



mambaApi Reference Manual

Author:
Nicolas BEUCHER

August 23, 2009

Contents

1	Data Structures	11
2	File List	11
3	MB_Image Struct Reference	13
3.1	Detailed Description	13
3.2	Field Documentation	13
3.2.1	BITPPIX	13
3.2.2	NUMCOL	13
3.2.3	PIXARRAY	13
3.2.4	PLINES	13
4	MB_ListControl Struct Reference	15
4.1	Detailed Description	15
4.2	Field Documentation	15
4.2.1	firstx	15
4.2.2	firsty	15
4.2.3	lastx	15
4.2.4	lasty	15
5	MB-Token Struct Reference	16
5.1	Detailed Description	16
5.2	Field Documentation	16
5.2.1	nextx	16
5.2.2	nexty	16
5.2.3	tag	16
6	mambaApi.h File Reference	17
6.1	Detailed Description	19
6.2	Define Documentation	19
6.2.1	MB_LINE_COUNT	19
6.2.2	MB_LINE_OFFSET	19
6.3	Function Documentation	19
6.3.1	MB_Abs32	19
6.3.2	MB_Add	19
6.3.3	MB_And	19
6.3.4	MB_BinHitOrMiss	20
6.3.5	MB_BldNb8	20
6.3.6	MB_BldNbb	20
6.3.7	MB_Check	21
6.3.8	MB_Compare	21
6.3.9	MB_ConAdd	21
6.3.10	MB_ConDiv	21
6.3.11	MB_ConMul	21
6.3.12	MB_ConSet	22
6.3.13	MB_ConSub	22
6.3.14	MB_Convert	22
6.3.15	MB_Copy	22
6.3.16	MB_CopyBitPlane	22
6.3.17	MB_CopyBytePlane	23
6.3.18	MB_Create	23
6.3.19	MB_depthRange	23
6.3.20	MB_Destroy	23
6.3.21	MB_Diff	23
6.3.22	MB_DiffNb8	24
6.3.23	MB_DiffNbb	24
6.3.24	MB_Distanceb	24

6.3.25	MB_DualBldNb8	25
6.3.26	MB_DualBldNbb	25
6.3.27	MB_Extract	25
6.3.28	MB_Extrema	25
6.3.29	MB_Frame	26
6.3.30	MB_GetPixel	26
6.3.31	MB_Histo	26
6.3.32	MB_Inf	27
6.3.33	MB_InfNb32	27
6.3.34	MB_InfNb8	27
6.3.35	MB_InfNbb	28
6.3.36	MB_Inv	28
6.3.37	MB_Labelb	28
6.3.38	MB_Load	28
6.3.39	MB_Lookup	29
6.3.40	MB_Mask	29
6.3.41	MB_Mul	29
6.3.42	MB_Or	29
6.3.43	MB_PutPixel	30
6.3.44	MB_Range	30
6.3.45	MB_ResetWindow	30
6.3.46	MB_SetSizeAndWindow	30
6.3.47	MB_SetWindow	30
6.3.48	MB_Shift32	31
6.3.49	MB_Shiftb	31
6.3.50	MB_Sub	31
6.3.51	MB_Sup	31
6.3.52	MB_SupNb32	32
6.3.53	MB_SupNb8	32
6.3.54	MB_SupNbb	32
6.3.55	MB_Thresh	33
6.3.56	MB_Volume	33
6.3.57	MB_Watershed	33
6.3.58	MB_Xor	33
6.4	Variable Documentation	34
6.4.1	MB_h	34
6.4.2	MB_refcounter	34
6.4.3	MB_sz_x	34
6.4.4	MB_sz_y	34
6.4.5	MB_w	34
6.4.6	MB_wx_size	34
6.4.7	MB_wx_start	34
6.4.8	MB_wy_size	34
6.4.9	MB_wy_start	34
7	mambaApi_loc.h File Reference	35
7.1	Detailed Description	35
7.2	Define Documentation	35
7.2.1	MB_PROBE_PAIR	35
7.3	Variable Documentation	35
7.3.1	MB_HierarchicalList	35
7.3.2	MB_Label_CEQ	35
7.3.3	MB_Label_EQ	35
7.3.4	MB_Label_maxEQ	35
7.3.5	MB_TokensArray	35
7.3.6	MB_VolumePerByte	36

8	mambaCommon.h File Reference	37
8.1	Detailed Description	37
8.2	Define Documentation	37
8.2.1	MB_X_LEFT	37
8.2.2	MB_X_RIGHT	38
8.2.3	MB_Y_BOTTOM	38
8.2.4	MB_Y_TOP	38
8.3	Typedef Documentation	38
8.3.1	PIX32	38
8.3.2	PIX8	38
8.3.3	PLINE	38
8.3.4	PLINE32	38
8.3.5	Sint16	38
8.3.6	Sint32	38
8.3.7	Sint8	38
8.3.8	Uint16	38
8.3.9	Uint32	38
8.3.10	Uint8	38
9	MB_Abs32.c File Reference	39
9.1	Detailed Description	39
9.2	Function Documentation	39
9.2.1	MB_Abs32	39
10	MB_Add.c File Reference	40
10.1	Detailed Description	40
10.2	Function Documentation	40
10.2.1	MB_Add	40
11	MB_And.c File Reference	41
11.1	Detailed Description	41
11.2	Function Documentation	41
11.2.1	MB_And	41
12	MB_BinHitOrMiss.c File Reference	42
12.1	Detailed Description	42
12.2	Function Documentation	42
12.2.1	MB_BinHitOrMiss	42
13	MB_BldNb8.c File Reference	43
13.1	Detailed Description	43
13.2	Typedef Documentation	43
13.2.1	TSWITCHEP	43
13.3	Function Documentation	43
13.3.1	MB_BldNb8	43
14	MB_BldNbb.c File Reference	44
14.1	Detailed Description	44
14.2	Typedef Documentation	44
14.2.1	TSWITCHEP	44
14.3	Function Documentation	44
14.3.1	MB_BldNbb	44
15	MB_Check.c File Reference	45
15.1	Detailed Description	45
15.2	Function Documentation	45
15.2.1	MB_Check	45

16 MB_Conpare.c File Reference	46
16.1 Detailed Description	46
16.2 Function Documentation	46
16.2.1 MB_Conpare	46
17 MB_ConAdd.c File Reference	47
17.1 Detailed Description	47
17.2 Function Documentation	47
17.2.1 MB_ConAdd	47
18 MB_ConDiv.c File Reference	48
18.1 Detailed Description	48
18.2 Function Documentation	48
18.2.1 MB_ConDiv	48
19 MB_ConMul.c File Reference	49
19.1 Detailed Description	49
19.2 Function Documentation	49
19.2.1 MB_ConMul	49
20 MB_ConSet.c File Reference	50
20.1 Detailed Description	50
20.2 Function Documentation	50
20.2.1 MB_ConSet	50
21 MB_ConSub.c File Reference	51
21.1 Detailed Description	51
21.2 Function Documentation	51
21.2.1 MB_ConSub	51
22 MB_Convert.c File Reference	52
22.1 Detailed Description	52
22.2 Function Documentation	52
22.2.1 MB_Convert	52
22.2.2 MB_Convert1to8	52
22.2.3 MB_Convert8to1	52
23 MB_CopyBitPlane.c File Reference	53
23.1 Detailed Description	53
23.2 Function Documentation	53
23.2.1 EXTRACT_BIT_PLANE	53
23.2.2 MB_CopyBitPlane	53
23.2.3 SET_BIT_PLANE	53
24 MB_CopyBytePlane.c File Reference	54
24.1 Detailed Description	54
24.2 Function Documentation	54
24.2.1 MB_CopyBytePlane	54
25 MB_Create.c File Reference	55
25.1 Detailed Description	55
25.2 Function Documentation	55
25.2.1 MB_Create	55
25.2.2 MB_Destroy	55
25.3 Variable Documentation	55
25.3.1 MB_refcounter	55

26 MB_Diff.c File Reference	56
26.1 Detailed Description	56
26.2 Function Documentation	56
26.2.1 MB_Diff	56
27 MB_DiffNb8.c File Reference	57
27.1 Detailed Description	57
27.2 Typedef Documentation	57
27.2.1 TSWITCHEP	57
27.3 Function Documentation	57
27.3.1 MB_DiffNb8	57
28 MB_DiffNbb.c File Reference	58
28.1 Detailed Description	58
28.2 Typedef Documentation	58
28.2.1 TSWITCHEP	58
28.3 Function Documentation	58
28.3.1 MB_DiffNbb	58
29 MB_Distanceb.c File Reference	59
29.1 Detailed Description	59
29.2 Typedef Documentation	59
29.2.1 TSWITCHEP	59
29.3 Function Documentation	59
29.3.1 MB_Distanceb	59
30 MB_DualBldNb8.c File Reference	60
30.1 Detailed Description	60
30.2 Typedef Documentation	60
30.2.1 TSWITCHEP	60
30.3 Function Documentation	60
30.3.1 MB_DualBldNb8	60
31 MB_DualBldNbb.c File Reference	61
31.1 Detailed Description	61
31.2 Typedef Documentation	61
31.2.1 TSWITCHEP	61
31.3 Function Documentation	61
31.3.1 MB_DualBldNbb	61
32 MB_Error.c File Reference	62
32.1 Detailed Description	62
32.2 Function Documentation	62
32.2.1 MB_StrErr	62
32.3 Variable Documentation	62
32.3.1 err_str	62
33 MB_error.h File Reference	63
33.1 Detailed Description	63
33.2 Enumeration Type Documentation	63
33.2.1 MB_errcode	63
33.3 Function Documentation	63
33.3.1 MB_StrErr	63
34 MB_Extrema.c File Reference	64
34.1 Detailed Description	64
34.2 Function Documentation	64
34.2.1 MB_Extrema	64

35	MB_Frame.c File Reference	65
35.1	Detailed Description	65
35.2	Function Documentation	65
35.2.1	MB_Frame	65
36	MB_Histo.c File Reference	66
36.1	Detailed Description	66
36.2	Function Documentation	66
36.2.1	MB_Histo	66
37	MB_Inf.c File Reference	67
37.1	Detailed Description	67
37.2	Function Documentation	67
37.2.1	MB_Inf	67
38	MB_InfNb32.c File Reference	68
38.1	Detailed Description	68
38.2	Typedef Documentation	68
38.2.1	TSWITCHEP	68
38.3	Function Documentation	68
38.3.1	MB_InfNb32	68
39	MB_InfNb8.c File Reference	69
39.1	Detailed Description	69
39.2	Typedef Documentation	69
39.2.1	TSWITCHEP	69
39.3	Function Documentation	69
39.3.1	MB_InfNb8	69
40	MB_InfNbb.c File Reference	70
40.1	Detailed Description	70
40.2	Typedef Documentation	70
40.2.1	TSWITCHEP	70
40.3	Function Documentation	70
40.3.1	MB_InfNbb	70
41	MB_Inv.c File Reference	71
41.1	Detailed Description	71
41.2	Function Documentation	71
41.2.1	MB_Inv	71
42	MB_Labelb.c File Reference	72
42.1	Detailed Description	72
42.2	Typedef Documentation	72
42.2.1	TSWITCHEP	72
42.3	Function Documentation	72
42.3.1	MB_Labelb	72
42.4	Variable Documentation	72
42.4.1	MB_Label_CEQ	72
42.4.2	MB_Label_EQ	73
42.4.3	MB_Label_maxEQ	73
43	MB_LoadExtract.c File Reference	74
43.1	Detailed Description	74
43.2	Function Documentation	74
43.2.1	MB_Copy	74
43.2.2	MB_Extract	74
43.2.3	MB_Extract32	74
43.2.4	MB_Extract8	75
43.2.5	MB_Load	75

43.2.6	MB_Load32	75
43.2.7	MB_Load8	75
44	MB_Lookup.c File Reference	76
44.1	Detailed Description	76
44.2	Function Documentation	76
44.2.1	MB_Lookup	76
45	MB_Mask.c File Reference	77
45.1	Detailed Description	77
45.2	Function Documentation	77
45.2.1	MB_Mask	77
46	MB_Mul.c File Reference	78
46.1	Detailed Description	78
46.2	Function Documentation	78
46.2.1	MB_Mul	78
47	MB_ngb.h File Reference	79
47.1	Detailed Description	79
47.2	Define Documentation	79
47.2.1	BIN_FILL_VALUE	79
47.2.2	GREY_FILL_VALUE	79
47.2.3	I32_FILL_VALUE	79
47.3	Enumeration Type Documentation	79
47.3.1	MB_bordermode_t	79
47.3.2	MB_grid_t	79
48	MB_Or.c File Reference	80
48.1	Detailed Description	80
48.2	Function Documentation	80
48.2.1	MB_Or	80
49	MB_Pixel.c File Reference	81
49.1	Detailed Description	81
49.2	Function Documentation	81
49.2.1	MB_GetPixel	81
49.2.2	MB_PutPixel	81
50	MB_Range.c File Reference	82
50.1	Detailed Description	82
50.2	Function Documentation	82
50.2.1	MB_depthRange	82
50.2.2	MB_Range	82
51	MB_Sub.c File Reference	83
51.1	Detailed Description	83
51.2	Function Documentation	83
51.2.1	MB_Sub	83
52	MB_Sup.c File Reference	84
52.1	Detailed Description	84
52.2	Function Documentation	84
52.2.1	MB_Sup	84

53 MB_SupNb32.c File Reference	85
53.1 Detailed Description	85
53.2 Typedef Documentation	85
53.2.1 TSWITCHEP	85
53.3 Function Documentation	85
53.3.1 MB_SupNb32	85
54 MB_SupNb8.c File Reference	86
54.1 Detailed Description	86
54.2 Typedef Documentation	86
54.2.1 TSWITCHEP	86
54.3 Function Documentation	86
54.3.1 MB_SupNb8	86
55 MB_SupNbb.c File Reference	87
55.1 Detailed Description	87
55.2 Typedef Documentation	87
55.2.1 TSWITCHEP	87
55.3 Function Documentation	87
55.3.1 MB_SupNbb	87
56 MB_Thresh.c File Reference	88
56.1 Detailed Description	88
56.2 Function Documentation	88
56.2.1 MB_Thresh	88
57 MB_Volume.c File Reference	89
57.1 Detailed Description	89
57.2 Function Documentation	89
57.2.1 MB_Volume	89
57.3 Variable Documentation	89
57.3.1 MB_VolumePerByte	89
58 MB_Watershed.c File Reference	90
58.1 Detailed Description	90
58.2 Define Documentation	90
58.2.1 NORM_WATER_LEVEL	90
58.3 Typedef Documentation	90
58.3.1 TSWITCHEP	90
58.4 Function Documentation	90
58.4.1 MB_Watershed	90
58.5 Variable Documentation	91
58.5.1 current_water_level	91
58.5.2 hxNbDir	91
58.5.3 MB_HierarchicalList	91
58.5.4 MB_TokensArray	91
58.5.5 sqNbDir	91
59 MB_Window.c File Reference	92
59.1 Detailed Description	92
59.2 Define Documentation	92
59.2.1 MB_MAX_IMAGE_X_SIZE	92
59.2.2 MB_MAX_IMAGE_Y_SIZE	92
59.2.3 MB_ROUND_X	92
59.2.4 MB_ROUND_Y	92
59.3 Function Documentation	92
59.3.1 MB_ResetWindow	92
59.3.2 MB_SetSizeAndWindow	93
59.3.3 MB_SetWindow	93

59.4	Variable Documentation	93
59.4.1	MB_h	93
59.4.2	MB_sz_x	93
59.4.3	MB_sz_y	93
59.4.4	MB_w	93
59.4.5	MB_wx_size	93
59.4.6	MB_wx_start	93
59.4.7	MB_wy_size	93
59.4.8	MB_wy_start	93
60	MB_Xor.c File Reference	94
60.1	Detailed Description	94
60.2	Function Documentation	94
60.2.1	MB_Xor	94

List of Figures

1	Image structure and variables	14
---	-------------------------------	----

Data Structure Index

1 Data Structures

Here are the data structures with brief descriptions:

<code>MB_Image</code>	13
<code>MB_ListControl</code>	15
<code>MB_Token</code>	16
File Index	

2 File List

Here is a list of all documented files with brief descriptions:

<code>mambaApi.h</code>	17
<code>mambaApi_loc.h</code>	35
<code>mambaCommon.h</code>	37
<code>MB_Abs32.c</code>	39
<code>MB_Add.c</code>	40
<code>MB_And.c</code>	41
<code>MB_BinHitOrMiss.c</code>	42
<code>MB_BldNb8.c</code>	43
<code>MB_BldNbb.c</code>	44
<code>MB_Check.c</code>	45
<code>MB_Compare.c</code>	46
<code>MB_ConAdd.c</code>	47
<code>MB_ConDiv.c</code>	48
<code>MB_ConMul.c</code>	49
<code>MB_ConSet.c</code>	50
<code>MB_ConSub.c</code>	51
<code>MB_Convert.c</code>	52
<code>MB_CopyBitPlane.c</code>	53
<code>MB_CopyBytePlane.c</code>	54
<code>MB_Create.c</code>	55
<code>MB_Diff.c</code>	56
<code>MB_DiffNb8.c</code>	57

MB_DiffNb.c	58
MB_Distance.c	59
MB_DualBldNb8.c	60
MB_DualBldNb.c	61
MB_Error.c	62
MB_error.h	63
MB_Extrema.c	64
MB_Frame.c	65
MB_Histo.c	66
MB_Inf.c	67
MB_InfNb32.c	68
MB_InfNb8.c	69
MB_InfNb.c	70
MB_Inv.c	71
MB_Label.c	72
MB_LoadExtract.c	74
MB_Lookup.c	76
MB_Mask.c	77
MB_Mul.c	78
MB_ngb.h	79
MB_Or.c	80
MB_Pixel.c	81
MB_Range.c	82
MB_Sub.c	83
MB_Sup.c	84
MB_SupNb32.c	85
MB_SupNb8.c	86
MB_SupNb.c	87
MB_Thresh.c	88
MB_Volume.c	89
MB_Watershed.c	90
MB_Window.c	92

MB_Xor.c**94**

Data Structure Documentation

3 MB_Image Struct Reference

```
#include <mambaCommon.h>
```

Data Fields

- unsigned int **BITPPIX**
- unsigned int **NUMCOL**
- **PLINE * PLINES**
- **PIX8 * PIXARRAY**

3.1 Detailed Description

Images structure with the depth (**BITPPIX**); the pixels array (**PIXARRAY**) and entry point array to each line of the image (**PLINES**)

3.2 Field Documentation

3.2.1 unsigned int **MB_Image::BITPPIX**

The depth of the image

3.2.2 unsigned int **MB_Image::NUMCOL**

number of colors used

3.2.3 **PIX8*** **MB_Image::PIXARRAY**

pixel array

3.2.4 **PLINE*** **MB_Image::PLINES**

accessors to pixel lines

The documentation for this struct was generated from the following file:

- **mambaCommon.h**

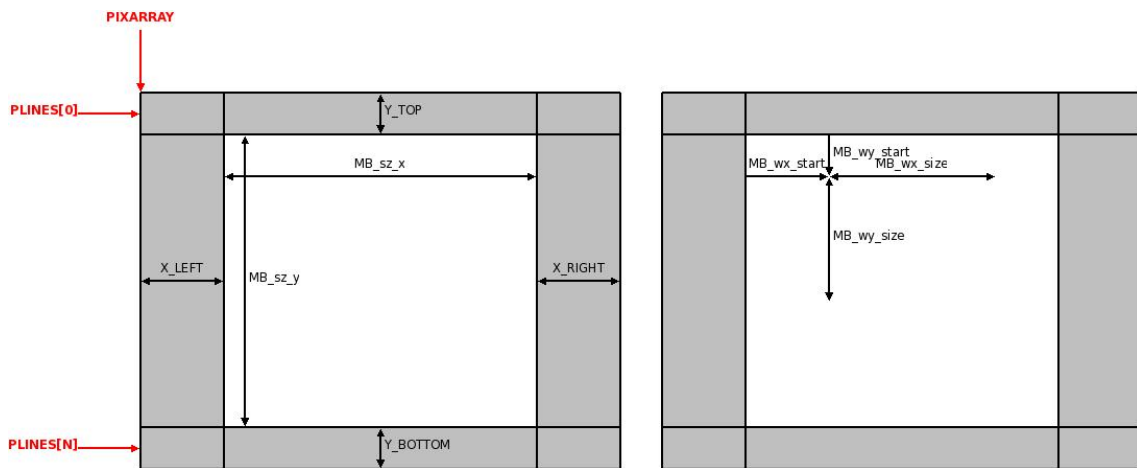


Figure 1: Image structure and variables

4 MB_ListControl Struct Reference

```
#include <mambaApi_loc.h>
```

Data Fields

- int **firstx**
- int **firsty**
- int **lastx**
- int **lasty**

4.1 Detailed Description

List control structure that give you the index of the first and last elements of list.

4.2 Field Documentation

4.2.1 int MB_ListControl::firstx

first token of the list (x)

4.2.2 int MB_ListControl::firsty

first token of the list (y)

4.2.3 int MB_ListControl::lastx

last token of the list (x)

4.2.4 int MB_ListControl::lasty

last token of the list (y)

The documentation for this struct was generated from the following file:

- **mambaApi_loc.h**

5 MB_ Token Struct Reference

```
#include <mambaApi_loc.h>
```

Data Fields

- **PIX32 tag**
- **int nextx**
- **int nexty**

5.1 Detailed Description

Token used in hierarchical list. A token gives its tag and the next (by position nextx, nexty in image) token in the list (-1 if the list ends).

5.2 Field Documentation

5.2.1 int MB_ Token::nextx

next token (x)

5.2.2 int MB_ Token::nexty

next token (y)

5.2.3 PIX32 MB_ Token::tag

The tag associated to the pixel

The documentation for this struct was generated from the following file:

- **mambaApi_loc.h**

6 mambaApi.h File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdint.h>
#include <malloc.h>
#include "mambaCommon.h"
#include "MB_error.h"
#include "MB_ngb.h"
```

Defines

- #define MB_LINE_COUNT(im) ((MB_wx_size*im → BITPPIX)/CHARBIT)
- #define MB_LINE_OFFSET(im) (MB_X_LEFT(im)+(MB_wx_start*im → BITPPIX)/CHARBIT)

Functions

- MB_errcode MB_Create (MB_Image *image, int depth)
- MB_errcode MB_Destroy (MB_Image *image)
- MB_errcode MB_SetSizeAndWindow (int *w, int *h)
- MB_errcode MB_SetWindow (int x1, int y1, int x2, int y2)
- MB_errcode MB_ResetWindow (void)
- MB_errcode MB_Load (MB_Image *image, PIX8 *indata, int len)
- MB_errcode MB_Extract (MB_Image *image, PIX8 **outdata, int *len)
- MB_errcode MB_Convert (MB_Image *src, MB_Image *dest)
- MB_errcode MB_Copy (MB_Image *src, MB_Image *dest)
- MB_errcode MB_PutPixel (MB_Image *srcdest, int pixVal, unsigned int x, unsigned int y)
- MB_errcode MB_GetPixel (MB_Image *src, int *pixVal, unsigned int x, unsigned int y)
- MB_errcode MB_And (MB_Image *src1, MB_Image *src2, MB_Image *dest)
- MB_errcode MB_Or (MB_Image *src1, MB_Image *src2, MB_Image *dest)
- MB_errcode MB_Xor (MB_Image *src1, MB_Image *src2, MB_Image *dest)
- MB_errcode MB_Inv (MB_Image *src, MB_Image *dest)
- MB_errcode MB_Inf (MB_Image *src1, MB_Image *src2, MB_Image *dest)
- MB_errcode MB_InfNbb (MB_Image *src, MB_Image *srcdest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)
- MB_errcode MB_InfNb8 (MB_Image *src, MB_Image *srcdest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)
- MB_errcode MB_InfNb32 (MB_Image *src, MB_Image *srcdest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)
- MB_errcode MB_Sup (MB_Image *src1, MB_Image *src2, MB_Image *dest)
- MB_errcode MB_SupNbb (MB_Image *src, MB_Image *srcdest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)
- MB_errcode MB_SupNb8 (MB_Image *src, MB_Image *srcdest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)
- MB_errcode MB_SupNb32 (MB_Image *src, MB_Image *srcdest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)
- MB_errcode MB_Add (MB_Image *src1, MB_Image *src2, MB_Image *dest)
- MB_errcode MB_Sub (MB_Image *src1, MB_Image *src2, MB_Image *dest)
- MB_errcode MB_Mul (MB_Image *src1, MB_Image *src2, MB_Image *dest)
- MB_errcode MB_Shiftb (MB_Image *src, MB_Image *dest, unsigned int dirnum, unsigned int count, int long_filler_pix, enum MB_grid_t grid)
- MB_errcode MB_Shift32 (MB_Image *src, MB_Image *dest, unsigned int dirnum, unsigned int count, int long_filler_pix, enum MB_grid_t grid)

- **MB_errcode MB_Diff** (MB_Image *src1, MB_Image *src2, MB_Image *dest)
- **MB_errcode MB_ConAdd** (MB_Image *src, int value, MB_Image *dest)
- **MB_errcode MB_ConSub** (MB_Image *src, int value, MB_Image *dest)
- **MB_errcode MB_ConMul** (MB_Image *src, int value, MB_Image *dest)
- **MB_errcode MB_ConDiv** (MB_Image *src, int value, MB_Image *dest)
- **MB_errcode MB_ConSet** (MB_Image *dest, int value)
- **MB_errcode MB_Volume** (MB_Image *src, Uint32 *pVolume)
- **MB_errcode MB_Check** (MB_Image *src, int *isEmpty)
- **MB_errcode MB_Lookup** (MB_Image *src, MB_Image *dest, Uint32 *ptab)
- **MB_errcode MB_Histo** (MB_Image *src, Uint32 *phisto)
- **MB_errcode MB_Compare** (MB_Image *src, MB_Image *cmp, MB_Image *dest, int *px, int *py)
- **MB_errcode MB_Thresh** (MB_Image *src, MB_Image *dest, int low, int high)
- **MB_errcode MB_BldNbb** (MB_Image *mask, MB_Image *srcdest, unsigned int dirnum, Uint32 *pVolume, enum MB_grid_t grid, enum MB_bordermode_t border)
- **MB_errcode MB_BldNb8** (MB_Image *mask, MB_Image *srcdest, unsigned int dirnum, Uint32 *pVolume, enum MB_grid_t grid, enum MB_bordermode_t border)
- **MB_errcode MB_DualBldNbb** (MB_Image *mask, MB_Image *srcdest, unsigned int dirnum, Uint32 *pVolume, enum MB_grid_t grid, enum MB_bordermode_t border)
- **MB_errcode MB_DualBldNb8** (MB_Image *mask, MB_Image *srcdest, unsigned int dirnum, Uint32 *pVolume, enum MB_grid_t grid, enum MB_bordermode_t border)
- **MB_errcode MB_DiffNbb** (MB_Image *src, MB_Image *srcdest, unsigned int dirnum, enum MB_grid_t grid, enum MB_bordermode_t border)
- **MB_errcode MB_DiffNb8** (MB_Image *src, MB_Image *srcdest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)
- **MB_errcode MB_Mask** (MB_Image *src, MB_Image *dest, int maskf, int maskt)
- **MB_errcode MB_Range** (MB_Image *src, int *min, int *max)
- **MB_errcode MB_depthRange** (MB_Image *src, int *min, int *max)
- **MB_errcode MB_CopyBitPlane** (MB_Image *src, MB_Image *dest, unsigned int plane)
- **MB_errcode MB_CopyBytePlane** (MB_Image *src, MB_Image *dest, unsigned int plane)
- **MB_errcode MB_BinHitOrMiss** (MB_Image *src, MB_Image *dest, unsigned int es0, unsigned int es1, enum MB_grid_t grid, enum MB_bordermode_t border)
- **MB_errcode MB_Labelb** (MB_Image *src, MB_Image *dest, unsigned int lblow, unsigned int lbhigh, int *pNbobj, enum MB_grid_t grid)
- **MB_errcode MB_Abs32** (MB_Image *src, MB_Image *dest)
- **MB_errcode MB_Distanceb** (MB_Image *src, MB_Image *dest, enum MB_grid_t grid, enum MB_bordermode_t border)
- **MB_errcode MB_Watershed** (MB_Image *src, MB_Image *marker, unsigned int max_level, enum MB_grid_t grid)
- **MB_errcode MB_Extrema** (MB_Image *src, MB_Image *dest, enum MB_grid_t grid, enum MB_bordermode_t border)
- **MB_errcode MB_Frame** (MB_Image *src, int thresval, int *ulx, int *uly, int *brx, int *bry)

Variables

- **Uint32 MB_w**
- **Uint32 MB_h**
- **Uint32 MB_sz_x**
- **Uint32 MB_sz_y**
- **Sint32 MB_wx_start**
- **Sint32 MB_wy_start**
- **Uint32 MB_wx_size**
- **Uint32 MB_wy_size**
- **Uint32 MB_refcounter**

6.1 Detailed Description

Date:

11-4-2007

This file contains the various definitions, global variables macro, struct and functions created for the library.

6.2 Define Documentation

6.2.1 `#define MB_LINE_COUNT(im) ((MB_wx_size*im → BITPPIX)/CHARBIT)`

Returns the size in bytes of an image line

6.2.2 `#define MB_LINE_OFFSET(im) (MB_X_LEFT(im)+(MB_wx_start*im → BITPPIX)/CHARBIT)`

Returns the size in bytes of global variable `MB_wx_start` plus standard offset `X_LEFT`

6.3 Function Documentation

6.3.1 `MB_errcode MB_Abs32 (MB_Image * src, MB_Image * dest)`

Computes the absolute value image out of a 32 bits image.

Parameters:

src image source (in signed 32 bits format)

dest destination image (in signed 32 bits format)

Returns:

An error code (`NO_ERR` if successful)

6.3.2 `MB_errcode MB_Add (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Adds the pixels of two images and put the result in the third image. Depending on the format of the target image, the result may be saturated or not. You can perform the following additions : 1-bit + 1-bit = 1-bit (call the `MB_Or` function) 1-bit + 8-bit = 8-bit (saturated) 1-bit + 8-bit = 32-bit 1-bit + 32-bit = 32-bit 8-bit + 8-bit = 8-bit (saturated) 8-bit + 8-bit = 32-bit 8-bit + 32-bit = 32-bit 32-bit + 32-bit = 32-bit

Parameters:

src1 image 1

src2 image 2

dest image resulting of the addition of image 1 and 2.

Returns:

An error code (`NO_ERR` if successful)

6.3.3 `MB_errcode MB_And (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Performs a logic AND between the pixels of two images.

Parameters:

src1 image 1

src2 image 2

dest destination image

Returns:

An error code (`NO_ERR` if successful)

6.3.4 MB_errcode MB_BinHitOrMiss (MB_Image * *src*, MB_Image * *dest*, unsigned int *es0*, unsigned int *es1*, enum MB_grid_t *grid*, enum MB_bordermode_t *border*)

Performs a binary Hit or miss operation on image *imIn* using the structuring element *es0* and *es1*. Structuring elements are integers value coding which direction must be taken into account. *es0* indicating which neighbor of the current pixel will be checked for 0 value. *es1* those which will be evaluated for 1 value.

For example, in hexagonal grid, it means that if you want to look for a pattern where the neighbors in direction 6 and 1 are true while the current pixel is false as long as neighbors 2 and 5, you will encode this in the elements *es0* and *es1* like this : 1 1 0 0 0 X X $es0 = 1+4+32$ $es1 = 64+2$

Parameters:

src output image

dest input image

es0 structuring element for 0 value.

es1 structuring element for 1 value.

grid grid configuration

border border configuration

Returns:

An error code (NO_ERR if successful)

6.3.5 MB_errcode MB_BldNb8 (MB_Image * *mask*, MB_Image * *srcdest*, unsigned int *dirnum*, Uint32 * *pVolume*, enum MB_grid_t *grid*, enum MB_bordermode_t *border*)

(re)Builds an image according to a direction and a mask image. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

mask the mask image

srcdest the rebuild image

dirnum the direction number

pVolume the computed volume of the output image

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.6 MB_errcode MB_BldNbb (MB_Image * *mask*, MB_Image * *srcdest*, unsigned int *dirnum*, Uint32 * *pVolume*, enum MB_grid_t *grid*, enum MB_bordermode_t *border*)

(re)Builds an image according to a direction and a mask image. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

mask the mask image

srcdest the rebuild image

dirnum the direction number

pVolume the computed volume of the output image

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.7 MB_errcode MB_Check (MB_Image * *src*, int * *isEmpty*)

Verify that the image is not empty (all pixels to 0).

Parameters:

src the source image

isEmpty an integer which is set to 1 if empty or 0 if not

Returns:

An error code (NO_ERR if successful)

6.3.8 MB_errcode MB_Compare (MB_Image * *src*, MB_Image * *cmp*, MB_Image * *dest*, int * *px*, int * *py*)

Performs a comparison between a source image and a given base image.

Parameters:

src the source image

cmp the image to which the source image is compared

dest destination image

px position in x of the first different pixel between the two images (-1 if images are similar)

py position in y of the first different pixel between the two images (-1 if images are similar)

Returns:

An error code (NO_ERR if successful)

6.3.9 MB_errcode MB_ConAdd (MB_Image * *src*, int *value*, MB_Image * *dest*)

Adds a constant value to the pixels of an image.

Parameters:

src the source image

value the constant value to add to the pixels

dest the image resulting of the addition of image 1 and value.

Returns:

An error code (NO_ERR if successful)

6.3.10 MB_errcode MB_ConDiv (MB_Image * *src*, int *value*, MB_Image * *dest*)

Divides a constant value to the pixels of an image.

Parameters:

src the source image

value the constant value to add to the pixels

dest the image resulting of the addition of image 1 and value.

Returns:

An error code (NO_ERR if successful)

6.3.11 MB_errcode MB_ConMul (MB_Image * *src*, int *value*, MB_Image * *dest*)

Multiplies a constant value to the pixels of an image.

Parameters:

src the source image

value the constant value to add to the pixels

dest the image resulting of the addition of image 1 and value.

Returns:

An error code (NO_ERR if successful)

6.3.12 MB_errcode MB_ConSet (MB_Image * *dest*, int *value*)

Fill an image with a specific value

Parameters:

dest the image
value the value to fill the image

Returns:

An error code (NO_ERR if successful)

6.3.13 MB_errcode MB_ConSub (MB_Image * *src*, int *value*, MB_Image * *dest*)

Subtracts a constant value to the pixels of an image.

Parameters:

src the source image
value the constant value to add to the pixels
dest the image resulting of the subtraction of image 1 and value.

Returns:

An error code (NO_ERR if successful)

6.3.14 MB_errcode MB_Convert (MB_Image * *src*, MB_Image * *dest*)

Convert an image of a given depth into another depth

Parameters:

src source image
dest destination image

Returns:

An error code (NO_ERR if successful)

6.3.15 MB_errcode MB_Copy (MB_Image * *src*, MB_Image * *dest*)

Copies an image data content into another image

Parameters:

src the source image
dest the destination image

Returns:

An error code (NO_ERR if successful)

6.3.16 MB_errcode MB_CopyBitPlane (MB_Image * *src*, MB_Image * *dest*, unsigned int *plane*)

Inserts or extract the bit plane in/out an image *src* into *dest*.

Parameters:

src source image
dest destination image
plane the plane number

Returns:

An error code (NO_ERR if successful)

6.3.17 `MB_errcode MB_CopyBytePlane (MB_Image * src, MB_Image * dest, unsigned int plane)`

Inserts or extract the byte plane in/out an image *src* into/from *dest*.

Parameters:

src source image
dest destination image
plane the plane number

Returns:

An error code (NO_ERR if successful)

6.3.18 `MB_errcode MB_Create (MB_Image * image, int depth)`

Creates an image (memory allocation) with the correct depth given as argument

Parameters:

image the created image
depth the depth of the created image

Returns:

An error code (NO_ERR if successful)

6.3.19 `MB_errcode MB_depthRange (MB_Image * src, int * min, int * max)`

gives the minimum and maximum value possible of the image pixels given the image depth

Parameters:

src source image
min the minimum possible value of the pixels
max the maximum possible value of the pixels

Returns:

An error code (NO_ERR if successful)

6.3.20 `MB_errcode MB_Destroy (MB_Image * image)`

Destroy an image (memory freeing)

Parameters:

image the image to be destroy

Returns:

An error code (NO_ERR if successful)

6.3.21 `MB_errcode MB_Diff (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Computes the diff between two image. The result image pixel value is the pixel value of image 1 if this value was greater than value of pixel 2 otherwise the minimum possible (e.g. 0 or -32765) value is set for the pixel

Parameters:

src1 source image 1
src2 source image 2
dest destination image

Returns:

An error code (NO_ERR if successful)

6.3.22 `MB_errcode MB_DiffNb8 (MB_Image * src, MB_Image * srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)`

Looks for the minimum in a grey scale image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

count if src and srctest are the same, the operation is repeated count times

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.23 `MB_errcode MB_DiffNbb (MB_Image * src, MB_Image * srctest, unsigned int dirnum, enum MB_grid_t grid, enum MB_bordermode_t border)`

Performs a set difference in a binary image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.24 `MB_errcode MB_Distanceb (MB_Image * src, MB_Image * dest, enum MB_grid_t grid, enum MB_bordermode_t border)`

Computes for each pixel the distance to the border of the set in which the pixel is found

The algorithm works with two passes, first from the top left to the bottom right and then back.

Parameters:

src the binary source image

dest the 32-bit image in which the distance for each pixel is stored

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.25 `MB_errcode MB_DualBldNb8 (MB_Image * mask, MB_Image * srcdest, unsigned int dirnum, Uint32 * pVolume, enum MB_grid_t grid, enum MB_bordermode_t border)`

(re)Builds (dual operation) an image according to a direction and a mask image. The direction depends on the grid used (see `MB_ngh.h` for definitions of directions).

Parameters:

mask the mask image
srcdest the rebuild image
dirnum the direction number
pVolume the computed volume of the output image
grid the grid used (either square or hexagonal)
border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (`NO_ERR` if successful)

6.3.26 `MB_errcode MB_DualBldNb6 (MB_Image * mask, MB_Image * srcdest, unsigned int dirnum, Uint32 * pVolume, enum MB_grid_t grid, enum MB_bordermode_t border)`

(re)Builds (dual operation) an image according to a direction and a mask image. The direction depends on the grid used (see `MB_ngh.h` for definitions of directions).

Parameters:

mask the mask image
srcdest the rebuild image
dirnum the direction number
pVolume the computed volume of the output image
grid the grid used (either square or hexagonal)
border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (`NO_ERR` if successful)

6.3.27 `MB_errcode MB_Extract (MB_Image * image, PIX8 ** outdata, int * len)`

Reads an image data content and put it in an array

Parameters:

image the image to read
outdata pointer to the array created (malloc) and filled with the pixel data of the image
len the length in bytes of data extracted

Returns:

An error code (`NO_ERR` if successful)

6.3.28 `MB_errcode MB_Extrema (MB_Image * src, MB_Image * dest, enum MB_grid_t grid, enum MB_bordermode_t border)`

Computes the local extrema out of a greyscale image. The result is put into a 32-bit image. The function returns the local maxima or the minima coded into the 32-bit image.

The extrema are coded as follow into the 32 bit values. | 0 | 1 | 2 | 3 | |——|——|——|——| | MIN | MAX | isMin | isMax | They can be accessed using the function `MB_CopyBytePlane`

Parameters:

src the grey-scale source image
dest the 32-bit image produced
grid the grid used (either square or hexagonal)
border the kind of border to use

Returns:

An error code (NO_ERR if successful)

6.3.29 MB_errcode MB_Frame (MB_Image * src, int thresval, int * ulx, int * uly, int * brx, int * bry)

Returns the smallest frame that contains all the pixels of image that are greater or equal to the given threshold value. using the four last pointer to describe it

Parameters:

src source image
thresval the threshold value used to compute the frame
ulx the x-coordinate of the upper left corner of the frame
uly the y-coordinate of the upper left corner of the frame
brx the x-coordinate of the bottom right corner of the frame
bry the y-coordinate of the bottom right corner of the frame

Returns:

An error code (NO_ERR if successful)

6.3.30 MB_errcode MB_GetPixel (MB_Image * src, int * pixVal, unsigned int x, unsigned int y)

Gets the pixel value inside the image at the given position

Parameters:

src the image
pixVal the returned pixel value
x position in x of the pixel targeted
y position in y of the pixel targeted

Returns:

An error code (NO_ERR if successful)

6.3.31 MB_errcode MB_Histo (MB_Image * src, Uint32 * phisto)

Computes the histogram of an image. The histogram is an array

Parameters:

src source image
phisto pointer to the histogram array

Returns:

An error code (NO_ERR if successful)

6.3.32 `MB_errcode MB_Inf (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Determines the inferior value between the pixels of two images. The result is put in the corresponding pixel position in the destination image.

Parameters:

src1 image 1
src2 image 2
dest destination image

Returns:

An error code (NO_ERR if successful)

6.3.33 `MB_errcode MB_InfNb32 (MB_Image * src, MB_Image * srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)`

Looks for the minimum in a 32-bit image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image
srctest destination image
dirnum the direction index
count if src and srctest are the same, the operation is repeated count times
grid the grid used (either square or hexagonal)
border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.34 `MB_errcode MB_InfNb8 (MB_Image * src, MB_Image * srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)`

Looks for the minimum in a grey scale image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image
srctest destination image
dirnum the direction index
count if src and srctest are the same, the operation is repeated count times
grid the grid used (either square or hexagonal)
border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.35 `MB_errcode MB_InfNbb (MB_Image * src, MB_Image * srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)`

Looks for the minimum in a binary image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image
srctest destination image
dirnum the direction index
count if src and srctest are the same, the operation is repeated count times
grid the grid used (either square or hexagonal)
border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.36 `MB_errcode MB_Inv (MB_Image * src, MB_Image * dest)`

Inverts the pixels value (negation) of the source image.

Parameters:

src source image
dest destination image

Returns:

An error code (NO_ERR if successful)

6.3.37 `MB_errcode MB_Labelb (MB_Image * src, MB_Image * dest, unsigned int lblow, unsigned int lbhigh, int * pNobj, enum MB_grid_t grid)`

Labelling the object found in src image.

Parameters:

src the binary source image where the object must be labelled
dest the 32-bit image where object are labelled
lblow the lowest value allowed for label on the low byte (must be inferior to lbhigh)
lbhigh the first high value NOT allowed for label on the low byte (maximum allowed is 256)
pNobj the number of object founds
grid the grid used (either square or hexagonal)

Returns:

An error code (NO_ERR if successful)

6.3.38 `MB_errcode MB_Load (MB_Image * image, PIX8 * indata, int len)`

Loads an image data with data given in argument

Parameters:

image the image to fill
indata the data to fill the image with (complete pixels values)
len the length of data given

Returns:

An error code (NO_ERR if successful)

6.3.39 MB_errcode MB_Lookup (MB_Image * *src*, MB_Image * *dest*, Uint32 * *ptab*)

Apply the function in the lookup table to the pixels of source image. A pixel value is changed to a new value accordingly with its current value

Parameters:

src source image
dest destination image
ptab the lookup table pointer

Returns:

An error code (NO_ERR if successful)

6.3.40 MB_errcode MB_Mask (MB_Image * *src*, MB_Image * *dest*, int *maskf*, int *maskt*)

Convert a binary image in a grey scale image (8-bit) or in a 32-bit image using value *maskf* to replace 0 and *maskt* to replace 1.

Parameters:

src binary source image
dest destination image
maskf for 0 (false) pixel value
maskt for 1 (true) pixel value

Returns:

An error code (NO_ERR if successful)

6.3.41 MB_errcode MB_Mul (MB_Image * *src1*, MB_Image * *src2*, MB_Image * *dest*)

Multiplies the pixels of two images and put the result in the third image. Depending on the format of the target image, the result may be saturated or not. You can perform the following additions : 1-bit * 1-bit = 1-bit (call the MB_And function) 1-bit * 8-bit = 8-bit 1-bit * 8-bit = 32-bit 1-bit * 32-bit = 32-bit 8-bit * 8-bit = 8-bit (saturated) 8-bit * 8-bit = 32-bit 8-bit * 32-bit = 32-bit 32-bit * 32-bit = 32-bit

Parameters:

src1 image 1
src2 image 2
dest image resulting of the addition of image 1 and 2.

Returns:

An error code (NO_ERR if successful)

6.3.42 MB_errcode MB_Or (MB_Image * *src1*, MB_Image * *src2*, MB_Image * *dest*)

Applies a OR on the pixels of two images. All the images must have the same depth for correct work.

Parameters:

src1 image 1
src2 image 2
dest destination image

Returns:

An error code (NO_ERR if successful)

6.3.43 MB_errcode MB_PutPixel (MB_Image * *srcdest*, int *pixVal*, unsigned int *x*, unsigned int *y*)

Puts the pixel value inside the image at the given position

Parameters:

srcdest the image
pixVal the pixel value
x position in x of the pixel targeted
y position in y of the pixel targeted

Returns:

An error code (NO_ERR if successful)

6.3.44 MB_errcode MB_Range (MB_Image * *src*, int * *min*, int * *max*)

gives the minimum and maximum value of the image pixels i.e its range.

Parameters:

src source image
min the minimum value of the pixels
max the maximum value of the pixels

Returns:

An error code (NO_ERR if successful)

6.3.45 MB_errcode MB_ResetWindow (void)

Reset the window to the maximum size (the full image).

Returns:

An error code (NO_ERR if successful)

6.3.46 MB_errcode MB_SetSizeAndWindow (int * *w*, int * *h*)

Set the image sizes correctly and puts the window size to the maximum (all the image is processed).

Parameters:

w the width
h the height

Returns:

An error code (NO_ERR if successful)

6.3.47 MB_errcode MB_SetWindow (int *x1*, int *y1*, int *x2*, int *y2*)

Set the window size so that it includes the position given in argument. The window size must also respect certain rules to be valid.

Parameters:

x1 the x position of the upper left pixel (INSIDE the window)
y1 the y position of the upper left pixel (INSIDE the window)
x2 the x position of the lower right pixel (INSIDE the window)
y2 the y position of the lower right pixel (INSIDE the window)

Returns:

An error code (NO_ERR if successful)

6.3.48 `MB_errcode MB_Shift32 (MB_Image * src, MB_Image * dest, unsigned int dirnum, unsigned int count, int long_filler_pix, enum MB_grid_t grid)`

Shifts the content of a 32-bit image in a given direction with a given amplitude. The direction depends on the grid used (see `MB_ngh.h` for definitions of directions).

Parameters:

src source image
dest destination image
dirnum the direction index
count the amplitude of the shift
long_filler_pix the value used to fill the created space
grid the grid used (either square or hexagonal)

Returns:

An error code (`NO_ERR` if successful)

6.3.49 `MB_errcode MB_Shiftb (MB_Image * src, MB_Image * dest, unsigned int dirnum, unsigned int count, int long_filler_pix, enum MB_grid_t grid)`

Shift the source image in the given direction. The direction depends on the grid used (see `MB_ngh.h` for definitions of directions).

Parameters:

src source image
dest destination image
dirnum the direction (from 0 to 8)
count the amplitude of the shift (in pixels)
long_filler_pix the value used to fill the created space
grid the grid used (either square or hexagonal)

Returns:

An error code (`NO_ERR` if successful)

6.3.50 `MB_errcode MB_Sub (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Subtracts the values of pixels of the second image to the values of the pixel in the first image

Parameters:

src1 image 1
src2 image 2
dest image resulting of the subtraction

Returns:

An error code (`NO_ERR` if successful)

6.3.51 `MB_errcode MB_Sup (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Determines the superior value between the pixels of two images. The result is put in the corresponding pixel position in the destination image.

Parameters:

src1 image 1
src2 image 2
dest destination image

Returns:

An error code (`NO_ERR` if successful)

6.3.52 `MB_errcode MB_SupNb32 (MB_Image * src, MB_Image * srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)`

Looks for the maximum in a 32-bit image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image
srctest destination image
dirnum the direction index
count if src and srctest are the same, the operation is repeated count times
grid the grid used (either square or hexagonal)
border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.53 `MB_errcode MB_SupNb8 (MB_Image * src, MB_Image * srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)`

Looks for the maximum in a grey scale image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image
srctest destination image
dirnum the direction index
count if src and srctest are the same, the operation is repeated count times
grid the grid used (either square or hexagonal)
border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.54 `MB_errcode MB_SupNbb (MB_Image * src, MB_Image * srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)`

Looks for the maximum in a binary image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image
srctest destination image
dirnum the direction index
count if src and srctest are the same, the operation is repeated count times
grid the grid used (either square or hexagonal)
border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

6.3.55 MB_errcode MB_Thresh (MB_Image * *src*, MB_Image * *dest*, int *low*, int *high*)

Fills a binary image according to the following rules : if pixel value lower than low or higher than high the binary pixel is set to 0 in others case the pixel is set to 1.

Parameters:

src source image
dest destination image
low low value for threshold
high high value for treshold

Returns:

An error code (NO_ERR if successful)

6.3.56 MB_errcode MB_Volume (MB_Image * *src*, Uint32 * *pVolume*)

Computes the volume of an image. The volume is the sum of the pixel values (i.e. integration of the image)

Parameters:

src source image
pVolume pointer to the volume variable

Returns:

An error code (NO_ERR if successful)

6.3.57 MB_errcode MB_Watershed (MB_Image * *src*, MB_Image * *marker*, unsigned int *max_level*, enum MB_grid_t *grid*)

Perform a watershed segmentation of the image using the marker image as a starting point for the flooding. The result is put into a the 32 bit image that start with the marker.

The segmentation is coded as follow into the 32 bit values. | 0 | 1 | 2 | 3 | |———|———|———|———| | Segment label | isLine | They can be accessed using the function MB_CopyBytePlane. isLine is a value indicating if the pixel belong to the watershed (255 if this is the case, undefined otherwise).

Parameters:

src the greyscale image to segment
marker the marker image in which the result of segmentation will be put
max_level the maximum level reach by the water.
grid the grid used (either square or hexagonal)

Returns:

An error code (NO_ERR if successful)

6.3.58 MB_errcode MB_Xor (MB_Image * *src1*, MB_Image * *src2*, MB_Image * *dest*)

Applies a XOR on the pixels of two images. All the images must have the same depth for correct work.

Parameters:

src1 image 1
src2 image 2
dest destination image

Returns:

An error code (NO_ERR if successful)

6.4 Variable Documentation

6.4.1 **Uint32** MB_h

original Height of the images computed

6.4.2 **Uint32** MB_refcounter

image counter

6.4.3 **Uint32** MB_sz_x

Width of the images computed

6.4.4 **Uint32** MB_sz_y

Height of the images computed

6.4.5 **Uint32** MB_w

original Width of the images computed

6.4.6 **Uint32** MB_wx_size

Width of the window computation

6.4.7 **Sint32** MB_wx_start

offset to the first pixel position in x of the window computation

6.4.8 **Uint32** MB_wy_size

Height of the window computation

6.4.9 **Sint32** MB_wy_start

offset to the first pixel position in y of the window computation

7 mambaApi_loc.h File Reference

```
#include "mambaApi.h"
```

Data Structures

- struct MB_Token
- struct MB_ListControl

Defines

- #define MB_PROBE_PAIR(im_in, im_out) (((im_in → BITPPIX) << 7) + (im_out) → BITPPIX)

Variables

- MB_ListControl MB_HierarchicalList [256]
- MB_Token * MB_TokensArray
- PIX32 * MB_Label_EQ
- PIX32 * MB_Label_CEQ
- Uint32 MB_Label_maxEQ
- Uint32 MB_VolumePerByte [256]

7.1 Detailed Description

Date:

11-4-2007

This file contains the various definitions, global variables macro, struct and functions that are shared between components of the library but are not meant to be exported to the outside world.

7.2 Define Documentation

7.2.1 #define MB_PROBE_PAIR(im_in, im_out) (((im_in → BITPPIX) << 7) + (im_out) → BITPPIX)

Returns the value of the images combination MB_PAIR_x_x

7.3 Variable Documentation

7.3.1 MB_ListControl MB_HierarchicalList[256]

The hierarchical list entries for watershed segmentation

7.3.2 PIX32* MB_Label_CEQ

Labelling arrays and variables

7.3.3 PIX32* MB_Label_EQ

Labelling arrays and variables

Equivalence between label in an image

7.3.4 Uint32 MB_Label_maxEQ

The greatest number of label possible according to current image size

7.3.5 MB_Token* MB_TokensArray

The memory used to hold the elements of the hierarchical list

7.3.6 Uint32 MB_VolumePerByte[256]

Volume arrays

8 mambaCommon.h File Reference

```
#include <stdint.h>
```

Data Structures

- struct `MB_Image`

Defines

- `#define MB_X_LEFT(im) X_LEFT`
- `#define MB_X_RIGHT(im) X_RIGHT`
- `#define MB_Y_TOP(im) Y_TOP`
- `#define MB_Y_BOTTOM(im) Y_BOTTOM`

Typedefs

- `typedef uint8_t Uint8`
- `typedef uint16_t Uint16`
- `typedef uint32_t Uint32`
- `typedef int8_t Sint8`
- `typedef int16_t Sint16`
- `typedef int32_t Sint32`
- `typedef uint8_t PIX8`
- `typedef PIX8 * PLINE`
- `typedef int32_t PIX32`
- `typedef PIX32 * PLINE32`

8.1 Detailed Description

Date:

31-03-2009

This file contains the various definitions, macro, struct that are commons between the various modules of the library

The copyright license of Mamba is reminded here :

Copyright (c) <2009>, <Nicolas BEUCHER and ARMINES for the Centre de Morphologie Mathématique(CMM), common research center to ARMINES and MINES Paristech>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

Except as contained in this notice, the names of the above copyright holders shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without their prior written authorization.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

8.2 Define Documentation

8.2.1 `#define MB_X_LEFT(im) X_LEFT`

Getting image frame offset from left

8.2.2 `#define MB_X_RIGHT(im) X_RIGHT`

Getting image frame offset from right

8.2.3 `#define MB_Y_BOTTOM(im) Y_BOTTOM`

Getting image frame offset from bottom

8.2.4 `#define MB_Y_TOP(im) Y_TOP`

Getting image frame offset from top

8.3 Typedef Documentation

8.3.1 `typedef int32_t PIX32`

Signed 32-bit pixels value type

8.3.2 `typedef uint8_t PIX8`

grey-scale pixels value type

8.3.3 `typedef PIX8* PLINE`

Pixels line pointers type

8.3.4 `typedef PIX32* PLINE32`

32-bit pixels line pointers type

8.3.5 `typedef int16_t Sint16`

Signed 16 bit value type

8.3.6 `typedef int32_t Sint32`

Signed 32 bit value type

8.3.7 `typedef int8_t Sint8`

Signed 8 bit value type

8.3.8 `typedef uint16_t Uint16`

Unsigned 16 bit value type

8.3.9 `typedef uint32_t Uint32`

Unsigned 32 bit value type

8.3.10 `typedef uint8_t Uint8`

Unsigned 8 bit value type

9 MB_Abs32.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Abs32 (MB_Image *src, MB_Image *dest)`

9.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

9.2 Function Documentation

9.2.1 `MB_errcode MB_Abs32 (MB_Image * src, MB_Image * dest)`

Computes the absolute value image out of a 32 bits image.

Parameters:

src image source (in signed 32 bits format)

dest destination image (in signed 32 bits format)

Returns:

An error code (NO_ERR if successful)

10 MB_Add.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Add (MB_Image *src1, MB_Image *src2, MB_Image *dest)`

10.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-2-2007

10.2 Function Documentation

10.2.1 `MB_errcode MB_Add (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Adds the pixels of two images and put the result in the third image. Depending on the format of the target image, the result may be saturated or not. You can perform the following additions : 1-bit + 1-bit = 1-bit (call the `MB_Or` function) 1-bit + 8-bit = 8-bit (saturated) 1-bit + 8-bit = 32-bit 1-bit + 32-bit = 32-bit 8-bit + 8-bit = 8-bit (saturated) 8-bit + 8-bit = 32-bit 8-bit + 32-bit = 32-bit 32-bit + 32-bit = 32-bit

Parameters:

src1 image 1

src2 image 2

dest image resulting of the addition of image 1 and 2.

Returns:

An error code (`NO_ERR` if successful)

11 MB_And.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_And (MB_Image *src1, MB_Image *src2, MB_Image *dest)`

11.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-2-2007

11.2 Function Documentation

11.2.1 `MB_errcode MB_And (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Performs a logic AND between the pixels of two images.

Parameters:

src1 image 1

src2 image 2

dest destination image

Returns:

An error code (NO_ERR if successful)

12 MB_BinHitOrMiss.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_BinHitOrMiss (MB_Image *src, MB_Image *dest, unsigned int es0, unsigned int es1, enum MB_grid_t grid, enum MB_bordermode_t border)`

12.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-15-2007

12.2 Function Documentation

12.2.1 `MB_errcode MB_BinHitOrMiss (MB_Image * src, MB_Image * dest, unsigned int es0, unsigned int es1, enum MB_grid_t grid, enum MB_bordermode_t border)`

Performs a binary Hit or miss operation on image *imIn* using the structuring element *es0* and *es1*. Structuring elements are integers value coding which direction must be taken into account. *es0* indicating which neighbor of the current pixel will be checked for 0 value. *es1* those which will be evaluated for 1 value.

For example, in hexagonal grid, it means that if you want to look for a pattern where the neighbors in direction 6 and 1 are true while the current pixel is false as long as neighbors 2 and 5, you will encode this in the elements *es0* and *es1* like this : 1 1 0 0 0 X X *es0* = 1+4+32 *es1* = 64+2

Parameters:

src output image

dest input image

es0 structuring element for 0 value.

es1 structuring element for 1 value.

grid grid configuration

border border configuration

Returns:

An error code (NO_ERR if successful)

13 MB_BldNb8.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **Uint32** linoff_inout, **PLINE** *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** *p_volume, **PIX8** border_val)

Functions

- **MB_errcode** MB_BldNb8 (**MB_Image** *mask, **MB_Image** *srcdest, unsigned int dirnum, **Uint32** *pVolume, enum **MB_grid_t** grid, enum **MB_bordermode_t** border)

13.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-16-2008

13.2 Typedef Documentation

- 13.2.1** typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **Uint32** linoff_inout, **PLINE** *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** *p_volume, **PIX8** border_val)

typedef for the definition of function arguments

13.3 Function Documentation

- 13.3.1** **MB_errcode** MB_BldNb8 (**MB_Image** * *mask*, **MB_Image** * *srcdest*, unsigned int *dirnum*, **Uint32** * *pVolume*, enum **MB_grid_t** *grid*, enum **MB_bordermode_t** *border*)

(re)Builds an image according to a direction and a mask image. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

mask the mask image

srcdest the rebuild image

dirnum the direction number

pVolume the computed volume of the output image

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

14 MB_BldNbb.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(PLINE *plines_inout, **Uint32** linoff_inout, PLINE *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** *p_volume, binaryT border_val)

Functions

- **MB_errcode** MB_BldNbb (MB_Image *mask, MB_Image *srctest, unsigned int dirnum, **Uint32** *pVolume, enum MB_grid_t grid, enum MB_bordermode_t border)

14.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-16-2008

14.2 Typedef Documentation

- 14.2.1 typedef void(**TSWITCHEP**)(PLINE *plines_inout, **Uint32** linoff_inout, PLINE *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** *p_volume, binaryT border_val)

typedef for the definition of function arguments

14.3 Function Documentation

- 14.3.1 **MB_errcode** MB_BldNbb (MB_Image * *mask*, MB_Image * *srctest*, unsigned int *dirnum*, **Uint32** * *pVolume*, enum MB_grid_t *grid*, enum MB_bordermode_t *border*)

(re)Builds an image according to a direction and a mask image. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

mask the mask image

srctest the rebuild image

dirnum the direction number

pVolume the computed volume of the output image

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

15 MB_Check.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Check (MB_Image *src, int *isEmpty)`

15.1 Detailed Description

Author:

Nicolas Beucher

Date:

29-1-2009

15.2 Function Documentation

15.2.1 `MB_errcode MB_Check (MB_Image * src, int * isEmpty)`

Verify that the image is not empty (all pixels to 0).

Parameters:

src the source image

isEmpty an integer which is set to 1 if empty or 0 if not

Returns:

An error code (NO_ERR if successful)

16 MB_Compare.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Compare (MB_Image *src, MB_Image *cmp, MB_Image *dest, int *px, int *py)`

16.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-6-2008

16.2 Function Documentation

16.2.1 `MB_errcode MB_Compare (MB_Image * src, MB_Image * cmp, MB_Image * dest, int * px, int * py)`

Performs a comparison between a source image and a given base image.

Parameters:

src the source image

cmp the image to which the source image is compared

dest destination image

px position in x of the first different pixel between the two images (-1 if images are similar)

py position in y of the first different pixel between the two images (-1 if images are similar)

Returns:

An error code (NO_ERR if successful)

17 MB_ConAdd.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_ConAdd (MB_Image *src, int value, MB_Image *dest)`

17.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-25-2007

17.2 Function Documentation

17.2.1 `MB_errcode MB_ConAdd (MB_Image * src, int value, MB_Image * dest)`

Adds a constant value to the pixels of an image.

Parameters:

src the source image

value the constant value to add to the pixels

dest the image resulting of the addition of image 1 and value.

Returns:

An error code (NO_ERR if successful)

18 MB_ConDiv.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_ConDiv (MB_Image *src, int value, MB_Image *dest)`

18.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

18.2 Function Documentation

18.2.1 `MB_errcode MB_ConDiv (MB_Image * src, int value, MB_Image * dest)`

Divides a constant value to the pixels of an image.

Parameters:

src the source image

value the constant value to add to the pixels

dest the image resulting of the addition of image 1 and value.

Returns:

An error code (NO_ERR if successful)

19 MB_ConMul.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_ConMul (MB_Image *src, int value, MB_Image *dest)`

19.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

19.2 Function Documentation

19.2.1 `MB_errcode MB_ConMul (MB_Image * src, int value, MB_Image * dest)`

Multiplies a constant value to the pixels of an image.

Parameters:

src the source image

value the constant value to add to the pixels

dest the image resulting of the addition of image 1 and value.

Returns:

An error code (NO_ERR if successful)

20 MB_ConSet.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_ConSet (MB_Image *dest, int value)`

20.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-29-2007

20.2 Function Documentation

20.2.1 `MB_errcode MB_ConSet (MB_Image * dest, int value)`

Fill an image with a specific value

Parameters:

dest the image

value the value to fill the image

Returns:

An error code (NO_ERR if successful)

21 MB_ConSub.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_ConSub (MB_Image *src, int value, MB_Image *dest)`

21.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-13-2007

21.2 Function Documentation

21.2.1 `MB_errcode MB_ConSub (MB_Image * src, int value, MB_Image * dest)`

Subtracts a constant value to the pixels of an image.

Parameters:

src the source image

value the constant value to add to the pixels

dest the image resulting of the subtraction of image 1 and value.

Returns:

An error code (NO_ERR if successful)

22 MB_Convert.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Convert1to8 (MB_Image *src, MB_Image *dest)`
- `MB_errcode MB_Convert8to1 (MB_Image *src, MB_Image *dest)`
- `MB_errcode MB_Convert (MB_Image *src, MB_Image *dest)`

22.1 Detailed Description

Author:

Nicolas Beucher

Date:

5-29-2008

22.2 Function Documentation

22.2.1 MB_errcode MB_Convert (MB_Image * *src*, MB_Image * *dest*)

Convert an image of a given depth into another depth

Parameters:

src source image

dest destination image

Returns:

An error code (NO_ERR if successful)

22.2.2 MB_errcode MB_Convert1to8 (MB_Image * *src*, MB_Image * *dest*)

Convert a binary image to an 8-bit image

Parameters:

src source image

dest destination image

Returns:

An error code (NO_ERR if successful)

22.2.3 MB_errcode MB_Convert8to1 (MB_Image * *src*, MB_Image * *dest*)

Convert an 8-bit image to a binary image

Parameters:

src source image

dest destination image

Returns:

An error code (NO_ERR if successful)

23 MB_CopyBitPlane.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- **Uint8 SET_BIT_PLANE** (**Uint8** value, **Uint8** bitval, unsigned int pos)
- **Uint8 EXTRACT_BIT_PLANE** (**Uint8** value, unsigned int pos)
- **MB_errcode MB_CopyBitPlane** (**MB_Image** *src, **MB_Image** *dest, unsigned int plane)

23.1 Detailed Description

Author:

Nicolas Beucher

Date:

5-29-2008

23.2 Function Documentation

23.2.1 **Uint8 EXTRACT_BIT_PLANE** (**Uint8** value, unsigned int pos)

Extracts the bit at the given position.

Parameters:

value the value in which the bit will be extracted

pos the position of the bit

Returns:

the value of the bit

23.2.2 **MB_errcode MB_CopyBitPlane** (**MB_Image** * src, **MB_Image** * dest, unsigned int plane)

Inserts or extract the bit plane in/out an image src into dest.

Parameters:

src source image

dest destination image

plane the plane number

Returns:

An error code (NO_ERR if successful)

23.2.3 **Uint8 SET_BIT_PLANE** (**Uint8** value, **Uint8** bitval, unsigned int pos)

Sets the bit at the given position.

Parameters:

value the value in which the bit will be modified

bitval the bit value we want to set

pos the position of the bit

Returns:

the modified value

24 MB_CopyBytePlane.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_CopyBytePlane (MB_Image *src, MB_Image *dest, unsigned int plane)`

24.1 Detailed Description

Author:

Nicolas Beucher

Date:

5-29-2008

24.2 Function Documentation

24.2.1 `MB_errcode MB_CopyBytePlane (MB_Image * src, MB_Image * dest, unsigned int plane)`

Inserts or extract the byte plane in/out an image *src* into/from *dest*.

Parameters:

src source image

dest destination image

plane the plane number

Returns:

An error code (NO_ERR if successful)

25 MB_Create.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Create (MB_Image *image, int depth)`
- `MB_errcode MB_Destroy (MB_Image *image)`

Variables

- `Uint32 MB_refcounter`

25.1 Detailed Description

Author:

Nicolas Beucher

Date:

5-27-2008

25.2 Function Documentation

25.2.1 `MB_errcode MB_Create (MB_Image * image, int depth)`

Creates an image (memory allocation) with the correct depth given as argument

Parameters:

image the created image

depth the depth of the created image

Returns:

An error code (NO_ERR if successful)

25.2.2 `MB_errcode MB_Destroy (MB_Image * image)`

Destroy an image (memory freeing)

Parameters:

image the image to be destroy

Returns:

An error code (NO_ERR if successful)

25.3 Variable Documentation

25.3.1 `Uint32 MB_refcounter`

image counter

26 MB_Diff.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Diff (MB_Image *src1, MB_Image *src2, MB_Image *dest)`

26.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-29-2007

Author:

Nicolas Beucher

Date:

6-7-2008

26.2 Function Documentation

26.2.1 `MB_errcode MB_Diff (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Computes the diff between two image. The result image pixel value is the pixel value of image 1 if this value was greater than value of pixel 2 otherwise the minimum possible (e.g. 0 or -32765) value is set for the pixel

Parameters:

src1 source image 1

src2 source image 2

dest destination image

Returns:

An error code (NO_ERR if successful)

27 MB_DiffNb8.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **Uint32** linoff_inout, **PLINE** *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** count, **Uint32** border_val)

Functions

- **MB_errcode** **MB_DiffNb8** (**MB_Image** *src, **MB_Image** *srctest, unsigned int dirnum, unsigned int count, enum **MB_grid_t** grid, enum **MB_bordermode_t** border)

27.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

27.2 Typedef Documentation

- 27.2.1** typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **Uint32** linoff_inout, **PLINE** *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** count, **Uint32** border_val)

typedef for the definition of function arguments

27.3 Function Documentation

- 27.3.1** **MB_errcode** **MB_DiffNb8** (**MB_Image** * *src*, **MB_Image** * *srctest*, unsigned int *dirnum*, unsigned int *count*, enum **MB_grid_t** *grid*, enum **MB_bordermode_t** *border*)

Looks for the minimum in a grey scale image in a given direction. The direction depends on the grid used (see **MB_ngh.h** for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

count if src and srctest are the same, the operation is repeated count times

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (**NO_ERR** if successful)

28 MB_DiffNbb.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void (TSWITCHEP)(PLINE *plines_inout, Uint32 linoff_inout, PLINE *plines_in, Uint32 linoff_in, Uint32 bytes_in, Uint32 count, binaryT border_val)

Functions

- MB_errcode MB_DiffNbb (MB_Image *src, MB_Image *srctest, unsigned int dirnum, enum MB_grid_t grid, enum MB_bordermode_t border)

28.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

28.2 Typedef Documentation

28.2.1 typedef void (TSWITCHEP)(PLINE *plines_inout, Uint32 linoff_inout, PLINE *plines_in, Uint32 linoff_in, Uint32 bytes_in, Uint32 count, binaryT border_val)

typedef for the definition of function arguments

28.3 Function Documentation

28.3.1 MB_errcode MB_DiffNbb (MB_Image * src, MB_Image * srctest, unsigned int dirnum, enum MB_grid_t grid, enum MB_bordermode_t border)

Performs a set difference in a binary image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

29 MB_Distanceb.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(PLINE *plines_out, **UInt32** linoff_out, PLINE *plines_in, **UInt32** linoff_in, **UInt32** bytes_in, **UInt32** bytes_out, binaryT border_val)

Functions

- **MB_errcode** MB_Distanceb (MB_Image *src, MB_Image *dest, enum MB_grid_t grid, enum MB_bordermode_t border)

29.1 Detailed Description

Author:

Nicolas Beucher

Date:

12-27-2008

29.2 Typedef Documentation

- 29.2.1** typedef void(**TSWITCHEP**)(PLINE *plines_out, **UInt32** linoff_out, PLINE *plines_in, **UInt32** linoff_in, **UInt32** bytes_in, **UInt32** bytes_out, binaryT border_val)

typedef for the definition of function arguments

29.3 Function Documentation

- 29.3.1** **MB_errcode** MB_Distanceb (MB_Image * *src*, MB_Image * *dest*, enum MB_grid_t *grid*, enum MB_bordermode_t *border*)

Computes for each pixel the distance to the border of the set in which the pixel is found

The algorithm works with two passes, first from the top left to the bottom right and then back.

Parameters:

src the binary source image

dest the 32-bit image in which the distance for each pixel is stored

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

30 MB_DualBldNb8.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **Uint32** linoff_inout, **PLINE** *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** *p_volume, **PIX8** border_val)

Functions

- **MB_errcode** MB_DualBldNb8 (**MB_Image** *mask, **MB_Image** *srcdest, unsigned int dirnum, **Uint32** *pVolume, enum **MB_grid_t** grid, enum **MB_bordermode_t** border)

30.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-16-2008

30.2 Typedef Documentation

- #### 30.2.1 typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **Uint32** linoff_inout, **PLINE** *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** *p_volume, **PIX8** border_val)

typedef for the definition of function arguments

30.3 Function Documentation

- #### 30.3.1 **MB_errcode** MB_DualBldNb8 (**MB_Image** * *mask*, **MB_Image** * *srcdest*, unsigned int *dirnum*, **Uint32** * *pVolume*, enum **MB_grid_t** *grid*, enum **MB_bordermode_t** *border*)

(re)Builds (dual operation) an image according to a direction and a mask image. The direction depends on the grid used (see **MB_ngh.h** for definitions of directions).

Parameters:

mask the mask image

srcdest the rebuild image

dirnum the direction number

pVolume the computed volume of the output image

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (**NO_ERR** if successful)

31 MB_DualBldNbb.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(TSWITCHEP)(PLINE *plines_inout, Uint32 linoff_inout, PLINE *plines_in, Uint32 linoff_in, Uint32 bytes_in, Uint32 *p_volume, binaryT border_val)

Functions

- MB_errcode MB_DualBldNbb (MB_Image *mask, MB_Image *srcdest, unsigned int dirnum, Uint32 *pVolume, enum MB_grid_t grid, enum MB_bordermode_t border)

31.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-16-2008

31.2 Typedef Documentation

- 31.2.1 typedef void(TSWITCHEP)(PLINE *plines_inout, Uint32 linoff_inout, PLINE *plines_in, Uint32 linoff_in, Uint32 bytes_in, Uint32 *p_volume, binaryT border_val)

typedef for the definition of function arguments

31.3 Function Documentation

- 31.3.1 MB_errcode MB_DualBldNbb (MB_Image * *mask*, MB_Image * *srcdest*, unsigned int *dirnum*, Uint32 * *pVolume*, enum MB_grid_t *grid*, enum MB_bordermode_t *border*)

(re)Builds (dual operation) an image according to a direction and a mask image. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

mask the mask image

srcdest the rebuild image

dirnum the direction number

pVolume the computed volume of the output image

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

32 MB_Error.c File Reference

```
#include "MB_error.h"
```

Functions

- `char * MB_StrErr (MB_errcode error_nb)`

Variables

- `char * err_str []`

32.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-4-2008

32.2 Function Documentation

32.2.1 `char* MB_StrErr (MB_errcode error_nb)`

Returns an explanation of the error code

32.3 Variable Documentation

32.3.1 `char* err_str[]`

Initial value:

```
{
    "No error",
    "Incorrect image size",
    "Incorrect image depth",
    "Bad parameter",
    "Bad value",
    "Incorrect direction for given grid",
    "Memory allocation impossible",
    "Incorrect image dimensions"
}
```

Error value interpretation

33 MB_error.h File Reference

Enumerations

- enum MB_errcode {
 NO_ERR, ERR_BAD_SIZE, ERR_BAD_DEPTH, ERR_BAD_PARAMETER,
 ERR_BAD_VALUE, ERR_BAD_DIRECTION, ERR_CANT_ALLOCATE_MEMORY,
 ERR_BAD_IMAGE_DIMENSIONS }

Functions

- char * MB_StrErr (MB_errcode error_nb)

33.1 Detailed Description

Date:

11-4-2007

This file contains the complete list of error code returned by the API functions.

33.2 Enumeration Type Documentation

33.2.1 enum MB_errcode

Type definition for error code

Enumerator:

NO_ERR Value returned by function when no error was encountered.

ERR_BAD_SIZE Value returned by function when an error of size inside the image was encountered. For example if the width of a first image was not corresponding with the width of a second image when both are used in the same computations.

ERR_BAD_DEPTH Value returned by function when an error of depth inside the image was encountered. For example, the function expected 8-bit pixels and got 1-bit pixels.

ERR_BAD_PARAMETER Value returned by function when a given parameter was incorrect.

ERR_BAD_VALUE Value returned by function when a given value (as argument) was incorrect.

ERR_BAD_DIRECTION Value returned when a function requiring a direction (shift, neighbor) is given an incorrect argument for direction (allowed value depends on the grid).

ERR_CANT_ALLOCATE_MEMORY Value returned when the allocation for image memory failed.

ERR_BAD_IMAGE_DIMENSIONS Value returned when the dimension of the image given in argument of a function is incorrect.

33.3 Function Documentation

33.3.1 char* MB_StrErr (MB_errcode error_nb)

Returns an explanation of the error code

34 MB_Extrema.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Extrema (MB_Image *src, MB_Image *dest, enum MB_grid_t grid, enum MB_bordermode_t border)`

34.1 Detailed Description

Author:

Nicolas Beucher

Date:

02-20-2008

34.2 Function Documentation

34.2.1 `MB_errcode MB_Extrema (MB_Image * src, MB_Image * dest, enum MB_grid_t grid, enum MB_bordermode_t border)`

Computes the local extrema out of a greyscale image. The result is put into a 32-bit image. The function returns the local maxima or the minima coded into the 32-bit image.

The extrema are coded as follow into the 32 bit values. | 0 | 1 | 2 | 3 | |———|———|———|———| | MIN | MAX | isMin | isMax | They can be accessed using the function `MB_CopyBytePlane`

Parameters:

- src* the grey-scale source image
- dest* the 32-bit image produced
- grid* the grid used (either square or hexagonal)
- border* the kind of border to use

Returns:

An error code (`NO_ERR` if successful)

35 MB_Frame.c File Reference

```
#include "mambaApi_loc.h"
#include <stdint.h>
```

Functions

- `MB_errcode MB_Frame (MB_Image *src, int thresval, int *ulx, int *uly, int *brx, int *bry)`

35.1 Detailed Description

Author:

Nicolas Beucher

Date:

3-3-2009

35.2 Function Documentation

35.2.1 `MB_errcode MB_Frame (MB_Image * src, int thresval, int * ulx, int * uly, int * brx, int * bry)`

Returns the smallest frame that contains all the pixels of image that are greater or equal to the given threshold value. using the four last pointer to describe it

Parameters:

src source image

thresval the threshold value used to compute the frame

ulx the x-coordinate of the upper left corner of the frame

uly the y-coordinate of the upper left corner of the frame

brx the x-coordinate of the bottom right corner of the frame

bry the y-coordinate of the bottom right corner of the frame

Returns:

An error code (NO_ERR if successful)

36 MB_Histo.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Histo (MB_Image *src, Uint32 *phisto)`

36.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-25-2007

36.2 Function Documentation

36.2.1 `MB_errcode MB_Histo (MB_Image * src, Uint32 * phisto)`

Computes the histogram of an image. The histogram is an array

Parameters:

src source image

phisto pointer to the histogram array

Returns:

An error code (NO_ERR if successful)

37 MB_Inf.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Inf (MB_Image *src1, MB_Image *src2, MB_Image *dest)`

37.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-25-2007

37.2 Function Documentation

37.2.1 `MB_errcode MB_Inf (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Determines the inferior value between the pixels of two images. The result is put in the corresponding pixel position in the destination image.

Parameters:

src1 image 1

src2 image 2

dest destination image

Returns:

An error code (NO_ERR if successful)

38 MB_InfNb32.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **Uint32** linoff_inout, **PLINE** *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** count, **PIX32** border_val)

Functions

- **MB_errcode** **MB_InfNb32** (**MB_Image** *src, **MB_Image** *srctest, unsigned int dirnum, unsigned int count, enum **MB_grid_t** grid, enum **MB_bordermode_t** border)

38.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

38.2 Typedef Documentation

- 38.2.1** typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **Uint32** linoff_inout, **PLINE** *plines_in, **Uint32** linoff_in, **Uint32** bytes_in, **Uint32** count, **PIX32** border_val)

typedef for the definition of function arguments

38.3 Function Documentation

- 38.3.1** **MB_errcode** **MB_InfNb32** (**MB_Image** * *src*, **MB_Image** * *srctest*, unsigned int *dirnum*, unsigned int *count*, enum **MB_grid_t** *grid*, enum **MB_bordermode_t** *border*)

Looks for the minimum in a 32-bit image in a given direction. The direction depends on the grid used (see **MB_ngh.h** for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

count if src and srctest are the same, the operation is repeated count times

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (**NO_ERR** if successful)

39 MB_InfNb8.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(TSWITCHEP)(PLINE *plines_inout, Uint32 linoff_inout, PLINE *plines_in, Uint32 linoff_in, Uint32 bytes_in, Uint32 count, Uint32 border_val)

Functions

- MB_errcode MB_InfNb8 (MB_Image *src, MB_Image *srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)

39.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

39.2 Typedef Documentation

- 39.2.1 typedef void(TSWITCHEP)(PLINE *plines_inout, Uint32 linoff_inout, PLINE *plines_in, Uint32 linoff_in, Uint32 bytes_in, Uint32 count, Uint32 border_val)

typedef for the definition of function arguments

39.3 Function Documentation

- 39.3.1 MB_errcode MB_InfNb8 (MB_Image * src, MB_Image * srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)

Looks for the minimum in a grey scale image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

count if src and srctest are the same, the operation is repeated count times

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

40 MB_InfNbb.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **UInt32** linoff_inout, **PLINE** *plines_in, **UInt32** linoff_in, **UInt32** bytes_in, **UInt32** count, binaryT border_val)

Functions

- **MB_errcode** MB_InfNbb (**MB_Image** *src, **MB_Image** *srctest, unsigned int dirnum, unsigned int count, enum **MB_grid_t** grid, enum **MB_bordermode_t** border)

40.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

40.2 Typedef Documentation

40.2.1 typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **UInt32** linoff_inout, **PLINE** *plines_in, **UInt32** linoff_in, **UInt32** bytes_in, **UInt32** count, binaryT border_val)

typedef for the definition of function arguments

40.3 Function Documentation

40.3.1 **MB_errcode** MB_InfNbb (**MB_Image** * *src*, **MB_Image** * *srctest*, unsigned int *dirnum*, unsigned int *count*, enum **MB_grid_t** *grid*, enum **MB_bordermode_t** *border*)

Looks for the minimum in a binary image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

count if src and srctest are the same, the operation is repeated count times

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

41 MB_Inv.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Inv (MB_Image *src, MB_Image *dest)`

41.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-25-2007

41.2 Function Documentation

41.2.1 `MB_errcode MB_Inv (MB_Image * src, MB_Image * dest)`

Inverts the pixels value (negation) of the source image.

Parameters:

src source image

dest destination image

Returns:

An error code (NO_ERR if successful)

42 MB_Labelb.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(PLINE *plines_out, **Uint32** linoff_out, PLINE *plines_in, **Uint32** linoff_in, **Uint32** bytes_in)

Functions

- **MB_errcode** MB_Labelb (MB_Image *src, MB_Image *dest, unsigned int lblow, unsigned int lbhigh, int *pNbobj, enum MB_grid_t grid)

Variables

- **PIX32** * MB_Label_EQ
- **PIX32** * MB_Label_CEQ
- **Uint32** MB_Label_maxEQ

42.1 Detailed Description

Author:

Nicolas Beucher

Date:

12-27-2008

42.2 Typedef Documentation

42.2.1 typedef void(**TSWITCHEP**)(PLINE *plines_out, **Uint32** linoff_out, PLINE *plines_in, **Uint32** linoff_in, **Uint32** bytes_in)

typedef for the definition of function arguments

42.3 Function Documentation

42.3.1 **MB_errcode** MB_Labelb (MB_Image * *src*, MB_Image * *dest*, unsigned int *lblow*, unsigned int *lbhigh*, int * *pNbobj*, enum MB_grid_t *grid*)

Labelling the object found in src image.

Parameters:

src the binary source image where the object must be labelled

dest the 32-bit image where object are labelled

lblow the lowest value allowed for label on the low byte (must be inferior to lbhigh)

lbhigh the first high value NOT allowed for label on the low byte (maximum allowed is 256)

pNbobj the number of object founds

grid the grid used (either square or hexagonal)

Returns:

An error code (NO_ERR if successful)

42.4 Variable Documentation

42.4.1 **PIX32*** MB_Label_CEQ

Labelling arrays and variables

42.4.2 PIX32* MB_Label_EQ

Equivalence between label in an image

42.4.3 Uint32 MB_Label_maxEQ

The greatest number of label possible according to current image size

43 MB_LoadExtract.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Load8 (MB_Image *image, PIX8 *indata, int len)`
- `MB_errcode MB_Load32 (MB_Image *image, PIX8 *indata, int len)`
- `MB_errcode MB_Load (MB_Image *image, PIX8 *indata, int len)`
- `MB_errcode MB_Extract8 (MB_Image *image, PIX8 **outdata, int *len)`
- `MB_errcode MB_Extract32 (MB_Image *image, PIX8 **outdata, int *len)`
- `MB_errcode MB_Extract (MB_Image *image, PIX8 **outdata, int *len)`
- `MB_errcode MB_Copy (MB_Image *src, MB_Image *dest)`

43.1 Detailed Description

Author:

Nicolas Beucher

Date:

5-28-2008

43.2 Function Documentation

43.2.1 `MB_errcode MB_Copy (MB_Image * src, MB_Image * dest)`

Copies an image data content into another image

Parameters:

src the source image

dest the destination image

Returns:

An error code (NO_ERR if successful)

43.2.2 `MB_errcode MB_Extract (MB_Image * image, PIX8 ** outdata, int * len)`

Reads an image data content and put it in an array

Parameters:

image the image to read

outdata pointer to the array created (malloc) and filled with the pixel data of the image

len the length in bytes of data extracted

Returns:

An error code (NO_ERR if successful)

43.2.3 `MB_errcode MB_Extract32 (MB_Image * image, PIX8 ** outdata, int * len)`

Reads a 32-bit image data content and put it in an array

Parameters:

image the image to read

outdata pointer to the array created (malloc) and filled with the pixel data of the image

len the length in bytes of data extracted

Returns:

An error code (NO_ERR if successful)

43.2.4 MB_errcode MB_Extract8 (MB_Image * *image*, PIX8 ** *outdata*, int * *len*)

Reads a greyscale image data content and put it in an array

Parameters:

image the image to read

outdata pointer to the array created (malloc) and filled with the pixel data of the image

len the length in bytes of data extracted

Returns:

An error code (NO_ERR if successful)

43.2.5 MB_errcode MB_Load (MB_Image * *image*, PIX8 * *indata*, int *len*)

Loads an image data with data given in argument

Parameters:

image the image to fill

indata the data to fill the image with (complete pixels values)

len the length of data given

Returns:

An error code (NO_ERR if successful)

43.2.6 MB_errcode MB_Load32 (MB_Image * *image*, PIX8 * *indata*, int *len*)

Loads a 32-bit image data with data given in argument

Parameters:

image the image to fill

indata the data to fill the image with (complete pixels values)

len the length of data given

Returns:

An error code (NO_ERR if successful)

43.2.7 MB_errcode MB_Load8 (MB_Image * *image*, PIX8 * *indata*, int *len*)

Loads a greyscale image data with data given in argument

Parameters:

image the image to fill

indata the data to fill the image with (complete pixels values)

len the length of data given

Returns:

An error code (NO_ERR if successful)

44 MB_Lookup.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Lookup (MB_Image *src, MB_Image *dest, Uint32 *ptab)`

44.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-25-2007

44.2 Function Documentation

44.2.1 `MB_errcode MB_Lookup (MB_Image * src, MB_Image * dest, Uint32 * ptab)`

Apply the function in the lookup table to the pixels of source image. A pixel value is changed to a new value accordingly with its current value

Parameters:

src source image

dest destination image

ptab the lookup table pointer

Returns:

An error code (NO_ERR if successful)

45 MB_Mask.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Mask (MB_Image *src, MB_Image *dest, int maskf, int maskt)`

45.1 Detailed Description

Author:

Nicolas Beucher

Date:

5-29-2008

45.2 Function Documentation

45.2.1 `MB_errcode MB_Mask (MB_Image * src, MB_Image * dest, int maskf, int maskt)`

Convert a binary image in a grey scale image (8-bit) or in a 32-bit image using value *maskf* to replace 0 and *maskt* to replace 1.

Parameters:

src binary source image

dest destination image

maskf for 0 (false) pixel value

maskt for 1 (true) pixel value

Returns:

An error code (NO_ERR if successful)

46 MB_Mul.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Mul (MB_Image *src1, MB_Image *src2, MB_Image *dest)`

46.1 Detailed Description

Author:

Nicolas Beucher

Date:

2-27-2009

46.2 Function Documentation

46.2.1 `MB_errcode MB_Mul (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Multiplies the pixels of two images and put the result in the third image. Depending on the format of the target image, the result may be saturated or not. You can perform the following additions : 1-bit * 1-bit = 1-bit (call the `MB_And` function) 1-bit * 8-bit = 8-bit 1-bit * 8-bit = 32-bit 1-bit * 32-bit = 32-bit 8-bit * 8-bit = 8-bit (saturated) 8-bit * 8-bit = 32-bit 8-bit * 32-bit = 32-bit 32-bit * 32-bit = 32-bit

Parameters:

src1 image 1

src2 image 2

dest image resulting of the addition of image 1 and 2.

Returns:

An error code (`NO_ERR` if successful)

47 MB_ngb.h File Reference

```
#include <stdint.h>
```

Defines

- #define **BIN_FILL_VALUE**(border) ((border==MB_FILLED_BORDER) ? UINT32_MAX:0)
- #define **GREY_FILL_VALUE**(border) ((border==MB_FILLED_BORDER) ? UINT32_MAX:0)
- #define **I32_FILL_VALUE**(border) ((border==MB_FILLED_BORDER) ? INT32_MAX:INT32_MIN)

Enumerations

- enum **MB_grid_t** { **MB_HEXAGONAL_GRID** = 1, **MB_SQUARE_GRID** = 0 }
- enum **MB_bordermode_t** { **MB_EMPTY_BORDER** = 0, **MB_FILLED_BORDER** = 1 }

47.1 Detailed Description

Date:

6-29-2008

This file contains the various definitions, global variables macro, struct and functions used when working with direction and neighboring concepts in mambaApi library.

47.2 Define Documentation

47.2.1 #define **BIN_FILL_VALUE**(border) ((border==MB_FILLED_BORDER) ? **UINT32_MAX**:0)

How to fill the border (binary bits images)

47.2.2 #define **GREY_FILL_VALUE**(border) ((border==MB_FILLED_BORDER) ? **UINT32_MAX**:0)

How to fill the border (8 bits images)

47.2.3 #define **I32_FILL_VALUE**(border) ((border==MB_FILLED_BORDER) ? **INT32_MAX**:**INT32_MIN**)

How to fill the border (32 bits images)

47.3 Enumeration Type Documentation

47.3.1 enum **MB_bordermode_t**

enumerate for border mode : either empty or filled

47.3.2 enum **MB_grid_t**

enumerate for grid : either square or hexagonal the value is supposed to give the number of directions allowed

48 MB_Or.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Or (MB_Image *src1, MB_Image *src2, MB_Image *dest)`

48.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-1-2008

48.2 Function Documentation

48.2.1 `MB_errcode MB_Or (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Applies a OR on the pixels of two images. All the images must have the same depth for correct work.

Parameters:

src1 image 1

src2 image 2

dest destination image

Returns:

An error code (NO_ERR if successful)

49 MB_Pixel.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_PutPixel (MB_Image *srcdest, int pixVal, unsigned int x, unsigned int y)`
- `MB_errcode MB_GetPixel (MB_Image *src, int *pixVal, unsigned int x, unsigned int y)`

49.1 Detailed Description

Author:

Nicolas Beucher

Date:

29-1-2009

49.2 Function Documentation

49.2.1 `MB_errcode MB_GetPixel (MB_Image * src, int * pixVal, unsigned int x, unsigned int y)`

Gets the pixel value inside the image at the given position

Parameters:

src the image
pixVal the returned pixel value
x position in x of the pixel targeted
y position in y of the pixel targeted

Returns:

An error code (NO_ERR if successful)

49.2.2 `MB_errcode MB_PutPixel (MB_Image * srcdest, int pixVal, unsigned int x, unsigned int y)`

Puts the pixel value inside the image at the given position

Parameters:

srcdest the image
pixVal the pixel value
x position in x of the pixel targeted
y position in y of the pixel targeted

Returns:

An error code (NO_ERR if successful)

50 MB_Range.c File Reference

```
#include "mambaApi_loc.h"  
#include <stdint.h>
```

Functions

- `MB_errcode MB_Range (MB_Image *src, int *min, int *max)`
- `MB_errcode MB_depthRange (MB_Image *src, int *min, int *max)`

50.1 Detailed Description

Author:

Nicolas Beucher

Date:

5-29-2008

50.2 Function Documentation

50.2.1 `MB_errcode MB_depthRange (MB_Image * src, int * min, int * max)`

gives the minimum and maximum value possible of the image pixels given the image depth

Parameters:

src source image
min the minimum possible value of the pixels
max the maximum possible value of the pixels

Returns:

An error code (NO_ERR if successful)

50.2.2 `MB_errcode MB_Range (MB_Image * src, int * min, int * max)`

gives the minimum and maximum value of the image pixels i.e its range.

Parameters:

src source image
min the minimum value of the pixels
max the maximum value of the pixels

Returns:

An error code (NO_ERR if successful)

51 MB_Sub.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Sub (MB_Image *src1, MB_Image *src2, MB_Image *dest)`

51.1 Detailed Description

Author:

Nicolas Beucher

Date:

13-6-2007

51.2 Function Documentation

51.2.1 `MB_errcode MB_Sub (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Subtracts the values of pixels of the second image to the values of the pixel in the first image

Parameters:

src1 image 1

src2 image 2

dest image resulting of the subtraction

Returns:

An error code (NO_ERR if successful)

52 MB_Sup.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Sup (MB_Image *src1, MB_Image *src2, MB_Image *dest)`

52.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-25-2007

52.2 Function Documentation

52.2.1 `MB_errcode MB_Sup (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Determines the superior value between the pixels of two images. The result is put in the corresponding pixel position in the destination image.

Parameters:

src1 image 1

src2 image 2

dest destination image

Returns:

An error code (NO_ERR if successful)

53 MB_SupNb32.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **UInt32** linoff_inout, **PLINE** *plines_in, **UInt32** linoff_in, **UInt32** bytes_in, **UInt32** count, **PIX32** border_val)

Functions

- **MB_errcode** **MB_SupNb32** (**MB_Image** *src, **MB_Image** *srctest, unsigned int dirnum, unsigned int count, enum **MB_grid_t** grid, enum **MB_bordermode_t** border)

53.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

53.2 Typedef Documentation

- 53.2.1** typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **UInt32** linoff_inout, **PLINE** *plines_in, **UInt32** linoff_in, **UInt32** bytes_in, **UInt32** count, **PIX32** border_val)

typedef for the definition of function arguments

53.3 Function Documentation

- 53.3.1** **MB_errcode** **MB_SupNb32** (**MB_Image** * *src*, **MB_Image** * *srctest*, unsigned int *dirnum*, unsigned int *count*, enum **MB_grid_t** *grid*, enum **MB_bordermode_t** *border*)

Looks for the maximum in a 32-bit image in a given direction. The direction depends on the grid used (see **MB_ngh.h** for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

count if *src* and *srctest* are the same, the operation is repeated count times

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (**NO_ERR** if successful)

54 MB_SupNb8.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void (TSWITCHEP)(PLINE *plines_inout, Uint32 linoff_inout, PLINE *plines_in, Uint32 linoff_in, Uint32 bytes_in, Uint32 count, Uint32 border_val)

Functions

- MB_errcode MB_SupNb8 (MB_Image *src, MB_Image *srctest, unsigned int dirnum, unsigned int count, enum MB_grid_t grid, enum MB_bordermode_t border)

54.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

54.2 Typedef Documentation

- 54.2.1 typedef void (TSWITCHEP)(PLINE *plines_inout, Uint32 linoff_inout, PLINE *plines_in, Uint32 linoff_in, Uint32 bytes_in, Uint32 count, Uint32 border_val)

typedef for the definition of function arguments

54.3 Function Documentation

- 54.3.1 MB_errcode MB_SupNb8 (MB_Image * *src*, MB_Image * *srctest*, unsigned int *dirnum*, unsigned int *count*, enum MB_grid_t *grid*, enum MB_bordermode_t *border*)

Looks for the maximum in a grey scale image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

count if *src* and *srctest* are the same, the operation is repeated count times

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

55 MB_SupNbb.c File Reference

```
#include "mambaApi_loc.h"
```

Typedefs

- typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **UInt32** linoff_inout, **PLINE** *plines_in, **UInt32** linoff_in, **UInt32** bytes_in, **UInt32** count, binaryT border_val)

Functions

- **MB_errcode** MB_SupNbb (**MB_Image** *src, **MB_Image** *srctest, unsigned int dirnum, unsigned int count, enum **MB_grid_t** grid, enum **MB_bordermode_t** border)

55.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-18-2008

55.2 Typedef Documentation

- 55.2.1 typedef void(**TSWITCHEP**)(**PLINE** *plines_inout, **UInt32** linoff_inout, **PLINE** *plines_in, **UInt32** linoff_in, **UInt32** bytes_in, **UInt32** count, binaryT border_val)

typedef for the definition of function arguments

55.3 Function Documentation

- 55.3.1 **MB_errcode** MB_SupNbb (**MB_Image** * *src*, **MB_Image** * *srctest*, unsigned int *dirnum*, unsigned int *count*, enum **MB_grid_t** *grid*, enum **MB_bordermode_t** *border*)

Looks for the maximum in a binary image in a given direction. The direction depends on the grid used (see MB_ngh.h for definitions of directions).

Parameters:

src source image

srctest destination image

dirnum the direction index

count if src and srctest are the same, the operation is repeated count times

grid the grid used (either square or hexagonal)

border the kind of border to use (behavior for pixel near edge depends on it)

Returns:

An error code (NO_ERR if successful)

56 MB_Thresh.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Thresh (MB_Image *src, MB_Image *dest, int low, int high)`

56.1 Detailed Description

Author:

Nicolas Beucher

Date:

6-7-2008

56.2 Function Documentation

56.2.1 `MB_errcode MB_Thresh (MB_Image * src, MB_Image * dest, int low, int high)`

Fills a binary image according to the following rules : if pixel value lower than low or higher than high the binary pixel is set to 0 in others case the pixel is set to 1.

Parameters:

src source image

dest destination image

low low value for threshold

high high value for treshold

Returns:

An error code (NO_ERR if successful)

57 MB_Volume.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Volume (MB_Image *src, Uint32 *pVolume)`

Variables

- `Uint32 MB_VolumePerByte [256]`

57.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-25-2007

57.2 Function Documentation

57.2.1 `MB_errcode MB_Volume (MB_Image * src, Uint32 * pVolume)`

Computes the volume of an image. The volume is the sum of the pixel values (i.e. integration of the image)

Parameters:

src source image

pVolume pointer to the volume variable

Returns:

An error code (`NO_ERR` if successful)

57.3 Variable Documentation

57.3.1 `Uint32 MB_VolumePerByte[256]`

Initial value:

```
{
    00,01,01,02,01,02,02,03,01,02,02,03,02,03,03,04,
    01,02,02,03,02,03,03,04,02,03,03,04,03,04,04,05,
    01,02,02,03,02,03,03,04,02,03,03,04,03,04,04,05,
    02,03,03,04,03,04,04,05,03,04,04,05,04,05,05,06,
    01,02,02,03,02,03,03,04,02,03,03,04,03,04,04,05,
    02,03,03,04,03,04,04,05,03,04,04,05,04,05,05,06,
    02,03,03,04,03,04,04,05,03,04,04,05,04,05,05,06,
    03,04,04,05,04,05,05,06,04,05,05,06,05,06,06,07,
    01,02,02,03,02,03,03,04,02,03,03,04,03,04,04,05,
    02,03,03,04,03,04,04,05,03,04,04,05,04,05,05,06,
    02,03,03,04,03,04,04,05,03,04,04,05,04,05,05,06,
    03,04,04,05,04,05,05,06,04,05,05,06,05,06,06,07,
    02,03,03,04,03,04,04,05,03,04,04,05,04,05,05,06,
    03,04,04,05,04,05,05,06,04,05,05,06,05,06,06,07,
    03,04,04,05,04,05,05,06,04,05,05,06,05,06,06,07,
    04,05,05,06,05,06,06,07,05,06,06,07,06,07,07,8
}
```

Volume arrays

58 MB_Watershed.c File Reference

```
#include "mambaApi_loc.h"
```

Defines

- `#define NORM_WATER_LEVEL(value) ((value < current_water_level) ? current_water_level : value)`

Typedefs

- `typedef void(TSWITCHEP)(PLINE *plines_im, Uint32 linoff_im, PLINE *plines_marker, Uint32 linoff_marker, int x, int y)`

Functions

- `MB_errcode MB_Watershed (MB_Image *src, MB_Image *marker, unsigned int max_level, enum MB_grid_t grid)`

Variables

- `PIX8 current_water_level = 0`
- `MB_ListControl MB_HierarchicalList [256]`
- `MB-Token * MB_TokensArray`
- `const int sqNbDir [9][2]`
- `const int hxNbDir [2][7][2]`

58.1 Detailed Description

Author:

Nicolas Beucher

Date:

01-02-2009

58.2 Define Documentation

58.2.1 `#define NORM_WATER_LEVEL(value) ((value < current_water_level) ? current_water_level : value)`

Normalize the value using the current water level

58.3 Typedef Documentation

58.3.1 `typedef void(TSWITCHEP)(PLINE *plines_im, Uint32 linoff_im, PLINE *plines_marker, Uint32 linoff_marker, int x, int y)`

typedef for the definition of neighbor function arguments

58.4 Function Documentation

58.4.1 `MB_errcode MB_Watershed (MB_Image * src, MB_Image * marker, unsigned int max_level, enum MB_grid_t grid)`

Perform a watershed segmentation of the image using the marker image as a starting point for the flooding. The result is put into a the 32 bit image that start with the marker.

The segmentation is coded as follow into the 32 bit values. | 0 | 1 | 2 | 3 | |———|———|———|———| | Segment label | isLine | They can be accessed using the function `MB_CopyBytePlane`. `isLine` is a value indicating if the pixel belong to the watershed (255 if this is the case, undefined otherwise).

Parameters:

src the greyscale image to segment

marker the marker image in which the result of segmentation will be put
max_level the maximum level reach by the water.
grid the grid used (either square or hexagonal)

Returns:

An error code (NO_ERR if successful)

58.5 Variable Documentation

58.5.1 PIX8 current_water_level = 0

Variable indicating which level in the hierarchical list the "water" as attained. Only this level and above can be fille with new tokens.

58.5.2 const int hxNbDir[2][7][2]

Initial value:

```

{
  {{0,0},{0,-1},{1,0},{0,1},{-1,1},{-1,0},{-1,-1}},
  {{0,0},{1,-1},{1,0},{1,1},{0,1},{-1,0},{0,-1}}
}
    
```

Table giving the offset for the neighbor in hexagonal grid (x and y)

58.5.3 MB_ListControl MB_HierarchicalList[256]

The hierarchical list entries for watershed segmentation

58.5.4 MB_Token* MB_TokensArray

The memory used to hold the elements of the hierarchical list

58.5.5 const int sqNbDir[9][2]

Initial value:

```

{
  {0,0},{0,-1},{1,-1},{1,0},{1,1},{0,1},{-1,1},{-1,0},{-1,-1}}
}
    
```

Table giving the offset for the neighbor in square grid (x and y)

59 MB_Window.c File Reference

```
#include "mambaApi_loc.h"
```

Defines

- `#define MB_ROUND_X 64`
- `#define MB_ROUND_Y 2`
- `#define MB_MAX_IMAGE_X_SIZE 4100`
- `#define MB_MAX_IMAGE_Y_SIZE MB_MAX_IMAGE_X_SIZE`

Functions

- `MB_errcode MB_SetSizeAndWindow (int *w, int *h)`
- `MB_errcode MB_SetWindow (int x1, int y1, int x2, int y2)`
- `MB_errcode MB_ResetWindow ()`

Variables

- `Uint32 MB_w = 0`
- `Uint32 MB_h = 0`
- `Uint32 MB_sz_x = 0`
- `Uint32 MB_sz_y = 0`
- `Sint32 MB_wx_start = 0`
- `Sint32 MB_wy_start = 0`
- `Uint32 MB_wx_size = 0`
- `Uint32 MB_wy_size = 0`

59.1 Detailed Description

Author:

Nicolas Beucher

Date:

2-28-2008

59.2 Define Documentation

59.2.1 `#define MB_MAX_IMAGE_X_SIZE 4100`

Image limit size in x

59.2.2 `#define MB_MAX_IMAGE_Y_SIZE MB_MAX_IMAGE_X_SIZE`

Image limit size in Y

59.2.3 `#define MB_ROUND_X 64`

Making sure the image size is multiple of 64 for the width

59.2.4 `#define MB_ROUND_Y 2`

Making sure the image size is multiple of 2 for the height

59.3 Function Documentation

59.3.1 `MB_errcode MB_ResetWindow (void)`

Reset the window to the maximum size (the full image).

Returns:

An error code (`NO_ERR` if successful)

59.3.2 MB_errcode MB_SetSizeAndWindow (int * w, int * h)

Set the image sizes correctly and puts the window size to the maximum (all the image is processed).

Parameters:

w the width
h the height

Returns:

An error code (NO_ERR if successful)

59.3.3 MB_errcode MB_SetWindow (int x1, int y1, int x2, int y2)

Set the window size so that it includes the position given in argument. The window size must also respect certain rules to be valid.

Parameters:

x1 the x position of the upper left pixel (INSIDE the window)
y1 the y position of the upper left pixel (INSIDE the window)
x2 the x position of the lower right pixel (INSIDE the window)
y2 the y position of the lower right pixel (INSIDE the window)

Returns:

An error code (NO_ERR if successful)

59.4 Variable Documentation

59.4.1 Uint32 MB_h = 0

original Height of the images computed

59.4.2 Uint32 MB_sz_x = 0

Width of the images computed

59.4.3 Uint32 MB_sz_y = 0

Height of the images computed

59.4.4 Uint32 MB_w = 0

original Width of the images computed

59.4.5 Uint32 MB_wx_size = 0

Width of the window computation

59.4.6 Sint32 MB_wx_start = 0

offset to the first pixel position in x of the window computation

59.4.7 Uint32 MB_wy_size = 0

Height of the window computation

59.4.8 Sint32 MB_wy_start = 0

offset to the first pixel position in y of the window computation

60 MB_Xor.c File Reference

```
#include "mambaApi_loc.h"
```

Functions

- `MB_errcode MB_Xor (MB_Image *src1, MB_Image *src2, MB_Image *dest)`

60.1 Detailed Description

Author:

Nicolas Beucher

Date:

11-24-2007

60.2 Function Documentation

60.2.1 `MB_errcode MB_Xor (MB_Image * src1, MB_Image * src2, MB_Image * dest)`

Applies a XOR on the pixels of two images. All the images must have the same depth for correct work.

Parameters:

src1 image 1

src2 image 2

dest destination image

Returns:

An error code (NO_ERR if successful)