N-601

TASIC Programs

S. BEUCHER

FONTAINEBLEAU JUIN 1979

TASIC Programs :

Explicatory notes

- Gradient
- Gradient and densitogram
- Ultimate erosions
- Skeleton with LUD Card
- Balls Separation
- Skeleton with Ring Logic
- Picture segmentation by conditional bisectrix

```
 1 PNT(' GRADIENT (DILATATION)')
 2 DLM1000
 3 CMR
 4 SYR
 5 INP ('$THRESHOLD INTERVAL ? ')I1
 6 INP ('$DILATATION SIZE ? ')I2
 7 FNCI3=99-I1
 8 DLSGRAD,I4=1,I3
 9 DNAO,I4
10 DEF7,3,1,I2
11 ASG3,2,3
12 MOVO,1
13 FNCI5=I4+I1
14 DNAI5
15 ASG2,2,2
16 MOVO,1,1
17 CLD
18 ACX2,3,3
19 MOVO,1,1
20 ACX3,0,0,4
21 MOVO,1
22 DLEGRAD
23 DSGO,2
24 HLT
25 HME
26
```

Program n° 1 : Gradient

This program detects the picture-points characterized by a gradient modulus greater than a given value defined by the two parameters :"Threshold Interval" and "Dilatation Size" (I1 and I2)

Listing Comments

- instructions 3 and 4

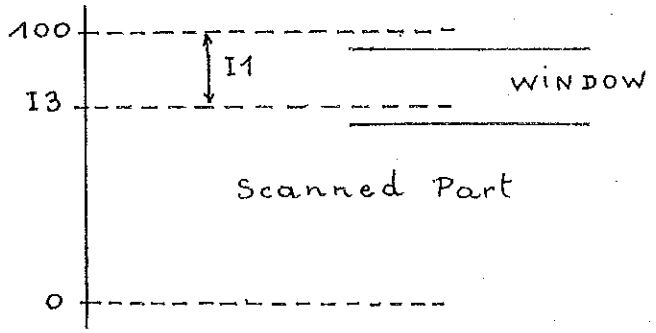
Synchrone Read off, and clear memory ϕ. This memory will contain the final result of the detection.

- instructions 5 and 6

I1, I2 input ; The value of the minimum gradient modulus is proportional to the ratio I1/I2. The highest resolution is obtained when I2 = 1.

- instruction 7

The window defined with I1 is scanned along the grey scale. To avoid program errors, we define here the upper limit of scanning (see figure below)



- instructions 8 to 22

Scanning of the window. Its position is defined by I4.

- instructions 9 to 12

Selection of the threshold [ϕ, I4] which is dilated (dilatation size, I2) and stored in memory 3.

- instructions 13 to 16

Selection of the threshold [I5, 99], dilatation and storage in memory 2.

- instructions 18, 19

Intersection of memories 2 and 3 result in memory 3. At each iteration, we perform the following transformation :

$$(S_{[\phi,I4]} \oplus H_{I2}) \cap (S_{[I5,99]} \oplus H_{I2})$$

- instructions 20 to 22

The intermediary result is stored in memory ϕ. At the end of the program, this memory contains the set defined by :

$$\bigcup_{I4 = 1,I3} {}^{S}{}_{[\phi,I4]} \oplus H_{I2}) \cap (S_{[I5,99]} \oplus H_{I2})$$

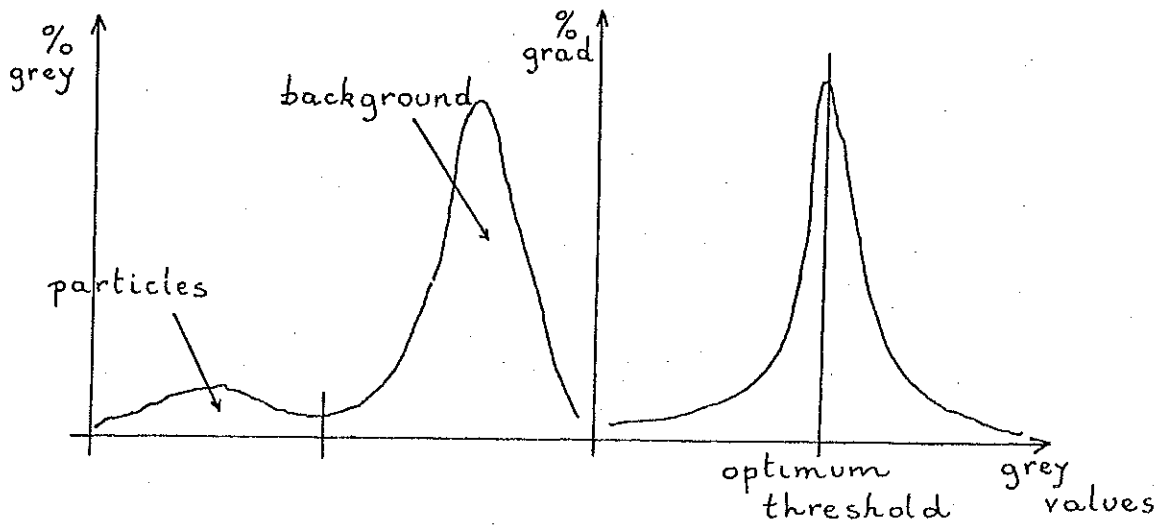
which is the set of all the points of the image whose gradients are greater than I1/I2 (up to a multiplicative constant).

When increasing both the values of I1 and I2, the detected gradient is unchanged. But the final result has a lower resolution. It is possible to reduce the effects of noise, but this method also has a smoothing effect on the final picture.

```
 1 CMR2
 2 SYR
 3 INP('$TAILLE ')I6
 4 INP('$PARAM. ')I4
 5 FNCI5=99-I4
 6 DLSBAL,I1=1,I5
 7 FNCI2=I1+I4
 8 DNA0,I1
 9 ASG0,2
10 MOV0,1,1
11 DEF7,1,0,1
12 ASG0,2,3
13 MOV0,1,1
14 CLD
15 ACX0,3,3,3
16 MOV0,1
17 FNCI3=I2
18 DNAI3,0
19 ASG0,2,0
20 DEF7,1,3,I6
21 MOV0,1,1
22 CLD
23 ACX3,0,1
24 AREX1(I1)
25 MOV0,1,1
26 ACX2,1,2,4
27 MOV0,1
28 DLEBAL
29 DSG2,2
30 PLOX1(1),I5,0,1,0,0,0,6,80
31 HLT
32 HME
33
```

Program n° 2 : Gradient and densitogram

A gradient densitogram is included in this program. The gradient densitogram is useful for thresholding a picture when the objects of interest in the picture are of small size (The typical example is a diamond powder). The maximum value of the densitogram gives the optimum threshold. This maximum corresponds to a minimum of the classical densitogram. But it is generally easier to recognize a maximum than a fuzzy minimum (see example below)



Listing comments

The gradient detection algorithm is not symetric.

- instructions 3 to 5

I4 is the size of the threshold window, I6 the size of Dilation, I5 the upper limit of Scanning.

- instructions 6 to 28

gradient detection : loop BAL

- instructions 8 to 10

Selection of a threshold [ϕ, I1] and storage in memory ϕ.

- instructions 11 to 14

Erosion (of size 1) of the content of memo ϕ. Result in memo 3.

- instructions 15, 16

Set difference $(3)^c \cap (\phi)$. Result in (3). This memory now contains the boundary of the set stored in (ϕ).

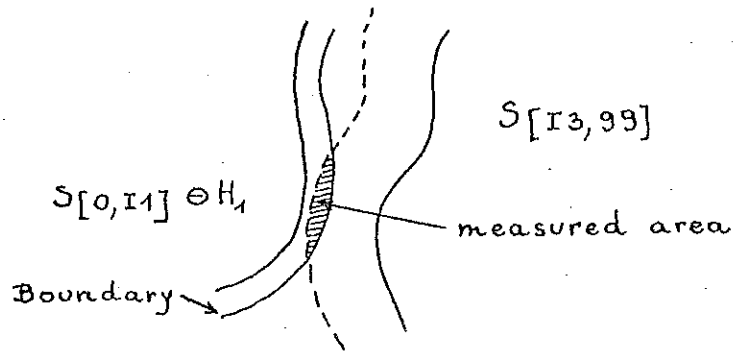
- instructions 17 to 22

Selection of a threshold [I3,99], (I3 = I1 + I4), dilation of size I6, and storage in memory ϕ.

- instructions 23 to 28

Intersection $(3) \cap (\phi)$. The area is measured (value in array X1), and the resulting set is added to memory 2 (previously cleared at the beginning of the program).

Figure below illustrates the transformation :



- instructions 29 to 31

Display of the result and plotting of the densitogram.

The densitogram is normalized if we divide, at each step, the value X1 (I1) by the area of memory 3. (measured after instruction 15).

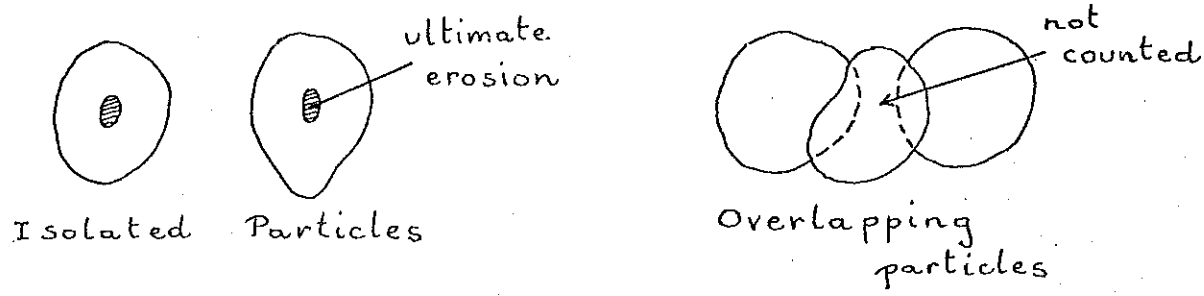
```
 1 GTO MAIN
 2 LBL BLW
 3 FNC R3=0
 4 DEF 7,2,3,1
 5 LBL 3
 6 FNC R2=R3
 7 ACX 2,3,2,0
 8 ARE R3
 9 MOV 0,1,1
10 IFC R3-R2,+,+,3
11 CLD
12 ACX 2,3,2,0
13 MOV 0,1,1
14 RSB
15 LBL MAIN
16 CMR 0
17 PCT
18 SYR 1,0
19 ASG 2,2,4
20 MOV 0,1,1
21 ASG 2,2,3
22 LBL 2
23 ARE R1
24 MOV 0,1,1
25 IFC R1,1,1,+
26 DEF 7,1,0,1
27 ASG 3,2,2
28 MOV 0,1,1
29 CLD
30 GSB BLW
31 ACX 3,2,1,3
32 MOV 0,1,1
33 ACX 1,0,0,4
34 MOV 0,1,1
35 DEF 7,1,0,1
36 ASG 3,2,3
37 GTO 2
38 LBL 1
39 CLD
40 ASG 4,2,1
41 MOV 0,1,1
42 DCX 1,0,3,0
43 HLT
44
```

Program n° 3 : Ultimate erosions

The use of ultimate erosions is very powerful when one wants to separate, or simply count, overlapping particules. It is also possible to use it to isolate grains from clusters (see program POWDAN. Powder Analysis - Version 2).

Important Remark :

When using this program, we must keep in mind the fundamental hypothesis concerning the shape of the analysed particules : The isolated particles are supposed to be connected for any erosion. Furthermore, the overlapping particles must not overlap each other too much; the figure below illustrates this situation :



Listing comments

A Blow-up subroutine is included (LBL BLW).

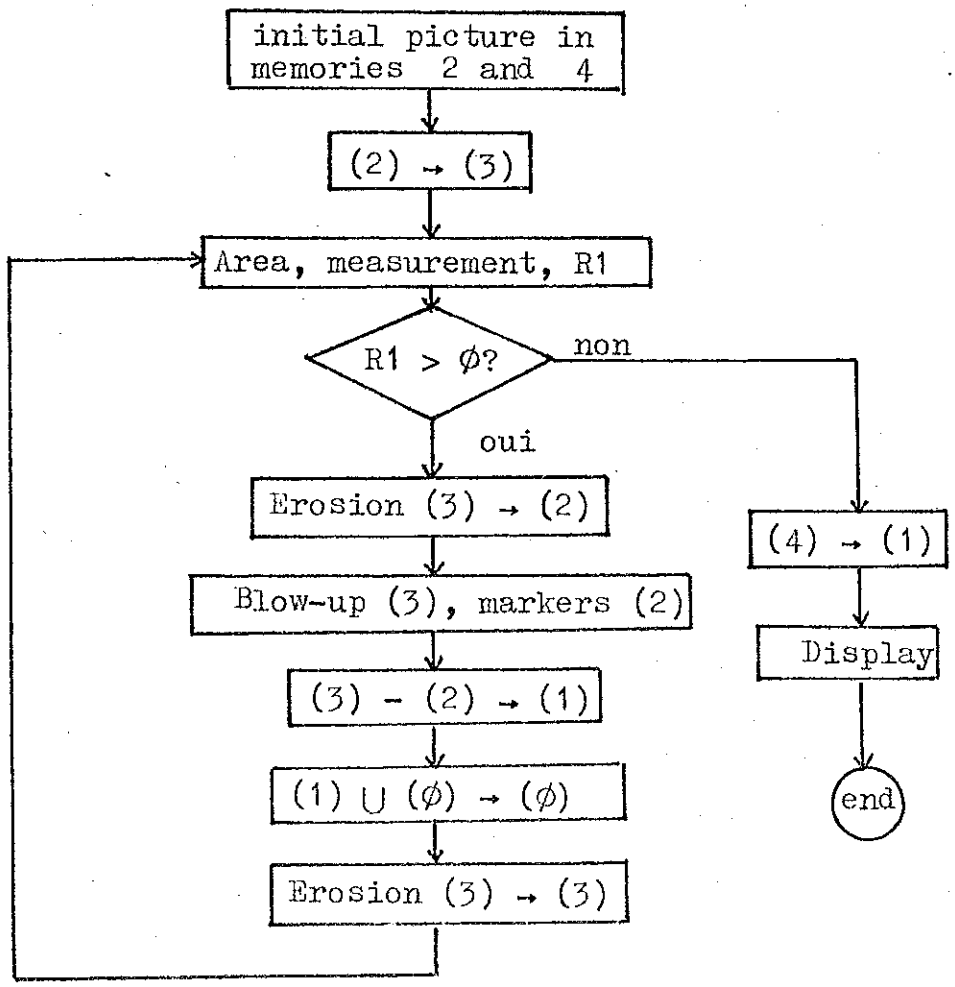
The markers are in memory 2 and the initial set in memory 3.
(see flow-chart of the program).

Let us consider the erosion of size i of the initial set, and the erosion of size i+1 :

$$\begin{cases} X \ominus H_i = Y_i \\ X \ominus H_{i+1} = Y_{i+1} \end{cases} \qquad X, \text{ initial set}$$

Each connected component of Y_i which cannot be reconstructed from Y_{i+1}, is an ultimate eroded set.

(If this was not the case, there would exist a connected component of Y_{i+1}, included in the component of Y_i). Thus the algorithm consists in the detection, for each size of erosion, of these connected components :



- instructions 31,32

At each step, the memory 1 contains the ultimate erosions detected for the size i.

- instructions 33,34

These ultimate erosions are added to those previously detected, in memory ∅. (final result in this memory).

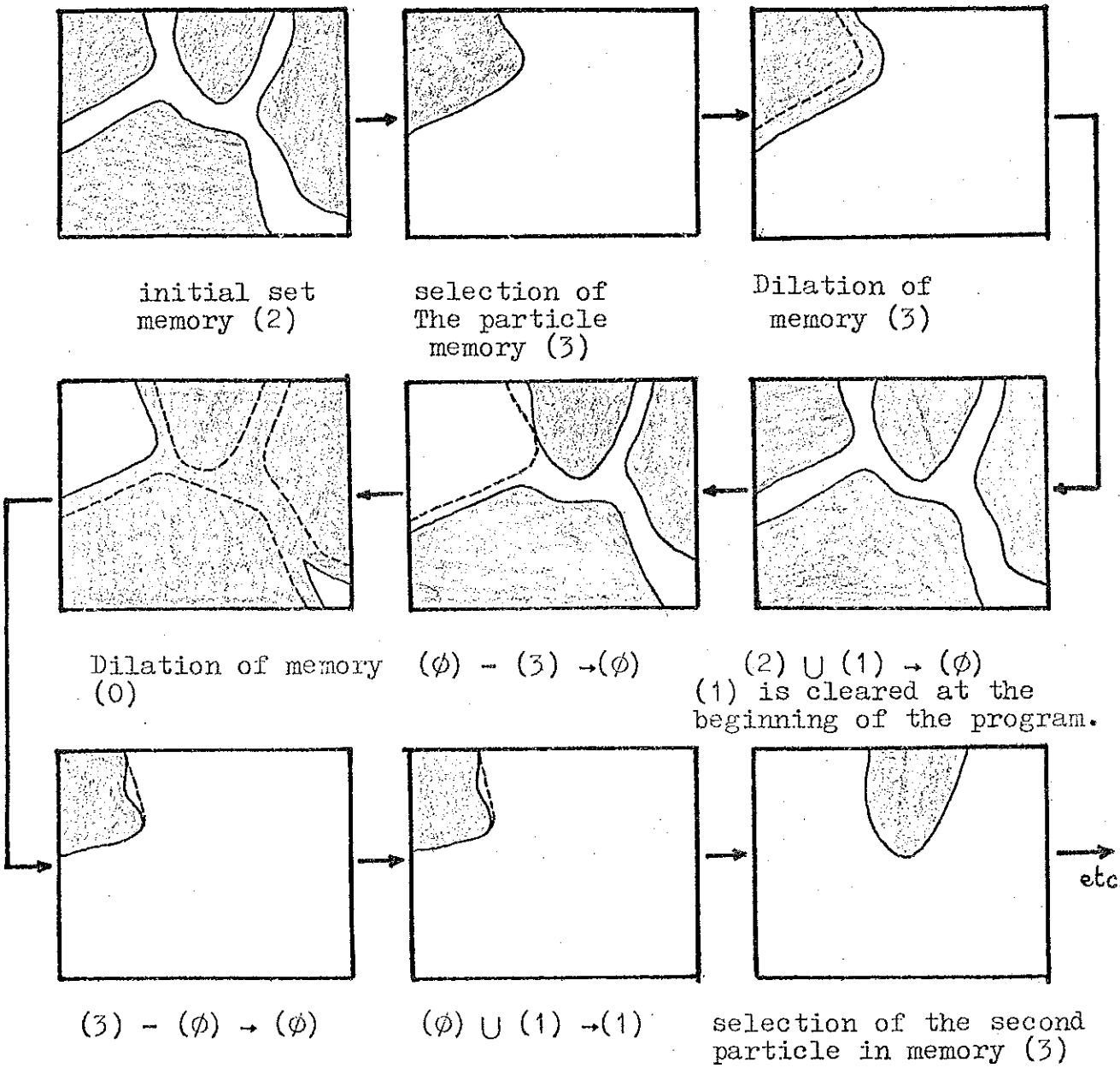
```
 1 PCT
 2 FNCR2=0
 3 LBLITR
 4 FNCR1=R2
 5 CMR1
 6 LBLIND
 7 AIA1,0,1,I1
 8 IFCI1,+,FIND,+
 9 D1M3
10 ACX2,1,0,4
11 MOV0,1,1
12 ACX0,3,0,3
13 MOV0,1,1
14 D1M0
15 ACX3,0,0,3
16 MOV0,1,1
17 ACX0,1,1,4
18 MOV0,1,1
19 GTOIND
20 LBLFIND
21 ASG1,2,2
22 ARER2
23 MOV0,1,1
24 IFCR2-R1,ITR,+,ITR
25 DSG2,2
26 HLT
27 HME
28
```

Program n° 4 : Skeleton with LUD Card. (particle by particle)

This skeleton program uses only the size 1. Dilation. It can be used to skeletonize tessellations of grains. Its use for any other type of image will be hazardous (for fibers, for instance).

Algorithm description

Figures below illustrate this algorithm



For each particle, we dilate and test that the dilation does not cut any other particle.
(See example on the hexagonal grid).

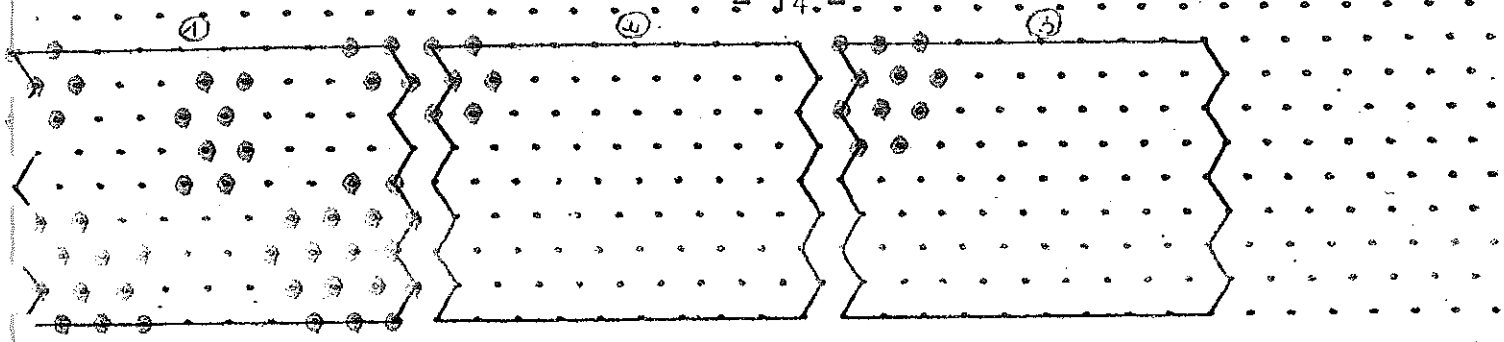
Listing comments

- instructions 6 to 19 :

Individual analysis and growth of the particles.

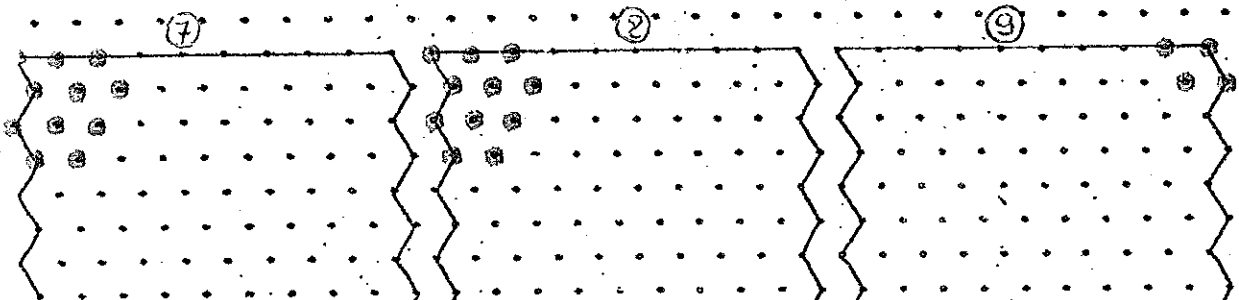
- instructions 20 to 24

The result of the first step is stored in memory (2). If the skeleton is not finished (R1 $\neq$ R2), we return to instruction 6 and perform a new step.

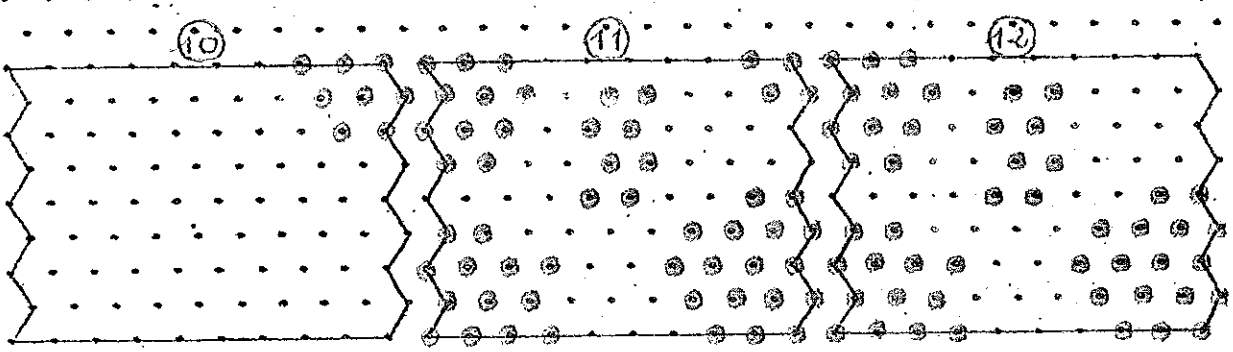


memo 2 memo 3 memo 3

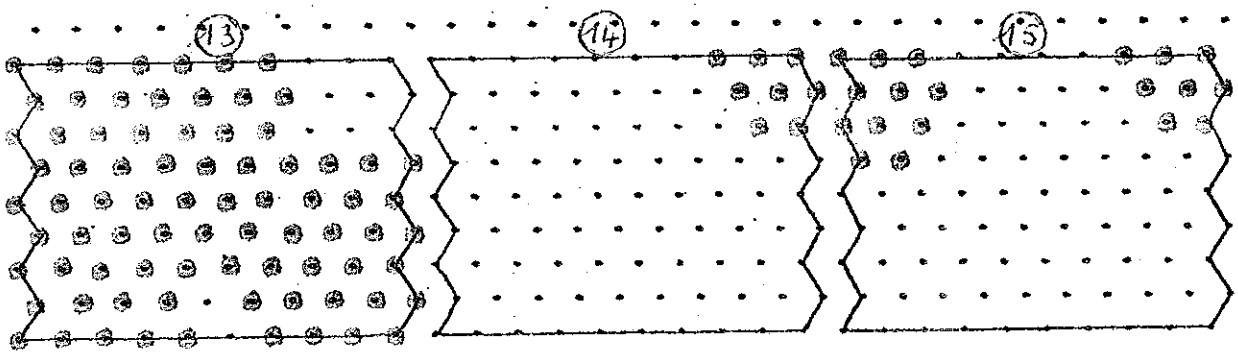
memo ø memo ø memo ø



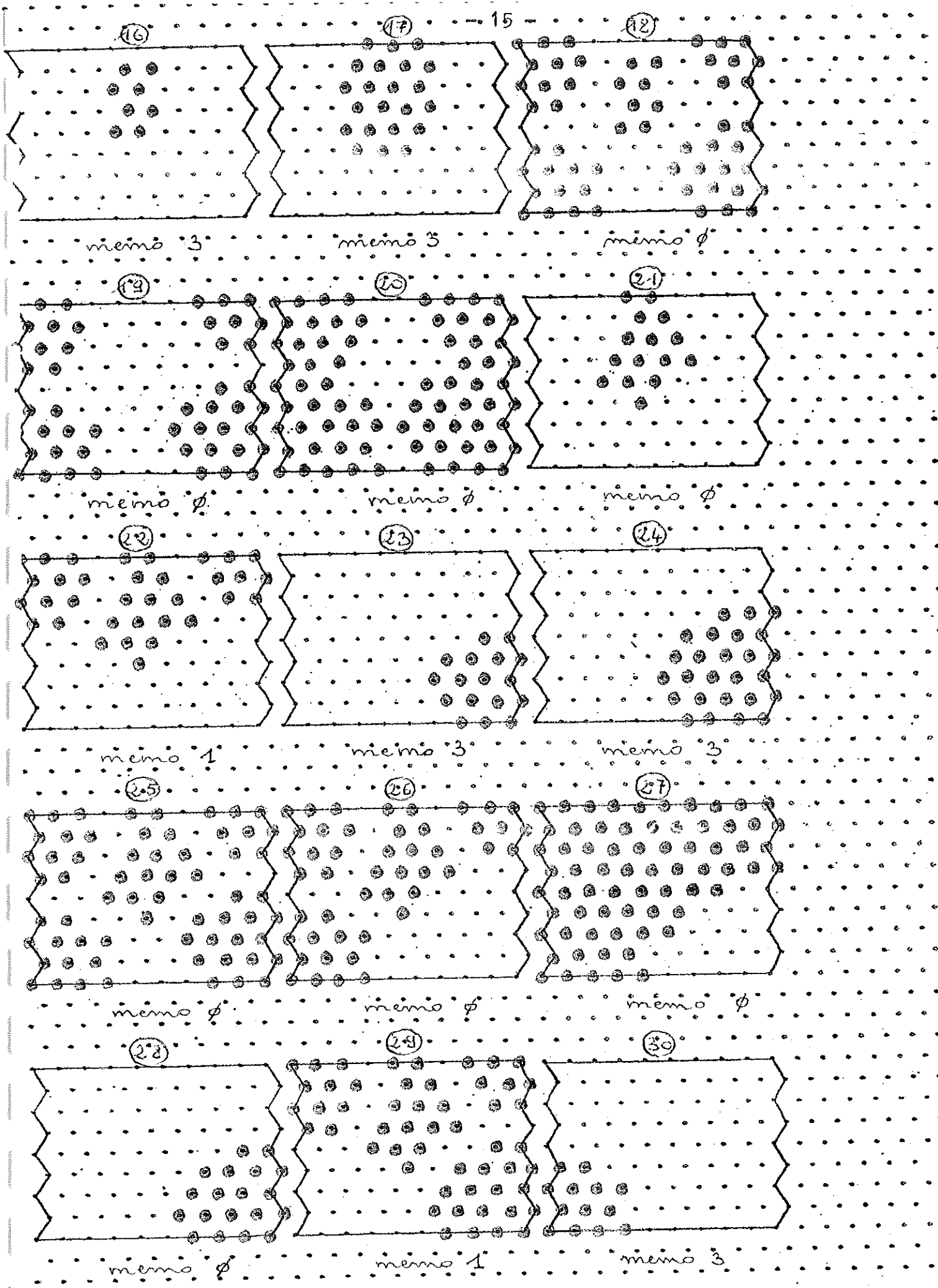
memo ø memo 1 memo 3

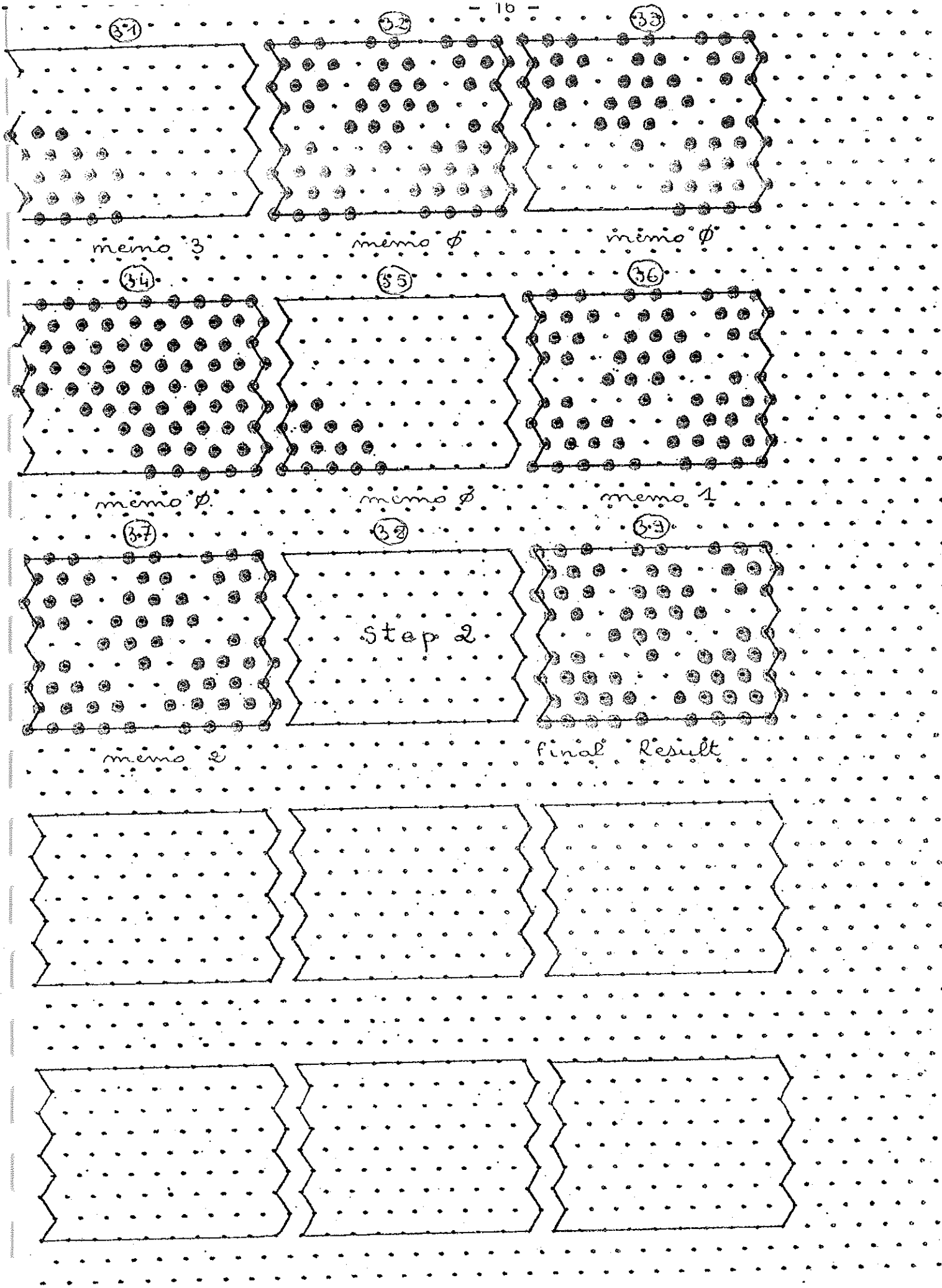


memo 3 memo ø memo ø



memo ø memo ø memo 1


16
17
18
memo 3
memo 3
memo ϕ
19
20
21
memo ϕ
memo ϕ
memo ϕ
22
23
24
memo 1
memo 3
memo 3
25
26
27
memo ϕ
memo ϕ
memo ϕ
28
29
30
memo ϕ
memo 1
memo 3


3.1
3.2
3.3
memo 3
memo ϕ
memo ϕ
3.4
3.5
3.6
memo ϕ
memo ϕ
memo 1
3.7
3.8
3.9
Step 2
memo 2
final Result

```
 1 GTOMAIN
 2 LBLER
 3 ASG4,2,3
 4 MOV0,1,1
 5 ASG3,2,3
 6 LBLB
 7 IFCI1-8,C,+,+
 8 DEF7,1,0,8
 9 MOV0,1,1
10 FNCI1=I1-8
11 GTOB
12 LBLC
13 IFCI1,E,E,+
14 DEF7,1,0,I1
15 MOV0,1,1
16 LBLE
17 CLD
18 RSB
19 LBLMAIN
20 PCT
21 SYR1
22 ASG2,2,4
23 MOV0,1,1
24 FNCI2=-1
25 LBLA
26 FNCI2=I2+1
27 DEF7,1,0,1
28 ASG2,2,2
29 ARER1
30 MOV0,1,1
31 IFCR1,+,+,A
32 CLD
33 CMR2
34 DLSREC,I9=0,I2
35 FNCI1=I2-I9
36 GSBER
37 LBL1
38 FNCR2=R1
39 DEF7,3,0,1
40 ASG2,2,2
41 RNG"27,"7,"0,"200
42 MOV0,1,1
43 RNG"56,"16,"0,"200
44 MOV0,1,1
45 RNG"35,"34,"0,"200
46 MOV0,1,1
47 RNG"72,"70,"0,"200
48 MOV0,1,1
49 RNG"65,"61,"0,"200
50 MOV0,1,1
51 RNG"53,"43,"0,"200
52 MOV0,1,1
53 CRI
54 CLD
55 ACX2,3,2
56 ARER1
57 MOV0,1,1
```

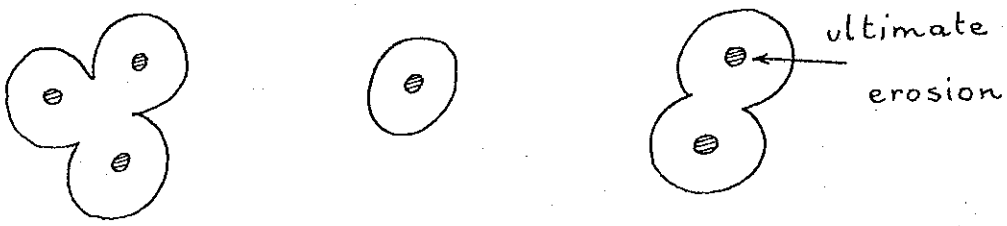
```
58 IFCR1-R2,+,+,1
59 LBL2
60 FNCR2=R1
61 DEF7,3,0,1
62 ASG2,2,2
63 RNG"17,"0,"0,"200
64 MOV0,1,1
65 RNG"36,"0,"0,"200
66 MOV0,1,1
67 RNG"74,"0,"0,"200
68 MOV0,1,1
69 RNG"71,"0,"0,"200
70 MOV0,1,1
71 RNG"63,"0,"0,"200
72 MOV0,1,1
73 RNG"47,"0,"0,"200
74 MOV0,1,1
75 CRI
76 CLD
77 ACX2,3,2
78 ARER1
79 MOV0,1,1
80 IFCR1-R2,+,+,2
81 ASG2,2,1
82 MOV0,1,1
83 BUP3,2
84 ACX3,2,2,3
85 MOV0,1,1
86 ACX1,2,2,4
87 MOV0,1,1
88 DLEREC
89 DSG2,2
90 HLT
91 HME
92
```

Program n° 5 : balls separation

This program separates overlapping discs in a picture. It uses the "Ultimate erosions" program (n° 3) and reconstruction of particles starting from the ultimate erosions ; This reconstruction is performed with a skeleton program (Ring Logic is used).

Algorithm :

Let X be a population of particles (almost convex). Some of them may overlap (figure below)



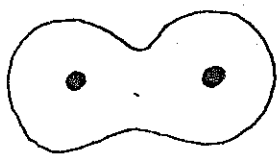
We saw that it is possible to mark each particle with its ultimate erosion. It is also possible to separate overlapping particles. But, it is not possible to skeletonize the complementary set of the Ultimate erosion to obtain the boundary because, in this case, these boundaries would be wrong. The reason is that all the ultimate erosions do not appear at the same moment (see example on hexagonal grid).

Let us see, with an example, the different steps of the algorithm (figures a to f) :

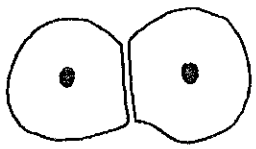
On figure a, the initial set, the ultimate erosions, and the various erosions are drawn.

We determine, first, the maximum size of erosion. This maximum size is the size for which the erosion is equal to ϕ. (On the example, it is 5). We denote it by I.

Then, starting from the erosion of size i ($0 \leq i < I$), we reconstruct the eroded set of size i-1. But this reconstruction uses a skeleton algorithm, so that the number of connected components of the reconstructed set is equal to the number of connected components of the "markers" set. (figure below)



Blow-up reconstruction



Skeleton reconstruction

In order to preserve the boundaries obtained in the previous steps, we replace, at step j+1, the markers by the result of step j.

The great advantage of this method is that each component of the ultimate erosions begins graving at the right moment. (see figures a to f).

Listing comments

A fast erosion subroutine is included (Subroutine ER, instructions 2 to 18).

- Subroutine ER

The initial and the eroded sets are in memory 3. The size of erosion is given by I1. We can write :

$$I1 = 8\,k + r \quad (0 \leq r < 8)$$

So we make k erosions of size 8 and one of size r.

- MAIN program

- instructions 22 to 23

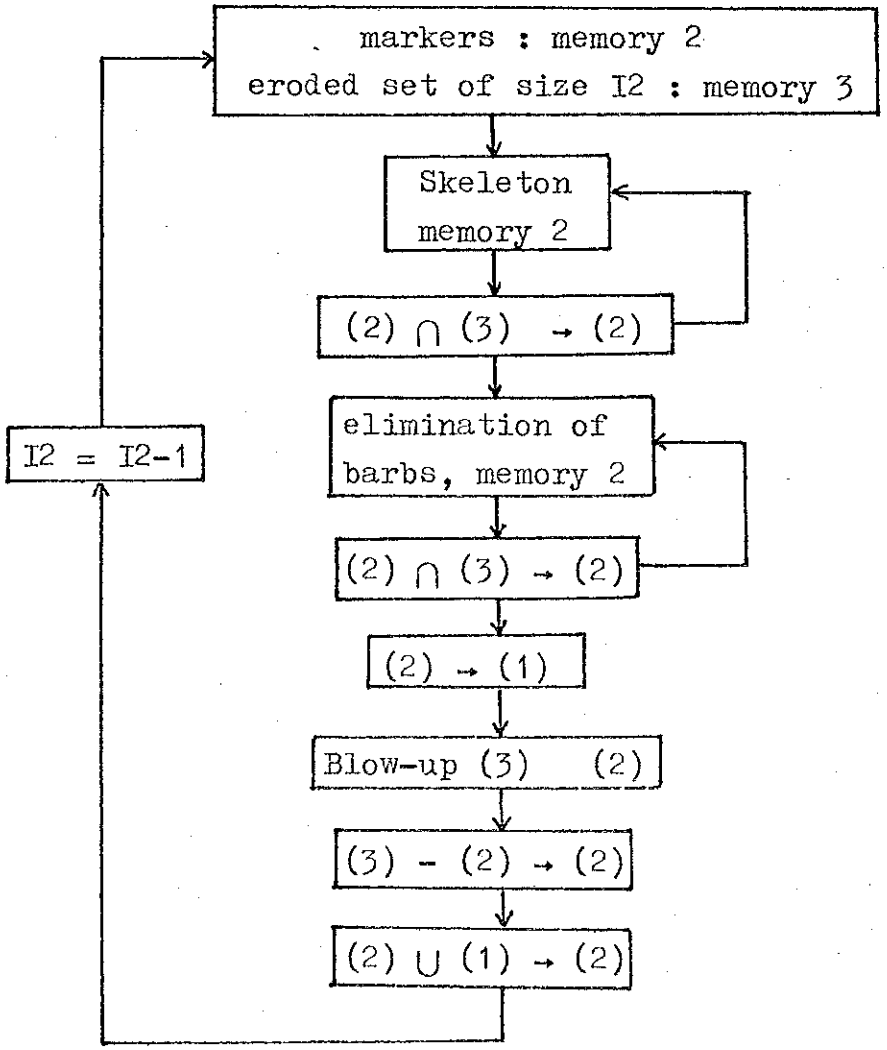
The initial set is stored in memory 4.

- instructions 24 to 32

Calculation of the size of the maximal erosion, I2 is equal to I-1 (I, maximum erosion)

- instructions 34 to 88 : Loop REConstruction

The flow-chart is the following :



- instructions 89,90

Final result of segmentation in memory 2.

- Skeleton

For this part of the program, see program n° 6.

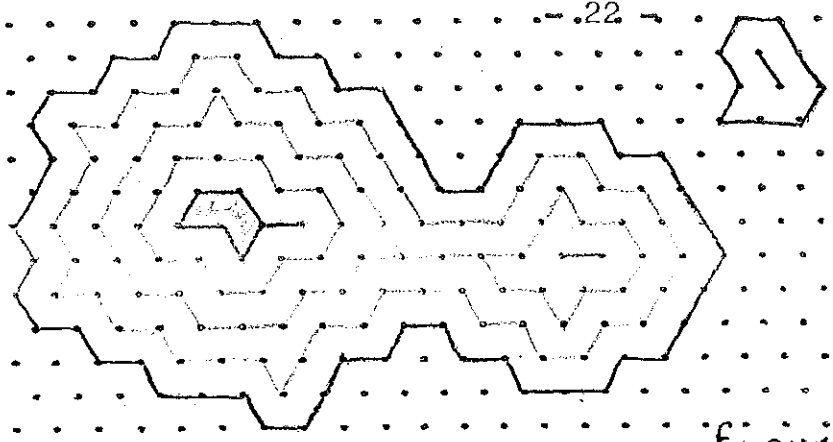


figure a : initial set

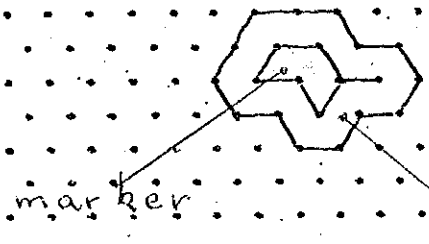


figure b : step 1

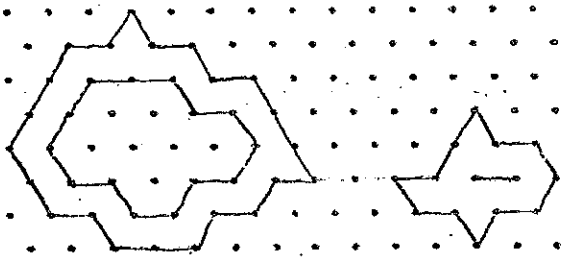


figure c : step 2

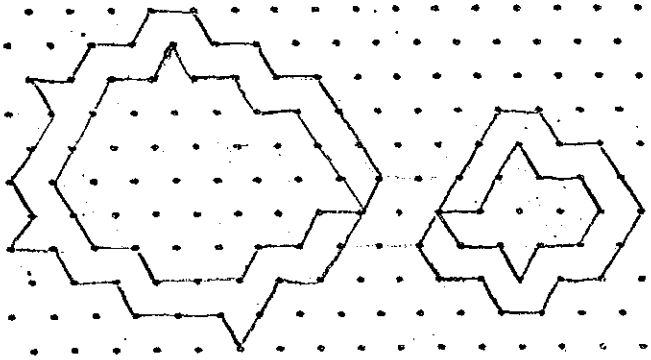


figure d : step 3

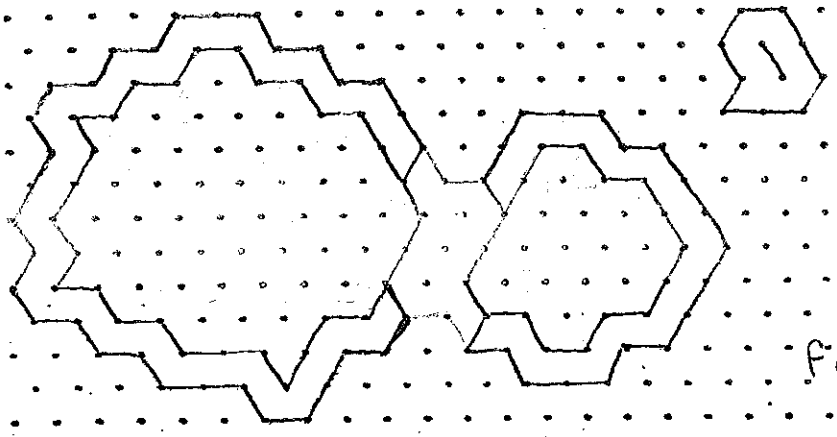


figure e : step 4

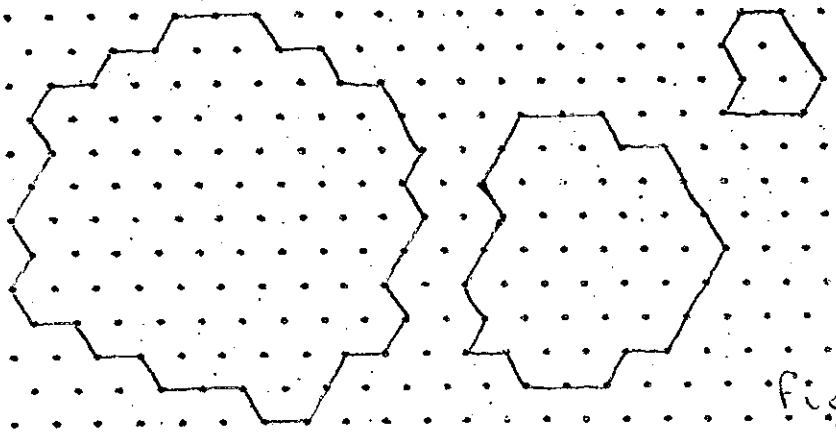


figure f : final Result

```
 1 PNT('    SKELET2')
 2 DLM2000
 3 GTOMAIN
 4 LBLRL
 5 DEF7,3,0,1
 6 ASG2,2,2
 7 LBLA
 8 FNCR2=R1,I8=I8+1
 9 DLSP1,I5=1,6
10 RNGI1,I2,0,128
11 ARER1
12 MOV0,1,1
13 FNCI3=I1/32,I4=I2/32
14 FNCI1=2*I1-63*I3,I2=2*I2-63*I4
15 DLEP1
16 IFCR1-R2,+,+,A
17 CLD
18 CRI
19 RSB
20 LBLMAIN
21 PCT
22 SYR1
23 ASG2,2,4
24 ARER1
25 MOV0,1,1
26 INP('$WHICH SKELETON ? ')I6,I7
27 FNCI1=27,I2=3,I8=0
28 IFCI6,+,B,+
29 FNCI1=23,I2=7
30 LBLB
31 GSBRL
32 FNCI8=I8*I6+I8-1
33 IFCI8,E,E,+
34 FNCI1=15,I2=0
35 IFCI7,E,C,+
36 LBLD
37 CMR3
38 FNCI8=I8-1
39 DLSP2,I5=1,6
40 DEF7,3,0,1
41 ASG2,2
42 RNGI1,I2
43 MOV0,1,1
44 CLD
45 CRI
46 ACX3,0,3,4
47 MOV0,1,1
48 FNCI3=I1/32,I4=I2/32
49 FNCI1=2*I1-63*I3,I2=2*I2-63*I4
50 DLEP2
51 ACX2,3,2,4
52 MOV0,1,1
53 IFCI8,E,E,D
54 LBLC
55 GSBRL
56 LBLE
57 DSG2,2
```

```
58 HLT
59 HME
60
```

Program n° 6, n° 7 : Skeleton with ring logic

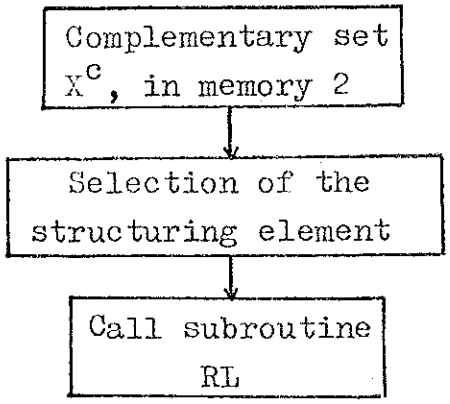
Warning ! : The correct version of this program is version n° 7 (Listing included). Program n° 6 must be destroyed.

Let X be the set to skeletonize. To avoid edge effects, this program works on the complementary set X^c.

To perform a skeleton with ring logic, we can use two structuring elements :

Algorithm :

The algorithm is described below :



The six directions of the Ring are successively used and the points detected at each step are added to the previous set :

Set $Y = X^c$; then Ring Logic, direction 1. The set Z_1 is added to Y : $Y = Y \cup Z_1$. Then, we use the ring logic in direction 2 and so on. (See example on the hexagonal raster)

Barbs of the skeleton

Three modes of elimination are possible :

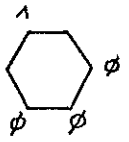
- no elimination
- complete elimination
- suppression proportional to the number of rotations of the skeleton ring logic

Listings comments

The subroutine RL (Ring Logic) is used for all ring logic operations (Skeleton and barbs elimination)

Subroutine RL : (instructions 4 to 19)

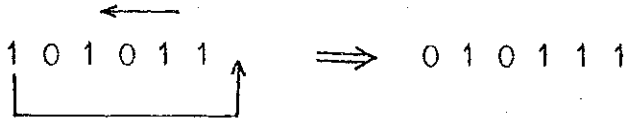
We use two parameters I1 and I2 to program the ring configuration. For example, we have :



I1 = 1 0 1 0 1 1 = 43

I2 = 1 0 0 0 1 1 = 35

The rotation of the ring element is obtained by shifting I1 and I2 :



It is possible to make the union $Y \cup Z_i$, for each direction i, in one instruction :

Y, value of center point	Z_i, Result of Ring	$Y \cup Z_i$
0	0	0
1	0	1
0	1	1
1	1	1

$Y \cup Z_i$ is equal to 1 if the result of the ring is 1, and equal to the value of the center point if not. Thus the ring instruction is :

R N G I1, I2, ∅ , 128

Sides not used ——↑ ↑—— 00|10|000000

Result of ring

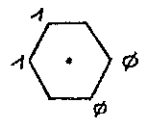
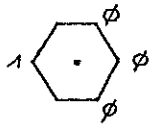
The ring is performed in memory (2) ((2) = Y)

- instructions 9 to 15

Loop P1 : The six directions are successively taken into account. The calculation of the new values of I1 and I2 is made in the instructions 13 and 14. I8 is the number of rotations.

Main program

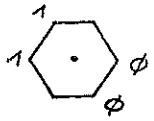
The two parameters I6 and I7 select the skeleton ring and the mode of elimination of barbs :

I6 = ∅	I6 = 1
	

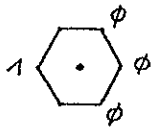
I7

- 1 : No elimination
0 : Complete elimination
1 : Proportional to I8

- instruction 27

I1 = 27, I2 = 3 011011 / 000011 

- instruction 29

I1 = 23, I2 = 7 010111 / 000111 

- instructions 36 to 53

When I7 = 1, we can no longer use the subroutine RL to suppress barbs. This subroutine, indead, would suppress too many points. For instance, if the barb is like this : ,RL subroutine will suppress 3 points and give :

So, we must use the ring logic in permutation mode ; to be sure to suppress one and only one point per rotation. We need 3 memories :

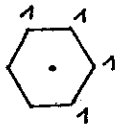
memory (2) for the initial set

memory (0) for the result of the ring in each direction

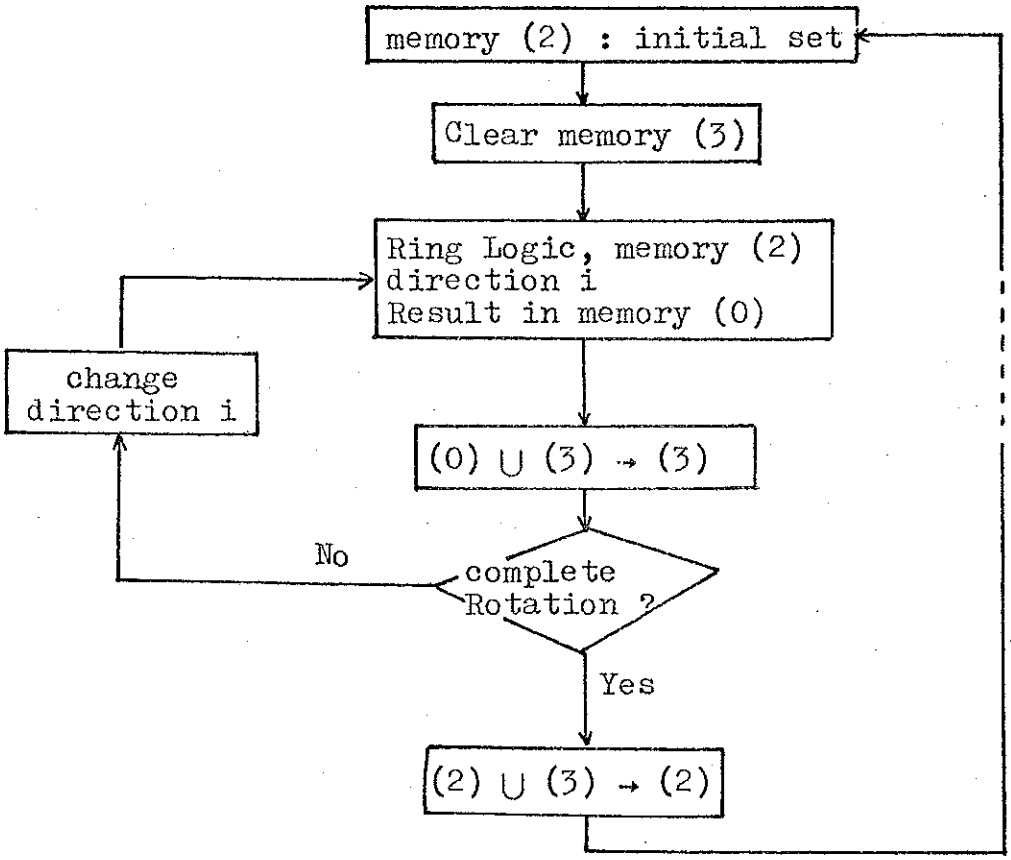
memory (3) for adding the different results of the ring in the six directions.

The ring configuration is given in instructions 34 and 42

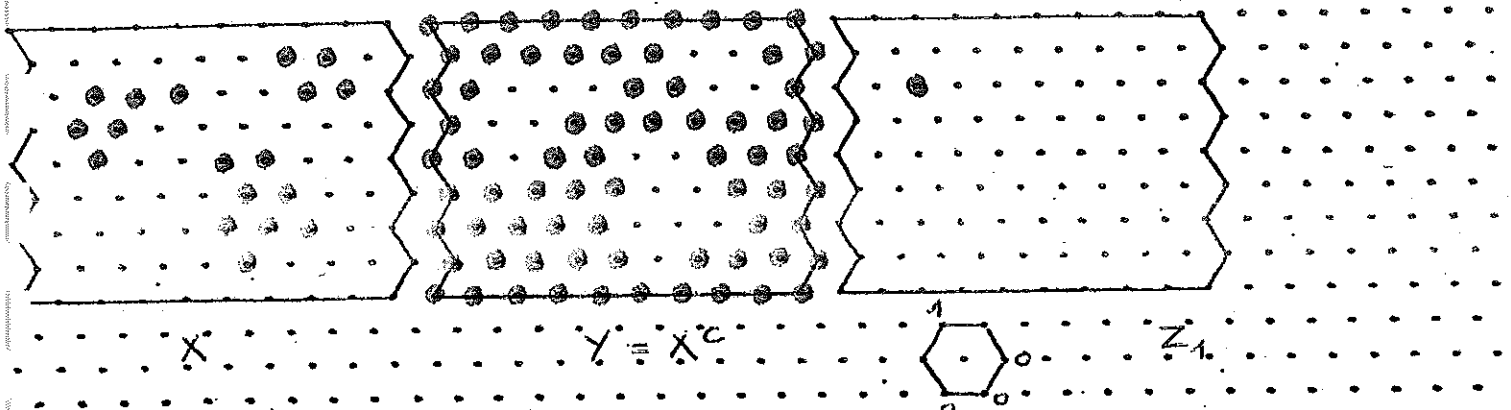
RNG I1, I2 I1 = 15, I2 = 0



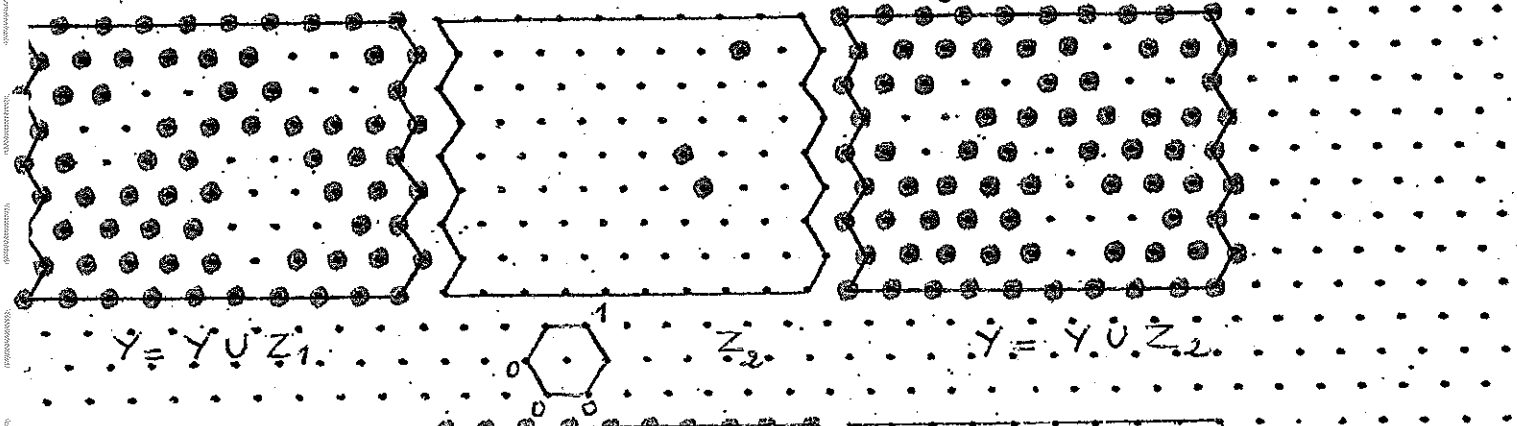
The flow-chart of this part of the program is the following :



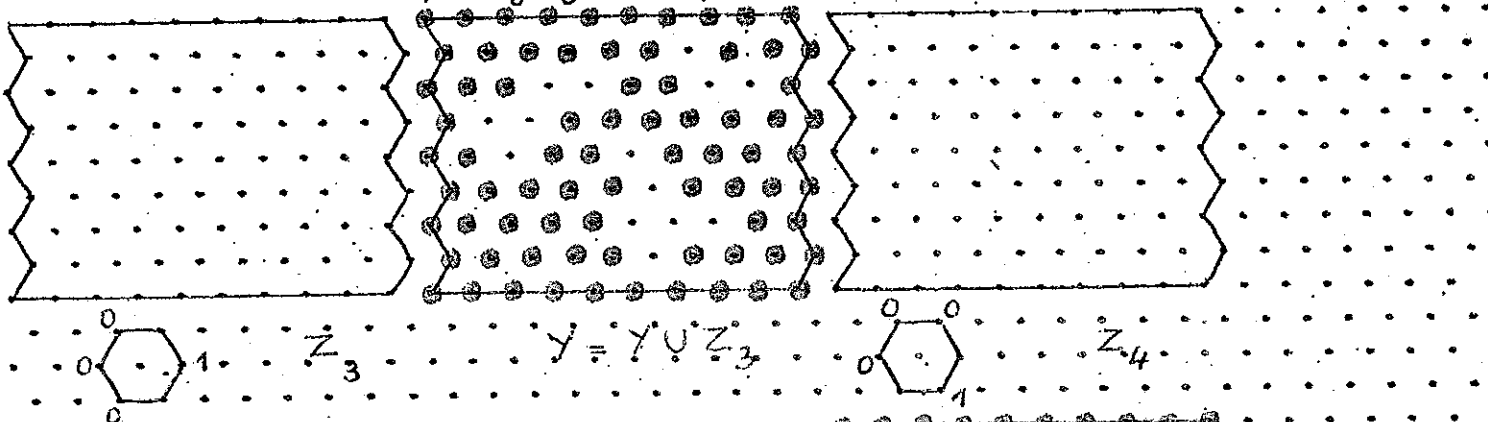
Attention : Before program run, check the content of memory (2) and invert it if necessary (program working on the complementary phase).



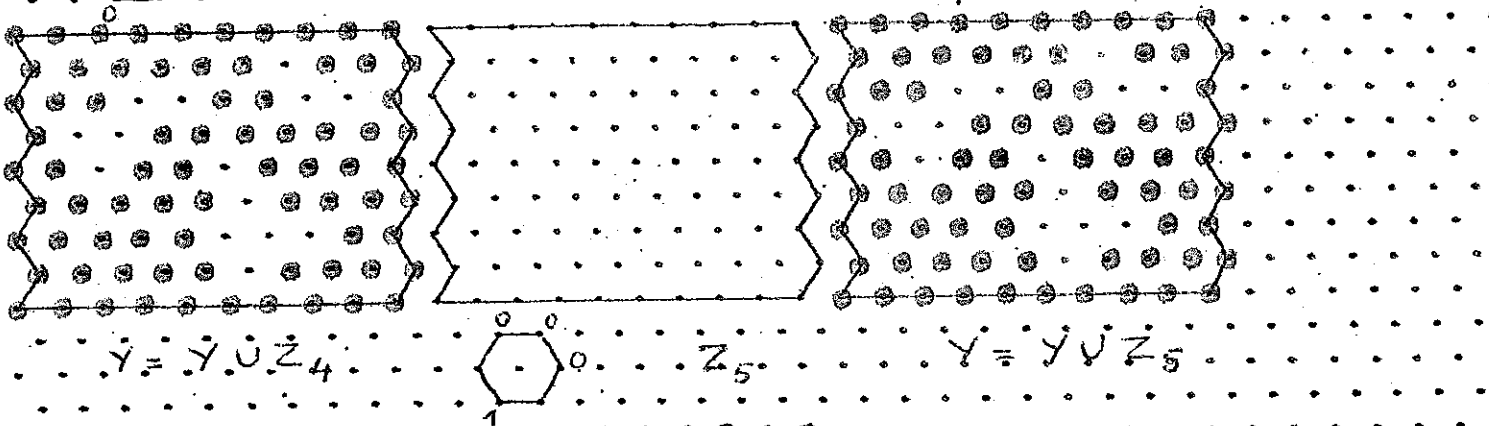
X $Y = X^c$ Z_1



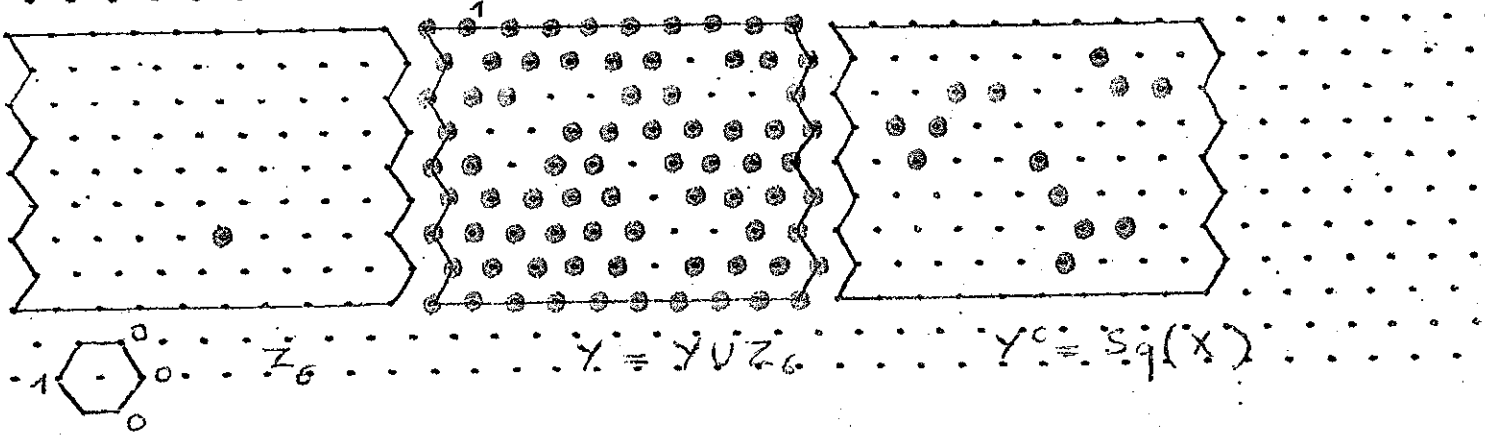
$Y = Y \cup Z_1$ Z_2 $Y = Y \cup Z_2$



Z_3 $Y = Y \cup Z_3$ Z_4



$Y = Y \cup Z_4$ Z_5 $Y = Y \cup Z_5$



Z_6 $Y = Y \cup Z_6$ $Y^c = Sq(X)$

```
  1 PGE -1
  2 WRT 7,TITL1
  3 WRT 7,TITL2
  4 WRT 7,COMM
  5 FRM TITL1,('       PICTURE SEGMENTATION BY')
  6 FRM TITL2,('       CONDITIONAL BISECTRIX',/)
  7 FRM COMM,('        C.M.M. - FONTAINEBLEAU',/,'                ( JUNE 1978 )
  8 DLM 1500
  9 GTO MAIN
 10 LBL DIL
 11 ASG 3,2,3
 12 LBL 2
 13 IFC I7-16,3,+,+
 14 DEF 7,3,3,8
 15 MOV 0,1,1
 16 FNC I7=I7-16
 17 GTO 2
 18 LBL 3
 19 CLD
 20 IFC I7-8,4,+,+
 21 DEF 7,1,3,8
 22 MOV 0,1,1
 23 FNC I7=I7-8
 24 GTO 3
 25 LBL 4
 26 IFC I7,5,5,+
 27 DEF 7,1,3,I7
 28 MOV 0,1,1
 29 LBL 5
 30 CLD
 31 RSB
 32 LBL MAIN
 33 PCT
 34 FNC I3=0
 35 INP ('$ EROSION SIZE ? ')I1
 36 INP ('$ DILATATION SIZE ? ')I2
 37 INP ('$ DILATATION SHIFT ? ')I6
 38 CMR 0
 39 SYR 1,0
 40 ASG 2,2,4
 41 MOV 0,1,1
 42 LBL CB
 43 FNC I3=I3+1,I7=I2
 44 DEF 7,1,0,I1
 45 ASG 2,2,3
 46 MOV 0,1,1
 47 GSB DIL
 48 ACX 2,3,3,3
 49 MOV 0,1,1
 50 FNC I7=I3-I6-1
 51 GSB DIL
 52 DEF 7,1,0,1
 53 ASG 2,2,2
 54 ARE R1
 55 MOV 0,1,1
 56 CLD
 57 ACX 3,0,0,4
```

```
 58 MOV 0,1,1
 59 IFC R1,+,+,CB
 60 ASG 4,2,3
 61 MOV 0,1,1
 62 DCX 3,0,3,0
 63 HLT
 64 FNC R7=0
 65 LBL 7
 66 FNC R8=R7
 67 DEF 7,3,1,2
 68 ASG 0,2,1
 69 MOV 0,1,1
 70 CLD
 71 DEF 7,1,3,1
 72 ACX 1,0,1,3
 73 MOV 0,1,1
 74 ACX 0,1,0,3
 75 MOV 0,1,1
 76 CLD
 77 ACX 0,3,0,0
 78 ARE R7
 79 MOV 0,1,1
 80 IFC R7-R8,+,+,7
 81 FNC I2=0
 82 LBL 8
 83 RCT 14,1
 84 DEF 7,3,0,1
 85 ASG 0,2,0
 86 FNC R9=R8,I2=I2+1
 87 RNG "55,"41,"0,"200
 88 MOV 0,0,1
 89 RNG "33,"3,"0,"200
 90 MOV 0,0,1
 91 RNG "66,"6,"0,"200
 92 MOV 0,0,1
 93 RNG "55,"14,"0,"200
 94 MOV 0,0,1
 95 RNG "33,"30,"0,"200
 96 MOV 0,0,1
 97 RNG "66,"60,"0,"200
 98 MOV 0,0,1
 99 CLD
100 CRI
101 ACX 0,3,0,0
102 ARE R8
103 MOV 0,1,1
104 IFC R8-R9,+,+,8
105 RCT 14,1
106 DEF 7,3,0,1
107 ASG 0,2,0
108 DLS CLIP,I1=1,I2
109 RNG "74,"0,"0,"200
110 MOV 0,0,1
111 RNG "71,"0,"0,"200
112 MOV 0,0,1
113 RNG "63,"0,"0,"200
114 MOV 0,0,1
115 RNG "47,"0,"0,"200
116 MOV 0,0,1
117 RNG "17,"0,"0,"200
118 MOV 0,0,1
119 RNG "36,"0,"0,"200
120 MOV 0,0,1
121 DLE CLIP
122 CLD
123 CRI
```

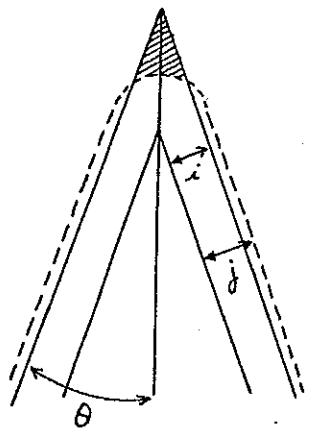
```
124 ACX 0,3,0,0
125 MOV 0,1,1
126 DSG 0,2,0
127 HLT
128 INP ('$ TYPE 1 IF NEW PICTURE          REQUESTED,0 IF NOT    ')I1
129 IFC I1-1,+,MAIN,-
130 HME
131
```

Program n° 8 : Picture segmentation by conditional bisectrix

This program of segmentation of particles uses a conditional bisectrix algorithm.

Conditional bisectrix

Let us consider an angle 2θ in a set X



An erosion of size i, followed by a dilation of size j gives a new set Y :

$$Y = (X \ominus i\ B) \oplus j\ B$$

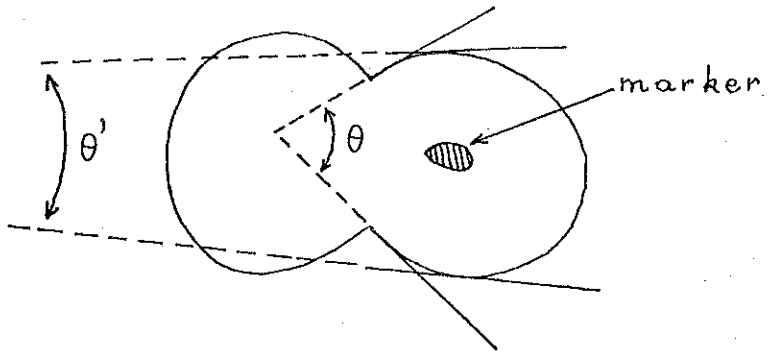
We can see that (X-Y) is different from the empty set if we have :

$$j < \frac{i}{\text{Sin } \theta} \Rightarrow \text{Sin } \theta < \frac{i}{j}$$

This transformation marks every angle which is less than $\frac{1}{2}$ Arc Sin $(\frac{i}{j})$.

Separation of particles

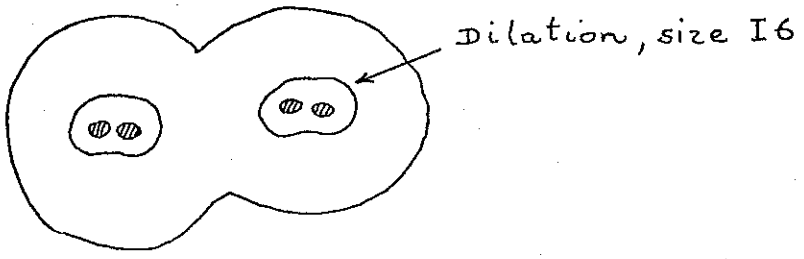
Consider two overlapping particles :



For proper values of i and j, the conditional bisectrix marks the centers of the particles. We have :

$$\theta' < \frac{1}{2} \text{ Arc Sin } \left(\frac{i}{j}\right) < \theta$$

Each marker may be disconnected. So, to avoid this, each marker is dilated in the program. The size of dilation is entered at the beginning of the program (parameter I6).



Starting from the dilated marks, the particles are reconstructed by a skeleton : first a fast skeleton using openings, and then the classical one (Ring logic skeleton, and barbs elimination).

Listing comments

Explanations of the program are given on the listing ; the "erosion size" I1, and "dilation size" I2 are the two parameters of the conditional bisectrix.

This program has two major drawbacks :

- It uses parameters, so we must find the right values of these parameters. Furthermore, these values depend upon the size of the particles. So this method can be used only for particles having the same size.

- The reconstruction is not controlled (as it is in the segmentation by ultimate erosion).

```
 1 PGE -1
 2 WRT 7,TITL1
 3 WRT 7,TITL2
 4 WRT 7,COMM
 5 FRM TITL1,('       PICTURE SEGMENTATION BY')
 6 FRM TITL2,('       CONDITIONAL BISECTRIX',/)
 7 FRM COMM,('        C.M.M. - FONTAINEBLEAU',/,'                ( JUNE 1978 )
 8 DLM 1500
 9 GTO MAIN ---------------------------- FAST DILATION SUBROUTINE
10 LBL DIL
11 ASG 3,2,3 <------- image in memory 3 , result in 3
12 LBL 2
13 IFC I7-16,3,+,+
14 DEF 7,3,3,8 <------ Dilation of size 16
15 MOV 0,1,1
16 FNC I7=I7-16
17 GTO 2
18 LBL 3
19 CLD
20 IFC I7-8,4,+,+
21 DEF 7,1,3,8 <------ Dilation of size 8
22 MOV 0,1,1
23 FNC I7=I7-8
24 GTO 3
25 LBL 4
26 IFC I7,5,5,+
27 DEF 7,1,3,I7
28 MOV 0,1,1
29 LBL 5
30 CLD
31 RSB ------------------------------- MAIN PROGRAM
32 LBL MAIN
33 PCT <------------ Draw Picture in memory 2
34 FNC I3=0
35 INP ('$ EROSION SIZE ? ')I1
36 INP ('$ DILATATION SIZE ? ')I2   } Parameters input
37 INP ('$ DILATATION SHIFT ? ')I6
38 CMR 0 <---------- Clear (Ø)
39 SYR 1,0
40 ASG 2,2,4 <------- Save initial set in memory 4
41 MOV 0,1,1
42 LBL CB
43 FNC I3=I3+1,I7=I2
44 DEF 7,1,0,I1
45 ASG 2,2,3 <------- érosion size I1 , memory(2) ; Destination(3
46 MOV 0,1,1
47 GSB DIL <-------- call subroutine DIL ; Dilation size I2
48 ACX 2,3,3,3
49 MOV 0,1,1 <------- (2) - (3) ⟶ (3)
50 FNC I7=I3-I6-1
51 GSB DIL <-------- DIL ; Dilation size (I3-I6-1)
52 DEF 7,1,0,1
53 ASG 2,2,2
54 ARE R1 <--------- erosion size 1 , memory(2)
55 MOV 0,1,1
56 CLD
57 ACX 3,0,0,4 <----- (3) ∪ (0) ⟶ (0)
```

```
MOV 0,1,1
IFC R1,+,+,CB
ASG 4,2,3
MOV 0,1,1
DCX 3,0,3,0
HLT
FNC R7=0
LBL 7
FNC R8=R7
DEF 7,3,1,2
ASG 0,2,1
MOV 0,1,1
CLD
DEF 7,1,3,1
ACX 1,0,1,3
MOV 0,1,1
ACX 0,1,0,3
MOV 0,1,1
CLD
ACX 0,3,0,0
ARE R7
MOV 0,1,1
IFC R7-R8,+,+,7
FNC I2=0
LBL 8
RCT 14,1
DEF 7,3,0,1
ASG 0,2,0
FNC R9=R8,I2=I2+1
RNG "55,"41,"0,"200
MOV 0,0,1
RNG "33,"3,"0,"200
MOV 0,0,1
RNG "66,"6,"0,"200
MOV 0,0,1
RNG "55,"14,"0,"200
MOV 0,0,1
RNG "33,"30,"0,"200
MOV 0,0,1
RNG "66,"60,"0,"200
MOV 0,0,1
CLD
CRI
ACX 0,3,0,0
ARE R8
MOV 0,1,1
IFC R8-R9,+,+,8
RCT 14,1
DEF 7,3,0,1
ASG 0,2,0
DLS CLIP,I1=1,I2
RNG "74,"0,"0,"200
MOV 0,0,1
RNG "71,"0,"0,"200
MOV 0,0,1
RNG "63,"0,"0,"200
MOV 0,0,1
RNG "47,"0,"0,"200
MOV 0,0,1
RNG "17,"0,"0,"200
MOV 0,0,1
RNG "36,"0,"0,"200
MOV 0,0,1
DLE CLIP
CLD
CRI
```

← Branch CB, if content of (2) ≠ ∅

← Result of conditional bisectrix

SKELETON WITH OPENINGS

← closing size 2, memory (0) → (1)

← Dilation, size 1 of $(1) \cap (0)^c \rightarrow (1)$

← Dilation, size 1 of $(0) \cap (1)^c \rightarrow (0)$

← $(0) \cap (3) \rightarrow (0)$

← return lbl 7 if size increased

SKELETON with RING LOGIC
memory (0)

← $(0) \cap (3) \rightarrow (0)$

BARBS ELIMINATION
memory (0)

```
124 ACX 0,3,0,0
125 MOV 0,1,1
126 DSG 0,2,0
127 HLT
128 INP ('$ TYPE 1 IF NEW PICTURE          REQUESTED,0 IF NOT     ')I1
129 IFC I1-1,+,MAIN,-
130 HME
131
```