup (Y \cap X^c) = (X \cup Y)/(X \cap Y)$$

1.3.3. Commutativité, associativité, distributivité

L'union, la différence symétrique, l'intersection sont des opérations commutatives et associatives. Les mêmes propriétés sont vérifiées pour le sup et l'inf.

commutativité : $X \cup Y = Y \cup X$

$f \vee g = g \vee f$

associativité : $(X \cup Y) \cup Z = X \cup (Y \cup Z) = X \cup Y \cup Z$

$(f \vee g) \vee h = f \vee (g \vee h)$

L'union et l'intersection sont distributives l'une par rapport à l'autre :

$$(X \cup Y) \cap Z = (X \cap Z) \cup (Y \cap Z)$$
$$(X \cap Y) \cup Z = (X \cup Z) \cap (Y \cup Z)$$

Le sup et l'inf sont distributifs l'une par rapport à l'autre :

$$(f \vee g) \wedge h = (f \wedge h) \vee (g \wedge h)$$
$$(f \wedge g) \vee h = (f \vee h) \wedge (g \vee h)$$

1.4. Définitions diverses

Rappelons brièvement quelques définitions concernant les transformations, définitions dont nous retrouverons l'intérêt tout au long de ces exercices.

1.4.1. Croissance

Une transformation Ψ est dite croissante si elle vérifie :

$$X \subset Y \Rightarrow \Psi(X) \subset \Psi(Y)$$
$$f \leq g \Rightarrow \Psi(f) \subset \Psi(g)$$

1.4.2. Extensivité

Ψ est extensive si :

$$X \subset \Psi(X) \text{ ou } f \leq \Psi(f)$$

1.4.3. Idempotence

Ψ est idempotente si :

$$\Psi[\Psi(X)] = \Psi(X) \text{ ou } \Psi[\Psi(f)] = \Psi(f)$$

1.4.4. Dualité

Φ et Ψsont deux transformations duales si :

$$\Phi(X) = [\Psi(X^c)]^c \text{ ou } \Phi(f) = [\Psi(f)^c]^c$$

EXERCICES

Exercice n° 1

Montrez que les opérateurs sur les fonctions inf et sup peuvent se ramener à des opérations ensemblistes sur les sous-graphes. (On montrera notamment que l'intersection des sous-graphes de f et g est le sous-graphe de l'inf de f et de g).

Solution

Immédiat :

$$\forall (x, y), (x, y) \in U(f) \cap U(g) \Leftrightarrow (x, y) \in U(f) \text{ et } (x, y) \in U(g)$$
$$\Leftrightarrow y \leq f(x) \text{ et } y \leq g(x)$$
$$\Leftrightarrow y \leq \inf(f(x), g(x))$$
$$\Leftrightarrow y \leq \inf(f, g)(x)$$
$$\Leftrightarrow (x, y) \in U(\inf(f, g))$$

Exercice n° 2

1) Démontrez et vérifiez à l'aide des images binaires fournies les relations ensemblistes énoncées plus haut (en particulier, les formules de De Morgan).

2) On définit une transformation Ψde la façon suivante :

$$\Psi(X) = X \cup Y, \text{ Y fixe}$$

- Cette transformation est-elle croissante, extensive, idempotente?

- Existe-t-il une transformation duale, et si oui, laquelle?

Solution

1) Formules de De Morgan

Démontrons que :

$$(X \cap Y)^c = X^c \cup Y^c$$

Soit x appartenant au complémentaire de $\mathbf{X} \cap Y$, x n'appartient pas à $X \cap Y$. Mais si x n'appartient pas à la fois à X et à Y, cela signifie qu'il n'appartient pas au moins à l'un des deux: $x \notin X$ ou $x \notin Y$. Q.E.D.

2) Soit $\Psi(X) = X \cup Y$, Y fixé.

Ψ est croissante :

$$\forall X_1 \subset X_2,\ \Psi(X_1) = X_1 \cup Y \subset X_2 \cup Y = \Psi(X_2)$$

Ψ est extensive (évident).

Ψ est idempotente :

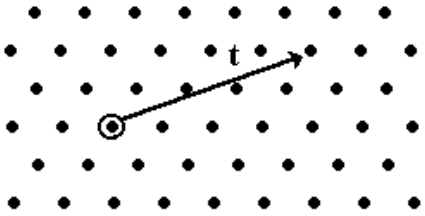
$$\Psi[\Psi(X)] = (X \cup Y) \cup Y = X \cup Y = \Psi(X)$$

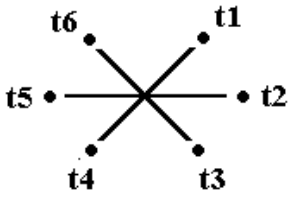
S'il existe une transformation Φ duale de Ψ, elle vérifie :

$$[\Psi(X^c)]^c = \Phi(X), \text{ soit :}$$
$$\Phi(X) = (X^c \cup Y)^c = X \cap Y^c$$

Exercice n° 3

En trame hexagonale, chaque point de la trame possède six voisins. On peut aisément définir des translations sur cette trame.





Chaque translation est la composition de plusieurs translations élémentaires. Par exemple : $t = t_1 \circ t_1 \circ t_2 \circ t_2 \circ t_2$

L'ensemble des translations muni de la loi de composition q constitue un groupe, appelé groupe des déplacements.

Les opérateurs de décalage de MICROMORPH vérifient-ils cette structure de groupe? Vérifiez et expliquez votre réponse. (Ce phénomène est la première occurrence des effets de bord engendrés lorsqu'on travaille sur un champ d'analyse fini).

Solution

(évident).

Rappelons que les effets de bord sont dus au fait que MICROMORPH assume que les points susceptibles de tomber hors du champ sont forcés à zéro. Cette convention aura des conséquences importantes lors de la définition des transformées géodésiques, mais aussi des transformées euclidiennes classiques. Les exercices suivants vous le démontreront facilement.

Chapitre 2

EROSIONS & DILATATIONS

2.1. Erosions, dilatations, rappels

2.1.1. Cas binaire

La dilatation d'un ensemble X par un ensemble B de centre O appelé élément structurant est un ensemble Y défini comme :

$$Y = X \oplus \check{B} = \{x \ : B_x \cap X = \emptyset \ \}$$

où $B_x = \left\{ x + \overrightarrow{Ob}, b \in B \right\}$.

$(X \oplus \check{B})$peut encore s'écrire :

$$X \oplus \check{B} = \bigcup_{b \in \check{B}} X_{\overrightarrow{Ob}}$$

$\check{B}$ est le transposé de B (i. e. le symétrique de B par rapport à O). Le dilaté Y est donc l'union des translatés de X.

De la même façon, l'érodé est défini par :

$$Z = X \ominus \check{B} = \{x \ : B_x \subset X\}$$

qui s'écrit encore :

$$X \ominus \check{B} = \bigcap_{b \in \check{B}} X_{\overrightarrow{Ob}}$$

Ces deux transformations ont les propriétés suivantes :

$$\begin{aligned} (X \cup Y) \oplus \check{B} &= (X \oplus \check{B}) \cup (Y \oplus \check{B}) \\ (X \cap Y) \ominus \check{B} &= (X \ominus \check{B}) \cap (Y \ominus \check{B}) \\ (X \oplus \check{B}_1) \oplus \check{B}_2 &= X \oplus (\check{B}_1 \oplus \check{B}_2) \\ (X \ominus \check{B}_1) \ominus \check{B}_2 &= X \ominus (\check{B}_1 \oplus \check{B}_2) \end{aligned} \qquad (1)$$

et ce, quels que soient les centres de B, B_1 et B_2 .

2.1.2. Cas numérique

On peut éroder et dilater le sous-graphe U(f) d'une fonction f par un élément structurant tridimensionnel B et montrer que, sous certaines conditions, la dilatation (resp. l'érosion) d'un sous-graphe est à nouveau le sous-graphe d'une fonction, appelée dilatée (resp. érodée) de f par B et notée $f \oplus \check{B}$ (resp. $f \ominus \check{B}$).

Si les éléments structurants sont plans, la dilatation d'une fonction f peut s'écrire :

$$(f \oplus \check{B})(x) = \sup_{b \in \check{B}} \left(f\left(x + \overrightarrow{Ob} \right) \right)$$

qui s'écrit encore :

$$f \oplus \check{B} = \sup_{b \in \check{B}} f_{\overrightarrow{Ob}}$$

où $f_{\overrightarrow{Ob}}$ est la fonction translatée de f de vecteur $\overrightarrow{Ob}$.

De la même façon, l'érosion est définie par :

$$(f \ominus \check{B})(x) = \inf_{b \in \check{B}} \left(f\left(x + \overrightarrow{Ob} \right) \right)$$

qui s'écrit encore :

$$f \ominus \check{B} = \inf_{b \in \check{B}} f_{\overrightarrow{Ob}}$$

Les relations (1) se transposent immédiatement aux fonctions.

$$(f \vee g) \oplus B = (f \oplus B) \vee (g \oplus B) \qquad (1')$$
$$(f \wedge g) \ominus B = (f \ominus B) \wedge (g \ominus B)$$
$$(f \oplus B_1) \oplus B_2 = f \oplus (B_1 \oplus B_2)$$
$$(f \ominus B_1) \ominus B_2 = f \ominus (B_1 \oplus B_2)$$

2.2. Gradients

2.2.1. Gradient classique

Le gradient d'une fonction f définie sur R est défini comme étant le vecteur :

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

Dans le cas digital, on peut utiliser comme expression des dérivées partielles les différences premières :

$$(\nabla_x f)(x, y) \approx f(x, y) - f(x-1, y)$$
$$(\nabla_y f)(x, y) \approx f(x, y) - f(x, y-1)$$

On reconnaît les convolutions digitales de f par les noyaux [-1 1] et $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$. On pourrait prendre aussi comme expression des dérivées partielles les différences suivantes :

$$(\nabla_{2x} f)(x, y) \approx f(x+1, y) - f(x-1, y)$$
$$(\nabla_{2y} f)(x, y) \approx f(x, y+1) - f(x, y-1)$$

qui présentent l'avantage d'être centrées en (x,y). On reconnaît à nouveau les convolutions digitales de f par les noyaux [-1 0 1] et $\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$.

On utilise dans la pratique encore d'autres différences, qui s'expriment elles aussi en terme de convolutions digitales. Une importante bibliographie illustre ce sujet. Par exemple l'opérateur de Sobel est donné par les convolutions suivantes :

$$\frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ et } \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & -1 \end{bmatrix}$$

2.2.2. Gradient morphologique

Le gradient de Beucher d'une fonction f définie sur $\mathbb{R}^2$ ou sur un sous-ensemble est défini par :

$$g(f) = \lim_{\lambda \to 0} \frac{(f \oplus \lambda B) - (f \ominus \lambda B)}{2\lambda}$$

où λB désigne une boule de rayon λ.

Sur une trame hexagonale, on obtient:

$$g(f) = \frac{(f \oplus B) - (f \ominus B)}{2}$$

où H est un hexagone de taille 1. Ce gradient est égal au module du gradient d'une fonction f continûment dérivable.

EXERCICES

Exercice n° 1

1) Démontrez les relations (1) et (1').

2) Vérifiez ou infirmez les assertions suivantes :

- Croissance de l'érosion et de la dilatation.
- Extensivité, anti-extensivité de ces deux transformations.
- Idempotence.
- Dualité de l'érosion et de la dilatation.

Solution

1) Démontrons que :

$$(X \cup Y) \oplus \check{B} = (X \oplus \check{B}) \cup (Y \oplus \check{B})$$

$$(X \cup Y) \oplus \check{B} = \bigcup_{b \in \check{B}} (X \cup Y)_b = \bigcup_{b \in \check{B}} (X_b \cup Y_b) = \left(\bigcup_{b \in \check{B}} X_b \right) \cup \left(\bigcup_{b \in \check{B}} Y_b \right)$$

soit $(X \oplus \check{B}) \cup (Y \oplus \check{B})$.

On démontre de la même façon la deuxième relation. Démontrons la troisième relation:

$$(X \oplus \check{B}_1) \oplus \check{B}_2 = \bigcup_{b_2 \in \check{B}_2} (X \oplus \check{B}_1)_{b_2} = \bigcup_{b_2 \in \check{B}_2} \left(\bigcup_{b_1 \in \check{B}_1} X_{b_1} \right)_{b_2} = \bigcup_{b_1 \in \check{B}_1, b_2 \in \check{B}_2} (X_{b_1 + b_2})$$

$$= \bigcup_{b \in \check{B}_1 \oplus \check{B}_2} (X_b) = X \oplus (\check{B}_1 \oplus \check{B}_2)$$

2) Vérification des assertions :

- L'érosion et la dilatation sont des transformations croissantes : Vrai (conséquence de la croissance de l'union et de l'intersection ensemblistes).
- Extensivité ou anti-extensivité?

Examinons le cas de l'érosion.

$$X \ominus \check{B} = \bigcap_{b \in \check{B}} X_b$$

L'érodé étant l'intersection des translatés de X, il ne contient pas ce dernier, à moins que B soit réduit au point origine.

Inversement, pour affirmer que $X \ominus \check{B} \subset X$, il faudrait pouvoir écrire :

$$X \ominus \check{B} = X \cap \left(\bigcap_{b \in \check{B} - \{\emptyset\}} X_b \right)$$

ce qui n'est exact que lorsque l'élément structurant B contient son origine. En règle générale, l'érosion n'est donc ni extensive, ni anti-extensive.

Il en est de même pour la dilatation.

- L'érosion et la dilatation sont idempotentes : faux (évident).
- L'érosion et la dilatation sont des transformations duales :

$$(X \oplus \check{B})^c = \left(\bigcup_{b \in \check{B}} X_b \right)^c = \bigcap_{b \in \check{B}} (X_b)^c = \bigcap_{b \in \check{B}} (X)_b^c = (X^c \ominus \check{B}) \text{ Q.E.D.}$$

Exercice n° 2

1) Utilisez les primitives du langage MICROMORPH pour définir les transformations suivantes :

- érosion et dilatation par un segment de taille n (constitué de n+1 points consécutifs) dans les cas binaire et numérique.

$$X \ominus \check{L}_n,\ X \oplus \check{L}_n,\ f \ominus \check{L}_n,\ f \oplus \check{L}_n$$

- érosion et dilatation par un doublet de points distants de n :

$$X \ominus \check{K}_n,\ X \oplus \check{K}_n,\ f \ominus \check{K}_n,\ f \oplus \check{K}_n$$

(l'origine des deux éléments structurants est choisie arbitrairement à l'une des extrémités.)

2) Montrez que :

$$X \ominus \check{L}_n \subset X \ominus \check{K}_n$$

[procédures **direro** ; **dirdil** ; **dblero** ; **dbldil**]

Solution

1) Définissons les transformées suivantes :

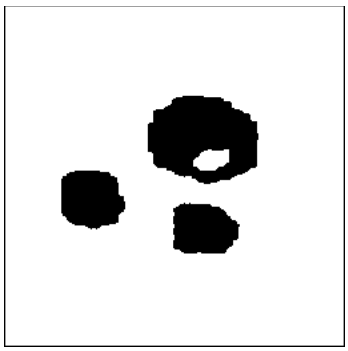
- Erosions linéaires binaires et numériques :

$X \ominus \check{L}_n$ et $f \ominus \check{L}_n$

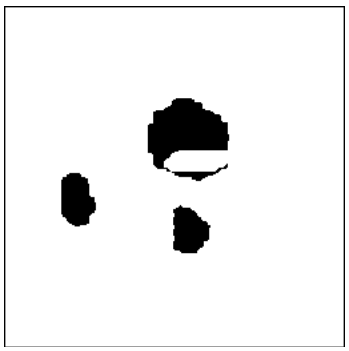
$\underbrace{(0.....)}$

n points

```
deproc direro direro dir s d sz
syntax "direro imin imout dir size -> directional line erosion of size sz "
  imcopy s d
  iminfngb d d dir sz
end
```



(a)



(b)

Erosion linéaire binaire
(a) original, (b) érodé horizontal de taille 23

- Dilatations linéaires binaires et numériques :

$X \oplus \check{L}_n$ et $f \oplus \check{L}_n$

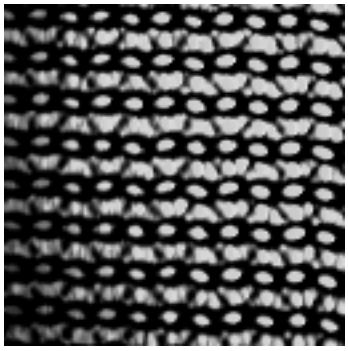
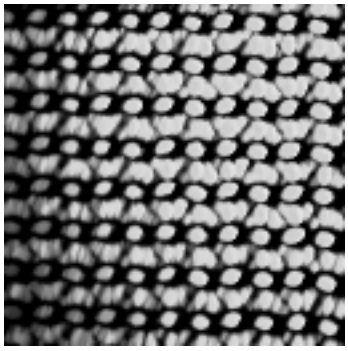
```
deproc dirdil dirdil dir s d sz
syntax "dirdil dir imin imout size --> directional line dilation"
    imcopy s d
    imsupngb d d dir sz
 end
```

- Erosion par un doublet de points :

$X \ominus \check{K}_n$ et $f \ominus \check{K}_n$

$\underbrace{(0 \qquad .)}$

distance n

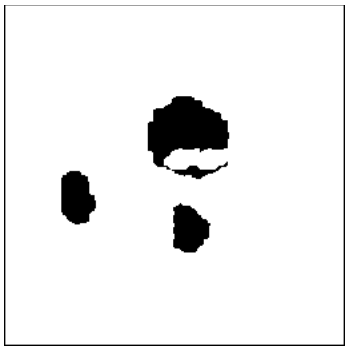



(a) (b)

Dilatation linéaire numérique
(a) original, (b) dilaté linéaire de taille 5

```
deproc dblero dblero dir s d sz
syntax "greydblero dir greyin greyout size"
  int w ;
  w := imalloc imdepth s
  imcopy s w
  imcopyngb w w dir sz impixmin s
  iminf s w d
  imfree w
end
```

Cette transformation utilise une mémoire de travail. La transformation est appelée covariance en M.M. (voir plus loin).



Covariance de taille 23

- Dilatation par un doublet de points :
$X \oplus \check{K}_n$, $f \oplus \check{K}_n$

```
deproc dbldil dbldil dir s d sz
syntax "greydbldil dir greyin greyout size"
  int w ;
  w := imalloc imdepth d
  imcopy s w
  imcopyngb w w dir sz impixmin s
  imsup s w d
```

```
    imfree w
end
```

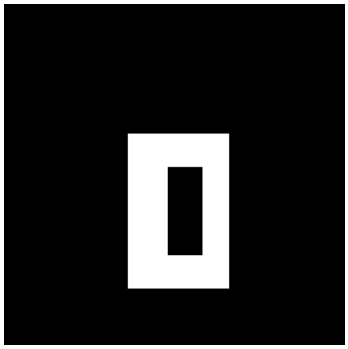
2) Montrons que $X \ominus \check{L}_n \subset X \ominus \check{K}_n$. Soit x un point de $X \ominus \check{L}_n$. L'élément structurant L_n translaté en x est donc inclus dans X. Ses deux extrémités le sont donc a fortiori. Donc x appartient à l'érodé $X \ominus \check{K}_n$. (Q.E.D.)

Exercice n° 3

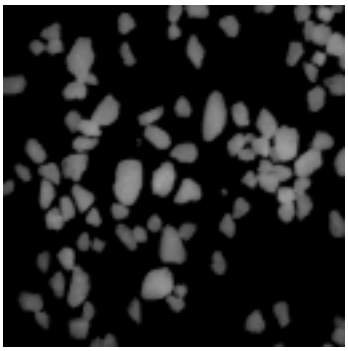
1) Montrez qu'un hexagone élémentaire peut être généré par trois dilatations successives d'un point par des segments judicieusement orientés. En déduire un algorithme permettant d'obtenir l'érosion et la dilatation par un hexagone. Programmez ces transformations ainsi que les transformations équivalentes en trame carrée et vérifiez vos algorithmes à l'aide des images binaires CAT, OBJECTS, SHAPE et des images numériques KNITTING, SALT et CIRCUIT.

CAT

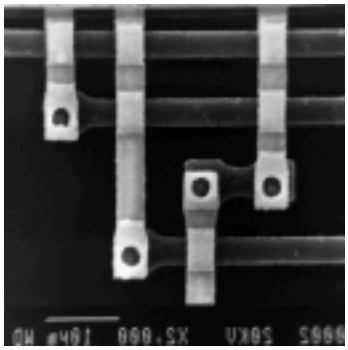
OBJECTS



SHAPE



SALT



CIRCUIT

2) Vérifiez la dualité des transformations. Etes-vous satisfait du résultat?

[Procédures **ero** ; **dil**]

Solution

1) La figure ci-après explicite la génération d'un hexagone à partir de trois segments. On a donc :

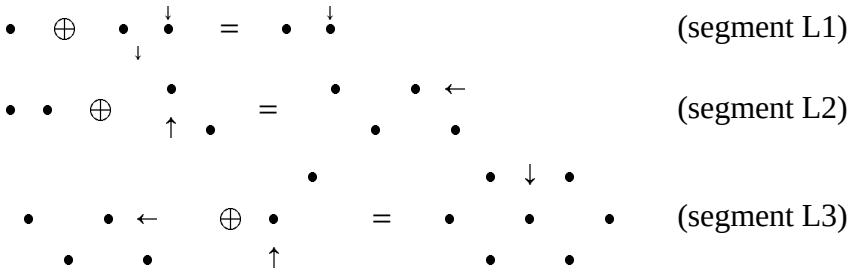
$$H = L1 \oplus L2 \oplus L3$$

On peut donc écrire :

$$X \oplus H = ((X \oplus L1) \oplus L2) \oplus L3$$

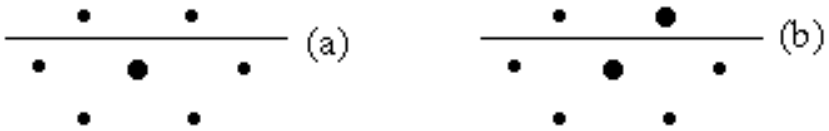
et encore :

$$X \ominus H = X \ominus (L1 \oplus L2 \oplus L3) = ((X \ominus L1) \ominus L2) \ominus L3$$



Le même procédé de construction s'applique, avec une direction supplémentaire, pour l'érosion carrée Cette érosion peut donc être définie de la manière suivante :

```
deproc ero ero s d sz
syntax "ero imin imout size"
   imcopy s d
   if (grid = 1) then
    for 1 to sz do
      iminfngb d d 1 1
      iminfngb d d 3 1
      iminfngb d d 5 1
    end
  else
   for 1 to sz do
     iminfngb d d 1 1
     iminfngb d d 3 1
     iminfngb d d 5 1
     iminfngb d d 7 1
   end
 end
```



(a) image initiale
(b) dilatation linéaire à 60 . Le point tombe hors du champ et est perdu.

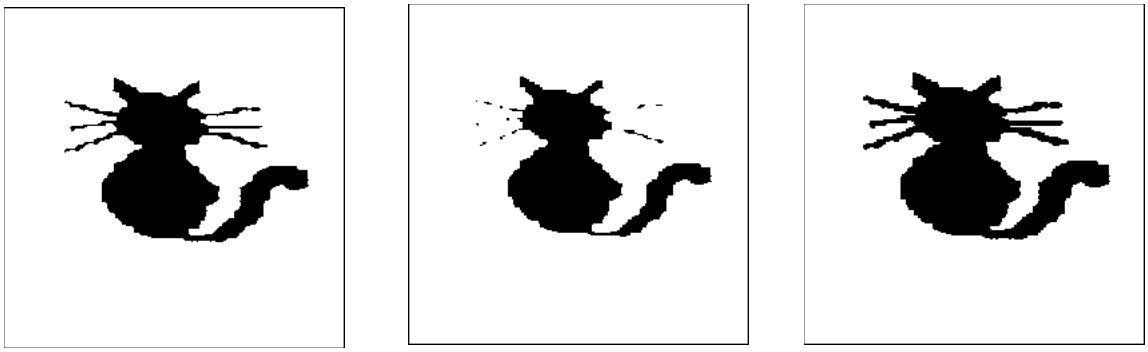
On pourrait utiliser la même formule dans la définition de la dilatation hexagonale. Mais c'est compter sans les effets de bord. En effet (voir ci-dessus), un point sur le bord du champ ne sera pas toujours correctement dilaté, du fait que la dilatation linéaire peut propager un point blanc à l'extérieur du champ. Ce point blanc est alors irrémédiablement perdu.

On remarquera que ce phénomène ne se produit pas en trame carrée.

La solution consiste à effectuer des translatés dans les six directions de la trame hexagonale. La dilatation obtenue est alors la suivante (carrée ou hexagonale).

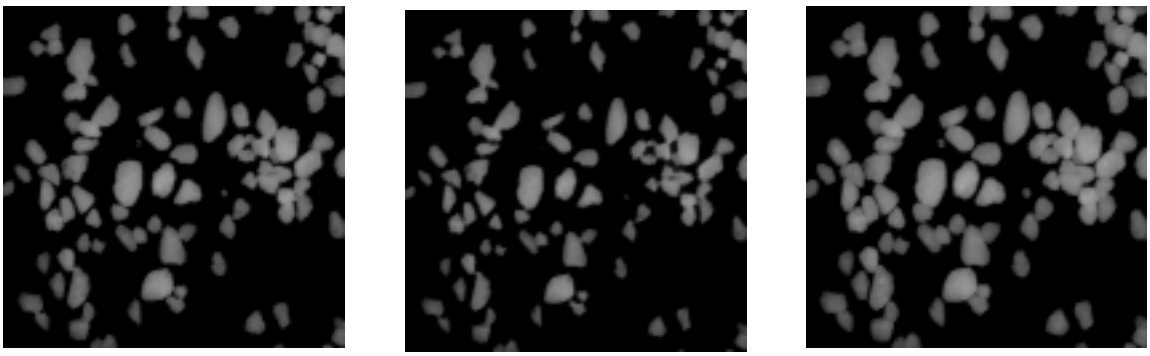
```
deproc dil dil s d sz
syntax "dil in out size"
  int w i ;
  imcopy s d
```

```
  if (grid = 1) then
    w := imalloc imdepth s
    for 1 to sz do
      i := 0
     imcopy d w
      for 1 to 6 do
        imsupngb d w ++ i 1
      end
     imcopy w d
    end
    imfree w
  else
    for 1 to sz do
      imsupngb d d 1 1
      imsupngb d d 3 1
      imsupngb d d 5 1
      imsupngb d d 7 1
    end
  end
end
```



(a) (b) (c)

Erosion, dilatation hexagonales binaires
(a) original, (b) érodé, (c) dilaté



(a) (b) (c)

Erosion, dilatation hexagonales numériques

2) Vérification de la dualité (voir note importante ci-dessous)

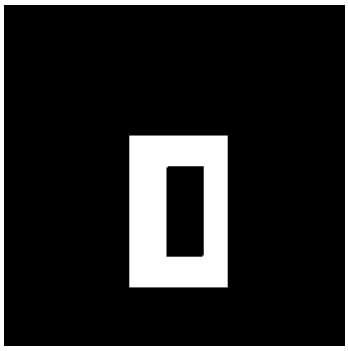
Sur une image quelconque (prendre une image binaire pour mieux visualiser le phénomène) stockée en mémoire b1, on effectue :

dil b1 b2 1
imdisplay b2 "dilaté"

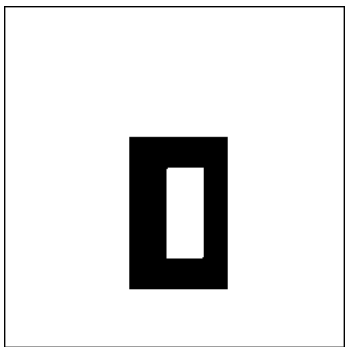
puis :

iminv b1 b3
ero b3 b3 1
iminv b3 b3
imdisplay b3 "dualité"

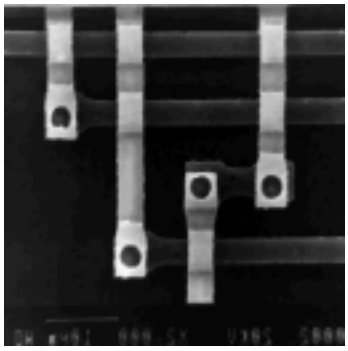
On constate l'apparition d'un cadre sur la deuxième image, dû aux effets de bord (l'inversion n'opère que dans le champ de l'image).



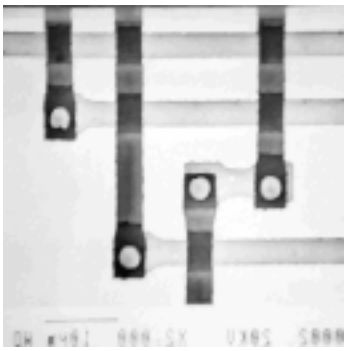
(a)



(b)



(c)



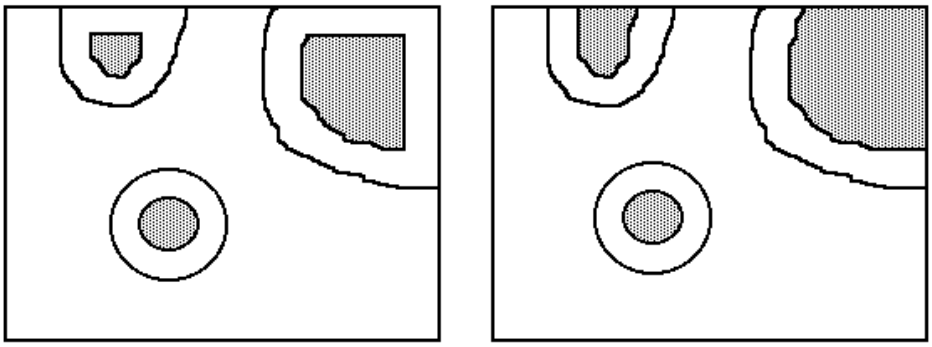
(d)

Vérification de la dualité
(a) érosion de l'image binaire, (b) dilatation du complémentaire
(c) érosion de l'image numérique, (d) dilatation du complémentaire

IMPORTANT! Transformations élémentaires et effets de bord

Il est important pour le bon fonctionnement des exercices précédents que la variable **edge** soit égale à 0. Si ce n'est pas le cas, faire **imsetedge 0** avant de lancer les transformations. Vous remarquerez dans ce cas que les érosions binaires ou numériques laissent une bordure noire autour des images. Dans ce mode, tout se passe comme si l'extérieur du champ d'analyse était vide. Les objets sont considérés comme étant totalement inclus dans le champ d'analyse. Cependant ce mode est souvent pénalisant surtout pour les images numériques car il réduit considérablement l'image au fur et à mesure que la taille des érosions augmente. Il existe donc un autre mode de fonctionnement des érosions, appelé "mode standard" (c'est en fait un

mode géodésique, cf. chapitre sur les transformations géodésiques), où l'élément structurant B lorsqu'il empiète sur le bord du champ D ne prend en compte que les points à l'intérieur de ce champ. Cela revient donc à effectuer la transformation avec l'élément structurant $B \cap D$. Dans le cas de la dilatation, ce mode n'est pas différent du précédent. Ce n'est pas vrai dans le cas de l'érosion comme l'illustre la figure ci-dessous :



Différence entre l'érosion en mode 0 (à gauche) et en mode "standard" (à droite)

Pour réaliser cette transformation, on pourrait penser qu'il suffit de positionner **edge** à 1 par l'instruction **imsetedge 1**. Malheureusement, dans le cas hexagonal cela ne suffit pas car l'érosion obtenue est entachée des mêmes défauts que la dilatation lorsqu'elle est effectuée en utilisant trois directions. Il est donc nécessaire, si on désire des transformations hexagonales non biaisées sur le bord du champ, quelque soit le mode utilisé, de réaliser l'érosion hexagonale comme la dilatation par translation selon les six directions principales de cette trame. La procédure correspondante est donc :

```
deproc ero ero s d sz
syntax "ero in out size"
  int w i ;
  imcopy s d
  if (grid = 1) then
    w := imalloc imdepth s
    for 1 to sz do
      i := 0
     imcopy d w
      for 1 to 6 do
        iminfngb d w ++ i 1
      end
     imcopy w d
    end
    imfree w
   else
    for 1 to sz do
      iminfngb d d 1 1
      iminfngb d d 3 1
```

```
        iminfngb d d 5 1
        iminfngb d d 7 1
      end
    end
  end
```

Cette procédure a l'avantage d'être exacte en trame hexagonale et lorsque edge est égal à 1 de respecter la dualité vis-à-vis de la complémentation. Son unique défaut provient de la nécessité de mettre en oeuvre davantage de ressources mémoire et d'être plus longue.

Les transformations morphologiques ero et dil exactes ont été implantées dans MICROMORPH. Cependant on trouvera dans le fichier ch2misc.mic les versions de l'érosion et de la dilatation hexagonales rapides (utilisant trois directions). Les résultats sont biaisés sur le bord du champ, mais on pourra les utiliser lorsqu'on souhaite privilégier la vitesse de traitement au détriment de l'exactitude de la transformation sur le bord du champ. Les deux procédures s'appellent **erode** et **dilate**. Si on veut les utiliser à la place de ero et dil, il suffit de les renommer et de compiler le fichier. Ces procédures sont les suivantes (rappelons qu'elles n'apportent aucune amélioration de vitesse en trame carrée) :

```
deproc erode erode s d sz
    syntax "erode imin imout size"
      imcopy s d
      if (grid = 1) then
        for 1 to sz do
          iminfngb d d 1 1
          iminfngb d d 3 1
          iminfngb d d 5 1
        end
      else
        for 1 to sz do
          iminfngb d d 1 1
          iminfngb d d 3 1
          iminfngb d d 5 1
          iminfngb d d 7 1
        end
      end
```

(c'est en fait l'érosion initialement définie en début d'exercice).

```
deproc dilate dilate s d sz
    syntax "dilate imin imout size"
      imcopy s d
      if (grid = 1) then
        for 1 to sz do
          imsupngb d d 1 1
          imsupngb d d 3 1
          imsupngb d d 5 1
        end
      else
        for 1 to sz do
```

```
    imsupngb d d 1 1
    imsupngb d d 3 1
    imsupngb d d 5 1
    imsupngb d d 7 1
  end
end
```

Exercice n° 4

1) Programmez l'érosion par un triangle élémentaire, pointe en haut (on prendra cette pointe pour origine).

Programmez également l'érosion par un carré de taille 2x2.

2) Que se passe-t-il lorsqu'on réitère la transformation hexagonale? (effectuez $(X \ominus \check{B}) \ominus \check{B}$). Représentez l'élément structurant B' tel que :

$$(X \ominus \check{B}) \ominus \check{B} = X \ominus \check{B}'$$

Une façon simple de savoir à quoi ressemble cet élément est de dilater un point par B, deux fois. En effet :

$$\left[\{.\} \oplus \check{B}\right] \oplus \check{B} = \check{B} \oplus \check{B} = \overset{\vee}{\overline{B \oplus B}}$$

$$(X \ominus \check{B}) \ominus \check{B} = X \ominus (\check{B} \oplus \check{B}) \Rightarrow B' = B \oplus B$$

3) Programmez la dilatation par un triangle pointe en bas.

4) Effectuez les dilatations par les éléments structurants suivants :

Si H est l'hexagone élémentaire, quel est l'élément structurant équivalent à :

$$B_1 \oplus H \oplus B_2$$

[procédures **miniero** ; **minidil** ; **binb1dil** ; **binb2dil**]

Solution

1) Erosion par un triangle (pointe en haut)

Cette érosion s'obtient par l'intersection de deux érosions par des segments orientés dans les directions 3 et 4.

L'érosion par le carré élémentaire est similaire (directions 3 et 5).

```
deproc miniero miniero s d
syntax "miniero imin imout"
```

```
  int w;
  w := imalloc imdepth s
  imcopy s w
  if (grid = 1) then
     iminfngb s w 3 1
     iminfngb s w 4 1
  else
     iminfngb s w 3 1
     iminfngb w w 5 1
   end
   imcopy w d
   imfree w
end
```

2) La répétition de deux transformations triangulaires aboutit à une transformation triangulaire de taille 2.

(a) (b) (c)

Erosion, dilatation triangulaires
(a) original binaire (b) érodé triangulaire (c) dilaté triangulaire

3) Dilatation par un triangle (pointe en bas)

On l'obtient par l'union de deux dilatations linéaires par des segments orientés dans les directions 1 et 6.

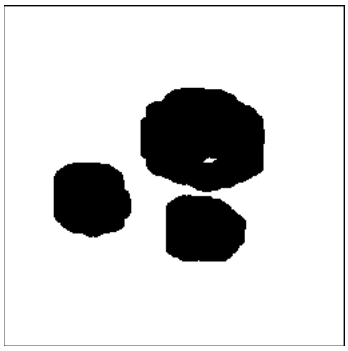
La procédure suivante effectue aussi la dilatation par le carré élémentaire conjugué.

```
deproc minidil minidil s d
syntax "minidil imyin imout"
   int w ;
   w := imalloc imdepth s
    imcopy s w
    if (grid = 1) then
      imsupngb s w 1 1
      imsupngb s w 6 1
    else
      imsupngb s w 7 1
      imsupngb w w 1 1
    end
```

```
    imcopy w d
    imfree w
  end
```

4) Dilatations par les éléments structurants B1 et B2
Voici les procédures binaires et numériques :

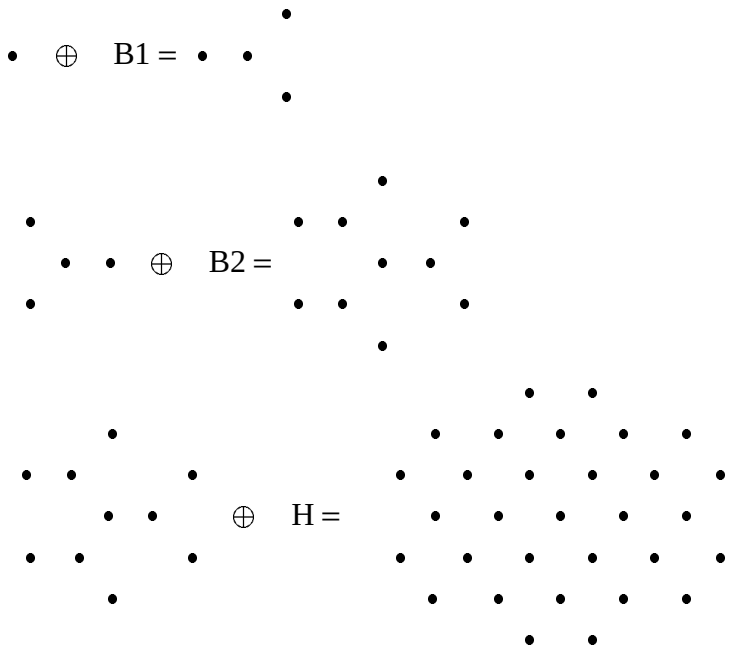
```
deproc b1dil b1dil s d
syntax "b1dil imin imout"
  int w;
  if (grid <> 1) then makeerror 10001 end
 w := imalloc imdepth s
  imcopy s w
  imsupngb s w 1 1
  imsupngb s w 3 1
  imsupngb s w 5 1
  imcopy w d
  imfree w
end
```



Dilatation dodécagonale

```
deproc b2dil b2dil s d
  syntax "b2dil imin imout"
   int  w ;
   if (grid <> 1) then makeerror 10001 end
   w := imalloc imdepth s
   imcopy s w
   imsupngb s w 4 1
   imsupngb s w 4 1
   imsupngb s w 4 1
   imcopy w d
   imfree w
end
```

Trois dilatations successives par B1, B2 et H sont équivalentes à une dilatation dodécagonale.



Exercice n° 5 : Fonction distance

Soit d, une distance définie sur les points de la trame (euclidienne, longueur du plus court chemin tracé sur la trame entre deux points). En tout point x de X, la valeur de la fonction distance est $dist(x) = d(x, X^c) = \min_{y \in X^c} d(x, y)$.

Montrez que si d est la distance définie sur la trame hexagonale comme la longueur du plus court chemin tracé sur la trame joignant deux points, la fonction distance peut être obtenue grâce aux érodés successifs de X par un hexagone.
[procédures **distance**]

Solution

Pour cette distance, l'ensemble des points situés à une distance inférieure ou égale à n d'un point x est l'hexagone de taille n centré en x.

Soit x un point de X situé à une distance n de X^c (la fonction distance en x vaut n). L'hexagone de taille n centré en x contient un point de X^c (par hypothèse) et l'hexagone de taille n-1 centré en x ne contient pas de point de X^c (sinon la distance serait inférieure à n). Donc x appartient à l'érodé de X par un hexagone de taille n-1 et n'appartient pas à l'érodé de X par un hexagone de taille n.

On en déduit un algorithme de détermination de la fonction distance :

```
deproc distance distance s d
syntax "distance s d"
   int w ;
   w := imalloc 1
   imcopy s w
   imset impixmin d d
   while ( imvolume w ) do
      imadd w d d
      ero w w 1
   end
```

```
    imfree w
  end
```

Exercice n° 6

Sur l'image ROAD résolvez les problèmes suivants :

1) Programmez et appliquez le gradient de Sobel.

2) Programmez et appliquez le gradient de Beucher de taille λ d'une fonction f :

$$g(f) = (f \oplus \lambda B) - (f \ominus \lambda B)$$

où λB désigne une boule de taille λ (on prendra ici B = H).

[procédures **gradient** ; **sobel**]

ROAD

Solution

1) L'opérateur de convolution générant une image signée (en fait pseudo-signée, les valeurs positives étant codées de 0 à 127 et les valeurs négatives de 128 à 255), définissons préalablement la fonction permettant d'obtenir la valeur absolue d'une image.

```
deproc greyabs greyabs s d
syntax "greyabs greyin greyout"
  int w;
  if (imdepth s = 8) then
    w := imalloc imdepth s
    imcopy s w
    imcsub s 127 w
    imsub s w d
  else
   if (imdepth s = 16) then
     imabs16 s d
   else
    makeerror 1404
   end
  end
  imfree w
end
```

Le résultat est une image dont les valeurs sont comprises entre 0 et 128 (image 8 bits).

L'opérateur de gradient de Sobel est donné par :

```
deproc sobel sobel s d
syntax "sobel greyin greyout"
  int w;
  w := imalloc 8
  imconvolve s w  "sobelx"
  greyabs w w
 imconvolve s d "sobely"
 greyabs d d
 imsup w d d
 imfree w
end
```

2) Le gradient morphologique "épais" se programme de la façon suivante :

```
deproc gradient gradient s d sz
syntax "gradient greyin greyout size : gradient"
  int w ;
  w := imalloc imdepth s
  dil s w sz
  ero  s d sz
  imsub w d d
  imfree w
end
```

RESUME

A l'issue de cette séance d'exercices, votre dictionnaire MICROMORPH devrait contenir les transformations dont la liste suit. Bien évidemment, vous n'êtes pas tenu de respecter la syntaxe utilisée ci-dessous. Néanmoins, nous vous y engageons par souci de lisibilité de ce manuel ainsi que des corrigés. A la fin de chaque chapitre d'exercices, vous trouverez une liste similaire. Consultez-la avant de programmer les transformations afin de vous éviter un fastidieux travail de mise en correspondance.

direro dir s d sz
érosion par un segment de taille sz dans une direction dir de l'image s dans l'image d.
dirdil dir s d sz
dilatation par un segment de taille sz dans une direction dir de l'image s dans l'image d.
dblero dir s d sz
érosion par un doublet de points distants de sz dans une direction dir de l'image s dans l'image d.
dbldil dir s d sz
dilatation par un doublet de points distants de sz dans une direction dir de l'image s dans l'image d.
ero s d sz
érosion hexagonale ou carrée de taille sz de l'image s dans l'image d.
dil s d sz
dilatation hexagonale ou carrée de taille sz de l'image s dans l'image d.

miniero s d
érosion par un triangle (pointe en haut) ou un carré 2x2 de l'image s dans l'image d.
minidil s d
dilatation par un triangle (pointe en bas) ou un carré 2x2 de l'image s dans l'image d.
b1dil s d
dilatation par l'élément structurant B_1 de l'image s dans l'image d.
b2dil s d
dilatation par l'élément structurant B_2 de l'image s dans l'image d.
distance s d
fonction distance par érosions successives de l'image binaire s. Le résultat est placé dans l'image numérique d.
sobel s d
gradient de Sobel de l'image numérique s dans l'image numérique d.
gradient s d sz
gradient morphologique par un hexagone de taille sz de l'image numérique s dans l'image numérique d.

Chapitre 3

OUVERTURES, FERMETURES

3.1. Ouvertures, fermetures, définitions

3.1.1. Cas binaire

L'érosion et la dilatation permettent la définition de deux nouvelles transformations, l'ouverture γ et la fermeture φ :

$$\gamma(X) = (X \ominus \check{B}) \oplus B$$
$$\varphi(X) = (X \oplus \check{B}) \ominus B$$

3.1.2. Cas numérique

L'ouverture et la fermeture sont définies de la même manière par :

$$\gamma(f) = (f \ominus \check{B}) \oplus B$$
$$\varphi(f) = (f \oplus \check{B}) \ominus B$$

3.1.3. Propriétés, granulométries

L'ouverture est une opération croissante et anti-extensive. La fermeture est croissante et extensive. Ces deux transformations sont idempotentes.

Si λB désigne l'homothétique du convexe B, l'ouverture est une transformation granulométrique. En particulier :

$$\text{si } \lambda > \mu,\ [(X)_{\lambda B}]_{\mu B} = \left[(X)_{\mu B}\right]_{\lambda B} = (X)_{\lambda B}$$
$$[(f)_{\lambda B}]_{\mu B} = \left[(f)_{\mu B}\right]_{\lambda B} = (f)_{\lambda B}$$

L'ouverture et la fermeture sont utilisées à des fins d'analyse granulométrique d'une part, et à des fins de filtrage d'autre part. Ces deux utilisations seront illustrées dans les exercices.

3.2. Chapeau haut-de-forme

Le chapeau haut-de-forme est une transformation que l'on n'utilise que pour des images à teinte de gris. On définit le chapeau haut-de-forme WTH d'une fonction f par :

$$WTH(f) = f - \gamma(f) \quad \text{(white Tophat)}$$

De manière similaire, le chapeau haut-de-forme conjugué BTH d'une fonction f est défini par :

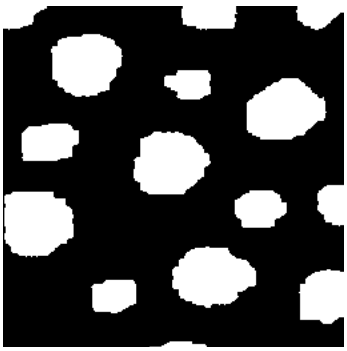
$$BTH(f) = \varphi(f) - f \quad \text{(black Tophat)}$$

EXERCICES

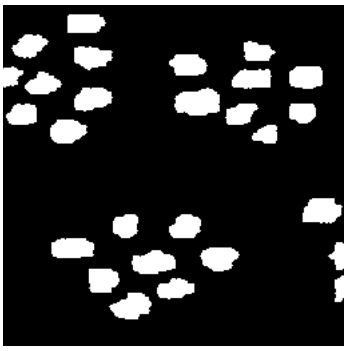
Exercice n° 1

1) Utilisez, pour programmer les transformations suivantes, les transformations (érosions, dilatations) déjà générées dans le dictionnaire :
- ouverture et fermeture par un hexagone de taille n,
- ouverture et fermeture par un segment de taille n et ouverture et fermeture par un doublet de points distants de n.

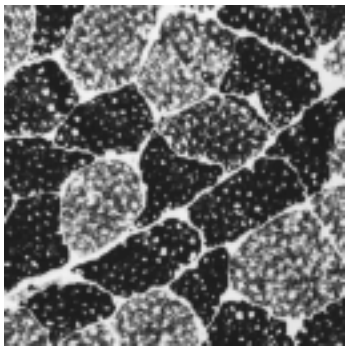
2) Vérifiez les propriétés énoncées plus haut. Démontrez en particulier la dualité de l'ouverture et de la fermeture. Vérifiez également que l'ouverture par un doublet de points n'est pas une granulométrie (des ouvertures de petite taille suppriment plus de points que des ouvertures de taille plus grande). Utilisez en particulier les images PARTIC2 et KNITTING.
[procédures **open** ; **close** ; **diropen** ; **dirclose**]



PARTIC1



PARTIC2

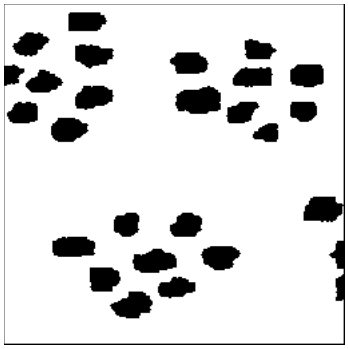


MUSCLE

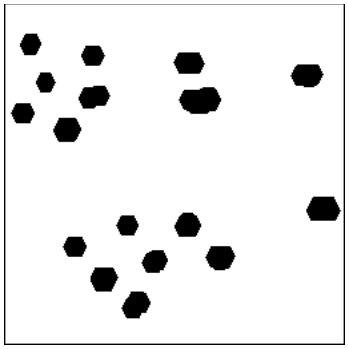
Solution

1) Voici les définitions des transformations :
- <u>Ouverture et fermeture hexagonales</u> :

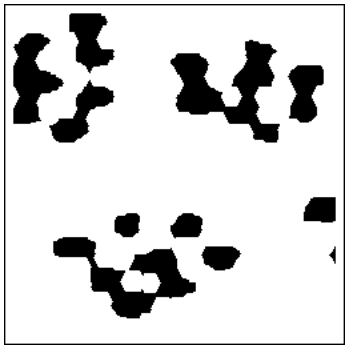
```
deproc open open s d sz
syntax "open imin imout size"
   ero s d sz
   dil d d sz
  end
```



(a)



(b)



(c)

Ouvertures, fermetures binaires
(a) original, (b) (c) transformations hexagonales

```
deproc close close s d sz
syntax "close imin imout size"
  dil s d sz
  ero d d sz
end
```

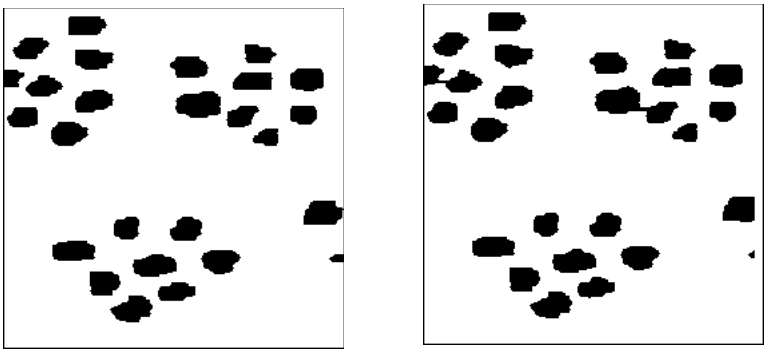
- Ouverture et fermeture linéaires :

```
deproc diropen diropen dir s d sz
syntax "diropen dir imin imout size"
   direro dir s d sz
   dirdil dirtranspose dir d d sz
end

deproc dirclose dirclose dir s d sz
syntax "dirclose dir imin imout size"
  dirdil dir s d sz
  direro dirtranspose dir d d sz
end
```

On remarquera l'utilisation du mot **dirtranspose** (fourni dans la librairie standard) qui permet de calculer la direction transposée de l'élément structurant utilisé lorsque celui-ci n'est pas isotrope.

- Ouverture et fermeture par un doublet de points :

```
deproc dblopen dblopen dir s d sz
syntax "dblopen dir binin binout size"
 dblero dir s d sz
 dbldil dirtranspose d d sz
end
```



(a) (b)

Ouvertures, fermetures binaires (suite)
(a) ouverture, (b) fermeture linéaires

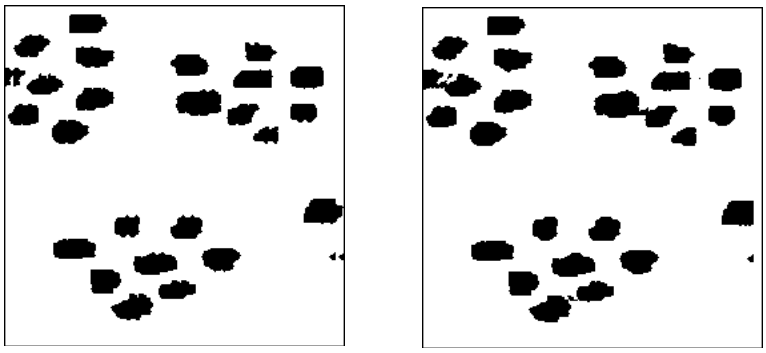
```
deproc dblclose dblclose dir s d sz
syntax "dblclose dir binin binout size"
```

```
  dbldil dir s d sz
  dblero dirtranspose d d sz
end
```

2) Dualité de l'ouverture et de la fermeture

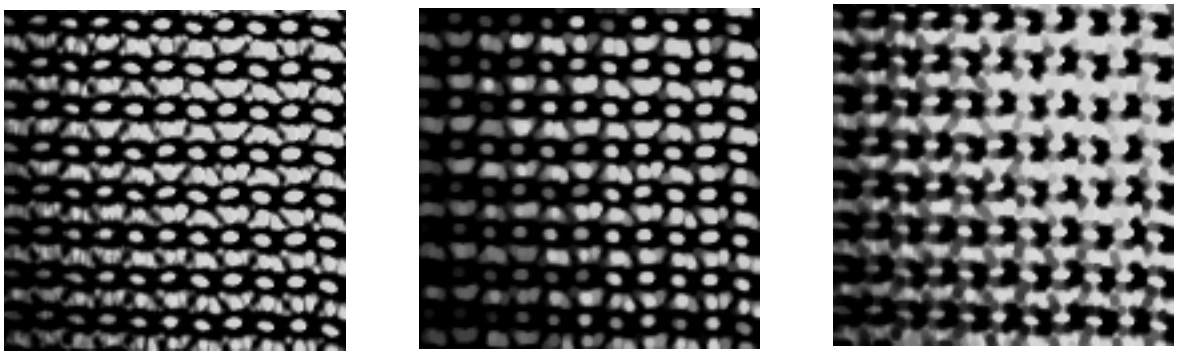
On peut écrire :

$$(X)_B = (X \ominus \check{B}) \oplus B = (X^c \oplus \check{B})^c \oplus B$$
$$= ((X^c \oplus \check{B}) \ominus B)^c = \left[(X^c)^B\right]^c, \text{ Q.E.D.}$$



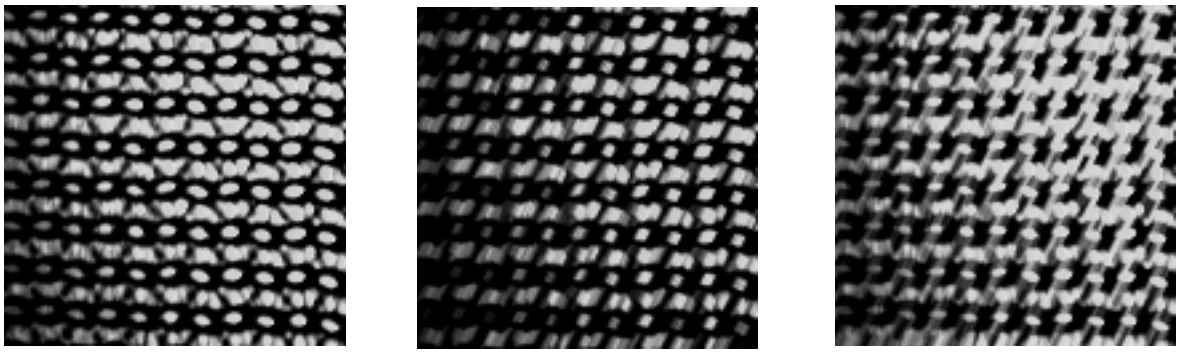
(a) (b)

Ouverture (a), fermeture (b) par un doublet de points



(a) (b) (c)

Ouvertures, fermetures numériques
(a) original, ouverture (b), fermeture (c) hexagonales



(a) (b) (c)

Ouvertures, fermetures numériques (suite)
(a) original, ouverture (b), fermeture (c) linéaires

Exercice n° 2

L'objet de cet exercice est de vous familiariser avec le comportement de l'ouverture et de la fermeture. Pour cela, vous avez toute latitude pour utiliser les transformations générées dans l'exercice précédent sur les images fournies (GRAINS1, GRAINS2, PARTIC1, PARTIC2, BALLS, METAL1, METAL2, SALT, KNITTING, MUSCLE). Pour vous aider dans votre quête, voici quelques directives :

1) Vérifiez le comportement granulométrique de l'ouverture, en effectuant des transformations de taille croissante sur les images GRAINS2, BALLS, SALT, KNITTING, MUSCLE.

2) La notion de granulométrie n'est pas liée à la notion de particule. Les fermetures permettent d'effectuer (par dualité) la granulométrie des pores, et mettent ainsi en évidence la répartition spatiale des composantes connexes d'un ensemble (images PARTIC2, SALT, KNITTING, MUSCLE).

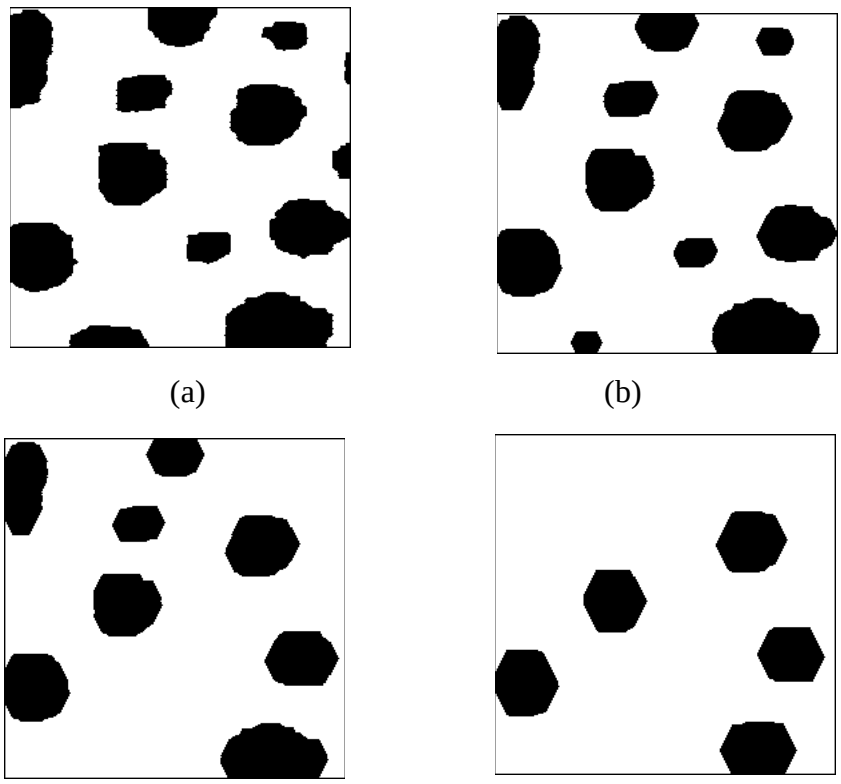
3) L'ouverture (comme la fermeture) "tamise" les composantes connexes d'un ensemble en fonction de leur taille, mais également de leur forme. Constatez-le en effectuant des ouvertures hexagonales de taille croissante sur l'image BALLS. En outre, l'ouverture "hexagonalise" les composantes connexes non éliminées.

Sur une image numérique, constatez également cet effet sélectif en effectuant des ouvertures par des segments de taille croissante (images CIRCUIT et BURNER).

[procédures **granul** ; **isogranul** ; Ch. 11]

Solution

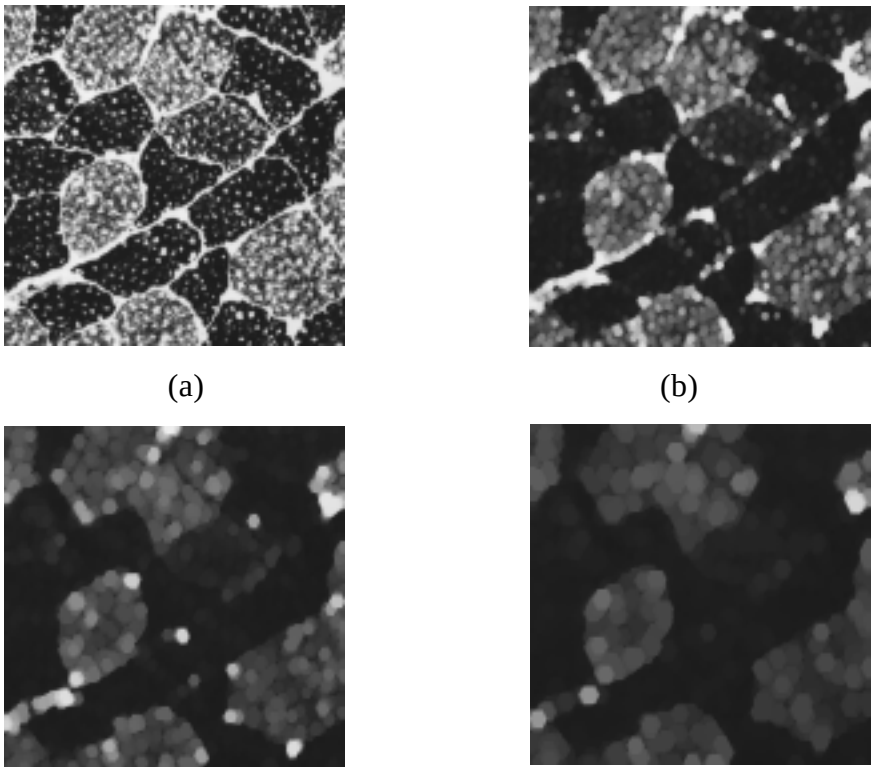
1) Propriétés granulométriques de l'ouverture par des homothétiques convexes.



(a) (b)

(c) (d)

Granulométrie par ouvertures
(a) original, (b) (c) (d), ouvertures de taille croissante



(a) (b)

(c) (d)

Granulométrie par ouvertures
(a) original, (b) (c) (d), ouvertures de taille croissante

Les images ci-dessus illustrent l'utilisation des ouvertures comme transformations granulométriques. Les transformées de tailles croissantes agissent comme un tamis.

2) Granulométries par fermeture

La fermeture, transformation duale de l'ouverture peut être utilisée pour effectuer la granulométrie de l'espace inter-particulaire. On utilise ainsi totalement une des particularités de la notion morphologique de granulométrie, qui est d'être une notion indépendante de la notion de particule (ou de composante connexe).

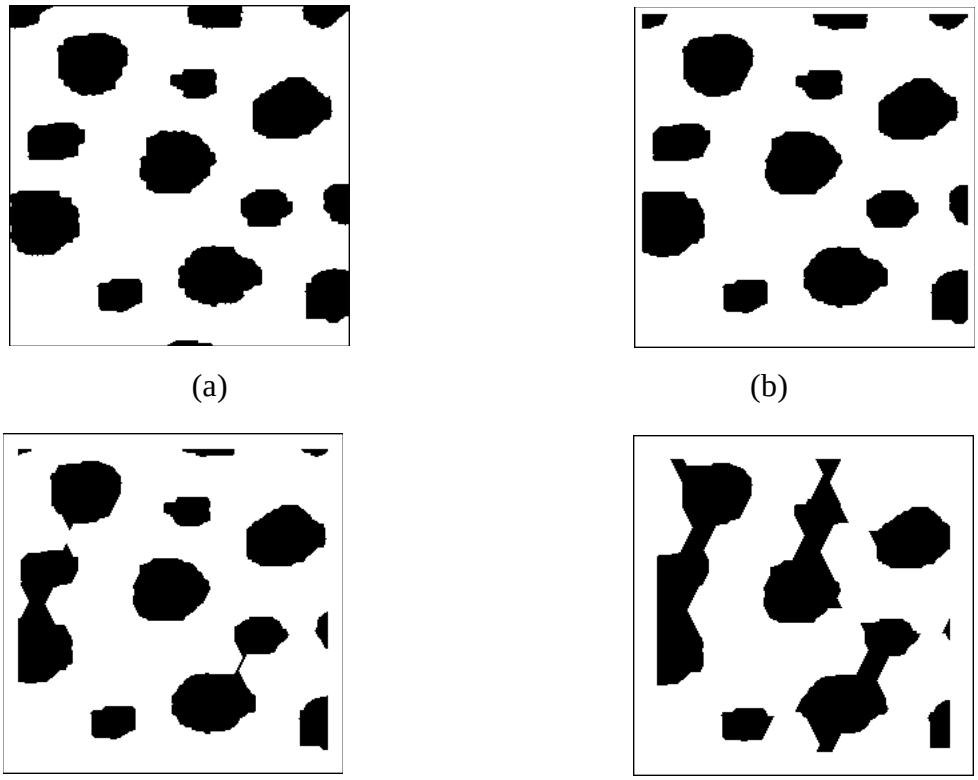
La procédure **granul** (dans le fichier curves.mic) permet le calcul de granulométries à la fois sur des images binaires ou numériques.

3) Influence de la forme sur l'ouverture

Pour vérifier l'influence de la forme de l'élément structurant sur l'ouverture, il convient de définir l'ouverture par des dodécagones.

Les procédures pour les images binaires et numériques sont données ci-après :

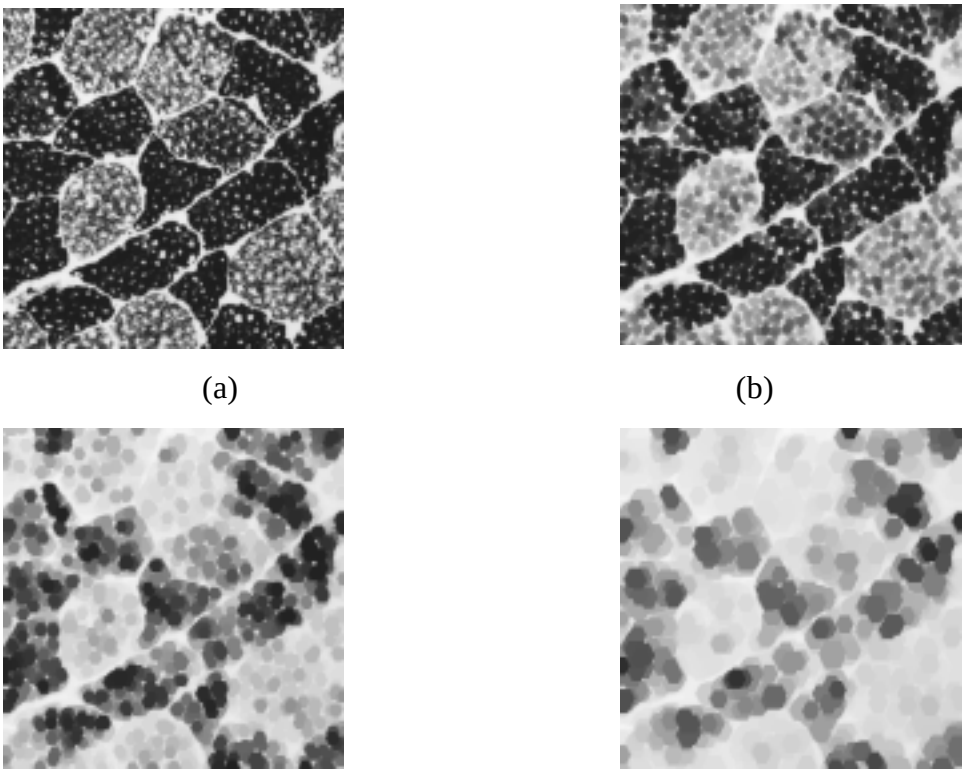
```
deproc ddcopen ddcopen s d sz
syntax "ddcopen binin binout size"
  int w ;
   w := imalloc imdepth s
   imcopy s d
   for 1 to sz do
   imcopy d w
   iminfngb 3 w d
   iminfngb 5 w d
   iminfngb 1 w d
```



(a) (b)

(c) (d)

Granulométrie par fermetures
(a) original, (b) (c) (d), fermetures de taille croissante



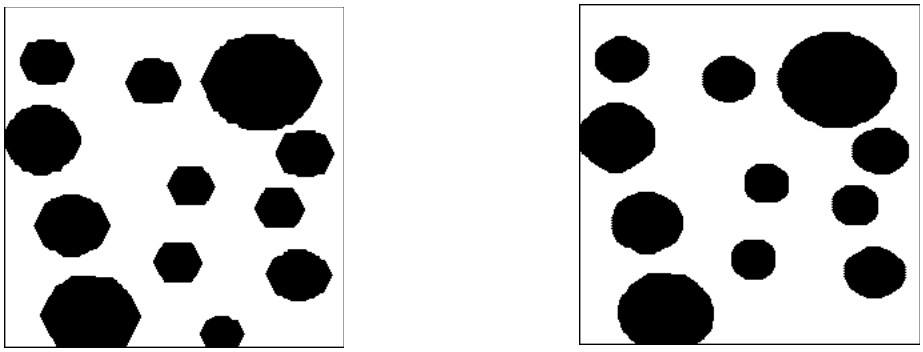
(a) (b)

(c) (d)

Granulométrie par fermetures
(a) original, (b) (c) (d), fermetures de taille croissante

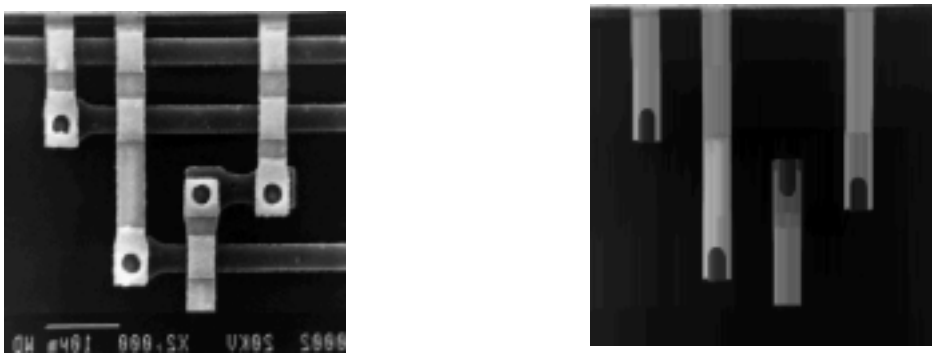
```
    imcopy d w
    iminfngb 4 w d
    iminfngb 6 w d
    iminfngb 2 w d
     ero d d 1
  end
  for 1 to sz do
    b1dil  d d
    b2dil  d d
    dil d d 1
  end
  imfree w
end
```

On constate en effectuant des ouvertures par des segments verticaux de grande taille sur l'image CIRCUIT que seules les pistes possédant une orientation donnée sont conservées.



(a) (b)

(a) ouverture hexagonale, (b) ouverture dodécagonale de taille équivalente



(a) (b)

(a) original, (b) ouverture par des segments verticaux de taille 50

Exercice n° 3

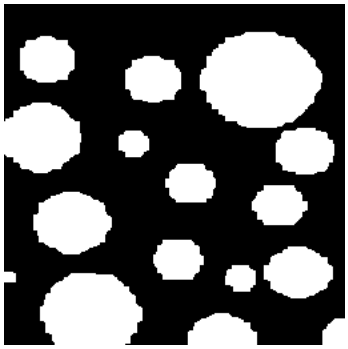
L'ouverture et la fermeture constituent de bonnes transformations pour filtrer le bruit des images. L'image NOISE représente un ensemble bruité par un nuage de points et l'image ELECTROP représente une image à teintes de gris bruitée.

1) Effectuez l'ouverture de l'image par un hexagone élémentaire. Même opération avec la fermeture.

2) Effectuez les deux opérations successivement. L'ordre des opérations intervient-il?

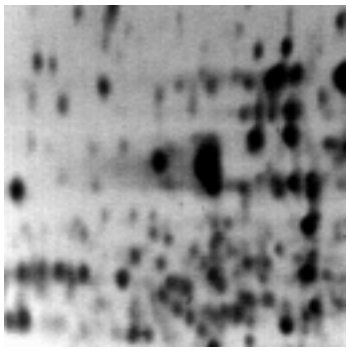
3) Pouvez-vous affiner le filtrage? (utilisation des ouvertures et fermetures triangulaires ou par des carrés 2x2).

[procédures **open** ; **close** ; **miniopen** ; **miniclose**]



BALLS

NOISE

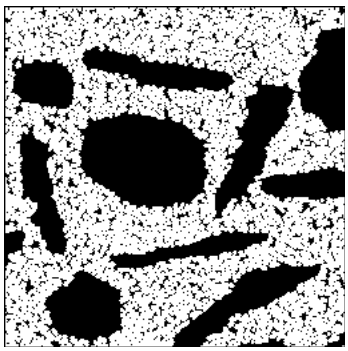


ELECTROP

Solution

1) Ouverture et fermeture hexagonales

Les images suivantes montrent les résultats des opérations d'ouverture et de fermeture hexagonales sur l'image NOISE.



Ouverture et fermeture hexagonales de l'image NOISE

2) Opérations successives

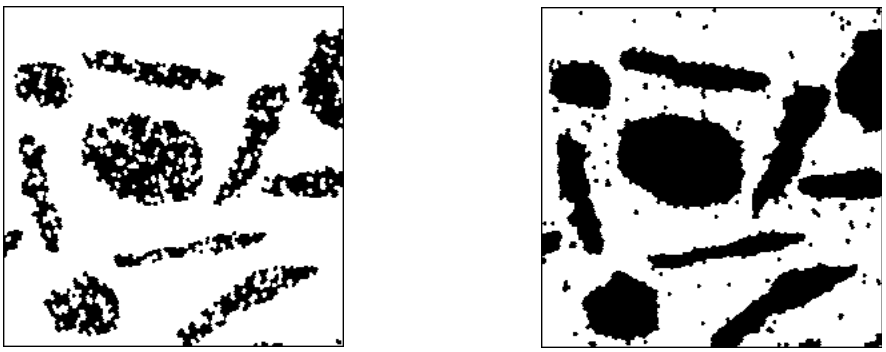
L'ordre des opérations intervient comme le montrent les deux images ci-dessous.

3) Utilisation des ouvertures et fermetures triangulaires ou carrées 2x2

```
deproc miniopen miniopen s d
syntax "miniopen binin binout"
  miniero s d
  minidil d d
end

deproc miniclose miniclose s d
syntax "miniclose binin binout"
```

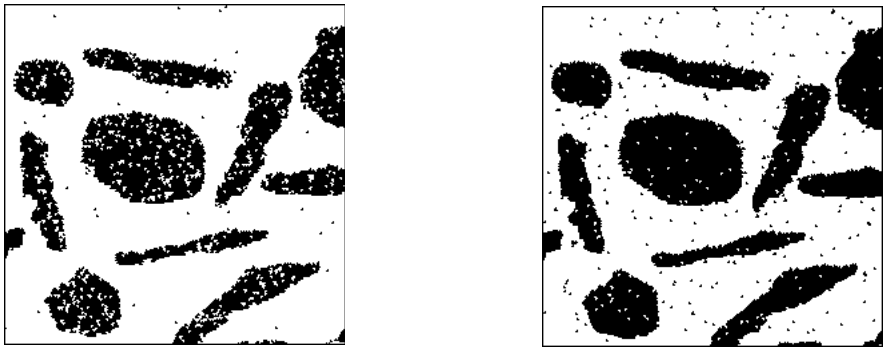
```
minidil s d
miniero d d
end
```

L'ordre intervient encore, mais de façon moins prononcée.



(a) (b)

(a) Ouverture puis fermeture hexagonale, (b) Ordre des opérations inversé



(a) (b)

(a) Ouverture puis fermeture triangulaires, (b) ordre inverse

Afin d'améliorer le filtrage, il conviendrait d'effectuer les mêmes opérations avec un triangle retourné.

Exercice n° 4 : Généralisation de la notion d'ouverture

La notion d'ouverture peut être définie de manière plus générale. On appelle ouverture toute transformation :

- croissante
- idempotente
- anti-extensive
- invariante par translation

On définit de manière duale la notion de fermeture : croissante, idempotente, extensive et invariante par translation.

1) Donnez des exemples d'ouvertures et de fermetures.

2) L'opération consistant à extraire les particules possédant au moins un trou est-elle une ouverture? (cas binaire)

3) Une façon d'obtenir de nouvelles ouvertures est de considérer une famille (γ_i) d'ouvertures. Montrez que :

$\gamma = \sup_i \gamma_i$ resp. $\gamma = \bigcup_i \gamma_i$ est une ouverture.
$\varphi = \inf_i \varphi_i$ resp. $\varphi = \bigcap_i \varphi_i$ est une fermeture.
Programmez γ pour la famille des ouvertures par des segments dans les trois directions de la trame et la fermeture correspondante.

4) Constatez sur les images CIRCUIT et TOOLS que les nouvelles ouvertures et fermetures ont un effet sélectif différent de celles vues jusqu'ici. Nous verrons plus loin (chapitre 5) comment construire encore d'autres ouvertures au comportement différent.
[procédures **lineopen** ; **lineclose** ; **infopen** ; **supclose**]

Solution

1) L'opération consistant à boucher les trous est une fermeture. L'opération consistant à extraire les particules dont la surface est supérieure à N est une ouverture. En revanche, l'opération consistant à extraire les particules dont la surface est inférieure à N n'est pas une ouverture (pas croissante).

En numérique, l'écrêtage d'une fonction au niveau λ (tout ce qui dépasse λ est ramené à la valeur λ) est une ouverture (c'est même une granulométrie).

2) Non, car elle n'est pas croissante. Il suffit de prendre une composante connexe X ne possédant pas de trou et de prendre $Y \subset X$ égal au contour intérieur de X.

3) Montrons que $\sup_i \gamma_i$ est une ouverture :

Croissance :
Soit $f < g$.

$\forall i, \gamma_i(f) \leq \gamma_i(g)$ car γ_t est une ouverture (donc croissante)
$\forall i, \gamma_i(f) \leq \sup_j \gamma_j(g)$
$\sup_i \gamma_i(f) \leq \sup_j \gamma_j(g)$ Q.E.D.

Anti-extensivité :

$\forall i, \gamma_i(f) \leq f$
donc $\left(\sup_i \gamma_i\right)(f) \leq f$ Q.E.D.

Idempotence :
Notons $\Psi = \sup_i \gamma_i$.

$\forall j, \Psi(f) \geq \gamma_i(f)$ par définition
$\forall j, \gamma_i(\Psi(f)) \geq \gamma_i(\gamma_j(f)) = \gamma_j(f)$
car γ_j est croissante et idempotente.
$\forall j, \left(\sup_k \gamma_k\right)(\Psi(f)) \geq \gamma_j(f)$ a fortiori.
on a alors $\forall j, \Psi(\Psi(f)) \geq \gamma_j(f)$
donc $\Psi(\Psi(f)) \geq \sup_j(\gamma_j(f))$
c'est-à-dire : $\Psi(\Psi(f)) \geq \Psi(f)$
d'où $\Psi(\Psi(f)) = \Psi(f)$ puisque Ψ est anti-extensive. Q.E.D.

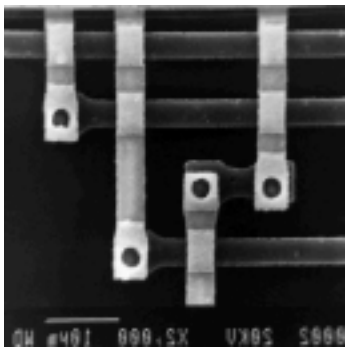
Invariance par translation :
évident puisque les g le sont.

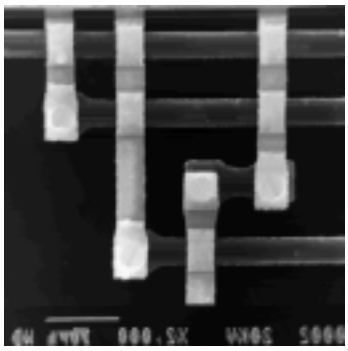
Voici, à titre d'illustration les procédures définissant des ouvertures (resp. des fermetures) à l'aide des familles d'ouvertures (resp. de fermetures) linéaires par des segments dans les différentes directions de la trame.

```
deproc lineopen lineopen s d sz
syntax "lineopen imin imout size"
  int w1 w2 i k;
   k := imdepth s
   w1 := imalloc k
   w2 := imalloc k
   imset impixmin w2 w2
   if (grid = 1) then
     i := 0
     for 1 to 3 do
       diropen ++i s w1 sz
        imsup w1 w2 w2
      end
   else
     i := 1
      for 1 to 2 do
         dirclose i s w1 sz
         iminf w1 w2 w2
         i := 3
       end
    end
    imcopy w2 d
    imfree w1
    imfree w2
 end

 deproc lineclose lineclose s d sz
 syntax "lineclose imin imout size"
    int w1 w2 i k;
    k := imdepth s
   w1 := imalloc k
   w2 := imalloc k
   imset impixmax w2 w2
   i := 1
   if (grid = 1) then
     for 1 to 3 do
       dirclose i s w1 sz
       iminf w1 w2 w2
       ++ i
     end
  else
     for 1 to 2 do
      dirclose i s w1 sz
      iminf w1 w2 w2
      i := 3
   end
 end
```

```
  imcopy w2 d
  imfree w1
  imfree w2
end
```

4) Les images suivantes montrent l'effet de fermetures isotropes comparé à celui de fermetures par intersection de fermetures linéaires. La différence de comportement est particulièrement évidente sur les objets allongés. La première transformation filtre les objets en fonction de leur épaisseur, alors que la seconde est contrôlée par l'allongement : un objet allongé est conservé, même s'il est étroit.





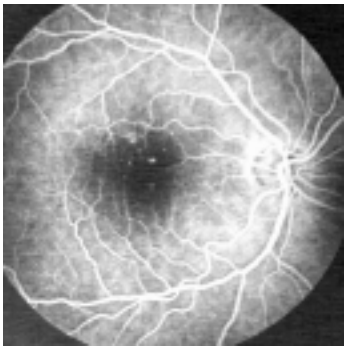
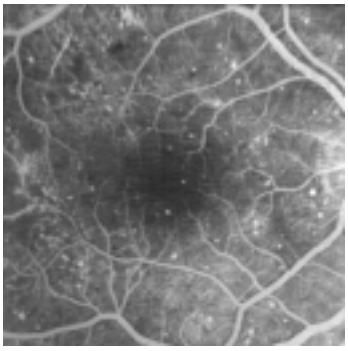
(a) (b) (c)

(a) original, (b) fermeture hexagonale de taille 20
(c) famille des fermetures linéaires de taille 20

Exercice n° 5

1) Programmez le chapeau haut-de-forme associé à l'ouverture par un hexagone de taille n et le chapeau haut-de-forme conjugué.

2) Appliquez ces transformations sur les images CIRCUIT, ELECTROP, GRAINS3, RETINA1 et RETINA2.

RETINA1 **RETINA2**

3) Vous remarquerez que ces opérations ne permettent pas de discriminer, sur les images RETINA1 et RETINA2, les vaisseaux des anévrismes (petites taches blanches). Trouvez un chapeau haut-de-forme permettant de les discriminer.
[procédures **openth** ; **closeth**]

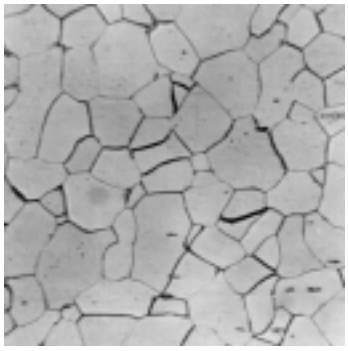
Solution

1) Voici les procédures des transformations chapeau haut-de-forme WTH(f) et chapeau haut-de-forme conjugué BTH(f) :

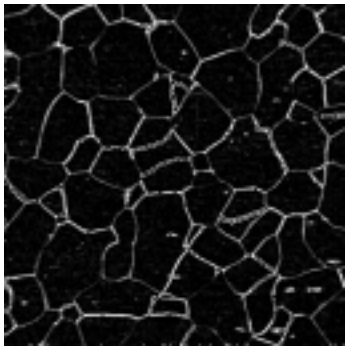
```
deproc openth openth s d sz
syntax "openth greyin greyout size : white top hat"
   int w;
  w := imalloc imdepth s
  open s w sz
  imsub s w d
  imfree w
end

deproc closeth closeth s d sz
syntax "closeth greyin greyout size : black top hat"
  int w;
  w := imalloc imdepth s
  close s w sz
  imsub w s d
  imfree w
end
```

2) Application à quelques images

On trouvera ci-après deux exemples illustrant ces transformations.

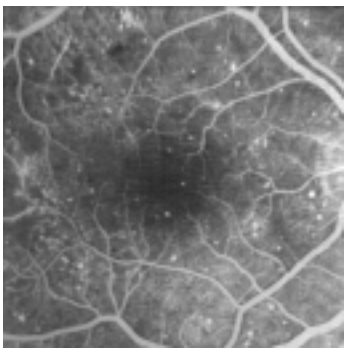


(a)

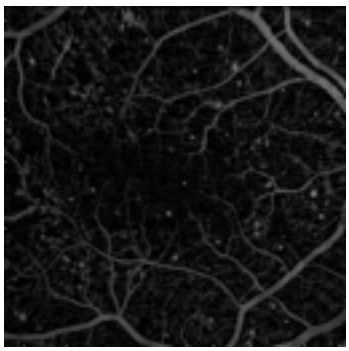


(b)

(a) original, (b) chapeau haut-de-forme par fermeture



(a)



(b)

(a) original, (b) chapeau haut-de-forme par ouverture

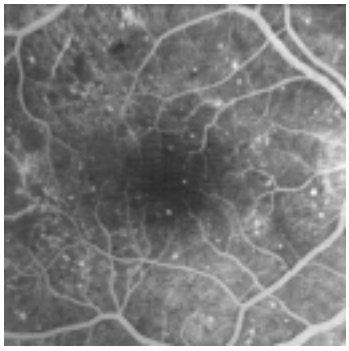
3) Le chapeau haut-de-forme basé sur l'ouverture hexagonale ne permet pas en effet de discriminer les vaisseaux des anévrismes. Cet opérateur extrait tout ce qui est plus étroit qu'une certaine taille : que l'objet soit long ou rond, il disparaît après l'ouverture. L'idée est alors d'utiliser une ouverture qui ne ferait disparaître que les objets ronds (par exemple). Or nous disposons d'un tel outil : l'ouverture obtenue comme sup de la famille des ouvertures par des segments.

On peut alors construire le chapeau haut-de-forme associé.

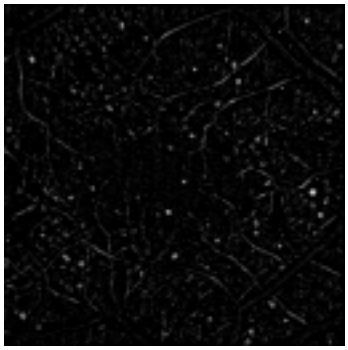
```
deproc lineopenth lineopenth s d sz
syntax "lineopenth greyin greyout size"
   int w;
  w := imalloc imdepth s
  lineopen s w sz
  imsub s w d
  imfree w
end
```

On peut définir également l'opération duale :

```
deproc linecloseth linecloseth s d sz
syntax "linecloseth greyin greyout size"
  int w;
  w := imalloc imdepth s
  lineclose s w sz
  greysub s w d
  imfree w
 end
```



(a)



(b)

(a) original, (b) chapeau haut-de-forme associé à des ouvertures par des segments

Exercice n° 6

Montrez que le chapeau haut-de-forme peut être utilisé pour homogénéiser le niveau de gris du fond de l'image. Utilisez l'image TOOLS.

Solution

Une ouverture de grande taille de l'image TOOLS (la taille doit être supérieure à l'épaisseur des objets) fera disparaître les particules tout en conservant le fond. Le chapeau haut-de-forme en soustrayant ce fond homogénéisera l'image. Il est alors possible de la seuiller

beaucoup plus facilement. Très souvent même, les procédures de seuillage automatique sont efficaces.

L'image ci-après illustre le résultat de ce chapeau haut-de-forme de grande taille. La valeur de cette taille n'est pas critique pourvu qu'elle soit supérieure à la taille des objets. Elle pourrait être déterminée automatiquement en analysant la courbe granulométrique.

(a)

(b)

(a) original, (b) chapeau haut-de-forme par ouverture de grande taille

RESUME

A l'issue de ces exercices, votre dictionnaire contiendra les transformations suivantes :

open s d sz
ouverture hexagonale ou carrée de taille sz de l'image s dans l'image d.
close s d sz
fermeture hexagonale ou carrée de taille sz de l'image s dans l'image d.
diropen dir s d sz
ouverture par un segment de taille sz dans une direction dir de l'image s dans l'image d.
dirclose dir s d sz
fermeture par un segment de taille sz dans une direction dir de l'image s dans l'image d.
lineopen s d sz
sup d'ouvertures linéaires de taille sz de l'image s dans l'image d.
lineclose s d sz
inf de fermetures linéaires de taille sz de l'image s dans l'image d.
openth s d sz
chapeau haut-de-forme associé à l'ouverture par un hexagone de taille sz de l'image numérique s dans l'image numérique d.
closeth s d sz
chapeau haut-de-forme associé à la fermeture par un hexagone de taille sz de l'image numérique s dans l'image numérique d.
lineopenth s d sz
chapeau haut-de-forme associé à la famille des ouvertures par des segments de taille sz de l'image numérique s dans l'image numérique d.
linecloseth s d sz
chapeau haut-de-forme associé à la famille des fermetures par des segments de taille sz de l'image numérique s dans l'image numérique d.

Chapitre 4

FILTRAGE MORPHOLOGIQUE

4.1. Filtres, rappels

Avant d'extraire les objets d'une image en teintes de gris, il est souvent nécessaire d'améliorer l'image. L'amélioration de l'image est essentiellement obtenue par une opération de filtrage. Un filtre morphologique est une transformation ϕ dotée des deux propriétés suivantes:

(i) ϕ est croissante

(ii) ϕ est idempotente.

4.1.1. Filtre alterné séquentiel

Le filtre alterné séquentiel (FAS) blanc (resp. noir) consiste, comme son nom l'indique, à alterner des ouvertures et des fermetures morphologiques (resp. des fermetures et des ouvertures) de tailles de plus en plus grandes. Soit γ_n une granulométrie et φ_n une anti-granulométrie, le filtre alterné séquentiel blanc de taille n d'une fonction f est défini par:

$$\phi_n(f) = \varphi_n\gamma_n\varphi_{n-1}\gamma_{n-1}...\varphi_2\gamma_2\varphi_1\gamma_1(f)$$

Le filtre alterné séquentiel noir de taille n de f est défini de même par :

$$\psi_n(f) = \gamma_n\varphi_n\gamma_{n-1}\varphi_{n-1}...\gamma_2\varphi_2\gamma_1\varphi_1(f)$$

4.1.2. Centre morphologique

4.1.2.1. Définition

Soit (Ψ_i) une famille de transformations croissantes. Posons :

$$\eta = \wedge\Psi_i \text{ et } \zeta = \vee\Psi_i$$

Le centre c de la famille (Ψ_i) est :

$$c = (I \vee \eta) \wedge \zeta$$

(I représente la fonction identité).

Le centre n'est pas un filtre (car pas idempotent), mais il est convergent quand on l'itère et sa limite est un filtre.

4.1.2.2. Exemples

1. pour $(\Psi_i) = (\gamma\varphi, \varphi\gamma)$:

$$c(f) = [f \vee (\gamma\varphi(f) \wedge \varphi\gamma(f))] \wedge (\gamma\varphi(f) \vee \varphi\gamma(f))$$

2. pour $(\Psi_i) = (\gamma\varphi\gamma, \varphi\gamma\varphi)$:

$$c(f) = [f \vee (\gamma\varphi\gamma(f) \wedge \varphi\gamma\varphi(f))] \wedge (\gamma\varphi\gamma(f) \vee \varphi\gamma\varphi(f))$$

Ce centre est encore appelé filtre automédian (c'est sa limite qui est un filtre).

4.2. Contrastes, rappels

Soient η une transformation anti-extensive et ξ une transformation extensive d'une fonction f. On appelle contraste à trois états de primitives η et ξ toute transformation κ telle que, pour tout f :

(i) $\kappa(f)(x)$ ne dépend que de $\xi(f)(x)$, $\eta(f)(x)$ et $f(x)$ et d'éventuelles constantes.

(ii) $\kappa(f)(x)$ ne peut prendre que l'une de ces trois valeurs (le choix dépendant d'une règle de décision).

En outre, si $\kappa(f)(x)$ ne peut pas prendre la valeur $f(x)$, le contraste est dit à deux états.

EXERCICES

Exercice n° 1

1) Vous avez déjà sans le savoir (chapitre 3, exercice n° 7) effectué un filtre alterné séquentiel sur l'image NOISE à l'aide d'ouvertures et fermetures hexagonales et triangulaires.

Poursuivre cet exercice :

- en augmentant le nombre d'itérations (taille du filtre).
- en prenant des tailles différentes pour les ouvertures et fermetures (par exemple : tailles des fermetures double de celles des ouvertures).

2) Programmez les filtres alternés séquentiels pour des images à teintes de gris en utilisant les diverses ouvertures et fermetures vues jusqu'ici (hexagonales, triangulaires, par sup d'ouvertures linéaires, etc.).

Testez-les sur les images RETINA3, BURNER et ELECTROP.

[procédures **asf** ; **lineasf** ; **buildasf**]

Solution

1) En augmentant la taille du filtre alterné séquentiel, des détails, aussi bien blancs que noirs, de tailles croissantes s'effacent progressivement. Les objets principaux sont conservés dans leur forme. Les contours, eux, subissent un lissage.

(a) (b) (c)

(d) (e) (f)

filtre alterné séquentiel blanc de taille (a) 1, (b) 4, (c) 3
filtre alterné séquentiel noir de taille (d) 1, (e) 4, (f) 3

En prenant des tailles différentes pour les ouvertures et les fermetures, on va privilégier le blanc par rapport au noir ou l'inverse selon le cas.

2) FAS avec diverses ouvertures et fermetures

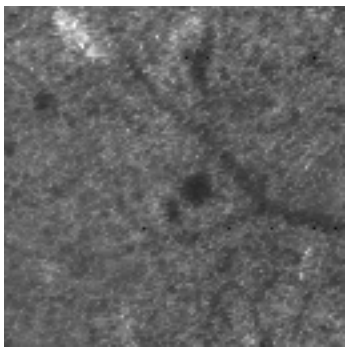
(a) ouvertures et fermetures par des hexagones

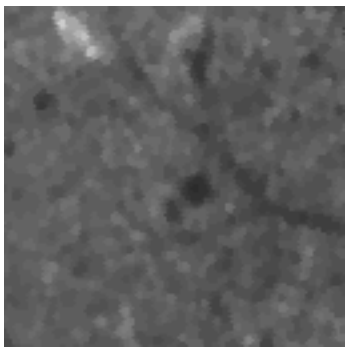
On définit d'abord le filtre alterné élémentaire:

```
deproc af af s d sz t
 syntax "af imin imout size type"
    if (t <> 1) then
      open s d sz
      close d d sz
    else
      close s d sz
      open d d sz
    end
 end
```

Puis le filtre alterné général :

```
deproc fullasf fullasf s d sz t
syntax "fullasf imin imout size type"
  int i ;
  imcopy s d
  i := 1
  for 1 to sz do
    af d d i t
    ++ i
  end
 end
```





(a) (b)

(a) Original (détail), (b) filtre alterné séquentiel blanc de taille 1

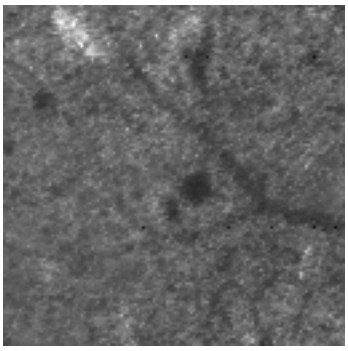
On remarque que plus les tailles sont élevées, plus le filtre est long sans que des modifications importantes apparaissent sur les images intermédiaires. C'est pourquoi on peut définir un filtre alterné séquentiel où la progression des tailles plus rapide :

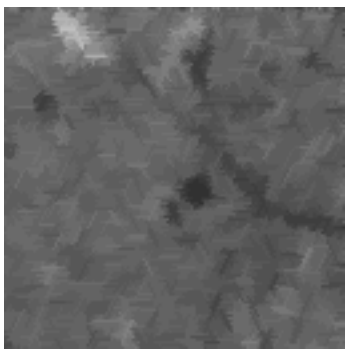
```
deproc asf asf s d sz  t
syntax "asf : imin imout size type"
```

```
  int i j  ;
   i := 1  j := 0
   imcopy s d
   while (i < (sz + 1)) do
     af d d i t
     j := ( j + 1 )
     i := ( i + j )
   end
 end
```

(b) ouvertures sup d'ouvertures linéaires et fermetures inf de fermetures linéaires

```
deproc lineaf lineaf s d sz t
syntax "lineaf imin imout size type"
  if (t <> 1) then
    lineopen s d sz
    lineclose d d sz
  else
   lineclose s d sz
   lineopen d d sz
 end
end

deproc lineasf lineasf s d sz  t
syntax "lineasf : imin imout size type"
   int i j  ;
   i := 1  j := 0
   imcopy s d
   while (i < (sz + 1)) do
     lineaf d d i t
     j := ( j + 1 )
     i := ( i + j )
   end
  end
```





(a) (b)

(a) Original (détail), (b) filtre alterné séquentiel blanc linéaire de taille 5

Exercice n° 2

Soit κ le contraste de primitives ξ et η et dont la règle de décision est la suivante :
si $\xi(f)(x) - f(x) \leq f(x) - \eta(f)(x)$
alors $\kappa(f)(x) = \xi(f)(x)$ sinon $\xi(f)(x) = \eta(f)(x)$

1) κ est un contraste à combien d'états?

2) Programmez κ :

a) pour η = érosion de taille N et ξ = dilatation de taille N.
b) pour η = ouverture morphologique de taille N et ξ = fermeture morphologique de taille N.
c) pour η = ouverture de taille N et ξ = fermeture de taille 5*N.

Appliquez ces transformations à l'image CHROMOSO.

3) Vérifiez que le contraste défini en 2-a n'est pas idempotent alors que celui défini en 2-b l'est.

[procédure **contrast**]

Solution

1) κ est un contraste à deux états (évident).

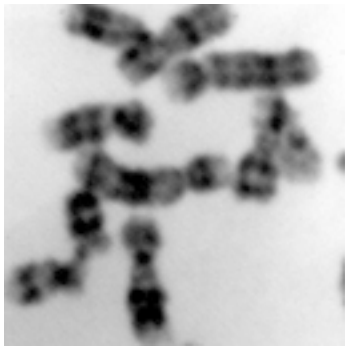
2) Programmation de κ

La procédure ci-dessous utilise selon la valeur du paramètre t l'une des trois combinaisons de transformations :

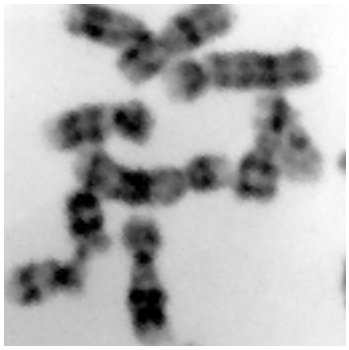
```
deproc contrast contrast s d sz t
 syntax "contrast greyin greyout size type (0: ero-dil, 1: open-close, 2:
close-5*open)"
  int w1 w2 w3 w4 bw ;
  w1 := imalloc imdepth s
  w2 := imalloc imdepth s
  w3 := imalloc imdepth s
  w4 := imalloc imdepth s
  bw := imalloc 1
  if (t > 0) then
   if (t > 1) then
    open s w1 sz
    close s w2 (5 * sz)
   else
    open s w1 sz
    close s w2 sz
   end
  else
   ero s w1 sz
   dil s w2 sz
  end
  imsub s w1 w3
  imsub w2 s w4
  imsub w4 w3 w3
  imthresh w3 1 impixmax w3 bw
  immask bw 0 impixmax d d
  iminf w1 d d
```

```
    iminv bw bw
    immask bw 0 impixmax w4 w4
    iminf w2 w4 w4
    imsup w4 d d
    imfree w1
    imfree w2
  imfree w3
  imfree w4
  imfree bw
end
```

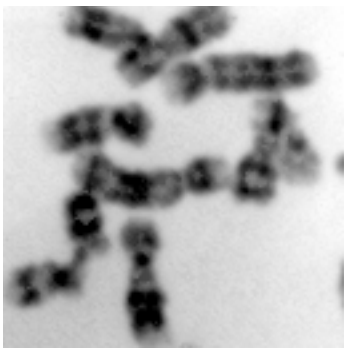
Les images ci-après illustrent ces diverses procédures.



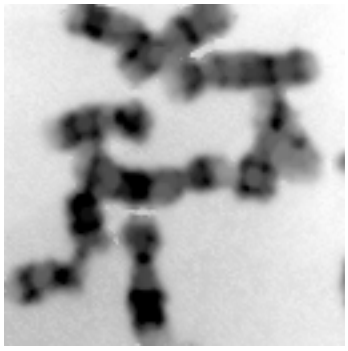
(a)



(b)



(c)



(d)

Contrastes : (a) image originale, (b) contraste dilatation/érosion de taille 1, (c) contraste ouverture/fermeture de taille 6, (d) contraste ouverture/fermeture dissymétrique de taille 5

3) Pour vérifier ou infirmer l'idempotence des différents contrastes définis plus haut, faire :

```
contrast g1 g2 1 0
contrast g2 g3 1 0
imsub g3 g2 g4
print imvolume g4
```

Puis :

```
contrast g1 g2 1 1
```

```
contrast g2 g3 1 1
imsub g2 g3 g4
print imvolume g4
```

Exercice n° 3

Soit la transformation définie par :

$$\kappa'(f) = 3f - \gamma(f) - \varphi(f)$$

(γ est une ouverture, φv est une fermeture).

1) κ' est-il un contraste au sens où il a été défini plus haut?

2) Programmez κ' et appliquez-le à l'image CHROMOSO.

Solution

1) On peut simplifier l'écriture de κ' en constatant que :

$$\kappa'(f) = 3f - \gamma(f) - \varphi(f) = f + (f - \gamma(f)) - (\varphi(f) - f)$$

On reconnaît les chapeaux haut de forme blanc et noir de f.

Si, pour tout f, nous définissons $\eta(f)$ et $\xi(f)$ ainsi :

$$\eta(f) = \min(f, f + (f - \gamma(f)) - \varphi(f) - f)$$
$$\xi(f) = \max(f, f + (f - \gamma(f)) - (\varphi(f) - f))$$

on a alors, $\forall f$:

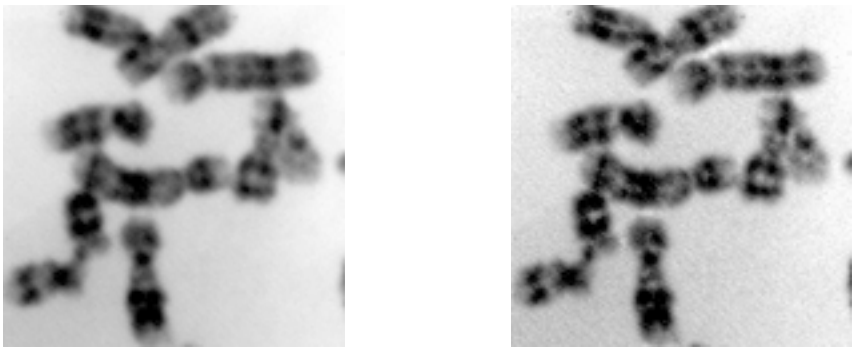
$$\eta(f) \leq f \text{ et } \xi(f) \geq f$$

η est donc anti-extensive et ξ extensive.

κ' s'identifie alors au contraste κ défini par :

$$\kappa(f)(x) = \eta(f)(x) \text{ si } \eta(f)(x) < f(x)$$
$$\kappa(f)(x) = \xi(f)(x) \text{ sinon.}$$

2) La procédure correspondante s'appelle Nthcontrast (voir ci-dessous sa définition).



(a) (b)

Contraste par chapeau haut-de-forme
(a) original, (b) transformée de taille 4

```
deproc contrasth contrasth s d sz
syntax "Nthcontrast greyin greyout size"
  int w1 ;
 w1 := imalloc imdepth s
 w := imalloc imdepth d
 imcopy s d
openth s w1 sz
```

```
    imadd w1 d d
   closeth s w1 sz
    imsub d w1 d
    imfree w1
  end
```

Exercice n° 4

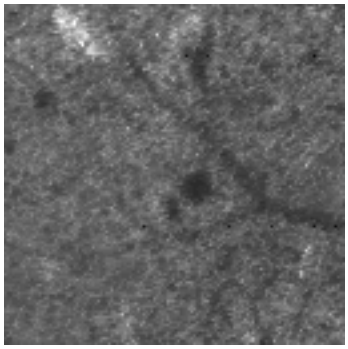
1) Programmez le filtre automédian (qui n'est pas un filtre au sens de la définition donnée plus haut) et appliquez-le aux images RETINA3 et BURNER.

2) Au bout de combien d'itérations a-t-on moins de 10 % de pixels modifiés? Essayez avec des tailles différentes.

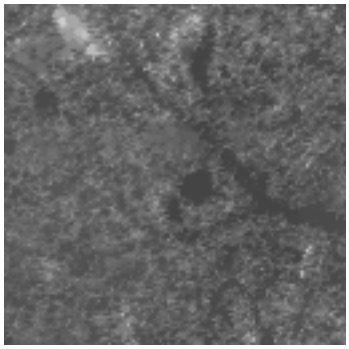
Solution

1) Filtre automédian

La procédure est fournie ci-après.



(a)



(b)

(a) Original (détail), (b) Filtre automédian de taille 5

```
deproc automed automed s d sz
 syntax "automed in out size"
   int oc co w sdepth;
   sdepth := imdepth s
   co := imalloc sdepth
   oc := imalloc sdepth
    w := imalloc sdepth
    af s oc sz 1
    af s co sz 0
    imcopy s d
    imcopy oc w
    imsup co w w
    iminf w d d
    imcopy oc w
    iminf co w w
    imsup w d d
    imfree w
    imfree co
    imfree oc
```

```
end
```

2) Pour comparer les images et calculer le nombre de points modifiés, on peut utiliser la procédure **masksup** fournie ci-dessous. On définit alors la procédure suivante où chaque frappe d'une touche différente de <Enter> lancera une itération et affichera le nombre de points modifiés.

```
deproc masksup masksup g1 g2 b
syntax "masksup greyin1 greyin2 mask_out"
  int w ;
  w := imalloc imdepth g1
  imdiff g1 g2 w
  imthresh w (impixmin g1 +1) impixmax g1 b
  imfree w
end

deproc iterautomed iterautomed s d sz
  syntax "iterautomed greyin greyout size"
    int w b1 b2 nb v ;
    b1 := imalloc 1
    b2 := imalloc 1
    w := imalloc imdepth s
    imcopy s w
    v := readkey
    while (v <> 13) do
      automed w d sz
      immasksup w d b1
      masksup d w b2
      imor b1 b2
      nb := imvolume b2
      print [ "nombre de pixels modifiés: " nb ]
      imcopy d w
      v := readkey
    end
    imfree b1
    imfree b2
    imfree w
end
```

Ainsi, avec un filtre de taille 1, l'image RETINA2 a 40852 pixels modifiés à la première itération, et aucun à la seconde. Par contre, pour l'image BURNER, les pixels modifiés sont au nombre de 32449 à la première itération, 1230 à la seconde, 326 à la troisième, 91 à la quatrième.

RESUME

Nous vous conseillons de conserver dans votre dictionnaire, les transformations suivantes :

af s d sz type
filtre alterné composé de **close** suivi de **open** (type 1) ou de **open** suivi de **close** (type 0)

fullasf s d sz type
filtre alterné séquentiel de taille sz de l'image numérique s dans l'image numérique d, commençant par la fermeture **close** de taille 1 si type=1, ou par l'ouverture **open** de taille 1 si type différent de 1.

lineasf s d sz type
filtre alterné séquentiel de taille sz de l'image numérique s dans l'image numérique d, commençant par la fermeture **lineclose** de taille 1 si type=1, ou par l'ouverture **lineopen** de taille 1 si type différent de 1.

contrast s d sz type
contraste taille sz de l'image numérique s dans l'image numérique d, par erosion et dilatation si type=0, par ouverture et fermeture si type=1, par ouverture et fermeture 5 fois plus grande si type>1.

contrasth s d sz
contraste par chapeau haut-de-forme blanc et noir de taille sz de l'image numérique s dans l'image numérique d.

automed s d sz
centre morphologique entre les deux filtres alternés **af** (sz) de type 0 et 1.

Chapitre 5

GEODESIE ET CONNEXITE

5.1. Rappels, cas binaire

Etant donné un ensemble X, on définit la distance géodésique entre deux points x et y de X comme la longueur du plus court chemin L(x,y) inclus dans X et joignant ces deux points. Ayant défini cette distance, on peut définir des boules de taille λ dans cette métrique, et par là, l'érosion et la dilatation d'un ensemble Y inclus dans X par une boule géodésique B.

Lorsque l'on travaille sur des ensembles digitalisés, on montre que la dilatation géodésique élémentaire est définie par :

$$D_X(Y) = (Y \oplus H) \cap X$$

De la même façon, l'érosion géodésique élémentaire est définie par :

$$E_X(Y) = X \cap [(Y \cup X^c) \ominus H]$$

H étant la boule digitale élémentaire (hexagone ou carré).

5.2. Cas numérique

Dans le cas numérique, l'espace géodésique considéré peut être soit un ensemble X, soit une fonction g (ce deuxième cas étant la généralisation immédiate de la notion binaire appliquée aux sous-graphes). Sur des ensembles digitalisés, on peut donner les définitions suivantes :

La dilatation géodésique de f dans l'ensemble X par un hexagone élémentaire centré en O est définie en tout point x par :

$$D_X(f)(x) = \sup_{\substack{b \in H \\ x + \overrightarrow{Ob} \in X}} \left(f\left(x + \overrightarrow{Ob} \right) \right) = \sup\{ f(x); x \in H_x \cap X \}$$

De même l'érosion :

$$E_X(f)(x) = \inf_{\substack{b \in H \\ x + \overrightarrow{Ob} \in X}} \left(f\left(x + \overrightarrow{Ob} \right) \right) = \inf\{ f(x); x \in H_x \cap X \}$$

La dilatation géodésique de f conditionnellement à la fonction g par un hexagone élémentaire centré en O est définie par :

$$D_g(f) = \inf(f \oplus H, g)$$

De même, l'érosion :

$$E_g(f) = \sup(f \ominus H, g)$$

NB : On notera que la dualité n'est pas du même type qu'en binaire. Ici, on a simplement remplacé f par m-f (où m = 255 pour les images 8 bits).

EXERCICES

Exercice n° 1

Ce premier exercice a pour but de programmer les érosions et les dilatations géodésiques de taille n définies ci-dessus. Veillez à élaborer l'algorithme de façon à ce que la transformation soit exacte dans le cas particulier où X est égal à D, le champ de mesures.
[procédures **gdsdil** ; **gdsero**]

Solution

Afin de programmer une érosion géodésique exacte, même quand le champ X est égal à D, il faut veiller à ce que **edge** soit toujours égal à 1.

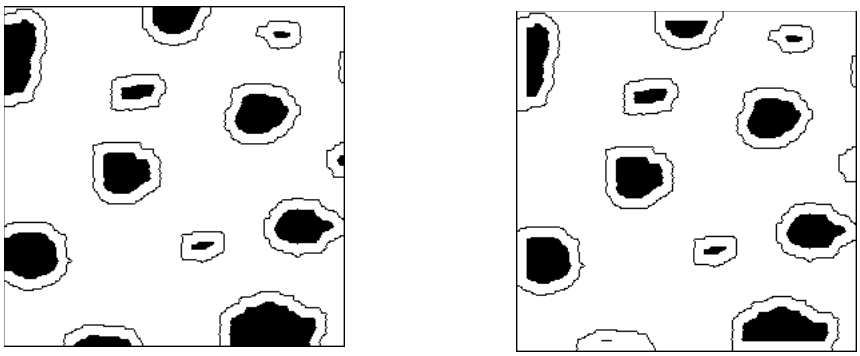
La définition des transformations peut être la suivante (m est l'ensemble géodésique, s l'ensemble de départ) :

```
deproc gdsdil gdsdil s m d sz
syntax "gdsdil in bin_or_grey_mask out size"
  int z w;
  z := imalloc imdepth d
  imcopy s z
  if ((imdepth m > 1) | (imdepth s = 1 )) then
    for 1 to sz do
      dil z z 1
      iminf m z z
    end
  else
    w := imalloc imdepth s
    immask m 0 (impixmax w) w
    iminf w z z
    for 1 to sz do
      dil z z 1
      iminf w z z
    end
    imfree w
  end
  imcopy z d
  imfree z
end
```

L'érosion peut s'obtenir par dualité (passage au complémentaire). La procédure est scindée en deux parties, une binaire et une numérique :

```
deproc bingdsero bingdsero s m d sz
syntax "bingdsero binin binmask binout size"
  int w ;
  w := imalloc imdepth d
  imcopy s w
  for 1 to sz do
    imdiff m w w
```

```
      dil w w 1
      imdiff m w w
    end
    imcopy w d
    imfree w
end

deproc greygdsero greygdsero s m d sz
syntax "greygdsero greyin bin_or_grey_mask greyout size"
    int w1 w2 ;
    w1 := imalloc imdepth s
    w2 := imalloc imdepth m
    iminv s w1
    iminv m w2
    gdsdil w1 w2 d sz
    iminv d d
    imfree w1
    imfree w2
end

 deproc gdsero gdsero s m d sz
 syntax " gdsero bin_or_greyin bin_or_grey_mask bin_or_greyout size"
    if (imdepth s > 1) then
      greygdsero s m d sz
    else
      bingdsero s m d sz
    end
 end
```



(a) (b)

Erodés géodésique et euclidien
(a) érodé géodésique (le masque est constitué du champ d'analyse)
(b) érodé euclidien (edge est égal à 0)

Exercice n° 2 : Reconstruction binaire

Soit un ensemble X constitué de plusieurs composantes connexes $\{X_i\}$. Un ensemble Y inclus dans X marque une ou plusieurs composantes connexes de X. Le présent exercice consiste à reconstituer les composantes connexes de X marquées par Y.

Ces composantes connexes sont constituées des points x de X qui sont à une distance géodésique d (x,Y) finie de Y.

Définissez l'algorithme de reconstitution à l'aide de dilatations géodésiques. Quel critère d'arrêt devez-vous utiliser?
[procédure **build**]

Solution

ms est l'ensemble initial, mq est l'ensemble marqueur et également l'ensemble contenant le résultat de la reconstruction.

```
deproc binbuild binbuild ms mq
syntax "binbuild binmask binmarker"
  int a b ;
  b := imvolume mq
  while (a <> b) do
    a := b
   gdsdil mq ms mq 1
    b := imvolume mq
  end
end
```

Le critère d'achèvement de la reconstruction est la non-augmentation de surface de l'ensemble reconstruit.

Exercice n° 3 : Reconstruction numérique

La fonction **build**, dilatation géodésique de f conditionnellement à g jusqu'à idempotence, est déjà implantée efficacement dans MICROMORPH.

1) Testez cette fonction sur l'image TOOLS en prenant comme fonction marqueur une image partout à 0 sauf en un point (choisi de préférence à l'emplacement d'un objet) où la valeur sera mise à 255.

2) Effectuez la même opération sur l'image RETINA1 en plaçant le point sur le réseau sanguin.
[procédure **build**]

Solution

1) Reconstruction numérique

On effectue les opérations suivantes après avoir chargé l'image TOOLS dans g1.

```
imset 0 g2
imwritepix 255 118 155 g2
build g1 g2
```

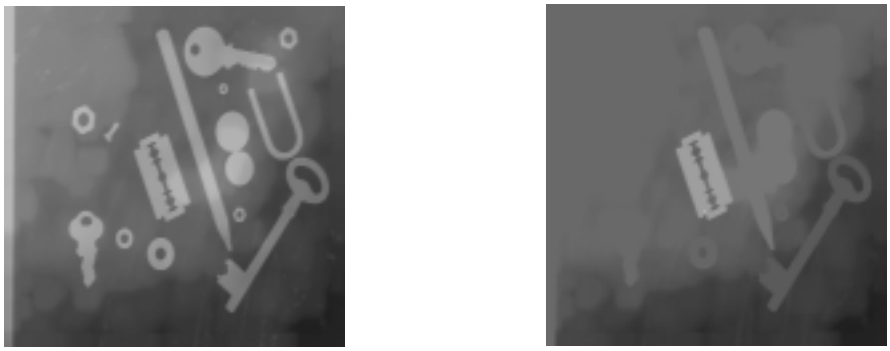
On constate ici un comportement semblable à la reconstruction binaire, seul l'objet marqué est reconstruit.

2) Le même effet peut être constaté sur l'image RETINA1 (chargez l'image RETINA1 dans g1).

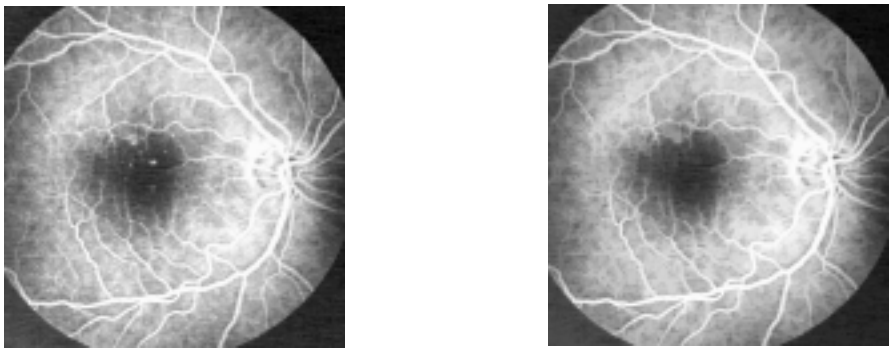
```
imset 0 g2
```

imwritepix 255 210 145 g2
build g1 g2

On remarque ici que les "taches" blanches situées au centre ne sont pas reconstruites. En effet, elles ne sont pas connectées au réseau. "x et y sont connectés" signifie ici : il existe un chemin descendant joignant un point x du marqueur à un point y du masque.



(a) (b)
(a) original, (b) résultat de la reconstruction numérique



(a) (b)
(a) original, (b) résultat de la reconstruction numérique

Exercice n° 4 : Ouverture par érosion - reconstruction

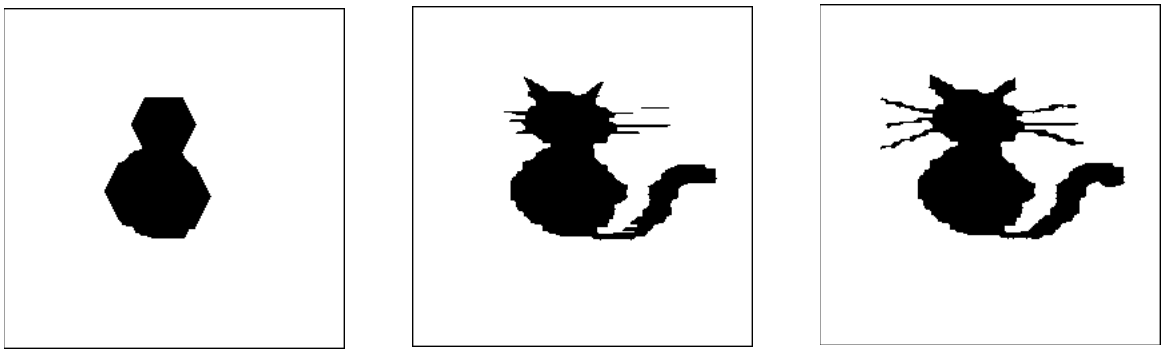
Montrez que la reconstruction par dilatation géodésique de l'érodé de f (resp. X) de taille n par l'élément structurant B conditionnellement à f (resp. X) est une ouverture (le centre de l'élément structurant B doit être un point de B).

Programmez ces opérations, ainsi que les fermetures duales et comparez-les sur les images RETINA1, RETINA2 et CAT aux ouvertures déjà vues jusqu'ici.
[procédures **buildopen** ; **buildclose**]

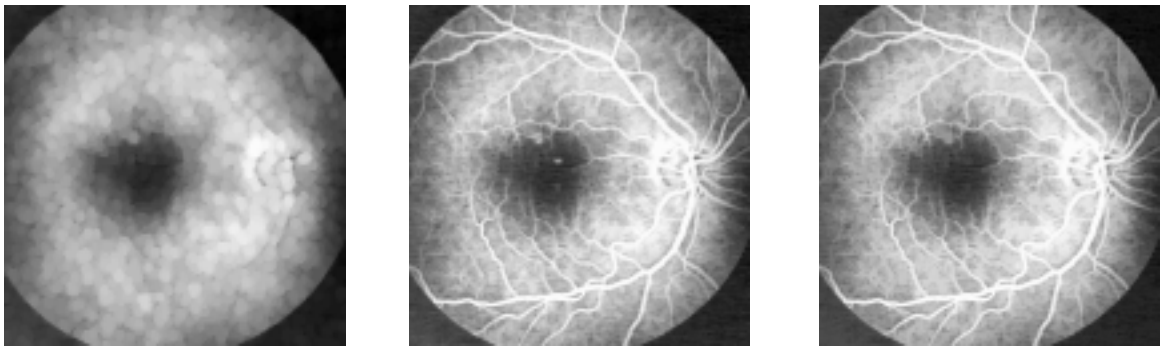
Solution

Notons $\Psi_0(X) = X \ominus nH$, $\Psi_m(X) = (\Psi_{m-1}(X) \oplus H) \cap X$.
On appellera Ψ la limite de Ψ_m lorsque $m \to \infty$.
Croissance :
Soit $Y \subset X$. $Y \ominus nH \subset X \ominus nH$ car l'érosion est croissante.
$(Y \ominus nH) \oplus H \subset (X \ominus nH) \oplus H$ car la dilatation est croissante.
$((Y \ominus nH) \oplus H) \cap Y \subset ((Y \ominus nH) \oplus H) \cap X \subset ((X \ominus nH) \oplus H) \cap X$ car $Y \subset X$.
Donc $\Psi_1(Y) \subset \Psi_1(X)$. Par itération, on aboutit à $\Psi_m(Y) \subset \Psi_m(X)$. En appelant $\Psi(X)$, la limite de cette itération lorsque $m \to \infty$, on a $\Psi(Y) \subset \Psi(X)$.

anti-extensivité :
évident.



(a) (b) (c)
(a) Ouverture hexagonale, (b) Sup des ouvertures linéaires, (c) Ouverture par érosion-reconstruction



(a) (b) (c)
(a) Ouverture hexagonale, (b) Sup des ouvertures linéaires, (c) Ouverture par érosion-reconstruction

idempotence :
Montrons que $\Psi(\Psi(X)) = \Psi(X)$.
Pour cela, nous allons montrer que $X \ominus nH \subset \Psi(X) \ominus nH$. On en déduit en effet immédiatement que $\Psi(\Psi(X)) = \Psi(X)$, Ψ étant anti-extensive et l'érosion étant croissante ($\Psi(X) \ominus nH \subset X \ominus nH$).
Il nous reste à montrer que $X \ominus nH \subset \Psi(X) \ominus nH$.
Montrons préalablement que :
1. $\Psi_m(X) \subset \Psi(X)$
2. $\forall m \leq n, \Psi_m(X) = (X \ominus nH) \oplus mH$
3. $((X \ominus nH) \oplus nH) \ominus nH = X \ominus nH$
1. Le centre de H étant pris dans H, alors $\forall m, \Psi_m(X) \subset \Psi_{m+1}(X)$. Par passage à la limite on en déduit que $\Psi_m(X) \subset \Psi(X)$.
2. Soit $m \leq n$. On a :

$$(X \ominus nH) \oplus mH \subset (X \ominus nH) \oplus nH \subset X$$

Donc $\Psi_m(X) = (X \ominus nH) \oplus mH$, l'intersection avec X à chaque étape revenant à ne rien faire.
3. $\subset$: car l'ouverture est anti-extensive.
$\supset$: car la fermeture est extensive.
Considérons maintenant m>n. $\Psi n(X) \subset \Psi_m(X)$, donc :

$\Psi_n(X) \ominus nH \subset \Psi_m(X) \ominus nH$ et $\Psi_n(X) \ominus nH = ((X \ominus nH) \oplus nH) \ominus nH$ (propriété 2).
Soit encore (propriété 3) : $\Psi_n(X) \ominus nH = X \ominus nH$.
Donc, $\forall m > n, X \ominus nH \subset \Psi_m(X) \ominus nH$.
Par passage à la limite, on a $X \ominus nH \subset \Psi(X) \ominus nH$ Q.E.D.

Les transformations sont les suivantes :

```
deproc buildopen buildopen s d sz
syntax "buildopen binin binout size"
  int w ;
  w := imalloc imdepth s
  ero s w sz
  build s w
  imcopy w d
  imfree w
end

deproc buildclose buildclose s d sz
syntax "bbuildclose binin binout size"
  int w ;
  w := imalloc imdepth d
  iminv s s
  buildopen s w sz
  iminv s s
  iminv w d
  imfree w
end
```

RESUME

Ce chapitre vous a permis d'introduire dans le dictionnaire les transformations suivantes :

gdsdil s m d sz
dilatation géodésique de taille sz de l'image s selon le masque m, résultat dans l'image d.
gdsero s m d sz
érosion géodésique de taille sz de l'image s selon le masque m, résultat dans l'image d.
build m imout
reconstruction par dilatation géodésique infinie de imout dans l'image m.
buildopen s d sz
reconstruction par dilatation géodésique de l'érodé de taille sz de l'image s. Le résultat est placé dans l'image d.
buildclose s d sz
transformation duale de buildopen.

Chapitre 6

APPLICATIONS DE LA GEODESIE

La première application de la géodésie est la reconstruction (binaire et numérique). Elle a déjà été introduite au chapitre précédent. D'autres applications seront introduites maintenant, beaucoup d'entre elles faisant usage de la reconstruction.

EXERCICES

Exercice n° 1 : Analyse individuelle de particules

1) Elaborez un algorithme d'analyse individuelle de particules, c'est-à-dire capable d'extraire d'une image, chaque composante connexe afin de la mesurer.

2) Application à la mesure des surfaces des grains de l'image PARTIC1.

3) Vérifiez que la transformation $\Psi(X)$ définie de la façon suivante :

$$X = \bigcup_{i=1}^{n} X_i,\ X_i \text{ composante connexe de } X$$

$$\Psi_\lambda(X) = \cup X_j \text{ tels que } mes(X_j) > \lambda$$

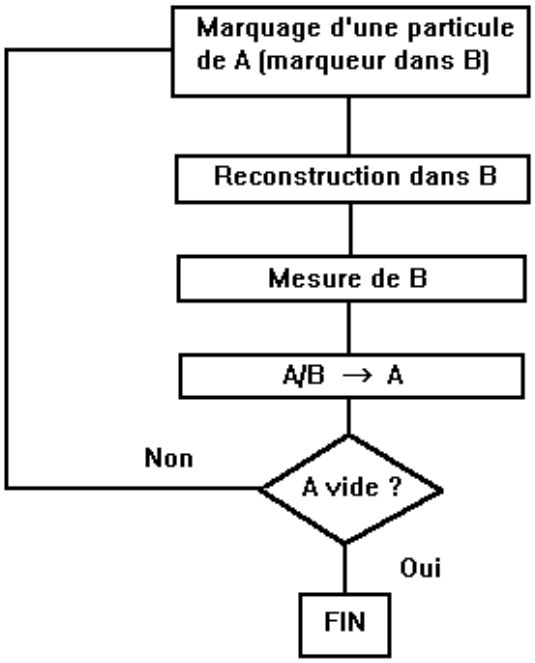
est une transformation granulométrique.

[procédure **fgrain**]

Solution

1) Analyse individuelle des particules

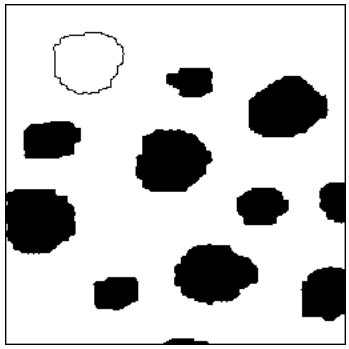
L'analyse individuelle de particules consiste à marquer successivement chaque particule d'un ensemble et à la reconstruire. Ceci fait, on peut effectuer un certain nombre de mesures



sur elle, avant de l'éliminer et de recommencer avec la particule suivante, ceci jusqu'à élimination totale des particules de l'ensemble initial.

L'algorithme correspondant à cette opération est écrit dans l'ordinogramme donné ci-dessus. La procédure MICROMORPH équivalente est donnée plus loin appliquée à la mesure de la surface.

2) Mesure de l'aire des grains de PARTIC1



Reconstruction et élimination d'une particule

```
deproc fgrain fgrain s d
syntax "fgrain binin binout:in d, first particle of s "
  int w flag ;
  w := imalloc imdepth d
  imset 0 w
  flag := imcompare s w w
  if (flag <> -1) then
    build s w
    imdiff s w s
  end
  imcopy w d
  imfree w
end
```

Le marquage s'effectue à l'aide de la transformation **imcompare** qui extrait le premier point de la première particule rencontrée dans l'image.

Pour lancer l'opération sur l'image PARTIC1, effectuez l'opération suivante, après avoir chargé l'image dans b1 :

```
fgrain b1 b2
print imvolume b2
```

L'opération itérée sur toutes les particules de l'image vous permet d'obtenir le tableau de résultats ci-après.

3) $\Psi_\lambda(X)$ est une granulométrie :

- Elle est idempotente : En effet, ne conserver d'un ensemble de composantes connexes que celles de surface supérieure à λ, puis recommencer l'opération sur ce nouvel ensemble ne modifie en rien le résultat.
- Quand $\lambda \geq \mu$, $\Psi_\lambda(X) \subset \Psi_\mu(X)$ (évident).

- Enfin : $\Psi_\lambda[\Psi_\mu(X)] = \Psi\mu[\Psi\lambda(X)] = \Psi_{\sup(\lambda,\mu)}(X)$.

N° particule	Aire
1	408
2	658
3	455
4	1 983
5	620
6	2 068
7	1 038
8	3 109
9	517
10	876
11	2 150
12	2 093
13	1 170
14	720
15	144

Exercice n° 2 : Bouchage de trous (binaire et numérique), objets en bord de champ

1) Appliquez l'algorithme de reconstruction géodésique à la suppression des particules coupant le bord du champ. Quel peut être en particulier l'ensemble marqueur Y? Application à l'image GRAINS2.

2) Comment boucher les trous des particules? Elaborez un algorithme et testez-le sur les images HOLES et GRUYERE.

3) Sur une image à teinte de gris on peut appeler trou ce que l'on entend naturellement par cuvette si l'on considère l'image comme un relief. On peut alors imaginer de boucher ces trous, tout comme le ferait la pluie en remplissant d'eau les cuvettes, le surplus d'eau se déversant en dehors des limites de l'image. De façon semblable au cas binaire, élaborez un algorithme de bouchage de trous sur des images à teinte de gris et appliquez-le aux images CIRCUIT et TOOLS.

[procédure **cloholes**]

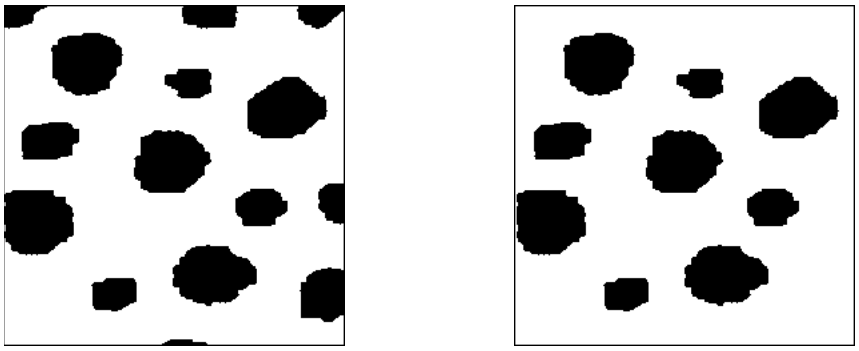
Solution

1) Suppression des particules coupant le bord du champ

L'ensemble marqueur est constitué de l'intersection de l'ensemble initial s et du contour du champ.

```
deproc  border border z
syntax "border binout (z = border of the bin-image field)"
  int k ;
  k := edge
  imsetedge 0
 imset impixmax z z
 iminfngb z z 1 1
 iminfngb z z 3 1
 iminfngb z z 5 1
 if ( grid = 0 ) then
```

```
        iminfngb z z 7 1
      end
      iminv z z
      imsetedge k
    end

    deproc edgeoff edgeoff s d
    syntax "edgeoff binin binout"
      int w ;
      w := imalloc imdepth d
      border w
      iminf s w w
      build s w
      imdiff s w w
      imcopy w d
      imfree w
    end
```



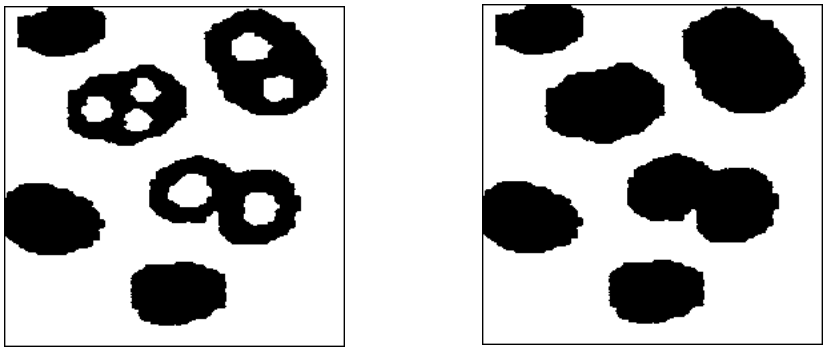
(a) (b)

Suppression des objets coupant le bord du champ
(a) original, (b) résultat de la transformation

2) Bouchage des trous (image binaire)

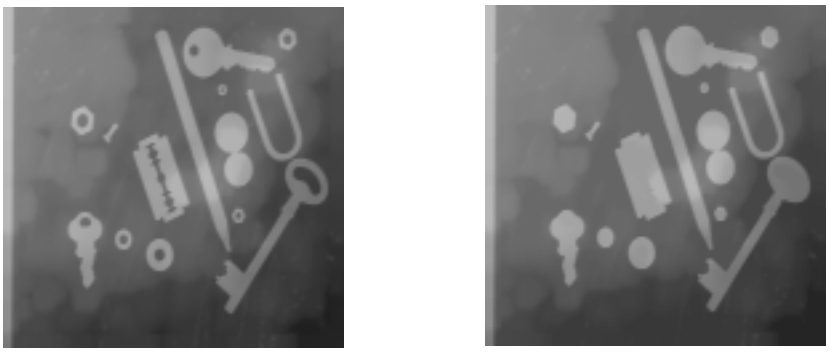
L'ensemble de départ de la reconstruction géodésique est l'ensemble complémenté. Les trous deviennent alors les composantes connexes inaccessibles à partir du bord du champ. Le marqueur est le même que dans l'exemple précédent :

```
  deproc clohole clohole s d
  syntax "clohole binin binout"
    int w ;
   w := imalloc imdepth s
    iminv s s
    border w
   iminf s w w
   build s w
   iminv s s
   iminv w d
   imfree w
 end
```



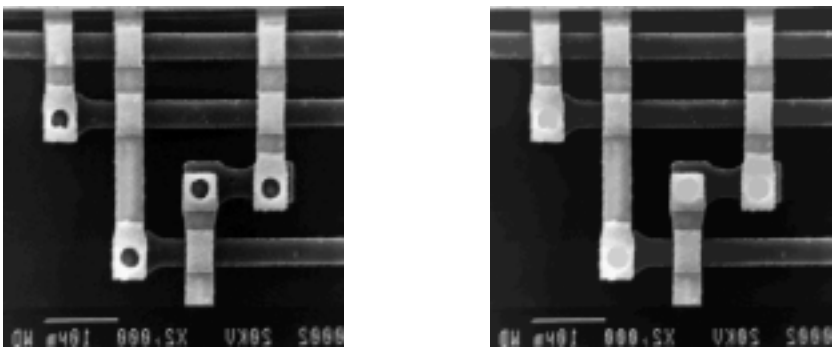
(a) (b)

Bouchage des trous
(a) original, (b) objets sans trous



(a) (b)

Bouchage des trous, (a) original, (b) objets sans trous



(a) (b)

Bouchage des trous, (a) original, (b) objets sans trous

3) Bouchage des trous (image numérique)

Seule la reconstruction par dilatation est implantée dans MICROMORPH. Il nous faut donc inverser l'image. Celle-ci constituera notre masque. Le marqueur utilisé est une image à valeur 255 sur les bords et 0 partout ailleurs.

Exercice n° 3 : Maxima - Minima régionaux (numérique)

Soient p_0 ,...,p_n des points de l'image I. $V(p_i)$ est la valeur de l'image en p_i .

p_0 p_1 ...p_n est appelé chemin si, $\forall i$, p_{i+1} et p_i sont voisins.

p_0 ...p_n est dit strictement ascendant si, $\forall i$, $V(p_{i+1}) \geq V(p_i)$ et $V(p_0) < V(p_n)$.

p_0 ...p_n est dit strictement descendant si, $\forall i$, $V(p_{i+1}) \leq V(p_i)$ et $V(p_0) > V(p_n)$

Un ensemble X connexe est un maximum régional s'il n'existe pas de chemin strictement ascendant issu de X :

$\forall x \in X, \forall (p_i)_{i\in\langle 1...N\rangle} \in I^N, xp_0...p_N$ n'est pas strictement ascendant.

1) Trouvez un algorithme permettant de déterminer les maxima régionaux utilisant les seuils successifs de l'image et l'opération de reconstruction binaire.

2) Dans la pratique, on n'utilise pas cette méthode mais la suivante : on soustrait 1 de l'image et on reconstruit l'image obtenue conditionnellement à l'image initiale. Les maxima régionaux sont là où l'image obtenue diffère de l'image originale. L'algorithme dual permet d'obtenir les minima régionaux.

Programmez cette transformation. Appliquez-la à l'image ELECTROP. Que constatez-vous? Comment l'expliquez-vous? Quelle solution pouvez-vous apporter?

3) La méthode algorithmique ci-dessus permet une généralisation de la notion de minima et maxima régionaux. On obtient les maxima régionaux étendus de hauteur h en enlevant la constante h à l'image initiale. Les maxima régionaux étendus sont là où l'image obtenue diffère de l'image originale. L'algorithme dual permet d'obtenir les minima régionaux étendus. Programmez cette transformation. Appliquez-la pour des hauteurs croissantes à l'image ELECTROP.

[procédures **maxima** ; **minima** ; **extmaxima** ; **extminima**]

Solution

1) Notons S le seuil entre n et 255 de l'image f :

$S_n(x) = 1$ si $f(x) \geq n$

$S_n(x) = 0$ si $f(x) < n$.

Démontrons d'abord le lemme suivant :

Lemme : si S_n contient une composante connexe C qui est absente dans S_{n+1} , alors C est un maximum régional.

Preuve : $\forall x \in C$, $f(x)=n$ puisque $f(x)<n+1$ ($x \notin S_{n+1}$) et $f(x)>n$ ($x \in S_n$). C étant une composante connexe de S_n , pour tout y voisin d'un point de C, $S_n(y)=0$, donc $f(y)<n$. Il ne peut donc exister de chemin strictement ascendant issu d'un point de C.

Il suffit donc, pour chaque seuil S_n, de reconstruire les composantes connexes marquées par le seuil S_{n+1} et de conserver les composantes connexes qui n'ont pas été reconstruites.

D'où l'algorithme :

```
deproc threshmax threshmax s d
syntax "threshmax greyin binout -> maxima successive thresholds"
  int level lev1 sn snplus1;
 sn := imalloc 1
 snplus1 := imalloc 1
 imset 0 d
 level := impixmax s
 lev1 := level
 imthresh s level level sn
 for 1 to lev1 do
   -- level
   imcopy sn snplus1
   imthresh s level lev1 sn
   binbuild sn snplus1
   imdiff sn snplus1 snplus1
```

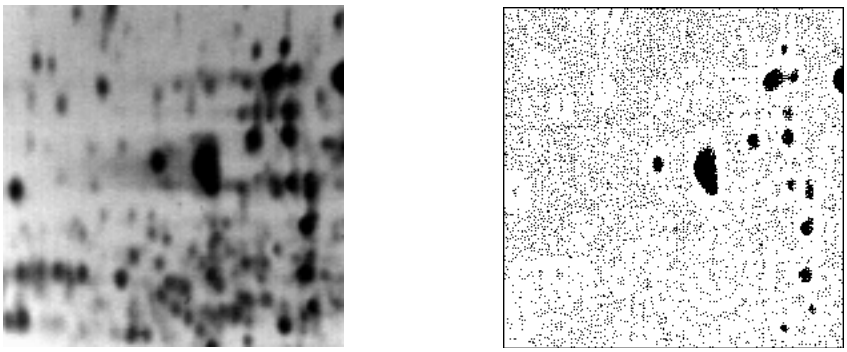
```
    imor snplus1 d d
  end
  imfree sn
  imfree snplus1
end
```

Cette méthode est coûteuse puisqu'il faut déterminer tous les seuils de l'image.

2) Algorithme par reconstruction directe

L'algorithme par reconstruction directe utilise la procédure **build** implantée dans MICROMORPH.

```
deproc maxima maxima s d
syntax "maxima greyin binout"
  int w;
  w := imalloc imdepth s
  imcopy s w
  imcsub w 1 w
  build s w
  imsub s w w
  imthresh w 1 impixmax s d
  imfree w
end
```



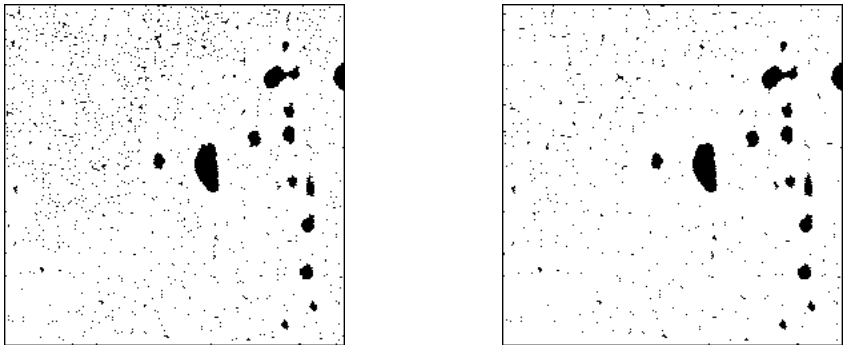
(a) (b)

Minima par reconstruction directe
(a) image originale, (b) minima

```
deproc minima minima s d
syntax "minima greyin binout"
  iminv s s
  maxima s d
  iminv s s
end
```

On constate ici que l'image d'électrophorèse possède un grand nombre de minima régionaux, le plus souvent réduits à un seul point. Ceci est caractéristique d'une image bruitée. On peut alors appliquer les filtres que nous avons vus pour supprimer, au moins en partie, le

bruit d'une image. Effectuons par exemple une ouverture hexagonale et déterminons les nouveaux minima. Le résultat d'une telle opération est illustré sur les images suivantes.



(a) (b)

Minima de l'image filtrée
minima de l'image ouverte (a) de taille 1, (b) de taille 2

3) Détermination des maxima étendus

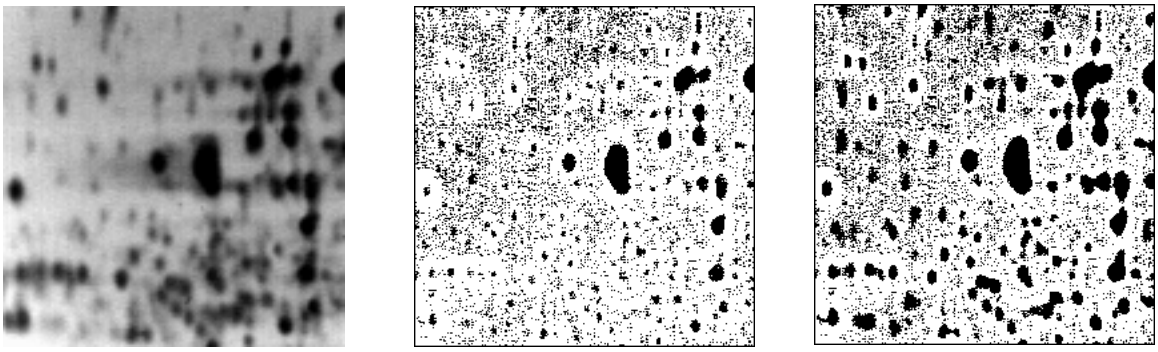
Ces maxima étendus sont caractérisés par le fait que, non seulement ce sont des maxima, mais encore ces maxima ont une hauteur h, c'est-à-dire qu'il faut descendre d'une hauteur supérieure à h avant de pouvoir emprunter un nouveau chemin ascendant vers un autre maximum. La procédure est la suivante :

```
deproc extmaxima extmaxima s d h
syntax "extmaxima greyin binout hauteur"
   int w;
  w := imalloc imdepth s
   imcopy s w
   imcsub w h w
   build s w
  imsub s w w
  imthresh w 1 impixmax s d
  imfree w
end

deproc extminima extminima s d h
syntax "extminima greyin binout hauteur"
  iminv s s
 extmaxima s d h
 iminv s s
end
```

Appliquons l'extraction de minima étendus de hauteurs croissantes à l'image ELECTROP :

```
imdisplay g1 "electrop"
extminima g1 b1 50
imdisplay b1 "profondeur 50"
```

extminima g1 b2 100
imdisplay b2 "profondeur 100"



(a) (b) (c)

Minima régionaux étendus
(a) image originale, (b) minima de profondeur 50, (c) de profondeur 100

RESUME

Ce chapitre d'exercices vous a permis d'introduire dans le dictionnaire les transformations suivantes :

maxima s d
maxima régionaux de l'image numérique s dans l'image binaire d.
minima s d
minima régionaux de l'image numérique s dans l'image binaire d.
extmaxima s d h
maxima régionaux étendus de hauteur h de l'image numérique s dans l'image binaire d.
extminima s d h
minima régionaux étendus de hauteur h de l'image numérique s dans l'image binaire d.
fgrain s d
place en d la première composante connexe de s, dans le sens du balayage et la supprime de s.

Chapitre 7

SQUELETTES

Les exercices suivants introduisent des transformations morphologiques liées à la notion de boules maximales. Cette notion permet de définir le squelette d'ensembles binaires.

EXERCICES

Exercice n° 1 : Erodé ultime d'un ensemble (binaire)

Soit un ensemble X. L'érodé ultime de X est défini par :

$$U(X) = \bigcup_n \{x \in X \ominus nH; d_{X \ominus nH}(x, X \ominus (n+1)H) = +\infty\}$$

U(X) est donc l'ensemble des composantes connexes des érodés successifs de X que l'on ne peut reconstituer à partir de l'érodé de taille immédiatement supérieure.

1) Elaborez l'algorithme et programmez l'érodé ultime d'un ensemble X.

2) Application à l'image CELLS. Calculez le nombre de cellules de l'agglomérat.

3) Essayez de dégager les limites de cette transformation comme outil de séparation des particules.

[procédure **binultim**]

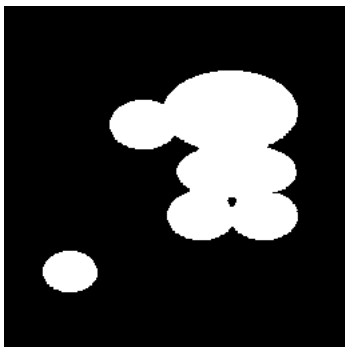
Solution

1) L'érodé ultime se programme comme suit. La détermination des points des érodés successifs à distance géodésique infinie des érodés de taille immédiatement inférieure s'effectue à l'aide de la procédure build.

```
deproc binultim binultim s d
syntax "binultim binin binout"
  int w i ;
  w := imalloc 1
  imcopy s w
  i := 1
  while i do
    ero w d 1
    build w d
    imdiff w d d
    ero w w 1
    i := imvolume w
    imor d d w
  end
  imfree w
end
```

On remarquera dans cette procédure que les érodés ultimes détectés à chaque étape sont ajoutés à l'ensemble érodé. Cette variante par rapport à la définition ne change rien au résultat, mais présente l'avantage de ne pas occuper de mémoire supplémentaire.

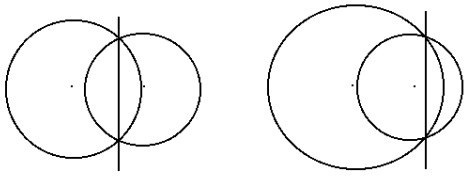
2) Pour vérifier l'algorithme, effectuez les étapes suivantes après avoir chargé l'image CELLS dans b1 :



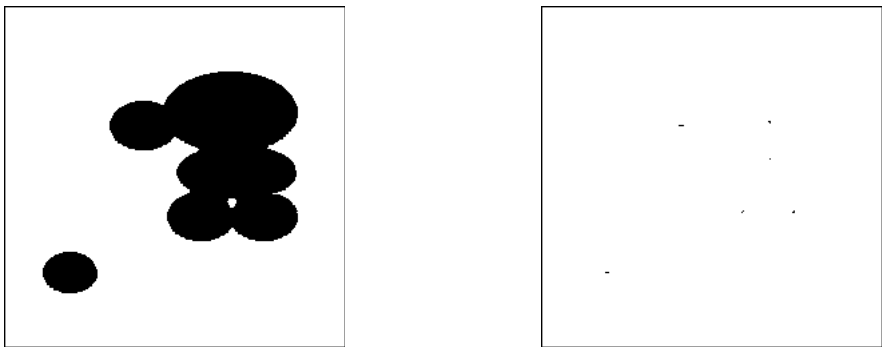
CELLS

Si tout est correct, le nombre de composantes connexes de l'érodé ultime doit être 6.

3) La séparation par érodé ultime ne donne des résultats satisfaisants que lorsque l'agrégat est composé de particules rondes qui ne s'interpénètrent pas trop. Dans ce cas simple où l'agrégat est formé de disques, pour que l'érosion ultime fonctionne, l'axe radical de deux disques connectés doit séparer leurs centres :



Séparation Pas de séparation



(a) (b)

Erosion ultime : (a) original, (b) érodé ultime

Exercice n° 2 : Squelette par boules maximales (binaire)

Soit un ensemble X. Une boule de rayon λ incluse dans X est dite boule maximale si et seulement si il n'existe pas de boule de rayon strictement supérieur incluse dans X et contenant la boule de rayon λ.

B_λ maximale : $B_\lambda \subset X$; il n'existe pas B_μ , $\mu > \lambda$, $B_\lambda \subset B_\mu \subset X$

Le lieu des centres des boules maximales de X, S(X) est appelé squelette de X. Lorsque X est défini sur la trame hexagonale, la notion de boule maximale est remplacée par celle d'hexagone maximal. L'objet de cet exercice est de définir un algorithme permettant d'obtenir le squelette S(X) de X.

1) Soit $X \ominus nH$, l'érodé de taille n de X. Montrer que si x est un point de $X \ominus nH$ qui n'appartient pas à l'ouvert $(X \ominus nH)_H$, il est centre d'un hexagone maximal de taille n.

2) En déduire l'algorithme d'obtention du squelette S(X) de X.

3) A tout point x de S(X), on peut associer le rayon r(x) de l'hexagone maximal centré en x. La fonction r(x) de support S(X) est appelée fonction d'étanchéité. Vérifiez que le doublet (S(X),r(x)) suffit à reconstituer l'ensemble X :

$$X = \bigcup_{x \in S(X)} H(x, r(x))$$

4) Comparez le squelette de l'ensemble X avec son érodé ultime.

5) Quels inconvénients présente cette transformation dans le cas digital?

[procédure **binopenskel**]

Solution

1) Soit un point x tel que :

$$x \in (X \ominus nH) \text{ et } x \notin (X \ominus nH)_H$$

Si x appartient à l'érodé de taille n, x est par définition le centre d'un hexagone de taille n inclus dans X.

Supposons que $X \in (X \ominus nH)_H$. x est donc inclus dans un hexagone de taille 1 inclus dans l'érodé $(X \ominus nH)$.

La dilatation de taille n de cet hexagone fournit un hexagone de taille n+1 inclus dans X. Donc l'hexagone de taille n centré en x est recouvert par cet hexagone de taille n+1. Il ne peut donc être maximal. Inversement, supposons que x, bien que n'appartenant pas à l'ouvert $(X \ominus nH)_H$, ne soit pas centre d'un hexagone maximal de taille n. Alors, il est inclus dans un hexagone de taille m>n recouvrant l'hexagone de taille n centré en x. L'érosion de cet hexagone de taille m par un hexagone de taille n fournit un hexagone de taille $m - n \geq 1$ contenant x. Donc, $x \in (X \ominus nH)_H$. Il y a contradiction.

On peut donc énoncer :

Une condition nécessaire et suffisante pour qu'un point x soit le centre d'un hexagone maximal de taille n est qu'il appartienne à l'ensemble :

$$(X \ominus nH)/(X \ominus nH)_H$$

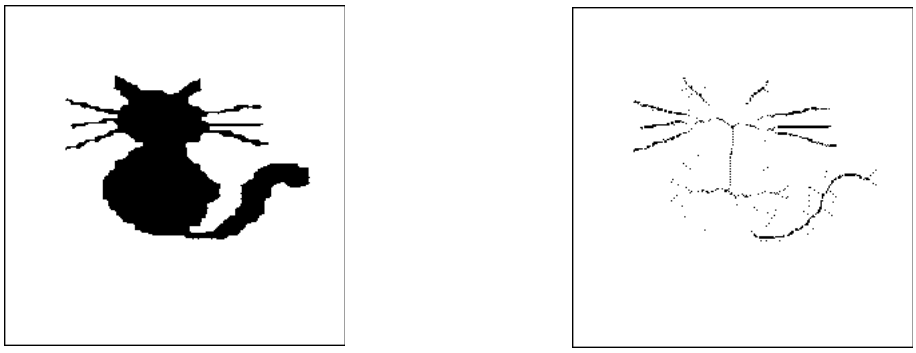
2) D'après ce qui vient d'être dit, pour obtenir le squelette S(X) de l'ensemble X, il suffit d'effectuer la différence ensembliste précédente pour toutes les tailles possibles d'érodés, soit :

$$S(X) = \bigcup_{n=0}^{\infty} [(X \ominus nH)/(X \ominus nH)_H]$$

Voici un exemple de procédure réalisant ce type de squelette :

```
deproc binopenskel binopenskel s d
syntax "binopenskel binin binout"
  int w i ;
  w := imalloc 1
  i := 1
  imcopy s w
  while i do
```

```
    open w d 1
    imdiff w d d
    ero w w 1
    i := imvolume w
    imor d w
  end
  imfree w
end
```



(a) (b)

Squelette non connexe
(a) image originale, (b) squelette par ouvertures

3) Démontrons que tout point x de X appartient à une boule maximale.

En effet, en supposant que ce ne soit pas le cas, il existe alors une boule centrée en x et contenue dans X (boule peut-être réduite à x lui-même). Cette boule n'est incluse dans aucune autre boule incluse dans X. (Sinon cette boule incluante serait elle-même soit maximale, soit incluse dans une boule maximale). La boule centrée en x est donc une boule maximale. x appartient à une boule maximale de X, ce qui contredit l'hypothèse.

$$\forall x \in X, x \in \text{boule maximale de } X$$

L'union de toutes les boules maximales de X est donc égale à X.

4) On peut constater aisément que l'érodé ultime d'un ensemble est contenu dans le squelette.

Soit, en effet, x un point de l'érodé ultime de X. Supposons ce point apparu à l'érosion de taille n. $x \in (X \ominus nH)$. Ce point x est à une distance géodésique infinie de $X \ominus (n+1)H$. Cela revient à dire que le dilaté $[X \ominus (n+1)H] \oplus H$ ne rencontre pas x (puisque ce dilaté représente l'ensemble des points de l'espace à une distance géodésique inférieure ou égale à 1 de $X \ominus (n+1)H$).

Or :

$$[X \ominus (n+1)H] \oplus H = (X \ominus nH)_H$$

Donc :

$$x \in (X \ominus nH)/(X \ominus nH)_H \Rightarrow x \in S(X)$$

Exercice n° 3 : Fonction distance et maxima

Cet exercice reprend la notion de fonction distance introduite au chapitre 2 et la procédure distance correspondante.

1) Montrez que les maxima locaux de la fonction distance sont les points du squelette par boules maximales. Constatez-le sur l'image COFFEE.

2) Montrez que les maxima régionaux de la fonction distance sont les érodés ultimes. Constatez-le sur l'image COFFEE. Quel peut être l'intérêt des maxima régionaux étendus de la fonction distance? Vérifiez-le sur l'image COFFEE.

Solution

1) Démontrons la proposition.

Soit x un point du squelette par ouvertures hexagonales.

x est centre d'un hexagone maximal de taille n. Il en résulte que dist(x)=n+1.

Supposons qu'il existe y voisin de x tel que dist(y) > dist(x).

Alors H^{n+1} (hexagone de centre y de taille n+1) $\subset$ X.

Or $H_x^n \subset H_y^{n+1}$. Il existe donc un hexagone de taille n+1 contenant x et inclus dans X, ce qui contredit l'hypothèse selon laquelle H est un hexagone maximal.

Donc $\forall$y voisin de x, dist(y) $\leq$ dist(x). x est donc un maximum local de la fonction distance.

Inversement, soit x un maximum local de la fonction distance hexagonale.

Soit n = dist(x)-1. H_x^n est le plus grand hexagone de centre x inclus dans X.

Supposons que H^n n'est pas maximal, c-à-d. que :

$$\exists H_y^{n'} \text{ tel que } H_x^n \subset H_y^{n'} \text{ et } H_x^n \neq H_y^n$$

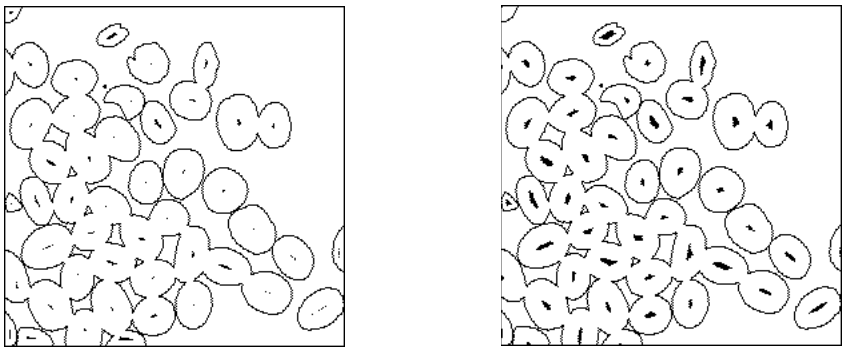
$H_y^{n'} \ominus H_x^n \neq \emptyset$ et c'est un hexagone H_z^m. De plus $m \geq 1$ car $H_y^{n'} \neq H_x^n$ et $x \in H_z^m$.

L'érosion de H_z^m par un hexagone élémentaire est un hexagone H_z^{m-1} et $\forall y \in H_z^m / H_z^{m-1}$, y a un voisin dans H_z^{m-1}. Or $x \notin H_z^m \ominus H$ (sinon $H_x^{n+1} \subset X$, or H_x^n est le plus grand hexagone de centre x inclus dans X) et $H_z^m \ominus H \neq \emptyset$. Donc x a un voisin y dans H_z^{m-1}. On en déduit que $H_y^{n+1} \subset X$ donc que $dist(y) \geq n+2 = dist(x)+1$. Ceci contredit l'hypothèse selon laquelle x est un maximum local de la fonction distance. Donc H_x^n est maximum. Q.E.D.

Vérifions-le après avoir chargé l'image COFFEE dans b1 :

binopenskel b1 b2
distance b1 g1
dil g1 g2 1
imsub g2 g1 g2
imthresh g2 0 0 b3

2) Un érodé ultime est une composante connexe de l'érodé de taille n qui ne peut pas être reconstituée à partir de l'érodé de taille n+1. Or (question 1) l'érodé de taille n est le seuil entre n+1 et 255 de la fonction distance. Les érodés ultimes sont donc simplement les maxima régionaux de la fonction distance.



(a) (b)

(a) érodé ultime, (b) maxima régionaux étendus de hauteur 2

On peut constater sur l'image COFFEE que certains grains sont marqués par plusieurs érodés ultimes, ce qui sera préjudiciable dans la suite des exercices (segmentation). Les maxima régionaux étendus peuvent apporter une solution pour connecter ces marqueurs. On obtiendra alors une meilleure segmentation.

Vérifions sur l'image COFFEE la correspondance entre les érodés ultimes et les maxima régionaux de la fonction distance. L'image COFFEE sera préalablement chargée dans b1.

distance b1 g1
maxima g1 b2
binultim b1 b3

RESUME

Ce chapitre d'exercices vous a permis d'introduire dans le dictionnaire les transformations suivantes :

binultim s d
érodé ultime de l'image binaire s dans l'image binaire d.
binopenskel s d
squelette par boules maximales de l'image binaire s dans l'image binaire d.

Chapitre 8

AMINCISSEMENTS, EPAISSISSEMENTS

8.1. Introduction

Les transformations que nous allons élaborer dans les exercices qui vont suivre ont un degré de complexité plus élevé. Pour reprendre la comparaison utilisée plus haut, elles forment de véritables "machines-outils" pour la MM. Elles sont à la jonction des notions de géodésie et d'homotopie. Nous avons déjà manipulé les transformations géodésiques. Rappelons seulement la notion d'homotopie et surtout de transformation homotopique.

8.2. Amincissements binaires, rappel

Soit $T = (T_1 , T_2)$ un élément structurant biphasé. La transformée en tout ou rien d'un ensemble X par T est égale à :

$$X * T = (X \cap \check{T}_1) \cap (X^c \cap \check{T}_2)$$

L'épaississement de X par T est égal à :

$$X \odot T = X \cup (X * T)$$

L'amincissement quant à lui est défini par :

$$X \bigcirc T = X/(X * T)$$

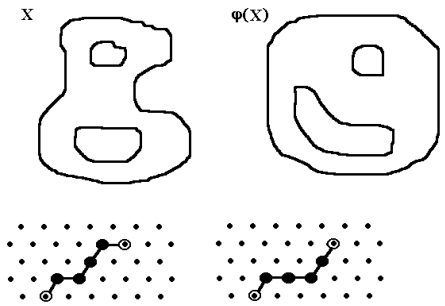
Amincissement et épaississement sont deux transformations duales :

$$(X^c \odot T)^c = [X^c \cup (X^c * T)]^c = X \cap (X^c * T)^c = X / (X * T') = X \bigcirc T'$$

où $T' = (T_2 , T_1)$.

8.3. Homotopie

Deux chemins d'un ensemble X sont homotopes s'il est possible de les superposer par une série de déformations continues. Par continue, on entend que la déformation doit se faire sans coupures, et tous les chemins intermédiaires doivent être inclus dans X.



Chemins homotopes sur la trame hexagonale

Par extension, une transformation Ψ qui préserve l'homotopie est dite homotopique. De façon intuitive, une transformation homotopique $Y = \Psi(X)$ transforme l'ensemble X en un ensemble Y superposable sur X par déformation continue.

Lorsque l'ensemble X est digitalisé suivant une trame (carrée ou hexagonale), la comparaison des chemins est aisée, puisque tout chemin peut être défini comme la concaténation d'arêtes élémentaires. Une transformation homotopique ne brise pas les chemins.

8.4. Amincissements numériques

Soient $m(x) = \sup_{y \in T_{2x}} f(y)$ et $M(x) = \inf_{y \in T_{1x}} f(y)$.

L'épaississement de f par $T = (T_1, T_2)$ est défini par :

$$(f \odot T)(x) = M(x), \text{ ssi } m(x) \leq f(x) < M(x)$$
$$(f \odot T)(x) = f(x), \text{ sinon.}$$

L'amincissement de f par $T = (T_1, T_2)$ est défini par :

$$(f \bigcirc T)(x) = m(x), \text{ ssi } m(x) < f(x) \leq M(x)$$
$$(f \bigcirc T)(x) = f(x), \text{ sinon.}$$

EXERCICES

Exercice n° 1

Les éléments structurants $T = (T_1, T_2)$ présentant quelque intérêt en trame hexagonale sont ceux définis sur l'hexagone élémentaire :

$$T_1 \subset H,\ T_2 \subset H$$

On peut même écrire : $T_1 \cap T_2 = \emptyset$. En effet, T_1 et T_2 ne doivent avoir aucun point commun si on désire que X * T soit différent de l'ensemble vide.

1) Programmez l'épaississement et l'amincissement binaires et numériques par tout élément structurant défini sur l'hexagone élémentaire. La transformation en tout ou rien **imhitormiss** existe déjà.

2) Appliquez les algorithmes à quelques images binaires. En particulier, effectuez les opérations suivantes :

- sur l'image NOISE, suppression des points isolés blancs et noirs.
- obtention du contour d'un ensemble en une seule transformation.

Solution

1) Procédures binthin et binthick

imhitormiss existe déjà dans le dictionnaire. Programmons l'amincissement :

```
deproc binthin binthin se1 se0 s d
syntax "binthin se_for_1 se_for_0 binin binout"
  int w oldedge;
  w := imalloc 1
  oldedge := edge
   imsetedge 0
   imhitormiss s se1 se0  w
  imdiff s w d
```

```
  imsetedge oldedge
  imfree w
end
```

La procédure peut être réitérée pour toutes les rotations de l'élément structurant :

```
  deproc thinturn thinturn se1 se0 s d
  syntax "thinturn se_for_1 se_for_0 binin binout"
    imcopy s d
    for 1 to ngbnb do
      binthin se1 se0 d d
      se0 := serotate se0
      se1 := serotate se1
    end
  end
```

Puis l'épaississement :

```
deproc binthick binthick se1 se0 s d
syntax "binthick se_for_1 se_for_0 binin binout"
  int w oldedge ;
  w := imalloc 1
  oldedge := edge
  imsetedge 0
  imhitormiss s se1 se0  w
  imsup s w d
  imsetedge oldedge
  imfree w
end
```

Enfin, la procédure en rotation :

```
  deproc thickturn thickturn se1 se0 s d
  syntax "thickturn se_for_1 se_for_0 binin binout"
    imcopy s d
    for 1 to ngbnb do
      binthick se1 se0 d d
      se0 := serotate se0
      se1 := serotate se1
    end
  end
```

Afin de programmer les procédures équivalentes en numérique, définissons préalablement l'érosion et la dilatation d'une image numérique par un élément structurant quelconque se (défini sur un hexagone unitaire).

```
  deproc greyseero greyseero se s d
 syntax "greyseero struct_elt greyin greyout"
```

```
    int i j w;
    w := imalloc imdepth s
    imset impixmax w w
    j := 0
    i := 1
    for 0 to ngbnb do
      if (se && i) then
        iminfngb s w j 1
      end
      i := (i * 2)
      ++ j
    end
    imcopy w d
    imfree w
  end

  deproc greysedil greysedil se s d
  syntax "greysedil struct_elt greyin greyout"
    int i j w;
    w := imalloc imdepth s
    imset 0 w
    j := 0
    i := 1
    for 0 to ngbnb do
      if (se && i) then
        imsupngb s w j 1
      end
      i := (i * 2)
      ++ j
    end
    imcopy w d
    imfree w
  end
```

L'amincissement et l'épaississement s'écrivent alors :

```
  deproc greythin greythin sei ses s d
  syntax "greythin se_for_inf se_for_sup greyin greyout"
    int m M bw1 bw2 gw;
    m := imalloc imdepth s
    M := imalloc imdepth s
    gw := imalloc imdepth s
    bw1 := imalloc 1
    bw2 := imalloc 1
    greysedil ses s m
    greyseero sei s M
    imsub s M M
    imthresh M 1 impixmax s bw1          { bw1 = where s > M }
```

```
    iminv bw1 bw1                    { bw1 = where s <= M }
    imsub s m M                      { M is no longer used }
    imthresh M 1 impixmax s bw2          { bw2 = where m < s }
    imsup bw2 bw1 bw1                { bw1 = where m < s <= M }
    immask bw1 0 impixmax s gw
    iminf gw m m
     imcopy s d
    iminv gw gw
    iminf gw d d
    imsup m d d
    imfree m
    imfree M
   imfree gw
   imfree bw1
   imfree bw2
end

 deproc greythick greythick sei ses s d
 syntax "greythick se_for_inf se_for_sup greyin greyout"
    int m M bw1 bw2 gw;
    m := imalloc imdepth s
    M := imalloc imdepth s
    gw := imalloc imdepth s
    bw1 := imalloc 1
    bw2 := imalloc 1
    greysedil ses s m
    greyseero sei s M
    imsub m s m
    imthresh m 1 impixmax s bw1{ bw1 = where m > s }
    iminv bw1 bw1                    { bw1 = where m <= s }
    imsub M s m                      { m is no longer used }
    imthresh m 1 impixmax s bw2{ bw2 = where s < m }
    iminf bw2 bw1 bw2                { bw1 = where m <= s < M }
    immask bw1 0 impixmax s gw
    iminf gw M M
    imcopy s d
    iminv gw gw
   iminf gw d d
   imsup M d d
   imfree m
   imfree M
   imfree gw
   imfree bw1
   imfree bw2
end
```

2) Exemples

- Suppression des points isolés

Chargez l'image NOISE dans b1 et faire :

binthin 1 126 b1 b1
binthick 126 1 b1 b1

Puis, visualisez le résultat.

- Contour d'un ensemble

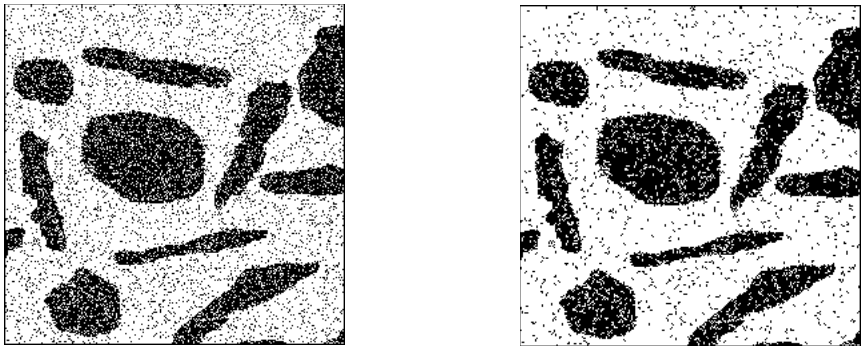
On a :

$$C(X) = X/(X \cap H) = X \cap (X \cap H)^c$$

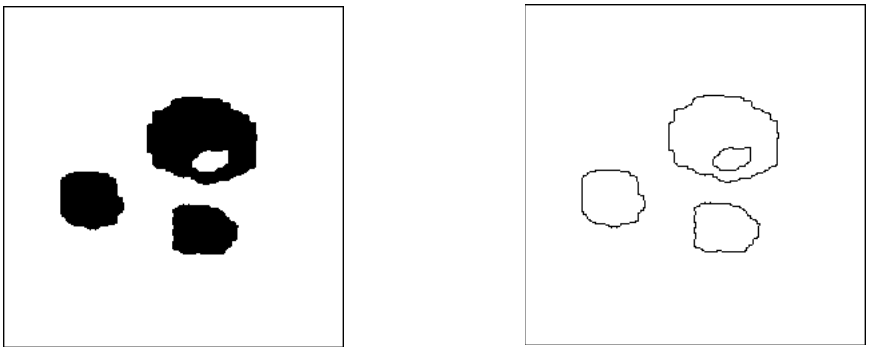
Donc :

$$C(X) = X \odot T \text{ , avec } T = (H,\varnothing)$$

```
deproc bincontour bincontour s d
syntax "bincontour binin binout"
  binthin 127 0 s d
end
```



Suppression des points isolés



Contour d'un ensemble

Exercice n° 2 : Epaississements, amincissements géodésiques

Soit un ensemble X contenu dans Z. L'épaississement géodésique de X par un élément structurant T contenu dans l'hexagone élémentaire peut être défini par :

$$(X \odot T) = (X \odot T) \cap Z$$

1) Programmez cette transformation.

2) Sachant que l'amincissement est l'opération duale de l'épaississement, on définit l'amincissement géodésique par :

$$(X \bigcirc T) = Z/[(Z/X) \odot T']$$

Simplifiez et programmez cette transformation.

Solution

1) Epaississement géodésique

```
deproc bingdsthick bingdsthick se1 se0 s m d
syntax "bingdsthick se_for_1 se_for_0 binin binmask binout"
  binthick se1 se0 s d
  imand m d
end

deproc gdsthickturn gdsthickturn se1 se0 s m d
syntax "gdsthickturn se_for_1 se_for_0 binin binmask binout"
   int w ;
   w  := imalloc 1
   imcopy s w
   for 1 to ngbnb do
      bingdsthick se1 se0 w m w
      se0 := serotate se0
      se1 := serotate se1
   end
  imcopy w d
  imfree w
end
```

2) Amincissement géodésique

L'amincissement géodésique doit être réalisé par le biais d'un épaississement par l'élément structurant dual, ceci afin d'éviter les effets de bord que l'on a déjà rencontrés lors des érosions.

```
deproc bingdsthin bingdsthin se1 se0 s m d
syntax "bingdsthin se_for_1 se_for_0 binin binmask binout"
  bindiff m s d
  binthick se0 se1 d d
  bindiff m d d
end

 deproc gdsthinturn gdsthinturn se1 se0 s m d
  syntax "gdsthinturn se_for_1 se_for_0 binin  binmask binout"
     int w ;
     w := imalloc 1
     imdiff m s w
     gdsthickturn se0 se1 w m w
     imdiff m w d
     imfree w
 end
```

Exercice n° 3

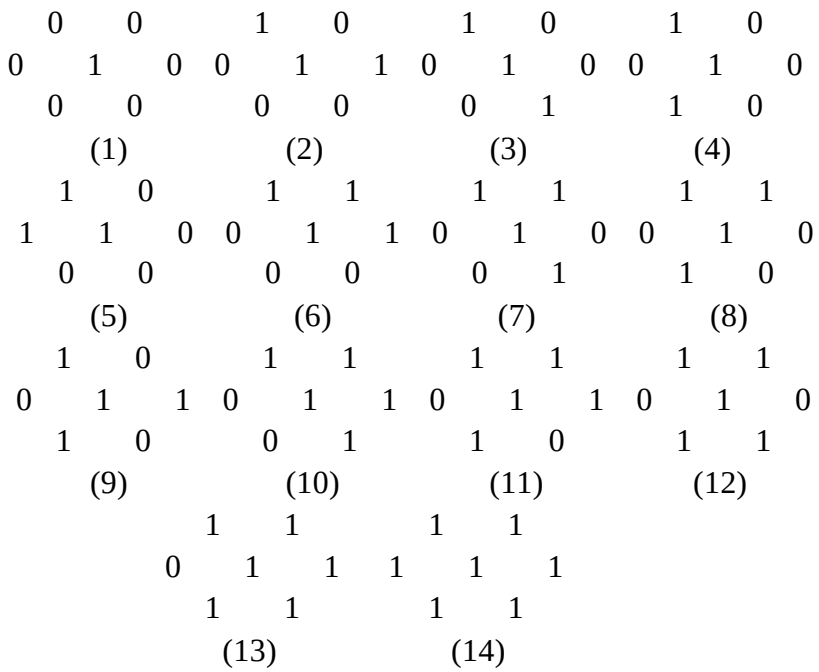
1) Trouvez toutes les configurations possibles (à une rotation près) pour le voisinage d'un point en trame hexagonale (il y en a 14).

2) Montrez que la relation d'homotopie pour deux chemins C_1 et C_2 de même origine et de même extrémité inclus dans un ensemble X est une relation d'équivalence.

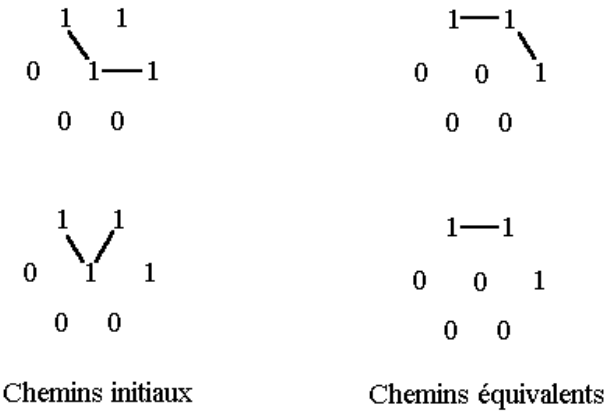
3) En déduire, parmi les configurations trouvées, celles qui génèrent un amin- cissement homotopique.

Solution

1) Voici les 14 configurations (le point central est supposé à 1) :



2) Evident. La relation d'homotopie est une relation d'équivalence.



Configuration 6

3) L'amincissement par une des 14 configurations possibles a pour effet de remplacer le point central à 1 par 0. Pour mettre en évidence les configurations qui génèrent des amincissements homotopiques, il suffit de vérifier que tout chemin passant par le point central

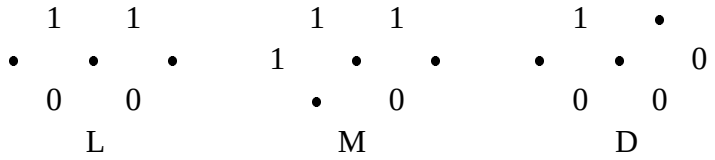
peut être, après amincissement, remplacé par un chemin équivalent dans le voisinage hexagonal.

Illustrons-le par exemple dans le cas de la configuration n° 6. Il suffit de vérifier que tous les chemins sont conservés :

En utilisant la même procédure pour les 13 autres configurations, on constate que seules celles dont le contour est formé d'une composante connexe génèrent des amincissements homotopiques, soient les configurations 2, 3, 6, 10 et 13.

Exercice n° 4

1) Programmez le squelette connexe en utilisant les éléments structurants L, M et D. Comparez les résultats obtenus.

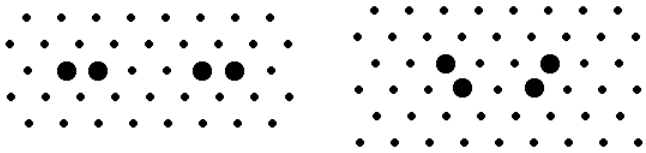


2) Donnez les transformations permettant d'extraire les points caractéristiques du squelette :

- points extrémités.
- points n-uples.
- points isolés.

3) Le sens de rotation et l'orientation de départ pour les éléments structurants utilisés par les différents squelettes sont totalement arbitraires. Il en résulte des variations plus ou moins importantes dans le squelette obtenu. Ces variations peuvent même parfois générer des artefacts.

Générez l'une des deux images suivantes :



Effectuez sur cette image un squelette de type L par épaississement, et jugez du résultat obtenu. Existe-t-il un moyen d'améliorer l'algorithme?

[procédures **Lthin** ; **Mthin** ; **Dthin** ; **multpoint** ; **endpoint**]

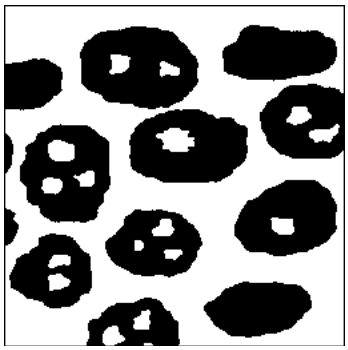
Solution

1) Donnons la procédure utilisée pour le squelette réalisé à l'aide de l'élément L.

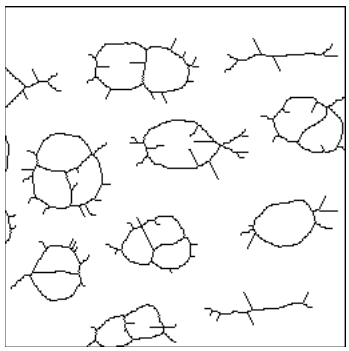
Ce squelette faisant intervenir la rotation des éléments structurants L, on peut le définir en deux étapes. On définit d'abord une procédure permettant d'amincir jusqu'à idempotence par une rotation d'un élément structurant es.

```
deproc thin thin se1 se0 s d
syntax "thin se_for_1 se_for_0 binin binout"
   int i1 i2 ;
   imcopy s d
   i1 := imvolume d
   if (edge = 0) then
```

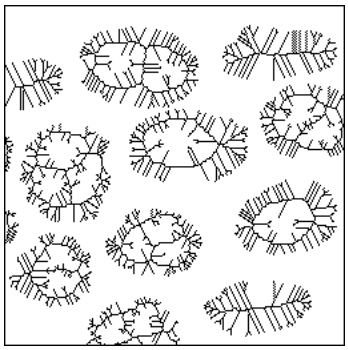
```
        while (i1 <> i2) do
          i2 := i1
          thinturn se1 se0 d d
          i1 := imvolume d
        end
      else
      iminv d d
      while (i1 <> i2) do
        i2 := i1
        thickturn se0 se1 d d
        i1 := imvolume d
      end
    iminv d d
    end
  end
```



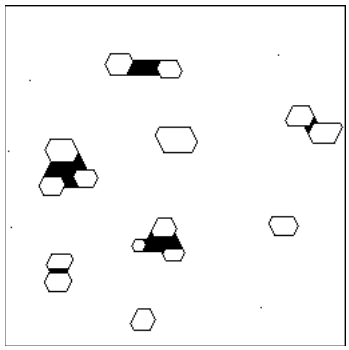
(a)



(b)



(c)



(d)

Squelettes, (a) original, (b) squelette L, (c) squelette M, (d) squelette D

Le squelette L s'écrit alors :

```
deproc Lthin Lthin s d
syntax "Lthin binin binout"
   if (grid <> 0) then
     thin 66 24 s d
   else
     thin 262 112 s d
```

```
    end
  end
```

On définit de la même façon les procédures **Mthin** et **Dthin** : il suffit de changer les éléments structurants.

```
 deproc Mthin Mthin s d
 syntax "Mthin binin binout"
   if (grid <> 0) then
     thin 56 2 s d
   else
     thin 120 258 s d
   end
 end

 deproc Dthin Dthin s d
 syntax "Dthin binin binout"
    if (grid <> 0) then
      thin 2 56 s d
    else
      thin 258 120 s d
   end
end
```

On constate que ces trois transformations homotopiques fournissent des résultats assez différents :
- **Mthin** produit un nombre relativement élevé de barbules ;
- **Dthin** réduit tout ensemble simplement connexe à un point ;
- **Lthin** est la seule transformation dont le comportement correspond à l'idée intuitive que l'on a du squelette d'un ensemble.

2) Extraction des points singuliers du squelette

- Extrémités

Les extrémités du squelette correspondent aux configurations (voir exercice n° 3) 1, 2, 3 (à une rotation près).

Afin de simplifier l'élément structurant, on constate que :

$$\begin{matrix} 0 & 0 & & 1 & 0 & & 1 & 1 & & \bullet & \bullet \\ 0 & 1 & 0 & \cup\ 0 & 1 & 0 & \cup\ 0 & 1 & 0 & = 0 & 1 & 0 \\ 0 & 0 & & 0 & 0 & & 0 & 0 & & 0 & 0 \end{matrix}$$

```
deproc endpoints endpoints s d
syntax "endpoint binin binout"
  int se0 se1 w w1 oldedge ;
  w := imalloc 1
  w1 := imalloc 1
 oldedge := edge
 imsetedge 0
 if (grid <> 0) then
   se0 := 30
```

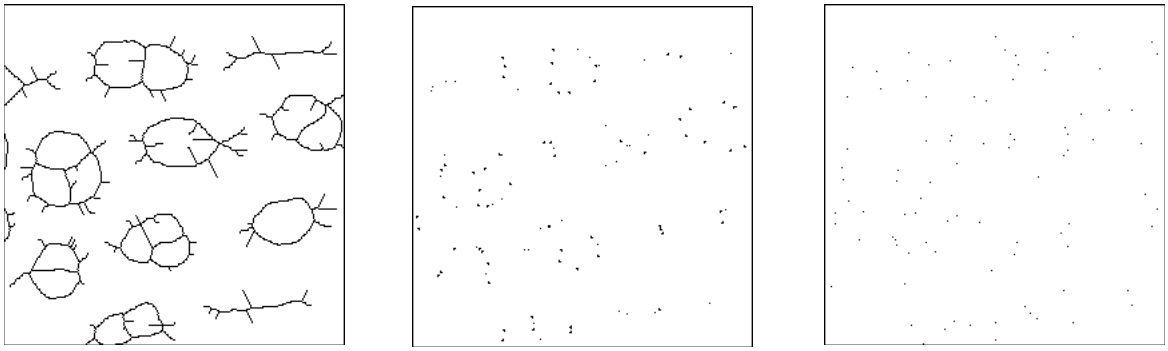
```
      se1 := 1
    else
      se0 := 62
      se1 := 1
    end
    for 1 to ngbnb do
      imhitormiss s se1 se0 w
      imor w w1 w1
      se0 := serotate se0
      se1 := serotate se1
    end
    imcopy w1 d
    imsetedge oldedge
    imfree w1
    imfree w
  end
```

- Points multiples

Seules les configurations 6, 7, 8, 9, 10, 11, 12, 13 et 14 correspondent à des points multiples. Compte tenu du fait que les extrémités correspondent aux configurations 1, 2 et 3, l'obtention des points multiples peut s'opérer de la façon suivante :
a) Amincissement par les configurations 4, 5 en rotation.
b) Suppression des extrémités.

```
  deproc mulpoints mulpoints s d
  syntax "mulpoints binin binout"
    int w w1 se0 se1 oldedge ;
    w  := imalloc 1
    w1 := imalloc 1
    oldedge := edge
    imsetedge 0
    if (grid = 1) then
      endpoints s w1
      se1 := 11
      se0 := 116
      for 1 to 6 do
        imhitormiss s se1 se0 w
        imsup w w1 w1
        se0 := serotate se0
        se1 := serotate se1
      end
      se1 := 19
      se0 := 108
    for 1 to 3 do
      imhitormiss s se1 se0 w
      imsup w w1 w1
      se0 := serotate se0
      se1 := serotate se1
```

```
  end
  imdiff s w1 w1
else
  imset impixmin w1 w1
  se1 := 169
  se0 := 340
  for 1 to 4 do
    imhitormiss s se1 se0 w
    imsup w w1 w1
    se0 := s4rotate se0
    se1 := s4rotate se1
  end
  se1 := 169
  se0 := 2
  for 1 to 4 do
    imhitormiss s se1 se0 w
    imsup w w1 w1
    se0 := s4rotate se0
    se1 := s4rotate se1
   end
   se1 := 170
   se0 := 1
   for 1 to 4 do
     imhitormiss s se1 se0 w
     imsup w w1 w1
     se0 := s4rotate se0
     se1 := s4rotate se1
   end
  end
  imcopy w1 d
  imsetedge oldedge
  imfree w
  imfree w1
end
```



(a) (b) (c)

(a) original, (b) points multiples, (c) extrémités

- Points isolés

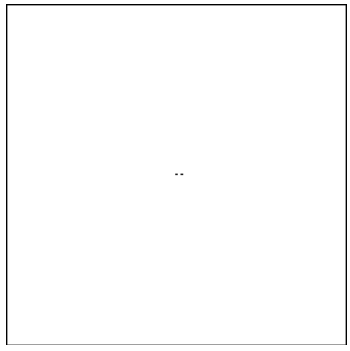
Evident (c'est la configuration 1).

3) Pour générer les images test dans l'image b1 (par exemple), faire :

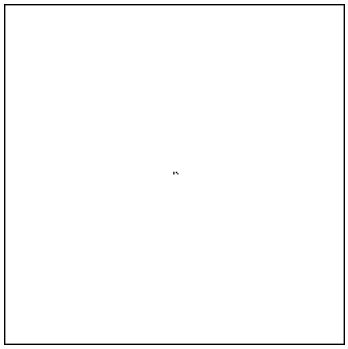
imset 0 b1
imwritepix 1 127 127 b1
imwritepix 1 128 127 b1
imwritepix 1 131 127 b1
imwritepix 1 132 127 b1

et pour la deuxième :

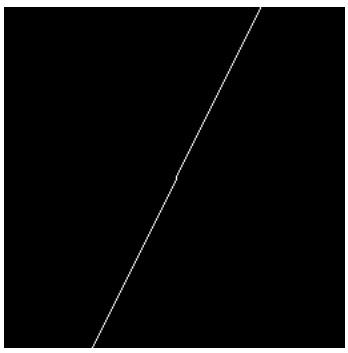
imset 0 b1
imwritepix 1 127 127 b1
imwritepix 1 130 127 b1
imwritepix 1 127 126 b1
imwritepix 1 129 126 b1



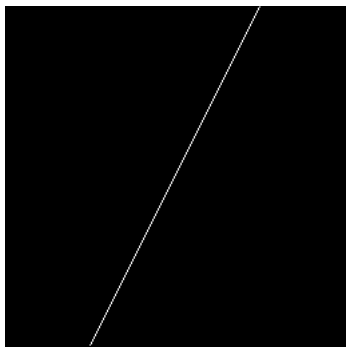
(a)



(b)



(c)



(d)

Comportement du squelette
(a), (b) Configurations testées, (c), (d) Squelettes correspondants

Le résultat de la squelettisation par épaississement (écrire une procédure **Lthick** semblable à **Lthin**) est donné ci-dessus.

Cette divergence provient du fait que l'algorithme de squelettisation n'est pas isotrope, puisque l'on travaille direction après direction.

Des algorithmes plus performants existent. Ils utilisent notamment des éléments structurants définis sur des hexagones de taille 2, ce qui leur permet de prendre en compte en même temps toutes les rotations.

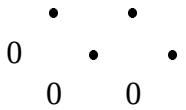
```
deproc thick thick se1 se0 s d
syntax "thick se_for_1 se_for_0 binin binout"
  int i1 i2 ;
   imcopy s d
  i1 := imvolume d
  if (edge = 1) then
     while (i1 <> i2) do
       i2 := i1
       thickturn se1 se0 d d
       i1 := imvolume d
     end
  else
     iminv d d
    while (i1 <> i2) do
      i2 := i1
      thinturn se0 se1 d d
      i1 := imvolume d
     end
     iminv d d
  end
end

deproc Lthick Lthick s d
 syntax "Lthick binin binout"
   if (grid <> 0) then
      thick 24 66 s d
   else
      thick 112 262 s d
   end
 end

 deproc Mthick Mthick s d
 syntax "Mthick binin binout "
   if (grid <> 0) then
     thick 56 2 s d
   else
     thick 120 258 s d
   end
 end

 deproc Dthick Dthick s d
 syntax "Dthick binin binout"
    if (grid <> 0) then
       thick 2 56 s d
```

```
    else
      thick 258 120 s d
    end
  end
```

Exercice n° 5

L'exercice n° 3 vous a permis de déterminer les amincissements homotopiques sur l'hexagone élémentaire.

1) Parmi les configurations sélectionnées, étudiez leur effet sur la longueur des chemins homotopes, après amincissement. Montrez notamment en quoi la configuration suivante est particulière.



2) En déduire un algorithme permettant de mettre en évidence le centre géodésique d'ensembles simplement connexes. Programmez cet algorithme.
[procédures **Dthin** ; **gdsthin** (appliqué à D)]

Solution

1) On sait que pour chaque point d'un ensemble X simplement connexe (sans trous), on peut associer la valeur de la plus grande distance géodésique entre ce point et tout autre point de X. On définit ainsi une fonction sur X dont les maxima correspondent aux extrémités de X et le minimum (unique) au centre géodésique. Considérons la configuration $\begin{matrix} & 1 & 1 \\ 0 & 1 & 1 \\ & 0 & 0 \end{matrix}$ appartenant à X et soit n la valeur de la fonction précédemment définie, au point central. On peut montrer que dans cette configuration, il existe toujours un point du contour pour lequel la valeur de la fonction est également n :

$$\begin{matrix} & n & n' \\ 0 & n & n'' \\ & 0 & 0 \end{matrix} \quad \text{(par exemple)}$$

Nous admettrons ce résultat (on peut le démontrer en étudiant les différents chemins géodésiques possibles issus du point central).

Sa conséquence immédiate est que l'amincissement direction par direction de X à l'aide de cet élément structurant produit un ensemble Y dont la fonction plus grande distance, bien qu'amputée de certains points, est identique sur Y à celle définie sur X.

Les deux ensembles ont donc mêmes extrémités (aux points éventuellement éliminés près) et même centre géodésique.

2) Question piège! On pourrait effectivement penser que l'algorithme d'amincissement utilisant l'élément structurant D dans lequel les trois configurations :

$$\text{(a)}\ \begin{matrix} & 1 & 1 \\ 0 & 1 & 1 \\ & 0 & 0 \end{matrix} \qquad \text{(b)}\ \begin{matrix} & 1 & 1 \\ 0 & 1 & 0 \\ & 0 & 0 \end{matrix} \qquad \text{(c)}\ \begin{matrix} & 1 & 0 \\ 0 & 1 & 0 \\ & 0 & 0 \end{matrix}$$

qui le constituent ont été séparées serait plus apte à fournir le centre géodésique. En effet, l'élément (a) ne modifie pas la fonction plus grande distance, alors que (c) la diminue de 1

systématiquement. Malheureusement le comportement de (b) est parfois similaire à (a), parfois à (c).

On peut cependant améliorer le placement du centroïde obtenu par **Dthin** par rapport au vrai centre géodésique, en effectuant la procédure ci-après.

Notons qu'il existe un algorithme exact de recherche du centre géodésique. Il fait appel à un type particulier de transformées morphologiques appelées propagations.

(a)

(b)

Centres géodésiques
(a) avec squelette D, (b) avec la procédure décrite

```
deproc gdscentre gdscentre s d
syntax "gdscentre binin binout"
   int se10 se11 se20 se21 se30 se31 w w1 ;
   if (grid <> 1) then makeerror 10001 end
    w := imalloc 1
    w1 := imalloc 1
    se10 := 112
    se11 := 14
    se20 := 120
    se21 := 6
    se30 := 124
    se31 := 2
    imcopy s d
    imset 0 w
    while (imcompare w d w <> -1) do
       while (imcompare w d w <> -1) do
          imcopy d w
          thinturn se11 se10 d d
       end
      thinturn se21 se20 d d
      thinturn se31 se30 d d
    end
   imfree w
   imfree w1
 end
```

Exercice n° 6

Programmez, à l'aide des transformations géodésiques, le squelette géodésique par épaississement. Vérifiez en particulier que seul l'élément structurant :

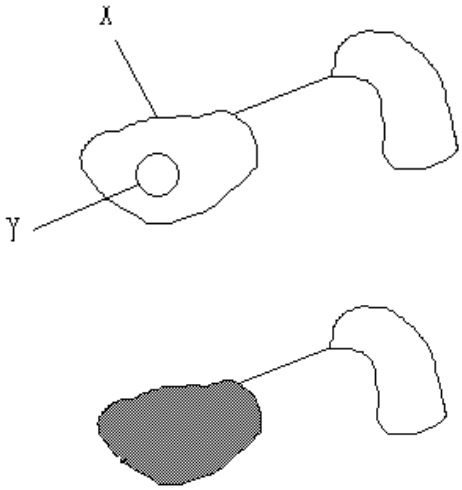
0 0
• • 0
1 •

permet d'obtenir une transformation exacte.

Solution

On doit utiliser l'élément structurant M. En effet, il est le seul à pouvoir pénétrer dans les zones sans épaisseur de l'ensemble géodésique X :

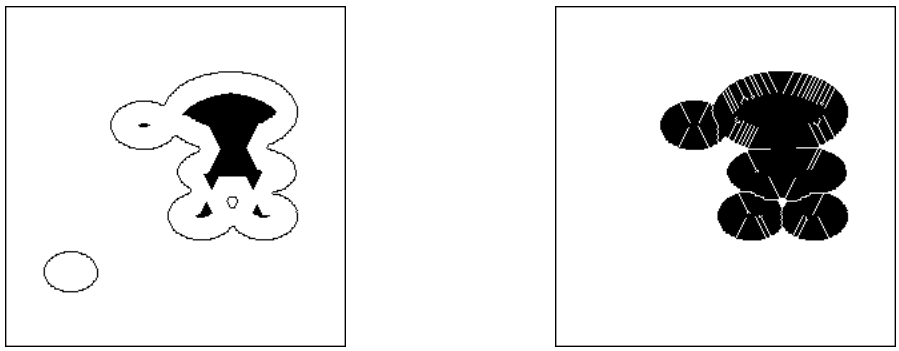
Avec L, on obtiendrait :



La définition de ce squelette se fera en deux temps comme pour les squelettes euclidiens :

```
deproc gdsthick gdsthick se1 se0 s m d
syntax "gdsthick se_for_1 se_for_0 binin binmask binout"
  int i1 i2 w ;
  w := imalloc 1
  imcopy s w
  i1 := imvolume w
  while (i1 <> i2) do
     i2 := i1
     gdsthickturn se1 se0 w m w
     i1 := imvolume w
  end
  imcopy w d
  imfree w
end

deproc gdsMthick gdsMthick s m d
syntax "gdsMthick binin binmask binout"
```

```
    int se0 se1 ;
    se0 := 56
    se1 := 2
    gdsthick se1 se0 s m d
end
```



(a) (b)

Squelette géodésique
(a) ensemble à squelettiser, et ensemble géodésique, (b) résultat

Exercice n° 7

Le gradient morphologique d'une fonction f de $\mathbb{R}^2$ et prenant ses valeurs sur $\mathbb{R}$ dans la direction α est défini par :

$$g^{a}(f) = \lim_{\lambda \to 0} \frac{(f \oplus \lambda L^{a}) - (f \ominus \lambda L^{a})}{2\lambda}$$

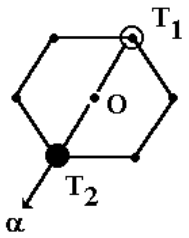
où λL^{a} est un segment de longueur l dans la direction α. Dans la version digitale, le gradient est défini par :

$$g^{a}(f) = (f \oplus L^{a}) - (f \ominus L^{a})$$

où L^{α} est le segment élémentaire dans la direction α de la trame.

L'azimut du gradient est la direction, dans le plan horizontal, du vecteur $\overrightarrow{\text{grad}}(f)$. On peut le définir dans les directions de la trame de digitalisation.

On peut définir un nouveau gradient dans la direction α à l'aide d'épaississements et d'amincissements. Le gradient directionnel dans la direction α est défini par :



$$g_{a}(f) = (f \odot T_{a}) - (f \bigcirc T_{a})$$

où T_1 et T_2 constituent un élément structurant biphasé $T_{\alpha} = (T_1, T_2)_{\alpha}$.

Le gradient directionnel maximum peut apparaître dans plusieurs directions en même temps. Il faut calculer la direction la plus probable. Pour cela, on notera que le calcul de ces gradients implique que trois directions au plus peuvent être extraites en trame hexagonale resp. quatre en trame carrée.

1) Trouvez toutes les configurations possibles de gradients directionnels maximaux (à une rotation près, et en trame hexagonale). Il y en a 5.

2) Si les directions ne sont pas adjacentes, le gradient est considéré comme nul. Si les directions sont adjacentes, une direction unique est choisie, moyenne de toutes les directions présentes. Quelles sont les configurations résultantes (à une rotation près). Il y en a 3. Quel effet obtient-on et que peut-on faire pour l'éviter?

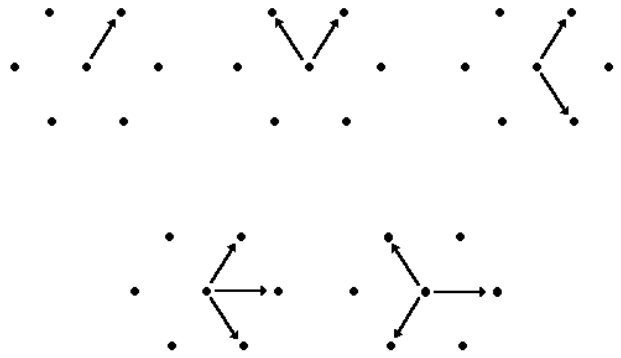
3) Programmez le gradient directionnel et appliquez l'azimut à l'image PETROLE. [procédure **gradvect**]

Solution

1) Le calcul de l'azimut du gradient nécessite le calcul des gradients directionnels $g^{\alpha}(f)$ pour toutes les directions α que l'on peut exhiber sur la trame de digitalisation. L'azimut du gradient sera alors la direction α correspondant au gradient directionnel le plus élevé. Ayant effectué les divers traitements en trame hexagonale, il y aurait donc six directions possibles pour l'azimut. Cependant, il peut arriver que plusieurs directions de gradients directionnels fournissent des valeurs maximales. Ce phénomène est gênant car par définition le vecteur gradient est unique en tout point de l'image. Il faut donc corriger l'image brute des azimuts obtenue par simple détection de la ou des directions de plus fort gradient directionnel. Pour cela, une première transformation permet de mettre en évidence toutes les directions pour lesquelles le gradient directionnel est maximum.

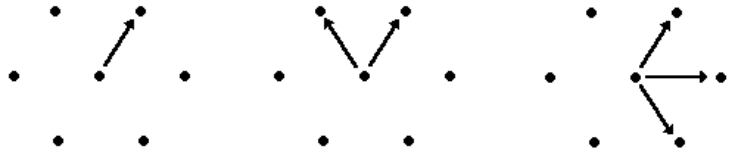
La procédure de calcul du gradient directionnel est fournie ci-dessous.

```
deproc dirgradient dirgradient dir s d
 syntax "dirgradient dir greyin greyout"
    int w se1 se2 ;
    w := imalloc imdepth s
    se1 := (2 power dir)
    se2 := setranspose se1
    greythickstep se1 se2 s w
    greythinstep se1 se2 s d
    imsub w d d
   imfree w
 end
```

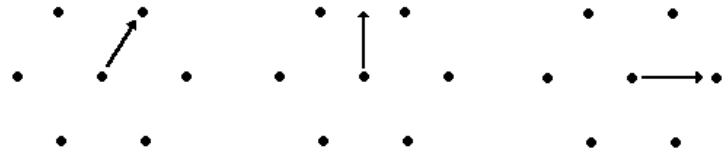


Ensemble des configurations possibles de gradients directionnels maximaux
(aux rotations près)

2) Le calcul de ces gradients par épaississement/amincissement implique que trois directions au plus peuvent être extraites. Ensuite, un tri des différentes configurations est effectué. Si les directions marquées ne sont pas adjacentes, alors le gradient est considéré comme nul. Si les directions rencontrées sont adjacentes, alors une direction unique est choisie, moyenne de toutes les directions présentes. En fait ce cas se résume aux trois situations ci-dessous (à une rotation près) :



Directions initiales



Directions sélectionnées

La deuxième configuration est assez intéressante. En effet dans ce cas, la direction moyenne est une des directions conjuguées de la trame. C'est la raison pour laquelle l'azimut du gradient final est codé sur douze directions et non pas six. Chaque direction est codée par une valeur numérique prise dans l'intervalle [0,12].

3) Pour programmer le gradient complet (module et azimut), nous allons définir préalablement deux procédures utilitaires générant les masques des sups de deux images et une procédure de masquage d'une image de gris consistant à forcer à une valeur quelconque les pixels tombant dans un masque sans modifier les autres.

```
deproc masksup masksup g1 g2 b
syntax "masksup greyin1 greyin2 mask_out"
   int w ;
   w := imalloc imdepth g1
   imsub g1 g2 w
   imthresh w 1 impixmax w b
   imfree w
end

deproc masksupequal masksupequal g1 g2 b
syntax "masksupequal greyin1 greyin2 mask_out"
   int w ;
   w := imalloc imdepth g1
   imsub g1 g2 w
   imthresh w 0 0 b
   imfree w
end
```

```
deproc greymask greymask v m g
syntax "greymask val mask greyinout"
   int w ;
   w := imalloc imdepth g
   iminv m m
   immask m 0 impixmax w w
   iminf w g g
   iminv m m
   immask m 0 v w
   imsup w g g
   imfree w
end
```

Commençons par définir une procédure grossière où on se contente de coder les directions d'apparition des maxima du gradient directionnel sans corriger le fait que plusieurs directions peuvent apparaître au même point :

```
deproc vectgrad0 vectgrad0 g md az
syntax "vectgrad0 greyin module azimut(wo. corrections)"
   int i j w m gr ;
   gr := grid
   imsetgrid 1
   w := imalloc imdepth g
   m := imalloc 1
   imset 0 md
   imset 0  az
   i := 1
   for 1 to 6 do
      dirgradient i g w
      masksup w md m
      greymask 0 m az
      masksupequal w md m
      imsup w md md
      j := (i - 1)
      imcopyplane m j az
     ++ i
  end
  imfree w
  imfree m
 imsetgrid gr
end
```

Puis le gradient vectoriel vrai est obtenu en éliminant les directions incohérentes et en tenant compte des directions conjuguées comme indiqué plus haut. Cette procédure utilise une anamorphose programmée grâce à la routine **imlookup** fournie dans la librairie standard. Le fichier d'anamorphose appelé **grad.lut** est un fichier contenant à la ligne i (codage des anciennes directions) le codage de la nouvelle direction de gradient. Par exemple, un codage initial égal à 7 (directions 1, 2 et 3) deviendra égal à 3 (direction 2) après correction. Douze

directions sont ainsi définies, les directions paires correspondants aux directions principales de la trame hexagonale, les directions impaires aux directions conjuguées.

```
deproc gradvect gradvect g md az
syntax "gradvect greyin module azimut"
    int w m ;
    w := imalloc imdepth g
    m := imalloc 1
    vectgrad0 g md w
    imlookup w az "grad.lut"
    imthresh az 0 0 m
    greymask 0 m md
    imfree w
    imfree m
end
```

L'image des azimuts du gradient ne prenant que treize valeurs n'est pas très visible. On peut cependant mieux la distinguer, soit en multipliant ses valeurs, soit en définissant une nouvelle palette de fausses couleurs (voir les fonctions permettant de réaliser cela dans le fichier d'aide de MICROMORPH).

Les images suivantes illustrent ce gradient sur l'image PETROLE.

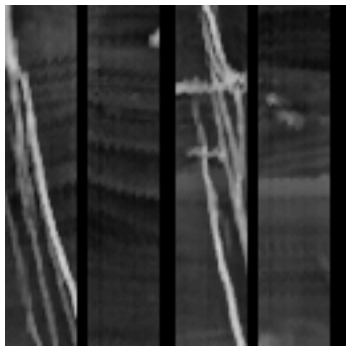
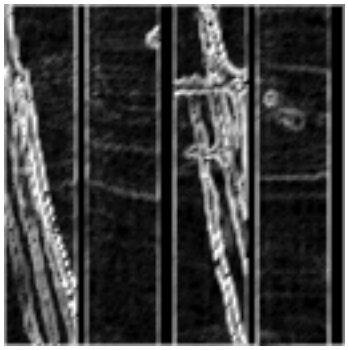
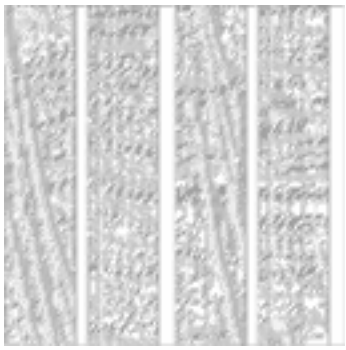


Image originale



(a)



(b)

a) module du gradient, b) azimut (8x)

RESUME

Les transformations suivantes doivent désormais faire partie de votre dictionnaire :

thinturn se1 se0 s d
procédure d'amincissement par toutes les rotations de l'élément structurant se de l'image binaire s dans l'image binaire d.
thin se1 se0 s d
procédure de squelette par amincissement complet par itération de l'image binaire s dans l'image binaire d.
Lthin s d
L-squelette par amincissement de l'image binaire s dans l'image binaire d.
Mthin s d
M-squelette par amincissement de l'image binaire s dans l'image binaire d.
Dthin s d
D-squelette par amincissement de l'image binaire s dans l'image binaire d.
thickturn se1 se0 s d
procédure d'épaississement par toutes les rotations de l'élément structurant se de l'image binaire s,placé dans l'image binaire d.
thick se1 se0 s d
procédure de squelette par épaississement complet par itération de l'image binaire s, placé dans l'image binaire d.
Lthick s d
L-squelette par épaississement de l'image binaire s, placé dans l'image binaire d.
Mthick s d
M-squelette par épaississement de l'image binaire s dans l'image binaire d.
Dthick s d
D-squelette par épaississement de l'image binaire s dans l'image binaire d.
endpoints s d
points extrêmes de l'image binaire s dans l'image binaire d.
mulpoints s d
points multiples de l'image binaire s dans l'image binaire d.
gdscentre s d
centre géodésique de l'image binaire s dans l'image binaire d.
gdsthinturn se1 se0 s m d
procédure d'amincissement géodésique par toutes les rotations de l'élément structurant se de l'image binaire s dans l'image binaire d.
gdsthin s m d
procédure de squelette géodésique complet par itération de l'image binaire s dans l'image binaire d.
gdsthickturn se1 se0 s m d
procédure d'épaississement géodésique par toutes les rotations de l'élément structurant se de l'image binaire s dans l'image binaire d.
gdsthick s m d
procédure de squelette géodésique complet par itération de l'image binaire s dans l'image binaire d.
greyseero se s d

érosion par tout élément structurant se défini sur l'hexagone élémentaire de l'image numérique s dans l'image numérique d.

greysedil se s d

dilatation par tout élément structurant se défini sur l'hexagone élémentaire de l'image numérique s dans l'image numérique d.

greythinstep sei ses s d

amincissement par tout élément structurant défini sur l'hexagone élémentaire de l'image numérique s dans l'image numérique d.

greythickstep sei ses s d

épaississement par tout élément structurant défini sur l'hexagone élémentaire de l'image numérique s dans l'image numérique d.

gradvect g md az

gradient en module (md) et azimut (az) de l'image numérique g.

Chapitre 9

LIGNE DE PARTAGE DES EAUX

9.1. Zone d'influence

Rappelons brièvement ce qu'on appelle zone d'influence d'une composante connexe d'un ensemble.

Soit un ensemble X, composé de plusieurs sous-ensembles connexes :

$$X = \bigcup_i X_i$$

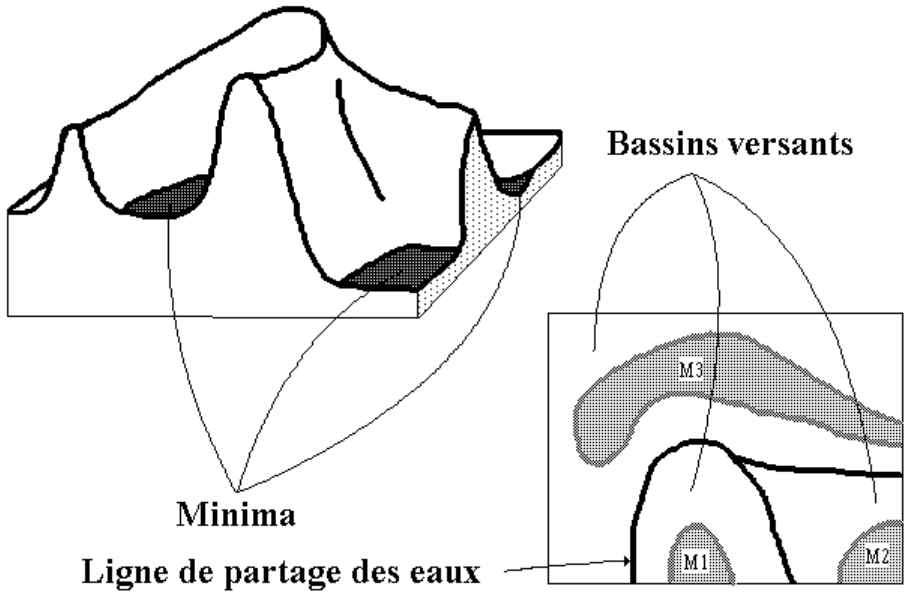
On appelle zone d'influence $Z(X_i)$ de la composante connexe X_i, l'ensemble des points de l'espace plus proches de X_i que de toute autre composante connexe de X.

$$x \in Z(X_i) \Leftrightarrow d(x, X_i) < d(x, X_j), \forall j \neq i$$

Les points de l'espace n'appartenant à aucune zone d'influence sont les points du squelette par zones d'influence, ou SKIZ (pour Skeleton by Influence Zones).

9.2. Ligne de partage des eaux

Soit f, une image numérique. f est supposée prendre des valeurs discrètes dans l'intervalle $[h_{min}, h_{max}]$.



Ligne de partage des eaux par inondation

Notons $T_h(f)$ le seuil de f au niveau h :

$$T_h(f) = \{p, f(p) \leq h\}$$

$Min_h(f)$ est l'ensemble des minima de f au niveau h, et $IZ_A(B)$ représente la zone d'influence de B dans A.

L'ensemble des bassins versants d'une image f est égal à l'ensemble $X_{h_{min}}$ obtenu après la récursion suivante :

(i) $X_{h_{min}} = T_{h_{min}}(f)$

(ii) $\forall h \in [h_{min}, h_{max} - 1],\ X_{h+1} = Min_{h+1} \cup IZ_{T_{h+1}(f)}(X_h)$

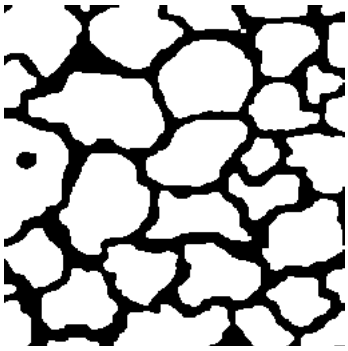
La ligne de partage des eaux (LPE) de f correspond au complémentaire de $X_{h_{max}}$, c'est-à-dire, l'ensemble des points qui n'appartiennent à aucun bassin versant.

La récursion détaillée ci-dessus peut être perçue comme un processus d'inondation. L'image f est alors vue comme une surface topographique et des trous sont percés en chacun de ses minima régionaux. On plonge alors progressivement la surface topographique dans l'eau. Des barrages sont construits chaque fois que les eaux provenant de deux minima régionaux distincts se rencontrent, afin d'éviter qu'elles ne se mélangent. A la fin du processus d'inondation, les barrages correspondent à la ligne de partage des eaux de f, et ils délimitent les bassins versants de f.

EXERCICES

Exercice n° 1 : Squelette par zones d'influence

L'image ALUMINE représente une section polie de grains d'alumine. Les grains métalliques sont en fait jointifs dans le matériau. Le joint de grains ayant une épaisseur qui n'excède pas quelques angströms, il est donc parfaitement invisible. Pour le mettre en évidence, il faut procéder à une attaque chimique. Cette attaque chimique grossit énormément les joints de grains. Pour étudier les relations de voisinage des grains, il est nécessaire d'amincir les joints.



ALUMINE

1) Effectuez le SKIZ des grains.

2) Montrez que le squelette par zones d'influence peut s'obtenir par une ligne de partage des eaux en utilisant judicieusement la fonction distance.

3) Comparez, à l'aide de l'image COFFEE, le résultat obtenu par cette méthode à celui obtenu à l'aide de la transformation directe **skiz**.

[Le SKIZ équivaut à une LPE sur la fonction distance]

[procédures **clip** ; **skiz**]

Solution

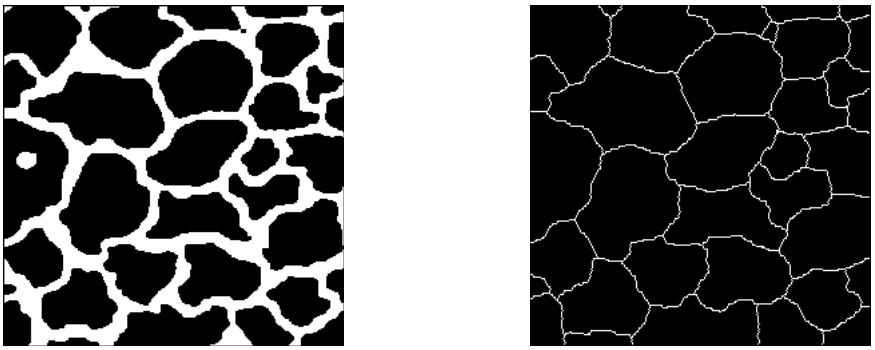
1) Le squelette par zone d'influence peut être effectué en combinant un squelette par épaississement suivi d'un ébarbulage.

On utilisera la procédure **Mthick** et **clip**. Donnons la définition de **clip** :

```
deproc clip clip s d
syntax "clip binin binout"
  int se0 se1 ;
  if (grid <> 0) then
     se1 := 1 se1 := 30
  else
     se1 := 129 se0 :=62
  end
  thin se1 se0 s d
end
```

On peut donc définir le mot **skiz** :

```
deproc skiz skiz s d
syntax "skiz binin binout"
  if (grid <> 0) then
    thick 2 56 s d
  else
    minidil s d
    thick 258 120 d d
  end
  iminv d d
  clip d d
end
```



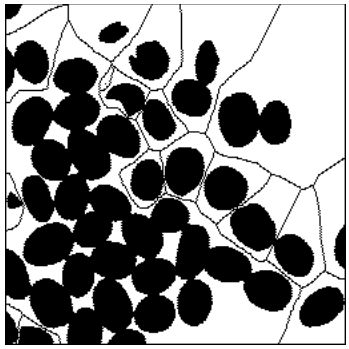
(a) (b)

Squelette par zones d'influence
(a) original, (b) zones d'influence

2) Soit X un ensemble. Considérons la fonction distance de X^c . Le squelette par zone d'influence de X apparaît alors simplement comme les lignes de crête de cette fonction. Or, ce sont précisément ces lignes de crête que la ligne de partage des eaux met en évidence. Les minima régionaux sont constitués des composantes connexes de X.

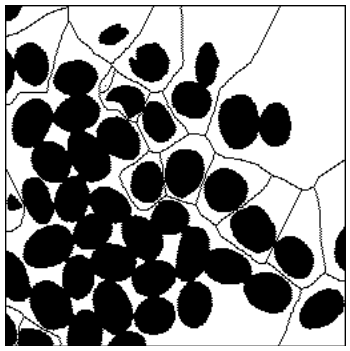
L'algorithme s'écrit ainsi (l'image initiale est supposée être en b1) :

```
iminv b1 b2
distance b2 g1
threshwshed g1 b3
```



SKIZ par ligne de partage des eaux de la fonction distance de X

3) Le "guide" constitué par la fonction distance lors des skiz successifs permet d'éviter pour une bonne part les problèmes de digitalisation et d'ordre de passage des éléments structurants lors des épaississements successifs (ceci est peu perceptible sur cet exemple).



SKIZ obtenu par binskiz

Exercice n° 2

Programmez de même le squelette par zone d'influence géodésique.
[procédure **gdsskiz**]

Solution

Le squelette par zones d'influence géodésique est réalisé de la façon suivante :

```
deproc gdskiz gdskiz s m d
syntax "gdskiz binin binmask binout"
  int i1 i2 gr ;
  gr := grid imsetgrid 1
  imcopy s d
  i1 := imvolume d
  while (i1 <> i2) do
```

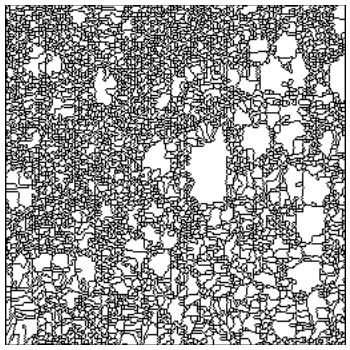
```
    gdsthickturn 2 56 d m d
    thick 30 1 d d
    imand d m d
    i2 := i1
    i1 := imvolume d
  end
  imsetgrid gr
end
```

Exercice n° 3

Programmez la ligne de partage des eaux énoncée ci-dessus. Appliquez-la à l'image ELECTROP.
[procédures **wshed** ; **mwshed**]

Solution

Nous avons vu que les minima sont les composantes connexes du seuil entre 0 et N et qui ne peuvent pas être reconstruites à partir du seuil entre 0 et N-1. L'algorithme ci-dessous combine, afin d'accélérer la procédure, la détermination des minima et le processus d'inondation (skiz géodésiques).



Ligne de partage des eaux de l'image ELECTROP

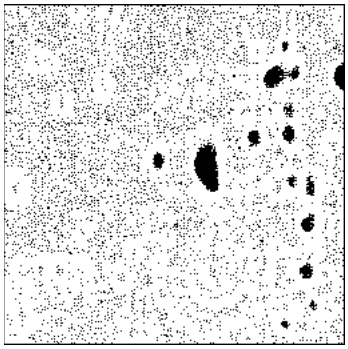
```
deproc threshwshed threshwshed s d
syntax "threshwshed greyin binout"
  int min max SN N w sn ;
  w := imalloc 1
  SN := imalloc 1
  min := imabsmin s
  max := imabsmax s
  imset 0 d
  N := min
  for min to max do
    imthresh s 0 N SN
    gdskiz d SN d
    imcopy d w
    build SN w
    imdiff SN w w
    imsup w d d
```

```
    ++ N
  end
  imfree SN
  imfree w
end
```

Cette méthode, si elle est longue, a cependant l'avantage d'être peu gourmande en espace mémoire (seules des images binaires sont utilisées).

Le résultat peut surprendre par sa piètre qualité. Ceci doit vous forcer à garder à l'esprit que l'image résultat possède autant de bassins versants que l'image initiale possède de minima régionaux. Notre image étant ici très bruitée, un grand nombre de minima "parasites" sont présents comme le montre l'image ci-après.

On peut également calculer la ligne de partage des eaux en réduisant le nombre de seuils pris en compte. La procédure suivante effectue ce genre d'opération avec une progression géométrique ou arithmétique des seuils. On remarquera également que, dans ce cas, les minima de la fonction initiale sont utilisés, ce qui rend le résultat final assez proche du précédent avec cependant un gain de vitesse appréciable.



Minima de l'image ELECTROP

```
deproc wshed wshed s d1 d2 t
syntax "wshed: greyin; binout1(divides line) binout2(minima); type(in) {t=0 for
geom.progression , t=1 for arithm. one }"
  int min max i w  k  ;
  w := imalloc 1
  min := imabsmin s
  max := imabsmax s
  minima s d2
  imcopy d2 w2
  imcopy d2 d1
  k := ( max - min )
  if ( t = 0 ) then
    i := 1
    while ( i <  k ) do
      imthresh s 0 ( min + i - 1 ) w
      imsup d2 w w
      gdskiz d1 w d1
      i := ( 2 * i )
    end
```

```
  else
    i := 0
    while ( i < k ) do
      imthresh w1 0 (min + i) w
      imsup d2 w w
      gdskiz d1 w d1
      i := i + ( k / 10 )
    end
  end
  imthresh s 0 max w
  gdskiz d1 w d1
  imfree w
end
```

RESUME

Les transformations suivantes doivent désormais faire partie de votre dictionnaire :

clip s d
ébarbulage de l'image binaire s dans l'image binaire d.

skiz s d
squelette par zones d'influence de l'image binaire s dans l'image binaire d.

gdskiz s m d
squelette par zone d'influence géodésique de l'image binaire s dans l'image binaire d.

wshed s d1 d2 type
Ligne de partage des eaux par inondation de l'image numérique s placée dans l'image binaire d1; d2 contient les minimaux de s; type=1 correspond à une progression arithmétique des seuils et type=0 à une progression géométrique.

Chapitre 10

SEGMENTATION

Ce chapitre illustre l'utilisation de la ligne de partage des eaux introduite au chapitre précédent dans quelques problèmes de segmentation d'images binaires ou numériques.

EXERCICES

Exercice n° 1

L'électrophorèse bidimensionnelle est une technique de séparation et d'identification de protéines. Les protéines migrent sur le gel en fonction de leur poids moléculaire et de leur charge électrique. Sur l'image ELECTROP, on désire extraire le contour de chaque spot de protéines.

1) Déterminez les minima régionaux de l'image. Qu'en concluez-vous? Quelles transformations peut-on faire subir à l'image? Déterminez les nouveaux minima. Nous travaillerons désormais sur cette image.

2) Déterminez le gradient morphologique de taille 1. Déterminez les minima du gradient. Effectuez la ligne de partage des eaux à l'aide de la procédure wshed. Le résultat est-il satisfaisant?

3) Nous allons tenter de parvenir à un meilleur résultat. Pour cela nous allons chercher à déterminer des marqueurs situés à l'intérieur des taches à contourer ainsi que des marqueurs extérieurs.

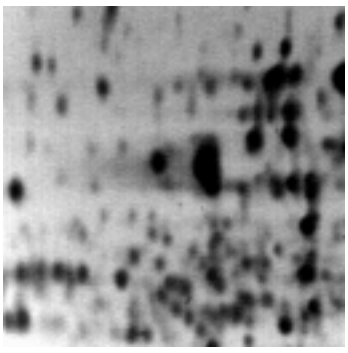
- Que peut-on prendre comme marqueurs intérieurs?
- Et comme marqueurs extérieurs?
- Modifier l'image du gradient pour qu'elle ne possède que ces marqueurs comme minima (utiliser pour cela la reconstruction numérique).
- Effectuez la ligne de partage des eaux de cette nouvelle image.

[procédures **swamping**]

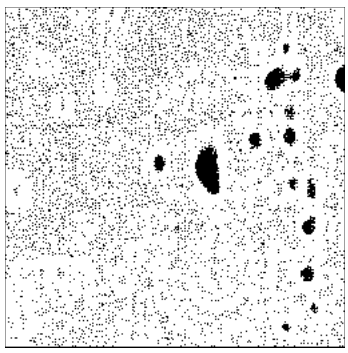
Solution

1) Chargez l'image ELECTROP dans g1 et faites :

minima g1 b1



(a)



(b)

(a) image originale, (b) minima

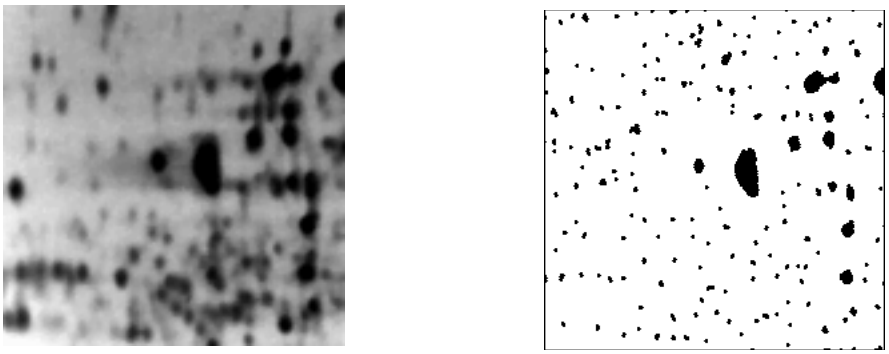
Les minima sont extrêmement nombreux, indice d'une image fortement bruitée. Il paraît peu raisonnable de travailler directement sur cette image; nous allons donc lui faire subir un léger filtrage. Une ouverture suivie d'une fermeture de taille 1 conviendront parfaitement à notre problème. On constate en effet que les nouveau minima sont plus significatifs (voir page suivante).

On effectuera donc les opérations suivantes :

open g1 g1 1
close g1 g1 1
minima g1 b1

2) Calculons le gradient morphologique de l'image après filtrage et déterminons ses minima :

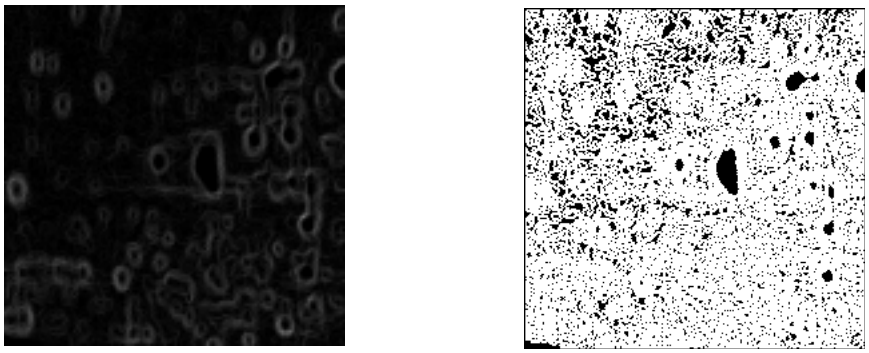
gradient g1 g2 1
minima g2 b2



(a) (b)

(a) image filtrée, (b) minima

Le gradient possède lui aussi beaucoup de minima. On y trouve bien entendu les minima de l'image elle-même ainsi que les maxima (points de gradient nul) mais aussi d'autres points (points de gradient minimum non nul). Rappelons que par gradient nous entendons toujours "module du gradient".



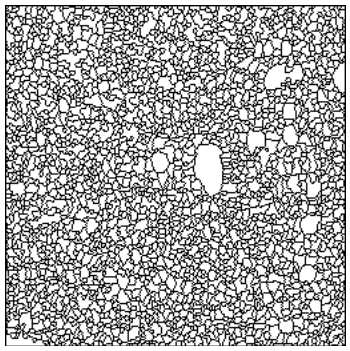
(a) (b)

(a) gradient, (b) minima du gradient

Effectuons la ligne de partage des eaux de l'image gradient à l'aide de la procédure par inondation programmée précédemment.

threshwshed g2 b2

Le résultat est très "sur-segmenté" et ne nous satisfait pas complètement. Il y a pourtant des raisons d'être optimiste. On peut constater en effet que les contours recherchés font partie des contours détectés même si beaucoup sont surnuméraires.



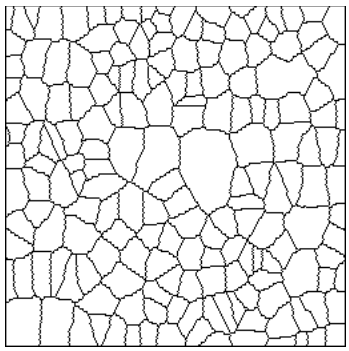
ligne de partage des eaux du gradient

3) Amélioration de la segmentation

- L'objectif est de pouvoir marquer de manière unique chaque tache. Or les minima de l'image filtrée déterminés plus haut paraissent tout à fait adaptés.
- Pour bien contourer les taches il nous faut non seulement savoir où elles sont mais aussi où elles ne sont pas. Nous savons qu'alors un contour pertinent pourra être trouvé sous la forme des lignes de crête du gradient situées entre les marqueurs intérieurs et les marqueurs extérieurs.

Pour cela on pourrait prendre le squelette par zone d'influence des minima de l'image :

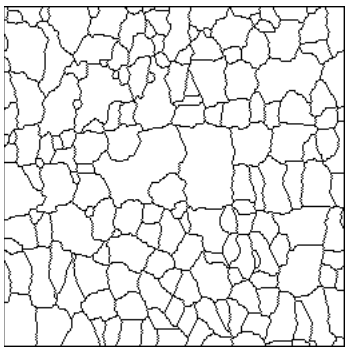
skiz b1 b2



SKIZ des minima de l'image

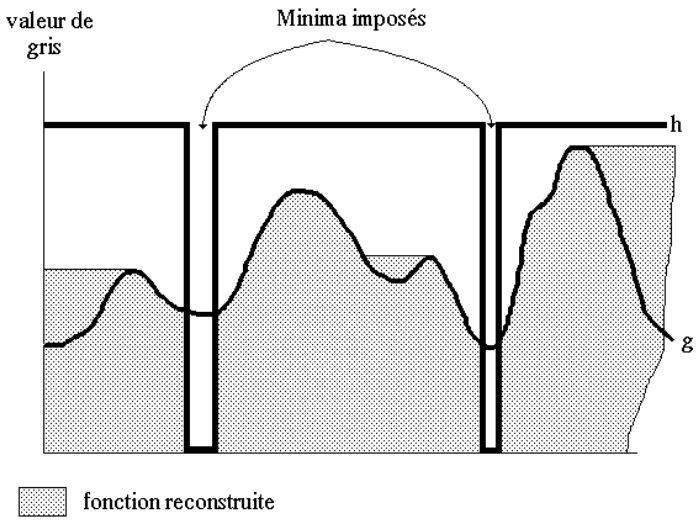
Cependant, ceci ne tient pas compte de la topographie effective de l'image, notamment de la dimension des taches. C'est pourquoi il est plus judicieux d'effectuer une ligne de partage des eaux de l'image. Les bassins versants se substituent aux zones d'influence déterminées ci-dessus. On est sûr maintenant que les lignes de séparation passent à l'extérieur des taches.

threshwshed g1 b2



ligne de partage des eaux de l'image

- La procédure consiste à construire une image appelée marqueur que nous allons éroder conditionnellement à l'image du gradient comme l'illustre la figure ci-dessous.



Modification de l'image gradient, principe

L'image b1 contient les marqueurs intérieurs, b2 contient les marqueurs extérieurs, g2 contient l'image gradient.

imcopy b1 b3
imor b2 b3 b3 {union des marqueurs intérieurs et extérieurs}
iminv b3 b3
immask b3 0 255 g3

L'image g3 est l'image masque utilisée par la reconstruction.

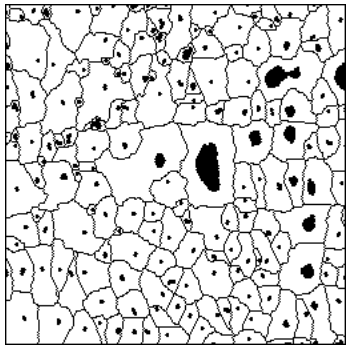
Seule la reconstruction rapide par dilatation géodésique est implantée. Nous allons donc réaliser cette opération sur les images inversées.

imcadd 1 g2 ; car le gradient possède des minima nuls
iminf g2 g3 g3
iminv g2 g2
iminv g3 g3
build g2 g3

iminv g2 g2
iminv g3 g3

On peut vérifier que l'image reconstruite possède bien les seuls minima désirés. Pour cela on effectue :

minima g3 b3

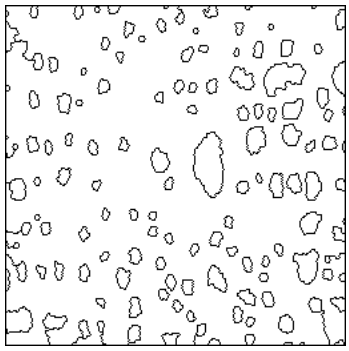


minima imposés

- La ligne de partage des eaux de ce gradient modifié nous donne le résultat désiré.

On utilisera la LPE par seuils :

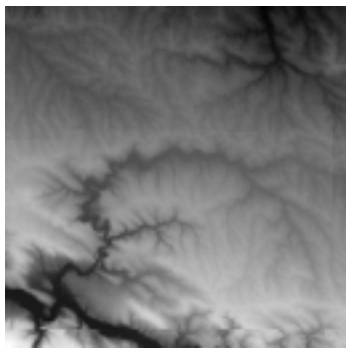
threshwshed g3 b4



ligne de partage des eaux finale

Exercice n° 2 : modèle numérique de terrain

Cette étude utilise la segmentation d'images par la ligne de partage des eaux. Elle est consacrée à l'extraction des bassins versants d'un modèle numérique d'altitude.



RELIEF

L'image RELIEF représente un Modèle Numérique de Terrain (MNT). Dans ce modèle, les altitudes de la surface topographique sont échantillonnées aux noeuds d'une grille carrée. Le but de cette étude est de définir un algorithme permettant l'extraction des bassins versants de la topographie. On pourrait penser qu'il n'y a là aucune difficulté puisqu'il existe déjà une telle transformation dans la boîte à outils dont vous disposez. On verra néanmoins que cette application est plus difficile qu'il n'y paraît à cause de la présence de bruit dans l'image.

1) Extrayez les minima régionaux de l'image RELIEF.

2) Segmentez l'image RELIEF en ses différents bassins versants et vérifiez qu'à chaque minimum régional correspond bien un et un seul bassin versant.

La présence de minima régionaux parasites à l'intérieur du modèle numérique de terrain provoque une sur-segmentation. En réalité, en supposant qu'il n'existe pas de dépressions fermées, tous les minima régionaux doivent correspondre à des exutoires et apparaître au bord du champ de l'image.

3) Quelle procédure peut-on utiliser pour supprimer les minima régionaux situés à l'intérieur du modèle. Visualisez le masque des pixels modifiés par cette procédure. Quelles sont les propriétés de cette transformation?

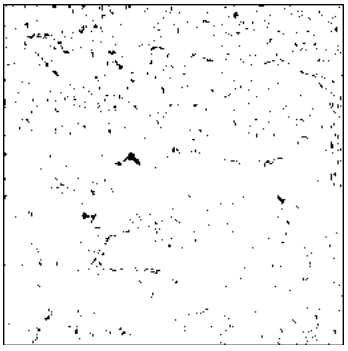
4) Appliquez la transformation de la ligne de partage des eaux sur le RELIEF modifié.

5) Proposez et testez une méthode pour obtenir le bassin versant relatif à un point quelconque du modèle.

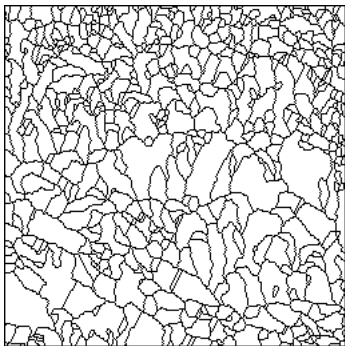
6) Quelle stratégie adopteriez-vous s'il existait réellement des dépressions fermées sur la surface topographique (par exemple, dolines ou cratères volcaniques)?

Solution

1) On utilise pour cela la procédure minima définie précédemment.



minima de l'image RELIEF

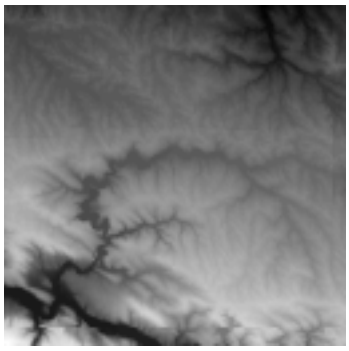


LPE de l'image RELIEF

2) La procédure watershed permet de segmenter l'image en ses différents bassins versants (l'image est supposée être dans b1). Remarquez la sur-segmentation obtenue.

threshwshed g1 b1

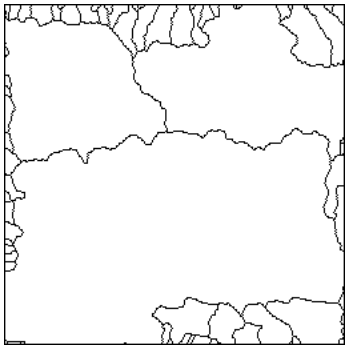
3) On utilise ici la procédure **clohole** vue au chapitre 5.



Elimination des cuvettes intérieures

Cette opération possède toutes les propriétés d'une fermeture : idempotence, croissance et extensivité.

4) Cette fois-ci, on obtient bien la segmentation désirée. Tous les bassins sont connectés au bord du champ de l'image.

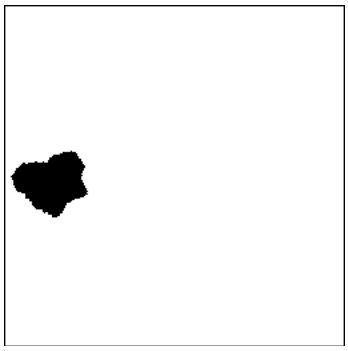


Bassins versants vrais de l'image RELIEF

5) Pour obtenir le bassin versant relatif à un point quelconque, il suffit de créer un minimum local à l'endroit souhaité, et de relancer la transformation de LPE.

Effectuons par exemple les opérations suivantes, le modèle numérique "bouché" étant en g1.

imwritepix 0 60 140 g1
threshwshed g1 b1
imset 0 b2
imwritepix 1 60 140 b2
iminv b1 b1
build b1 b2

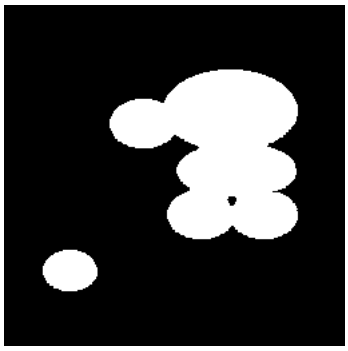


Bassin versant relatif à un point

6) L'opérateur de ligne de partage des eaux génère pour chaque minimum régional un et un seul bassin versant. Si l'on désire conserver les bassins versants associés à certaines dépressions situées à l'intérieur même de l'image, il nous faudra partir d'une image où ces dépressions sont encore marquées par un minimum régional. Le bouchage des trous "à partir du bord" utilisé plus haut se révèle ici inadapté. Il nous faudra d'abord obtenir par des moyens exogènes les marqueurs des dépressions à conserver et procéder à un bouchage de trous où ces marqueurs seront ajoutés au marqueur que constitue le bord (le choix d'un point arbitraire dans l'image comme vu en 5) est un moyen exogène d'obtention d'un marqueur).

Exercice n°3 : Séparation de particules (Première approche)

Vous avez déjà déterminé (chapitre 7, exercice n° 1) l'érodé ultime de l'image CELLS. En reprenant cette même image, essayez d'élaborer un algorithme de segmentation des cellules utilisant les seules opérations binaires.



CELLS

1) Utilisez le squelette géodésique, ou mieux, le SKIZ géodésique des érodés ultimes dans l'ensemble de départ. Etes-vous satisfait de la segmentation?

2) Comment améliorer cette segmentation? [tenir compte de l'ordre des érodés ultimes]. Reprenez la définition de la ligne de partage des eaux et appliquez-la à la fonction :

$$f(x) = d(x,X^c)$$

- A quoi correspondent les seuils Y_λ de la fonction f précédente :

$$Y_\lambda = \{x : d(x, X^c) > \lambda\}$$

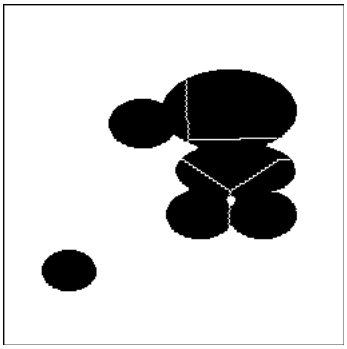
- Quelle relation y a-t-il entre l'érodé ultime de X et les minima de f(x)?
- Programmez l'algorithme.

3) Comparez le résultat obtenu à celui que l'on obtiendrait en appliquant la ligne de partage des eaux déjà implantée dans MICROMORPH (**wshed**), à la fonction distance inversée (obtenue par le mot **distance**, lui aussi existant).

Solution

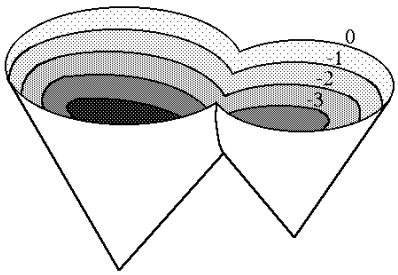
1) Le squelette par zone d'influence géodésique de l'érodé ultime donne l'image suivante (l'image CELLS est dans b1) :

binultim b1 b2
gdskiz b2 b1 b3



La segmentation n'est pas très bonne, car la procédure ne tient aucun compte du moment d'apparition des diverses composantes de l'érodé ultime.

2) Les lignes de séparation correspondent en fait à la ligne de partage des eaux de la fonction f(x) = -d(x,X) (distance à la frontière).



La ligne de partage des eaux d'une fonction f peut être programmée de façon itérative, en construisant étape par étape l'ensemble W correspondant aux bassins versants de f. On utilise pour cela les différents seuils X_i de la fonction f.

$$X_i = \{x ; f(x) < i\}$$

Dans le cas présent la fonction f étant définie par :

$$f(x) = -d(x, X^c)$$

elle est négative ou nulle.

Soit x un point de X tel que : $f(x) \leq -i$. Alors : $d(x, X^c) \geq i$. La boule de rayon i centrée en x appartient donc à X_i et $x \in X \ominus iB$.

Soit i_{max} la plus petite taille d'érosion de X telle que $X \ominus i_{max}B = \emptyset$. Alors g(x) : f(x) + i_{max} est positive ou nulle et a même LPE que f.

Soit X_i un seuil de g(x) :

$$X_i = \{x; g(x) \leq i\} = \{x; f(x) \leq i - i_{max}\}$$

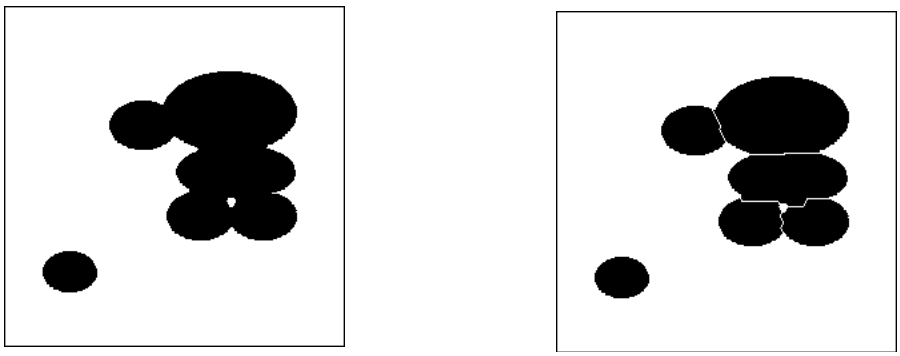
Ce qui donne :

$$X_i = X \ominus (i_{max} - i)B$$

La procédure peut donc être définie de la façon suivante :

```
deproc binsegment binsegment s d
syntax "binsegment binin binout"
   int i j w1 w2;
   w1 := imalloc 1
   w2 := imalloc 1
   imcopy s d
  while imvolume d do
      ++ i
      ero d d 1
   end
   j := i
   while j do
       imcopy s w1
       ero w1 w1 -- j
       gdskiz d w1 d
       gdsdil d w1 w2 1
       imdiff w1 w2 w2
       imsup w2 d d
   end
   imfree w1
   imfree w2
end
```

Chargez l'image CELLS dans b1 et lancez la procédure.



(a) (b)

Segmentation par LPE, (a) image originale, (b) segmentation

On constate immédiatement que les minima de la fonction distance ajoutés au fur et à mesure de leur apparition correspondent aux différentes composantes connexes de l'érodé ultime.

La segmentation appliquée à l'image CELLS fait bien apparaître les séparations au niveau des jonctions.

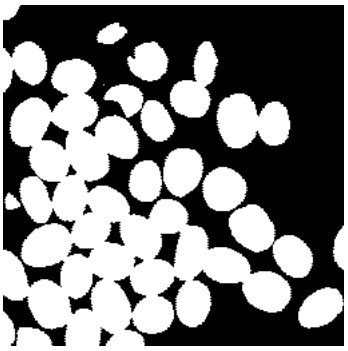
La digitalisation, cependant, produit des séparations non rectilignes.

3) L'opération watershed implantée dans MICROMORPH ne procède pas par épaississements géodésiques. De ce fait, elle subit moins les aléas de la digitalisation comme on peut le constater sur l'image obtenue (l'image CELLS est dans b1) :

distance b1 g1
iminv g1 g1
threshwshed g1 b2
imdiff b1 b2 b3

Exercice n° 4 : Séparation de particules (Deuxième exemple)

L'exercice précédent a montré comment programmer à l'aide des seules opérations binaires la segmentation par ligne de partage des eaux. Elle a également présenté le cas idéal d'une segmentation qui fonctionne tout de suite. Il n'en va pas toujours ainsi comme va le prouver l'exemple de l'image COFFEE.



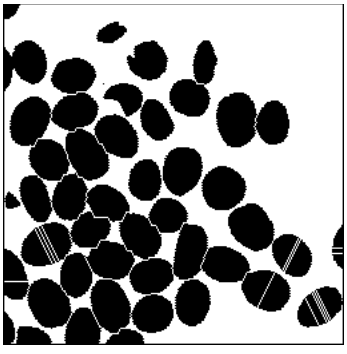
COFFEE

1) Appliquez l'algorithme de segmentation déjà vu. Que constatez-vous? Quelles en sont les raisons?

2) Montrez comment un filtrage de la fonction distance permet de surmonter cette difficulté.

Solution

1) L'utilisation directe de l'algorithme de segmentation sur l'image COFFEE se traduit par ce que l'on appelle couramment une "sur-segmentation". Des composantes qui sont à l'évidence constituées d'un seul grain de café sont découpées en plusieurs composantes connexes.



Si l'on y regarde de plus près, on constate cependant que chaque nouvelle composante connexe est bel et bien marquée par un seul érodé ultime. La segmentation n'a de valeur que celle de l'érodé ultime : si celui-ci marque de plusieurs composantes connexes un même grain de café, ce grain sera à la fin découpé en plusieurs morceaux.

Mais pourquoi l'érodé ultime engendre-t-il autant de marqueurs pour un même grain? Ceci est dû en fait aux irrégularités des contours des grains : une petite concavité sur le bord peut entraîner la scission de l'érodé ultime, que l'on peut encore interpréter en termes de maxima régionaux de la fonction distance.

2) Le problème qui se pose est : comment parvenir à n'obtenir qu'un seul marqueur par grain de café, l'érodé ultime ayant échoué dans cette tâche?

On pourrait agir directement sur l'image elle-même, afin de faire disparaître les concavités, objets de notre ressentiment. Cette méthode est cependant difficile à mettre en oeuvre : il n'est pas simple de boucher des concavités dans le domaine digital, et il ne faudrait pas faire disparaître les concavités pertinentes, indices de la présence de deux grains jointifs.

En fait on préfère agir sur les érodés ultimes eux-mêmes. Mieux encore, ceux-ci pouvant être considérés comme maxima régionaux de la fonction distance, nous allons nous attacher à transformer la fonction distance en une nouvelle fonction dont les maxima seront de meilleurs marqueurs des grains.

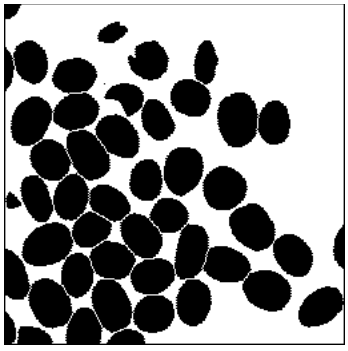
Nous avons déjà vu que les maxima régionaux étendus permettaient d'obtenir de meilleurs marqueurs. Le filtrage de la fonction distance correspondant est une opération d'ouverture par soustraction d'une constante et reconstruction.

En vertu des propriétés particulières de la fonction distance (qui n'est pas une image numérique comme les autres) on obtient le même résultat en effectuant une ouverture par érosion-reconstruction (puisque la fonction distance de l'érodé est obtenue par soustraction de la taille de l'érosion de la fonction distance originale).

En revanche, on n'obtiendra pas le même résultat en effectuant une ouverture simple de la fonction distance.

On peut également faire une fermeture sur la fonction distance, ce qui revient en fait à faire la même fermeture sur l'image binaire des érodés ultimes.

distance b1 g1
buildopen g1 g1 3
iminv g1 g1
threshwshed g1 b2
imdiff b1 b2 b2



Chapitre 11

MESURES

11.1. Rappels

Toute transformation morphologique doit tôt ou tard aboutir à une mesure sur l'ensemble transformé. La transformation (ou la suite de transformations) a pour fonction essentielle de mettre en évidence les objets à mesurer : les ouvertures mettant en évidence les granulométries en sont un exemple.

Les mesures vérifiant de bonnes propriétés de compatibilité avec les translations et les homothéties sont peu nombreuses. Ce sont :

- la surface
- les variations diamétrales
- le périmètre
- le nombre de connexité

11.2. Mesures et transformations morphologiques

Toute mesure est constituée d'un doublet : transformation + comptage des points de l'image transformée. Pour la surface, la transformation est très simple : l'identité. Mesurer une surface revient donc à calculer le nombre de points de l'ensemble. Cette mesure est d'ailleurs la seule dont vous disposiez déjà dans les primitives du langage MICROMORPH.

Aux variations diamétrales correspondent les nombres d'intercepts sur une image digitalisée :

$$I_1 = N(01)\ ;\ I_2 = N\begin{pmatrix} & 1 \\ 0 & \end{pmatrix}\ ;\ I_3 = N\begin{pmatrix} 1 & \\ & 0 \end{pmatrix}$$

La transformation associée à cette mesure est donc celle qui consiste à mettre en évidence les intercepts.

Le calcul du nombre de connexité est un peu particulier, puisqu'il nécessite deux transformations et deux comptages :

$$v = N\begin{pmatrix} 0 & 0 \\ & 1 \end{pmatrix} - N\begin{pmatrix} & 0 & \\ 1 & & 1 \end{pmatrix}$$

Rappelons que dans le plan, le nombre de connexité s'interprète comme le nombre de composantes connexes de l'ensemble, diminué du nombre de trous.

Programmer des mesures ne suffit pas. Il faut aussi savoir interpréter les résultats. Certains exercices doivent vous permettre de vous familiariser avec cette étape importante des traitements morphologiques.

11.3. Suggestions

Dans les exercices présentés dans ce chapitre, on fait une large utilisation des teneurs. Ces valeurs comprises entre 0 et 1 ne sont donc pas manipulables par MICROMORPH qui ne

reconnaît que les valeurs entières. Pour contourner cette difficulté, nous utiliserons les teneurs exprimées en %, valeurs entières comprises entre 0 et 1000 dans le cas binaire et entre 1 et 100 dans le cas numérique (cf. la procédure **granul**).

EXERCICES

Exercice n° 1

1) Programmez les mesures des nombres d'intercepts dans les diverses directions de la trame hexagonale.

2) Programmez la mesure du nombre de connexité. Vérifiez vos algorithmes à l'aide des images HOLES, CELLS, etc. .

3) Programmez le calcul du périmètre. Donnez plusieurs façons de calculer le périmètre, et comparez leurs précisions respectives.

[procédures **diameter** ; **cnumber** ; **digperim** ; **perim**]

Solution

1) Programmation des nombres d'intercepts

```
defunc diameter diameter dir s
syntax "diameter dir binin -> diametral variation, i.e.number of intercepts"
   int w eg ;
   w := imalloc 1
   eg := edge
   imsetedge 0
   iminv s w
   iminfngb s w dir 1
  diameter := imvolume w
   imsetedge eg
   imfree w
end
```

2) Programmation du nombre de connexité

Pour obtenir le nombre de connexité, il suffit d'appliquer la formule fournie dans le manuel d'exercices. Il faut cependant veiller à positionner l'origine du deuxième élément structurant sur le point 1, faute de quoi la mesure serait biaisée sur le bord du champ d'analyse.

```
defunc cnumber cnumber s
 syntax "cnumber binin; connectivity number of binin"
   int w eg ;
   w := imalloc 1
   eg := edge
   imsetedge 0
   oldgrid := grid
   imsetgrid 1
   imcopy s w
   imdiffngb s w 1
   imdiffngb s w 6
```

```
    cnumber := ( imvolume w )
     imcopy s w
     iminfngb  s w 2
     imdiffngb s w 1
     cnumber := (cnumber - ( imvolume w ) )
     imsetgrid oldgrid
     imsetedge eg
     imfree w
  end
```

3) Calcul du périmètre

Parmi les différentes façons (bonnes ou mauvaises) d'obtenir le périmètre, on peut citer:

- la mesure du contour intérieur

Cette mesure du contour intérieur consiste à calculer le nombre de configurations :

```
    0
  1   1
```

ainsi que toutes leurs rotations. Les configurations :

```
    0
  1   1
    0
```

doivent être comptées deux fois.

La procédure permettant ce calcul peut s'écrire comme pour les mesures précédentes à l'aide des primitives de décalage d'images de MICROMORPH.

```
defunc digperim digperim s d
syntax "digperim binin binout -> digital contour length for hex. grid"
   int w  w1 w2 oldgrid i ;
  oldgrid := grid  imsetgrid 1
  w := imalloc 1
  w1 := imalloc 1
  w2 := imalloc 1
  iminv s w2
  imset 0 d
  i := 1
  for 1 to 6 do
    imcopy s w1
    iminfngb s w1 i 1
    iminfngb  w2 w1 (i mod 6 + 1) 1
   digperim := ( digperim + imvolume w1)
    imor w1 d d
    i := (i + 1)
  end
  imfree w
  imfree w1
  imfree w2
  imsetgrid oldgrid
end
```

- Mesure du contour extérieur

La mesure du contour extérieur revient à mesurer le contour de l'ensemble complémentaire. On pourrait inverser l'ensemble X et lancer la procédure digperim sur X^c . Cependant, des effets de bord apparaissent dus au fait que la complémentation ne peut se faire qu'à l'intérieur du champ d'analyse.

- La formule de Cauchy (ou du moins sa version digitale)

Cette formule lie le périmètre à la moyenne des variations diamétrales D_α dans toutes les directions α :

$$L = \frac{1}{2} \int_{2\pi} D_\alpha d\alpha = \int_{\pi} D_\alpha d\alpha$$

Sa version digitale est donnée par :

$$L = \Sigma I_i$$

ou I est le nombre d'intercepts dans la direction i (3 directions principales).

On pourrait se contenter d'ajouter les trois valeurs d'intercepts données par bininterc1, bininterc2 et bininterc3. Cependant, la mesure du périmètre sur l'ensemble complémentaire donnerait des résultats différents dus aux mêmes effets de bord que ceux rencontrés lors de l'élaboration de binperim2.

Afin d'obtenir une mesure symétrique donnant des résultats identiques sur X et X , il suffit de postuler que tout intercept calculé sur le bord du champ ne peut intervenir dans le calcul du périmètre. Cependant, il faut alors considérer les intercepts de X dans les six directions de la trame hexagonale. En effet, l'hypothèse précédente fait qu'il peut ne pas y avoir le même nombre d'intercepts dans la direction 1 et dans la direction 4. La formule digitale est alors :

$$L = \frac{1}{2} \sum_{i=1}^{6} I_i$$

```
defunc perim perim s
syntax "perim binin -> value: perimeter from Cauchy formula"
  int w1 w2 i perim1 perim2 eg ;
  w1 := imalloc 1
  w2 := imalloc 1
  eg := edge
  imsetedge 0
  iminv s w1
  if (grid = 0) then
    i := 1
    for 1 to 4 do
      imcopy w1 w2
      iminfngb s w2 (2 * i) 1
      perim1 := (perim1 + imvolume w2)
      i := ( i + 1)
    end
    perim1 := (( perim1 * 5) / 7)
    i := 1
    for 1 to 4 do
      imcopy w1 w2
     iminfngb s w2 ((2 * i) - 1) 1
     perim2 := (perim2 + imvolume w2)
     i := (i + 1)
   end
```

```
    perim := (((perim1 + perim2) * 11) / 28)
    else
      for 1 to 6 do
        imcopy w1 w2
        iminfngb s w2 ++ i 1
        perim :=  (perim + imvolume w2)
      end
     perim := (perim * 11 / 21)
    end
    imsetedge eg
    imfree w1
    imfree w2
  end
```

Cette dernière mesure est la seule symétrique. On pourrait symétriser les deux premières mesures, en prenant leur moyenne. Mais on peut vérifier que :

$$digperim(X^c) - digperim(X) = 6n$$ (n est le nombre de connexité)

On a alors :

$$(digperim(X^c) + binperim(X))/2 = digperim(X) + 3n$$

Cette moyenne est donc biaisée puisqu'elle dépend du nombre de composantes connexes de l'image. De plus, elle est inexacte pour les objets coupant le bord du champ.

Exercice n° 2 : hypothèses transitive et stationnaire

L'observation de l'image GRAINS1 suggère que l'ensemble étudié est entièrement connu et inclus dans le champ de mesures. Dans ce cas, il est parfaitement légitime de parler de l'aire de l'ensemble, de son nombre de connexité. De la même façon, il est permis de définir l'aire de l'érodé, ou du dilaté pourvu que ce dilaté soit encore entièrement contenu dans le champ de mesures. Lorsque l'on peut faire l'hypothèse de la connaissance exhaustive de l'objet à analyser, on dit que l'on travaille en mode transitif.

Inversement, sur l'image GRAINS2, une telle hypothèse est difficile à faire. A l'évidence, ce qui apparaît dans le champ n'est qu'une partie d'un ensemble plus étendu. Dans ce cas, les seules mesures ayant un sens, sont les mesures rapportées à l'unité de surface : teneur, nombre spécifique de connexité, etc. . On dit alors que l'on travaille en mode stationnaire. Les mesures sont effectuées en construisant des estimateurs sans biais. Ainsi, la teneur d'un ensemble X est estimée par :

$$t = \frac{mes(X \cap D)}{mes(D)}$$

D est le champ de mesure, et $X \cap D$ correspond à la partie connue de l'ensemble X.

1) Comment calculer un estimateur sans biais de la teneur de l'érodé de X? (L'érodé de X est faux dans le champ D. La question revient donc à trouver un champ D' dans lequel l'érodé est exact et donc à utiliser ce champ pour calculer un estimateur de la teneur).

2) Même question pour le dilaté.

3) De la même façon, donnez un estimateur de la teneur de l'ouvert et du fermé. (La figure ci-dessous vous fournit un élément de réponse).

[clé : edge 01, attention! il reste à diviser par la surface du champ érodé]

Solution

1) La construction d'un estimateur sans biais de la teneur d'un érodé $X \ominus B$, sachant que X est connu uniquement dans un champ D, consiste donc à chercher un champ D' dans lequel l'érodé n'est pas biaisé.

On doit donc avoir :

$$[(X \cap D) \ominus B] \cap D' = (X \ominus B) \cap D'$$

soit : $(X \ominus B) \cap (D \ominus B) \cap D' = (X \ominus B) \cap D'$

Pour que cette égalité soit vérifié, quelque soit X, on doit avoir :

$$(D \ominus B) \cap D' = D'$$

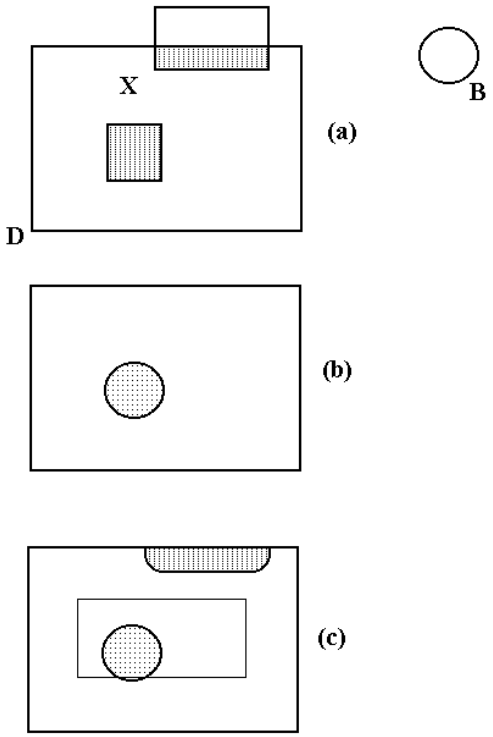
Supposons que $D' \supset D \ominus B$. Alors :

$$(D \ominus B) \cap D' = D \ominus B = D'$$

Le plus grand masque D' dans lequel l'érodé n'est pas biaisé est donc le masque D érodé par B.

Un estimateur sans biais de la teneur de l'érodé sera donc :

$$t_{X \ominus B} = \frac{\text{mes}[(X \cap D) \ominus B]}{\text{mes}(D \ominus B)}$$



(a) Image originale, (b) Ouvert obtenu, (c) Ouvert vrai

2) Si la teneur du dilaté $X \oplus B$ n'est pas biaisée, la teneur de l'érodé $X^c \ominus B$ ne l'est pas, et réciproquement. Cela signifie que le masque D' dans lequel la teneur du dilaté est calculée est encore une fois le masque $D \ominus B$.

Vérifions-le a posteriori. Le masque D' doit vérifier :

$$[(X \cap D) \oplus B] \cap D' = (X \oplus B) \cap D'$$

Montrons l'égalité quand $D' = D \ominus B$. Le deuxième membre peut s'écrire :

$$(X \oplus B) \cap D' = (X \oplus B) \cap (D \ominus B)$$

$$= ([(X \cap D) \cup (X \cap D^c)] \oplus B) \cap (D \ominus B)$$

$$= [[(X \cap D) \oplus B] \cup [(X \cap D^c) \oplus B]] \cap (D \ominus B)$$
$$= [[(X \cap D) \oplus B] \cap (D \ominus B)] \cup \underbrace{[[(X \cap D^c) \oplus B] \cap (D \ominus B)]}_{(b)}$$

Le terme (b) est vide. En effet :

$$(X \cap D^c) \oplus B \subset D^c \oplus B = (D \ominus B)^c$$

Donc, $(X \cap D)^c \oplus B$ est inclus dans le complémentaire de $(D \ominus B)$. L'intersection est vide. Q.E.D.

On peut écrire :

$$t_{X \oplus B} = \frac{mes[[(X \cap D) \oplus B] \cap (D \ominus B)]}{mes(D \ominus B)}$$

L'ouvert $\gamma(X)$ est la succession d'une érosion par B, suivie d'une dilatation par B. L'ouvert est donc non biaisé dans le masque :

$$D \ominus \check{B} \oplus B = D \ominus (\check{B} \oplus B)$$
$$t_{\gamma(X)} = \frac{mes[\gamma(X \cap D) \cap (D \ominus \check{B} \ominus B)]}{mes(D \ominus \check{B} \ominus B)}$$

Exercice n° 3

On appelle covariance de pas l, C(l), la mesure de l'aire (cas transitif) ou la teneur (cas stationnaire) de l'érodé de X par un doublet de points distants de l. On donne d'ailleurs souvent le nom de covariance à la transformation elle-même.

Dans le cas stationnaire, la covariance est un estimateur de la probabilité d'inclure le doublet de points dans l'ensemble X supposé s'étendre dans tout l'espace.

1) Programmez la covariance. Utilisez pour cela les suggestions fournies plus loin. (Notamment en ce qui concerne le tracé de la courbe).

2) Interprétez les traits caractéristiques de la courbe C(l), et en particulier C(0), C(∞), la tangente à l'origine, l'allure générale de la courbe.

3) Application aux images PARTIC1, PARTIC2 et EUTECTIC.

Solution

1) Les mesures de covariance sont stockées dans un fichier texte.

```
deproc bincov bincov dir stat spc s1 s2  sz fname
syntax "bincov dir mode (1: intrinsic; 0 : transitive) spacing binin1 binin2 size_max
        file_name"
  int i j p1 p2 v w z c;
  w := imalloc 1
  v := imalloc 1
  z := imalloc 1
 output fname
  i := 0 j := 0
  imset impixmax z z
  imcopy s1 w
  p1 := ((1000 * (imvolume w)) / imvolume z)
  p2 := ((1000 * (imvolume s2)) / imvolume z)
  while (i < sz) do
     imcopy s2 v
     iminf w v v
```

```
    imdisplay v "bincov : v"
    c := imvolume v
    if (stat <> 0) then
      c := ((1000 * (imvolume v)) / imvolume z)
    end
    print [  i " " c ]
    for 1 to spc do
      imcopyngb w w dir 1 0
      iminfngb z z dir 1
    end
    i := (i + spc)
  end
 output ""
  if (stat <> 0) then
    print [ " asymptot "((p1 * p2) / 1000) ]
  end
  imfree v
  imfree w
  imfree z
end
```

2) Comme le montre la définition de la covariance, C(0) est égale à la teneur de l'ensemble X.

C(∞) est la probabilité d'inclusion d'un couple de points à une distance infinie l'un de l'autre. Cette probabilité est égale à la probabilité d'inclusion de deux points indépendants l'un de l'autre (événements non corrélés). Soit :

$$C(\infty) = C^2(0)$$

La tangente à l'origine de la courbe est égale en valeur absolue à -C'(0). Un estimateur de cette valeur est donné par :

$$-C'(0) = C(0) - C(1)$$

$$-C'(0) = \frac{mes(X \cap D') - mes[(X \ominus \check{K}_1) \cap D']}{mes(D')}$$

avec $D' = D \ominus K_1$

Le numérateur peut s'écrire :

$$mes[(X \cap D')/[(X \ominus \check{K}_1) \cap D']] = mes[X \cap (X \ominus \check{K}_1)^c \cap D']$$

soit :

$$mes(X \cap X_{-1}^c \cap D')$$

Or $X \cap X_{-1}^c$ est l'érodé de X par l'élément structurant 01, permettant le calcul du nombre d'intercepts horizontaux.

-C'(0) est donc un estimateur sans biais de la variation diamétrale verticale de X.

3) Tracé des courbes

On pourra remarquer comment le caractère périodique de l'image EUTECTIC se répercute sur la courbe de covariance.

Exercice n° 4

On a déjà défini la granulométrie par ouverture (voir exercice 6, chapitre 2). On désigne par F(λ) la teneur de l'ouvert de taille λ d'un ensemble X.

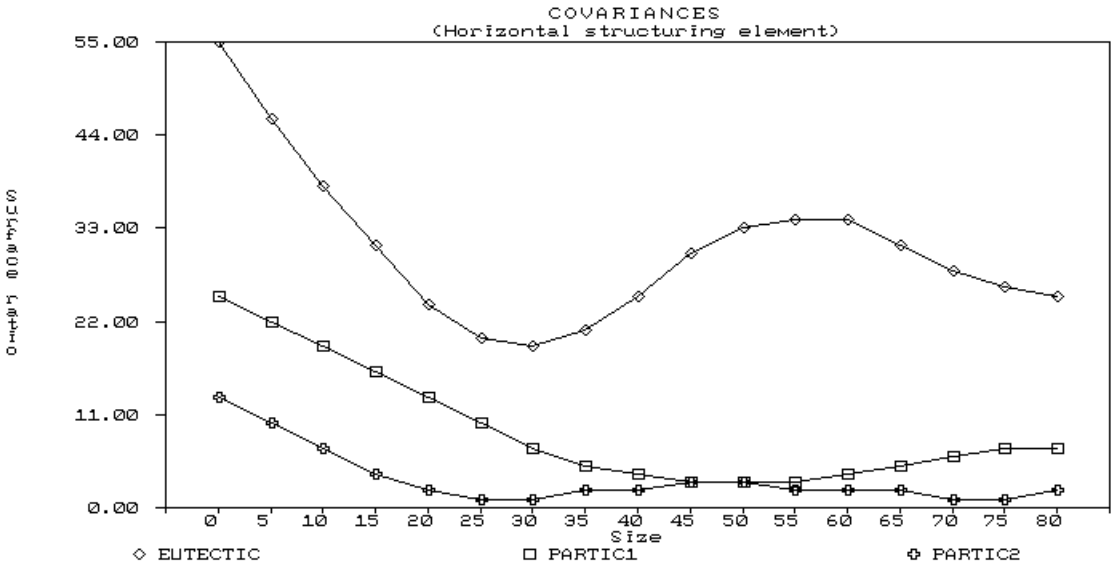
1) Vérifiez que :

$$G(\lambda) = \frac{F(0) - F(\lambda)}{F(0)}$$

est une grandeur toujours comprise entre 0 et 1, et calculez-la en fonction de l'aire de l'ouvert et de l'aire du champ de mesure érodé D' ($D' = D \ominus 2\lambda H$).

2) Programmez cette mesure. Application à la granulométrie des images METAL1 et METAL2.

[procédures **granul** ; **isogranul**]



Tracés de courbes de covariance

Solution

1) Vérifions que G(λ) est compris entre 0 et 1.

On a :

$$G(\lambda) = 1 - \frac{F(\lambda)}{F(0)}$$

G(λ) est monotone, car l'ouverture est anti-extensive, avec G(0) = 0. Quand $\lambda \to \infty$,$F(\lambda) \to 0$, $G(\lambda) \to 1$.

Un estimateur de G(λ) est donné par :

$$G(\lambda) = \frac{mes(X \cap D') - mes(X_{\lambda B} \cap D')}{mes(X \cap D')}$$

soit encore :

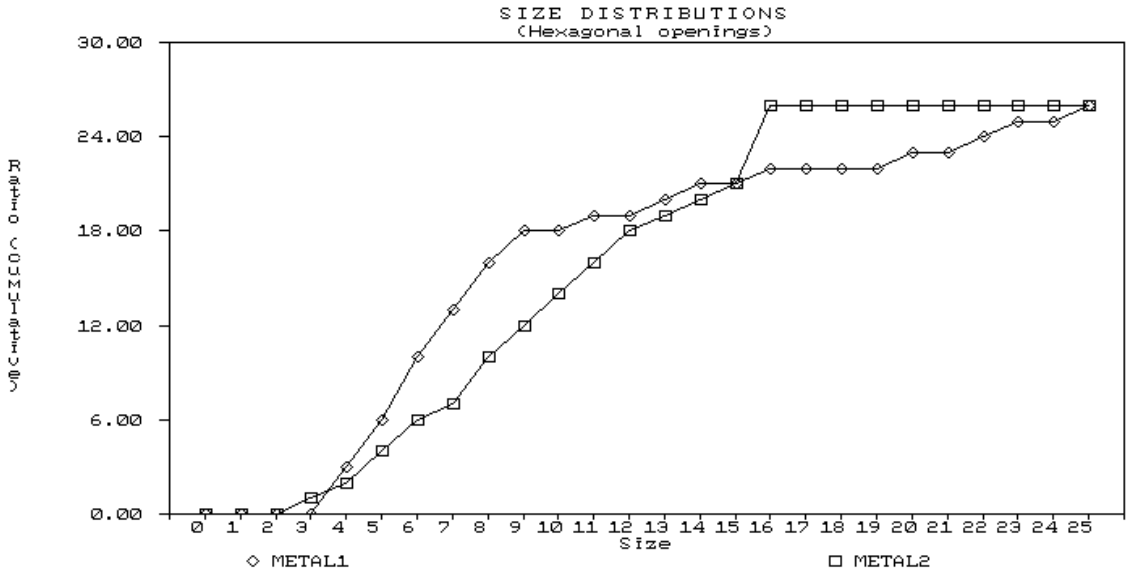
$$G(\lambda) = \frac{mes[X/X_{\lambda B} \cap D']}{mes(X \cap D')} \quad (D' = D \ominus 2\lambda B)$$

2) Calcul de G(n)

La procédure de calcul de G(n) est la suivante (cas binaire) :

defunc bingranul bingranul s spc sz fname
syntax "bingranul: binin spacing size file_name"

```
 int z w1 w2 i a ;
 w1 := imalloc 1
 w2 := imalloc 1
 z := imalloc 1
 i := 0
 imcopy s w1
imset impixmax z z
imdisplay w2 cat [ "bingranul : " w2 ]
output fname
while (i < sz) do
   ero w1 w1 spc
   dil w1 w2 (i + spc)
   if (edge = 0)  then
      ero z z (2 * spc)
      iminf z w2 w2
     a := imvolume z
  else
    a := imvolume s
  end
  print[  (i + spc) "   " (1000 - (imvolume w2 * 1000 / a))]
  i := (i + spc)
  end
bingranul := (1000 - (imvolume w2 * 1000 / a))
output ""
imfree w1
imfree w2
imfree z
end
```



Courbes granulométriques cumulées

RESUME

A l'issue de ces exercices, votre dictionnaire doit s'être enrichi des transformations suivantes :

diameter dir s
nombre d'intercepts dans une direction dir de l'image binaire s.
cnumber s tr
nombre de connexité de l'image binaire s; si tr=1, représentation transitive, si tr=0, représentation intrinsèque.
digperim s d
périmètre de l'image binaire s (par contour intérieur) et image contour dans d.
perim s
périmètre de l'image binaire s (par formule de Cauchy).
bincov dir st sp s1 s2 sz
covariance croisée de taille maximale sz, entre les images binaires s1 et s2 dans la direction dir, selon le pas sp; st=1 correspond au mode intrinsèque et st=0 au mode transitif.
granul s sp sz
granulométrie de taille maximale sz de l'image binaire s selon le pas sp.