ACTA STEREOL 1990; 9/1: 61-69
PROC ECS 5 FREIBURG I. BR. 1989
ORIGINAL SCIENTIFIC PAPER

# TIPS ON MORPHOLOGICAL RECONSTRUCTION

Serge Ramon (*) and Yves Bringer (**)

(*) Misis Image 10D rue de la Productique St Etienne

(**) Equipe Traitement Image, Laboratoire T.S.I., 23 rue Dr.Michelon,

42023 St Etienne . FRANCE

## ABSTRACT

During the development of a basic library on the parallel architecture board PC-OEIL, using "Data Flow" processors (Ramon,87), a special customer requirement appeared. Because fast image workstations already exist, processing only binary images, most image programmers, using forced binarization, prompted us to develop a granulometry unit.

Besides the different shape parameters (area, perimeter, Feret diameter, fractal dimension approximation of contour according to Chermant & Coster method, etc...) (Chermant,89), the primary tool for such an individual analysis is the recovery of each object. The aim of this paper is the description of the particular tips implemented around the classical morphologic method of conditional reconstruction from the first point of the object. Other methods may be as efficient (contour extraction...), but we decided to concentrate upon the morphological reconstruction.

We would like to point out the fact that improvements presented in this program are not especially dedicaded to "Data Flow" architecture, but on the contrary, classical high level language programming will take fool advantage of them. Saving in execution time from several hundred to one most of the time, up to 512 to one, refered to the Basic algorithm, makes this program an original and efficient tool for automatic individual analysis. As a matter of fact, for some objects, two raster scannings are sufficient for complete reconstruction, independently of their size.

Keywords : Morphological reconstruction, Data Flow, Automatic individual analysis.

## PC-OEIL BOARD OVERVIEW

The extension board, "PC_ŒIL", transforms an IBM-PC compatible micro computer into an efficient image processing workstation. In addition to giving all traditional facilities for scanning, storing and displaying images, this workstation reaches a 20 mips output, thanks to 4 NEC µPD 7281 data - flow processors, organized in a ring (NEC,1985). It is different from boards with specialized processors, such as masks, which will inevitably be short of the specific operator the user is opened to need.

Two storage configurations can be selected by software :

* four 8-bits images of 512x512 pixels

* two 8-bits images of 512x512 pixels and 16 binary images of 512x512 pixels.

The "data-flow" concept, originated in the late 1960s in the United States, encapsulates a software model and a hardware architecture, closely related and mutually dedicated, so as to build high performance parallel multi-processors machines.

The advantage of this system over specialized boards for image processing is that the organization of its image processing unit is not dedicated to a unique task.

Each data-flow processor, plays the part of a specialized processor in traditional boards, but its actual role will depend on the way it will have been "configured" in the system. More precisely, we will load in the data-flow processor a real program to specify the processing it will have to execute.

This extension board, in addition to provide a wealth of unconventional programming facilities, offers an interesting compromise between real time, hardware based, and costly image treatments, and software based programs whose performances are not convenient for industrial applications.

## LIBRARY'S FACILITIES

In order to allow the user to build his own user-defined processing, a library of one hundred primitives has been developed. Each primitive can be called from any high level language as a typical subroutine of the host program. Furthermore it is possible to develop applications using a mouse and pull-down menus to call the primitives individually and perform a step-by-step elaboration of user-defined analysis and techniques. The following execution times show up the board's capabilities :

Grey Level

| | |
|---|---|
| - 3x3 Convolution | : 800 ms |
| - Constant Image Creation | : 200 ms |
| - Multiplication by a constant value | : 400 ms |
| - Affin Transform (Homothetis + Rotation + Translation) | : 1900 ms |

Binary

| | |
|---|---|
| - 3x3 | : 70 ms |
| - Inversion | : 30 ms |
| - Contour extraction | : 75 ms |

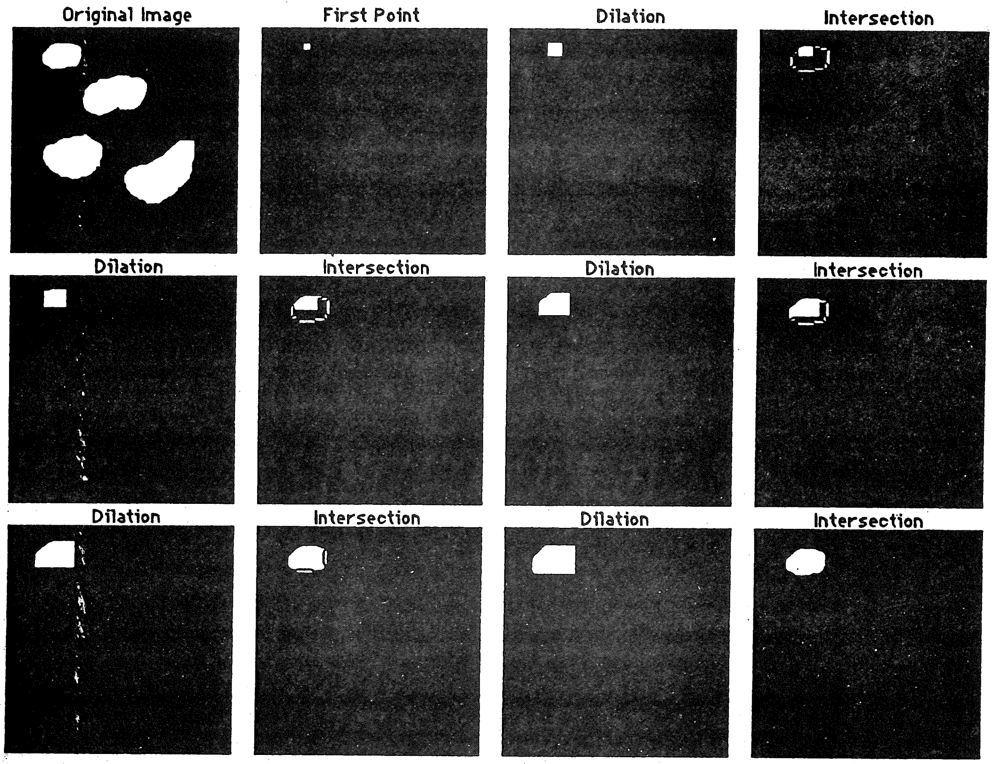
Among the existing primitives, we can find any operation needed by an individual analysis : Dilation, Intersection, Area process, and First Point extraction. Consequently, we are able to write the individual analysis program using theses facilities :

```
Program Analisys;
Var Plane1, Plane2 : Byte;
    GlobalArea, Aire1, Area2: Real;
BEGIN
   Plane1 := 1;
   Plane2 := 2;
   LoadBinImage(Plane1,'ImageToTreat');
   Area(Plane1,GlobalArea);
   WHILE GlobalArea <> 0 DO BEGIN
                                  (* First Object Detection *)
                                  ClearBinImage(Plane2);
                                  FirstPoint(Plane1,Plane2);
                                  Area1:=0;
                                  Area(Plane2,Area2);
                                  REPEAT
                                     Area1:=Area2;
                                     DilationBin(Plane2,Plane2,1);
                                     Intersection(Plane1,Plane2,Plane2);
                                     Area(Plane2,Area2);
                                  UNTIL Area1=Area2;
(* Eventual parameters extraction from current isolated object in Plane2 *)
                                  SetDifference(Plane1,Plane2,Plane1);
                                  Area(Plane1,GlobalArea);
                               END;
END.
```

We should note that the real Pascal program is written word for word like the above one, showing there is no need to be a data-processing expert to be able to write any imagery program.

Nevertheless, as it was said before, any subroutine's calling needs the loading of the special processor's program. Time needed for the extraction of a 100 pixels diameter disk is of 70 seconds.

So, we get the following images :



As the extraction time is too long when we have to treat too many objects, it becomes interesting to create a new program for the µPD 7281, which has to isolate the first object.

## FIRST OBJECT PROGRAM OPTIMIZATION

### Inconvenience due to the dilation

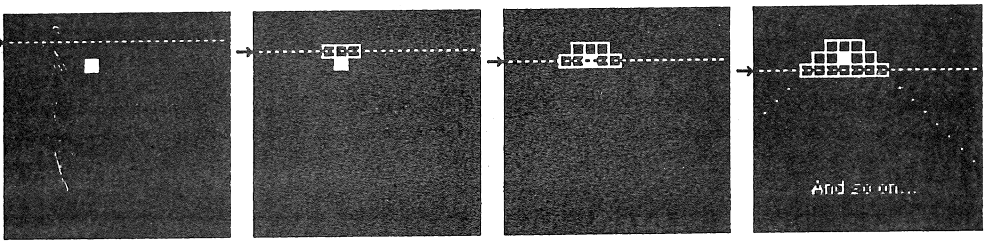
First of all, we are going to try and explain how the dilation program is built. The only way offered by the Basic library is a one-step dilation (BasicDilation program) from a plane to one which must be different from the first one. The possibility chosen in the last program use a work plane :

```
PROCEDURE DILATIONBIN(SourcePlane,DestPlane,N : BYTE);
Var WorkPlane : Byte;
BEGIN
   CopyPlanBin(SourcePlane,WorkPlane);
   FOR  I:=1 TO n DIV 2 DO BEGIN
                              DilationBase(WorkPlane,DestPlane);
                              DilationBase(DestPlane,WorkPlane);
                          END;
    IF n  MOD  2 = 0  THEN CopyPlanBin(WorkPlane,DestPlane);
                      ELSE DilationBase(WorkPlane,DestPlane);
END;
```

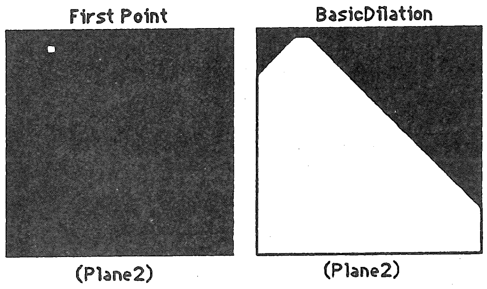
Note that the specialized processors of this board do not dispose of enough buffer storage memory to allow the technology of the delay line, often used for convolutions. The source plane will be read three times; Three vertical pixels are read (in fact, 3x16 because binary images are stored as one point per bit, and Read/Write operations use 16 bits-word) and constitute with the last 6 pixel read (2x3 vertical pixels) the 3x3 mask which will permit to determine central pixel value after dilation.

## Conditional dilation

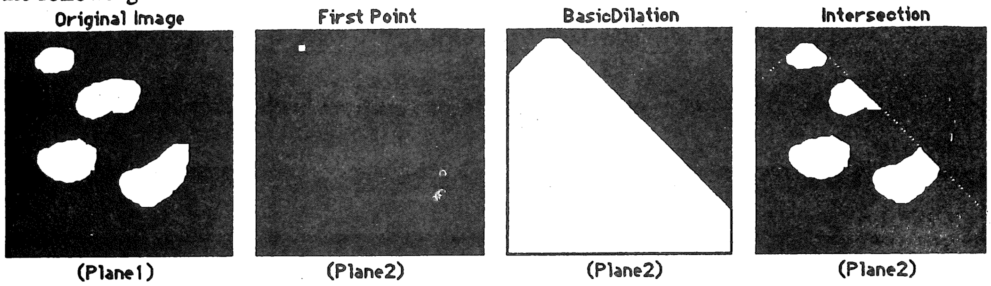
Needing two different planes for the dilation represents the most important problem to build the First Object program. As a matter of fact, the processors are not well adapted to manage the flip-flop between WorkPlane and Destination plane. Fortunatly, a simple remark gives us the solution : let's see what happens when we ignore the forbidden dilation on the same plane; a resulting point is processed thanks to its 8 neighbours : the 3 neighbours of the upper line result from the dilation, but on the contrary the other neighbours are really the original pixels we have to dilate; consequently, the dilation phenomenon is going to propagate and create a pyramid from the dilated pixel:



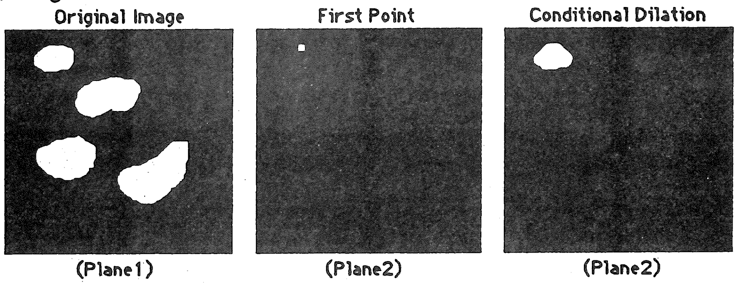
So we have :



This note gives us a piece of solution, but it is not able to solve itself the whole problem; as a matter of fact when we replace the BinDilation subroutine by the BasicDilation one, we get the following results :



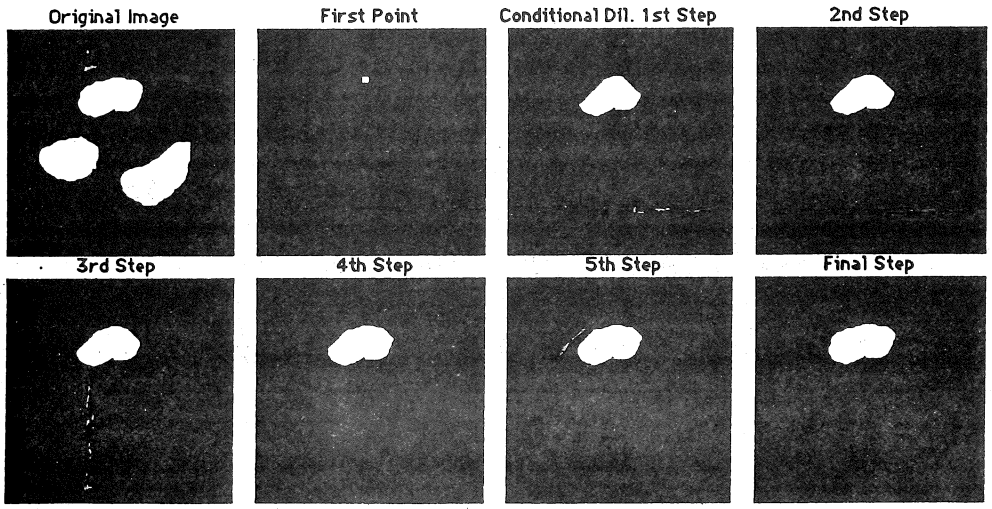
The solution will appear from the BasicDilation subroutine, because we will process the intersection during dilation result writting instead of performing a global intersection after the dilation. We'll just have to read the source plane (Plane1) and perform an And operation between the pixel resulting of the dilation and the original one, just before writting the result. Because of the pipelined architecture of the processors, reading the original image (Plane1) doesn't increase notably the execution time of the Basic dilation in order to obtain a conditional dilation. So, we get :



We should note up to now that the reconstruction of an object included in a cone needs only one pass.

## Stop Test

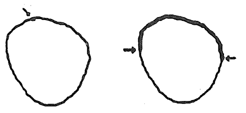
To branch out of the loop when stability is reached, we will have to perform a stop test during the result image writting. We'll only use a boolean, true initialised at each pass, and which becomes false when a resulted dilated point is different from the point to be dilated; so we have :



## First Point Efficiency

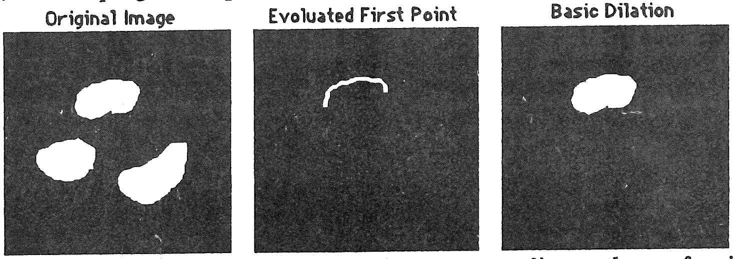
The saving of time is yet appreciable especially for the large objects, but an extension of the first point detection will give us the possibility to build every convexes with only one pass.

Regarding the great number of points written with one pass from the first point, it was very useful to replace the first point by the part of the object viewed from the upside of the image.



This program consists of an horizontal and downward propagation from the first point. We'll take advantage of the board's parallelism using two processors to perform right and left propagations.

For example, let's explain the right propagation mechanism. Propagating horizontaly from the first point, we will recopy the one-point on the marker plane (Plane2). When we meet a zero-point, we restart the process from the point under the last one-point found, as long as its value is one, else the program stops.



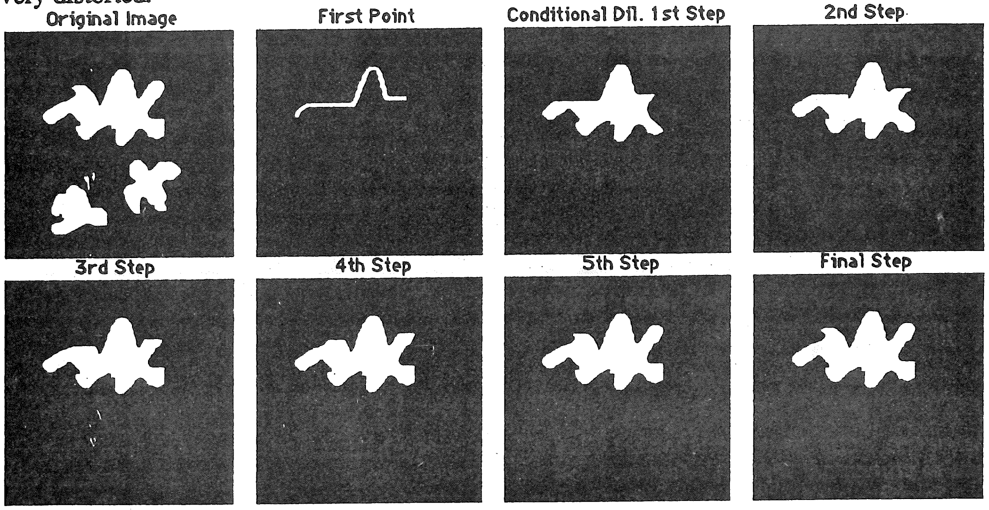
We must note that this improvement is quite easy to realize, and very fast in execution time, so that it can't be compared at all with any extraction coutour method.

## Non Convexes Problem

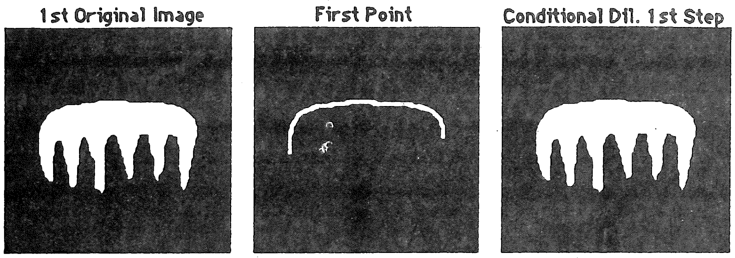
We are able up to now to reconstruct every convex with one pass, even the white plane, which is the worst case for the classical reconstruction method.

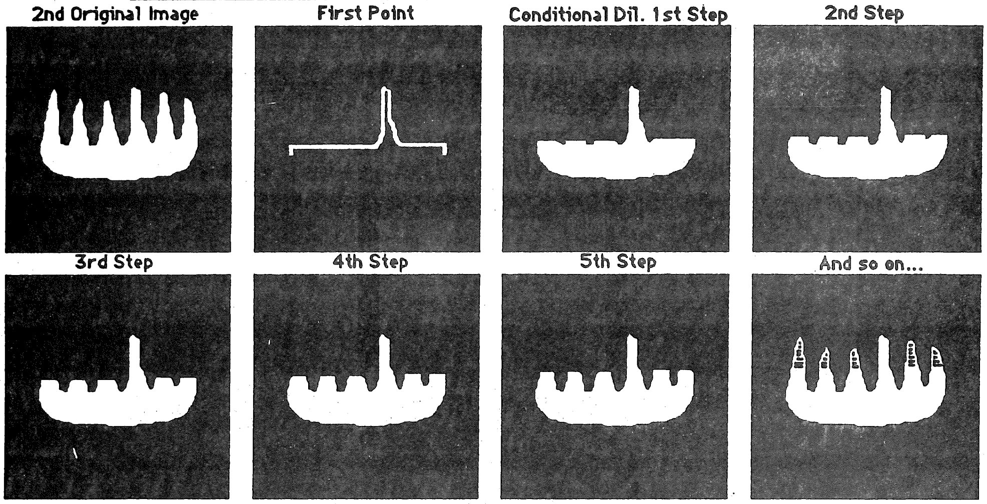
It is clear that a single method can't be the best one for every classes of objects. The isolation of a particle should better be depedent to its shape's complexity.

In particular, the previous program looses its efficiency with any object whose contour is very distorted.



Observing the phenomenal difference of execution time between the two following images, we can guess the right solution

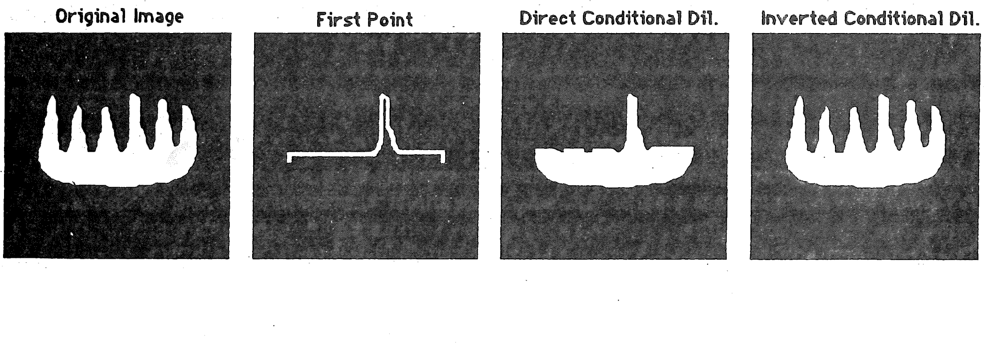




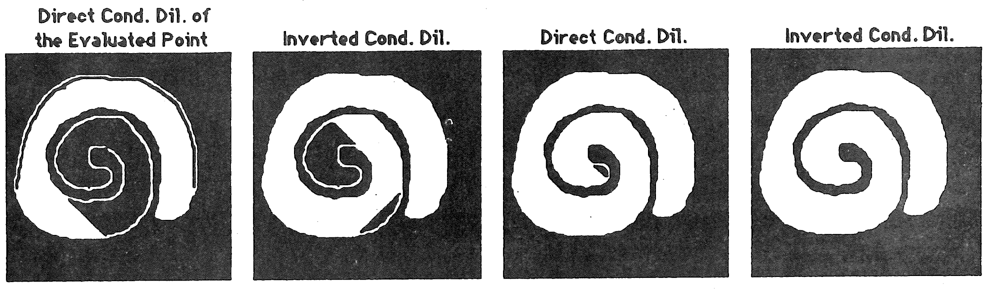
We can note that the non-convex object of the first image is either reconstructed in one pass.

The first idea coming to the mind is the image rotation, we should effectively make and rotate of 180° the resulting image at each pass, taking care of rotating the original image too. This method may be advantageously replaced by inverted Read/Write. That is to say instead of scanning the images line by line and from left to right, which starting point is the left upper corner point and the stopping one is the right downer point, we will scan the images always line by line, but from right to left, the starting point being the right downer corner point and the stopping one the left upper point.

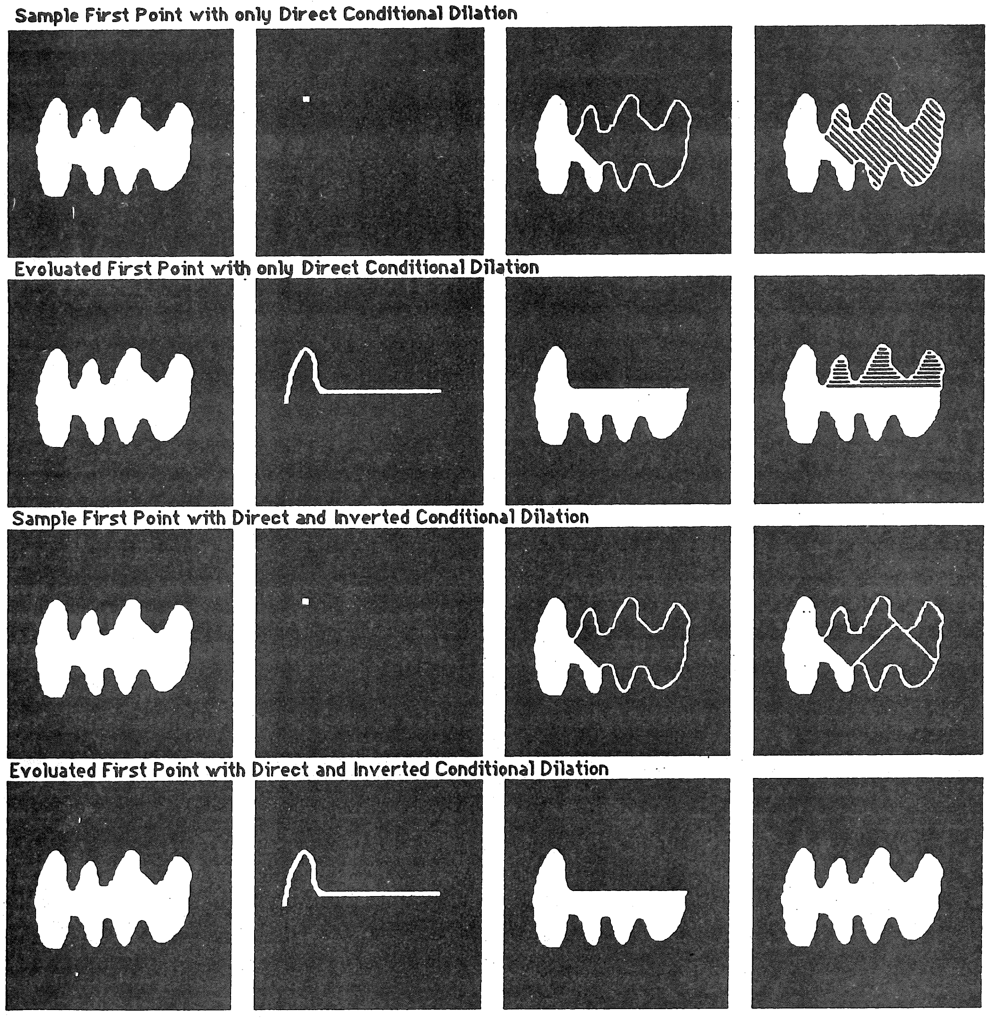
So we get :



So, the problem is solved for the comb-like shape, but the difficult problem concerning snail-like shape still remains :



Note : The inverted Read/Write is not a useless employment with the first new point.


Sample First Point with only Direct Conditional Dilation
Evoluated First Point with only Direct Conditional Dilation
Sample First Point with Direct and Inverted Conditional Dilation
Evoluated First Point with Direct and Inverted Conditional Dilation

## CONCLUSION

As we divide the execution time by 100 on average, this program offers besides, if we suppress the first point, very high performances for other programs such as Closing Holes, Detection of the objects on the edges, and reconstruction with a marker.

## REFERENCES

Barreault G,Rivoire A,Jourlin M,Labouré MJ,Ramon S,Zeboudj R, Pinoli JC, Automated Image Processing : an efficient pipeline data flow architecture, "4th International Symposium on Opto-Electronic applied science and engineering" 1987 La Haye.
Chermant JL and Coster M, Précis d'Analyse d'Image, Editions du CNRS, 1989.
Chong YM, A Data Flow Architecture for Digital Image Processing, Natick Technology Center nov. 1984.
Chong YM, Data Flow Chip Optimizes Image Processing, Computer Design oct. 1984.
Meshach W, Data Flow IC makes short work of tough processing chores, Electronic Design may 1984.
Nec Product Description, µPD7281 (Image Pipelined Processor) and µPD 9305 (Memory Access and General Bus Interface Chip).
Ramon S and Rivoire A ,A new generation board for image processing, Congress ISS 1987, Caen.