REFERENCE HANDBOOK

MICROMORPH®

(COMPOSED WORDS)

Reference Manual

MICROMORPH for WINDOWS

Composed operations

Preface

The structure of Micromorph allows the use of all the previously compiled functions and procedures as primitives in the other ones. Nevertheless, it is possible to classify them in three pedagogical categories:

i) primitive words
ii) the composed operations
iii) the tutorial exercises on Mathematical Morphology

This manual, deals with the second level and is valid for the Windows version of Micromorph. The operations are classified by subjects. Their index placed at the beginning of the manual. For each procedure (function), three modes are indicated, according to the following convention:

grid:

0 : the function works only in square grid
1 : the function works only in hexagonal grid
1, 0 : the function works both in hexagonal and square grids
[0], 1 : the function changes the grid, to the hexagonal one, while working (and returns to initial grid when finishes)
0, [1] : the function changes the grid, to the square one, while working (and returns to initial grid when finishes)

edge:

0: the function works only in intrinsic or transitive modes
1: the function works only in standard mode
1, 0 : the function works in both modes
[0], 1 : the function changes the mode to "1" while working (and returns to the initial one when finishes)
0, [1] : the function changes the mode to "0" while working (and returns to the initial one after finishes)

depth:

1: function applies to binary (1bit) images
8: function applies to greytone (8bit) images
16: function applies to greytone (16bit) images

The primitive words which assign the modes *grid* and *edge* are named **imsetgrid** and **imsetedge** (see reference manual - part 1).
The correspondences between the present terminology and the one from the MS-DOS version, as well as the alphabetical index of all functions and procedures described in the manual are given at the end of this manual.

Index of procedures:

1 - UTILITIES

2 - DISPLAY TOOLS

- EROSIONS AND DILATIONS

3-1 basic erosions and dilations

dil, ero	dilation, erosion, by a square or hexagon
dirdil, direro	dilation, erosion by a segment
minidil, miniero	dilation, erosion by a little square or triangle
distance	distance function
dbldil, dblero	dilation, erosion by a pair of points
cont, sq4cont	internal contour of an image (field)
gradient, sobel	morphological gradient and Sobel gradient

3-2 other dilations and erosions

isodil, isoero	isotropic dilation, erosion
isodist	isotropic distance function
conedil, cyldil	dilation by a cone, a cylinder
crossdil	dilation of a set by another set
rhombodil, rhomboero	dilation, erosion by a rhombododecagon
diamdil, diamero	dilation, erosion by a diamond
ringdil1, ringdil2	dilation by a ring or by the summits of an hexagon
(intermediary routines)	polygonal dilations

4 - RANK OPERATORS

hexrank	hexagonal rank operator
median	median filter
binsegmi	intersection of dil. by segments

5 - CONVOLUTIONS

vgauss1, hgauss1, gauss1	gaussian filters (vertical, horizontal and isotropic of size 1)
gauss	gaussian filter of size n

6 - OPENINGS, CLOSINGS

6-1 basic openings, closings

open, close	opening, closing by dil and ero
diropen, dirclose	opening, closing by dirdil and direro
lineopen, lineclose	union (intersection) of the above
openth, closeth	top-hat by opening, closing
lineopenth, linecloseth	top-hat by lineopen, by lineclose

6-2 other openings, closings

isopen, isoclose	isotropic opening, closing
infopen, supclose	limit of inf (sup) of directional openings (closings)
miniopen, miniclose	opening, closing by minidil, miniero
regrad, pregrad	regularized gradients
stopen	inf of directional openings

7 - GEODESY AND CONNECTIVITY

gdsdil, gdsero	geodesic dilation, erosion
build	binary reconstruction
buildopen, buildclose	opening, closing by reconstruction
areaopen	opening according to the area
ringbuild	reconstruction by a ring
gdsdist	geodesic distance
recons	binary reconstruction (similar to build)
levelling	levelling transform

8 - APPLICATIONS OF GEODESY

edgeoff	deletes the grains that touch the border
fgrain	extracts the first particle
border	internal contour of the field
clohole	filling the internal pores
minima, maxima	extrema of a function
extmaxima, extminima	extended extrema
swamping	modification of the homotopy by markers
extrema	extrema of an image inside a mask
grainclose	grain smoothing and closing
indivaf	individual closing of particles
dynamics	dynamics of the maxima or minima of an image

9 - FILTERS

9-1 main filters

af	alternating filter
asf, fullasf	alternating sequential filter (2 types of progression)
lineaf, lineasf	alternating sequential filter with lineopen
automed, centre	automedian operator, morphological centre
contrast	contrast for isoero-isodil, isopen-isoclose
contrasth	contrast for top-hat

9-2 other filters

minifilt	mini alternating filter
isaf, isasf	isotropic alternating filter, alternating sequential filter
buildaf, buildasf	alternating filter, alternating sequential, by reconstruction
closoropen	toggle between opening and closing
minibuildaf	filter by reconstruction using miniopen and miniclose
binmiddle	binary median set
grmiddle	grey median image

10 - MAXIMAL BALLS AND SKELETON

binopenskel	skeleton by maximal squares (hexagons)
binultim	ultimate erosion

centroid	centroid (last eroded set)
condbis	conditional bisector

11 - THINNINGS, THICKENINGS

11-1 binary

(gds)thickturn, (-)thinturn	basic (geodesic) thinning, thickening, cycle
(gds)thick, (gds)thin	thinning, thickening (until idempotence)
Dthick, Lthick, Mthick	homotopic thickening
Dthin, Lthin, Mthin	homotopic thinning
endpoints	ending points
multpoints	multiple points
gdscentre	geodesic centre

11-2 greytone

greyseero, greysedil	erosion, dilation by se
greythickstep, greythinstep	basic thickening, thinning
dirgradient	directional gradient
vectgrad0	complete coarse gradient
gradvect	true vector gradient
mainthin	thinning and clipping
greyhmt	greytone hit-or-miss
greythickturn	greytone basic thickening
greythick	greytone thickening

12 - WATERSHED TRANSFORMATION

clip	clipping
(gds)skiz	skeleton by influence zone
threshwshed	watershed by thresholding
mwshed, wshed	watershed, with or without markers

13 - SEGMENTATION

mgradwshed, gradwshed	watershed, with or without markers, of gradient
shapeseg	segmentation by distance function
smoothcnc, jumpcnc, jump	contrast extraction
mosaic	mosaic image
wfall	waterfall transformation
kheops	pyramid of mosaic images

14 - MEASURES

14-1 basic measures

diameter	diameter variation
digperim, perim	digital and Euclidean perimeters
cnumber	connexity number
binh(v)ferret	Ferret's diameter

14-2 other measures

var0, var1, var2	variances of order 0, 1 and 2
rug	roughness
mse	mean quadratic difference
bincount	number of connected components
flatcount	number of flat zones
flatzone	area of the flat zones

15 - CURVES

bincov	binary covariance
vario1, vario2	order 1 and 2 variograms
modcont	module of continuity
isogranul , granul	granulometry
covar	covariance
pvonl	measure of the binary linear erosion
p1vonl	binary linear granulometry in number
p2vonl	binary linear granulometry in measure

16 - RANDOM SIMULATIONS

16-1 boolean simulations

point, points, regpoints	random points
isobool, isobool2	simulation of Boolean isotropic sets
elbool, tribool, dropbool	simulation of Boolean anisotropy sets
rose, hard, disjoint, flake	simulation of hierarchies
rocky	rock bases

16.2 lines and partitions

lines, diags	Poisson lines
nestlines	Hierarchy of Poisson lines
steps	Poisson strips

17. GRAPHS

Gero, **Gdil**	erosion, dilation on graphs
Gopen, **Gclose**	opening, closing on graphs
setborder	outlines borders in a grey image
Gbuild	planar graph reconstruction
label	generates a label image

18. 3DUTILITIES

Seqload	loads a sequence
Seqsave	saves a sequence
Seqclr	clears a sequence
Seqset	sets a sequence to a given value
Seqcopy	copies a sequence
Seqinv	inverts a sequence

Seqadd	adds all the images of a sequence
Seqcadd	adds a constant value to a sequence
Seqseqadd	adds two sequences
Seqcsub	subtracts a constant value from a sequence
Seqseqsub	subtracts two sequences
Seqcmul	multiplies a constant to a sequence
Seqmean	mean image of a sequence
Seqseqmean	computes the mean of two sequences
Seqdiff	module of the difference between two sequences
Seqvolume	volume of a sequence
Seqinf	inf a all the images of a sequence
Seqseqinf	inf between two sequences
Seqsup	sup of all the images of a sequence
Seqseqsup	sup between two sequences
Seqbuild	reconstruction of a sequence from a marker sequence
Seqthresh	sequence grey thresholding
Seqgrad	gradient of a sequence
Seqhist	histogram of a sequence
Scroll, Scroll2, **Scroll3**	sequence scrollings
Visu	perspective display

19. 3D PROCEDURES

ero3D, dil3D	3D erosion and dilation
open3D, **close3D**	3D opening, closing
gradient3D	3D gradient
build3D	3D reconstruction
buildopen3D, **buildclose3D**	3D opening and closing by reconstruction
openth3D, **closeth3D**	3D top-hats
maxima3D, **minima3D**	3D extrema
af3D, **asf3D**	3D alternate sequential filters
buildaf3D, **buildasf3D**	3D AS filters by reconstruction
timdil, **timero**	linear erosion and dilation along the time axis
timopen, **timclose**	linear opening and closing along the time axis

20. MOUSE

fill	draws and fills a polygon
dline	draws lines
delobj	removes connected components
pointinfo	coordinates and grey value of a point
binpointer	points and extracts a particle
objinfo	area of a pointed particle
brush	brush function

CORRESPONDENCE
between MSDOS and Windows MICROMORPH operators

MSDOS version	Windows version		
Names	Names	chapters	modifications
Ch. 2			
Nbinlinero, Ngreylinero	direro	2-1	+
Nbinlindil, Ngreylindil	dirdil	2-1	+
binerode, greyerode	ero	2-1	+
Nbinerode, Ngreyerode	ero	2-1	+
bindilate, greydilate	dil	2-1	+
Nbindilate, Ngreydilate	dil	2-1	+
bintriero, greytriero	miniero	2-1	+
bintridil, greytridil	minidil	2-1	+
binb1dil, greyb1dil	id, id	2-2	0
binb2dil, greyb2dil	id, id	2-2	0
bindbero, greydbero	id, id	2-2	0
bindbldil, greydbldil	id, id	2-2	0
Ch. 3			
Nbingen, Ngreyopen	open	2-3	+
Nbiclose, Ngreyclose	close	2-3	+
Nbinlinopen, Ngreylineopen	diropen	2-3	+
Nbinlinclose, Ngreylineclose	dirclose	2-3	+
Ngreysupopen	lineopen	2-3	+
Ngreyinfclose	lineclose	2-3	+
Ch. 4			
bininterc	diameter	11-1	+
binnumber	cnumber	11-1	+
binperim1	digperim	11-1	+
binperim	perim	11-1	+
binperim2	*removed*		
Nbincovalc	bincov	11-4	++
Nbingranul	granul	11-4	++
binbuild	id(= fast build)	4-2	0
binvbuild	*removed*		
Ch. 5			
Nwtophat	openth	3-1	+
Nbtophat	closeth	3-1	+
NSupopen th	lineopenth	3-1	+
Ninfcloseth	linecloseth	3-1	+
sobel	sobel	3-1	0
Ngradient	gradient	3-1	+
Npreggrad, Nnpreggrad	*removed*		
Ch.6			
bingdsdil, Nbingdsdil	gdsdil	4-1	+
greygdsdil, Ngreygdsdil	gdsdil	4-1	+

bingdsero, Nbingdsdil	gdsero	4-1	+
greygdsero, Ngreygdsero	gdsero	4-1	0
binborder	border	3-1	+
binedgeoff	edgeoff	5-1	
bincloholes, greycloholes	cloholes	5-1	+
binindiana	areaopen	4-2	+

Ch. 7

binultim	id	5-1	+
binopenskel	id	3-2	+
maxima, minima	id, id	5-2	+
extmaxima, extminima	id, id	5-2	+
erodistance	isodist	2-2	++
Nbbuildopen, Ngbuildopen	buildopen	4-2	+
Ngbuildopen, Ngbuildclose	buildclose	4-2	+
threshmax	*removed*		

Ch. 8

greyseerode, greysedilate	greyseero, greysedil	7-2	0
greythick, greythin	greythickstep, greythinstep	7-2	0
bincontour	cont	3-1	+
dirgradient,vectgrad0	id, id	7-2	0
gradvect, gradpalette	id, id	7-2	0
masksup(equal), greymask	id, id	1	0
bin(gds) thin, bin(gds)thick, sedual	*removed*		

Ch. 9

binsk1turn, binsk2turn	thinturn, thickturn	7-1	++
binskel1, binskel2	thin, thick	7-1	++
binLskel1, binMskel1, binDskel1	Lthin, Mthin, Dthin	7-1	+
binLskel2, binMskel2, binDskel2	Lthick, Mthick, Dthick	7-1	+
binendpoints	endpoints	7-1	+
binmulpoints	mulpoints	7-1	+
bingdscenter	gdscentre	7-1	+
bingsk1turn, bingsk2turn	gdsthinturn, gdsthickturn	7-1	+
bingdskel1, bingdskel2	gdsthin, gdsthick	7-1	+
binbclipall	clip	7-1	++
bin(gds)skiz	(gds)skiz	7-1	+
binhferret	id	11-2	0
binvferret	*removed*		

Ch.10 - Ch.11 -Ch.12

Threshwshed	id	9-1	0
Nedcontrast, Noccontrast, Noc5contrast	contrast	6-2	+
Nthcontrast	contrasth	6-2	0
wasf, basf	fullasf	6-1	+
wlinasf, blinasf	lineasf	6-1	+
wbuildasf, bbuildasf	buildasf	6-1	
automed	automed	6-2	++
iterautomed	*removed*		

1. UTILITIES

PROCEDURE **clr**

SYNTAX **clr** im

SUBJECT Clears the image *im*

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

FUNCTION **inside**

SYNTAX v := (binin1 **inside** binin2)

SUBJECT Returns 1 if *binin1* is "inside" *binin2*, otherwise returns 0

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1***

SEE ALSO **inferior**

PROCEDURE **clean**

SYNTAX **clean** greyin threshold_level greyout

SUBJECT Points that are lower or equal to *threshold_level* are replaced by 0.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

FUNCTION **inferior**

SYNTAX v := (imin1 **inferior** imin2)

SUBJECT Returns 1 if *imin1* is inferior or equal to *imin2*, otherwise returns 0.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

SEE ALSO **inside**

PROCEDURE **div2**

SYNTAX **div2** greyin greyout

SUBJECT Puts in *greyout* the image *greyin* divided by 2

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

FUNCTION **div**

SYNTAX v := (a **div** b)

SUBJECT Divides a by b. Rounds to the closest integer.

PROCEDURE **mean**

SYNTAX **mean** greyin1 greyin2 greyout

SUBJECT Puts in *greyout* the arithmetic mean of *greyin1* and *greyin2*

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

FUNCTION **immean**

SYNTAX v := **immean** in

SUBJECT Calculates the mean value of the pixels from the image in.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8***

PROCEDURE **incrust**

SYNTAX **incrust** greyin binmaskin greyout

SUBJECT Insert *greyin* into the zone of the *greyinout* image defined by *binmask*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1*** *and* ***8, 16***

PROCEDURE **bintogrey**

SYNTAX **bintogrey** binin greyout

SUBJECT Creates in *greyout* a greytone image. Abbreviation for

"**immask** *binin* **impixmin** *greyout* **impixmax** *greyout greyout*"
where **:** **"impixmin** *greyout"* = 0 or - 32767
and : **"impixmax** *greyout"* = 255 or 32767

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1*** *and* ***8, 16***

PROCEDURE **translate**

SYNTAX **translate** imin imout signx x signy y

SUBJECT Puts into *imout* the image *imin* translated by the vector (signx x ; signy y). The outside of the in image is equal to 0. Directions are coded as follows:

right : signx = 3 ; up : signy = 1 ; left : signx = 7 ; down : signy = 5

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

REMARK The translation vector is defined on the square grid.

PROCEDURE **s4rotate**

SYNTAX v := **s4rotate** structuring_element

SUBJECT Returns the code of the *structuring_element* after rotation by 90 degrees.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1***

REMARK Note the difference with 8 rotations of the analogous procedure **serotate;** here, there are only 4 possible rotations.

SEE ALSO **serotate**

PROCEDURE **power**

SYNTAX v := (a **power** b)

SUBJECT Computes a^b (a and b are integer values).

FUNCTION **mod**

SYNTAX **v := (**a **mod** b)

SUBJECT Returns the rest of the integer division of a by b.

EXEMPLE Function **mod** is useful for incrementing rotations from any origin. In hexagonal grid, for example, the sequence :

```
int i a ;
i := 2
for 1 to 6 do
  a := (( i mod 6 ) + 1 )
  ++ i
end
```

returns successively the six values 3, 4, 5, 6, 1, 2

PROCEDURE **immul**

SYNTAX **immul** imin1 imin2 imout

SUBJECT Multiplies *imin1* by *imin2*, pixel by pixel, and places the result in *imout*. *Imout* must have 16 bit.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16*** *and* ***16***

PROCEDURE **masksup**

SYNTAX **masksup** greyin1 greyin2 binout

SUBJECT Puts "1" in *binout* if the corresponding point of *greyin1* is greater than the point taken from *greyin2*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16*** *and* ***1***

SEE ALSO **masksupequal**

PROCEDURE **masksupequal**

SYNTAX **masksupequal** grayin1 greyin2 binout

SUBJECT Puts "1" in *binout* if the corresponding point of *greyin1* is greater than or equal to the point taken from *greyin2*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16*** *and* ***1***

SEE ALSO **masksup**

PROCEDURE **greymask**

SYNTAX **greymask** value binin greyout

SUBJECT Creates in *greyout*, the greytone image (*value* * *binin*)

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1*** *and* ***8, 16***

REMARK *Value* must be a positive integer lower then 256 (or 32768)

PROCEDURE **abs**

SYNTAX v := **abs** a

SUBJECT Computes the absolute value of a.

PROCEDURE **greyabs**

SYNTAX **greyabs** greyin greyout

SUBJECT Gives the module of the greyin image. The origin is at 127 for 8-bit images and at 0 for 16-bit images.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

PROCEDURE **format**

SYNTAX **format** vertical horizontal

SUBJECT Initialises a new image format (vertical x horizontal). Allocates 10 binary images (1-bit) b1..b10, 10 greytone images (8-bit): g1...g10, and 6 16-bit images: h1 ...h5. All previously allocated images are destroyed.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

PROCEDURE **ngbnb**

SYNTAX v := **ngbnb**

SUBJECT Returns the number of neighbors on the current grid.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

SEE ALSO **grid**

PROCEDURE **dirtranspose**

SYNTAX v := **dirtranspose** direction

SUBJECT Computes the transposed direction.

PROCEDURE **adjust**

SYNTAX **adjust** greyin binout low_thresh high_thresh

SUBJECT Performs an interactive thresholding between *low_thresh* and *high_thresh* of the *greyin* image with the keypad. The initial thresholds are l*ow_thresh* and *high_thresh.* It is then possible to modify these values by using the 0, 1, 2, 3, 7, 8 and 9 keys of the numeric keypad.

0 : end of interactive thresholding
7 : N steps increase of low_thresh
1 : N steps decrease of low_thresh
9 : N steps increase of high_thresh
3 : N steps decrease of low_thresh
8 : N is increased by 1
2 : N is decreased by 1.
The initial value of N is 1.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

SEE ALSO **imthresh**

PROCEDURE **getccords**

SYNTAX **getcoords fpoint_code x y**

SUBJECT Converts the value *fpoint_code* returned by the fpoint or imcompare functions into x and y coordinates.

SEE ALSO **imcompare, fpoint**

PROCEDURE **gandb**

SYNTAX **gandb** greyin bin greyout

SUBJECT Restricts image *greyin* to set *binin* and places the result in *greyout.*

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

PROCEDURE **stop**

SYNTAX **stop**

SUBJECT Emits a sound to indicate the stop, and waits for a key to be pressed on the keyboard, for continuing the execution of the program.

SEE ALSO **readkey**

PROCEDURE **strech**

SYNTAX **strech** imin imout

SUBJECT Stretches the grey range of imin to the whole range 0-255.

MODES *grid:* ***1,0*** *edges:* ***1,0*** *depth:* ***8***

PROCEDURE **delta**

SYNTAX **delta** imin1 imin2 binout

SUBJECT The two inputs *imim1* and *imim2* may be binary or grey. The procedure places in *binout* the set of points where *imim1* <> *imim2* , and prints the volume of | *imim1* - *imim2* | if *imim1* and *imim2* are greytone images, or the area of their symmetrical difference, if they are binary.

MODES *grid:* ***1,0*** *edge:* ***1,0*** *depth:* ***1,8***

SEE ALSO **imdiff**

PROCEDURE **half**

SYNTAX **half** imin imout type

SUBJECT Reduces two times the size of the image imin and places the result in imout.

if *type* = 0 , input image is minified directly,
if *type* = 1 , an mini erosion (miniero) is performed before the minificaton,
if *type* = 2 , a sequencial filtering is (by miniopen and miniclose) is performed before the minification.
Reduced image occupies the bottom-left corner of the image imout.

MODES *grid:* ***1,0*** *edge:* ***1,0*** *depth:* ***1,8***

REMARK Procedure half intervenes as one step when one wishes to work on an image with a resolution reduced two times. After having used half procedure, the result has to be saved, then the working image format has to be changed (see command format), and finally the saved image has to be loaded in a working window with the new format.

2. DISPLAY TOOLS

PROCEDURE **ds**

SYNTAX **ds** im

SUBJECT Displays the image *im*. When this procedure is used inside another one (to display for instance a working memory), the memory number and the depth of the calling routine are displayed.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

PROCEDURE **dscol**

SYNTAX **dscol** imin1 imin2 zoom

SUBJECT If imin1 and imin2 are binary images, the procedure displays the first one in red and the second one in green. If there is only one binary image, it is displayed in red over the greytone image.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1,8, 16***

REMARK At least one of the two images must be a binary one. The display of the other images is frozen as long as this procedure is active.

PROCEDURE **shadow**

SYNTAX **shadow** greyin greyout

SUBJECT Shadow permits to visualize the binary image as the greytone relief. The procedure works better if slopes are close to one. (Specially prescribed for the distance function).

MODES *grid:* ***[1], 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

REMARK There exists also procedure **shadow2** (different shadow direction).

PROCEDURE **clrcol**

SYNTAX **clrcol** imin

SYNTAX Deactivates the color palette affected at image *imin*. When in is replaced by zero, all color images become greys.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8***

SEE ALSO **col, pscol, dscol**

FUNCTION **col**

SYNTAX [v :=] **col** imin pal

SYNTAX Affects palette *pal* (defined in a file and indicated as a string in inverted commas) to image *imin*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8***

SEE ALSO **pscol, clrcol, dscol**

FUNCTION **pscol**

SYNTAX [v :=] **pscol** imin imout

SUBJECT When input image *imin* is grey, colorises it with the standard palette, when *imin* is binary affects a different color with each connected component of *imin* and places them in the grey image *imout.*

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8***

SEE ALSO **col, clrcol, dscol**

PROCEDURE **cp**

SYNTAX **cp** imin imout

SUBJECT Copy image *imin* with its palette in image *imout*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8***

SEE ALSO **cppal**

PROCEDURE **cppal**

SYNTAX **cppal** imin imout

SUBJECT Affects the palette of image *imin* to image *imout*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8***

SEE ALSO **cp**

PROCEDURE **refresh**

SYNTAX **refresh** *imin*

SUBJECT Places the display of image *imin* above the other displays of images.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1,8,16***

PROCEDURE **profile**

SYNTAX **profile** imin

SUBJECT Takes a section of image imin defined with the mouse and displays the grey profile.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8***

PROCEDURE **coldisplay**

SYNTAX **coldisplay** imin1 imin2 zoom title

SUBJECT If *imin1* and *imin2* are binary images, the procedure displays the first one in red and the second one in green. If there is only one binary image, it is displayed in red over the greytone image. *"title"* is the title written in the displayed window.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1,8***

REMARK At least one of the two images must be a binary one. The display of the other images is frozen as long as this procedure is active

SEE ALSO **dscol**

3. EROSIONS, DILATIONS

3.1 Basic erosions and dilations

PROCEDURE **dil**

SYNTAX **dil** imin imout size

SUBJECT Dilation (binary or greytone - according to the source image), of size *size* by an hexagon or a square (according to the *grid*) in standard or geodesic mode.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

SEE ALSO **ero**

PROCEDURE **ero**

SYNTAX **ero** imin imout size

SUBJECT Erosion (binary or greytone - according to the source image), of size *size* by an hexagon or a square (according to the *grid*) in standard or geodesic mode.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

SEE ALSO **dil**

PROCEDURE **distance**

SYNTAX **distance** binin greyout

SUBJECT Distance function, on square or hexagonal grid, from the binin image (therefore equals to 0 in binin).

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

SEE ALSO **isodist**

PROCEDURE **dirdil**

SYNTAX **dirdil** dir imin imout size

SUBJECT Dilation (binary or greytone) by a segment of size *size* in a direction *dir* in hexagonal or square grid (according to *grid*). Directions are coded as follows:

```
                        6   1                       8   1   2
hexagonal :   5     0     2        square :     7   0   3
                    4   3                       6   5   4
```

where : 0 marks the origin of the structuring element.

REMARK The procedure shifts the image because the origin is not at the centre of the structuring element.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

SEE ALSO **direro**, **diropen** and **dirclose** in OPENINGS

PROCEDURE **direro**

SYNTAX **direro** dir imin imout size

SUBJECT Erosion (binary or greytone) by a segment of size *size* in a direction *dir* in hexagonal or square grid (according to *grid*). Directions are coded as it is described in the **dirdil** procedure

REMARK The shifting behaviour of **dirdil** also applies here.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

SEE ALSO **dirdil**, **diropen** and **dirclose** in OPENINGS.

PROCEDURE **minidil**

SYNTAX **minidil** imin imout

SUBJECT Elementary dilation, binary or greytone (according to the source image). The mini-structuring element consists of three points in the hexagonal grid and four in the square one :

```
                    1                       1   1
hexagonal:  1   1       square:     1   1
```

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

REMARK Note non-centering, to compare with that of miniero.

SEE ALSO **miniero**, **miniopen** and **miniclose** in OPENINGS.

PROCEDURE **miniero**

SYNTAX **miniero** imin imout

SUBJECT Elementary erosion, binary or greytone (according to the source image). Mini - structuring element has three points in the hexagonal grid and four in the square one, formed as follows:

hexagonal: 1 1 square: 1 1
1 1 **1**

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

REMARK Note non-centering, to compare with that of miniero.

SEE ALSO **mindil, miniopen** and **miniclose** in OPENINGS.

PROCEDURE **b1dil**

SYNTAX **b1dil** imin imout

SUBJECT Hexagonal dilation by the element B1:

MODES *grid:* ***1*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **b2dil**

PROCEDURE **b2dil**

SYNTAX **b2dil** imin imout

SUBJECT Hexagonal dilation by the element B2 (see procedure **b1bil**)

MODES *grid:* ***1*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **dbldil**

SYNTAX **dbldil** dir imin imout size

SUBJECT Dilation (binary or greytone - according to the input image) by a pair of points at a distance *size* in a direction *dir*. The directions are coded as described for the **dirdil** procedure.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK The procedure shifts images due to the non-centered structuring element.

SEE ALSO **dblero**, **dirdil**

PROCEDURE **dblero**

SYNTAX **dblero** dirin imin imout size

SUBJECT Erosion (binary or greytone - according to the input image) by a pair of points at a distance *size* in a direction *dir*. The directions are coded as described for the **dirdil** procedure.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK The remark concerning shifts in **dbldil** also applies here.

SEE ALSO **dbldil**, **direro**

PROCEDURE **gradient**

SYNTAX **gradient** greyin greyout size

SUBJECT Beucher gradient, or morphological gradient. Difference between the dilatation **dil** and the erosion **ero (**both of the size *size*)

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

PROCEDURE **sobel**

SYNTAX **sobel** greyin greyout

SUBJECT Sobel gradient.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

PROCEDURE **dilate**

SYNTAX **dilate** imin imout size

SUBJECT binary or greytone fast dilation (but with edge effects) of size *size*, by a square or an hexagon.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1,8, 16***

SEE ALSO **erode**.

PROCEDURE **erode**

SYNTAX **erode** imin imout size

SUBJECT binary or greytone fast erosion (but with edge effects) of size *size*, by a square or an hexagon.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1,8, 16***

SEE ALSO **dilate**.

PROCEDURE **sq4cont**

SYNTAX **sq4cont** binin binout

SUBJECT Internal contour of *binin* into *binout*, square (*connectivity* = 4).

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1***

PROCEDURE **cont**

SYNTAX **cont** binin binout

SUBJECT Internal contour, square or hexagonal, of *binin*, placed in *binout*. This procedure depends on *edge.*

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1***

3.2 Other erosions and dilations.

PROCEDURE **isodil**

SYNTAX **isodil** imin imout size

SUBJECT Isotropic dilation of size *size*, binary or greytone, in square or hexagonal grid. The procedure is dual to **isoero** for the complementary image.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **isoero**

PROCEDURE **isoero**

SYNTAX **isoero** imin imout size

SUBJECT Isotropic erosion of size *size*, binary or greytone, in square or hexagonal grid. The procedure is dual to **isodil** for the complementary image.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **isodil**

PROCEDURE **isodist**

SYNTAX **isodist** binin greyout size

SUBJECT Isotropic distance function. If greyout is a 8-bit image, the distance function is truncated to 255. No tuncation occurs if greyout is a 16-bit image

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1***

PROCEDURE **conedil**

SYNTAX **conedil** binin greyout grey_display size type

SUBJECT Dilation by the cone of hexagonal (type = 0) or dodecagonal (type = 1) base of size *size*. The origin is placed in the middle of the base (for *type* = 0) or at the top of the cone (*type* = 1). The procedure shows *greyout* in the *grey_display*.

MODES *grid:* ***1, [0]*** *edge:* ***1, 0*** *depth :* ***1, 8, 16***

REMARK The erosion procedure is dual to **conedil** and may be constructed by using inverted input image

SEE ALSO **cyldil**, **rhombodil**

PROCEDURE **cyldil**

SYNTAX **cyldil** greyin greyout unit_eight size

SUBJECT Dilation by the cylinder of hexagonal or square base (according to *grid*), of size *size*, and height (*unit_eight* * *size*). The origin is placed in the middle of the base

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

REMARK The erosion procedure is dual to **cyldil** and may be constructed by using the inverted input image.

SEE ALSO **conedil**, **rhombodil**

PROCEDURE **crossdil**

SYNTAX **crossdil** binin1 binin2 binout

SUBJECT Dilation of the set *binin2* by the contour of the set *binin1*, therefore by the set *binin1*. The origin is placed in the left upper corner (co-ordinates: 0, 0)

MODES *grid:* ***[1], 0*** *edge:* ***1, 0*** *depth :* ***1***

REMARK Procedure is faster if *binin* is near by the origin. Automatically switches to the square grid.

PROCEDURE **rhombodil**

SYNTAX **rhombodil** in out height size

SUBJECT Dilation by the rhombodecadon obtained by dilations of four semi diagonal lines of square base (*size***size*) and height = *(height* * size) , centered at the origin.

MODES *grid:* ***[1], 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK Automatically switches to the square grid. Lift image on the distance height.

SEE ALSO **conedil**, **cyldil**

PROCEDURE **rhomboero**

SYNTAX **rhomboero** in out height size

SUBJECT Erosion by the rhombododecagon obtained by dilations of four semi diagonal lines of square base (*size***size*) and a height = (*height* * *size*) , centered at the origin.

MODES *grid: **[1], 0** edge: **1, 0** depth : **8, 16***

REMARK The same as for **rombodil**; in addition, procedure works in two edge modes.

SEE ALSO **rhombodil**

PROCEDURE **bero**

SYNTAX **bero** imin imout

SUBJECT Procedure, dual for the complement, of the product (**b1dil** o **b2dil**)

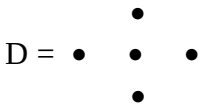
MODES *grid: **1** edge: **1** depth : **1, 8, 16***

SEE ALSO b1dil b2dil

PROCEDURE **diamdil**

SYNTAX **diamdil** imin imout size

SUBJECT Product of the *size* dilations of *imin* by the diamond D:



MODES *grid: **0** edge: **1** depth : **1, 8, 16***

SEE ALSO **diamero**

PROCEDURE **diamero**

SYNTAX **diamero** imin imout size

SUBJECT Product of the *size* erosions of *imin* by the diamond D (see procedure **diamdil**).

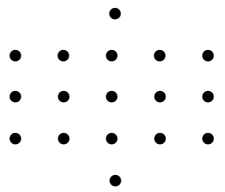
MODES *grid: **0** edge: **1** depth : **1, 8, 16***

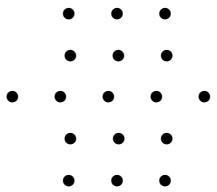
REMARK The procedure is dual to **diamero** for complementary image.

PROCEDURE **sh1dil**

SYNTAX **sh1dil** imin imout

SUBJECT Dilation by the hexagon SH1 :

SH1 = 

SH2 = 

MODES *grid: **0** edge: **1** depth : **1, 8, 16***

PROCEDURE **sh2dil**

SYNTAX **sh2dil** imin imout

SUBJECT Dilation by the hexagon SH2 (see procedure **sh1dil**).

MODES *grid: **0** edge: **1** depth : **1, 8, 16***

PROCEDURE **sh1ero**

SYNTAX **sh1ero** imin imout

SUBJECT Erosion dual to **sh1dil** for the complementary image.

MODES *grid: **0** edge: **1** depth : **1, 8, 16***

SEE ALSO **sh1dil**

PROCEDURE **sh2ero**

SYNTAX **sh2ero** imin imout

SUBJECT Erosion dual to **sh2dil** for the complementary image.

MODES *grid: **0** edge: **1** depth : **1, 8, 16***

SEE ALSO **sh2dil**

PROCEDURE **shdil**

SYNTAX **shdil** imin imout size

SUBJECT Product (**sh1dil** o **sh2dil**) repeated *size* times.

MODES *grid: **0** edge: **1** depth : **1, 8, 16***

REMARK As a structuring element is used the regular dodecagon constructed in square grid.

SEE ALSO **shero**

PROCEDURE **shero**

SYNTAX **shero** imin imout

SUBJECT Procedure dual to **shdil** for the complementary image.

MODES *grid:* ***0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **shdil**

PROCEDURE **ringdil1**

SYNTAX **ringdil1** imin imout size

SUBJECT Dilation of *imin* by the six vertices of hexagon of size *size*. Result placed in *imout*.

MODES *grid:* ***0*** *edge:* ***1,0*** *depth :* ***1, 8***

SEE ALSO **ringdil2**

PROCEDURE **ringdil2**

SYNTAX **ringdil2** imin imout size

SUBJECT Dilation of *imin* by the contour of hexagon of size *size*. Result placed in *imout*.

MODES *grid:* ***0*** *edge:* ***1,0*** *depth :* ***1, 8***

SEE ALSO **ringdil1**

4. RANK OPERATIONS

PROCEDURE **rank**

SYNTAX **rank** imin imout rank_order

SUBJECT Rank operation for binary or greytone images (according to the input image) in hexagonal grid. The neighbourhood size is equal to 1 (element made of 7 points) and the rank order changes from 1 to 7.

MODES *grid:* ***1*** *edge:* ***1, [0]*** *depth :* ***1, 8, 16***

REMARK The rank of order 1 is the dilation, the rank of order 4 - the median, and the rank of order 7 - the erosion. The ranks 2 and 3 are dual to 5 and 6. There are 3 versions of the procedure: **rank2**, **rank3** and **rank4**. The procedure always works in the standard mode.

SEE ALSO **median**

PROCEDURE **median**

SYNTAX **median** imin imout

SUBJECT Median for binary or greytone images (according to the input image) in hexagonal or square grid. The neighborhood is a hexagon of 7 points for hexagonal grid or a diamond element of 5 points for square grid. The operation is always performed in standard mode.

MODES *grid:* ***1, 0*** *edge:* ***1, [0]*** *depth :* ***1, 8, 16***

SEE ALSO **rank**

PROCEDURE **binsegmi**

SYNTAX **binsegmi** binin binout size

SUBJECT Intersection of dilations of the set *binin* by three equal segments for hexagonal and two for square grid. The segments are oriented along the main directions of the grid and have *size* lengths.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1***

5. CONVOLUTIONS

PROCEDURE **hgauss1**

SYNTAX **hgauss1** greyin greyout

SUBJECT Gaussian convolution of size 1 in the horizontal direction. It is based on thre points with following weights ¼, ½, ¼ ,

MODES *grid:* ***0*** *edge:* ***1*** *depth :* ***8, 16***

REMARK The output image is slightly biased to the dark due to integer division round-ups.

SEE ALSO **vgauss1, gauss1**

PROCEDURE **vgauss1**

SYNTAX **vgauss1** greyin greyout

SUBJECT Gaussian convolution of size 1 in the vertical direction. It is based on three points with following weights 1/4; 1/2; 1/4.

MODES *grid:* ***0*** *edge:* ***1*** *depth :* ***8, 16***

REMARK The output image is slightly biased to the dark due to the integer division round-ups.

SEE ALSO **hgauss1, gauss1**

PROCEDURE **gauss1**

SYNTAX **gauss1** greyin greyout

SUBJECT Gaussian isotropic convolution of size 1, obtained in square grid by means of the following weights:

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

MODES *grid:* ***0*** *edge:* ***1*** *depth :* ***8, 16***

REMARK Procedure is constructed as a composition of two procedures **hgauss1** and **vgauss1**

PROCEDURE **gauss**

SYNTAX **gauss** greyin greyout size

SUBJECT Gaussian isotropic convolution of size *size*, obtained by iterations of **gauss1**.

MODES *grid:* ***0*** *edge:* ***1*** *depth :* ***8, 16***

6. OPENINGS, CLOSINGS

6.1 Basic openings, closings

PROCEDURE **open**

SYNTAX **open** imin imout size

SUBJECT Opening (binary or greytone - according to the input image) of size *size*, by an hexagon or a square (according to the *grid*).

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK The procedure is a composition of the procedure **ero** followed by **dil**.

SEE ALSO **close**

PROCEDURE **close**

SYNTAX **close** imin imout size

SUBJECT Closing (binary or greytone - according to the input image) of size *size*, by an hexagon or a square (according to the *grid*).

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK The procedure is dual to **open** for the image complement.

SEE ALSO **open**

PROCEDURE **diropen**

SYNTAX **diropen** dir imin imout size

SUBJECT Opening (binary or greytone - according to the input image) by a segment of size *size* in a direction *dir*, in hexagonal or square grid. For the coding of directions, see procedure **dirdil**.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK This procedure is a composition of **direro** followed by **dirdil**.

SEE ALSO **dirclose**

PROCEDURE **dirclose**

SYNTAX **dirclose** dir imin imout size

SUBJECT Closing, dual to **diropen** for the complementary image.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK The procedure is a composition of **dirdil** followed by **direro**.

SEE ALSO **diropen**

PROCEDURE **lineopen**

SYNTAX **lineopen** imin imout size

SUBJECT Supremum of the **diropen** operations of size *size*, along three (two) principal directions for hexagonal (square) grid.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **lineclose**

PROCEDURE **lineclose**

SYNTAX **lineclose** imin imout size

SUBJECT The operation is dual to **lineopen** for the complementary image.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **linopen**

PROCEDURE **openth**

SYNTAX **openth** greyin greyout size

SUBJECT Residue of the opening **open**, in square or hexagonal grid, of size *size* (extracts peaks and ridges that are relatively light). This operation is called top-hat by opening .

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

SEE ALSO **closeth**

PROCEDURE **closeth**

SYNTAX **closeth** greyin greyout size

SUBJECT Residue of the closing **close**, in square or hexagonal grid, of size *size* (extracts hollows and valleys that are relatively dark). This operation is called top-hat by closing.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

SEE ALSO **openth**

PROCEDURE **lineopenth**

SYNTAX **lineopenth** greyin greyout size

SUBJECT Residue of the opening **lineopen**, in hexagonal or square grid, of size *size* (extracts peaks and ridges that are relatively light). This operation is called top-hat by union of linear openings.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

SEE ALSO **linecloseth**

PROCEDURE **linecloseth**

SYNTAX **linecloseth** greyin greyout size

SUBJECT Residue of the closing **lineclose**, in hexagonal or square grid, of size *size* (extracts hollows and valleys that are relatively dark). This operation is called top-hat by intersection of linear closing.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***8, 16***

SEE ALSO **lineopenth**

PROCEDURE **pregrad**

SYNTAX **pregrad** imin imout size

SUBJECT Regularized Beucher gradient of grey image *imin*, placed in grey image *imout*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8***

SEE ALSO **gradient, regrad**

PROCEDURE **regrad**

SYNTAX **regrad** imin imout

SUBJECT Supremum of regularized Beucher gradient of grey image *imin*, placed in grey image *imout*. To reduce the computation time, the regularisation ranges from 1 to 10.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1, 8***

SEE ALSO **gredient,pregrad**

6.2 Other openings and closings

PROCEDURE **miniopen**

SYNTAX **miniopen** imin imout

SUBJECT Opening (binary or greytone - according to the input image), in standard mode, by 3 points of the elementary triangle (hexagonal grid) or by 4 points of the elementary square (square grid)

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 1***6

SEE ALSO **miniclose**

PROCEDURE **miniclose**

SYNTAX **miniclose** imin imout

SUBJECT The operation is dual to **miniopen** for the complementary image.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **isopen**

SYNTAX **isopen** imin imout size

SUBJECT Isotropic opening (binary or greytone - according to the input image), in standard mode.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK This procedure uses **isodil** and **isoero.**

PROCEDURE **isoclose**

SYNTAX **isoclose** imin imout

SUBJECT The operation is dual to **isopen** for the complementary image.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **infopen**

SYNTAX **infopen** imin imout accuracy size

SUBJECT Limit opening obtained by iterations of an *inf* of three (for hexagonal) or two (for square) openings **diropen** of size *size* along the main directions of the grid.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK *Accuracy* is either positive integer or zero, which indicates what difference of *area* (*depth* = 1) or *volume* (*depth* = 8 or 16) between two successive iterations might be considered as negligible. If *accuracy* = 0, we have the indempotence limit *sensu stricto*.

SEE ALSO **supclose**

PROCEDURE **supclose**

SYNTAX **supclose** imin imout accuracy size

SUBJECT The procedure is dual to **infopen** for the complementary image.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **infopen**

PROCEDURE **stopen**

SYNTAX **stopen** imin imout size

SUBJECT Elementary component of the procedure **infopen**.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **infopen**

7. GEODESY AND CONNECTIVITY

Note that many operators introduced in this chapter are derived from the primitive word **imbuildngb**

PROCEDURE **gdsdil**

SYNTAX **gdsdil** imin immask imout size

SUBJECT Geodesic dilation of size *size* within the mask *immask*

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **gsero**

PROCEDURE **gdsero**

SYNTAX **gdsero** imin immask imout size

SUBJECT Geodesic erosion dual to **gdsdil** for the complementary image

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **gsdil**

PROCEDURE **build**

SYNTAX **build** mask iminout

SUBJECT Complete geodesic dilation of the image *iminout* under the image *mask*. The result is put in the *iminout* image. As an operation on i*minout*, with fixed mask, **build** is a closing. On the contrary, as a transform of the mask with fixed *iminout*, it is the connected opening of the mask by *iminout*.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **buildopen**

SYNTAX **buildopen** imin imout size

SUBJECT Opening by reconstruction of size *size* of *imin* in the *imout*.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **buildclose**

SYNTAX **buildclose** imin imout size

SUBJECT Closing by reconstruction of size *size* of *imin* in the *imout.*

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **areaopen**

SYNTAX **areaopen** binin binout size print

SUBJECT Stores in *binout* the grains of *binin* (connected components) whose area is greater or equal to *size* (in pixels). If *print* = 1, the procedure prints a list of areas for all the grains of *binin.*

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1***

PROCEDURE **ringbuild**

SYNTAX **ringbuild** imin immask imout size

SUBJECT Reconstruct *imin* by the inside mask *immask*, by means of dilations according to the vertices of hexagon of size *size*.

MODES *grid:* ***1*** *edge:* ***1,0*** *depth* ***: 1,8***

SEE ALSO **build, buildopen, buildclose**

PROCEDURE **gdsdist**

SYNTAX **gdsdist** binin binmask greyout

SUBJECT Geodesic distance function of binary image *binin*, inside binary mask *binmask*. The result is placed in grey image *greyout*.

MODES *grid:* ***1*** *edge:* ***1,0*** *depth* ***: 1,8***

SEE ALSO **distance**

PROCEDURE **recons**

SYNTAX **recons** imin immask imout type

SUBJECT Numerical reconstruction of image *imin* inside mask *immask*, the result is placed in *imout*. If *type*=0, the reconstruction works on the input images, and yields the reconstruction opening. If *type*=1, the procedure works on the dual form and yields the reconstruction closing.

MODES *grid:* ***1,0*** *edge:* ***1,0*** *depth* ***: 1,8***

SEE ALSO **build**

PROCEDURE **levelling**

SYNTAX **levelling** imin immask imout

SUBJECT Places in *imout* the levelling of *imin* according to marker *immask*.

MODES *grid:* ***1,0*** *edge:* ***1,0*** *depth* ***: 1,8,16***

REMARK The levelling is the commutative product of the opening by reconstruction, for a given marker, by the closing by reconstruction, for the same marker.

SEE ALSO **recons, buildopen, buildclose**

8. APPLICATIONS OF GEODESY

PROCEDURE **edgeoff**

SYNTAX **edgeoff** binin binout

SUBJECT Deletes objects that touch the image border.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1***

PROCEDURE **fgrain**

SYNTAX **fgrain** binin binout

SUBJECT Puts in *binout* the first grain of *binin* and removes it from the *binin* image.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1***

PROCEDURE **border**

SYNTAX **border** binout

SUBJECT The boundary of the field is stored in *binout*.

MODES *grid:* ***1, 0*** *edge:* ***[1],0*** *depth :* ***1, 8, 16***

PROCEDURE **clohole**

SYNTAX **clohole** imin imout

SUBJECT Closes internal pores (filling the basins - in greytone case) in *imin*.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **maxima**

SYNTAX **maxima** greyin binout

SUBJECT Stores in *binout* the maxima of *greyin.*

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1, 8, 16***

PROCEDURE **minima**

SYNTAX **minima** greyin binout

SUBJECT Puts in *binout* the minima of *greyin.*

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1, 8, 16***

PROCEDURE **extmaxima**

SYNTAX **extmaxima** greyin binout height

SUBJECT Puts in *binout* the maxima of *greyin* that are surrounded by non ascending zones higher or equal to *height.*

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1, 8, 16***

PROCEDURE **extminima**

SYNTAX **extminima** greyin binout height

SUBJECT Puts in *binout* the minima of *greyin* that are surrounded by non descending zones higher or equal to *height*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1, 8, 16***

PROCEDURE **swamping**

SYNTAX **swamping** greyin binmark greyout

SUBJECT Modification of the homotopy of *greyin* in order to assert on it all connected components from *binmark* as minima. Proceeds via closing by reconstruction of *greyin* relative to the complement of *binmask* considered as a greytone function.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth:* ***8, 16***

PROCEDURE **extrema**

SYNTAX **extrema** imin immask binout1 binout2

SUBJECT Provides in *binout1* (resp. *binout2*) the set of the highest (resp. lowest) values of *imin* inside *immask*. Prints the two extremum values.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1, 8***

SEE ALSO **maxima, minima, extmaxima, extminima**

PROCEDURE **grainclose**

SYNTAX **grainclose** imin imout size1 size2

SUBJECT Suppresses the small features of imin by a buildopen of size *size1*. Then close the results by closing *size2*, and takes the supremum with the initial image (for putting back the small features). Places the result in imout.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1, 8, 16***

REMARK Under iteration, grainclose becomes a closing. In practice, idempotence is quasi reached after first run. By taking *size1* = 0, one just performs closing step.

SEE ALSO **indivaf**

PROCEDURE **indivaf**

SYNTAX **indivaf** binin binout size1 size2

SUBJECT The procedure performs firstly the opening by reconstruction **buildopen** of size *size1* on the binary input *binin*, and then an individual closing of each particle, and take the union of the results.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1***

REMARK Initially, **indivaf** is an alternating filter. Indeed, idempotence is practically reached at the first run. By taking *size1* = 0, one bypasses the initial phase of opening by reconstruction.

SEE ALSO **grainclose**

PROCEDURE **dynamics**

SYNTAX **dynamics** imin imout height type

SUBJECT Selection of the maxima/minima of the grey image imin, whose dynamics is higher than *height*. If *type*=0 the minima are extracted, else the maxima. In both cases they are placed in the grey image *imout*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***8***

SEE ALSO **extmaxima, extminima**

9. FILTERS

9.1 Main filters

For all procedures described in this chapter, the parameter *type* forces operation to start with closing (*type* = 1) or opening (*type* ≠ 1)

PROCEDURE **af**

SYNTAX **af** imin imout size type

SUBJECT Alternating filter; product of **close** followed by **open** (*type* = 1) or **open** followed by **close** (*type* ≠ 1)

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **fullasf**

SYNTAX **fullasf** imin imout size type

SUBJECT Alternating sequential filter of step 1 constructed by means of **af**. In other words:

fullasf (*size*) = **af**(*size*) o ... **af**(3) o **af**(2) o **af**(1)

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK The parameter *type* has the same meaning as in the **af** procedure.

PROCEDURE **lineaf**

SYNTAX **lineaf** imin imout size type

SUBJECT Alternating filter constructed by means of **lineopen** and **lineclose**. The order is determined by the parameter *type* (*type* = 1 - **lineclose** followed by **lineopen**)

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **asf**

SYNTAX **asf** imin imout size type

SUBJECT Alternating sequential filter by increasing steps constructed by means of **af**. The progression fulfills the relationship:

size(i) = *size* (i - 1) + i - 1

So, for the first steps we have the following values: 1, 2, 4, 7, 11, 16 ... :

asf(10) = **asf**(7) = **af** (7) o **af**(4) o **af**(2) o **af**(1)

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK This procedure is faster then **fullasf**, especially for big values of *size*, and almost equally accurate.

SEE ALSO **fullasf**

PROCEDURE **lineasf**

SYNTAX **lineasf** imin imout size type

SUBJECT Alternating sequential filter constructed by means of **linea** according to the same progression of *sizes* as for **asf**

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **automed**

SYNTAX **automed** imin imout size

SUBJECT Morphological centre between two filters **af**(*size*) of *type* 1 and 0

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **centre**

SYNTAX **centre** imin imin1 imin2 imout

SUBJECT Morphological centre between two images *imin1* and *imin2* , related to the reference image *imin*

MODES *grid:* ***1, 0*** *edge:* ***1, 8*** *depth :* ***1, 8, 16***

PROCEDURE **contrast**

SYNTAX **contrast** greyin greyout sz type

SUBJECT Contrast $\kappa(f)$ in *greyin* := f , defined as follows:

If $\xi(f)(x) - f(x) \leq f(x) - \eta(f)(x)$ then $\kappa(f)(x) = \xi(f)(x)$
else $\kappa(f)(x) = \eta(f)(x)$.

and:

type = 1 $\Rightarrow$ $\xi(f)$:= **isodil** (f;sz) $\eta(f)$:= **isoero** (f;sz)

type = 2 $\Rightarrow$ $\xi(f)$:= **isopen** (f;sz) $\eta(f)$:= **isoclose** (f;sz)

type > 2 $\Rightarrow$ $\xi(f)$:= **isopen** (f;sz) $\eta(f)$:= **isoclose** (f;5×sz) .

MODES *grid :* ***1, 0*** *edge* : ***1*** *depth* : ***8, 16***

PROCEDURE **contrasth**

SYNTAX **contrasth** greyin greyout sz

SUBJECT Contrast κ(f) in *greyin* := f , using top-hat by opening and top-hat by closing:

κ(f) := 3f **-** **open** (f;sz) **- close** (f;sz)

MODES *grid :* ***1, 0*** *edge* : ***1*** *depth* : ***8, 16***

9.2 Supplementary filters

PROCEDURE **minifilt**

SYNTAX **minifilt** imin imout size type

SUBJECT Alternated filter constructed by means of **miniopen** and **miniclose**.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **isaf**

SYNTAX **isaf** imin imout size type

SUBJECT Isotropic alternating filter of size *size*, binary or greytone, in hexagonal or square grid. If *type*=1, the operation begins with closing, otherwise with opening.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **isasf**

PROCEDURE **isasf**

SYNTAX **isasf** imin imout size type

SUBJECT Isotropic alternating sequential filter of size *size*, binary or greytone, in hexagonal or square grid. If *type*=1 the operation begins with closing, otherwise with opening.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

PROCEDURE **buildaf**

SYNTAX **buildaf** imin imout size t

SUBJECT Alternating filter by reconstruction, obtained by composition of **buildopen** followed by **buildclose** (t = 0), or **buildclose** followed by **buildopen** (t=1)

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **buildasf**

PROCEDURE **buildasf**

SYNTAX **buildasf** imin imout size t

SUBJECT Alternating sequential filter by increasing steps constructed by means of **buildaf**. The progression fulfills the relationship:

size(i) = *size* (i - 1) + i - 1. For the first steps we have:
1, 2, 4, 7, 11, 16 ..:
asf (10) = **asf** (7) = **af** (7) o **af** (4) o **af** (2) o **af** (1)

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

REMARK The parameter *t* has the same meaning as in the **buildaf** procedure

---

PROCEDURE **minibuildaf**

SYNTAX **minibuildaf** imin imout

SUBJECT Product of the opening by reconstruction from **miniopen** by the closing by reconstruction from **miniclose**.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth :* ***1, 8, 16***

SEE ALSO **miniopen, miniclose**

PROCEDURE **closoropen**

SYNTAX **closoropen** greyin greyout size

SUBJECT Toggle between **gbuildopen**, **gbuildclose**, of size *size*, and the identity operator. If at point x the image is equal to closing, then we take opening, if it is equal to opening then we take closing, else the point is unchanged.

MODES *grid :* ***1, 0*** *edge* : ***1 , 0*** *depth* : ***8, 16***

PROCEDURE **binmiddle**

SYNTAX **binmiddle** binin1 binin2 binout

SUBJECT Constructs the median set between *binin1* and *binin2* and places it in *binout*.

MODES *grid :* ***1, 0*** *edge* : ***1 , 0*** *depth* : ***1***

REMARK The intersection between *binin1* and *binin2* must be non-empty.

SEE ALSO **grmiddle**

PROCEDURE **grmiddle**

SYNTAX **grmiddle** greyin1 greyin2 greyout

SUBJECT Generates the median greytone image between *greyin1* and *greyin2* and places it in *greyout*.

MODES *grid :* ***1, 0*** *edge* : ***1 , 0*** *depth* : ***8***

SEE ALSO **binmiddle**

10. MAXIMAL BALLS AND SKELETON

PROCEDURE **binopenskel**

SYNTAX **binopenskel** binin binout

MODES Non connected skeleton of *binin*, centers of the (or hexagonal) maximal balls.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1***

PROCEDURE **binultim**

SYNTAX **binultim** binin binout

SUBJECT Ultimate erosion of *binin*

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1***

PROCEDURE **centroid**

SYNTAX **centroid** binin binout

SUBJECT Puts in *binout* the first found pixel of the ultimate erosion

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1***

REMARK The centroid is always included in the source set**.**

PROCEDURE **condbis**

SYNTAX **condbis** binin binout size

SUBJECT Conditional bisector of the binary image *binin*, the result is placed in the binary image *binout*. Parameter *size* of the conditional bisector ranges from 1 to 10.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth :* ***1***

11. THINNINGS, THICKENINGS

11.1 Binary thinnings and thickenings

The thinnings and thickenings presented here are constructed in three stages. Let us see first the basic stage. For example, for the thickening of X by the structuring element {se1(α), se0(α)}, this stage is being written:

$$Y(X; se1(\alpha), se0(\alpha)) := Y(X, \alpha) := X \cup (X \ominus se1) \cap (X^c \ominus se0) \qquad (1)$$

Then, we compute the product of composition of $Y(X,\alpha)$ when the structuring element follows the six or eight rotations of the grid. In the case of hexagonal grid, for example, we have:

$$S (X) := Y(X, \alpha+5\pi/6) \text{ o } Y(X, \alpha+4\pi/6) \text{ o...o } Y(X, \alpha) \qquad (2)$$

And finally comes the iteration of the cycle S(X) until stability. Only the two last steps are made in the procedures (the first one is performed directly by using the primitive **imhitormiss**). Thus, the relationship (2) is executed by means of the procedure **thickturn**, and its complete iteration by **thick**. In the thinning case, $Y(X,\alpha)$ is replaced by the following:

$$Y'(X, \alpha) := X / (X \ominus se1) \cap (X^c \ominus se0) \qquad .$$

One can note that the duality for the complement images *inverts* the two parts of the structuring element. The basic stage becomes :

$$Y' (X ; se1, se0) = [Y(X^c; se0, se1)]^c \qquad (3)$$

and the inversion applies also for the pairs **thickturn-thinturn**, and **thick-thin**. As for geodesic or standard mode, either can be advantageous depending on the structuring element chosen.

Similarly this three steps construction is also used for the geodesic version of the algorithms. The only difference is at the beginning of the first stage (1):

$$Y(X, \alpha) := [X \cup (X \ominus se1) \cap (X^c \ominus se0)] \cap \text{mask}$$

where *mask* is the geodesic mask. This is the reason why the geodesic version of the algorithms are presented as variants of the standard procedures. Note that these routines are constructed to work with *edge*=0, that is, in the standard border mode.

PROCEDURE **thickturn**

SYNTAX **thickturn** se1 se0 binin binout

SUBJECT Puts in *binout* the result S of the elementary thickening cycle by the dipole {se1(α); se2 (α)} and its rotations. If *binin* = X, we obtain *binout*=S according to relationship (2)

MODES *grid :* ***1, 0*** *edge:* ***1,[0]*** *depth:* ***1***

REMARK There exists also the geodesic version:

REMARK **gdsthickturn** se1 se0 binin mask binout

SEE ALSO **thinturn**, **gdsthinturn**

PROCEDURE **thick**

SYNTAX **thick** se1 se0 binin binout

SUBJECT Iteration (until stability) of "**thickturn** *se1 se0 binin binout*"

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1***

REMARK There exists a geodesic version (in standard mode) :

gdsthick se1 se0 binin mask binout

SEE ALSO **thin**, **gdsthin**

PROCEDURE **thinturn**

SYNTAX **thinturn** se1 se0 binin binout

SUBJECT Thinning dual to **thickturn** for the complementary image.

MODES *grid:* ***1, 0*** *edge:* ***[1], 0*** *depth:* ***1***

REMARK See the inversion of structuring elements *se1* and *se2* in relationship (3). There exists a geodesic version **gdsthinturn**, obtained by applying the relationship (3) to **gdsthickturn**

SEE ALSO **thinturn**, **gdsthinturn**

PROCEDURE **thin**

SYNTAX **thin** se1 se0 binin binout

SUBJECT Iteration (until stability) of "**thinturn** *se1 se0 binin binout*"

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1***

REMARK There exists also the geodesic thinning

gdsthin se1 se0 binin mask binout
obtained by iterating **gdsthinturn** until stability

SEE ALSO **thick, gdsthick**

PROCEDURE **Dthick**

SYNTAX **Dthick** binin binout

SUBJECT A particular version of **thick** using structuring element D:

		*	1			1	1	*
hexagonal	0	*	*	;	square	*	*	0
		0	0			0	0	0

MODES *grid :* ***1 , 0*** *edge* : ***1, 0*** *depth* : ***1***

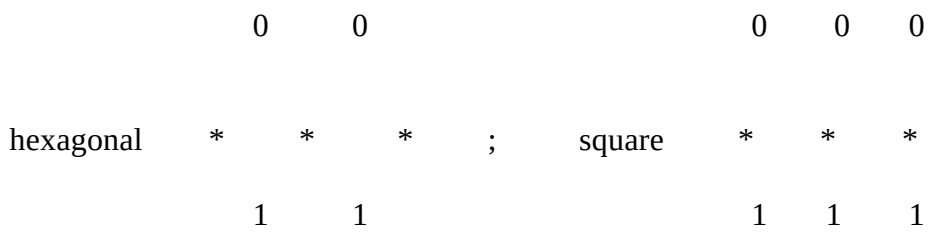
REMARK Pseudo convex hull of grains. Reduces simply connected components of the pores to a point. The procedure gives better results with *edge*=0. In hexagonal grid the procedure is used to launch thickenings for connected components reduced to a point.

SEE ALSO **Mthick**, **Lthick**

PROCEDURE **Lthick**

SYNTAX **Lthick** binin binout

SUBJECT A particular version of **thick** using structuring element L:



MODES *grid : **1 , 0** edge* : ***1, 0*** *depth* : ***1***

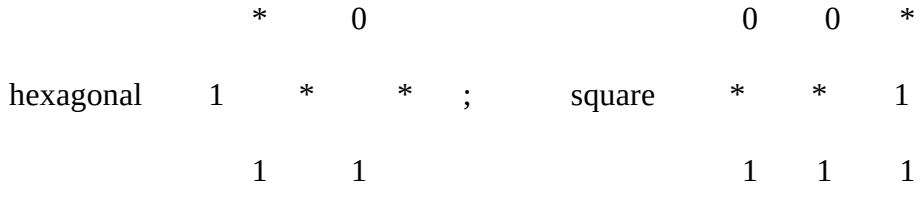
REMARK Among all structuring elements that preserve homotopy, this one gives the result closest to the skeleton of the background. The procedure gives better result with *edge*=1 and *grid*=1.

SEE ALSO **Lthick**, **Dthick**

PROCEDURE **Mthick**

SYNTAX **Mthick** binin binout

SUBJECT A particular version of **thick** using structuring element M:



MODES *grid : **1 , 0** edge* : ***1, 0*** *depth* : ***1***

REMARK Works as **Lthick**, but the result is more "noisy".

SEE ALSO **Lthick, Dthick**

PROCEDURE **Dthin**

SYNTAX **Dthin** binin binout

SUBJECT Procedure dual to **Dthick** for the complementary images, uses the same structuring element as **Dthick**.

MODES *grid : **1, 0** edge* : ***1, 0*** *depth* : ***1***

REMARK Marks by single points simple connected components.. The result is better with *edge* = 0.

SEE ALSO **Lthin, Mthin**

PROCEDURE **Lthin**

SYNTAX **Lthin** binin binout

SUBJECT Procedure dual to **Lthick** for complementary images.

MODES *grid :* ***1 , 0*** *edge* : ***1, 0*** *depth* : ***1***

REMARK Basic thinnings to simulate skeletons. Returns better results in hexagonal grid, since foreground and background have the same connectivity.

SEE ALSO **Mthin** , **Dthin**

PROCEDURE **Mthin**

SYNTAX **Mthin** binin binout

SUBJECT Procedure dual to **Mthick** for the complementary images.

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***1***

REMARK Seldom used

SEE ALSO **Lthin** , **Dthin**

PROCEDURE **endpoints**

SYNTAX **endpoints** binin binout

SUBJECT Ending points of the *binin* image are stored in *binout.*

MODES *grid :* ***1 , 0*** *edge* : ***1 ,[0]*** *depth* : ***1***

REMARK Is of special interest when *binin* is the result of an homotopic thinning or thickening.

PROCEDURE **mulpoints**

SYNTAX **mulpoints** binin binout

SUBJECT Multipoints of *binin*.

MODES *grid :* ***1 , 0*** *edge* : ***1, [0]*** *depth* : ***1***

REMARK Is of special interest when *binin* is the result of an homotopic thinning or thickening.

PROCEDURE **gdscentre**

SYNTAX **gdscentre** binin binout

SUBJECT Puts in *binout* the geodesic centre of *binin*.

MODES *grid* : ***1, 0*** *edge* : ***1, 0*** *depth* : ***1***

REMARK Refinement of the procedure **Dthin**, but much more time-consuming. Works only in hexagonal grid.

11.2 Greytone thinnings and thickenings

PROCEDURE **greyseero**

SYNTAX **greyseero** el_struct greyin greyout

SUBJECT Erosion of *greyin* by a structuring element *el_struct* consisting of the points of the unitary square or hexagonal neighbourhood, which is coded as for **imhitormiss**

MODES *grid :* ***1, 0*** *edge* : ***1*** *depth* : ***8, 16***

SEE ALSO **imhitormiss**

PROCEDURE **greysedil**

SYNTAX **greysedil** el_struct greyin greyout

SUBJECT Dilation, dual to **greyseero** when performed with the inverted image.

MODES *grid :* ***1, 0*** *edge* : ***1*** *depth* : ***8, 16***

PROCEDURE **greythinstep**

SYNTAX **greythinstep** se_for_inf se_for_sup greyin greyout

SUBJECT Step of thinning (greytone) of *greyin* by a mixed structuring element { *se_for_inf* ; *se_for_sup* }, result in *greyout*.

MODES *grid :* ***1, 0*** *edge* : ***1*** *depth* : ***8, 16***

PROCEDURE **greythickstep**

SYNTAX **greythickstep** se_for_inf se_for_sup greyin greyout

SUBJECT Thickening, dual to **greythinstep** for the negative image.

MODES *grid :* ***1, 0*** *edge* : ***1*** *depth* : ***8, 16***

PROCEDURE **dirgradient**

SYNTAX **dirgradient** dir greyin greyout

SUBJECT Gradient in the direction dir of the hexagonal grid , defined as the difference between **greythickstep** and **greythinstep**. The structuring element used is a pair of points +1 and -1 in the direction dir.

MODES *grid :* ***1*** *edge* : ***1*** *depth* : ***8, 16***

REMARK Using thinning and thickening instead of dilatation and erosion allows to obtain 12 azimuth directions.

PROCEDURE **vectgrad0**

SYNTAX **vectgrad0** greyin greyout1 greyout2

SUBJECT The complete "brut" gradient of *greyin* obtained by applying **dirgradient** for all the six hexagonal directions. The module is placed in *greyout1,* and the azimuth in *greyout2*.

MODES *grid :* ***1*** *edge* : ***1*** *depth* : ***8, 16***

REMARK Multiple or incoherent directions can appear *(* cf. **gradvect*)***

PROCEDURE **gradvect**

SYNTAX **gradvect** greyin greyout1 greyout2

SUBJECT The true vector gradient obtained by sorting directions in **gradvect0**. The module is placed in *greyout1* and the azimuth with 12 directions, plus zero, in *greyout2*. Even values correspond to existing directions, the odd ones to those lying in between.

MODES *grid :* ***1*** *edge* : ***1*** *depth* : ***8, 16***

PROCEDURE **mainthin**

SYNTAX **mainthin** binin binout size

SUBJECT Thins *binin* according to Lthinning algorithm, and suppresses then the branches of the thinning where length is shorter than *size*.

MODES *grid* ***: 1,0*** *edge* : ***1,0*** *depth* : ***1***

SEE ALSO **Lthin, clip**

PROCEDURE **greyhmt**

SYNTAX **greyhmt** imin es1 es2 imout

SUBJECT Hit or miss transformation for the 8-bits image *imin*, by the two structuring elements : *es1* (for the domes) and *es2* (for the valleys).

MODES *grid* ***: 1,0*** *edge* : ***1,0*** *depth* : ***8***

SEE ALSO **greythickstep, greythinstep**

PROCEDURE **greythickturn**

SYNTAX **greythickturn** es1 es2 imin imout

SUBJECT One cycle of six (**grid**=1) or eight (**grid**=0) elementary steps of grey thickenings (i.e. **greythickstep** procedures) according to the different directions of the grid.

MODES *grid* ***: 1,0*** *edge* : ***1,0*** *depth* : ***8***

SEE ALSO **greythickstep, greythinstep**

PROCEDURE **greythick**

SYNTAX **greythick** se1 se2 imin imout

SUBJECT Grey thickening of *imin* by the structuring element *se1* (for the domes) and *se2* (for the valleys), placed in *imout*.

MODES *grid* ***: 1,0*** *edge* : ***1,0*** *depth* : ***8***

SEE ALSO **greythickstep, greythickturn, greythinstep**

12. WATERSHEDS

PROCEDURE **clip**

SYNTAX **clip** binin binout

SUBJECT Clips the image *binin*, i.e. thins it by means of the element:

			*	0			*	0	0
hexagonal	*	1	0	;	square	1	1	0	
			0	0			*	0	0

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***1***

REMARK Borders are treated according to *edge*.

PROCEDURE **skiz**

SYNTAX **skiz** binin binout

SUBJECT Puts in *binout* an homotopic thickening, totally clipped off, of the complement of the binary image *binin*. This procedure, often used in mathematical morphology, is called Skeleton by Influence Zone.

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***1***

REMARK The result depends on the value of *edge*. There exist also geodesic version:

gdskiz binin binmask binout

PROCEDURE **threshwshed**

SYNTAX **threshwshed** greyin binout

SUBJECT Non marked watershed of *greyin* stored in *binout*.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth:* ***8, 16 to 1***

REMARK Because of the arithmetical progression of the thresholding value, the procedure is slow.

PROCEDURE **mwshed**

SYNTAX **mwshed** greyin binmark binout type

SUBJECT Conditional watershed of *greyin* with miarkers in *binmark*. The result is put in *binout*. The algorithm runs through successive thresholds using an arithmetical (*type*=1) or geometrical (*type*=0) increment.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth:* ***8, 16 to 1***

REMARK The type 0 is more suitable for gradient images where the first thresholds are specially relevant.

PROCEDURE **wshed**

SYNTAX **wshed** greyin binout1 binout2 type

SUBJECT Watershed. The result is put in *binout1*, the minima are stored in *binout2*. The algorithm runs through successive thresholds using an arithmetical (*type*=1) or geometrical (*type*=0) increment.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth:* ***8, 16 to 1***

13. SEGMENTATION

PROCEDURE **mgradwshed**

SYNTAX **mgradwshed** greyin binmark binout type

SUBJECT Procedure **mwshed** applied to the **gradient** of *greyin*, the parameters *binmark*, *binout*, and *type* have the same definitions as for **mwshed**.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth:* ***8, 16 to 1***

PROCEDURE **gradwshed**

SYNTAX **gradwshed** greyin binout1 binout2 type

SUBJECT Procedure **wshed** applied to the **gradient** of *greyin*, the parameters *binmark*, *binout*, and *type* have the same definitions as for **wshed**.

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth:* ***8, 16 to 1***

PROCEDURE **shapeseg**

SYNTAX **shapeseg** greyin binin greyout size t

SUBJECT The procedure sets to zero (in *greyout*) the points of *greyin* which belong to the watershed of the complement of the distance function of *binin*. To avoid over-segmentation, the distance function is filtered by an opening by reconstruction of size *size* and type *t* (**diropen** if t=1, **open** if t > 1)

MODES *grid:* ***1, 0*** *edge:* ***1*** *depth:* ***8, 16 and 1***

PROCEDURE **mosaic**

SYNTAX **mosaic** greyin greyout1 greyout2

SUBJECT Mosaic image of *greyin* (placed in *greyout1*) and gradient of this mosaic image (placed in *greyout2*). The mosaic image has for flat zones the catchment basins of the gradient image of *greyin*.

MODES *grid:* ***1, 0*** *edge:* ***1,0*** *depth:* ***8***

SEE ALSO **wfall, kheops**

PROCEDURE **wfall**

SYNTAX **wfall** greyin greyout binout

SUBJECT Waterfall algorithm applied to *greyin* (ie a step in the construction of the watershed pyramid procedure **kheops**).

MODES *grid:* ***1, 0*** *edge:* ***1,0*** *depth:* ***8***

SEE ALSO **mosaic, kheops**

PROCEDURE **kheops**

SYNTAX **kheops** greyin greyout filename

SUBJECT Pyramid of mosaic images obtained from *greyin*, by successive watersheds and waterfalls. The various mosaic images are displayed in *greyout*, and saved in the Micromorph directory as "*filename i.bmp*", where *i* stands for the i-th level.

MODES *grid:* ***1, 0*** *edge:* ***1,0*** *depth:* ***8***

SEE ALSO **mosaic, wfall**

PROCEDURE **jump**

SYNTAX **jump** greyin greyout jump_size type

SUBJECT Segments the grey image *greyin* by climbing from the minima by the value *jump_size*, descending from the maxima by value *jump_size*, and iterating the process. Type varies from 1 to 3, and insert a small regularisation during the process (3 is more regular than 1).

MODES *grid:* ***1, 0*** *edge:* ***1,0*** *depth:* ***8***

SEE ALSO **jumpcnc**

PROCEDURE **smoothcnc**

SYNTAX **smoothcnc** greyin binout slope_size neighborhood_size

SUBJECT The procedure generates a contrast extraction on grey input *greyin* in such a way that in each connected component of the binary output binout, the grey fluctuations in any hexagon of size *neighborhood_size* are smaller than that of parabola y=(*slope_size*)*t2.

MODES *grid:* ***1,[0]*** *edge:* ***1,0*** *depth:* ***8***

SEE ALSO **jumpcnc, jump**

PROCEDURE **jumpcnc**

SYNTAX **jumpcnc** greyin binout slope_size type

SUBJECT Generates a contrast extraction on the grey input *greyin*, such that in each connected component of the binary output *binout*, the grey tones range inside 2*(*slope_size*). The connected components are obtained:

if *type*=0 ,from the minima,
if *type*=2 ,from the maxima,
if *type*=1 in a self-dual way, from both minima and maxima.

MODES *grid:* ***1,[0]*** *edge:* ***1,0*** *depth:* ***8***

SEE ALSO **smoothcnc**

14. MEASURES

Generally, the values which are returned by the MICROMORPH functions are simply stored in a variable then displayed by the **print** command. If these values must be saved, this can be done by opening a file before calling the function. This file must be closed at the end of the process. As an example, saving the variation of the perimeter values of the successive erosions of the image b1 will be performed by means of the following operations:

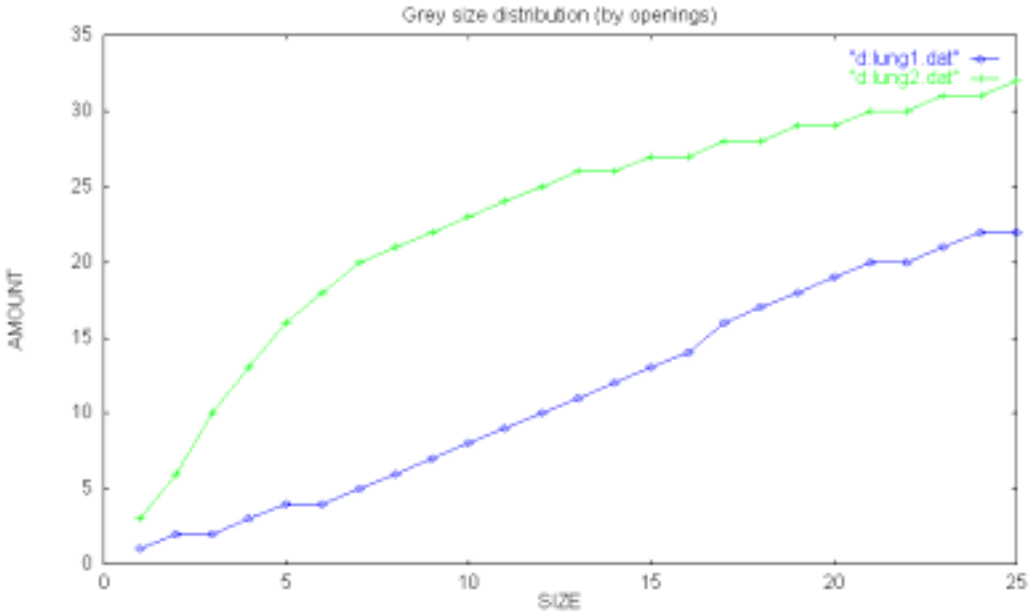
```
output "c:\b1.dat"
while  (area b1) do
print perim b1
ero b1 b1 1
end
output " "
```

However, it is possible to set up the name of the file as a parameter. It is the case in particular for the functions defined in chapter 15 (curves). For instance, the size distribution from size 1 to size 25 computed on the *lung1* and *lung2* images will be defined as (the initial images are initially stored in g1):

```
granul  g1  1  25 "lung1.dat"
granul  g1  1  25  "lung2.dat"
```

The size distribution values are therefore stored in the *lung1.dat* and *lung2.dat* files. These files can be used in any further treatment. In particular, they can be sent to an external program *(gnuplot* for instance) in order to display the size distribution curves. Here are the gnuplot commands which produce the curves below::

```
set xlabel "SIZE"  0
set ylabel "AMOUNT" 0
 set title "Grey size distribution (by openings)"
 plot "c:\lung1.dat" with linespoint , "c:\lung2.dat" with linespoint
```



14.1 Basic measures

This chapter deals with measures and focuses on the basic stereological functions. The area already belongs to the primitive MICROMORPH words under the name **imvolume**. So, we need only to define the *diameter variation*, the *perimeter* and the *connectivity number* or Euler-Poincaré constant.

FUNCTION **diameter**

SYNTAX v := **diameter** dir binin

SUBJECT Returns the number of entries in the grains of *binin* in the direction *dir*

MODES *grid:* ***1, 0*** *edge:* ***0, [1]*** *depth:* ***1***

REMARK The function doesn't take into account the intersections of grains with the borders.

SEE ALSO **binhferret**, **binvferret**

FUNCTION **digperim**

SYNTAX v := **digperim** binin binout

SUBJECT Returns the number of elements of the hexagonal contour of the set *binin*. Places in *binout* the union of the contours.

MODES *grid:* ***1, [0]*** *edge:* ***0, [1]*** *depth:* ***1***

REMARK **Digperim** counts twice the lines of thickness "1" and ignore the isolated points (the result is different from the area of **cont** S). However, since the digital contour of S differs from that of S^c , the function **digperim** is not autodual. However, when S does not touch the field borders, we have:

digperim S - **digperim** S^c = 6 * **cnumber** S

SEE ALSO **cont**, **sq4cont**, **perim**, **cnumber**

FUNCTION **perim**

SYNTAX v := **perim** binin

SUBJECT Returns the perimeter of *binin* estimated by using the Cauchy formulae.

MODES *grid:* ***1, 0*** *edge:* ***0, [1]*** *depth:* ***1***

REMARK This is the auto-dual measure supposed to estimate without bias the Euclidean perimeter, when existing, starting from the data on the square or the hexagonal grid.

SEE ALSO **cont**, **sq4cont**, **digperim**.

FUNCTION **cnumber**

SYNTAX v := **cnumber** binin tr

SUBJECT Returns the number of connectivity of the source image *binin*. If *tr* = 0, the function takes into account the objects included in the image field and not touching the borders, otherwise it takes into account all objects.

MODES *grid:* ***1, [0]*** *edge:* ***0, [1]*** *depth:* ***1***

FUNCTION **binhferret**

SYNTAX v := **binhferret** binin

SUBJECT Ferret diameter of *binin* in the horizontal direction (diametral variation of the convex hull of binin)

MODES *grid:* ***1*** *edge:* ***1*** *depth:* ***1***

REMARK The vertical ferret diameter exists as well : **binvferret**

14.2 Other measures

FUNCTION **var0**

SYNTAX v := **var0** greyin

SUBJECT Order 0 variance of *greyin*. The function calculates:
E [| f(x) - m |], where f(x) is the grey-level at the point x, and m is its mean grey-level.

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***8, 16***

FUNCTION **var1**

SYNTAX v := **var1** greyin

SUBJECT Order 1 variance of *greyin*. The function calculates 0.5 E [|f(x) - f(y)|], where x and y run through greyin independently. The algorithm is:

$$\text{var1}(f) = \int_{T\,G(t)} [\,1 - G(t)\,]\,dt$$

where G(t) is the greytone distribution function.

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***8, 16***

REMARK If greyin is modeled by a random function with an order 1 finite variance, then var1(greyin) estimates the asymptotic value of vario1(greyin)

FUNCTION **var2**

SYNTAX v := **var2** greyin

SUBJECT Variance of *greyin*. The function calculates E [(f(x) - m)^2], where f(x) is the grey level at point x, and m is the mean grey level.

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***8, 16***

REMARK If greyin is modeled by a random function with a finite variance, then **var2**(*greyin*) estimates the asymptotic value of **vario2**(*greyin*)

FUNCTION **rug**

SYNTAX v := **rug** imin

SUBJECT Estimates the roughness of a binary or greytone image, defined as the mean of the square of the curvature (of the set or section of the function) by calculating the second derivative at the origin of the density of the intercepts distribution.

MODES *grid:* ***[1], 0*** *edge:* ***1, 0*** *depth:* ***1, 8, 16***

PROCEDURE **mse**

SYNTAX v := **mse** greyin1 greyin2

SUBJECT Returns the mean quadratic difference between the two inputs.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***8***

PROCEDURE **bincount**

SYNTAX **bincount** binin

SUBJECT Returns the number of connected components of the set *binin*.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1***

SEE ALSO **cnumber**

PROCEDURE **floatcount**

SYNTAX **floatcount** greyin

SUBJECT Returns the number of flat zones with a non-empty erosion.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***1, 8, 16***

PROCEDURE **flatzone**

SYNTAX **flatzone** greyin

SUBJECT Function, which returns the area occupied by the flat zones of the input grey image. A zone of the support of grey is flat when it is connected, when its unit erosion is not empty, and when greyin is constant over it.

MODES *grid:* ***1, 0*** *edge:* ***1, 0*** *depth:* ***8, 16***

15. CURVES

All the procedures and functions given in this chapter store their values in a file. The name of the file is entered as a parameter. This file can be used in a graphical representation of the result (for instance with gnuplot). These results can also easily be used for further processing with

---

PROCEDURE **bincov**

SYNTAX **bincov** dir st spacing binin1 binin2 sizemax fname

SUBJECT Cross- covariance between the two sets *binin1* and *binin2* in the direction *dir* (in hexagonal or square grid), of size from 0 to *sizemax*, by step *spacing*. Parameter *st* permits to chose between standard (*st*=0) or geodesic(*st*=1) modes. The results are stored in the file *fname*. Divide by 1000 to get the covariance.

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***1***

REMARK To obtain ordinary covariance one should take *binin2 = binin1*, and to obtain variogram one should takes *binin2* = $(binin1)^C$.

SEE ALSO **vario1**, **vario2** .

PROCEDURE **covar**

SYNTAX **covar** dir spacing imin sizemax fname

SUBJECT Covariance of the greytone image *greyin* in the direction *dir* (in square or hexagonal grid), of *size* 0 to *sizemax*, using *step* spacing. The parameter *st* allows to choose between standard (*st*=0) or geodesic(*st*=1) modes. The results are stored in the file *fname*.

MODES *grid :* ***1, 0*** *edge* : ***1, [0]*** *depth* : ***8, 16***

REMARK When computing a covariance there is the theoretical assumption that the algorithm refers to an actual covariance. However, this theoretical

covariance may not exist. The technique to check it consist in calculating the variogram.(see **vario2**).

SEE ALSO **vario2, bincov**

FUNCTION **vario1**

SYNTAX **vario1** dir st spacing greyin sizemax fname

SUBJECT Variogram of order 1 of the greytone image *greyin*, *i.e.* semi-expectation of the module of f(x+h) - f(x) , in the direction *dir* (in hexagonal or square grid) of size 0 to *sizemax*, using step *spacing*. Parameter *st* permits to chose between standard (*st*=0) or geodesic(*st*=1) modes. Divided by 10 to obtain the variogram of order 1. The results are stored in the file *fname*.

MODES *grid :* ***1, 0*** *edge* :***1 , [0]*** *depth* : ***8, 16***

REMARK The sill of **vario1**, if exists, is set by means of the variance "**var1** *greyin*".

FUNCTION **vario2**

SYNTAX **vario2** dir st spacing greyin sizemax fname

SUBJECT Variogram of order 2, i.e. semi-variance of the difference f(x+h) -f(x) of the greytone image *greyin* in the direction *dir* (in square or hexagonal grid), of size 0 to *sizemax*, using step *spacing*. The parameter *st* allows to choose between standard (st=0) or geodesic(st=1) modes. The results are stored in the file *fname*.

MODES *grid :* ***1, 0*** *edge* : ***1 , [0]*** *depth* : ***8, 16***

REMARK The sill of **vario1**, if exists, is set by means of the variance "**var1** greyin". In this case, and only in this case, it is a covariance and it is equal to the

variance of greyin minus the variogram of order 2. Otherwise, the covariance doesn't exist (this is why it has not been computed directly).

FUNCTION **modcont**

SYNTAX **modcont** greyin size fname

SUBJECT The continuity module of *greyin* in square grid, or variogram of order ∞. The results are stored in the file *fname*.
modcont greyin := (greyin - greyin⊖ kC) ∨ (greyin⊕ kC - greyin)

MODES *grid : **0*** *edge* : ***1*** *depth* : ***8, 16***

FUNCTION **isogranul**

SYNTAX **isogranul** imin sp sizemax fname

SUBJECT Isotropic size distribution, binary or greytone. The procedure is automatically perfiormed in square grid. Works only in standard mode, according the same considerations as for **granul**. The results are stored in the file *fname*.

MODES *grid : **[1], 0*** *edge* : ***1*** *depth* : ***1, 8, 16***

REMARK Two versions are available: **bisogranul** et **gisogranul .**

SEE ALSO **granul**

FUNCTION **granul**

SYNTAX **granul** imin sp sizemax fname

SUBJECT Size distribution of *imin*, binairy or greytone (according to imin), in the standard or intrinsic mode, on square or hexagonal grid. The results are stored in the file *fname*. The step is given by *sp*, the maximal size is qiven by *sizemax*. In standard mode, the size distribution is given by :

granul (i) = 1000 [1 - **volume** (binin o B_i) / **volume** binin] (binary),

SUBJECT **granul** (i) = 100 [1 - **volume** (greyin o B_i) / **volume** greyin] (greytone).

where B_i is a square or an hexagon of size *i*. When *imin* (or, in greytone mode, its domain) does not touch the borders of field, the standard mode is equivalent to the standard version of the size distribution. When **edge** = 0, it is given by :

granul (i) = 1000 [1 - **volume** (binin o B_i) / **volume** (champ ⊖ B_{2i})], (bin).

granul (i) = 100 [1 - **volume** (greyin o B_i) / **volome** (champ ⊖ B_{2i})], (grey)

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***1, 8, 16***

REMARK The anti-granulometry by closing is constructed by duality, starting from **granul**. Exists in two versions : **bingranul** et **greygranul.**

PROCEDURE **pvonl**

SYNTAX **pvonl** dir spc szmax fname

SUBJECT Returns the area proportion of the eroded versions of *binin*, in direction *dir*, for sizes incrementing by spacing *spc*, from 1 to the maximum size *szmax*. The result are put in file *fname* in the Micromorph directory (*fname* must be written with inverted commas).

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***1***

SEE ALSO **p1vonl, p2vonl**

PROCEDURE **p1vonl**

SYNTAX **p1vonl** dir spc szmax fname

SUBJECT Returns the area cumulative histogram of the intercepts of *binin*, weighted in number, in direction *dir*, for sizes incrementing by spacing *spc*, from 1 to the maximum size *szmax*. The result are put in file *fname* in the Micromorph directory (*fname* must be written with inverted commas).

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***1***

SEE ALSO **pvonl, p2vonl**

PROCEDURE **p2vonl**

SYNTAX **p2vonl** dir sps szmax fname

SUBJECT Returns the area cumulative histogram of the intercepts of *binin*, weighted in length, in direction *dir*, for sizes incrementing by spacing *spc*, from 1 to the maximum size *szmax*. The result are put in file *fname* in the Micromorph directory (*fname* must be written with inverted commas).

MODES *grid :* ***1, 0*** *edge* : ***1, 0*** *depth* : ***1***

SEE ALSO **pvonl, p1vonl**

16. RANDOM SIMULATIONS

16.1 Boolean simulations

PROCEDURE **point**

SYNTAX **point** imout

SUBJECT Places a point in the middle of *imout* (in case of greytone image, the point's value is equal to 100)

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

PROCEDURE **points**

SYNTAX **points** imout n

SUBJECT Simulates n Poisson points in *imout* (for a greytone image, the grey level is equal to 100)

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

PROCEDURE **regpoints**

SYNTAX **regpoints** greyin binout grey_spacing

SUBJECT Simulates Poisson points of density regionalized by the intensities of *greyin.* The positive parameter *grey_spacing* indicates the interval between the levels of grey taken into account. It also controls the number of points. Typically its range is from 1 to 10.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1***

PROCEDURE **isobool**

SYNTAX **isobool** binout size n

SUBJECT Simulates a Boolean set with *n* germs and as the primary grain takes the disc of radius *size.*

MODES *grid :* ***1, 0*** *edge :* ***[1], 0*** *depth :* ***1***

PROCEDURE **isobool2**

SYNTAX **isobool2** binout size n

SUBJECT Simulates a Boolean set with 100 * *n* germs, and as primary grain takes the disc of random radius k * *size* , where k is taken upon a Poisson law of parameter 2.

MODES *grid :* ***1, 0*** *edge :* ***[1], 0*** *depth :* ***1***

EXAMPLE Typical values : *size* = 3 ; n = 2 .

PROCEDURE **eldil**

SYNTAX **eldil** dir binin binout size

SUBJECT Dilates the set binin by an ellipse of size *size* and the direction *dir*. Typical *size*s are 1, 2 or 3.

MODES *grid :* ***[1], 0*** *edge :* ***1, 0*** *depth :* ***1***

REMARK There exists also **dropdil**, in which the structuring element is an asymptotic droplet.

PROCEDURE **elbool**

SYNTAX **elbool** binout size n1 n2

SUBJECT Simulates the union of two Boolean sets having for primary grains the ellipses of **eldil**, of size *size* horizontally oriented and of amount *n1* for the first one, vertically oriented and of amount *n2* for the second one.

MODES *grid :* ***[1], 0*** *edge :* ***1, 0*** *depth :* ***1***

REMARK The parameter *size* should be 1 or 2, because of space limitation. There exists a procedure **dropbool**, where the droplets of **dropdil** (of *size* 1 or 2) are used as the primary grains, and a procedure **tribool** (in hexagonal) with triangular primary grains (useful *size*s from 1 through 30).

PROCEDURE **hard**

SYNTAX **hard** binout1 binout2 size1 size2 n1 n2

SUBJECT Hierarchical simulation, similar to **rose**, but the second set is constrained to be disjoint from the first one, instead of sticking to it.

MODES *grid :* ***[1], 0*** *edge :* ***1*** *depth :* ***1***

EXAMPLE Typical values : *size1* = 10 ; *size2* = 1 ; *n1* = 500 ; *n2* = 70

PROCEDURE **rose**

SYNTAX **rose** binout1 binout2 size1 size2 n1 n2

SUBJECT Hierarchical simulation of two Boolean sets, where the second is reduced to grains that meet the first one. The latter is of the type "**isobool** : *size1 n1*", the former is the union of two "**elbool**: *size2 n2 n2*", in horizontal and vertical directions . Their union is stored in *binout2*, then added to **isobool** iin *binout1*.

MODES *grid :* ***[1], 0*** *edge :* ***1*** *depth :* ***1***

EXAMPLE Typically : *size1* = 15 ; *size2* = (1 or 2) ; *n1* = 50 ; *n2* = 150

PROCEDURE **disjoint**

SYNTAX **disjoint** binin binout size n

SUBJECT Simulates discs of radius *size*, disjoint pairwise and from *binin*. The union of these discs and *binin* is put in *binout*. It is performed by picking at random *n* times the centres of the discs. Every of them is maintained only if the corresponding disk does not meet the preceding ones.

MODES *grid :* ***1, 0*** *edge :* ***1*** *depth :* ***1***

EXAMPLE Typically : *size* = 15 ; *n* = 150 .

PROCEDURE **flake**

SYNTAX **flake** binout size n st

SUBJECT The random version (and not autohomotopic) of the famous Von Koch snow flake, by means of a technique which generalizes the procedure rose. First *n* Poisson's points are dilated by a disc of radius *size*. Then, *n+st* Poisson's points are generated. The points that do not touch the preceding dilated points are removed, a circular dilation of size *size - 1* is performed on the others. This process is repeated tup to the size *1*.

MODES *grid : **1, [0]*** *edge : **1*** *depth : **1***

EXAMPLE Domains of variation : *size* = { 5 trough 25 } ; *n* = { 2 trough 6 } ; *st* = { 10 trough 60 }.Typically : 18, 6, 30

PROCEDURE **conebool**

SYNTAX **conebool** greyout size n

SUBJECT Simulation of hexagonal boolean cones with a slope equal to 1 and a height equal to *size*. n cones are generated.

MODES *grid : **1, [0]*** *edge : **1*** *depth : **8,16***

REMARK n must be less than 250.

SEE ALSO **conebool2** where the primary grains are dodecagonal cones.

PROCEDURE **rocky**

SYNTAX **rocky** greyout greydisplay (shadow of greyout) width n type

SUBJECT Simulation of a random Boolean's function of a rocky basement. Cones with a small slope have either hexagonal bases (*type*=1) or decagonal ones (*type*=0). The altitudes of their summits are uniformly distributed between 0 and *width*, with *n* as density at each level.

MODES *grid : **1,[0]*** *edge : **1, 0*** *depth : **8, 16***

SEE ALSO Typical values : *width* = 10, *n* = 4

16. 2 Lines and partitions

PROCEDURE **lines**

SYNTAX **lines** binout n

SUBJECT Simulates Poisson's horizontal and vertical lines. Their number is *n* in each direction.

MODES *grid :* ***[1], 0*** *edge* : ***1, 0*** *depth* : ***1***

PROCEDURE **diags**

SYNTAX **diags** binout n

SUBJECT Simulates Poisson's lines of isotropic density *n* in each of two diagonal directions of the square and the four directions defined by the *"knight move"*.

MODES *grid :* ***[1], 0*** *edge :* ***1, 0*** *depth :* ***1***

PROCEDURE **steps**

SYNTAX **steps** dir greyout n

SUBJECT Simulates a Poisson's stripes, horizontal (*dir* = 3), or vertical (*dir* = 1), differing by jumps of value 10 on each border.

MODES *grid :* ***[1], 0*** *edge :* ***1*** *depth :* ***8, 16***

REMARK To avoid overflow, one should take $n \leq 24$.

PROCEDURE **nestlines**

SYNTAX **nestlines** binout n1 n2 k

SUBJECT Simulates rectangular Poisson's nestings. Performs a first simulation of **lines** with intensity *n1*, then some rectangles are chosen with probability $1/k$. In each polygon a new simulation of the procedure **lines** is performed with intensity *n2*. Finally, takes their union.

MODES *grid :* ***[1], 0*** *edge* : ***1, 0*** *depth* : ***1***

REMARK Typically : $n2 = 10*n1$ $k = 10$

17. GRAPHS

PROCEDURE **Gdil**

SYNTAX **Gdil** greyin labels greyout size

SUBJECT Generates the dilation of size *size* of grey image, considered as a planar graph of labels *labels*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***8***

REMARK The label image of *greyout* is still labels.

SEE ALSO **Gero, Gopen, Gclose, Gbuild, label**

PROCEDURE **Gero**

SYNTAX **Gero** greyin labels greyout size

SUBJECT Generates the erosion of size *size* of grey image, considered as a planar graph of labels *labels*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***8***

REMARK The label image of *greyout* is still labels.

SEE ALSO **Gdil, Gopen, Gclose, Gbuild, label**

PROCEDURE **Gopen**

SYNTAX **Gopen** greyin labels greyout size

SUBJECT Generates the opening of size *size* of grey image, considered as a planar graph of labels *labels*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***8***

REMARK The label image of *greyout* is still labels.

SEE ALSO **Gdil, Gero, Gclose, Gbuild, label**

PROCEDURE **Gclose**

SYNTAX **Gclose** greyin labels greyout size

SUBJECT Generates the closing of size *size* of grey image, considered as a planar graph of labels *labels*.

MODES *grid : **1, 0** edge : **1, 0** depth : **8***

REMARK The label image of *greyout* is still labels

SEE ALSO **Gdil, Gero, Gopen, Gbuild, label**

PROCEDURE **setborder**

SYNTAX **setborder** imin imout

SUBJECT Sets a thin border (of grey level 0) between the objects of different grey levels.

MODES *grid : **1, 0** edge : **1, 0** depth : **8***

PROCEDURE **Gbuild**

SYNTAX **Gbuild** greyin labelin binmask greyout labelout

SUBJECT Restitution of the planar graph (*greyin*; *labelin*) to its components that hit set *binmask*. The components that miss *binmask* are not taken into account, they are considered as outside of the graph and are regrouped under "label" zero.

MODES *grid : 1, 0 edge : 1, 0 depth : 8*

SEE ALSO **Gero, Gopen, Gclose, Gdil, label**

PROCEDURE **label**

SYNTAX **label** greyin greyout

SUBJECT Generate the label image associated with greyin, and which allows to treat it as a planar graph. The maxima of *greyin* are given label 1 and the maxima of (greyin minus its initial maxima) are given label 2, and so on.

MODES *grid: **1,0** edge: **1, 0** depth: **8***

REMARK 1. Every label is positive. Value zero may be used for the outside of the graph.

2. In all operations handling a graph i.e. **Gdil, Gero, Gopen, Gclose** and **Gbuild**, one have to introduce both the greytone image and its label image.

SEE ALSO **Gero, Gdil, Gopen, Gclose, Gbuild**

18. 3D UTILITIES

PROCEDURE **Seqload**

SYNTAX **Seqload** SeqIn SeqIdStart SeqMemStart NbOfImages

SUBJECT Loads a sequence of *NbOfImages* images to the memory. Loaded images should be named [*SeqIn* i], where i varies from *SeqIdStart* to (*SeqIdStart* + *NbOfImages* - 1). Images are placed from the plane *SeqMemStart* to (*SeqMemStart* + *NbOfImages* - 1). All these planes must be previously allocated (by means of imalloc procedure) and they must have the same depth as loaded images..

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8***

EXEMPLE **Seqload** "c:\wmmorph\seq\toto_" 10 25 50

loads 50 images into computer memory. :

imload *25 "c:\mmorph\seq\toto_10.bmp"*
imload *26 "c:\mmorph\seq\toto_11.bmp"*
...
...
imload *74 "c:\mmorph\seq\toto_59.bmp"*

SEE ALSO **Seqsave**

PROCEDURE **Seqsave**

SYNTAX **Seqsave** SeqIn SeqMemStart SeqStart NbOfImages

SUBJECT Saves a sequence of *NbOfImages* (starting from the image *SeqStart*) to the disk. Stored Images are named [*SeqIn* i] where i varies from *SeqIdStart* to (*SeqIdStart + NbOfImages* - 1).

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8***

EXEMPLE **Seqsave** "c:\wmmorph\seq\toto_" 25 10 50

Saves 50 images into disk.

imsave *25 "c:\mmorph\seq\toto_10.bmp"*
imsave *26 "c:\mmorph\seq\toto_11.bmp"*
...
...
imsave *74 "c:\mmorph\seq\toto_59.bmp"*

SEE ALSO **Seqload**

PROCEDURE **Seqclr**

SYNTAX **Seqclr** SeqInStart NbOfImages

SUBJECT Clears *NbOfImages* starting from the image *SeqInStart*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8***

EXEMPLE **Seqclr** 25 50 - clears 50 images, numbered from 25 to 74.

SEE ALSO **Segload**

PROCEDURE **Seqset**

SYNTAX **Seqset** Seqoutstart NbOfImages value

SUBJECT Affect the constant value value to all images from *Seqoutstart* to *Seqoutstart+NbOfImages-1*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

SEE ALSO **Seqload**

PROCEDURE **Seqcopy**

SYNTAX **Seqcopy** SeqInStart SeqOutStart NbOfImages

SUBJECT Copies *NbOfImages* starting from the image *SeqInStart* to the planes from *SeqOutStart*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqcopy** 25 75 50

copies image 25 to the plane 75, image 26 to the plane 76, 27 to 77 etc. The planes 75-124 must be previously allocated.

SEE ALSO **Seqload**

PROCEDURE **Seqinv**

SYNTAX **Seqinv** SeqInStart SeqOutStart NbOfImages

SUBJECT Inverts each image of the sequence starting from image *SeqInStart* and places it to the sequence starting from image *SeqOutStart*. Both sequences have *NbOfImages* images. All *NbOfImages* of output sequence must be previously allocated.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqinv** 25 75 50

inverts image 25 and places the result into image 75, inverts 26 and places it into 76 etc.i.e.

image 75 = ***iminv*** *(image 25)*
image 76 = ***iminv*** *(image 26)*
....
image 124 = ***iminv*** *(image 74)*

PROCEDURE **Seqadd**

SYNTAX **Seqadd** SeqInStart ImOut NbOfImages

SUBJECT Adds all *NbOfImages* images of the sequence starting from *SeqInStart* and places the result into the single image *ImOut* (usually 16bits image)

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqadd** 25 100 50 ; image 100 = (image 25 + image 26 + ...+ image 74)

SEE ALSO **Segcadd, Seqseqadd**

PROCEDURE **Seqcadd**

SYNTAX **Seqcadd** SeqInStart Const SeqOutStart NbOfImages

SUBJECT Adds to each image of *NbOfImages* of the sequence *SeqInStart* the constant value *Const*. Results are placed starting from image *SeqOutStart*. All *NbOfImages* of output sequence must be previously allocated.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqcadd** 25 10 75 50

adds to the 50 images of the sequence value 10 and places the results starting from image 75 i.e.

image 75 = image 25 + value 10
image 76 = image 26 + value 10
....
....
image 124 = image 74 + value 10

SEE ALSO **Seqcsub, Seqadd, Seqseqadd**

PROCEDURE **Seqseqadd**

SYNTAX **Seqseqadd** SeqIn1Start SeqIn2Start SeqOutStart NbOfImages

SUBJECT Adds *NbOfImages* successive images of first sequence (starting from the image *SeqIn1Start*) to *NbOfImages* successive images of second sequence (starting from the image *SeqIn2Start).* Results are placed in the sequence starting from the image *SeqOutStart*. All *NbOfImages* images of output sequence must be previously allocated.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqseqadd** 10 50 100 40

image 100 = image 10 + image 50
image 101 = image 11 + image 51
...
...
image 139 = image 49 + image 89

SEE ALSO **Seqadd, Seqcadd, Seqseqsub**

PROCEDURE **Seqcsub**

SYNTAX **Seqcsub** SeqInStart Const SeqOutStart NbOfImages

SUBJECT Subtracts the constant value *Const* from each image of *NbOfImages* images of the sequence *SeqInStart*. Results are placed starting from image *SeqOutStart*. All *NbOfImages* of output sequence must be previously allocated.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqcsub** 25 10 75 50

subtracts value 10 from the 50 images of the input sequence.
Places the results starting from image 75 i.e.

image 75 = image 25 - value 10
image 76 = image 26 - value 10
....
....
image 124 = image 74 - value 10

SEE ALSO **Seqcadd, Seqseqsub**

PROCEDURE **Seqseqsub**

SYNTAX **Seqseqsub** SeqIn1Start SeqIn2Start SeqOutStart NbOfImages

SUBJECT Subtract the *NbOfImages* successive images of second sequence (starting from the image *SeqIn2Start*) from *NbOfImages* successive images of first sequence (starting from the image *SeqIn1Start*) . Results are placed starting from the image *SeqOutStart*. All *NbOfImages* of output sequence must be previously allocated.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqseqsub** 10 50 100 40

image 100 = image 10 - image 50
image 101 = image 11 - image 51
...
...
image 139 = image 49 - image 89

SEE ALSO **Seqcsub, Seqseqadd**

PROCEDURE **Seqcmul**

SYNTAX **Seqcmul** Seqinstqrt Seqoutstqrt NbOfImages coeff

SUBJECT Multiplies all the images from *Seqinstart* to *Seqinstqrt+NbOfImages-1* by the integer coeff. Places the result sequencially from the memory nr. *Seqoutstart*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

SEE ALSO **Seqcadd**

PROCEDURE **Seqmean**

SYNTAX **Seqmean** SeqInStart ImOut NbOfImages

SUBJECT Calculates the mean image from all *NbOfImages* images of the sequence starting from the image *SeqInStart*. Result is placed in the image *ImOut*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqmean** 25 90 40

Places the mean of all 40 images of sequence (starting from image 25) into image 90 .i.e.

image 90 = (image 25 + image 26 + ... + image 74) / value 40

EXEMPLE

SEE ALSO **SeqSeqmean**

PROCEDURE **Seqseqmean**

SYNTAX **Seqseqmean** SeqIn1Start SeqIn2Start SeqOutStart NbOfImages

SUBJECT Calculates the mean for each image of sequence starting from the image *SeqIn1Start* and the corresponding image of sequence starting from *SeqIn2Start*. Result is placed in the sequence *SeqOutStart*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqseqmean** 10 50 100 40

performs the following 40 operations :

image 100 = (image 10 + image 50) / 2
image 101 = (image 11 + image 51) / 2
...
...
image 139 = (image 59 + image 89) / 2

SEE ALSO **Seqmean**

PROCEDURE **Seqdiff**

SYNTAX **Seqdiff** SeqInStart ImRef SeqOutStart NbOfImages

SUBJECT Calculates module between each of *NbOfImages* images from Sequence *SegInStart* and the image *ImRef*. Results are placed in the sequence starting from *SeqOutStart*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqdiff** 25 75 80 40

performs 40 following operations :

image 80 = | image 25 - image 75 |
image 81 = | image 26 - image 75 |
image 82 = | image 27 - image 75 |
...
...
image 119 = | image 64 - image 75 |

FUNCTION **Seqvolume**

SYNTAX **Seqvolume** SeqInStart NbOfImages Param

SUBJECT Calculates the volume of each image of sequence *SeqInStart* and subtracts the value *Param* from each result. The function returns the sum of *NbOfImages* such operations .

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE *result :=* **Seqvolume** 10 40 10000

performs the following operations :

a1 := imvolume (image 10) - 10000
a2 = imvolume (image 11) - 10000
...
a40 = imvolume (image 49) - 10000
result *= (a1 + a2 + ... a40)*

REMARK Parameter *Param* is introduced in order to decrease the obtained result. MicroMorph can use the integer numbers that are not smaller than -2.1E9 and not higher than 2.1E9. By summing the volumes of greytone sequence one can easily cross the upper range.

PROCEDURE **Seqinf**

SYNTAX **Seqinf** SeqInStart ImOut NbOfImages

SUBJECT Calculates inferior of all *NbOfImages* images of the sequence *SeqInStart*. Result is placed in the image *ImOut*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqinf** 10 80 40

performs the following operation :

image 80 = ***iminf*** *(image10, image11, image12, ..., image49)*

SEE ALSO **Seqseqinf, Seqsup**

PROCEDURE **Seqseqinf**

SYNTAX **Seqseqinf** SeqIn1Start SeqIn2Start SeqOutStart NbOfImages

SUBJECT Calculates inferior for each image from the sequence *SeqIn1Start* and the corresponding image from the sequence *SeqIn2Start*. Places the results in the sequence *SeqOutStart*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqseqinf** 10 50 100 40

performs the following operations :

image100 = ***iminf*** *(image10 image50)*
image101 = ***iminf*** *(image11 image51)*
...

...
image139 = ***iminf*** *(image49 image89)*

SEE ALSO **Seqinf, Seqseqsup**

PROCEDURE **Seqsup**

SYNTAX **Seqsup** SeqInStart ImOut NbOfImages

SUBJECT Calculates superior of all *NbOfImages* images of the sequence *SeqInStart*. Result is placed in the image *ImOut*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqsup** 10 80 40

performs the following operation :
image 80 = ***imsup*** *(image10, image11, image12, ..., image49)*

SEE ALSO **Seqseqsup, Seqinf**

PROCEDURE **Seqseqsup**

SYNTAX **Seqseqsup** SeqIn1Start SeqIn2Start SeqOutStart NbOfImages

SUBJECT Calculates superior for each image from the sequence *SeqIn1Start* and the corresponding image from the sequence *SeqIn2Start*. Places the results in the sequence *SeqOutStart*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqseqsup** 10 50 100 40

performs the following operations :
image100 = ***imsup*** *(image10 image50)*
image101 = ***imsup*** *(image11 image51)*
...
...
image139 = ***imsup*** *(image49 image89)*

SEE ALSO **Seqsup, Seqseqinf**

PROCEDURE **Seqbuild**

SYNTAX **Seqbuild** SeqMaskStart SeqInOutStart NbOfImages

SUBJECT Builds each image from the sequence *SeqInOutStart* into the corresponding mask from the sequence *SeqMaskStart*. Places the result into the sequence *SeqInOutStart*. Both sequences must have *NbOfImages* images.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqbuild** 10 80 40

performs the following operations :

build *10 80*
build *11 81*
...
...
build *49 119*

PROCEDURE **Seqthresh**

SYNTAX **Seqthresh** SeqInStart LoTh HiTh SeqOutStart NbOfImages

SUBJECT Performs thresholding for each image of the greytone sequence *SeqInStart* and places the results into the binary sequence *SeqOutStart*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqthresh** 10 0 5 50 40

performs the following operations :

imthresh *10 0 5 50*
imthresh *11 0 5 51*
...
...
imthresh *49 0 5 89*

PROCEDURE **Seqgrad**

SYNTAX **Seqgrad** SeqInStart SeqOutStart Size NbOfImages

SUBJECT For each image of the sequence *SeqInStart* calculates the morphological gradient of size *Size*. The result is placed in the sequence *SeqOutStart*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

EXEMPLE **Seqgrad** 10 50 2 40

performs the following operations :

gradient *10 50 2*
gradient *11 51 2*
...
...
gradient *49 89 2*

PROCEDURE **Seqhist**

SYNTAX **Seqhist** SeqInStart NbOfImages

SUBJECT For a selected point (by mouse), draws the changes of its grey level along the sequence.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

PROCEDURE **Scroll**

SYNTAX **Scroll** SeqInStart NbOfImages temp NbTimes

SUBJECT Scrolls the sequence starting at memory *SeqInStart* and of lenght *NbOfImages*. Parameter *Nbtimes* indicates the number of times the sequence will be scrolled. Parameter *temp* regulates the speed :

temp = 0 maximum speed (no temporisation)
temp = 1 step by step display
temp > 0 the delay between images increases with the temp value

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8***

SEE ALSO **Scroll2, Scroll3, Visu**

PROCEDURE **Scroll2**

SYNTAX **Scroll2** SeqIn1Start SeqIn2Start NbOfImages temp spacing NbTimes

SUBJECT Scrolls two sequences starting at memory *SeqIn1Start* and *SeqIn2Start*, of lenght *NbOfImages*. Parameter *temp* regulates the speed :

temp = 0 maximum speed (no temporisation)
temp = 1 step by step display
temp > 0 the delay between images increases with the temp value

When parameter *spacing* = 1 all images are taken,
when *spacing* <> 1, only odd images are taken.
Parameter *Nbtimes* indicates the number of times the sequence will be scrolled.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8***

SEE ALSO **Scroll, Scroll3, Visu**

PROCEDURE **Scroll3**

SYNTAX **Scroll3** SeqIn1Start SeqIn2Start SeqIn3Start NbOfImages temp spacing NbTimes

SUBJECT Scrolls three sequences starting at memory *SeqIn1Start*, *SeqIn2Start* and *SeqIn3Start*, of lenght *NbOfImages*. Parameter temp regulates the speed :

temp = 0 maximum speed (no temporisation)
temp = 1 step by step display
temp > 0 the delay between images increases with the temp value

When parameter *spacing* = 1 all images are taken,
when *spacing* <> 1, only odd images are taken.
Parameter *Nbtimes* indicates the number of times the sequence will be scrolled.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8***

SEE ALSO **Scroll, Scroll2, Visu**

PROCEDURE **Visu**

SYNTAX **Visu** SeqInStart NbOfImages greyout light k

SUBJECT Generates a perspective display of the binary sequence that begins at *SeqInStart* and ends at *SeqInStart+NbOfImages-1*, considered as a stack of the successive sections in a 3D material. It is seen either from above (parameter *k*=1) or from below (*k*<>1). The lightens of the display is indicated by *light* (between 0 and 100), and the perspective image is shown in *greyout*.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1***

SEE ALSO **Scroll, Scroll2, Scroll3**

19. 3D PROCEDURES

PROCEDURE **ero3D**

SUBJECT **ero3D** SeqInStart SeqOutStart Size NbOfImages

SUBJECT Performs a 3D erosion by a cube or cuboctahedron of size *Size* of sequence of *NbOfImages* images, starting from label *SeqInStart*. The result is placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

EXEMPLE **ero3D** 10 50 2 40

sequence of images 10...49 is eroded by a cube or cuboctahedron of size 2. Result is placed into the images 50...89.

SEE ALSO **dil3D**

PROCEDURE **dil3D**

SYNTAX **dil3D** SeqInStart SeqOutStart Size NbOfImages

SUBJECT Performs a 3D dilation by a cube or cuboctahedron of size *Size* of sequence of *NbOfImages* images, starting from label *SeqInStart*. The result is placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

EXEMPLE **dil3D** 10 50 2 40

sequence of images 10...49 is dilated by a cube or cuboctahedron of size 2. Result is placed into the images 50...89.

SEE ALSO **ero3D**

PROCEDURE **open3D**

SYNTAX **open3D** SeqInStart SeqOutStart Size NbOfImages

SUBJECT Performs a 3D opening by a a cube or cuboctahedron of size *Size* of sequence of *NbOfImages* images, starting from label *SeqInStart.* The result is placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

EXEMPLE **open3D** 10 50 2 40

Sequence of images 10...49 is eroded and dilated by a a cube or cuboctahedron of size 2. Result is placed into the images 50...89.

SEE ALSO **close3D**

PROCEDURE **close3D**

SYNTAX **close3D** SeqInStart SeqOutStart Size NbOfImages

SUBJECT Performs a 3D closing by a cube or cuboctahedron of size *Size* of sequence of *NbOfImages* images, starting from label *SeqInStart*. The result is placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

EXEMPLE **close3D** 10 50 2 40

Sequence of images 10...49 is dilated and eroded by a cube or cuboctahedron of size 2. Result is placed into the images 50...89.

SEE ALSO **open3D**

PROCEDURE **gradient3D**

SYNTAX **gradient3D** SeqInStart SeqOutStart Size NbOfImages

SUBJECT Calculates a 3D gradient of size *Size* of sequence of *NbOfImages* images, starting from label *SeqInStart*. The result is placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

EXEMPLE **gradient3D** 10 50 2 40

3D gradient of size 2 is calculated for a sequence of images 10...49. Result is placed into the images 50...89.

PROCEDURE **build3D**

SYNTAX **build3D** SeqMaskStart SeqInOutStart NbOfImages

SUBJECT Performs a 3D reconstruction of sequence starting from label *SeqInOutStart* into mask starting from label *SeqMaskStart* The result is placed starting from the image labelled *SeqInOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

EXEMPLE **build3D** 10 50 40

Sequence of images 10...49 is reconstructed into sequence of images 50...89. Result is placed into the images 10...49.

PROCEDURE **buildopen3D**

SYNTAX **buildopen3D** SeqInStart SeqOutStart Size NbOfImages

SUBJECT Performs a 3D opening by reconstruction, by a cube (or cubocthedron) of size *Size* of sequence of *NbOfImages* images, starting from label *SeqInStart*. The result is placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

EXEMPLE **buildopen3D** 10 50 2 40

3D opening by reconstruction of size 2 of the sequence of images 10...49 is performed. Result is placed into the images 50...89.

SEE ALSO **buildclose3D**

PROCEDURE **buildclose3D**

SYNTAX **buildclose3D** SeqInStart SeqOutStart Size NbOfImages

SUBJECT Performs a 3D by reconstruction by a cube (or cubocthedron) of size *Size* of sequence of *NbOfImages* images, starting from label *SeqInStart*. The result is placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

EXEMPLE **buildclose3D** 10 50 2 40

3D closing by reconstruction of size 2 of the sequence of images 10...49 is performed. Result is placed into the images 50...89.

SEE ALSO **buildclose3D**

PROCEDURE **openth3D**

SYNTAX **openth3D** SeqInStart SeqOutStart Size NbOfImages

SUBJECT Residue of the 3D opening **open3D** of size *Size* of the sequence of *NbOfImages* starting from label *SeqInStart*. The results are placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

SEE ALSO **closeth3D**

PROCEDURE **closeth3D**

SYNTAX **closeth3D** SeqInStart SeqOutStart Size NbOfImages

SUBJECT Residue of the 3D closing **close3D** of size *Size* of the sequence of *NbOfImages* starting from label *SeqInStart*. The results are placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

SEE ALSO **openth3D**

PROCEDURE **maxima3D**

SYNTAX **maxima3D** SeqInStart SeqOutStart NbOfImages

SUBJECT Calculates 3D maxima for the sequence of *NbOfImages* starting from label *SeqInStart*. The results (binary) are placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

SEE ALSO **minima3D**

PROCEDURE **minima3D**

SYNTAX **minima3D** SeqInStart SeqOutStart NbOfImages

SUBJECT Calculates 3D minima for the sequence of *NbOfImages* starting from label *SeqInStart*. The results (binary) are placed starting from the image labelled *SeqOutStart*.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

SEE ALSO **maxima3D**

PROCEDURE **af3D**

SYNTAX **af3D** SeqInStart SeqOutStart Size NbOfImages Type

SUBJECT 3D alternating filter of size *Size*. The *NbOfImages* images of input sequence is labelled from *SeqInStart*. The output sequence starts from *SeqOutStart*. If parameter *Type* = 1 then **close3D** is followed by **open3D**, otherwise **open3D** is followed by **close3D**.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

SEE ALSO **asf3D**

PROCEDURE **asf3D**

SYNTAX **asf3D** SeqInStart SeqOutStart Size NbOfImages Type

SUBJECT 3D alternating sequential filter of size *Size*. The *NbOfImages* images of input sequence is labelled from *SeqInStart*. The output sequence starts from *SeqOutStart*. If parameter *Type* = 1 then **close3D** is followed by **open3D**, otherwise **open3D** is followed by **close3D**.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

SEE ALSO **af3D**

PROCEDURE **buildaf3D**

SYNTAX **buildaf3D** SeqInStart SeqOutStart Size NbOfImages Type

SUBJECT 3D alternating filter by reconstruction of size *Size*. The *NbOfImages* images of input sequence is labelled from *SeqInStart*. The output sequence starts from *SeqOutStart*. If parameter *Type* = 1 then **buildclose3D** is followed by **buildopen3D**, otherwise **buildopen3D** is followed by **buildclose3D**.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

SEE ALSO **buildasf3D**

PROCEDURE **buildasf3D**

SYNTAX **buildasf3D** SeqInStart SeqOutStart Size NbOfImages Type

SUBJECT 3D alternating sequential filter of size *Size*. The *NbOfImages* images of input sequence is labelled from *SeqInStart*. The output sequence starts from *SeqOutStart*. If parameter *Type* = 1 then **buildclose3D** is followed by **buildopen3D**, otherwise **buildopen3D** is followed by **buildclose3D**.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8, 16***

SEE ALSO **buildaf3D**

PROCEDURE **timdil**

SYNTAX **timdil** SeqInOutStart size NbOfImages

SUBJECT Dilation of the sequence beginning at *SeqInOutStart* and of the lenght *NbOfImages* by an orthogonal segment of lenght *size*. The origin is at the bottom extremity of the segment. The result is placed in the same memories as the initial sequence.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8***

SEE ALSO **timero, timopen, timclose**

PROCEDURE **timero**

SYNTAX **timero** SeqInOutStart size NbOfImages

SUBJECT Erosion of the sequence beginning at *SeqInOutStart* and of the lenght *NbOfImages* by an orthogonal segment of lenght *size*. The origin is at the top extremity of the segment. The result is placed in the same memories as the initial sequence.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8***

SEE ALSO **timdil, timopen, timclose**

PROCEDURE **timopen**

SYNTAX **timopen** SeqInOutStart size NbOfImages

SUBJECT Opening of the sequence beginning at *SeqInOutStart* and of the lenght *NbOfImages* by an orthogonal segment of lenght *size*. The result is placed in the same memories as the initial sequence.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8***

SEE ALSO **timero, timdil, timclose**

PROCEDURE **timclose**

SYNTAX **timclose** SeqInOutStart size NbOfImages

SUBJECT Closing of the sequence beginning at *SeqInOutStart* and of the lenght *NbOfImages* by an orthogonal segment of lenght *size*. The result is placed in the same memories as the initial sequence.

MODES *grid : **[1], 0** edge : **1, 0** depth : **1, 8***

SEE ALSO **timero, timdil, timclose**

20. MOUSE

PROCEDURE **fill**

SYNTAX **fill** imin lev

SUBJECT The user clicks with the mouse at the angles of a polygonal line, whose inside is replaced by the constant value *lev*. Ends by a triple click.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

PROCEDURE **dline**

SYNTAX **dline** imin lev

SUBJECT Draw on image *imin* one (or more) segments of grey value *lev*, whose extremities are introduced via the mouse. Ends by a triple click.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

PROCEDURE **delobj**

SYNTAX **delobj** imin

SUBJECT In the binary image *imin*, suppress the connected component indicated by a click of the mouse. Ends by a triple click.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1***

PROCEDURE **pointinfo**

SYNTAX **pointinfo** imin

SUBJECT When the users clicks at a point of image *imin*, the procedure provides the coordinates of the point and its grey value. They are displayed in the title zone of *imin* image. If *imin* is a color image the value given indicate the corresponding grey of the palette. Ends by a triple click.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

PROCEDURE **binpointer**

SYNTAX **binpointer** binin binout type

SUBJECT Places in *binout* the particle of *binin* which has been clicked if the *type* values 1, or the point itself, if *type* values 0. Ends by a triple click.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1***

PROCEDURE **objinfo**

SYNTAX **objinfo** binin

SUBJECT When clicking on a given binary particle of *binin*, indicates the area of the particle in the title zone above the display image *binin*. Ends by a triple click.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1***

PROCEDURE **brush**

SYNTAX **brush** imin lev size

SUBJECT Places the point of level *lev* and dilates them by size *size*. May be used either for writing, or for deleting particles. Ends by a triple click.

MODES *grid :* ***1, 0*** *edge :* ***1, 0*** *depth :* ***1, 8, 16***

Index of procedures.

1. UTILITIES

2. DISPLAY TOOLS

3. EROSIONS, DILATIONS

4. RANK OPERATIONS

5. CONVOLUTIONS

6. OPENINGS, CLOSINGS

7. GEODESY AND CONNECTIVITY

8. APPLICATIONS OF GEODESY

9. FILTERS

10 MAXIMAL BALLS AND SKELRTON

11. THINNINGS THICKENINGS

12. WATERSHEDS

13. SEGMENTATION

14. MEASURES

15. CURVES

16. RANDOM SIMULATIONS

17. GRAPHS

18. 3DUTILITIES

19. 3D PROCEDURES

20. MOUSE

COMPOSED WORDS ALPHABETIC LIST