

7.3 Image Processing

7.3.1 Filter

Functionality

Filtering is a process in which each pixel value in a raster map is replaced with a new value. The new value is obtained by applying a certain function to each input pixel and its direct neighbours. These neighbours are usually the 8 adjacent pixels (in a 3 x 3 filter) or the 24 surrounding pixels (in a 5 x 5 filter). When you create your own filters, any odd sized matrix is allowed (5 x 1, 11 x 23, 25 x 25); the maximum user-defined filter size is 8000.

Filters can be used for a number of purposes:

- to enhance 'sharpness' of a satellite image for better visual interpretation,
- to reduce noise in an image prior to a multi-band image classification when classifying rather large areas
- to detect line features or edges in a satellite image,
- to calculate steepness, direction (aspect), and shape (concave, convex) of slopes in a DEM,
- to assign pixels which were classified as unknown during a multi-band image classification the value of the most frequent occurring class around the unknown pixel,
- morphologic filtering on bitmaps, etc.

Sharpening images

EDGESENH the edges enhancement filter leaves the overall look of the image the same, but pixels at the borders of light areas become light as well, and pixels at the borders of dark areas become dark.

Homogenous areas remain the same. This gives the impression that the image is sharper.

Noise reduction in images (smoothing)

AVG3x3 calculates the average value of 9 pixels and assigns the result to the central pixel. The AVG filter smoothes the image but may blur it.

MED3x3 reads and orders 9 pixel values from low to high and replaces the central pixel with the middle value (= the pixel value with rank order 5 = median value).

MED5x5 reads and orders 25 pixel values from low to high and replaces the central pixel with the middle value (= the pixel value with rank order 13 = median value).

The MED3x3 and MED5x5 filters remove point noise better than the AVG filter and they do not blur the image, but thin line features may disappear. Median filters can be used prior to a multi-band image classification if the classes to distinguish are 'large' areas (forest, agriculture) and not small or fine features (farm, roads).

Detecting line features and edges

Gradient filters: see DFDX, DFDY, D2FDX2, D2FDY2, etc. below.

LAPLACE The LAPLACE filter calculates gradients both in x- and y-direction (second partial derivative). The effect of using the LAPLACE filter is that when the value of the central pixel is relatively high compared to its neighbours, the value assigned to the central pixel will be even lower than the neighbours; and that when the value of the central pixel is relatively low compared to its neighbours, the value assigned to the central pixel will be even higher than the neighbours. Thus values which 'largely' differ from their neighbours come out as an opposite in the output map, while areas with more or less the same pixel value 'disappear'. The LAPLACE filter is quite sensitive to noise.

Using filters on DEMs

DFDX	detects slope differences in x-direction. Positive values in the output map mean that the terrain goes up from left to right (West to East); negative values in the output mean the terrain goes down.
DFDY	detects slope differences in y-direction. Positive values in the output map means that the terrain goes up from bottom to top (South to North); negative values mean that terrain goes down.
D2FDX2	detects slope shape differences in x-direction (second derivative). Positive values in the output map mean that the terrain is concave in x-direction; negative values mean that the terrain is convex.
D2FDY2	detects slope shape differences in y-direction (second derivative). Positive values in the output map mean that the terrain is concave in y-direction; negative values mean that the terrain is convex.
LAPLACE	the LAPLACE filter can be used to investigate which parts of the terrain are convex (negative value in output map) or concave (positive value in output map). Flat areas and uniform slopes obtain output value 0.
SHADOW	applies artificial illumination (from the Northwest) to the DEM. As a result, higher parts in the DEM become more pronounced because they obtain a shadow.

Using filters after a multi-band classification

MAJUNDEF	the undefined-majority assigns pixels classified as unknown during a multi-band image classification the value of the most frequent occurring class around the unknown pixel. It considers 8 neighbours around an unclassified pixel and assigns the predominant class name of the neighbours to the unclassified pixel in the output map.
----------	--

Morphologic filtering

DILATE4	assigns a 1 to the center pixel if one or more horizontal or vertical neighbours is true as well. The result of using DILATE4 is that true pixels 'grow' in horizontal and vertical direction.
DILATE8	assigns a 1 to the center pixel if any of the neighbours is true. The result of using DILATE8 is that true pixels 'grow' in all directions.
SHRINK4	assigns a 1 only when the center pixel is 1 and all horizontal and vertical neighbours are 1 as well. The result of using the SHRINK4 filter is that areas of true pixels 'shrink' along their edges; single lines of true pixels and single true pixels totally surrounded by false pixels disappear.
SHRINK8	assigns a 1 only when the center pixel is 1 and all neighbours are 1 as well. The result of using the SHRINK8 filter is that areas of true pixels 'shrink' along their edges; single lines of true pixels and single true pixels totally surrounded by false pixels disappear.
INBND4	assigns a 1 only when the center pixel is 1 and at least one of the horizontal or vertical neighbours is 0 (meaning it is located at the edge of a group of true pixels). The result of using this filter is that 4-connected areas of true pixels are replaced by their 4-connected outline (a 4-connected boundary of true pixels).
INBND8	assigns only a 1 when the center pixel is 1 and at least one of the 8 neighbours is 0 (meaning it is located at the edge of a group of true pixels). The result of using this filter is that areas of 8-connected true pixels are replaced by their 8-connected outline (an 8-connected boundary of true pixels).
OUTBND4	results in a 4-connected outline of groups of false pixels
OUTBND8	results in an 8-connected outline of groups of false pixels
CONN8TO4	when true pixels are only 8-connected, this filter makes them 4-connected by adding a true pixel in some places. This might give your binary image a better outlook.
BINMAJOR	replaces the pixel with the predominant value of its neighbours. This filter removes noise from your binary image.
PEPPSALT	assigns a 1 to false pixels when all neighbours are 1, and assigns a 0 to true pixels when all neighbours are 0. This filter removes single false and single true pixels from your binary image.

Standard filters are found in the ILWIS\SYSTEM directory. Filters can also be user-defined.

Input map requirements

Input map requirements depend on the used filter type:

- Linear filters: raster map with a value domain or the Image domain;
- Rank order filters: raster map with a value domain or the Image domain;
- Majority filters: any raster map; no restrictions on domain type;
- Binary filters: raster map with a value domain or the Image domain, a Bool domain, the Bit domain or a class or ID domain.

Domain and georeference of output map

An output domain is proposed depending on the domain of the input map and the filter type used. The value range and precision of this domain can be adjusted each time you perform the Filter operation. The output map uses the same georeference as the input map.

Dialog box

Filtering is a process in which each pixel value in a raster map is replaced with a new value. The new value is obtained by applying a certain function to each input pixel and its direct neighbours.

Dialog box options

- Input raster map: Select a raster map to be filtered. Open the list box and select an input map, or directly drag a raster map from the Catalog into this box.
- Filter type: Select the type of filter to use: linear, average, majority, rank order, median, pattern, binary, or standard deviation.
- Predefined: When you selected as the Filter type: Majority or Rank Order: select this check box if you want to use a standard predefined Majority filter (Majority, MajUndef, MajZero) or a standard predefined Rank order filter(Med3x3, Med5x5) on disk. Clear this check box if you want to construct your own majority or rank order filter.
- Filter name: For linear, predefined majority, predefined rank order and binary filter types: select the name of an existing filter on disk. When you selected as the Filter type: Linear; you can also create your own linear filter by clicking the create button .
- Rows Columns: Type values for the number of rows and columns to define the filter size. Row and column values must be odd values but do not need to be the same.
- Rank: Optionally, specify the rank order number of the considered pixels that should be used as output value (for user-defined rank order filters).
- Threshold: Optionally, specify a threshold value. An output value of the filter will only be used if the difference between the original and new pixel value is smaller than or equal to the threshold. If the difference between the original and new pixel value is larger than the threshold the original value will be used.
- Undefined: Select this check box to replace only undefined values.
- Output raster map: Type a name for the raster output map that will contain the filtered values.
- Domain: Accept the default domain, or select another domain for the output map, or create  your own domain.
- Value range: Accept the default value range, or specify your own range of possible values in the output map.

Precision:	Accept the default precision of output values, or specify your own precision.
Show:	Select this check box if you want the output map to be displayed in a map window when the operation has finished. Clear this check box if you do not want to see this map immediately: you simply define how the output map should be created.
Description:	Optionally, type a description for the output map. The description appears in the title bar when the output map is displayed.

A dependent output map is created.

Command line

The Filter operation can be directly executed by typing the following expression on the command line of the Main Window.

`OUTMAP = MapFilter(InputMapName, FilterName)`

where:

OUTMAP	is the name of your output raster map
MapFilter	is the command to start the Filter operation
InputMapName	is the name of your input raster map
<i>FilterName</i>	is the name of the filter you want to use

For the *FilterName*, a standard filter on disk can be used, or a user-defined filter on disk. However, you can also create your own filters on the command line. For more information, refer to User-defined filters on the command line.

All filters use input value maps; the Majority and the UndefMajority filters also work on maps of other domain types.

When the definition symbol `=` is used, a dependent output map is created; when the assignment symbol `:=` is used, the dependency link is immediately broken after the output map has been calculated.

Algorithm

Filtering is a process in which each pixel value in a raster map is replaced with a new value. The new value is obtained by applying a certain function to each input pixel and its direct neighbours.

Mathematically, several filter types can be distinguished:

- linear filters
- rank order filters
- majority filters
- binary filters

As each filter type works in its own way, it is recommended to refer to the individual filter types.

7.3.2 Standard filters

Standard filters are located in the ILWIS\SYSTEM directory, extension .FIL.

Linear convolution filters

AVG3x3	calculates the average value of each 9 pixel values considered (3x3 matrix)
EDGESENH	calculates Pixelvalue - Laplace; enhances edges (3x3 matrix)
LAPLACE	calculates second derivative in all directions (3x3 matrix)
SHADOW	applies artificial illumination from the Northwest (3x3 matrix)
DFDX	calculates first derivative in x-direction (5x1 matrix)
DFDY	calculates first derivative in y-direction (1x5 matrix)
DFDDN	calculates first derivative in downward diagonal (5x5 matrix)
DFDUP	calculates first derivative in upward diagonal (5x5 matrix)
D2FDX2	calculates second derivative in x-direction (5x5 matrix)
D2FDY2	calculates second derivative in y-direction (5x5 matrix)
D2FDXY	calculates second derivative in all directions (5x5 matrix)

Rank order filters

MED3x3	replaces the value of the center pixel by the median value of each 9 pixel values considered (3x3 matrix)
MED5x5	replaces the value of the center pixel by the median value of each 9 pixel values considered (5x5 matrix)

Majority filters

MAJORITY	The majority filter selects the predominant (most frequently occurring) value or class name of a pixel and its 8 neighbours. (3x3 matrix)
MAJUNDEF	The undef-majority filter only assigns the predominant value or class name when the central pixel is undefined. (3x3 matrix)
MAJZERO	The zero-majority filter only assigns the predominant value or class name when the central pixel has value 0. (3x3 matrix)

Binary filters

DILATE4	dilation of 4-connected true pixels (3x3 matrix)
DILATE8	dilation of 8-connected true pixels (3x3 matrix)
SHRINK4	shrinking of 4-connected true pixels (3x3 matrix)
SHRINK8	shrinking of 8-connected true pixels (3x3 matrix)
INBND4	gives the inner boundary of areas of 4-connected true pixels (3x3 matrix)
INBND8	gives the inner boundary of areas of 8-connected true pixels (3x3 matrix)
OUTBND4	gives the outer boundary of areas of 4-connected true pixels (3x3 matrix)
OUTBND8	gives the outer boundary of areas of 8-connected true pixels (3x3 matrix)
CONN8TO4	makes 8-connected true pixels 4-connected (3x3 matrix)

BINMAJOR	replaces the central pixel with the predominant value (0, or 1) of the 9 pixels considered (3x3 matrix)
PEPPSALT	removes noise out of a binary image (3x3 matrix)
LIFEGAME	game: small groups of true pixels die of loneliness, large groups of true pixels die of overcrowding, and when there is space (false pixels), true pixels reproduce and make one of their false neighbours true (3x3 matrix)

Filters are in ASCII, so you can have a look at them with any text editor, for example Notepad.

7.3.3 Filter types

Mathematically, filters can be described as:

- Linear (convolution) filters
- Rank order filters
- Majority filters
- Binary filters

The working of each filter type is explained in the separate topics.

Linear convolution filters

Linear convolution filters consist of a matrix with values and a gain factor. When considering a linear filter of size 3x3, the 9 matrix values are multiplied with 9 pixel values in the input raster map, this is summed and then multiplied by the gain factor. The result is assigned to the center pixel in the output map.

When a threshold is set (conditional filter), the value of the center pixel will only be replaced by the new value if the difference between the original and new pixel value is smaller than or equal to the threshold. At the moment, thresholds are not available for linear filters.

The following standard filters are linear convolution filters:

AVG3x3, EDGESENH, LAPLACE, SHADOW.

Also the standard gradient filters are linear convolution filters:

DFDX, DFDY, DFDDN, DFDUP, D2FDX2, D2FDY2, D2FDXY.

Further, in the Filter : dialog box, you can create a user-defined linear filter (click the create button), and you can modify the Average filter to define a user-defined average filter.

For user-defined linear filters, you can specify the size of the filter, matrix values and a gain factor. For user-defined average filters, you can specify the size of the filter. User-defined average filters can also be created on the command line.

Rank order filters

Rank order filters have a certain size, but do not have any matrix values nor gain factor. A rank order filter of size 3x3 for example, examines 9 pixel values of the input map at a time, sorts these values from small to large, and selects as output value that value which is encountered at a certain rank order number. So one value of the pixel values examined becomes the output value, without any calculation performed on the values itself.

When a threshold is set, the value of the center pixel will only be replaced with the new value if the difference between the original and new value is smaller than or equal to the threshold.

Standard rank order filters are the median filters: MED3x3, MED5x5. The median filters can for instance be used to smooth an image.

In the Filter dialog box and on the command line, user-defined rank order filters can be created: the size of a rank order filter, the rank order number and a threshold can be indicated. In this way you can obtain for example the minimum or maximum value of a number of pixels.

Majority filters

The majority filter (MAJORITY) is a 3 by 3 filter. As output value for each central pixel, the filter selects the predominant (most frequently occurring) value or class name of each input pixel and its 8 neighbours. If all 9 pixels have a different value or class name, the first value or class name encountered is used as output.

The undef-majority filter (MAJUNDEF) is a 3 by 3 filter which only selects the predominant value or class name if the central pixel is undefined. If the central pixel is not undefined, the value or class name remains the same.

The zero-majority filter (MAJZERO) is a 3 by 3 filter which only selects the predominant value or class name if the central pixel has value zero. If the central pixel does not have value zero, the value or class name remains the same.

In the Filter dialog box and on the command line, user-defined majority filters. can be created: the size of the filter and a condition can be defined.

Binary filters

Binary filters regard the input map as a binary map. This means that zero values are regarded as zero, and all other values as one. Depending on the central pixel value and its 8 neighbours, the filter produces a zero or one as output values. Binary filters are widely used for morphologic filtering.

The 9 binary pixels examined by a binary filters are put in a special order or bit position as below (where 0 means last position, 1 the one but last position, etc.):

5	6	7
4	8	0
3	2	1

This results in a number of 9 binary digits (when only lower right pixel is true: 000000010).

Thus depending on the position of true pixels a unique number is obtained. To decide whether the central pixel should be assigned a 0 or 1, this number is looked up in a table which is present in each binary filter itself.

Standard binary filters are: DILATE4, DILATE8, SHRINK4, SHRINK8, INBND4, INBND8, OUTBND4, OUTBND8, CONN8TO4, BINMAJOR, PEPPSALT, LIFEGAME.

User-defined filters

You can create and edit your own *linear filters*, for example in the Filter : dialog box (click the create button ), by selecting Create Filter command from the File menu of the Main window, or by double-clicking the NewFilter item in the Operation-list. You can specify the size of the linear filter, insert your own values in the matrix and specify a gain factor. For more information, refer to the Create Filter dialog box and the Edit Filter dialog box. You can also define a linear filter as an expression on the command line of the Main window.

Further, in the Filter : dialog box, you can modify the *Average filter*, *Majority filters*, *Median filters*, the *Pattern filter*, *Rank Order filters*, and the *Standard Deviation filter* to your wishes. You can specify the filter size, a rank or the threshold. For more information, refer to Filter : dialog box. You can also define these filters on the command line of the Main window.

User-defined average filters

The standard average filter is AVG3x3; it calculates the average value of 9 pixel values considered.

Modifying the standard AVG3x3 filter

In the Filter : dialog box, you can adapt the standard AVG3x3 filter according to your wishes.

- Choose for Filter type: Average,
- Then, specify the size of the average filter you want to use.

See also the example below.

Defining an average filter on the command line (advanced)

You can also perform the Filter operation from the command line of the Main window and define an average filter in the expression:

```
OUTMAP = MapFilter(InputMapName,Average(rows , cols))
```

where:

OUTMAP	is the name of your output map.
MapFilter	is the command to start the Filter operation.
InputMapName	is the name of your input map.
Average	is the command to defined an average filter.
rows	are the number of rows of the average filter.
cols	is the size of the average filter in columns.

Example

To obtain the average of each 5 horizontal pixels examined:

- in the Filter : dialog box, choose for Filter type: Average, specify 5 for the number of rows and 1 for the number of columns,
- on the command line, use a filter definition of: Average (5 , 1).

Restrictions

- *row* and *col* values must be odd values, but do not need to be the same.
- (*rows* * *cols*) may not exceed 8000.

User defined linear filters**Creating, editing and storing a linear filter on disk**

User-defined linear filters, can be created through the dialog box of the Filter operation (click the create button ), or through the File menu of the Main window, choose Create Filter, or by double-clicking the NewFilter command in the Operation-list. The Create Filter dialog box and the Edit Filter window will appear. You can specify the size of the filter, fill out values in the matrix, and specify a gain factor.

The standard linear filters cannot be edited.

Defining linear filters on the command line (advanced)

You can also perform the Filter operation from the command line of the Main window and define a linear filter in the expression:

```
OUTMAP = MapFilter(InputMapName, FilterLinear(rows, cols,
expression))
```

where:

OUTMAP	is the name of your output map.
MapFilter	is the command to start the Filter operation.
InputMapName	is the name of your input map.
FilterLinear	is the command to define a linear filter.

<i>rows</i>	is the size of the linear filter in rows.
<i>cols</i>	is the size of the linear in columns.
<i>expression</i>	is an expression in which you can use x, y, and r for the linear filter's matrix values.

Restrictions

- *row* and *col* values must be odd values.
- (*row* * *col*) may not exceed 8000.

To define a user-defined average filter, in the Filter : dialog box, choose for the Filter Type: Average, then specify the size of the filter.

User-defined majority filters

The standard majority filters are: Majority, MajUndef, and MajZero.

Modifying the standard Majority filters

In the Filter : dialog box, you can adapt the standard Majority filters according to your wishes.

- Choose for Filter type: Majority, and clear the Predefined check box,
- Then, specify the size of the majority filter you want to use and whether only undefined input values should be replaced or not.

Defining a majority filter on the command line (advanced)

You can also perform the Filter operation from the command line of the Main window and define a majority filter in the expression:

```
OUTMAP = MapFilter(InputMapName, Majority(rows,cols))
OUTMAP = MapFilter(InputMapName, UndefMajority(rows,cols))
OUTMAP = MapFilter(InputMapName, ZeroMajority(rows,cols))
```

where:

OUTMAP	is the name of your output map.
MapFilter	is the command to start the Filter operation.
InputMapName	is the name of your input map.
Majority	is the command to define a Majority filter.
UndefMajority	is the command to define an Undef-Majority filter.
ZeroMajority	is the command to define a Zero-Majority filter.
<i>rows</i>	are the number of rows in the majority filter.
<i>cols</i>	are the number of columns in the majority filter.

Example

To obtain a majority filter which only considers horizontal neighbours, use a filter definition of: Majority(1, 3)

Restrictions

- *row* and *col* values must be odd values.
- $(row * col)$ may not exceed 8000.

User-defined median filter

The overall statement to invoke filtering from the command line is:

```
OUTMAP = MapFilter(InputMapName,FilterName)
```

Standard median filters in ILWIS are MED3x3 and MED5x5. These median filters sort 9 resp. 25 pixel values and assign the 5th resp. 13th value (median value) to the center pixel in the output map. Median filters are a kind of special rank order filters because the rank is already fixed.

From the command line, filtering with user-sized median filters is allowed:

```
OUTMAP = MapFilter(InputMapName,Median(rows,cols))
```

```
OUTMAP = MapFilter(InputMapName,Median(rows,cols,threshold))
```

where:

OUTMAP	is the name of your output map.
MapFilter	is the command to start filtering.
InputMapName	is the name of your input map.
Median	is the command to invoke a user-defined median filter.
rows	is the size of the median filter in rows.
cols	is the size of the median filter in columns.
threshold	when a threshold is set, the value of the center pixel will only be replaced by the new value if the difference between the original and new pixel value is smaller than or equal to the threshold; else the original pixel value is retained.

Example

To obtain the median value of each 5 horizontal pixels examined, use a filter definition of: `Median(5 , 1)`.

Restrictions

- *row* and *col* values must be odd values.
- $(row * col)$ may not exceed 8000.

User-defined rank order filters

Standard rank order filters are: MED3x3 and MED5x5; they order each 9 or 25 input pixels encountered from low values to high values and the value found at rank 5 (for MED3x3) or at rank 13 (for MED5x5) is assigned to the output pixel.

Modifying the standard rank order filters

In the Filter : dialog box, you can adapt the MED3x3 and the MED5x5 filters according to your wishes.

- Choose for Filter type: Rank Order and clear the Predefined check box.
- Then, specify the size of the rank order filter you want to use, the rank order number and optionally a threshold.

See also the example below.

Defining rank order filters on the command line (advanced)

You can also perform the Filter operation from the command line of the Main window and define a rank order filter in the expression:

```
OUTMAP = MapFilter(InputMapName, Rankorder(rows,cols,rank))
```

```
OUTMAP = MapFilter(InputMapName,  
                    Rankorder(rows,cols,rank,threshold))
```

where:

OUTMAP	is the name of your output map.
MapFilter	is the command to start the Filter operation.
InputMapName	is the name of your input map.
Rankorder	is the command to defined a rank order filter.
rows	are the number of rows of the rank order filter.
cols	are the number of columns of the rank order filter.
rank	is the rank order number that determines the output value.
threshold	when a threshold is set, the value of the center pixel will only be replaced by the new value if the difference between the original and new pixel value is smaller than or equal to the threshold.

Examples

1. To obtain the minimum of each 9 pixels examined:
 - in the Filter : dialog box, select for the Filter type Rank Order and clear the Predefined check box, specify 3 for the number of rows and the number of columns, specify 1 for the rank, and optionally specify a threshold.
 - on the command line, use a filter definition of: Rankorder (3 , 3 , 1)
Rank 1 means the smallest value of each 9 (3 by 3) pixel values examined will be used for the output pixel.

2. To obtain the maximum of each 7 horizontal pixels examined:
 - in the Filter : dialog box, select for the Filter type Rank Order and clear the Predefined check box, specify 7 for the number of rows and 1 for the number of columns, specify 7 for the rank, and optionally specify a threshold.
 - on the command line, use a filter definition of: `Rankorder(7 , 1 , 7)`Rank 7 means that the largest value of each 7 (7 by 1) pixel values examined is used as output value.

Restrictions

- the number of rows and columns must be odd values, but do not need to be the same.
- $(rows * cols)$ may not exceed 8000.
- the minimum rank value is 1;
- the maximum rank value cannot exceed the total size of the filter $(rows * cols)$.

-
- ☞ With neighbourhood operators similar operations can be performed as with rank order filters: for example, the minimum, maximum, sum and predominant pixel value of each 9 pixels considered can be assigned as output value to the central pixel.
-

7.3.4 Stretch

Functionality

The Stretch operation re-distributes values of an input map over a wider or narrower range of values in an output map. Stretching can for instance be used to enhance the contrast in your map when it is displayed. Two stretch methods are available: linear stretching and histogram equalization.

Example linear stretching

In any satellite image, the possible pixel values are in the range of 0 to 255. In practice however, not all these values will occur in your images. Suppose you have a certain image in which the values range from 55 to 103. When this map is stretched linearly to output range 0 to 255: the minimum input value 55 is brought to output value 0, and maximum input value 103 is brought to output value 255, and all other values in between change accordingly (using the same formula). As 0 is by default displayed in black, and 255 in white, the contrast will be better when the image is displayed.

Example histogram equalization

When an image is stretched with histogram equalization, also the number of pixels with a certain value is taken into account. Input values are re-distributed over a user-defined number of output intervals; in principle, all output intervals contain an equal number of pixels. When a certain image with values between 55 and 103 is stretched with histogram equalization to 256 output intervals; the minimum input value 55 is brought to output value 0, and maximum input value 103 is brought to output value 255; the other input values are divided over these 256 output intervals

in such a way that each interval contains an equal number of pixels as far as possible. In general, it can be said that the part of an image's histogram with the largest amount of pixels is relatively stretched more than the parts with few pixels. For more information on both stretch methods, see Stretch : algorithm.

-
- ☞ The stretch operation produces a new output map with stretched values. Temporary linear stretching can be performed during the display of a value map: you can specify 'stretch from' range in the Display Options dialog box.
-

Input map requirements and type of output map

An input map for the Stretch operation must be a value map. The result of the Stretch operation is always another value map.

'Stretch from' and 'stretch to' range

During Stretch, the user has to specify an input value range to stretch from, and an output value range to stretch to. The 'stretch from' range can be specified as two values, which act as the lower and upper boundaries of input values that you want to stretch.

The 'stretch from' range can also be specified as a percentage (e.g. 1%). This percentage stands for the cumulative amount of pixels on each side of the map's histogram that should not be taken into account during stretch. In fact, the 1% range of values on the lower side of a map's histogram is brought to the lower 'stretch to' value and the 1% range of values on the upper side of a map's histogram is brought to the upper 'stretch to' value. Ignoring for instance 1% of the smallest values in the map and 1% of the largest values in a map during stretching can improve the overall stretch results as the ends of a map's histogram usually contain few sparse values.

For linear stretching, the 'stretch to' range is specified by the domain of the output map. When choosing the Image domain, the output stretch range is from 0 to 255. When choosing another value domain, you need to specify the minimum and maximum values to stretch to (value range) and a precision (resolution of output values).

For histogram equalization, the 'stretch to' range is the number of specified intervals; when 256 intervals are specified, the output values will range from 0 to 255. When another number of output intervals is specified, the first interval will obtain output value 1, the second value 2, etc.

Examples

In this example, a TM band is shown with its original histogram (Fig. 1), then the result a linear stretch (Fig. 2) of this image, and finally the result after performing histogram equalization (Fig. 3) on this image. In both cases, stretching was performed on all values in the input map (0%); for linear stretching these values

were stretched to the image domain (range 0-255, precision 1), and for histogram equalization the input values were stretched into 256 output intervals.

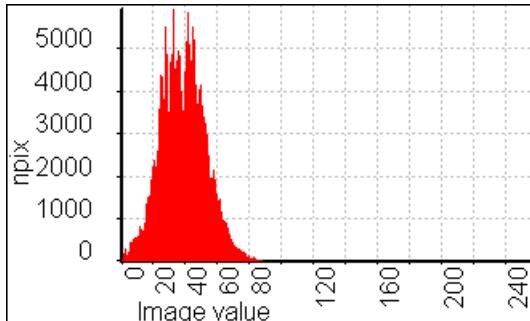


Fig. 1: Original histogram of TMB4.
The values in the original image range from 0 to 103.

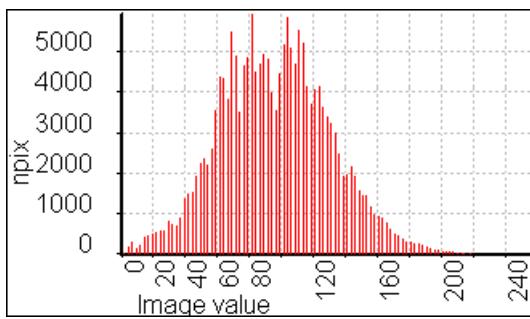


Fig. 2: Result after linear stretching.

After linear stretching, the values of the image range from 0 - 255. There are still a lot of pixels with values in the middle of the image range (histogram has a bell shape).

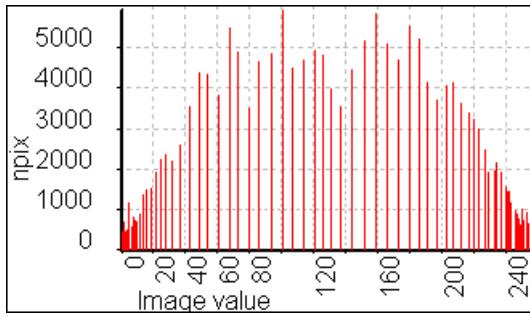


Fig. 3: Result after histogram equalization.

After histogram equalization, the values of the image range from 0 - 255. The peak of values which occurred in the center of the original histogram as well as in the result after linear stretching is smoothed out.

Dialog box

The Stretch operation re-distributes values of an input map over a wider or narrower range of values in an output map. Stretching can for instance be used to enhance the contrast in your map when it is displayed. Two stretch methods are available: linear stretching and histogram equalization.

Dialog box options

Input raster map:

Select an input raster map. Open the list box and select the desired input map, or drag a raster map directly from the Catalog into this box. The input map must be a value map.

Method:

- Linear Stretching
- Histogram Equalization

Intervals:

For histogram equalization: specify the number of output intervals. When 256 intervals are specified, the output values will range from 0 to 255; when another number of output intervals is specified, the first interval will obtain output value 1, the second value 2, etc.

Stretch from:

Min Max:

Percentage:

Specify the range of input values to be stretched. Enter lower and upper boundaries of the range of input values to be stretched, or

Enter a percentage of pixels not to be taken into account by during stretching. When entering for instance 1%, this means that 1% of the smallest values in the map and 1% of the largest values in the map are ignored.

Output raster map:

Type a name for the output raster map that will contain the stretched values.

Domain:

For linear stretching: select an existing value domain for the output map. When working with images, it is suggested to select system domain Image; the values in the input map are then stretched to 0 - 255. When working with other value maps, it is suggested to select the system domain 'value'; then also adjust the value range and precision of the selected domain; this determines the 'stretch to' range for linear stretching.

Value range:

For linear stretching with an output value domain but not domain Image: adjust these values to the values to which you want to stretch to.

Precision:

For linear stretching with an output value domain but not domain Image: adjust the precision if you

	want to change the 'resolution' of possible values in the output map.
Show:	Select this check box if you want the output map to be displayed in a map window when the operation has finished. Clear this check box if you do not want to see this map immediately: you simply define how the output map should be created.
Description:	Optionally, type a description for the output map. The description appears in the title bar when the output map is displayed.

A dependent output map is created. The output map is always a value map. For histogram equalization, the output domain is always a value domain, and the stretch to value range is determined from the number of specified intervals.

Command line

The Stretch operation can be directly executed by typing one of the following expressions on the command line of the Main window:

`OUTMAP = MapStretchLinear(InputMapName, Range from, Domain)`

`OUTMAP = MapStretchLinear(InputMapName, Range from, Domain, Range to)`

`OUTMAP = MapStretchLinear(InputMapName, Range from, Intervals)`

where:

<code>OUTMAP</code>	is the name of your output raster map.
<code>MapStretchLinear</code>	is the command to start the Stretch operation with linear stretching.
<code>MapStretchHistEq</code>	is the command to start the Stretch operation with histogram equalization.
<code>InputMapName</code>	is the name of your input raster map (must be a value map).
<code>Range from</code>	is the range of input values that should be stretched. Specify the 'stretch from range' as: <code>min:max /perc</code>
<code>min:max</code>	lower and upper 'stretch from' boundaries
<code>perc</code>	real value ≥ 0 , percentage of pixels to be ignored on both sides of the input map's histogram during stretching.
<code>Domain</code>	For linear stretching: when choosing output domain Image, the output stretch range is from 0 to 255; when choosing any other output domain Value, specify the minimum and maximum values to stretch to (value range) and a precision (resolution of output values).
<code>Range to</code>	For linear stretching when choosing an output Value domain but not Image: the range of output

<i>min:max</i>	values to which the 'range from' values need to be stretched. Specify the 'stretch to' range as: <i>min:max / min:max:precision</i>
<i>min:max:precision</i>	Lower and upper 'stretch to' values in output map.
Intervals	Lower and upper 'stretch to' values in output map, and precision of output values.

For histogram equalization: the number of output intervals: when 256 intervals are specified, the output values will range from 0 to 255; when another number of output intervals is specified, the first interval will obtain output value 1, the second value 2, etc.

When the definition symbol = is used, a dependent output map is created; when the assignment symbol := is used, the dependency link is immediately broken after the output map has been calculated.

Algorithm

The Stretch operation re-distributes values of an input map over a wider or narrower range of values in an output map. Stretching can for instance be used to enhance the contrast in your map when it is displayed. Two stretch methods are available: linear stretching and histogram equalization. For general information, see Stretch : functionality.

Linear stretching

The input values of a map are re-scaled to output values in the output map. Input values are specified by the 'stretch from' values; the lower and upper 'stretch from' boundary values are included in the stretching. Output values are specified by the output domain and the value range and precision of this domain. Linear stretching uses the following formula:

$$\text{OUTVAL} = (\text{INVAL} - \text{INLO}) * ((\text{OUTUP}-\text{OUTLO})/(\text{INUP}-\text{INLO})) + \text{OUTLO}$$

where:

OUTVAL	Value of pixel in output map
INVAL	Value of pixel in input map
INLO	Lower value of 'stretch from' range
INUP	Upper value of 'stretch from' range
OUTLO	Lower value of 'stretch to' range
OUTUP	Upper value of 'stretch to' range

When the 'stretch from' range is specified as values, these are INLO and INUP. When the 'stretch from' range is specified as a percentage (e.g. 1%), INLO and INUP are determined from the map's histogram (1% and 99% ncpct); all input values smaller than or equal to INLO are brought to OUTLO, and all input values greater than or equal to INUP are brought to OUTUP.

When choosing output domain Image, then OUTLO is 0, and OUTUP is 255.

When choosing domain Value, the minimum and maximum values specified as the value range are used as OUTLO and OUTUP ('stretch to' range).

The value specified as the precision of the output value domain is used to round OUTVAL.

The result of a linear stretch is that all input values are stretched to the same extent, see Fig. 1 below.

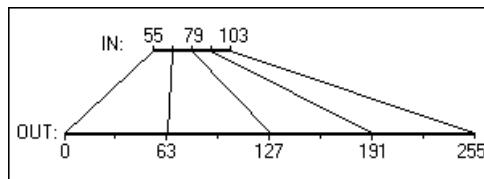


Fig. 1: Schematic explanation of a linear stretch.
Input values range from 55 to 103; output values range from 0 to 255.

Histogram equalization

Lower and upper boundaries of the 'stretch from' range are determined in the same way as with linear stretching. The 'stretch to' values are determined from the number of specified output intervals. When 256 intervals are specified, the output values will range from 0 to 255. When another number of output intervals is specified, the first interval will obtain output value 1, the second value 2, etc.

1. The histogram of the input map is calculated. The total number of pixels with values is calculated (undefined values are not considered).
2. The total number of pixels with values is divided by the number of required output intervals, in order to get a 'threshold' for the number of pixels that ideally should end up in each interval.
3. The 'threshold' divides the input histogram into the desired number of output intervals as follows:
4. For the first interval:
 - when in the histogram of the input map, the cumulative number of pixels reaches the 'threshold' value,
 - the input value in the histogram closest to this 'threshold' is selected, and
 - all input pixel values smaller or equal than this value are assigned the first output value.
5. For next intervals:
 - whenever a next factor of the 'threshold' is reached in the input histogram,
 - the value closest to this 'threshold' is selected, and
 - all input pixel values smaller or equal than this value but larger than the previous interval boundary value are assigned the next output value.

In this way all output intervals are filled with input values.

The result of a histogram equalization stretch is that parts of the histogram with many pixels are stretched more than parts with few pixels, see also Figure 2 below.

Figure 3 below shows step 3 as described above: the determination of which input values belong to which output interval, in the histogram of an input map.

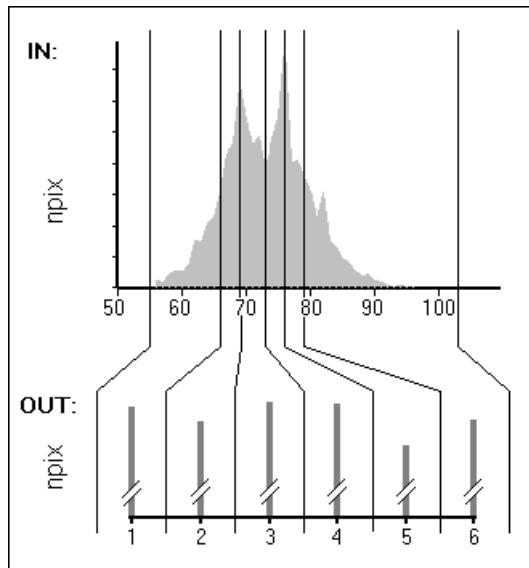


Fig. 2: Schematic explanation of a histogram equalization.
Input values range from 55 - 103. Number of output intervals 6.

value	npix	npixcum
55	61	61
56	543	604
57	445	1049
58	951	2000
59	1137	3137
60	1230	4367
61	1641	6008
62	3088	9096
63	3075	12171
64	4214	16385
65	4635	21020
66	6216	27236
67	8747	35983
68	9360	45343
69	13116	58459
70	10838	69297
71	9271	78568
72	9816	88384
73	7595	95979
74	9673	105652
75	10979	116631
76	15516	132147
77	7911	140058
78	8238	148296
79	7095	155391
80	6288	161679

Fig. 3: Part of an input map's histogram showing the determination of which input values belong to which output intervals. The total number of pixels in this map was 187000. The threshold value for 6 groups, is thus 311667.67 pixels. Every time a factor of this figure is reached in the histogram, a boundary is inserted; the value of the input map is closest to this threshold value is selected (66, 69, 73, 76, 79).

7.3.5 Slicing

Functionality

Slicing classifies the values of a raster map. Ranges of values of the input map are grouped together into one output class. A domain Groupshould be created beforehand; it lists the upper boundaries of the groups and the group names.

With Slicing, a map with for example slope values ranging from 0 to 200% can be grouped into relief classes: Flat (0-2%), Undulating (2-8%), Rolling (8-16%), Hilly (16-30%), Mountainous (>30%) (FAO). Also, a map containing Green Vegetation Index values (combination of satellite bands), can be sliced into user-defined intervals. This can be considered as a first classification.

Example

A map with slope values ranging from 0 to 200% is grouped into relief classes: Flat (0-2%), Undulating (2-8%), Rolling (8-16%), Hilly (16-30%), Mountainous (>30%) (FAO).

Input map:

6	10	20	40
4	12	20	25
3	7	12	10
1	5	5	5

Output map:

U	R	H	M
U	R	H	H
U	U	R	R
F	U	U	U

In the output map, read for F Flat, for U Undulating, for R Rolling, for H Hilly and for M Mountainous.

The Slicing is performed according to the following domain Group:

Upper Boundary	Group Name
2	Flat
8	Undulating
16	Rolling
30	Hilly
200	Mountainous

Input requirements

The input map should be a value map; further, a Group domain should be available.

Domain and georeference of output map

The output map uses the Group domain that is required for the Slicing operation. The output map uses the same georeference as the input map.

- ☞ The Slicing operation creates a new output map with new class names. Instead, you can also slice/classify a value map by creating and editing a representation for that map. You can then interactively modify the representation and redraw the map until satisfied:
1. Double-click a value map in the Catalog; in the Display Options dialog box, click the create button next to the Representation list box to create a representation for this map;
 2. Type a name for the new representation; then the Representation Value/Gradual editor is opened;
 3. Click OK in the Display Options dialog box of the map; the map is displayed in a map window;
 4. In the Representation Value/Gradual editor, insert boundaries and adjust colors;
 5. Keep the editor open and switch to the map window;
 6. Press the Redraw button  in the map window and the map is directly shown with this representation;

-
7. Switch to the Representation editor and continue to change boundaries, and colors; then switch to the map window to redraw the map, etc. until satisfied.

Slicing by modifying a representation takes much less time than (re-)calculating values as the Slicing operation would do; another advantage is that the exact input values are retained. For more information, see [How to slice an image](#).

☞ To get the best out of the Slicing operation:

- When creating a Group domain, also fill the representation of this domain.
 - Keep the domain window, representation window and output map window open simultaneously; you can modify classes, i.e. edit the domain Group, and change its representation;
 - Open the Properties of the map (Edit menu in the map window, Properties) and click the Make Up-to-Date button; the map is recalculated according to the new boundaries in the domain Group;
 - Redraw the map in the map window.
- ☞ For ILWIS 1.41 users: a domain Group is a synonym for an ILWIS 1.41 classify table.
- ☞ In ILWIS 2.1, both the Slicing operation as the MapCalc function CLFY(InputMap,DomainGroup) can be used to apply the classify table on a map.
- ☞ To perform a multi-spectral image classification, see the Classify operation.
-

Dialog box

Slicing classifies the values of a raster map. Ranges of values of the input map are grouped together into one output class. A domain Group should be created beforehand; it lists the upper boundaries of the groups and the group names.

With Slicing, a map with for example slope values ranging from 0 to 45% can be grouped into relief classes: Flat (0-2%), Undulating (2-8%), Rolling (8-16%), Hilly (16-30%), Mountainous (>30%) (FAO).

Dialog box options

Input raster map:	Select an input raster map. Open the list box and select the desired input map, or drag a raster map directly from the Catalog into this box. The input map should be a value map.
Output raster map:	Type a name for the output raster map that will contain the classified values as groups.
Domain:	Select the name of a domain Group which was created beforehand (click the list box), or create a new domain Group (click the little create button).
Show:	Select this check box if you want the output map to be displayed in a map window when the operation has finished. Clear this check box if you do not want to see this map immediately: you simply define how the output map should be created.

Description: Optionally, type a description for the output map. The description appears in the title bar when the output map is displayed.

A dependent output map is created.

To interactively slice a map, create and edit a representation Value/Gradual.

To perform a multi-spectral image classification, see the Classify operation.

Command line

The Slicing operation can be directly executed by typing one of the following expressions on the command line of the Main window:

`OUTMAP = MapSlicing(InputMapName, Domain)`

`OUTMAP = CLFY(InputMapName, Domain)`

where:

<code>OUTMAP</code>	is the name of your output raster map.
<code>MapSlicing</code>	is the command to start the Slicing operation.
<code>CLFY</code>	is a MapCalc function (this is in fact a short cut).
<code>InputMapName</code>	is the name of your input value map (raster map).
<code>Domain</code>	is the name of your domain Group which contains the boundaries of the values and the group names for the output map.

When the definition symbol `=` is used, a dependent output map is created; when the assignment symbol `:=` is used, the dependency link is immediately broken after the output map has been calculated.

Algorithm

Slicing classifies the values of a raster map. Ranges of values of the input map are grouped together into one output class. A domain Group should be created beforehand; it lists the upper boundaries of the groups and the group names. With Slicing, a map with for example slope values ranging from 0 to 200% can be grouped into relief classes: Flat (0-2%), Undulating (2-8%), Rolling (8-16%), Hilly (16-30%), Mountainous (>30%) (FAO).

7.3.6 Color separation

Functionality

The Color separation operation allows you to extract different 'bands' for instance from a scanned or digital color photo as if using color filters when taking the picture. After color extraction, you can perform the normal Image Processing operations like Filtering, Classification, etc. on these bands.

Maps that have a Picture domain or the (24 bit) Color domain store for each pixel three values: Red, Green and Blue. The Color separation operation allows you to retrieve for each pixel either the Red, Green or Blue value and store these in a separate map. You can also retrieve Yellow, Magenta, Cyan, combined Gray values, or Hue, Saturation or Intensity values for each pixel.

Color models

Display devices commonly use the Red, Green, Blue (RGB) additive color system. Printers commonly use the Yellow, Magenta, Cyan, Black (YMCK) subtractive color system. Professionals working with colors generally think in Hue, Saturation and Intensity (HSI).

RGB color system

Any color can be regarded as a combination of different contributions of the primary colors Red, Green and Blue (RGB). Figure 1 below presents a color cube with Red, Green and Blue axes. Along each of these axes the 'amount' of a primary color increases from 0 to 255, where 0 means no intensity (no contribution to the final color) and 255 means maximum intensity. Each point in the color cube thus represents a different color depending on the amounts of Red, Green and Blue.

The Gray scale is presented by a line from Black to White (dashed line in Figure 1). For each Gray color, the amounts of Red, Green, and Blue are equal.

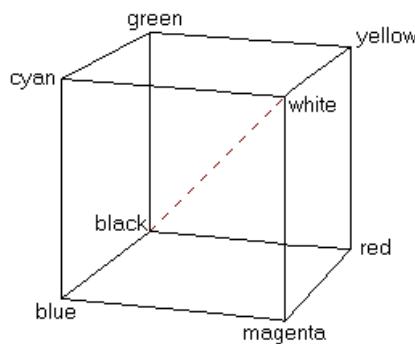


Fig 1: Color cube of Red, Green, Blue and Yellow, Magenta, Cyan and Gray.

YMC color system

The colors Yellow, Magenta, and Cyan (YMC) can be presented in the same color cube as the Red, Green, Blue color cube (Figure 1). The YMC color system uses the Yellow, Magenta, and Cyan subtractive primaries.

$$\begin{array}{lll} \text{Yellow} & = \text{Red} + \text{Green} & = \text{White} - \text{Blue} \\ \text{Magenta} & = \text{Red} + \text{Blue} & = \text{White} - \text{Green} \\ \text{Cyan} & = \text{Green} + \text{Blue} & = \text{White} - \text{Red} \end{array}$$

HSI color system

Colors can also be defined using the Hue, Saturation, and Intensity (HSI) system (see Figure 2).

- Hue is the basic component of a color and is the dominant or average wavelength.
- Saturation describes the 'color purity' or the purity of color relative to Gray. The relative amount of Red, Green and Blue determines the Hue and the Saturation.
- Intensity describes the brightness of a color. The total amount of Red+Green+Blue determines the brightness.
-

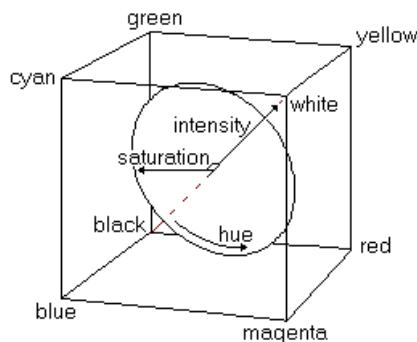


Fig 2: Color cube which also shows Hue, Saturation and Intensity.

The Color separation operation gives you the possibility to change to from one color system to one of the other models. The formulas used for the transformation from one model to another are described in Color separation : algorithm.

Input map requirements

An input raster map with a Picture domain or the Color domain is required. In a map with a picture domain the colors are stored in its representation. In a map with domain Color the colors are stored in the pixels of the map (map is stored using 4 bytes per pixel).

Domain and georeference of output map

The output raster map always obtains the Image domain. When extracting Red, Green, Blue, Yellow, Magenta, Cyan or Gray, the values in the output map range from 0 to 255. When extracting Hue, Saturation or Intensity, the values in the output map range from 0 to 240. This complies with the Windows definition. The output raster map uses the same georeference as the input raster map.

-
- ☞ When you for example extract the Red color from an image, the intensities in the output map are measured for this color. When you open the output image, the intensities for Red are given on a gray scale (and not in a red scale). You can select the Red representation (from black to red) in the Display Options dialog box of the map, or you can create your own representation.
-

Dialog box

The Color separation operation allows you to extract different 'bands' for instance from a scanned or digital color photo as if using color filters when taking the picture. After color extraction, you can perform the normal Image Processing operations like Filtering, Classification, etc. on these bands.

Maps that have a Picture domain or the (24 bit) Color domain store for each pixel three values: Red, Green and Blue. The Color separation operation allows you to retrieve for each pixel either the Red, Green or Blue value and store these in a separate map. You can also retrieve Yellow, Magenta, Cyan, combined Gray values, or Hue, Saturation or Intensity values for each pixel.

Dialog box options

Input raster map:	Select an input raster map. Open the list box and select the desired input map, or drag a raster map directly from the Catalog into this box. Only input maps with a picture domain or the color domain are allowed.
Color:	Select the color you want to retrieve: Red, Green, Blue, Yellow, Magenta, Cyan, Hue, Saturation, Intensity or Gray.
Output raster map:	Type a name for the output raster map that will contain the values of one color of the input map.
Show:	Select this check box if you want the output map to be displayed in a map window when the operation has finished. Clear this check box if you do not want to see this map immediately: you simply define how the output map should be created.
Description:	Optionally, type a description for the output map. The description appears in the title bar when the output map is displayed.

A dependent output map is created.

Command line

Color separation can be directly executed by typing one of the following expressions on the command line of the Main window:

```
OUTMAP = MapColorSeparation(InputMapName, Color)
```

```
OUTMAP = MapColorSep(InputMapName, Color)
```

```
OUTMAP = InputMapName.Color
```

where:

OUTMAP is the name of your output raster map.

MapColorSeparation is the command to start the Color separation operation.

MapColorSep is an alias to start the Color separation operation.

InputMapName is the name of your input raster map; the input maps must have a Picture domain or the Color domain.

Color is the color you want to extract, choose one of the following: red| green| blue| yellow| magenta| cyan| hue| saturation| sat| intensity| intens| gray| grey

When the definition symbol = is used, a dependent output map is created; when the assignment symbol := is used, the dependency link is immediately broken after the output map has been calculated.

Algorithm

The Color separation operation allows you to extract different 'bands' for instance from a scanned or digital color photo as if using color filters when taking the picture. After color extraction, you can perform the normal Image Processing operations like Filtering, Classification, etc. on these bands.

Maps that have a Picture domain or the (24 bit) Color domain store for each pixel three values: Red, Green and Blue. The Color separation operation allows you to retrieve for each pixel either the Red, Green or Blue value and store these in a separate map. You can also retrieve Yellow, Magenta, Cyan, combined Gray values, or Hue, Saturation or Intensity values for each pixel.

For each pixel the selected color is retrieved. Red, Green and Blue values are directly retrieved from the map or the picture domain. When another color is selected, one of the following formulas is applied:

```
yellow   =  255 - blue
magenta =  255 - green
cyan     =  255 - red
hue      =  255/PI * atan2(sqrt(3)/2*(green-blue), red-
(green+blue)/2) * 240/255
```

```

sat      =  sqrt(red*red + green*green + blue*blue - red*green -
                red*blue - green*blue) * 240/255
intens   =  (red+green+blue)/3 *240/255
gray     =  0,3*red + 0,59*green + 0,11*blue

```

In the output maps, the values for Red, Green, Blue, Yellow, Magenta, Cyan and Gray values range from 0 to 255. The values for Hue, Saturation and Intensity range from 0 to 240. This range complies with the Windows definition.

7.3.7 Cluster

Functionality

Clustering, or unsupervised classification, is a rather quick process in which image data is grouped into spectral clusters based on the statistical properties of all pixel values. It is an automated classification approach with a maximum of 4 input bands.

In the first phase of the operation, a multidimensional histogram of the input bands is calculated. The multidimensional histogram is a representation of the feature space. In the second phase, this feature space is split into several boxes to obtain the desired number of clusters.

After classification the desired number of clusters are available. In the output raster map, each pixel has a class name like Cluster 1, Cluster 2, ..., etc. These clusters are called spectral classes.

The program cannot automatically give you the meaning of the obtained spectral classes; it is up to the user to find out which land cover type or other feature corresponds to each cluster, and finding these relationships may not always be easy. You can compare the clusters in the output raster map to field observation data, or perform a Cross with a ground truth map. Of course, if you know the area, you can also visually interpret the output.

The difference between unsupervised classification (Cluster) and supervised classification (Sample and Classify) is that when using Cluster, the software automatically splits the feature space into clusters based on the spectral values and afterwards you have to identify these clusters. When using Sample and Classify, classification takes place according to training pixels in a sample set which is defined by the user.

Input map requirements

The input raster map list must contain raster maps that have the Image domain and the same georeference.

Domain of output map

The Cluster operation creates a new Class domain for the output map in which the class names are Cluster 1, Cluster 2, etc. This output domain is always stored by the output map (internal domain).

Georeference of output map

The output raster map uses the same georeference as the input raster maps.

- ☞ To view and edit the internal output domain of the Cluster operation, you can display the output map in a map window and choose the Domain command from the Edit menu in the map window.
- ☞ To change the colors of the clusters in the output map, you can display the map in a map window and choose the Representation command from the Edit menu in the map window. Press the Redraw button  in the map window to see the effect of changed colors.
- ☞ To view or edit the internal domain or the representation, you can also click the output map with the right mouse button in the Catalog, then choose Open as Domain or Open as Representation from the context-sensitive menu.
- ☞ For more information on internal domains and representations, refer to How to open internal domains/representations.

Dialog box

Clustering, or unsupervised classification, is a rather quick process in which image data is grouped into spectral clusters based on the statistical properties of all pixel values. It is an automated classification approach with a maximum of 4 input bands.

Dialog box options

- Number of input maps: Select the number of input bands to be used in the cluster analysis (1, 2, 3, or 4). For the selected number of input bands, a number of Input Map list boxes appear.
- Input maps: Select the input raster maps. Open each list box and select the desired input map, or drag raster maps directly from the Catalog into the list boxes. All input maps should use domain Image.
- Number of clusters: Type a value for the number of clusters that should be created. You can create between 2 and 60 clusters.
- Output raster map: Type a name for the output raster map that will contain the desired number of clusters after the unsupervised classification.
- Show: Select this check box if you want the output map to be displayed in a map window when the operation has finished. Clear this check box if you do not want to see this map immediately: you simply define how the output map should be created.
- Description: Optionally, type a description for the output map. The description appears in the title bar when the output map is displayed.

A dependent output map is created. Furthermore, cluster names for the output map are stored by the output map (internal class domain).

Command line

The Cluster operation can directly be executed by typing one of the following expressions on the command line of the Main Window:

OUTMAP = MapCluster(InputMapList, nr of clusters)

OUTMAP = MapCluster(mlist(map1, map2, ...), nr of clusters)

where:

OUTMAP	is the name of the output map.
MapCluster	is the command to start the Cluster operation.
InputMapList	is the name of an existing maplist which contains 1 up to 4 raster maps with the Image domain.
mlist (map1, map2, ...)	is a method to define a maplist and the raster maps that it should contain.
nr of clusters	is an integer value for the number of clusters/spectral classes you want to obtain in the output map; number of clusters between 2 and 60.

When the definition symbol = is used, a dependent output map is created; when the assignment symbol := is used, the dependency link is immediately broken after the output map has been calculated. Further, cluster names for the output map are stored by the output map (internal class domain).

Algorithm

Clustering, or unsupervised classification, is a rather quick process in which image data is grouped into spectral clusters based on the statistical properties of all pixel values. It is an automated classification approach with a maximum of 4 input bands.

To create the number of desired clusters a generalized form of the Heckbert quantization algorithm is used. For more information, see also Color composite: algorithm.

In the first phase of the application a multidimensional histogram of the input bands is calculated.

- When only 1 or 2 input bands are used, all values of the band(s) are taken into account;
- When 3 input bands are used, the pixel values 0-255 are linearly stretched to 0-31 using the 1% interval of the histograms;
- When 4 input bands are used, the pixel values 0-255 are linearly stretched to 0-15 using the 1% interval of the histograms.

The multidimensional histogram is regarded as a representation of the feature space.

In the second phase this feature space is split or divided into several boxes to obtain the desired number of clusters. In this phase the software starts with one cluster occupying the entire feature space. This cluster is divided into two along its longest

axis (largest variation in pixel values). The division is done in such a way that the two new clusters approximately contain the same amount of pixels. Then, the next cluster is divided into two along the longest axis (largest variation). This process continues until the required number of clusters is reached.

Output domain

The Cluster operation creates a new Class domain for the output map in which the class names are Cluster 1, Cluster 2, etc. This output domain is always stored by the output map (internal domain).

7.3.8 Sample

Sampling is an interactive process of selecting training pixels in a sample set prior to an image classification. To start sampling, create a new sample set or select an existing one in the Sampling dialog box.

Then the sample set editor is opened: select pixels that are characteristic for a certain type of a certain natural resource on the ground and that have similar spectral values in the maps in the map list, and assign a class name to them. The spectral values of these sampled pixels or training pixels provide the basis on which decisions are made during classification. These values can be inspected in the sample statistics of a certain class of training pixels and can be visualized in feature spaces. The result of Sampling is a filled sample set.

7.3.9 Classify

Functionality

The Classify operation performs a multi-spectral image classification according to training pixels in a sample set. Before classification, a sample set thus has to be prepared with Sample.

The following classification methods are available:

- Box classifier, using a multiplication factor
- Minimum Distance, using a threshold distance
- Minimum Mahalanobis distance, using a threshold distance
- Maximum Likelihood, using a threshold distance

As Sample was the training phase, where classes of pixels with similar spectral values are defined, Classify is the decision phase, where each output pixel is assigned a class name if the spectral values of that pixel are similar enough to a training class, or if not the undefined value is assigned. The decision whether pixels are similar enough to a class of training pixels depends on the classification method that you choose, and the parameters you use for that method. Of course, the classification depends in the first place on the spectral values of the pixels that you selected to serve as training pixels in Sample. As Classify uses the training pixels selected by the user, Classify is a supervised classification.

-
- ☞ During Sampling, you can already create some dependent output maps with the Classify operation (various methods, various parameters) using the current sample set as input. When creating the dependent output maps, do not mark the Show check box in dialog boxes; just create the definition for the output maps. In this way the dependent output maps exist only by their object definition file and are not yet calculated and stored on disk. If you then add these dependent maps to the pixel info window, you can already see the results of classifying during sampling.
-

The bands that you want to classify should be combined in a map list. This map list is part of the required input sample set. There is no limitation for the number of bands that can be classified. Training pixels should be assigned a class name; this is done in Sample. There is no limitation for the number of classes that a sample set can contain.

General introduction to classification methods

All classification methods calculate the means per band for each class of training pixels as defined in the sample set. The set of means per class is called a class mean below.

The Box classifier is based on the distances towards class means and the standard deviation per band of each class. The Minimum Distance classifier is based on the Euclidian distances towards class means only. Both the Minimum Mahalanobis distance classifier and the Maximum Likelihood classifier are based on the distances towards class means and the variance-covariance matrix of each class.

For the box classifier a multiplication factor has to be specified. This factor is multiplied with the standard deviations of the classes to make the boxes around the classes a bit wider. The larger you choose the multiplication factor, the easier a pixel will be assigned to a class.

For the other classifiers a threshold distance has to be specified. The threshold value has to be specified in terms of the spectral values to be classified, i.e. when classifying a satellite image, the threshold value must be a value between 0 and 255. The larger you choose the threshold, the easier a pixel will be assigned to a class.

Box classifier

Multi-dimensional boxes are drawn around class means based on the standard deviation of each class. The user can insert a multiplication factor (usually > 1) to make all boxes a bit wider. If the spectral values of a pixel to be classified fall inside a box, then that class name is assigned. If the spectral values of a pixel fall within two boxes, the class name of the box with the standard deviation is assigned. If the spectral values of a pixel do not fall within a box, the undefined value is assigned. The default multiplication factor is $\times 3$ for 3 bands.

Minimum Distance to Mean

For the spectral values of a pixel to be classified, the distances towards the class means are calculated. If the shortest (Euclidian) distance to a class mean is smaller

than the user-defined threshold, then this class name is assigned to the output pixel. Else the undefined value is assigned.

Minimum Mahalanobis Distance classifier

For the spectral values of a pixel to be classified, the distances towards the class means are calculated as Mahalanobis distance. The Mahalanobis distance depends on the distances towards class means and the variance-covariance matrix of each class. The class name with the shortest Mahalanobis distance is assigned, if this distance is smaller than the user-defined threshold value. Else, the undefined value is assigned.

Maximum Likelihood classifier

The Maximum Likelihood classification assumes that spectral values of training pixels are statistically distributed according to a 'multi-variate normal probability density function'.

For each set of spectral input values, the distance is calculated towards each of the classes is calculated using Mahalanobis distance. Another factor is added to compensate for within class variability. The class name with the shortest distance is assigned, if this distance is smaller than the user-defined threshold value. Else, the undefined value is assigned.

Input requirements

A sample set is required.

Domain and georeference of output map

The output raster map uses the same domain as the sample set. The output map uses the same georeference as the input maps of the sample set.

-
- ☞ Classification relies on the spectral separability of classes. Heterogeneous classes that contain elements of many other classes, for example towns (mixture of roofs, roads, forest, grass, water etc.), are in general difficult to classify. If all town pixels are to be classified as town, elsewhere forest, grass and water pixels are due to be misclassified.
 - ☞ The spectral values of pixels at the border of 2 distinct classes are usually influenced by both classes. In a feature space, these pixels are usually positioned in between the 2 classes. During classification such pixels may obtain the undefined value as they are not similar enough to either class.
 - ☞ The spectral values of elongated terrain objects, such as roads, having a width smaller than the image's pixel size, are usually influenced by the spectral values of surrounding pixels. Elongated features may therefore be difficult to classify.
- Classification is mainly suitable for classes that form areas in the terrain.
-

Dialog box

The classify operation performs a multi-spectral image classification according to training pixels in a sample set. The following classification methods can be used: Box classifier, Minimum distance, Minimum Mahalanobis distance, Maximum Likelihood.

Dialog box options

Sample set:	Select an input sample set. Open the list box and select the desired sample set, or drag a sample set directly from the Catalog into this box.
Classification method:	Select a classification method: Box Classifier, Minimum Distance, Minimum Mahalanobis Distance or Maximum Likelihood. For more information, see Classify : functionality.
Multiplication factor:	For the Box Classifier method, enter a multiplication factor, usually larger than 1. The multiplication factor allows you to make boxes drawn around class means on the basis of their standard deviation a bit wider.
Threshold distance:	For the other classification methods, enter a threshold distance in terms of spectral input values.
Output raster map:	Type a name for the output raster map that will contain the outcome of the classification.
Show:	Select this check box if you want the output map to be displayed in a map window when the operation has finished. Clear this check box if you do not want to see this map immediately: you simply define how the output map should be created.
Description:	Optionally, type a description for the output map. The description appears in the title bar when the output map is displayed.

A dependent output map is created.

Command line

The Classify operation can be directly executed by typing one of the following expressions on the command line of the Main Window:

```
OUTMAP = MapClassify(SampleSetName, ClassifierBox(factor))
OUTMAP = MapClassify(SampleSetName,
                      ClassifierMinDist(threshold))
OUTMAP = MapClassify(SampleSetName,
                      ClassifierMinMahaDist(threshold))
OUTMAP = MapClassify(SampleSetName, ClassifierMaxLikelihood
                      (threshold))
```

where:

OUTMAP	is the name of your output raster map.
MapClassify	is the command to start the Classify operation.
SampleSetName	is the name of your input sample map which contains the training pixels; a sample set is created through sampling.

<code>ClassifierBox</code>	is the option for Box classification.
<code>ClassifierMinDist</code>	is the option for Minimum Distance classification.
<code>ClassifierMinMahaDist</code>	is the option for Minimum Mahalanobis Distance classification.
<code>ClassifierMaxLikelihood</code>	is the option for Maximum Likelihood classification.
<i>factor</i>	the multiplication factor is a parameter for the Box classifier which allows you to make boxes drawn around class means on the basis of their standard deviation a bit wider; real value > 0 .
<i>threshold</i>	is the threshold distance used in the other classification methods to decide whether calculated distances are small enough; real value > 0 .

When the definition symbol = is used, a dependent output map is created; when the assignment symbol := is used, the dependency link is immediately broken after the output map has been calculated.

Algorithm

The Classify operation performs a multi-spectral image classification according to training pixels in a sample set.

The following classification methods are available:

- Box classifier, using a multiplication factor
- Minimum distance, using a threshold distance
- Minimum Mahalanobis distance, using a threshold distance
- Maximum Likelihood, using a threshold distance

Below, the expression class mean is used for the means of training pixels in a class. The expression feature vector is used for the spectral values of a pixel to be classified.

The set of spectral values referring to the same area on the ground of the bands that you want to classify is called a feature vector below. If you have want to classify an image of 7 bands, each feature vector contains 7 values.

Box classifier

For each class, a multi-dimensional box is drawn around the class mean.

For each class, the size of the box is calculated as:

class mean \pm standard deviation per band * multiplication factor

- if a feature vector falls inside a box, then the corresponding class name is assigned.
- if a feature vector falls within two boxes, the class name of the box with the smallest product of standard deviations is assigned.
- if a feature vector does not fall within a box, the undefined value is assigned.

Minimum Distance to Mean

For each feature vector, the distances towards class means are calculated.

- the shortest Euclidian distance to a class mean is found;
- if this shortest distance to a class mean is smaller than the user-defined threshold, then this class name is assigned to the output pixel.
- else the undefined value is assigned.

Minimum Mahalanobis distance

For each feature vector, the distances towards class means are calculated. For each class, the variance-covariance matrix is calculated.

The minimum Mahalanobis distance is calculated as:

$$D_i(x) = y^T V_i^{-1} y$$

For an explanation of the parameters, see Maximum Likelihood classifier.

- the shortest distance to a class mean is found;
- if this shortest distance to a class mean is smaller than the user-defined threshold, then this class name is assigned to the output pixel.
- else the undefined value is assigned.

Maximum Likelihood

For each feature vector, the distances towards class means are calculated. For each class, the variance-covariance matrix is calculated.

The formula used in Maximum Likelihood reads:

$$D_i(x) = \ln |V_i| + y^T V_i^{-1} y$$

where:

D_i	distance between feature vector x and a class mean based on probabilities
M	number of bands
V_i	the $M \times M$ variance-covariance matrix of a class
$ V_i $	the determinant of V_i
V_i^{-1}	the inverse of V_i
y	$x - m_i$ is the distance towards a class mean
y^T	the transposed of y

- the shortest distance D to a class mean is found;
- if this shortest distance to a class mean is smaller than the user-defined threshold, then this class name is assigned to the output pixel.
- else the undefined value is assigned.

7.3.10 Resample

Functionality

The Resample operation resamples the values of a raster map to another georeference. The coordinate of each output pixel is used to calculate a new value from close-by pixel values in the input map. Three resampling methods are available: nearest neighbour, bilinear, and bicubic.

The Resample operation needs to be used when you have raster maps of different sources or images of different dates and you want to perform raster operations to combine these maps or images (e.g. MapCalc, Cross). First create a georeference tiepoints for each set of maps/images, then use the Resample operation and resample the maps/images preferably to a georeference corners.

When you want to combine rasterized vector maps with satellite data, you can rasterize the vector data on the georeference tiepoints of the satellite images. In case you prefer North-oriented raster maps, rasterize the vector maps with a georeference corners, and Resample the images with the georeference tiepoints to this georeference corners.

Resampling methods

To resample an image, select an input image with a georeference (usually a georeference tiepoints), select a resampling method (nearest neighbour, bilinear, bicubic), type an output map name and select the output georeference (usually a georeference corners).

- When using nearest neighbour resampling, the value of the input pixel closest to a new output pixel is used as output value (see Fig. 1);
- when using bilinear resampling, the values of the 4 input pixels closest to a new output pixel are used to interpolate output values (see Fig. 2);
- when using bicubic resampling, the values of the 16 input pixels closest to a new output pixel are used to interpolate output values.

For more information, see also Resample : Algorithm.

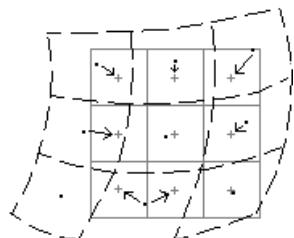


Fig 1: Nearest neighbour resampling. Input pixels are represented by dashed black lines, coordinates of input pixels by black dots; output pixels are represented by gray solid lines, coordinates of output pixels by gray plus signs. The arrows indicate how output values are determined.

When resampling using nearest neighbour (Figure 1), some values of the input map may be used twice in the output map, while other input values may not be used at all.



Fig 2: Bilinear resampling. Input pixels are represented by dashed black lines, coordinates of input pixels by black dots; output pixels are represented by solid gray, coordinates of output pixels by gray plus signs. The arrows indicate how output values are determined.

When resampling using a bilinear interpolation, the values of four input pixels determine the value of one output pixel.

Input map requirements:

The input map needs to have a georeference, usually a georeference tiepoints. To create a georeference tiepoints, refer to Tiepoint Editor, and How to add coordinates to a satellite image.

For nearest neighbour resampling, the input map can have any domain.

For bilinear and bicubic resampling, the input map needs to be a value map.

Domain and georeference of output map

The output map uses the same domain as the input map. If the input map has a value domain, the value range and precision can be adjusted for the output domain. The georeference for the output map has to be selected or created. You can usually select an existing georeference corners.

Dialog box

The Resample operation resamples the values of a raster map to another georeference. The coordinate of each output pixel is used to calculate a new value from close-by pixel values in the input map. Three resampling methods are available: nearest neighbour, bilinear, and bicubic.

The Resample operation needs to be used when you have raster maps of different sources or images of different dates and you want to perform raster operations to combine these maps or images (e.g. MapCalc, Cross). First create a georeference tiepoints for each set of maps/images, then use the Resample operation and resample the maps/images preferably to a georeference corners.

To resample an image:

- select an input image with a georeference (usually a georeference tiepoints),
- select a resampling method: nearest neighbour, bilinear, or bicubic,
- type an output map name and
- select the output georeference (usually an existing georeference corners).

Dialog box options

Input raster map:	Select an input raster map. Open the list box and select the desired input map, or drag a raster map directly from the Catalog into this box.
Resampling method:	Select a resampling method: Nearest Neighbour, Bilinear, or Bicubic. For more information, see Resample : functionality.
Output raster map:	Type a name for the output raster map that will contain resampled pixels.
Georeference:	Select an existing georeference for the output raster map to which the input map should be resampled; open the list box by clicking it. Or create a new georeference by clicking the little create button.
Value range:	In case the output map uses a value domain: accept the default value range, or specify your own range of possible values in the output map.
Precision:	Accept the default precision of output values, or specify your own precision.
Show:	Select this check box if you want the output map to be displayed in a map window when the operation has finished. Clear this check box if you do not want to see this map immediately: you simply define how the output map should be created.
Description:	Optionally, type a description for the output map. The description appears in the title bar when the output map is displayed.

A dependent output map is created.

Command line

The Resample operation can be directly executed by typing the following expression on the command line of the Main window:

```
OUTMAP = MapResample(InputMapName, Georeference,
                      NearestNeighbour | BiLinear | BiCubic)
```

where:

OUTMAP	is the name of your output raster map.
MapResample	is the command to start the Resample operation.
InputMapName	is the name of your input raster map.
Georeference	is the name of an existing georeference that should be used for the output raster map.
NearestNeighbour	is the parameter for Nearest Neighbour resampling.
BiLinear	is the parameter for BiLinear resampling.
BiCubic	is the parameter for BiCubic resampling.

When the definition symbol = is used, a dependent output map is created; when the assignment symbol := is used, the dependency link is immediately broken after the output map has been calculated.

Algorithm

The Resample operation resamples the values of a raster map to another georeference. The coordinate of each output pixel is used to calculate a new value from close-by pixel values in the input map. Three resampling methods are available: nearest neighbour, bilinear, and bicubic.

The resampling process consists of several steps:

- the selected output georeference determines the number of rows and columns in the output map; thus the XY-coordinate for each output pixel is known;
- then these positions are looked up in the original map and, according to the selected interpolation method, 1 (nearest neighbour), 4 (bilinear) or 16 (bicubic) neighbour pixels around this position in the input map are used to calculate a value for the output map.

Nearest neighbour resampling is the fastest method, but results in discontinuities because some input values may be used more than once as output value, while other input values may not be used at all. Bilinear resampling takes much less time than a bicubic resampling. A bilinear interpolation results in discontinuity of the first derivative. A bicubic interpolation remains continuous up to the second derivative.

Nearest neighbour resampling

With nearest neighbour resampling, first the coordinate of each pixel in the output map is determined. Then, for each output pixel, the pixel value of input pixel closest to this coordinate is used as output value.

Figure 1 below shows the position of a 'new' pixel in the output map, and the position and values of 4 surrounding pixels in the input map.

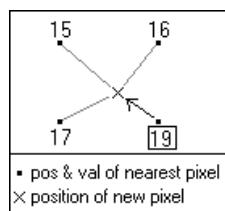


Fig 1: Nearest Neighbour resampling. The coordinates of input pixels are represented by black dots. The coordinate of an output pixel is represented by a cross. The arrow shows how the value of the nearest input pixel is assigned to the output pixel.

The value for the 'new' pixel in the output map is the value in the input map closest to the new coordinate (19).

Bilinear resampling:

With bilinear resampling, first the coordinate of each pixel in the output map is determined. Then the values of 4 surrounding pixels of the input map are used to calculate an interpolated value for each pixel in the output map.

Figure 2 below shows the position of a 'new' pixel in the output map, and the position and values of 4 surrounding pixels in the input map.

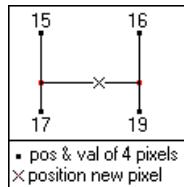


Figure 2: Bilinear interpolation in Bilinear Resampling. The coordinates of 4 neighbouring input pixels are represented by black dots. The coordinate of an output pixel is represented by a black cross. The straight black lines represent the interpolations; intermediate answers are represented by small gray dots.

The value of the 'new' pixel in the output map is calculated by:

- first 2 interpolations in y-direction (between values 15 and 17, and between values 16 and 19; resulting in two intermediate values which are represented in red),
- then 1 interpolation in x-direction (between the two intermediate values in red).

A straight line is drawn through each set of 2 points, and from this the value of the third point is known. A bilinear interpolation should not be used when you intend to calculate a derivative of the output map.

Bicubic resampling

With bicubic resampling, first the coordinate of each pixel in the output map is determined; then the values of 16 surrounding pixels of the input map are used to calculate an interpolated value for each pixel in the output map.

Figure 3 below shows the position of a 'new' pixel in the output map, and the position and values of 16 surrounding pixels in the input map.

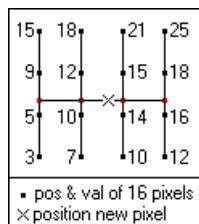


Fig 3: Bicubic interpolation in Bicubic resampling. The coordinates of 16 neighbouring input pixels are represented by black dots. The coordinate of an output pixel is represented by a black cross. The straight black lines represent the interpolations; intermediate answers are represented by small gray dots.

The value of the 'new' pixel in the output map is calculated by:

- first 4 interpolations in y-direction,
- then 1 interpolation in x-direction (between the 4 intermediate values, represented in red).

A third order polynomial is fitted through each set of 4 known points and from this the value of the fifth point is known. A bicubic interpolation gives a better estimate of the output value than a bilinear interpolation.