TP 5. Opérateurs sur images de contours

I. OPERATEURS BAS-NIVEAU

1. Affinage des contours *ALGORITHME DE CHIN & WAN*

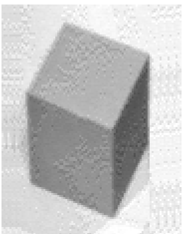
Q5.1. Transposer (Python) l'algorithme d'amincissement des contours de Chin & Wan. Interprétation des résultats

I := READ_IMAGE("cube.jpg") Nombre de colonnes : Nx := cols(I) Nx = 90 Nombre de lignes : Ny := rows(I) Ny = 116

Contours : J := Moyenne(I) K := NormeGradient4Diff(J) L := Binarisation(K, 13)

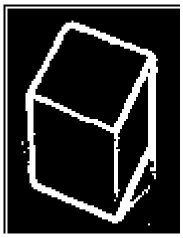
Affinage des contours : M := ChinWan(L) N := ChinWan(M)

Image brute



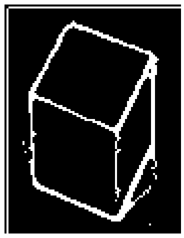
I

Image des contours



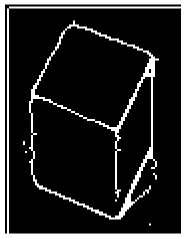
L

Contours affinés (1 fois)



M

Contours affinés (2 fois)



N

II. PRIMITIVES DE CLASSIFICATION

Q5.2. Transposer (Python) l'algorithme de coloriage (blob colouring) itératif (Blobi). Test sur image Ctest.bmp

1. Coloriage

ETIQUETAGE DES REGIONS

I := READ_IMAGE("Btest.bmp")

contours non unitaires (épais)

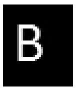
Nx := cols(I) Nx = 35 Ny := rows(I) Ny = 41

Etiquetage B := Binarisation(I, 100)

R := Blobi(B) *Blob itératif* R := READBMP("regn.bmp")

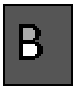
Visualisation des étiquettes K := Consecution(R) L := RecadrageDyn(K) L := READBMP("reg.bmp")

Image des contours fermés



I

Image des étiquettes



L

Nombre de Régions NbR := NbReg(R) NbR = 3

NbR := NbReg(K) NbR = 3

K := READBMP("regn2.bmp")

R =

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1
11	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1
12	0	1	1	1	1	1	1	1	0	0	59	59	59	59	59	0	0	0	1	1	1
13	0	1	1	1	1	1	1	1	0	0	59	59	59	59	59	59	0	0	0	1	1
14	0	1	1	1	1	1	1	1	0	0	59	59	59	59	59	59	0	0	0	1	1
15	0	1	1	1	1	1	1	1	0	0	59	59	59	59	59	59	0	0	0	1	1
16	0	1	1	1	1	1	1	1	0	0	59	59	59	59	59	59	0	0	1	1	1
17	0	1	1	1	1	1	1	1	0	0	59	59	59	59	59	0	0	0	1	1	1
18	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1
19	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1
20	0	1	1	1	1	1	1	1	0	0	109	109	109	109	109	109	0	0	0	1	1
21	0	1	1	1	1	1	1	1	0	0	109	109	109	109	109	109	109	0	0	0	1
22	0	1	1	1	1	1	1	1	0	0	109	109	109	109	109	109	109	0	0	0	1
23	0	1	1	1	1	1	1	1	0	0	109	109	109	109	109	109	109	0	0	0	1
24	0	1	1	1	1	1	1	1	0	0	109	109	109	109	109	109	109	0	0	0	1
25	0	1	1	1	1	1	1	1	0	0	109	109	109	109	109	109	0	0	0	1	1
26	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1
27	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1
28	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
29	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

K =

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
11	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
12	0	1	1	1	1	1	1	1	0	0	2	2	2	2	2	0	0	0	1	1	1	1	1	1	1	1	1
13	0	1	1	1	1	1	1	1	0	0	2	2	2	2	2	2	0	0	0	1	1	1	1	1	1	1	1
14	0	1	1	1	1	1	1	1	0	0	2	2	2	2	2	2	0	0	0	1	1	1	1	1	1	1	1
15	0	1	1	1	1	1	1	1	0	0	2	2	2	2	2	2	0	0	0	1	1	1	1	1	1	1	1
16	0	1	1	1	1	1	1	1	0	0	2	2	2	2	2	2	0	0	1	1	1	1	1	1	1	1	1
17	0	1	1	1	1	1	1	1	0	0	2	2	2	2	2	0	0	0	1	1	1	1	1	1	1	1	1
18	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
19	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
20	0	1	1	1	1	1	1	1	0	0	3	3	3	3	3	3	0	0	0	1	1	1	1	1	1	1	1
21	0	1	1	1	1	1	1	1	0	0	3	3	3	3	3	3	3	0	0	0	1	1	1	1	1	1	1
22	0	1	1	1	1	1	1	1	0	0	3	3	3	3	3	3	3	0	0	0	1	1	1	1	1	1	1
23	0	1	1	1	1	1	1	1	0	0	3	3	3	3	3	3	3	0	0	0	1	1	1	1	1	1	1
24	0	1	1	1	1	1	1	1	0	0	3	3	3	3	3	3	3	0	0	0	1	1	1	1	1	1	1
25	0	1	1	1	1	1	1	1	0	0	3	3	3	3	3	3	0	0	0	1	1	1	1	1	1	1	1
26	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
27	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
28	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
29	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

2. Transformation de Hough

Q5.3. Transposer sous Python la transformée de Hough. Interprétation des résultats

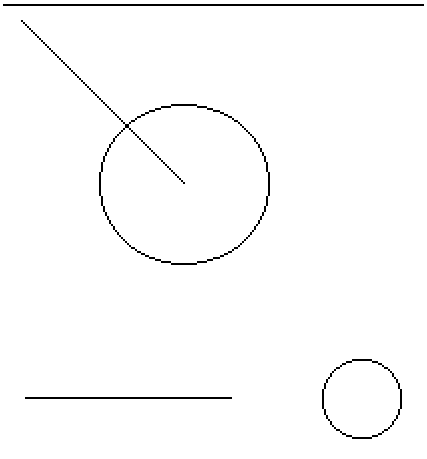
$I := \text{READ\_IMAGE}(\text{"Hough.bmp"})$ Nombre de colonnes : $Nx := \text{cols}(I)$ Nombre de lignes : $Ny := \text{rows}(I)$

$Nx = 210$ $Ny = 228$

Inversion video : $J := \text{InvVideo}(I)$ $\Delta\rho := 10$ $\Delta\theta := 10$ $TabH := \text{TabHough}(J, \Delta\rho, \Delta\theta)$

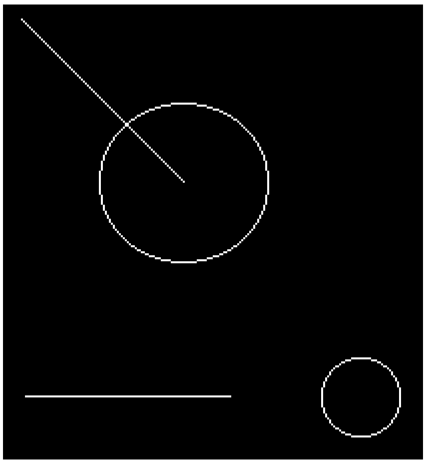
Transformation de Hough : $seuil := 100$ $K := \text{Hough}(J, \Delta\rho, \Delta\theta, seuil)$

Image des contours



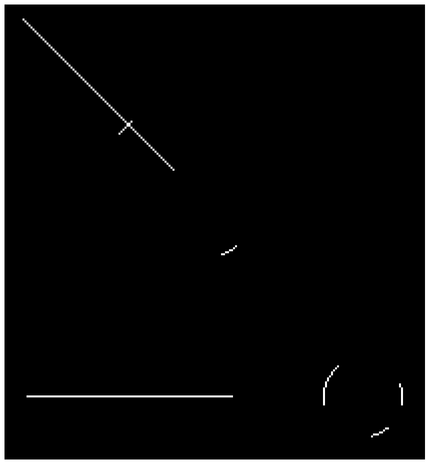
I

Image inverse video



J

Image après Transfo. de Hough



K

TabH =

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	123	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	22	73	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	20	64	29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	20	29	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	20	29	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	31	35	20	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	23	20	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	55	0	19	20	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fonctions Bibliothèque :

$$\mathrm{Histo}(I) \equiv \left| \begin{array}{l} \text{for } n \in 0..255 \\ \quad H_n \leftarrow 0 \\ \text{for } y \in 0..\mathrm{rows}(I)-1 \\ \quad \text{for } x \in 0..\mathrm{cols}(I)-1 \\ \quad\quad \left| \begin{array}{l} n \leftarrow I_{y,x} \\ H_n \leftarrow H_n + 1 \end{array} \right. \\ H \end{array} \right.$$

$$\mathrm{RecadrageDyn}(I) \equiv \left| \begin{array}{l} mini \leftarrow \min(I) \\ maxi \leftarrow \max(I) \\ \Delta \leftarrow \dfrac{255}{maxi - mini} \\ \text{for } y \in 0..\mathrm{rows}(I)-1 \\ \quad \text{for } x \in 0..\mathrm{cols}(I)-1 \\ \quad\quad J_{y,x} \leftarrow \mathrm{trunc}\left[\left(I_{y,x} - mini\right)\cdot\Delta\right] \\ J \end{array} \right.$$

$$\mathrm{RecadrageDyn\%}(I, nb, nh) \equiv \left| \begin{array}{l} \Delta \leftarrow \dfrac{255}{nh - nb} \\ \text{for } y \in 0..\mathrm{rows}(I)-1 \\ \quad \text{for } x \in 0..\mathrm{cols}(I)-1 \\ \quad\quad J_{y,x} \leftarrow \mathrm{round}\left[\left(I_{y,x} - nb\right)\cdot\Delta\right] \\ J \end{array} \right.$$

$$\mathrm{InvVideo}(I) \equiv \left| \begin{array}{l} \text{for } y \in 0..\mathrm{rows}(I)-1 \\ \quad \text{for } x \in 0..\mathrm{cols}(I)-1 \\ \quad\quad J_{y,x} \leftarrow 255 - I_{y,x} \\ J \end{array} \right.$$

$$\mathrm{Convol3x3}(I, Mask3x3) \equiv \left| \begin{array}{l} \text{for } y \in 1..\mathrm{rows}(I)-2 \\ \quad \text{for } x \in 1..\mathrm{cols}(I)-2 \\ \quad\quad \left| \begin{array}{l} som \leftarrow 0 \\ \text{for } i \in 0..2 \\ \quad \text{for } j \in 0..2 \\ \quad\quad som \leftarrow som + \mathrm{round}\left(I_{y+i-1,x+j-1}\cdot Mask3x3_{i,j}\right) \\ som \leftarrow 255 \text{ if } som > 255 \\ som \leftarrow 0 \text{ if } som < 0 \\ J_{y,x} \leftarrow som \end{array} \right. \\ J \end{array} \right.$$

$$\mathrm{Derivation}(I, Mask3x3) \equiv \left| \begin{array}{l} \text{for } y \in 1..\mathrm{rows}(I)-2 \\ \quad \text{for } x \in 1..\mathrm{cols}(I)-2 \\ \quad\quad \left| \begin{array}{l} som \leftarrow 0 \\ \text{for } i \in 0..2 \\ \quad \text{for } j \in 0..2 \\ \quad\quad som \leftarrow som + I_{y+i-1,x+j-1}\cdot Mask3x3_{i,j} \\ som \leftarrow |som| \\ som \leftarrow 255 \text{ if } som > 255 \\ som \leftarrow 0 \text{ if } som < 0 \\ J_{y,x} \leftarrow som \end{array} \right. \\ J \end{array} \right.$$

$$\mathrm{Binarisation}(I,seuil) \equiv \left| \begin{array}{l} \text{for } y \in 0..\mathrm{rows}(I)-1 \\ \quad \text{for } x \in 0..\mathrm{cols}(I)-1 \\ \quad\quad \left| \begin{array}{l} J_{y,x} \leftarrow 0 \quad \text{if } I_{y,x} < seuil \\ J_{y,x} \leftarrow 255 \quad \text{otherwise} \end{array} \right. \\ J \end{array} \right.$$

$$\mathrm{T2D}(I) \equiv \left| \begin{array}{l} \text{for } y \in 0..\mathrm{rows}(I)-1 \\ \quad \text{for } x \in 0..\mathrm{cols}(I)-1 \\ \quad\quad J_{y,x} \leftarrow I_{y,x} \\ J \end{array} \right.$$

$$\mathrm{T1DColonne}(I,f) \equiv \left| \begin{array}{l} \text{for } n \in 0..\mathrm{rows}(I)-1 \\ \quad g_n \leftarrow f_n \\ g \end{array} \right.$$

$$\mathrm{T1D}(I) \equiv \left| \begin{array}{l} \text{for } x \in 0..\mathrm{cols}(I)-1 \\ \quad \left| \begin{array}{l} f \leftarrow I^{\langle x \rangle} \\ J^{\langle x \rangle} \leftarrow \mathrm{T1DColonne}(I,f) \end{array} \right. \\ \text{return } J \end{array} \right.$$

$$\mathrm{Median}(I) \equiv \left| \begin{array}{l} \text{for } y \in 1..\mathrm{rows}(I)-2 \\ \quad \text{for } x \in 1..\mathrm{cols}(I)-2 \\ \quad\quad \left| \begin{array}{l} \text{for } i \in 0..2 \\ \quad \text{for } j \in 0..2 \\ \quad\quad v_{3\cdot i+j} \leftarrow I_{y+i-1,x+j-1} \\ vtri \leftarrow \mathrm{sort}(v) \\ J_{y,x} \leftarrow vtri_4 \end{array} \right. \\ J \end{array} \right.$$

$$\mathrm{ET}(x,y) \equiv \left| \begin{array}{l} 0 \quad \text{if } x = 0 \\ 0 \quad \text{if } y = 0 \\ 1 \quad \text{otherwise} \end{array} \right. \qquad \mathrm{OU}(x,y) \equiv \left| \begin{array}{l} 1 \quad \text{if } x = 1 \\ 1 \quad \text{if } y = 1 \\ 0 \quad \text{otherwise} \end{array} \right.$$

$$\mathrm{Moyenne}(I) \equiv \left| \begin{array}{l} M \leftarrow \begin{pmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{pmatrix} \\ J \leftarrow \mathrm{Convol3x3}(I,M) \\ J \end{array} \right.$$

$$\mathrm{Dilatation}(I) \equiv \left| \begin{array}{l} \text{for } y \in 1..\mathrm{rows}(I)-2 \\ \quad \text{for } x \in 1..\mathrm{cols}(I)-2 \\ \quad\quad \left| \begin{array}{l} \text{for } i \in 0..2 \\ \quad \text{for } j \in 0..2 \\ \quad\quad v_{3\cdot i+j} \leftarrow \dfrac{I_{y+i-1,x+j-1}}{255} \\ Test \leftarrow v_0 \\ \text{for } i \in 1..8 \\ \quad Test \leftarrow v_i \vee Test \quad \text{if } i \neq 4 \\ J_{y,x} \leftarrow 255 \quad \text{if } Test = 1 \\ J_{y,x} \leftarrow I_{y,x} \quad \text{otherwise} \end{array} \right. \\ J \end{array} \right.$$

$$\mathrm{DilatationGris}(I) \equiv \left| \begin{array}{l} \text{for } y \in 1..\mathrm{rows}(I)-2 \\ \quad \text{for } x \in 1..\mathrm{cols}(I)-2 \\ \quad\quad \left| \begin{array}{l} \text{for } i \in 0..2 \\ \quad \text{for } j \in 0..2 \\ \quad\quad v_{3\cdot i+j} \leftarrow I_{y+i-1,x+j-1} \\ J_{y,x} \leftarrow \max(v) \end{array} \right. \\ J \end{array} \right.$$

```
Erosion(I) ≡ | for y ∈ 1 .. rows(I) − 2
             |   for x ∈ 1 .. cols(I) − 2
             |     | for i ∈ 0 .. 2
             |     |   for j ∈ 0 .. 2
             |     |     v[3·i+j] ← I[y+i−1, x+j−1] / 255
             |     | Test ← v[0]
             |     | for i ∈ 1 .. 8
             |     |   Test ← v[i] ∧ Test  if i ≠ 4
             |     | J[y,x] ← I[y,x]  if Test = 1
             |     | J[y,x] ← 0  otherwise
             | J
```

```
ErosionGris(I) ≡ | for y ∈ 1 .. rows(I) − 2
                 |   for x ∈ 1 .. cols(I) − 2
                 |     | for i ∈ 0 .. 2
                 |     |   for j ∈ 0 .. 2
                 |     |     v[3·i+j] ← I[y+i−1, x+j−1]
                 |     | J[y,x] ← min(v)
                 | J
```

```
NormeGradient4Diff(I) ≡ | for y ∈ 1 .. rows(I) − 2
                        |   for x ∈ 1 .. cols(I) − 2
                        |     | d[0] ← I[y,x] − I[y,x−1]
                        |     | d[1] ← I[y−1,x] − I[y,x−1]
                        |     | d[2] ← I[y−1,x] − I[y,x]
                        |     | d[3] ← I[y−1,x−1] − I[y,x]
                        |     | for i ∈ 0 .. 3
                        |     |   D[i] ← |d[i]|
                        |     | NormeGradient[y,x] ← max(D)
                        | NormeGradient
```

```
Ampli(I, gain) ≡ | for y ∈ 0 .. rows(I) − 1
                 |   for x ∈ 0 .. cols(I) − 1
                 |     | r ← I[y,x]·gain
                 |     | J[y,x] ← r
                 |     | J[y,x] ← 255  if r > 255
                 |     | J[y,x] ← 0  if r < 0
                 | J
```

```
DirGradient4Diff(I) ≡ | for y ∈ 1 .. rows(I) − 2
                      |   for x ∈ 1 .. cols(I) − 2
                      |     | d[0] ← I[y,x] − I[y,x−1]
                      |     | d[1] ← I[y−1,x] − I[y,x−1]
                      |     | d[2] ← I[y−1,x] − I[y,x]
                      |     | d[3] ← I[y−1,x−1] − I[y,x]
                      |     | for i ∈ 0 .. 3
                      |     |   D[i] ← |d[i]|
                      |     | imax ← IndiceMax(D)
                      |     | DirGradient[y,x] ← imax  if d[imax] ≥ 0
                      |     | DirGradient[y,x] ← imax + 4  otherwise
                      | DirGradient
```

```
IndiceMax(v) ≡ | N ← length(v)
               | IndiceMax ← 0
               | Max ← v[0]
               | for i ∈ 1 .. N − 1
               |   if v[i] > Max
               |     | Max ← v[i]
               |     | IndiceMax ← i
               | IndiceMax
```

Codage RVB de la Direction du Gradient :

Noir (RVB=0,0,0) Blanc (RVB=255,255,255) Gris (RVB=127,127,127)
Rouge (RVB=255,0,0) Magenta (RVB=255,0,255) Marron(127,0,0)
Vert (RVB=0,255,0) Kaki (RVB=127,127,0)
Bleu (RVB=0,0,255) Cyan (RVB=0,255,255) Violet (RVB=127,0,127)
Jaune (RVB=255,255,0) Orange (RVB=255,127,0)

Direction 2 = N = Bleu
Direction 3 = NO = Orange Direction 1 = NE = Vert
Direction 4 = O = Magenta Direction 0 = E = Rouge
Direction 5 = SO = Kaki Direction 7 = SE = Jaune
Direction 6 = S = Cyan

$$
\mathrm{Rvb}(I) \equiv \left|\begin{array}{l}
\text{for } y \in 0..\mathrm{rows}(I)-1 \\
\quad \text{for } x \in 0..\mathrm{cols}(I)-1 \\
\quad\quad \left|\begin{array}{l}
R_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 0 \\
R_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 1 \\
R_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 2 \\
R_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 3 \\
R_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 4 \\
R_{y,x} \leftarrow 127 \text{ if } I_{y,x} = 5 \\
R_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 6 \\
R_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 7
\end{array}\right. \\
R
\end{array}\right.
\qquad
\mathrm{rVb}(I) \equiv \left|\begin{array}{l}
\text{for } y \in 0..\mathrm{rows}(I)-1 \\
\quad \text{for } x \in 0..\mathrm{cols}(I)-1 \\
\quad\quad \left|\begin{array}{l}
V_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 0 \\
V_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 1 \\
V_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 2 \\
V_{y,x} \leftarrow 127 \text{ if } I_{y,x} = 3 \\
V_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 4 \\
V_{y,x} \leftarrow 127 \text{ if } I_{y,x} = 5 \\
V_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 6 \\
V_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 7
\end{array}\right. \\
V
\end{array}\right.
\qquad
\mathrm{rvB}(I) \equiv \left|\begin{array}{l}
\text{for } y \in 0..\mathrm{rows}(I)-1 \\
\quad \text{for } x \in 0..\mathrm{cols}(I)-1 \\
\quad\quad \left|\begin{array}{l}
B_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 0 \\
B_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 1 \\
B_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 2 \\
B_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 3 \\
B_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 4 \\
B_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 5 \\
B_{y,x} \leftarrow 255 \text{ if } I_{y,x} = 6 \\
B_{y,x} \leftarrow 0 \text{ if } I_{y,x} = 7
\end{array}\right. \\
B
\end{array}\right.
$$

$$\Delta x \equiv (1 \quad 1 \quad 0 \quad -1 \quad -1 \quad -1 \quad 0 \quad 1)^T \qquad \Delta y \equiv (0 \quad -1 \quad -1 \quad -1 \quad 0 \quad 1 \quad 1 \quad 1)^T$$

$$
\mathrm{SuppressionNonMaximaLocaux}(M, D) \equiv \left|\begin{array}{l}
\text{for } y \in 1..\mathrm{rows}(M)-2 \\
\quad \text{for } x \in 1..\mathrm{cols}(M)-2 \\
\quad\quad \left|\begin{array}{l}
i \leftarrow D_{y,x} \\
j \leftarrow \mathrm{mod}(i+4, 8) \\
M_{y,x} \leftarrow 0 \text{ if } M_{y,x} \le M_{y+\Delta y_i,\, x+\Delta x_i} \\
M_{y,x} \leftarrow 0 \text{ if } M_{y,x} \le M_{y+\Delta y_j,\, x+\Delta x_j}
\end{array}\right. \\
M
\end{array}\right.
$$

$$D_1 \equiv \begin{pmatrix} 0&0&0&0&1\\0&0&0&1&1\\0&0&1&1&1\\0&0&0&0&0\\0&0&0&0&0 \end{pmatrix} \quad D_2 \equiv \begin{pmatrix} 0&0&1&1&1\\0&0&1&1&0\\0&0&1&0&0\\0&0&0&0&0\\0&0&0&0&0 \end{pmatrix} \quad D_3 \equiv \begin{pmatrix} 1&1&1&0&0\\0&1&1&0&0\\0&0&1&0&0\\0&0&0&0&0\\0&0&0&0&0 \end{pmatrix} \quad D_4 \equiv \begin{pmatrix} 1&0&0&0&0\\1&1&0&0&0\\1&1&1&0&0\\0&0&0&0&0\\0&0&0&0&0 \end{pmatrix}$$

$$D_5 \equiv \begin{pmatrix} 0&0&0&0&0\\0&0&0&0&0\\1&1&1&0&0\\1&1&0&0&0\\1&0&0&0&0 \end{pmatrix} \quad D_6 \equiv \begin{pmatrix} 0&0&0&0&0\\0&0&0&0&0\\0&0&1&0&0\\0&1&1&0&0\\1&1&1&0&0 \end{pmatrix} \quad D_7 \equiv \begin{pmatrix} 0&0&0&0&0\\0&0&0&0&0\\0&0&1&0&0\\0&0&1&1&0\\0&0&1&1&1 \end{pmatrix} \quad D_8 \equiv \begin{pmatrix} 0&0&0&0&0\\0&0&0&0&0\\0&0&1&1&1\\0&0&0&1&1\\0&0&0&0&1 \end{pmatrix}$$

```
Chanda(I) ≡ | for y ∈ 2 .. rows(I) − 3
            |   for x ∈ 2 .. cols(I) − 3
            |     SDG ← 0
            |     SG ← 0
            |     for k ∈ 1 .. 8
            |       S ← 0
            |       for j ∈ 0 .. 4
            |         for i ∈ 0 .. 4
            |           S ← S + [ I_(y+j−2, x+i−2) · (D_k)_(j,i) ]
            |       S ← S / 6
            |       S2 ← 0
            |       for j ∈ 0 .. 4
            |         for i ∈ 0 .. 4
            |           S2 ← S2 + [ I_(y+j−2, x+i−2) · (D_k)_(j,i) ]^2
            |       S2 ← S2 / 6
            |       D ← S − I_(y,x)
            |       G ← ( |S2 − S^2| )^(1/2)
            |       SDG ← SDG + D·G
            |       SG ← SG + G
            |     J_(y,x) ← I_(y,x) − SDG / SG
            |     J_(y,x) ← 0 if J_(y,x) < 0
            |     J_(y,x) ← 255 if J_(y,x) > 255
            | J
```

$$D_1 \equiv \begin{pmatrix} 0&0&0&0&1\\ 0&0&0&1&1\\ 0&0&1&1&1\\ 0&0&0&0&0\\ 0&0&0&0&0 \end{pmatrix} \quad D_2 \equiv \begin{pmatrix} 0&0&1&1&1\\ 0&0&1&1&0\\ 0&0&1&0&0\\ 0&0&0&0&0\\ 0&0&0&0&0 \end{pmatrix} \quad D_3 \equiv \begin{pmatrix} 1&1&1&0&0\\ 0&1&1&0&0\\ 0&0&1&0&0\\ 0&0&0&0&0\\ 0&0&0&0&0 \end{pmatrix} \quad D_4 \equiv \begin{pmatrix} 1&0&0&0&0\\ 1&1&0&0&0\\ 1&1&1&0&0\\ 0&0&0&0&0\\ 0&0&0&0&0 \end{pmatrix}$$

$$D_5 \equiv \begin{pmatrix} 0&0&0&0&0\\ 0&0&0&0&0\\ 1&1&1&0&0\\ 1&1&0&0&0\\ 1&0&0&0&0 \end{pmatrix} \quad D_6 \equiv \begin{pmatrix} 0&0&0&0&0\\ 0&0&0&0&0\\ 0&0&1&0&0\\ 0&1&1&0&0\\ 1&1&1&0&0 \end{pmatrix} \quad D_7 \equiv \begin{pmatrix} 0&0&0&0&0\\ 0&0&0&0&0\\ 0&0&1&0&0\\ 0&0&1&1&0\\ 0&0&1&1&1 \end{pmatrix} \quad D_8 \equiv \begin{pmatrix} 0&0&0&0&0\\ 0&0&0&0&0\\ 0&0&1&1&1\\ 0&0&0&1&1\\ 0&0&0&0&1 \end{pmatrix}$$

```
ChandaV(I) ≡ | for y ∈ 2 .. rows(I) − 3
             |   for x ∈ 2 .. cols(I) − 3
             |     SDG ← 0
             |     SG ← 0
             |     for k ∈ 1 .. 8
             |       S ← 0
             |       for j ∈ 0 .. 4
             |         for i ∈ 0 .. 4
             |           S ← S + [ I[y+j−2, x+i−2 · (D_k)[j, i] ]
             |       S ← S / 6
             |       S2 ← S^2 / 6
             |       D ← S − I[y, x]
             |       G ← (|S2 − S^2|)^(1/2)
             |       SDG ← SDG + D·G
             |       SG ← SG + G
             |     J[y, x] ← I[y, x] − SDG / SG
             |     J[y, x] ← 0   if J[y, x] < 0
             |     J[y, x] ← 255   if J[y, x] > 255
             | J
```

```
Nagao(I) ≡ | for  y ∈ 2 .. rows(I) − 3
           |   for  x ∈ 2 .. cols(I) − 3
           |     | for  j ∈ 0 .. 2
           |     |   for  i ∈ 0 .. 2
           |     |     | for  m ∈ 0 .. 2
           |     |     |   for  k ∈ 0 .. 2
           |     |     |     Tmp_{m·3+k} ← I_{y+j+m−2, x+i+k−2}
           |     |     | Moy_{3·j+i} ← mean(Tmp)  if  i·j ≠ 1
           |     |     | Moy_{3·j+i} ← 0  otherwise
           |     |     | Var_{3·j+i} ← max(Tmp) − min(Tmp)  if  i·j ≠ 1
           |     |     | Var_{3·j+i} ← 255  otherwise
           |     | Mini ← min(Var)
           |     | for  i ∈ 0 .. 8
           |     |   W_{y−2, x−2} ← Moy_i  if  Var_i = Mini
           | W

ChinWan(I) ≡ | for  y ∈ 2 .. rows(I) − 3
             |   for  x ∈ 2 .. cols(I) − 3
             |
             | J
```

$$
\mathrm{TabHough}(I,\Delta\rho,\Delta\theta) \equiv
\left|
\begin{array}{l}
\rho max \leftarrow \sqrt{\mathrm{cols}(I)^2 + \mathrm{rows}(I)^2} \\
\text{for } t \in 0 .. \dfrac{180}{\Delta\theta} \\
\quad \text{for } r \in 0 .. \mathrm{trunc}\left(2 \cdot \dfrac{\rho max}{\Delta\rho}\right) \\
\quad\quad Accu_{r,t} \leftarrow 0 \\
\text{for } y \in 0 .. \mathrm{rows}(I) - 1 \\
\quad \text{for } x \in 0 .. \mathrm{cols}(I) - 1 \\
\quad\quad \text{for } \theta \in -90, -90 + \Delta\theta .. 90 \qquad \text{if } I_{y,x} = 255 \\
\quad\quad\quad \left|
\begin{array}{l}
\theta r \leftarrow \theta \cdot \dfrac{\pi}{180} \\
\rho \leftarrow \mathrm{trunc}(x \cdot \cos(\theta r) + y \cdot \sin(\theta r)) \\
r \leftarrow \mathrm{trunc}\left(\dfrac{\rho + \rho max}{\Delta\rho}\right) \\
t \leftarrow \mathrm{trunc}\left(\dfrac{\theta + 90}{\Delta\theta}\right) \\
Accu_{r,t} \leftarrow Accu_{r,t} + 1
\end{array}
\right. \\
Accu
\end{array}
\right.
$$

$$
\mathrm{Hough}(I,\Delta\rho,\Delta\theta,seuil) \equiv
\left|
\begin{array}{l}
\rho max \leftarrow \sqrt{\mathrm{cols}(I)^2 + \mathrm{rows}(I)^2} \\
Accu \leftarrow \mathrm{TabHough}(I,\Delta\rho,\Delta\theta) \\
\text{for } y \in 0 .. \mathrm{rows}(I) - 1 \\
\quad \text{for } x \in 0 .. \mathrm{cols}(I) - 1 \\
\quad\quad \left|
\begin{array}{l}
J_{y,x} \leftarrow I_{y,x} \\
keep \leftarrow 0 \\
\text{for } \theta \in -90, -90 + \Delta\theta .. 90 \qquad \text{if } I_{y,x} = 255 \\
\quad \left|
\begin{array}{l}
\theta r \leftarrow \theta \cdot \dfrac{\pi}{180} \\
\rho \leftarrow \mathrm{trunc}(x \cdot \cos(\theta r) + y \cdot \sin(\theta r)) \\
r \leftarrow \mathrm{trunc}\left(\dfrac{\rho + \rho max}{\Delta\rho}\right) \\
t \leftarrow \mathrm{trunc}\left(\dfrac{\theta + 90}{\Delta\theta}\right) \\
keep \leftarrow 1 \quad \text{if } Accu_{r,t} \ge seuil
\end{array}
\right. \\
J_{y,x} \leftarrow 0 \quad \text{if } keep = 0
\end{array}
\right. \\
J
\end{array}
\right.
$$

Blob itératif : *Blobi(C)* où *C* est une image binaire de contours (fermés) (blancs sur fond noir) renvoie une image *R* constituée d'étiquettes (entiers positifs non forcément consécutifs) affectées aux pixels des régions (une même étiquette est affectée aux pixels d'une même région). L'étiquette 0 est attribuée aux pixels de contour.
La fonction *NbRegions(R)* compte ensuite le nombre d'étiquettes différentes de l'image d'étiquettes *R* donnant ainsi le nombre de régions de *R* donc de C.

```
minnonnul(v) ≡ | maxi ← 10^10
               | N ← length(v)
               | for n ∈ 0..N − 1
               |   | m ← min(v)
               |   | for k ∈ 0..N − 1        if m = 0
               |   |   v_k ← maxi  if  v_k = m
               | m ← min(v)
               | m ← 0  if  m = maxi
               | m
```

<- minnonnul(v) renvoie le plus petit élément non nul du vecteur 1D v , et renvoie 0 si tous les éléments de v sont nuls.

Gestion des effets de bord rendue inutile grâce au cadre initial de 0 dans Blobi(C)

```
iter1Blobi(J) ≡ | n ← 1
                | for y ∈ 1..rows(J) − 2
                |   for x ∈ 1..cols(J) − 2
                |     if J_{y,x} ≠ 0
                |       | v ← (J_{y−1,x−1}  J_{y−1,x}  J_{y−1,x+1}  J_{y,x−1})^T
                |       | J_{y,x} ← min(v)  if  v_0 ≠ 0 ∧ v_1 ≠ 0 ∧ v_2 ≠ 0 ∧ v_3 ≠ 0
                |       | otherwise
                |       |   | J_{y,x} ← n
                |       |   | n ← n + 1
                | J
```

<- iter1Blobi(J) renvoie pour chaque pixel p non nul de l'image J , le plus petit prédécesseur non nul de p,
renvoie l'étiquette n si un au moins des prédécesseurs de p est nul,
renvoie p inchangé si p est nul.

Gestion des effets de bord rendue inutile grâce au cadre initial de 0 dans Blobi(C)

<- iter2Blobi(J) renvoie pour chaque pixel p non nul de l'image J , le plus petit élément non nul du voisinage 3x3 de p,
renvoie p inchangé si p est nul.

```
iter2Blobi(J) ≡ | for y ∈ 1..rows(J) − 2
                |   for x ∈ 1..cols(J) − 2
                |     if J_{y,x} ≠ 0
                |       | v ← (J_{y−1,x−1}  J_{y−1,x}  J_{y−1,x+1}  J_{y,x−1}  J_{y,x}  J_{y,x+1}  J_{y+1,x−1}  J_{y+1,x}  J_{y+1,x+1})^T
                |       | J_{y,x} ← minnonnul(v)
                | J
```

Cadre initial avec des 0, non obligatoire mais c'est pour éviter de gérer les effets de bord !

```
Blobi(C) ≡ | n ← 1
           | for y ∈ 0..rows(C) − 1
           |   for x ∈ 0..cols(C) − 1
           |     | J[y,x] ← 0 if C[y,x] = 255
           |     | J[y,x] ← n otherwise
           | for y ∈ 0..rows(C) − 1
           |   for x ∈ 0..cols(C) − 1
           |     J[y,x] ← 0 if x = 0 ∨ y = 0 ∨ x = cols(C) − 1 ∨ y = rows(C) − 1
           | J ← iter1Blobi(J)
           | iter ← 1
           | while iter = 1
           |   | K ← J
           |   | J ← iter2Blobi(J)
           |   | iter ← 0 if J = K
           | J
```

<- Blobi(C) renvoie l'image d'étiquettes J

. à partir de l'image de contours C, constitution de l'image d'étiquettes J telle que les points contour (blancs) ont l'étiquette 0 et les autres points (noirs) ont l'étiquette 1.

. cadre de l'image J avec l'étiquette 0.

. J = iter1Blobi(J) 1 seule passe.

. plusieurs passes (jusqu'à stabilité, invariance) de iter2Blobi(J).

```
Consecution(I) ≡ | nbnivo ← 1
                 | nivo[nbnivo−1] ← I[0,0]
                 | for y ∈ 0..rows(I) − 1
                 |   for x ∈ 0..cols(I) − 1
                 |     | newnivo ← 1
                 |     | for n ∈ 0..nbnivo − 1
                 |     |   newnivo ← 0 if I[y,x] = nivo[n]
                 |     | if newnivo = 1
                 |     |   | nbnivo ← nbnivo + 1
                 |     |   | nivo[nbnivo−1] ← I[y,x]
                 | for y ∈ 0..rows(I) − 1
                 |   for x ∈ 0..cols(I) − 1
                 |     | for n ∈ 0..nbnivo − 1
                 |     |   index ← n if I[y,x] = nivo[n]
                 |     | J[y,x] ← index
                 | J
```

<- Consecution(I) réorganise l'image des étiquettes I de sorte à ce que les étiquettes soient consécutives (sans étiquettes manquantes) en partant de 0.

v NbReg(I) renvoie le nombre de régions de l'image des étiquettes I :
NbReg(I) = Nombre d'étiquettes différentes - 1
car est à retirer l'étiquette 0 (contours).
On a aussi : NbReg(I) = max(Consecution(I)).

```
NbReg(I) ≡ | nbnivo ← 1
           | nivo[nbnivo−1] ← I[0,0]
           | for y ∈ 0..rows(I) − 1
           |   for x ∈ 0..cols(I) − 1
           |     | newnivo ← 1
           |     | for n ∈ 0..nbnivo − 1
           |     |   newnivo ← 0 if I[y,x] = nivo[n]
           |     | if newnivo = 1
           |     |   | nbnivo ← nbnivo + 1
           |     |   | nivo[nbnivo−1] ← I[y,x]
           | nbnivo − 1
```
