bmmap

keys:	**[from]**	*<number>*	source picture
	[to]	*<number>*	destination picture
	with	*<number>*	picture containing one or more neighbourhood mapping tables
	mask	*<number>*	mask picture to restrict processing to specified region
	times	*<number>*	number of times to apply neighbourhood mapping tables (zero for infinite number of iterations)
	edge	*<number>*	edge value
options:	**source/output**		fix value of source/output edge pixels according to the value of the **edge** key

You use **bmmap** to carry out generalised binary 3 by 3 neighbourhood transformations. The way in which the transformation is carried out is determined by one or more neighbourhood mapping tables contained in the picture specified by means of the **with** key. The result of applying each mapping table is obtained by calculating the neighbourhood index for each source pixel and using this index to look up the result in the mapping table. The mapping table defines the result for all the 512 possible configurations in a binary 3 by 3 neighbourhood. The command **bmlut** provides a convenient way to generate mapping tables for use with **bmmap.**

Examples

```
create 99 size 512, 1 value 0; origin left; pixel 511 = 1
bmmap 1 with 99 times 1
```

This example removes one layer of pixels from all foreground objects in picture 1 (same result as **erode** 1 times 1).

```
bmlut 99 erode outline; bmmap 1 2 with 99
```

This example outputs outlines of foreground regions in picture 1 to picture 2 (same result as **erode 1 2 outline**).

```
bmlut 99 if (p4 & n8~=0) | (~p4 & n8=8); bmmap 1 with 99
```

This example removes salt-and-pepper noise from picture 1.

```
bmlut 99 dilate separately; bmmap 1 with 99 times 0 mask 3
```

This example dilates foreground regions in picture 1, keeping unconnected regions separate, until they

bmmap

fill the foreground regions in picture 3.

```
bmlut 99 median; bmmap 1 with 99 times 5
```

This example applies a binary median filter to picture 1 and gives the same result as the following

```
for i=1,5; median 1; loop
```

Description

The source picture is treated as a binary image: zero values denote background pixels and non-zero values denote foreground pixels.

Each row of the picture specified by means of the **with** key contains a mapping table which defines the result for each of the 512 possible binary 3 by 3 neighbourhood configurations. A 9-bit neighbourhood index value is constructed for each source pixel, where each bit in the index value is set according to the state of the corresponding pixel in the neighbourhood. The neighbourhood bits are arranged in the following way

```
0 3 6
1 4 7
2 5 8
```

Calculating the index value for a neighbourhood is equivalent to applying the following linear convolution kernel

```
2^0 2^3 2^6      1  8  64
2^1 2^4 2^7  =   2 16 128
2^2 2^5 2^8      4 32 256
```

provided that a pixel value in the neighbourhood is given a value of 0 or 1 according to whether the corresponding source pixel is zero or non-zero.

The index value is used to select the mapping table entry which defines the output result for the central pixel. If the mapping table entry is set to zero, a zero is output. Otherwise, a 1 is output. In this way, quite arbitrary 3 by 3 neighbourhood transformations can be caried out. It is important to note that the output values do not affect the result for neighbouring pixels during each pass through the image. In effect, the transformation is simultaneously applied to all the source pixels.

If the mapping table picture contains more than one row, a series of neighbourhood transformations will be applied in turn.

bmmap

For example, the morphological opening with a 3 by 3 structuring element can be specified if the first row of the picture contains the mapping table for erosion and the second row contains the mapping table for dilation

```
create 1 size 512, 1 value 0; origin left; pixel 511 =
create 2 size 512, 1 value 1; origin left; pixel   0 = 0
create 3 size 512, 2
paste 1 3 top
paste 2 3 bottom
bmmap ... with 3
```

The whole process can be repeated by specifying the number of iterations with the **times** key. The default is to carry out one iteration. A zero value for the **times** by specifies on on infinite number of iterations. Processing is stopped if no change is detected after applying one complete sequence of mapping tables.

The command **bmlut** provides a convenient way to generate mapping tables. For example the mapping tables for morphological opening, as given in the previous example, could be generated with the following command

```
bmult 3 if (p4 & n8 = 8), (p4 | n8 ~= 0)
```

The result for each neighbourhood configuration is defined as a logical expression which is evaluated for each of the 512 possible configurations. **Bmlut** also has keys and options which allow you to generate mapping tables for any of the neighbourhood tranformations supported by the commands **erode, dilate** and **median (bmmap** is usually much faster than **erode, dilate** or **median).**

The **mask** key can be used to restrict processing to specified regions of the image, that is, wherever mask pixels are zero, the source pixel value is output and elsewhere, the result of the 3 by 3 neighbourhood transformation is substituted. Of course, the mask picture must have the same dimensions as the source picture. For example, the mask image

```
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 1 1 1 1 1 1
0 1 1 1 1 1 1
0 1 1 1 1 1 1
0 0 0 0 0 0 0
```

when applied to the following example, where the neighbourhood transformation produces the eight-connected outline of the source image.

bmmap

source	result
0 0 0 0 0 0 0	0 0 0 0 0 0 0
0 0 0 1 1 0 0	0 0 0 1 1 0 0
0 0 1 1 1 1 0	0 0 1 0 0 1 0
0 1 1 1 1 1 0	0 1 0 0 0 1 0
0 1 1 1 1 0 0	0 1 0 0 1 0 0
0 1 1 1 1 0 0	0 1 1 1 1 0 0
0 0 0 0 0 0 0	0 0 0 0 0 0 0

gives the following result

0 0 0 0 0 0 0		. . . . .		0 0 0 0 0 0 0
0 0 0 1 1 0 0		. . . . .		0 0 0 1 1 0 0
0 0 1 1 1 1 0		. . . . .		0 0 1 1 1 1 0
0	+	. 1 0 0 0 1 0	=	0 1 0 0 0 1 0
0		. 1 0 0 1 0 0		0 1 0 0 1 0 0
0		. 1 1 1 1 0 0		0 1 1 1 1 0 0
0 0 0 0 0 0 0		. . . . .		0 0 0 0 0 0 0

So far, the result for a 3 by 3 neighbourhood transformation has not been defined round the edges of an image where any part of the neighbourhood falls outside the limits of the source picture. By default, source pixels round the edges of the image are replicated outwards.

As an alternative, the **source** option allows you to specify a value for any pixel positions outside the source image which contribute to the neighbourhood of a pixel. You specify the actual value with the **edge** key. All non-zero values for the **edge** key imply an edge value of 1. For example, if a zero edge value is specified, the neighbourhood transformation to produce the eight-connected outline of the source image would result in closed outlines for all regions. If a non-zero edge value is specified, outlines of regions which touch the edges of the picture are left open.

bmmap

source	result	
0 1 1 1 0 0 0	0 1 1 1 0 0 0	
1 1 1 1 0 0 0	1 0 0 1 0 0 0	
1 1 1 1 1 1 0	1 0 0 0 1 1 0	
0 1 1 1 1 1 1	0 1 1 0 0 0 1	**source edge 0**
0 0 0 1 1 1 1	0 0 0 1 0 0 1	
0 0 0 0 1 1 1	0 0 0 0 1 0 1	
0 0 0 0 1 1 1	0 0 0 0 1 1 1	
	0 1 0 1 0 0 0	
	1 0 0 1 0 0 0	
	1 0 0 0 1 1 0	
	0 1 1 0 0 0 1	**source edge 1**
	0 0 0 1 0 0 0	
	0 0 0 0 1 0 0	
	0 0 0 0 1 0 0	

Instead of using the **source** option, you can, with the **output** option and the **edge** key, specify the value to assign to output pixels wherever the result depends on undefined source pixels. A zero output edge value is particularly useful when using 3 by 3 neighbourhood transformations to detect particular neighbourhood configurations. It eliminates possible edge effects because output pixels can only be set if the entire neighbourhood region lies inside the source image.

Notes

see also: **bmlut, erode, dilate, median**

bmmap

Defaults and Ranges

keys/options	defaults	range
[from]	current picture, held in the variable *select*	valid picture number
[to]	source picture	valid picture number
with	none	valid picture number
mask	none	valid picture number
times	1	zero or positive integer
edge	0	real number
source/output	replicate edge pixels	