ALGORITHMES SEQUENTIELS

Fernand MEYER

Centre de Morphologie Mathématique - Ecole Nationale Supérieure des Mines de Paris
35 Rue Saint-Honoré 77305 FONTAINEBLEAU Cedex (France)

RESUME

Jusqu'ici, la plupart des travaux en analyse d'images ont porté sur les algorithmes parallèles. Dans cet article nous présentons les principaux algorithmes séquentiels issus de la Morphologie Mathématique :

- fonction distance, hexagonale, dodécagonale ou géodésique
- reconstruction de grains pour les images à niveaux de gris
- squelette d'un ensemble binaire en 3 passes
- squelette d'images à niveaux de gris.

I - INTRODUCTION

L'analyse d'image telle qu'elle est pratiquée quotidiennement, celle qui est câblée dans les appareils dédiés utilise quasi exclusivement des algorithmes parallèles. Pour augmenter la vitesse de traitement on a cherché surtout à rendre plus complexes les architectures spécialisées.

Pourtant ROSENFELD montre en 1966 que pour toute transformation parallèle il existe une transformation séquentielle qui lui est équivalente (Rosenfeld 1966). Rappelons la caractérisation d'une transformation séquentielle : dans le mode d'exploration de l'image, on distingue un passé, ensemble des points déjà traités, et un futur, ensemble des points à traiter. Une transformation séquentielle T se sert d'un élément structurant dans lequel les points du passé a_p possèdent leur nouvelle valeur $T(a_p)$, alors que les points du futur a_f, non encore transformés, conservent leur ancienne valeur.

Pour l'élément structurant hexagonal (fig. 1), lors d'un balayage vidéo classique $T(a_1)$ est une fonction de $(T(a_4), T(a_3), T(a_5), a_1, a_2, a_6, a_7)$. Pour un algorithme parallèle les arguments seraient $(a_1, a_2, a_3, a_4, a_5, a_6, a_7)$. Ainsi, dans cet algorithme parallèle un point n'a d'influence que sur ses voisins immédiats à chaque étape du traitement. Dans un algorithme séquentiel au contraire, un point x peut avoir une influence sur tous les points qui se trouvent dans le futur de x par rapport au sens de balayage. Ceci explique l'efficacité plus grande des algorithmes séquentiels.

Malheureusement, si Rosenfeld a montré l'équivalence entre algorithmes parallèles et séquentiels, sa démonstration ne permet pas de construire la version séquentielle d'un algorithme parallèle donné. Nous allons dans cet article voir comment :

- paralléliser la fonction distance de Rosenfeld
- construire une fonction distance dodécagonale
- effectuer une reconstruction de graines numériques
- obtenir le squelette d'un ensemble binaire ou une fonction numérique.

Notation : S'agissant de transformations séquentielles, nous signalerons par $f(a_i)$ que le point a_i appartenant au futur, possède encore son ancienne valeur. Pour $f'(a_i)$ nous indiquerons que le point a_i appartient au passé et possède donc déjà sa nouvelle valeur.

Voisinage : Tout au long de l'article nous utiliserons le voisinage du point central de la fig. 1 :

SUMMARY

Rosenfeld showed in 1966 the equivalence between parallel sequential algorithms. However most of the work in the field of image analysis has been devoted to the parallel algorithms. This paper presents sequential algorithms for the main transformations of Mathematical Morphology :

- distance transformation (hexagonal, dodecagonal, geodesic)
- grain reconstruction for grey tone images
- binary skeleton in 3 steps.
- grey tone image skeletons.

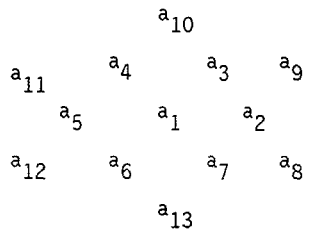


Figure 1

II - LA FONCTION ERODE EUCLIDIENNE, A DISTANCE HEXAGONALE

A/ Objet : Il s'agit d'obtenir en un nombre fini de passes, la fonction ERODE d'un ensemble binaire X, caractérisée par :

$$ERODE(x) = d_H(x, X^c)$$ où d_H est la distance hexagonale.

Remarque : Il existe plusieurs variantes de cet algorithme. Nous allons en expliciter deux. La première est celle de Rosenfeld, adaptée par B. Laÿ (LAY,1985) à la trame hexagonale. La deuxième montre comment paralléliser un algorithme séquentiel, en mettant un processus par ligne.

B/ Version 1 : Cette version a été d'abord décrite par A. Rosenfeld pour la trame carrée, adaptée par B. Laÿ à la trame hexagonale.

- Etape 1 : Balayage video direct

Si $f(a_1) = 0$ $\quad f'(a_1) = 0$

sinon $\quad f'(a_1) = 1 + \text{Min}[f'(a_5), f'(a_4), f'(a_3)]$

- Etape 2 : Balayage video inverse

$$f'(a_1) = \text{Min}[f(a_1),\ 1 + \text{Min}[f'(a_2), f'(a_7), f'(a_6)]$$

C/ Version 2 : Cette version a les avantages suivants :

- elle utilise des éléments structurants linéaires, ce qui permet d'imaginer facilement un dispositif avec une parallèlisation massive d'opérateurs, travaillant indépendamment les uns des autres, chacun sur une ligne ou une diagonale de l'image.
- ce même dispositif, ayant accès à l'image ligne par ligne permet à peu de frais d'engendrer tous les polygones de Steiner qu'on désire : dodécagones, poly-

gones à 18 ou 24 côtés. Il suffit que le générateur d'adressessoit capable d'extraire des lignes de l'image dans les directions correspondantes, 30°, 90°, etc... (resp. 15°, 75°, etc...).

Description

a) Ligne horizontale

Si $f(a_1) = 0 \quad f'(a_1) = 0$

sinon $f'(a_1) = 1 + f'(a_5)$

b) Ligne horizontale

$f'(a_1) = \text{Min}[f(a_1),\ 1 + f'(a_2)]$

c) Ligne à 60°

$f'(a_1) = \text{Min}[f(a_1),\ 1 + f'(a_3)]$

d) Ligne à 60°

$f'(a_1) = \text{Min}[f(a_1),\ 1 + f'(a_6)]$

e) Ligne à 120°

$f'(a_1) = \text{Min}[f(a_1),\ 1 + f'(a_4)]$

f) Ligne à 120°

$f'(a_1) = \text{Min}[f(a_1),\ 1 + f'(a_7)]$

III - LA FONCTION ERODE EUCLIDIENNE A DISTANCE DODECAGONALE

A/ Objet : Il s'agit d'obtenir en un nombre fini de passes la fonction ERODE d'un ensemble binaire X, caractérisé par :

ERODE(x) = $d_D(x, X^c)$ où d_H est la distance dodécagonale. La distance dodécagonale $d_D(x,y)$ se détermine de la manière suivante. On construit le pinceau d'angle d'ouverture égale à 30°, centréen x et contenant le point y et allant de $k\Pi$ à $(k+1)\frac{\Pi}{6}$

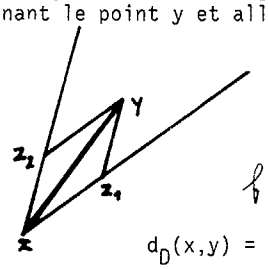


figure 1 bis

$$d_D(x,y) = d(x,z_1) + d(x,z_2)$$

Ainsi, plus les pinceaux d'angle utilisés sont étroits, plus on se rapproche de la distance euclidienne. A cet égard, la distance dodécagonale est bien meilleure que la distance hexagonale. (On pourra consulter utilement(Borgefors 1984) pour la description de fonctions distance en trame cubique dans R^n).

Un dernier problème reste à résoudre. Sur la trame, la distance entre deux points voisins sur une ligne à 30° est de $\sqrt{3}$. Il faut donc trouver une approximation correcte de $\sqrt{3}$ au moyen d'un rationnel. La meilleure approximation avec des termes petits est 7/4. En effet $(7/4)^2 = 49/16 \simeq 3 = 48/16$.

Ainsi on aura $d(a_1,a_2) = 4$

$d(a_1,a_9) = 7$

Ce préambule étant posé, les diverses versions de l'algorithme se décalquent aisément sur les versions présentées pour la distance hexagonale. Nous ne donnerons que la version globale, et non pas celle ligne par ligne.

Etape 1 : Balayage video direct

Si $f(a_1) = 0 \quad f'(a_1) = 0$

sinon

$$f'(a_1)=\text{Min}\{4+\text{Min}[f'(a_5),f'(a_4),f'(a_3)]$$
$$7+\text{min}[f'(a_9),f'(a_{10}),f'(a_{11})]\}$$

Etape 2 : Balayage video inverse

$$f'(a_1)=\text{Min}\{f(a_1),4+\text{Min}[f'(a_6),f'(a_7),f'(a_2)]$$
$$7+\text{Min}[f'(a_8),f'(a_{13}),f'(a_{12})]\}$$

IV - LA FONCTION ERODE EUCLIDIENNE, SELON UNE DISTANCE A 24 COTES

Dans ce cas, seules les versions ligne par ligne seront praticables et les espacements entre points à retenir sont les suivants :

Les distances suivantes offrent une bonne approximation des distances euclidiennes :

$d^*(x,y) = 12$

$d^*(x,z) = 21$

$d^*(x,t) = 32$

V - LA FONCTION ERODE D'UN ENSEMBLE BINAIRE X CONDITIONNELLEMENT A UN ENSEMBLE BINAIRE Y

A/ Objet : Il s'agit de construire rapidement, mais cette fois plus en un nombre fixe de passes la fonction ERODE(X) à l'aide de la distance d_{H,Y^c}

$d_{H,Y^c}(x,y)$ est la longueur du plus court chemin entre x et y représentable sur la trame, entièrement inclus dans Y^c.

Dans une autre terminologie, on peut dire qu'on cherche la fonction ERODE conditionnellement à l'ensemble Y.

Les érodés successifs sont ainsi :

$X_o = X \cup Y$

$X_1 = (X \ominus H) \cup Y$

$X_n = (X_{n-1} \ominus H) \cup Y$

B/ Description de l'algorithme 1

1) Initialisation et codage

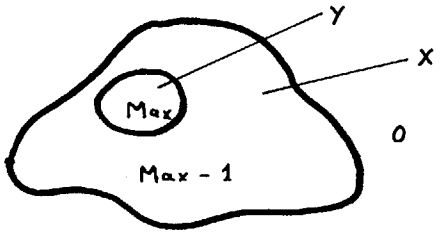


figure 2

Figure 2 : Codages de X et Y

On procède au codage suivant(cf. fig. 2)

$$\begin{cases} x \in Y & f(x) = \text{Max} \\ x \in X \cap Y^c & f(x) = \text{Max} - 1 \\ x \in X^c \cap Y^c & f(x) = 0 \end{cases}$$

2) Il faut répéter jusqu'à stabilité les deux étapes qui suivent, l'une en balayage direct, l'autre en balayage inverse.

a) Balayage video direct

Si $f(a_1) = Max$, ne rien faire

Sinon

$k = Min(f'a_3), f'(a_4), f'(a_5))$

Si $k \geq Max-1$, $f'(a_1) = f(a_1)$

Si $k < Max-1$:

$f'(a_1) = Min(f(a_1), 1+k)$

b) Balayage video inverse

Traitement obtenu à partir du précédent par symétrie autour du centre de l'élément structurant.

VI - RECONSTRUCTION DE GRAINS AVEC REECRITURE

A/ Objet

Il s'agit d'effectuer avec le moins de passes possibles une reconstruction de grains numériques d'une image numérique FX à partir d'une image numérique FY.

En d'autres termes, il faut obtenir en moins d'opérations le même résultat que l'algorithme classique :

Répéter jusqu'à stabilité :

$$f(Y) = Min[F(Y) + H, F(X)]$$

B/ Description de l'algorithme

Il faut répéter jusqu'à stabilité les deux étapes qui suivent, l'une en balayage direct, l'autre en balayage inverse.

a) Balayage vedeo direct

$$FY'(a_1) = Min[FX(a_1), Max(FY'(a_3), FY(a_1), FY'(a_4), FY'(a_5)]$$

b) Balayage video inverse

$$FY'(a_1) = Min[FX(a_1), Max(FY'(a_6), FY(a_1), FY'(a_7), FY'(a_2)]$$

VII - LE SQUELETTE EUCLIDIEN BINAIRE

A/ Objet

Il s'agit d'obtenir en 3 passes le squelette euclidien d'une image binaire en partant soit :

- de la fonction ERODE obtenue comme indiqué plus haut
- de la fonction MOD dérivée de la fonction ERODE :

  si ERODE(x)=0 MOD(x)=0
  si ERODE(x)+0 MOD(x)=1+(ERODE(x) modulo 3)

Ainsi la fonction MOD prend 4 valeurs et peut être codée sur 2 bits.

On trouvera un exposé plus complet sur le squelette dans (MEYER, 1985,1987).

Les versions en trames carrées ont été développées indépendamment par ARCELLI dès 1984 (ARCELLI,1985).

B/ La détection des points crête

L'examen du relief de la fonction distance montre l'existence de points crête, qui clairement appartiennent au squelette. La plupart des points cependant ne connaissent pas ces ruptures de pente. Nous avons formalisé cela et obtenu une caractérisation des points crête

$$\text{Points crête} = \left\{ \begin{array}{ccc} n-1 & n-1 & \cdot \quad \overline{n+1} \quad \cdot \quad (n+1) \\ n-1 \ (n) & n-1, & \cdot \ (n) \ (n) \ \cdot \ (n) \ (n) \\ n-1 & n-1 & \cdot \quad \overline{n+1} \quad \cdot \quad (n+1) \end{array} \right\} \quad (1)$$

où $\overline{n+1}$ signifie que le pixel correspondant vaut n ou n-1 mais pas n+1 et où (n) signifie que le point correspondant est point crête. Les configurations (1) sont données à une rotation de $\Pi/3$ près.

Tous les autres points sont des points non crête et sont du type :

$$\text{Points non crête} = \left\{ \begin{array}{ccc} n+1 & n+1 & \quad n+1 \quad n+1 \quad \quad n \quad n+1 \\ n & n & n+1, \quad n \quad n \quad n, \quad n-1 \quad n \quad n \\ n-1 & n & \quad n-1 \quad n-1 \quad \quad n-1 \quad n-1 \end{array} \right\} \quad (2)$$

Ces points sont liés de très près à l'homotopie. En effet, retirer d'un ensemble X tous ses points non crête d'altitude 1 constitue un amincissement homotopique de X.

C/ La remontée vers l'amont

Le squelette final s'obtient en ajoutant aux points crête les points qui se trouvent dans leur amont. Prendre comme points amont d'un point x tous les points d'altitude supérieure à x conduirait à un amont qui s'élargirait en entonnoir. On n'obtiendrait plus un squelette fin.

On effectuera une remontée vers l'amont plus sélective en appliquant les règles suivantes (à rotation de $k\Pi/3$ près)

(3a) $\begin{array}{cc} n & n+1 \\ (n) & n \end{array} \rightarrow \begin{array}{cc} n & (n+1) \\ (n) & n \end{array}$

(3b) $\begin{array}{cc} n+1 & n+2 \\ (n) & n+1 \end{array} \rightarrow \begin{array}{cc} n+1 & (n+2) \\ (n) & n+1 \end{array}$

(3c) $\begin{array}{ccc} \cdot & \overline{n+2} & \\ (n) & n+1 & \\ \cdot & \overline{n+2} & \end{array} \rightarrow \begin{array}{ccc} \cdot & \overline{n+2} & \\ (n) & (n+1) & \\ \cdot & \overline{n+2} & \end{array}$

Bien que les règles (3a) et (3b) de remontée vers l'amont engendrent un point fils non contigu au père, le squelette obtenu en partant des points crête est connexe et a la même homotopie que l'ensemble initial.

D/ Un algorithme séquentiel de remontée vers l'amont

La fonction distance ne peut produire n'importe quel relief (fonction lipschitzienne). Ces règles de remontée vers l'amont ne peuvent générer que des morceaux de segment de droite. En effet si y est l'amont d'un point x, un seul des cas de figure suivants est vrai :

- y n'a pas d'amont et est un point crête
- l'amont de y reste dans la direction xy si la règle (3c) a été employée
- l'amont de y est dans la direction du vecteur xy ou dans une des deux directions adjacentes $d\pm\Pi/6$, lorsque les règles (3a) ou (3b) ont été employées.

Une remontée séquentielle vers l'amont en 3 passes est donc possible

a) 1ère passe en balayage video direct
Elle concerne les directions
$-\Pi/6$, $-\Pi/3$, $-\Pi/2$, $-2\Pi/3$, $-5\Pi/6$.

b) 2ème passe en balayage video inverse
Elle concerne les directions
$\Pi/6$, $\Pi/3$, $\Pi/2$, $2\Pi/3$, $5\Pi/6$, Π

c) 3ème passe de gauche à droite
Elle concerne la direction 0 et peut se combiner avec la passe précédente. Aussitôt que le balayage de droite à gauche de la passe précédente s'est achevé, on peut enchaîner avec un balayage de gauche à droite en utilisant la règle (3c).

Squelette pour images à niveaux de gris

Les images à niveaux de gris peuvent être amincies de manière itérative comme les images binaires (Goettcharian). Il faut alors un nombre de passes proportionnel au rayon du plus grand des objets de la figure.

Nous proposons ici un algorithme séquentiel permettant d'obtenir un squelette numérique. L'algorithme se fait en 3 étapes comme dans le cas binaire.

1/ Création d'une fonction distance sur les plateaux

Dans une image numérique f(x) il peut exister des plateaux. Un algorithme séquentiel nous permet de générer une fonction distance g(x) sur chaque plateau : g(x) est égal à la distance de x au point le plus proche y d'altitude inférieure (f(y) < f(x))

ⓐ - Initialisation : $\forall x \in$ Image g(x) = max

ⓑ - Répéter jusqu'à stabilité les étapes suivantes

- balayage vidéo direct (cf. le voisinage de la fig. 1)
  * Si $\exists$ i $\in$ (3,4,5) tel que f(i) < f(1) alors g(x) = 1.
  * Pour tout i {3,4,5} appartenant au même plateau que 1 faire g'(1)=Min(g'(1),1+g'(i))
- balayage video inverse : symétrique du précédent.

ⓒ - Remarque : les points tels que g(x)=Max sont les minima régionaux de f(x).

2/ Construction des points crête

Les images f et g nous permettent de construire une relation d'ordre entre points voisins.
x>y ssi f(x)>f(y) ou (f(x)=f(y) et g(x)>g(y))

Munis de cette relation d'ordre on détecte tous les points crête. Ce sont tous les points qui ne vérifient pas une des configurations suivantes :

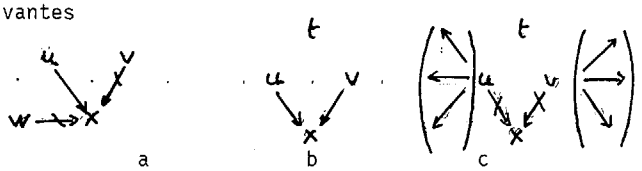
```
        > >           > >           > .
(a)  .  x  >  (b)  .  x  .  (c)  .  x  <
        < .           < <           < <
```

où > signifie un pixel supérieur à x
< signifie un pixel inférieur à x
. signifie l'absence de condition.

3/ Remontée vers l'amont

a - Détermination des pentes maximales

Pour chaque point non crête on peut déterminer le ou les directions descendantes maximales (d.d.m.). Dans le cas où il existe deux d.d.m., toutes les directions descendantes comprises entre ces deux directions sont aussi considérées comme des d.d.m.
Un point non crête sera l'amont d'un point x du squelette si x se trouve dans une des situations suivantes



où → représente une d.d.m.

⇸ représente l'absence d'une telle direction


représente l'existence d'une d.d.m. dans une des directions indiquées.

Dans les cas b et c :

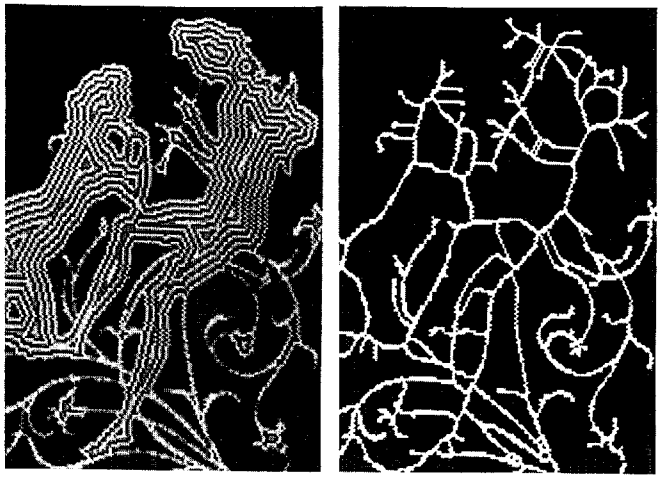
si u > v > x u est pris

si v > u > x v est pris

Si ((u=v) > x) et(t ≦ u) alors u et v sont pris

Si ((u = v) ≧ x) et(t > u)alors t est pris

On remonte vers l'amont dans le sens du balayage direct puis inverse. Contrairement au squelette binaire, le processus ne converge plus en un nombre fini de passes et il faut itérer jusqu'à convergence.



a) Fonction relief b) Squelette binaire

a)Image initiale b)Gradient c)Squelette numérique

Bibliographie

1 - ARCELLI C., SANNITI DI BAJA G. : A Width-Independent Fast Thinning Algorithm, PAMI-7, N° 4, 1985, pp. 463-474

2 - BORGEFORS G. : Distance Transformations in Arbitrary Dimensions, Computer Graphics and Image Processing 27, 1984, 321-345

3 - GOETCHARIAN V. : Parallel image processes and real-time texture analysis, PHD University College, London, 1980

4 - LAY B. : Morpholog, An Image Processing Software Package, IEEE Workshop on Computer Architecture for Pattern Analysis and Image Database Management, 1985, pp. 463-469

5 - MEYER F. : The Binary Skeleton in three Steps, IEEE Workshop on Computer Architecture for Pattern Analysis and Image Database Management, 1985, pp. 470-476

6 - MEYER F. : The euclidian skeleton on a hexagonal raster, Advances in Mathematical Morphology, J. Serra Ed., Academic Press, 1987

7 - ROSENFELD A., PFALTZ J.L. : Sequential Operations in Digital Picture Precessing, J. ACM 13, 1966, pp. 471-494.