Appendix A

New capabilities and functions of ILWIS for Windows

Windows user-interface

- Information is shown in windows; windows can be sized, minimized, maximized, etc. For more information, refer to Windows features : Introduction. Each window has a menu bar. When you select a menu command, usually a dialog box appears which allows you to answer questions and specify necessary parameters, see also Windows features : Using a dialog box.
- Most windows have a button bar that provides short cuts for often performed actions.
- Most windows have a context-sensitive menu which lists most relevant menu commands. A context-sensitive menu appears by clicking the right mouse button.
- The statusline at the bottom of a window gives brief information on selected menu commands or on elements within a window that you point at with the mouse pointer or that are selected.
- A more intuitive approach is provided by double-click and drag and drop facilities.

Windows general

- The Windows' clipboard encourages to copy and paste data (for instance ILWIS maps and tables) into other Windows-based application such as word processors, desk top publishing or drawing packages.
- Windows' use of extended memory largely solved DOS 640K memory limitation problems. Therefore, a lot of limitations that were present in ILWIS 1.4 could be removed. In case you run Windows on a computer that has only 4 or 8 MB of RAM, it is strongly advised to create a permanent Windows swap file of at least 8 MB.
- Windows' 'multi-tasking' allows you to work with multiple applications at the same time. For ILWIS for Windows, this means that you do not have to wait until any ILWIS operation has finished: you can continue with the next step (another ILWIS operation) or with any other Windows application.
- Windows supports on-line and context-sensitive Help.
- Windows handles communication with the graphic board and other devices such as printers and plotters. Therefore, all devices which can be installed under Windows can also be used by ILWIS. To properly work with ILWIS, make sure that your graphic board is configured with at least 256 colors. ILWIS also allows 24 bit display.

- Communication with digitizers is handled by ILWIS, similar as in version 1.4.
- ILWIS is capable of acting as a DDE server.

ILWIS window types

- ILWIS for Windows consists of the Main window with Catalog, Operation-list and command line; map windows; table windows with command line; graph windows; a pixel information window; and editor windows.
- The Main window is divided in a Catalog which gives an overview of maps, tables and other objects, and an Operation-list which gives an overview of available operations. The command line in the Main window allows you to type MapCalc formulas, and to perform operations as you would do on the DOS-command line in ILWIS 1.4.
- Each map is displayed in a map window. You can easily zoom in and scroll through a map. You can display combinations of raster and vector maps within one map window and save this as a map view. Multiple map windows can be displayed on the screen where each map uses its own colors. By pressing the left mouse button in a map window, you get information of the contents of that pixel, polygon, segment or point.
- Each table is displayed in a table window. Tables are either displayed as a whole or record by record. You can easily scroll through a table and change the width and order of columns. A command line is integrated in each table window which allows you to type TabCalc formulas and directly see the results in the table.
- The pixel information window allows you to see the values or meanings of all maps (raster and vector) added to this window at the current position of the mouse pointer in a map window or the digitizer cursor. When a map has an attribute table, the pixel info window shows also the attribute values. You can add as many maps to the pixel info window as you like.
- Especially in the Catalog and Operation-list, you can use double-click and drag and drop which facilitates actions such as opening maps and tables, starting operations, adding extra layers to a map window, adding maps and tables to the pixel info window, and so on. By pressing the right mouse button, a context-sensitive menu appears. For more information, see How to use the Catalog and How to use the Operation-list.

ILWIS data structures

- ILWIS for Windows works with ILWIS objects.
  - Point maps, segment maps , polygon maps, and raster maps, map lists, tables and columns are called data objects.
  - Domains, representations, georeferences and coordinate systems are called service objects.
  - Map views, histograms, sample sets, two-dimensional tables, matrices, filters, functions and scripts are called special objects.

- Each map, table and column in ILWIS uses a domain. A domain contains the set of IDs, Class names or values that you want to use in one or more maps, attribute tables, or columns. You can create for example a class domain Landuse filled with Landuse classes, or a class domain Soil filled with Soil classes. For values, you can for instance create value domain Height for which you define a certain range of values for an Elevation map, etc.
- One domain can be used for multiple maps and tables; not every class in a domain Class or every value in a domain Value etc. has to exist in each map for which you use that domain. Domain Group is a synonym for ILWIS 1.41 classify tables.

Some structural relations exist between objects

- Raster maps use a domain and a georeference; a domain uses a representation and a georeference uses a coordinate system.
- Vector maps use a domain and a coordinate system. A coordinate system may contain projection information.
- Tables use a domain and have columns. Each column uses a domain. For more information on structural relations between objects, see Appendices : relations between ILWIS objects.
- To see for instance which service objects are used by a data object, you can view (and edit) the properties of that data object .
- The properties of objects give some general information on the objects and shows the names of the objects that are used.
- In ILWIS, each object has an ASCII object definition file. This file contains all necessary ILWIS information for the object. For instance, the object definition file of a raster map contains information on its creation time and description, and contains further references to the name of the data file, the store type of the map, and the names of the domain and georeference that are used.
- When the map is created by a MapCalc formula or an operation, the MapCalc expression or the operation with all its parameters is stored .

Maps

- In ILWIS there are point, segment, polygon and raster maps and map lists.
- Maps use a domain which defines the possible values in a map (domain Value), contains the possible class names in a map (domain Class), or contains the possible identifiers in a map (domain ID). Other domain types are for example Image, Bit, Bool, and Picture.
- Once you created for instance a domain Class containing e.g. LandUse classes for a polygon map, you can use the same domain for the raster map and for the attribute table. Also, for other maps that contain the same units but that give for instance information on a different year, the same domain should be used.
- Value maps (i.e. maps using a value domain) can contain all values you like, depending on the value range and precision you chose. The scale factor for integer maps of version 1.4 is obsolete as raster maps can now also be stored as 4 and 8 byte maps ('real' maps).
- Representations replace color look up tables of version 1.4. Unlike ILWIS 1.4 where a color look up table was linked to a specific map, ILWIS representations

are linked to a domain. This ensures that all maps that use the same domain are automatically displayed in the same colors regardless of whether they are raster or vector maps. Also, a representation can be prepared for a domain of a column in a table to enable the proper display of attribute data.

- A domain Class, Picture, Image or Value can have a user-defined representation. Domain Class representations are edited in the representation Class editor, and domain Value representations are edited in the representation Value/Gradual editor.
- Value maps are temporarily classified or sliced by creating a user-defined representation Value or Gradual in the representation Value/Gradual editor. In a map window, the map appears classified, but the original values of the map remain.
- Raster maps using a value domain are permanently classified or sliced by using the Slicing operation or the CLFY function in MapCalc. A domain Group has to be created beforehand: it contains the value boundaries and the output class/group names. A domain Group replaces the classify tables (.CLT) of ILWIS 1.4.
- Two raster maps (using a class or ID domain) can be reclassified using a two-dimensional table.
- Maps are georeferenced by creating a georeference, or using an existing one. Three main types of georeferences exist: georeference corners, georeference tiepoints, and georeference 3D. A georeference uses a coordinate system which may contain projection information. Point, segment and polygon maps directly use a coordinate system that may contain projection information. It is advised to create one large coordinate system in which all maps fit that you are working with.
- Point and segment maps can easily be created and edited by using the mouse as well as a digitizer in the point editor and the segment editor. Polygon maps have to be created by first digitizing segments and label points, then polygonize the segments in the segment editor. Polygons can be edited in the polygon editor.
- Raster maps can be edited using the mouse in the pixel editor.
- The Sample set editor allows you to select training pixels prior to an image classification.

Attributes

- Maps of a domain Class or domain ID can have attribute information stored in an attribute table. An attribute table can be linked to a map (raster and vector) but also to a domain through the properties of the map or the domain.
- The link between maps and attribute tables is provided by a domain Class or a domain ID which must be used both by the map and the table.
- Domain Class and ID maps (raster and vector) that have an attribute table, can be 'displayed by attribute' through the display options dialog box of the map, i.e. without the need of creating an attribute map first. A map displayed by attribute, uses the colors of the domain of the attribute column. The display of maps by attribute can be considered as creating temporary attribute maps.

- Permanent attribute maps can be created from raster maps, but also from polygon, segment and point maps through an operation.

Annotation
Annotation can be added as one or more annotation layers to a map window: text, legend, scale bar, North arrow, grid lines, graticule, boxes, and bitmaps or pictures from disk. Pictures and bitmaps can also be pasted into ILWIS from other Windows application programs (via the clipboard). Annotation can be displayed in a map window in combination with maps: this can be saved as a map view.

Dependent objects
- By performing an operation in ILWIS, usually dependent objects are created. Dependent objects know how they are created and how they have to be re-calculated because the complete expression/definition (input objects, operation + parameters, MapCalc or TabCalc formula) of each output object is stored inside the object definition file of the output object.
- One of the advantages of working with dependent maps, tables or columns is that they can easily be recalculated when needed and input objects or expressions have changed in time.
- Another advantage of working with dependent raster maps is that most of them can be used in the Pixel Info window even before they are actually calculated and stored on disk. As Pixel Info reads the definition of the dependent object, output values of dependent raster maps are calculated on the fly.
- In the same way, you can also create a number of dependent maps at a go by only storing their definition, and leave the computer to calculate and store all of these maps on disk later on.
- Dependent output objects are created by performing an operation through a dialog box, or by using the definition symbol = in a MapCalc formula typed on the command line of the Main window, or a TabCalc formula typed on the command line in a table window.
- When the assignment symbol := is used on the command line of the Main window or the command line of a table window, a so-called source map or source-column is created. In that case, the formula typed on the command line is not stored inside the output map or column (1.4 method).

Map calculation and Table calculation:
MapCalc and TabCalc are integrated in ILWIS and thus use the same syntax, function names, etc. Map calculations are performed on the command line of the Main window or through the Map Calculate command on the Operations, Raster Operations menu.
Table calculations are performed on the command line of a table window. For a complete overview of MapCalc and TabCalc operators and functions, refer to Appendices : Operators and functions in MapCalc and TabCalc.

ILWIS operations general

- All operations are accessed through the Operations menu in the Main window, or through the Operation-list, or by dragging objects to each other within the Catalog, or by dragging objects from the Catalog to an item in the Operation-list;
- Operations can also be performed by typing a ILWIS command or a complete ILWIS expression on the command line.

Scripts

- Scripts offer about the same functionality as the batch files in version 1.4. For more information, refer to Appendices : ILWIS Script language.

Appendix B

ILWIS objects

B.1 ILWIS objects

Data objects

Raster maps, polygon maps, segment maps, point maps, tables and columns are called data objects.

Raster maps
A raster map consists of pixels (picture elements) of a certain size, e.g. 20m x 20m. Pixels are either codified by IDs, class names, values, or colors; this is determined by the domain of the map. A raster map should have coordinates, that is a georeference. In ILWIS, most spatial operations are performed on raster maps.

To obtain a raster map:
- rasterize an existing point, segment or polygon map, or
- create a new raster map and edit it with the pixel editor,
- use a satellite image which is already a raster map, or
- scan a map or photograph and import it into ILWIS.

Polygon maps
A polygon map is a vector map containing closed areas and the boundaries making up the areas. Polygon maps can for example contain uniquely codified areas such as cadastral plots, or mapping units such as land use classes, geological units or soil units. The areas are either codified by IDs, class names or values; this is determined by the domain of the map. Further, a polygon map uses a certain coordinate system.

Polygon maps are generally used as a stepping stone to raster maps.

To obtain a polygon map, you have to create a segment map, with an optional label point map, and then polygonize the segments, for instance in the segment editor.

Segment maps
A segment map is a vector map containing lines (for example roads, rivers or contour lines). Segments are either codified by IDs, class names or values (height map); this is determined by the domain of the map. Further, a segment map uses a certain coordinate system.

To obtain a segment map, you should create one and edit it with the segment editor (with or without digitizer).

Point maps
A point map contains points, for example water wells or sample points. Points are either identified by IDs, class names or values; this is determined by the domain of the map. Further, a point map uses a certain coordinate system.

To obtain a point map you should create one and edit it with the point editor (with or without digitizer).

Map lists
A map list is a set of raster maps, for example the bands of a satellite image. All raster maps in the map list must have the same georeference and the same domain.

A map list is used for:
- sampling and classification
- creating a color composite
- a principal components analysis, or
- (not yet available) a time series analysis.

A map list can be created from the File menu in the Main window or while starting an operation which requires a map list as input. You can include as many maps in a map list as you like.

(Attribute) Table
A table consists of records and columns. A table is an attribute table when the table stores additional information on elements in a map; i.e. extra tabular data which relates to mapping units, points, segments, or polygons in maps.

Raster, polygon, segment and point maps of the domain type Class or ID can have attribute tables. The domain of the attribute table should be the same as the domain of the map to which it relates. An attribute table can be linked to a map or to a domain through the Properties of a map or a domain.
An attribute table can be edited when the table is displayed in a table window. When the table is linked to a map or to a domain, and the map is displayed in a map window, you can also double-click in the map.

Columns
A table consists of columns. You can perform calculations with columns using TabCalc.
Each column has a domain. A column with a value domain contains values, a column with a class domain contains class names, a column with domain ID contains IDs, etc. Columns can also have domain String; you can use this to type for instance descriptions.
If in an attribute table you have columns with a class domains, or with user-defined value domains, you may consider to prepare a representation for these columns as well. When you open the map to which the attribute table is linked, and you can directly display the attributes.

Service objects

Domains, representations, georeferences, and coordinate systems are called service objects.

Domains

A domain a service object; it is the user-defined collection of IDs, class names, or values that can be used in a certain data object. All ILWIS data objects have a domain. One domain can be used for several data objects. User-defined domains can be listed in the Catalog.

The four main types of domains are:

- ID data object contains unique identifiers (e.g. plot 1024, plot 1025)
- Class data object contains classes (e.g. soil units: clay, sandy loam)
- Value data object contains measurable values (e.g. height, concentration)
- Image data object is a satellite image containing values between 0 and 255.

Maps using a domain Class, Value or Image can have a user-defined representation.

Representations

A representation is a service object containing:

- colors assigned to classes in maps with a class or group domain (representation class)
- colors assigned to ranges of values in map with a value domain (representation value/gradual),
- patterns or colors for polygons in polygon maps with a class domain; line width and colors (line type not yet) for segments and
- polygon boundaries; symbol type, color, size of symbol etc. for points in a point map; and colors in a map with a picture domain.

Representations are linked to domains. Maps using the same user-defined domain are by default displayed in the same colors. It is therefore advised to create a user-defined domain and a user-defined representation for maps which have a specific meaning (e.g. land use classes or height values) and need fixed colors.

Georeferences

A georeference is a service object which stores the relation between rows and columns of a raster map (row,col) and coordinates (X,Y). A georeference is needed for raster maps. A georeference uses a coordinate system.

It is advised that raster maps of the same area use the same georeference, because raster operations in which raster maps are combined only make sense if the pixels in the maps refer to the same area on the ground.

For rasterized vector maps, which are usually North-oriented, you can use a georeference corners. For other raster maps, for example satellite images, which are not North-oriented and in which the pixels do not represent exactly square areas on the ground, you can use a georeference tiepoints. To create three dimensional pictures using a Digital Elevation Model, a georeference 3D needs to be created. To combine raster maps with different georeferences, use the Resample operation.

Coordinate systems
A coordinate system is a service object which defines the kind of coordinates you are using in your maps. A coordinate system may be a national standard, a certain UTM zone with a geographic datum, or a regional, local or user-defined system.

A coordinate system defines the possible X and Y values that can be used in maps. Point, segment and polygon maps have a coordinate system. Raster maps have a georeference which uses a coordinate system.
A coordinate system may contain projection information. If projection parameters are filled out, then coordinate information is also available in geographic coordinates (lat,long), and transformations between different coordinate systems are possible.

☞ It is advised to use one coordinate system for all your maps. In case you have data of different sources in different projections, it is advised to transform all data to one common coordinate system.

Special objects

Map views, histograms, sample sets, two-dimensional tables, matrices, filters, functions, scripts are called special objects.

Map views
A map view is a saved map window. When a map view is opened, the set of data and/or annotation layers that it contains is directly displayed.

A map view contains the names of data layers and/or annotation layers to be displayed in one map window. Also, the display options of the layers are stored; so the system knows the colors, widths etc. for the display of each layer. Further, the georeference is stored, meaning that when you save a map view when zoomed in on a part of the map, this zoomed area will be displayed when opening the map view later.

Histograms
A histogram is a special object which lists frequency information on values, classes or IDs in a raster, polygon, segment or point map. A histogram is automatically calculated when displaying a value map with stretching; you can also use the Histogram operation. The values in a histogram are presented as a table; optionally a graph can be shown.
Summary information of a value histogram (mean, standard deviation, and intervals) can be viewed through the Properties dialog box of the histogram.

A raster histogram lists the number of pixels, the percentages and the areas per value, class or ID. If the input raster map uses a domain Value, also cumulative number of pixels and cumulative percentages are calculated.

A polygon histogram lists the number of polygons and the perimeter and area of polygons per class, ID, or value. If the input polygon map uses a

Value domain, also the cumulative number of polygons, cumulative perimeters and cumulative areas are calculated.

A segment histogram lists the number of segments and their length per class, ID or value. If the input segment map uses a Value domain, also the cumulative number of segments and cumulative lengths are calculated.

A point histogram lists the number of points per class, ID or value. If the input point map uses a Value domain, also the cumulative number of points are calculated.

Sample sets

Prior to an image classification, sample pixels or training pixels have to be selected in a sample set. To create a sample set, first a map list and a domain have to be specified. Then, with sampling, assign class names to groups of pixels that are supposed to represent a known feature on the ground and that have similar spectral values in the maps in the map list.

A sample set contains:

- a reference to a background map, on which you can locate your training pixels;
- a reference to a map list, that is the set of images you want to classify in a later stage. The spectral values of the images in the map list, at the position of the training pixels provide the basis on which decisions are made in the classification. During sampling, these values can be inspected in the sample statistics of a certain class of training pixels, and can be visualized in feature spaces;
- a reference to a class domain, that is the collection of class names that you want to assign to your training pixels and that are the classes that you want to obtain from the classification. The representation of this domain determines in which colors the training pixels are displayed during sampling;
- a reference to a raster map which is automatically created and obtains the same name as the sample set. This so-called sample map contains the locations of the training pixels and the class names assigned to them.

Two-dimensional tables

A two-dimensional table is used to combine two raster maps with class or ID domain. It defines a new value for each possible combination of input classes or IDs.

A two-dimensional table view consists of rows which represent the domain of one map and of columns which represent the domain of another map. You have to assign a new value, class name or ID to the fields which represent the combination of the domains.

Then you have to apply the two-dimensional table on the command line of the Main window. The output raster contains the values, classes or IDs as entered in the two-dimensional table.

Matrices

A matrix is a 2-dimensional array of values. Matrices are calculated by the Principal Components operation and by the Factor Analysis operation. The Principal Components operation calculates a.o. a variance-covariance matrix and the Factor Analysis operation calculates a.o. a correlation matrix. The matrices can be shown.

In the properties dialog box of a matrix, some additional information of the matrix can be viewed namely the total variances in the output bands.

Filters

User-defined filters can be created in the input dialog box of the Filter operation or through the File menu in the Main window, etc. For user-defined linear filters, you can specify filter size, matrix values and a gain factor.

Other filter types can also be user-defined, but this must be done on the input dialog box of the Filter operation: Average, Majority, Majority, Rank Order, Median and Pattern. You can specify the filter size, thresholds, ranks, etc.

A number of standard filters are available in the ILWIS\SYSTEM directory.

Functions

Functions can be used in all three calculators in ILWIS: MapCalc, in TabCalc, in scripts or in the pocket line calculator. Some 50 internal functions are available (see also MapCalc and TabCalc), but also user-defined functions can be created.

A user-defined function may contain any combination of operators and functions, and may use parameters representing maps and columns. Parameters in a user-defined function can have any name.

Scripts

A script is a sequence of ILWIS expressions. By creating a script, you can build a complete GIS or Remote Sensing analysis for your own research discipline. Scripts are equivalent to batch files in ILWIS version 1.4.

B.2 Relations between ILWIS objects

ILWIS data objects (maps, tables and columns) contain references to other objects (e.g. service objects) and to other binary files (e.g. binary data file). All these references are listed in an object's object definition file.
This Help topic deals with the relations that each object has with other objects by means of the references mentioned above. In the software, these internal relations of one object with other objects are visible as the properties of an object. Any object's properties can be viewed and edited.

☞ ILWIS data objects may also depend on other data objects: when a map is calculated from other maps or when a map is the outcome of an operation on another map. Such relations are called dependency links. For more information, see Basic concepts: Dependent data objects.

☞ In the text below, the word 'use' is used for the references that each object has to other objects. Instead of 'use', you may also read 'have' or 'have a reference to'.

Data objects

Raster maps use a domain and a georeference.
Polygon maps use a domain and a coordinate system.
Segment maps use a domain and a coordinate system.
Point maps use a domain and a coordinate system.
Maps of a domain type Class, ID, or Group may have an attribute table.

Tables use a domain.
Columns use a domain.
Map lists contain references to a number of raster maps which use one georeference.

Service objects

Domains of the Class, Group, Value and Picture type use a representation.
A domain Class, Group and Picture have a representation Class, a domain Value has a representation Value or a representation Gradual. Other domains use 1, 7, 15, or 31 system colors.
Representations use a domain.
Georeferences use a coordinate system. A georeference tiepoints also uses a background raster map on which you can position your tiepoints.
Coordinate systems may contain information on a projection.

Special objects

Histograms are tables. Histograms calculated from images or from existing raster, polygon, segment or point with a Class or ID domain, use the same domain as the map. Histograms calculated from maps with a Value domain, obtain domain None.
A map view uses one or more maps (one raster map, multiple polygon, segment, and/or point maps) and may contain annotation information.

A sample set uses a map list, a sample set raster map which stores the positions of the training pixels, a domain that contains the class names of training pixels, and a background raster map on which you can position your training pixels.

A two-dimensional table uses two domains which contain the classes or IDs that should be reclassified and one domain which contains the classes, IDs or values after reclassification.

A matrix has no direct relations to other objects. A matrix is a dependent object, created by the operations Principal Components and Factor Analysis.

Filters have no direct relations with other objects. Filters can be applied on raster maps.

Functions have no direct relations with other objects. Functions can be used on the command lines, in MapCalc and TabCalc to calculate with maps and/or tables, and in the pocket line calculator to calculate with constant values.

Scripts have no direct relations with other objects. Scripts may perform any calculation or operation on any data object, as defined in the script.

- When you create a domain for data with a meaning, do not forget create one or more representations for this domain as well. When creating a domain Class, a representation is automatically created, but you have to edit the colors yourself. The domain and representation together determine in what colors the map or attribute column can be displayed.
- Service objects can be shared by more than one data objects.
- Some special service objects exist: None.dom, None.grf, and Unknown.csy.

Appendix C

ILWIS operations

VISUALIZATION

Show Map
Show a map in a new map window.

Show Table
Show a table in a table window.

Color composite
A color composite is a combination of three raster bands. One band is displayed in shades of red, one in shades of green and one in shades of blue. Putting three bands together in one color composite map can give a better visual impression of the reality on the ground, than by displaying one band at a time. Examples of color composites are false color (or IR) images and 'natural color' images.

Display 3D
With Display 3D, you can create and edit a georeference 3D in order to obtain a three dimensional view of one or more maps. A Digital Elevation Model (DEM) is required to create a georeference 3D. The DEM can then be displayed as 3D grid lines with or without a drape of a raster map. The georeference 3D can be edited with the Georeference 3D editor.
When finished editing the 3D view, you can add point, segment and/or polygon maps and/or annotation to improve the 3D view.

Apply 3D
The Apply 3D operation resamples an input raster map according to a georeference 3D. This enables you to permanently and quickly display the output raster map in three dimensions, i.e. as a 3D view.
You can create a georeference 3D with Display 3D or Apply 3D; a Ditigal Elevation Model is required and you have to specify 3D view parameters using the Georeference 3D editor.

Slide Show
With Slide Show, you can display multiple raster maps that are combined in a map list as a slide show in a map window. The maps of the map list are displayed one after the other in the map window, at a user-specified rate. A slide show is designed to present multi-temporal changes in maps.

RASTER OPERATIONS

Map Calculation
Map Calculation can be used to perform calculations with raster maps. Map calculation is used for the execution of most spatial analysis functions and modeling operations. It integrates spatial and tabular data. The program enables the user to perform overlay, retrieval operations, and queries. Type your map calculation formulae on the command line of the Main Window.

The following operations can be executed:

- Manipulation of one or more raster maps by performing arithmetical, relational, logical, conditional, exponential, logarithmic and other operations;
- Creation of attribute maps from map-related tables with attribute data;
- Classification of domain Value maps according to a domain Group;
- Application of user-defined functions.

Attribute map of raster map
By creating an attribute map of a raster map, the class name or ID of each pixel in the original map is replaced by the value, class or ID found in a certain column in an attribute table.

A raster map using a Class or ID domain, can have extra attribute information on the classes or identifiers in the map. These attributes are stored in columns in an attribute table. The attribute table can be linked to the map to which it refers, or to the domain of the map. You can check whether an attribute table is linked to the raster map or to its domain through the Properties dialog box of the map or the domain.

Cross
The Cross operation performs an overlay of two raster maps. Pixels on the same positions in both maps are compared; the occurring combinations of class names, identifiers or values of pixels in the first input map and those of pixels in the second input map are stored. These combinations give an output cross map and a cross table. The cross table also includes the number of pixels that occur for each combination.

Aggregate map
The Aggregate map operation aggregates blocks of input pixels by applying an aggregation function: Average, Count, Maximum, Median, Minimum, Predominant, Standard Deviation or Sum. The Aggregate Map operation either creates a new georeference in which each block of input pixels corresponds to one output pixel (group) or the output raster map uses the same georeference as the input map (no group).

Distance calculation
The Distance operation assigns to each pixel the smallest distance in meters towards user-specified source pixels, for example distance to schools, to roads etc.

The output is called a distance map. Input for a distance calculation is a source map and optionally a weight map.

By using weight factors that are inversely proportional to the possible speed that can be obtained in different mapping units, a so-called time travel map can be calculated. Through a distance calculation, also a Thiessen map can be calculated.

Iteration
Iterations are a special type of map calculations. They are a successive repetition of a mathematical operation, using the result of one calculation as input for the next. These calculations are performed line by line, pixel by pixel and take place in all directions.

When a calculation in one direction is finished (for instance from top to bottom) a rotation takes place for the calculation in the next direction. The calculation stops when there are no more differences between an output map compared to the previous output map, or when a certain number of iterations is reached as defined before. Iterations are often used in combination with neighbourhood operations.

Area numbering
Area numbering is a raster operation which assigns unique identifiers to pixels with the same class names or values that are horizontally, vertically or diagonally connected. The output of the Area numbering operation is a map in which these connected areas are codified as Area 1, Area 2, etc. Further, an attribute table is created with the map, which contains the new Area IDs and the original class names, IDs or values.

Area numbering can be used to make a decision based on the size of individual groups of pixels (uniquely identified areas) instead of all pixels of a class name or value.

Sub-map of raster map
The Sub-map of raster map operation copies a rectangular part of a raster map into a new raster map. The user has to specify row and column numbers of the input map to indicate the part of the input map that should be copied into the new raster map.

Glue raster maps
The Glue raster maps operation glues or merges two georeferenced input raster maps into one output raster map. The output map then comprises the total area of both input maps. The domains of the two input maps are merged when needed. With the Glue raster maps operation, you can thus merge two adjacent or partly overlapping raster maps, or glue a fully overlapping smaller raster map onto a larger one. To produce a mosaic, you can perform this operation a number of times.

Mirror Rotate
The Mirror/Rotate operation allows you to reflect a raster map in a horizontal, vertical, or diagonal line, to transpose the map's rows and columns, or to rotate a raster map 90°, 180°, 270° (clock-wise).

IMAGE PROCESSING

Filter
Filtering is a raster operation 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.
Filtering is for instance used to sharpen a satellite image, to detect line features, etc.

Stretch
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.

Slicing
The Slicing operation classifies ranges of values of an input raster map into classes of an output map. A domain Group should be created beforehand; it lists the upper value boundaries of the groups and the output class names.

To perform an interactive slicing, you can create a representation value for the input map and change value boundaries and colors of the representation value.

Color separation
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.

Cluster
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.

Sample
Sample is an interactive process of selecting training pixels in a sample set prior to an image classification.

In the sample set editor, 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.

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

The following classification methods can be used:
- Box classifier;
- Minimum Distance to Mean classifier;
- Minimum Mahalanobis Distance classifier;
- Maximum Likelihood classifier.

Resample
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;
- 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.

STATISTICS

Histogram
The Histogram operation calculates the histogram of a raster, polygon, segment or point map. Histograms list frequency information on the values, classes, or IDs in your map. Results are presented in a table and optionally in a graph. Summary information of a histogram of a Value raster map can be viewed in the properties of the histogram (mean, standard deviation, and percentage intervals).

A raster histogram lists the number of pixels, the percentages and the areas per value, class or ID. If the input raster map uses a Value domain, also cumulative number of pixels and cumulative percentages are calculated.

A polygon histogram lists the number of polygons and the perimeter and area of polygons per class, ID, or value. If the input polygon map uses a Value domain, also the cumulative number of polygons, cumulative perimeters and cumulative areas are calculated.

A segment histogram lists the number of segments and their length per class, ID or value. If the input segment map uses a Value domain, also the cumulative number of segments and cumulative lengths are calculated.

A point histogram lists the number of points per class, ID or value. If the input point map uses a Value domain, also the cumulative number of points are calculated.

Raster

Auto correlation
The AutoCorrelation operation calculates the auto-correlation and semi-variance in a raster map. The Autocorrelation operation calculates the correlation between pixel values of a raster map and pixel values of the same raster map for different shifts (lags) in horizontal and vertical directions. The semi-variance, a measure for the spatial variability of a raster map is calculated for the same shifts.

Map list

Principal Components
The Principal Components analysis calculates the variance-covariance matrix for a map list. New output bands are constructed in such a way that the largest variation is written to new band 1, the second largest perpendicular variation to band 2, etc.

Factor Analysis
The Factor analysis calculates the correlation matrix for a map list. New output bands are constructed in such a way that the largest correlation is written to new band 1, the second largest perpendicular correlation in the second band, etc.

Variance-covariance matrix
The Variance-Covariance matrix operation calculates variances and covariances of raster maps in a map list. The variance is a means to express the variation of pixel values within a single raster map, i.e. a measure of the variation to the mean of the DN (Digital Number) values in a raster map. The covariance is a measure to express the variation of pixel values in two raster maps. It denotes the joint variation to the common mean of pixel values of the maps. Furthermore, the mean and standard deviation of each individual raster map is calculated.

Correlation matrix
The Correlation matrix operation calculates correlation coefficients of input raster maps of a map list. Correlation coefficients characterize the distribution of pixel values in two raster maps. Furthermore, the mean and standard deviation of each individual raster map is calculated.

Segments **Segment Direction Histogram**
The Segment Direction Histogram operation calculates the total length of segments per direction. The output is a table with directions from 0 to 179° and the length of the segments that run in that direction. The results can be displayed as a rose-diagram.

Points **Spatial correlation**
Spatial correlation calculates some point statistics: autocorrelation and (semi-) variance. Input is a point map with values or a point map with values in a column in an attribute table. Output is a table; graphs can be created from the output table.

Pattern Analysis
Pattern analysis gives information on the spatial distribution of points in a point map. The output is a table which contains six columns of probabilities of finding 1 point within a certain distance (P1), then 2 points (P2), 3 points (P3), etc. Another column (PAll) contains the sum of P1, P2, ..., P(n-1), in which n is the number of points in the map.

By inspecting the graphs of probabilities, patterns like random, clustered, regular, paired etc. can be recognized.

INTERPOLATION

Raster **Densify raster map**
The Densify raster map operation reduces the pixel size of your map. The number of rows and columns is increased and the new pixels in between the existing ones are assigned a value by nearest neighbour or by means of a bilinear or bicubic interpolation You should use densify after a point interpolation. Further, densify can be used to improve the quality of printed raster maps.

Segments **Contour interpolation**
Contour interpolation is an operation which first rasterizes segments of a domain Value segment map, and then calculates values for pixels that are not covered by segments by means of a linear interpolation. When using Contour interpolation on a segment map containing height (contour) information, the resulting raster map is a Digital Elevation Model.

Points **Point interpolation**
In a point interpolation, the input map is a point map, and the output map is a raster map. The pixel values in the raster output map are interpolated from the points.

There are four point interpolations:

- Nearest point: assigns to pixels the value, identifier or class name of the nearest point. This method is also called Nearest Neighbor or Thiessen;
- Moving average: assigns to pixels weighted averaged point values. Weights are calculated by a weight function. Points that are further away from an output pixel than the user-specified limiting distance, are assigned weight zero;

- Trend surface: calculates pixel values by fitting a surface through all point values in the map;
- Moving surface: calculates pixel values by fitting a surface through weighted point values. Weights are calculated by a weight function. Points that are further away from an output pixel than the user-specified limiting distance, are assigned weight zero.

Nearest point (Point interpolation)
The Nearest point operation requires a point map as input and returns a raster map as output. Each pixel in the output map is assigned the class name, identifier, or value of the nearest point. This method is also called Nearest Neighbour or Thiessen.

For example schools, hospitals, water wells, etc. can be represented by points. The output map of a nearest point operation on such a point map gives the 'service area' of the schools, hospitals or water wells, based on the shortest distance (as the crow flies) of a point and pixels.
When you have many points or when you want to weights (e.g. for travel time maps), it is advised to rasterize the points and then use Distance calculation and make a Thiessen map.

Moving average (Point interpolation)
The Moving average operation is a point interpolation which requires a point map as input and returns a raster map as output. To the output pixels, weighted averaged point values are assigned. Weights are calculated by a weight function. Points that are further away from an output pixel than the user-specified limiting distance, are assigned weight zero.

When interpolating point values, it is for time efficiency reasons , strongly advised to choose a rather large pixel size for the output map. Further interpolation on the raster map values can be performed using the Densify operation.

Trend surface (Point interpolation)
The Trend surface operation is a point interpolation which requires a point map as input and returns a raster map as output. One polynomial surface is calculated by a least square method approaching all point values in the map. To the output pixels, the calculated surface values are assigned.
When interpolating point values, it is for time efficiency reasons , strongly advised to choose a rather large pixel size for the output map. Further interpolation on the raster map values can be performed using the Densify operation.

Moving surface (Point interpolation)
The Moving surface operation is a point interpolation which requires a point map as input and returns a raster map as output. For each output pixel, a polynomial surface is calculated by a least square method approaching all point values. As points closer to an output pixel are more important than points further away, a weight function has to be specified. Points that are further away from an output pixel than the user-specified limiting distance, are assigned weight zero.

When interpolating point values, it is for time efficiency reasons , strongly advised to choose a rather large pixel size for the output map. Further interpolation on the raster map values can be performed using the Densify operation.

VECTOR OPERATIONS

Unique ID
The Unique ID operation can be used to assign a unique ID to all elements in a segment, polygon or point map. The result is an ID map that contains the same geographic information as the input map but now each point, segment or polygon has a unique ID.
Further an attribute table is created which uses the same ID domain as the output map; the table contains one column with the original classes, IDs or values of the input map. The domain of the table together with the column establishes the relation between the original classes, IDs or values in the input map and the output IDs.

Polygons

Attribute map of polygon map
By creating an attribute map of a polygon map, the class name or ID of each polygon in the original map is replaced by the value, class or ID found in a certain column in an attribute table.

A polygon map using a Class or ID domain, can have extra attribute information on the classes or identifiers in the map. These attributes are stored in columns in an attribute table. The attribute table can be linked to the map to which it refers, or to the domain of the map. You can check whether an attribute table is linked to the polygon map or to its domain through the Properties dialog box of the map or the domain.

Mask polygons
The Mask polygons operation allows you to selectively copy polygons of an input polygon map into a new output polygon map.
The user has to specify a mask to select and retrieve the class names, IDs or values of the polygons that are to be copied.

Assign labels to polygons
The Assign labels to polygons operation can be used to recode polygons in a polygon map according to label points in a point map. For each label point, the surrounding polygon is determined; then the class name, ID, or value of the label point is assigned to that polygon.

Transform polygon map
The Transform polygon map operation transforms polygons in a polygon map to a new coordinate system. Both coordinate systems require projection information.

Segments

Attribute map of segment map
By creating an attribute map of a segment map, the class name or ID of each segment in the original map is replaced by the value, class or ID found in a certain column in an attribute table.

A segment map using a Class or ID domain, can have extra attribute information on the classes or identifiers in the map. These attributes are stored in columns in an attribute table. The attribute table can be linked to the map to which it refers, or to the domain of the map. You can check whether an attribute table is linked to the segment map or to its domain through the Properties dialog box of the map or the domain.

Mask segments
The Mask segments operation allows you to selectively copy segments of an input segment map into a new output segment map.
The user has to specify a mask to select and retrieve the class names, identifiers or values of the segments that are to be copied.

Assign labels to segments
The Assign labels to segments operation can be used to recode segments in a segment map according to label points in a point map. For each label point, the closest segment is determined; then the class name, ID or value of the label point is assigned to that segment.

Sub-map of segment map
The Sub-map of segment map operation copies a rectangular part of a segment map into a new segment map. The user has to specify minimum and maximum XY-coordinates for the new segment map.

Glue segment maps
The Glue segment maps operation glues or merges two or more segment maps into one output map. By default, the output map then comprises the total area of all input maps. The domains of the input maps are merged when needed.

For each input map, the user can specify a mask to select and retrieve the class names, IDs or values of the segments that are to be copied into the output map. The user can also specify a clip boundary, to copy only those segments to the output map which fall within the specified coordinate boundaries of the output map.

Densify segment coordinates
The Densify segment coordinates operation allows you to obtain more intermediate coordinates within segments. The segments of an input map are copied, and extra intermediate coordinates are added to the segments in the output map at a user-specified distance. It is advised to use this operation before a Transform segments operation is performed.

Tunnel segments
The Tunnel segments operation reduces the amount of intermediate points within segments in a segment map. The segments of the input map are copied into a new segment map. However, for every three consecutive intermediate points within a segment, the middle one is omitted if it falls within a user-defined tunnel-width. Redundant nodes can also be removed.

Transform segment map
The Transform segment map operation transforms segments in a segment map to a new coordinate system. Both coordinate systems require projection information.

Points

Attribute map of point map
By creating an attribute map of a point map, the class name or ID of each point in the original map is replaced by the value, class or ID found in a certain column in an attribute table.

A point map using a Class or ID domain, can have extra attribute information on the classes or identifiers in the map. These attributes are stored in columns in an attribute table. The attribute table can be linked to the map to which it refers, or to the domain of the map. You can check whether an attribute table is linked to the point map or to its domain through the Properties dialog box of the map or the domain.

Mask points
The Mask points operation allows you to selectively copy points of an input point map into a new output point map. The user has to specify a mask to select and retrieve the class names, IDs or values of the points that are to be copied.

Sub-map of point map
The Sub-map of point map operation copies all points within a user-specified rectangle into a new point map. The user has to specify minimum and maximum XY-coordinates for the new point map.

Glue point maps
The Glue point maps operation glues or merges two or more point maps into one output map. By default, the output map then comprises the total area of all input maps. The domains of the input maps are merged when needed.

For each input map, the user can specify a mask to select and retrieve the class names, IDs or values of the points that are to be copied into the output map. The user can also specify a clip boundary, to copy only those points to the output map which fall within the specified coordinate boundaries of the output map.

Transform point map
The Transform point map operation transforms point in a point map to a new coordinate system. Both coordinate systems require projection information.

Transform coordinates
Transform coordinates allows you to type in XY-coordinates using a certain coordinate system, and shows the resulting XY-coordinates for another coordinate system. Both coordinate systems require projection information.

RASTERIZE

Polygons to raster
The Polygons to raster operation rasterizes a polygon map. The class names, IDs, or values in the polygon map are also used in the raster map, i.e. the domain of the polygon map is also the domain of the raster map. The user has to select or create a georeference for the output raster map.

Segments to raster
The Segments to raster operation rasterizes a segment map. The class names, IDs, or values in the segment map are also used in the raster map, i.e. the domain of the segment map is also the domain of the raster map. The user has to select or create a georeference for the output raster map.

Segment density
The Segment density operation rasterizes a segment map. For each output pixel, the total length of segment parts within the boundaries the output pixel is summed: this is the output value for the pixel. By using a mask you can specify the elements of the input map that are to be used in the calculation.

Points to raster
The Points to raster operation rasterizes a point map. The class names, IDs, or values in the point map are also used in the raster map; i.e. the domain of the point map is also the domain of the raster map. The user has to select or create a georeference for the output raster map.

Point density
The Point density operation rasterizes a point map. For each output pixel, the number of points found in the pixel is counted: this is the output value for the pixel. This operation can be used to examine the regional distribution of points.

VECTORIZE

Raster to Polygons
The Raster to Polygons operation extracts polygons from units in a raster map. The output polygon map uses the same domain as the input raster map, i.e. the class names, IDs, or values in the input raster map will also be used for the polygons in the output polygon map. No polygons are created for pixels with the undefined value.

Raster to Segments
The Raster to Segments operations extracts segments from the boundaries of mapping units in a raster map. The segments in the output map either obtain the code `Segments` or a special code which is a combination of the class names or IDs of the two mapping units found on either side of the segment.

Raster to Points
The Raster to Points operation extracts a point from each pixel in the raster map. Each point gets the value, class name or ID of the corresponding pixel.

Polygons to Segments
The Polygons to Segments operation extracts polygon boundaries and creates a segment map out of them. A mask can be specified to extract segments of specific polygons.

Polygons to Points
The Polygon to Points operation creates a point for each polygon in the polygon map. Each point obtains the class name, ID, or value of the corresponding polygon. In this way, polygon label points are created.

Segments to Polygons
The Segments to Polygons operation polygonizes a segment map. All segments in the segment map must be closed, i.e. connected to other segments or to themselves (islands) by nodes; dead ends are not allowed. A mask can be specified to polygonize specific segments.
Mind: to interactively polygonize segments, use the Polygonize option in the Segment editor.

Segments to Points
The Segments to Points operation creates a point map from a segment map.
The output point map can contain either:
- A point for each node in the segment map, or
- Points at a regular distance along the segments in the segment map, or
- Points for all stored coordinate pairs in the segment map.

TABLE OPERATIONS

Transpose table
The Transpose table operation interchanges the rows and columns of a table. Each row of the input table becomes a column in the output table; while column names of the input table become output domain records.

Appendix D

Operators and functions in MapCalc and TabCalc

By typing a MapCalc expression on the command line of the Main window, a raster map is calculated.
By typing a TabCalc expression on the command line of a table window, a column is calculated.

Operators and functions available in MapCalc and Tabcalc are largely the same. However, there are some operators and functions that can be applied on domain Value, Image, Bit, and Bool maps and columns, and other operators and functions apply for domain Class, ID, or Group or Picture maps or columns. Therefore, this topic contains besides general syntax information, 8 sections on operators and functions:

1. for maps and columns of domain Value
2. for maps and columns of domain Class, ID, Group
3. for maps of domain Class, ID, Group only
4. for columns of domain Value only
5. for columns of domain Class, ID, Group or String only
6. aggregations of domain Value columns
7. some predefined values and variables are described
8. special functions to calculate with coordinates, point data, properties and colors.

For more information, refer to MapCalc and TabCalc.

General syntax information

The general syntax for MapCalc and TabCalc expressions is:

```
OUTMAP  = expression
OUTMAP := expression
OUTCOL  = expression
OUTCOL := expression
```

1. The name of the output map or column; extensions are not required.
   Optionally, you can specify an existing domain and/or a value range and/or the precision for the output map or column in a pair of curly brackets.
   map3{dom=*domainname*;vr=*min:max:precision*} = *expression*
   map3{dom=*domainname*;vr=*min:max*} = *expression*
   map3{dom=*domainname*;vr=*::precision*} = *expression*
   map3{dom=*domainname*} = *expression*

map3{vr=*min:max:precision*;dom=*domainname*} = *expression*
map3{vr=*min:max*;dom=domainname} = *expression*
map3{vr=*min:max:precision*} = *expression*
map3{vr=*::precision*} = *expression*
The same applies for columns.

2. A definition symbol (=) or an assignment symbol (:=) to indicate whether to create a dependent output map or column, or an editable source object.
3. An expression consisting of one or more map or column names, operators, and/or functions.

Further:

- ILWIS expressions are not case-sensitive.
- Spaces are allowed after output object names, around the definition or assignment symbol, and before and behind brackets, commas, or semicolons.
- Spaces are not allowed before or behind colons.

1. For maps and columns of domain Value

Arithmetic operators

+	add operator; *a* + *b*
-	subtract operator; *a* - *b*
*	multiply operator; *a** *b*
/	divide operator; *a* / *b*
a MOD *b*	modulus operator; calculates *a* modulo *b*; returns the remainder of *a* divided by *b*, e.g. 10 MOD 3, returns 1
a DIV *b*	integer division operator; dividing two long integers *a* and *b*; returns the quotient, e.g. 10 DIV 3, returns 3

Relational operators

=	eq	*a* = *b*; equal to; tests whether the outcome of expression *a* is equal to the outcome of expression *b*
<	lt than	*a* < *b*; less than; tests whether the outcome of expression *a* is less the outcome of expression *b*
<=	le	*a* <= *b*; less than or equal to; tests whether the outcome of expression *a* is less than or equal to the outcome of expression *b*
>	gt	*a* > *b*; greater than ; tests whether the outcome of expression *a* is greater than the outcome of expression *b*
>=	ge	*a* >= *b*; greater than or equal to; tests whether the outcome of expression *a* is greater than or equal to the outcome of expression *b*
<>	ne	*a* <> *b*; not equal to; tests whether the outcome of expression *a* is not equal to the outcome of expression *b*

Logical operators

AND	(*a*) and (*b*)	logical AND (intersection), returns true if both expressions *a* and *b* are true
OR	(*a*) or (*b*)	logical OR (union), returns true if one or both of the expressions *a* and *b* is true
XOR	(*a*) xor (*b*)	logical XOR, returns true if only one of the expressions *a* and *b* is true
NOT	not (*a*)	logical NOT, returns true if expression *a* is false; returns false is *a* is true

Conditional IFF function

IFF(*a*,*b*,*c*)	If condition *a* is true, then return the outcome of expression *b*, else (when condition *a* is not true) return the outcome of expression *c*. When *a* is undefined, undefined is returned.

Relational function

INRANGE(*a*,*b*,*c*)	tests whether values of expression or map a are contained by *a* range or closed interval with endpoints *b* and *c*. Mathematic notation: $b \leq a \leq c$ or $a \in [b;c]$

Calculation with undefineds

ISUNDEF(*a*)	tests whether *a* is undefined.
IFUNDEF(*a*,*b*)	if condition *a* is undefined, then return the outcome of expression *b*. else the undefined remains.
IFUNDEF(*a*,*b*,*c*)	if condition *a* is undefined, then return the outcome of expression *b*, else (when condition *a* is not undefined), return the outcome of expression *c*.
IFNOTUNDEF(*a*,*b*)	if condition *a* is not undefined, then return the outcome of expression *b*, else the undefined remains.
IFNOTUNDEF(*a*,*b*,*c*)	if condition *a* is undefined, then return the outcome of expression *b*, else (when condition *a* is not undefined) return the outcome of expression *c*.

Exponential functions

SQ(a)	a^2	square function: *a*a*; *a* square
SQ(a,b)	$a^2 + b^2$	square function: *a*a* + *b*b*; *a* square plus *b* square
SQRT(a)	$\sqrt{a}$ of *a*	square root function: calculates the positive square root
HYP(a,b)	$\sqrt{(a^2 + b^2)}$	hypotenuse: calculates the positive square root of the sum of a square and *b* square
POW(a,b)	a^b form	exponential function: *a* raised to the power *b*. The n-th root of *a* ($\sqrt[n]{a}$) is found by using this function in the of: POW(*a*, 1/n)
EXP(a)	e^a	exponential function: value e (i.e. 2.718) raised to the power a

Logarithmic functions

LOG(a)	$^{10}\log(a)$	logarithm: calculates the 10-based logarithm of *a*
LN(a)	$^{e}\log(a)$	natural logarithm: calculates the e-based (2.718) logarithm of *a*

Random functions

RND(*long*)	returns random long integer values in the range [1;*long*]
RND(0)	returns a 0 or a 1 at random
RND()	returns random real values in the range [0;1> , i.e. between 0 and 1, including 0 but excluding 1

Sign functions

-(*a*)	returns *a* multiplied by -1
NEG(*a*)	returns *a* multiplied by -1
ABS(*a*)	returns the absolute (=positive) value of *a*
SGN(*a*)	returns -1 for negative values of *a*, 0 when *a* is 0, and 1 for positive values of *a*

Rounding functions

ROUND(*a*)	rounds *a* to a long integer value, e.g. ROUND(3.5) returns 4, and ROUND(-3.5) returns -3
FLOOR(*a*)	rounds down; returns the largest long integer value smaller than input value (truncation), e.g. FLOOR(3.6) returns 3, and FLOOR(-3.6) returns -4
CEIL(*a*)	rounds up; returns the smallest long integer value larger than input value, e.g. CEIL(3.6) returns 4, and CEIL(-3.6) returns -3

MinMax functions

MIN(*a*,*b*)	returns the minimum of two expressions *a* and *b*
MIN(*a*,*b*,*c*)	returns the minimum of three expressions *a*, *b* and *c*
MAX(*a*,*b*)	returns the maximum of two expressions *a* and *b*
MAX(*a*,*b*,*c*)	returns the maximum of three expressions *a*, *b* and *c*

NDVl function on images

NDVI(*a*,*b*)	(*a*-*b*) / (*a*+*b*) used to calculate the NDVI (Normalized Difference Vegetation Index) of 2 images. Use for a the band with near-infrared reflectance values and for b the band with visible or red reflectance.

Trigonometric functions

SIN(*a*)	sine; input angles must be specified in radians; returns real values in the range -1 to 1
COS(*a*)	cosine; input angles must be specified in radians; returns real values in the range -1 to 1
TAN(*a*)	tangent; input angles must be specified in radians

ASIN(*a*)	arc sine; $\sin^{-1}$; input values must be in the range -1 to 1; returns real values in radians in the range -π /2 to π/2
ACOS(*a*)	arc cosine; $\cos^{-1}$; input values must be in the range -1 to 1; returns real values in radians in the range 0 to π
ATAN(*a*)	arc tangent; $\tan^{-1}$; returns real values in radians in the range -π/2 to π/2
ATAN2(*y*,*x*)	returns the angle in radians of two input values; *y* is vertical, *x* is horizontal

Angular conversion functions

DEGRAD(*a*)	degrees to radians function: *a**2π/360
RADDEG(*a*)	radians to degrees function: (a*360/2π) MOD 360

Hyperbolic functions

SINH(*a*)	hyperbolic sine: $(e^{a}-e^{-a})/2$
COSH(*a*)	hyperbolic cosine: $(e^{a}+e^{-a})/2$
TANH(*a*)	hyperbolic tangent: tanh(*a*) = sinh(*a*)/cosh(*a*)

Special classify function

CLFY(*a* , *DomainGroup*)	classifies the values of *a* according to a domain Group.

2. For maps and columns of domain Class, ID, or Group

☞ When using class names or IDs within an expression, these class names and IDs should be put between double quotes, e.g. "coffee". In domains of the Class or Group type, you can enter codes and class names/group names. These codes can be an abbreviation of your class or group names. In expressions, also the codes can be used.

Relational operators

=	eq	equal to; *a*=*b*; tests whether the outcome of expression *a* is equal to the outcome of expression *b*
<>	ne	not equal; *a*<>*b*; tests whether the outcome of expression *a* is not equal to the outcome of expression *b*

Logical operators

AND	(*a*) and (*b*)	logical AND (intersection), returns true if both expressions *a and b* are true
OR	(*a*) or (*b*)	logical OR (union), returns true if one or both of expressions is true
XOR	(*a*) xor (*b*)	logical XOR, returns true if only one of the expressions *a* and *b* is true
NOT	not (*a*)	logical NOT, returns true if expression *a* is false; returns false if expression *a* is true

Conditional IFF function

IFF(*a*,*b*,*c*)	If condition *a* is true, then perform expression *b*, else (when condition *a* is not true) perform expression *c*

Calculating with undefineds

ISUNDEF(*a*)	tests whether *a* is undefined. Argument *a* can be the outcome of an expression or simply a map name
IFUNDEF(*a*,*b*)	if condition *a* is undefined, then return the outcome of expression *b*. else the undefined remains.
IFUNDEF(*a*,*b*,*c*)	if condition *a* is undefined, then return the outcome of expression *b*, else (when condition *a* is not undefined), return the outcome of expression *c*.
IFNOTUNDEF(*a*,*b*)	if condition *a* is not undefined, then return the outcome of expression *b*, else the undefined remains.
IFNOTUNDEF(*a*,*b*,*c*)	if condition *a* is undefined, then return the outcome of expression *b*, else (when condition *a* is not undefined) return the outcome of expression *c*.

3. For maps of domain Class, ID, Group only

Map.Column	Attribute map: creates an attribute map of attribute values in a column of the attribute table that is linked to the Class or ID map.
TwoDimTable[map1,map2]	Classifying two maps according to a two-dimensional table.

4. For columns of domain Value only

MinMax functions

MIN(*a*)	calculates the minimum value of all records that are the outcome of expression *a*, or column *a*
MAX(*a*)	calculates the maximum value of all records that are the outcome of expression *a*, or column *a*

Average function

AVG(*a*)	returns the average value of all records that are the outcome of expression *a*, column *a*, etc.

Conversion function

STRING(*a*)	returns value *a* as a string

5. For columns of domain Class, ID, Group, or String only

Special operators and functions

+	*s1* + *s2*; concatenation operator, glues *s1* and *s2* together
IN	*s1* IN *s2*; tests whether *s1* is part of *s2*
LENGTH(*s*)	returns the number of characters (the length) of *s*

LEFT(*s*,*int*)	returns the first int number of characters of *s*
RIGHT(*s*,*int*)	returns the last int number of characters of *s*
SUB(*s*,*int1*,*int2*)	returns a substring of *s*; starts with character at position *int1* and returns *int2* number of characters
STRLT(*s1*,*s2*)	returns true is if in alphabetical order string *s2* comes after string *s1*. Can also be used in the form of *s1* < *s2*.
STRLE(*s1*,*s2*)	returns true is if in alphabetical order string *s2* comes after or on the same place as string *s1*. Can also be used in the form of *s1* <= *s2*.
STRGT(*s1*,*s2*)	returns true is if in alphabetical order string *s2* comes before string *s1*. Can also be used in the form of *s1* > *s2*.
STRGE(*s1*,*s2*)	returns true is if in alphabetical order string *s2* comes before or on the same place as string *s1*. Can also be used in the form of *s1* >= *s2*.

Conversion function

VALUE(*s*)	converts *s* to a value

6. Column aggregations

AGGAVG(*col*)	calculates the average value of *col*
AGGAVG(*col*,*k*)	calculates the average value of *col* per group *k*
AGGAVG(*col*,*k*,*w*)	calculates the average value of *col* per group *k* using weights *w*
AGGCNT(*col*)	counts the number of times that *col* is true
AGGCNT(*col*,*k*)	counts the number of times that *col* is true per group *k*
AGGMIN(*col)*	determines the minimum value of *col*
AGGMIN(*col*,*k*)	determines the minimum value of *col* per group *k*
AGGMED(*col*)	calculates the median value of *col*
AGGMED(*col*,*k*)	calculates the median value of *col* per group *k*
AGGMED(*col*,*k*,*w*)	calculates the median value of *col* per group *k* using weights *w*
AGGMAX(*col*)	determines the maximum value of *col*
AGGMAX(*col*,*k*)	determines the maximum value of *col* per group *k*
AGGPRD(*col*)	determines the predominant value of *col*
AGGPRD(*col*,*k*)	determines the predominant value of *col* per group *k*
AGGPRD(*col*,*k*,*w*)	determines the predominant value of *col* per group *k* using weights *w*
AGGSTD(*col*)	calculates the standard deviation of *col*
AGGSTD(*col*,*k*)	calculates the standard deviation of *col* per group *k*
AGGSTD(*col*,*k*,*w*)	calculates the standard deviation of *col* per group *k* using weights *w*
AGGSUM(*col*)	calculates the sum of *col*
AGGSUM(*col*,*k*)	calculates the sum of *col* per group *k*

7. Predefined values and variables

PI	Value π :	3.141592653589...
PI2	Value 2 * π :	6.283185...
PIDIV2	Value ½ * π:	1.570796...
PIDIV4	Value¼ * π:	0.785398...
EXP(1)	Value e :	2.718281828479...

Predefined variables in MapCalc

%X	variable to calculate with X-coordinates in a map
%Y	variable to calculate with Y-coordinates in a map
%L	variable to calculate with Line or Row numbers in a map
%C	variable to calculate with Column numbers in a map

Predefined variables in TabCalc

%R	variable to calculate with record numbers in a table according to the order in which the items appear in the domain of the table (R for record).
%K	variable to calculate with class names or IDs or Groups in a table as present in the domain of the table (K for key).

Col[expr]
Table.Col[expr]
Table.ext.Col[expr]

8. Special calculations on coordinates, point data, properties and colors

Calc with coords	MAPCRD	PNTCRD	CRDX	CRDY
	COORD	MINCRDX	MAXCRDX	MINCRDY
	MAXCRDY	TRANSFORM	MAPVALUE	MAPROW
	DIST	DIST2	MAPCOL	
		RASVALUE		
Calc with points	PNTNR	PNTVAL	PNTCRD	
Calc with properties	MAPMIN	MAPMAX	PIXSIZE	PIXAREA
	MAPROWS	MAPCOLS		
Calc with colors	CLRRED	CLRGREEN	CLRBLUE	CLRGREY
	CLRYELLOW	CLRMAGENTA	CLRCYAN	CLRHUE
	CLRSAT	CLRINTENS	MAPCOLOR	COLOR
	RPRCOLOR	COLORHSI		

Appendix E

Commands and expressions (introduction)

ILWIS operations can be started through a menu command, by double-clicking an operation icon in the Operation-list or by typing a command or an expression on the command line in the Main window.

By typing an ILWIS command (optional parameters), a dialog box is displayed. By typing an ILWIS expression (with all parameters), the operation's dialog box is skipped (all parameters for the operation are automatically filled out), but an operation's definition dialog box is displayed. In the future, a complete ILWIS operation can be performed.

E.1 ILWIS commands

By typing an ILWIS command on the command line of the main window, usually a dialog box is opened. Before the command can be carried out, you have to supply additional information in the appearing dialog box, and click its OK button.

You can for instance obtain dialog boxes to display objects, edit objects, view the properties of objects, or create objects. You can also obtain the dialog box of an operation. When a dialog box is used for an operation, the result of the operation is always a dependent object.
Commands typed on the command line are not case sensitive.

ILWIS Commands contains three sections:
1. Open, edit, view properties or create an object.
2. Operations.
3. Data management: copy, delete, etc.

Open, edit, view properties or create an object

Several commands can be typed on the command line of the Main window to open, edit, or view the properties of an object. To open an object, it is generally faster though to double-click an object in the Catalog. To edit, view object properties, open an object as a table or to create objects you can use context-sensitive menu in the Catalog. For more information, see How to use the Catalog.

To open, edit or view the properties of an ILWIS object

`Open, Show, Map, Pol, Seg, Pnt, Tbl, Mpl, View`; Opens a Show dialog box in which you can select the object you want to display.
`Edit`; Opens the Edit Object dialog box in which you can select the object you want to edit.
`Prop`; Opens the View Properties of dialog box in which you can select the object which properties you want to view.

To directly open, edit, or view object properties:
The `Open, Show, Edit` and `Prop` commands may be followed by an object name with its extension, then the Show/Edit Object/View Properties of dialog boxes in which you can select an object are skipped and the object is shown immediately.

`Open` *objname.ext*	Opens object *objname.ext* immediately. Opens a Display Options dialog box in case of maps, opens a table window displaying the table, the two-dimensional table or the histogram, or shows the object otherwise immediately; for details, see next paragraphs.
`Show` *objname.ext*	Same as Open.
`Edit` *objname.ext*	Starts the appropriate editor or opens a dialog box in which you can edit the object.
`Prop` *objname.ext*	Show the Properties dialog box of the object. For more information, see Basic concepts : properties of objects.

To directly open specific objects, you can further use the following commands (no extensions required):

Map *Rasmapname*	Opens the Display Options dialog box for the raster map.
`Pol` *Polmapname*	Opens the Display Options - Polygon Map dialog box.
`Seg` *Segmapname*	Opens the Display Options - Segment Map dialog box.
`Pnt` *Pointmapname*	Opens the Display Options- Point Map dialog box.
`Mpl` *MapListname*	First opens the Display Options dialog box for a raster map in the map list; then opens the Display Options - Map List dialog box (slide show).
`Tbl` *Tablename*	Displays the table in a table window.
`View` *MapViewname*	Displays the map view in a map window.
`Dom` *Domname*	Opens the Domain Class/ID editor in case a of domain class, ID, Group or Picture, or opens the Edit Domain Value dialog box in case of an editable value domain.
`Rpr` *Reprname*	Opens the Representation Class editor in case of a representation Class or Picture, or opens the representation Value/Gradual editor in case of editable value or gradual representations.
`CopyMap`	Alias for the Map command.
`TabCalc`	Alias for the Tbl command.

To show objects as tables (advanced):

`ShowAsTbl`	Opens the Show As Table dialog box which allows you to select any object that can be opened as a table. For more information, see also How to open objects as a table.
`OpenAsTbl`	Alias for ShowAsTbl.
`ShowAsTbl` *pntmap.mpp*	Opens the point map *pntmap* immediately as a table.
`ShowAsTbl` *domain.dom*	Opens the domain *domain* immediately as a table.
`ShowAsTbl` *repres.rpr*	Opens the representation *repres* immediately as a table.
`ShowAsTbl` *georef.grf*	Opens the georeference tiepoints *georef* immediately as a table.

To show internal domains and representations (advanced):

`ShowAsDom` *objname.ext*	Opens the internal domain of object *objname.ext*.
`OpenAsDom`	Alias for `ShowAsDom`.
`ShowAsRpr` *objname.ext*	Opens the internal representation of object *objname.ext*.
`OpenAsRpr`	Alias for `ShowAsRpr`.

To create an ILWIS object

`Create`	Opens a popup menu in which you can choose the type of object you want to create.

The `Create` command may be followed by an object abbreviation:

`Create map`	Opens the `Create Raster Map` dialog box.
`Create pol`	Opens the `Create Polygon Map` dialog box.
`Create seg`	Opens the `Create Segment Map` dialog box.
`Create pnt`	Opens the `Create Point Map` dialog box.
`Create mpl`	Opens the `Create Map List` dialog box.
`Create tbl`	Opens the `Create Table` dialog box.
`Create dom`	Opens the `Create Domain` dialog box.
`Create rpr`	Opens the `Create Representation` dialog box.
`Create grf`	Opens the `Create Georeference` dialog box.
`Create csy`	Opens the `Create Coordinate System` dialog box.
`Create sms`	Opens the `Create Sample Set` dialog box.
`Create tb2`	Opens the `Create Two-Dimensional Table` dialog box.
`Create fil`	Opens the `Create Filter` dialog box.
`Create fun`	Opens the `Create Function` dialog box.
Create `isl`	Opens the `Create Script` dialog box ('isl' is an abbreviation for ILWIS Script Language).

To open the pixel info window

`Pixelinfo`	Opens the pixel info window.

To run a script

`Run` *scriptname var1 var2*	Runs the script *script name* with variables *var1* and *var2*. Variables are optional. By using the `Run` command, you can continue working with ILWIS while the script is running. For more information, see How to run scripts.

To obtain help

`Help`	Opens ILWIS.HLP on the Contents page.
`Help` *SearchString*	Opens the Search dialog box with the specified search string. For example, `help classify` shows the Search dialog box with the topics of 'classify'.
`PopupHelp` *.ext*	Opens ILWIS.HLP and shows the popup for the extension. For example `popuphelp .dom` shows a help popup on domains.

To start an operation

To obtain the dialog box of an ILWIS operation, type one of the following commands or an alias. An alias is listed when the command is not the same as the ILWIS 1.4 executable name. Aliases generally represent ILWIS 1.4 executable names.
To obtain the dialog box of an operation, you can also double-click the operation in the Operation-list or select it from the Operations menu.

Operation	**command**	**alias(es)**
Visualization		
Show Map	`Show`	`Open, Map, CopyMap`
Show Table	`Tbl`	`TabCalc`
Color Composite	`ColorComp`	`ColorCmp`
Display 3D	`Display3D`	
Apply 3D	`Apply3D`	
Slide Show	`Mpl`	
Raster Operations		
Map Calculate	`MapCalc`	`MCalc`
Attribute Map	`AttribRas`	
Cross	`Cross`	
Aggregate Map	`Aggregate`	
Distance	`Distance`	
Iteration	`MapIter`	`Iter, Iterp`
Area Numbering	`AreaNumb`	
Sub Map	`SubRas`	`SubMap`
Glue Maps	`Glue`	`GlueRas`
Mirror Rotate	`Mirror`	`MirrorRotate`

Image Processing		
Filter	`Filter`	`FiltrMap`
Stretch	`Stretch`	
Slicing	`Slicing`	`Clfy`
Color Separation	`ColorSep`	
Cluster	`Cluster`	
Sample	`Sample`	`Smpl`
Classify	`Classify`	
Resample	`Resample`	`GeoCorr`
Statistics		
Histogram	`Histogram`	`Histgram`
Raster		
Autocorrelation	`Autocorr`	
MapList		
Principal Components	`PrincCmp`	
Factor Analysis	`FactAnal`	
Variance Covariance	`MatVarCov`	
Correlation Matrix	`MatCorr`	
Polygons		
Neighbour Polygons	`HistNbPol`	
Segments		
Direction Histogram	`HistSegDir`	`LinAnal`
Points		
Spatial correlation	`SpatCorr`	
Pattern Analysis	`PattAnal`	
Interpolation		
Densify map	`DensRas`	`Densify, InterpolRas`
Contour Interpolation	`InterpolSeg`	`Interpol`
Point Interpolation		
Nearest Point	`NearestPnt`	
Trend Surface	`TrendSurface`	
Moving Average	`MovAverage`	
Moving Surface	`MovSurface`	`Gridding`
Vector Operations		
Unique ID	`UniqueID`	
Polygons		
Attribute Map	`AttribPol`	
Mask Polygons	`MaskPol`	
Assign Labels	`LabelPol`	
Transform Polygons	`TransfPol`	

Segments		
Attribute Map	`AttribSeg`	
Mask Segments	`MaskSeg`	`CopySeg`
Assign Labels	`LabelSeg`	
Sub Map	`SubSeg`	
Glue Maps	`GlueSeg`	
Densify Coordinates	`DensSeg`	
Transform Segments	`TransfSeg`	
Tunneling	`TunnelSeg`	
Points		
Attribute Map	`AttribPnt`	
Mask Points	`MaskPnt`	
SubMap	`SubPnt`	
Glue Maps	`GluePnt`	
Transform Points	`TransfPnt`	
Transform Coordinates	`Transform`	
Rasterize		
Polygon to Raster	`PolRas`	
Segment to Raster	`SegRas`	
Segment Density	`SegDensity`	`Ld`
Point to Raster	`PntRas`	
Point Density	`PntDensity`	
Vectorize		
Raster to Polygon	`RasPol`	`RasVec`
Raster to Segment	`RasSeg`	
Raster to Point	`RasPnt`	
Polygon to Segment	`PolSeg`	
Polygon to Point	`PolPnt`	
Segment to Polygon	`SegPol`	`Polygonize`
Segment to Point	`SegPnt`	
Table Operations		
Transpose Table	`Transpose`	
Other		
Import from 1.4	`Import14`	
Import	`Import`	
Export to 1.4	`Export14`	
Export	`Export`	
Convert14	`Convert14`	

Data Management

To copy or delete object, you can also use context-sensitive menu of an object that is clicked with the right mouse button in the Catalog. For more information, see How to use the Catalog.

`Copy`	Opens the first `Copy Object` dialog box in which you can select an object to be copied.
`Del`	Opens the `Delete Object` dialog box in which you can select an object to be deleted.

The Copy and Del commands may be followed by an object name with its extension:

`Copy` *objname.ext*	Opens the second `Copy Object` dialog box in which the input object name filled out: specify a name for the output object or specify a name of an existing directory.
`Del` *objname.ext*	Deletes object *objname.ext*.

Other

`cd\`	Goes to the root of the current drive.
`cd\`dir	Goes to the (sub)directory.
`cd..`	Goes one directory up.
`cd` ..	Goes one directory up.
`cd` *subdir*	Goes to the subdirectory.
`md`	Gives a dialog box to create a new subdirectory.
`md` *newdir*	Creates a new subdirectory.
`mkdir` *newdir*	Creates a new subdirectory.
`rd`	Gives a dialog box to remove a subdirectory.
`rd` *subdir*	Removes the subdirectory.
`rmdir` *subdir*	Removes the subdirectory.
`a:`	Goes to the specified drive.
`exit`	Leaves ILWIS.
`?`*expression*	Pocket line calculator, performs calculations with constant values.
`!`*expression*	Acts as the Windows Run command to start any Windows application program, batch file, or DOS application with a Windows .PIF file, etc. Applications that can be started from the command line may have the following extensions: .exe, .com, .bat, .pif. Type the application name directly after the exclamation mark (no spaces allowed). Example: to start Word, type: `!Winword`
`Script` *expression*	Executes the expression as if running a script. By using the `Script` command followed by any expression that is allowed in a script (see ILWIS script language), you cannot continue working with ILWIS while the expression is executed.
See also:	ILWIS expressions; ILWIS script language (syntax)

E.2.1 ILWIS expressions

Any ILWIS operation like Filter, Cross and Distance calculation, can be performed by typing an ILWIS expression on the command line of the Main window. You can also use these expressions in scripts. In this topic, the syntax of expressions of operations is described. For an overview of MapCalc and TabCalc expressions, see Appendices : ILWIS operators and functions (MapCalc/TabCalc). For details on creating expressions on the command line and in scripts, see Appendices : construction of expressions. For special script commands, see Appendices : ILWIS script language (syntax).

Introduction

The general syntax for expressions is:

```
OUTMAP = expression
OUTMAP := expression
```

The definition symbol (=) is used to create a dependent output object; the assignment symbol (:=) is used to create an editable object.

In the overview below:

- Arial Bold is used for menu commands;
- `Courier` is used for obliged parts in expressions or in parameters;
- *Italics* is used for parameters with special requirements, usually a short explanation follows;
- parameter 'rasmap' means an input raster map; 'maplist' means an input map list (set of raster maps with same domain and same georeference); 'polmap' means an input polygon map; 'segmap' means an input segment map; 'pntmap' means an input point map; 'table' means an input table; 'column' means an input column;
- parameter 'domain' means an existing domain; the domain will be used for the output object; 'georef' means an existing georeference except georef None; the georeference will be used for the output raster map; 'coordsys' means any existing coordinate system; 'sample set' means an input sample set; 'newdomain' means the output domain that will be created by the expression; 'newgeoref' means the output georeference that will be created by the expression;
- a vertical bar | represents a choice; a parameter in square brackets [ ] represents an optional parameter;
- the phrase 'value map' or 'map with a value domain' means that the map should have domain of type Value.

Any operation name in the list below starting with:

- `Map` creates an output raster map;
- `PolygonMap` creates an output polygon map;
- `SegmentMap` creates an output segment map;
- `PointMa`p creates an output point map;
- `Table` creates an output table;
- `Matrix` creates an output matrix.

☞ Some operations need a value input map. When your raster map is of domain type Class, ID or Group, and an attribute table is linked to the map with one or more suitable value columns, then you may type 'map' on the command line instead of parameter 'map' listed below. The operation then directly uses the values of the attribute column.

List of ILWIS expressions

The list below follows the order of the Operations menu in the Main window.

VISUALIZATION

Color Composite

`MapHeckbert`(*map list, nr of colors*)
`MapColorComp[Linear]`(*map list, range1, range2, range3*)
`MapColorCompHistEq` (*map list, range1, range2, range3*)
`MapColorComp24[Linear]`*(map list* [*, range1, range2, range3*])
`MapColorComp24HistEq` (*map list, range1, range2, range3*)
`MapColorComp24HSI` (*map list*)

map list	Map list which contains three raster maps of domain Image.
range1,2,3	min:max|perc
perc	Real value >= 0
nr of colors	2 > integer value < 230

Apply 3D

`MapApply3D`(*rasmap, georef3D*)

rasmap	Input raster map cannot have georef `None`.
georef3D	A georeference 3D.

RASTER OPERATIONS

Map Calculate

expression	See MapCalc and TabCalc.

Attribute Map

`MapAttribute`(*rasmap, column*) | ***Rasmap.column***

rasmap	Raster map with a Class, ID, or Group domain.
column	Column with Value, Class, ID, Group, Picture, or Color domain; by default a column from the attribute table of the raster map.

Cross

`MapCross`(*rasmap1, rasmap2*, output cross table)
`TableCross`(*rasmap1, rasmap2*)
`TableCross`(*rasmap1, rasmap2*, output cross rasmap)

rasmap2	Same georeference as input raster map1.

Aggregate Map

`MapAggregate`*`AggFunc`*(*rasmap, groupfactor*,*`group`* [,*rowoffset, coloffset*] [,newgeoref])
`MapAggregate`*`AggFunc`*(*rasmap, groupfactor*,*`nogroup`* [,*rowoffset, coloffset*])

`AggFunc`	`avg| cnt| max| med| min| prd| std| sum` When no aggregation function is specified, the upper left pixel of each block is used.
rasmap	Raster map with a value domain for aggregate functions `avg, max, min, std, sum`; raster map with a class, ID, or value domain for aggregate function med ; raster map with any domain for aggregate function `prd`.
groupfactor	A value (>= 1) to define the size of the blocks of input pixels to be aggregated; a value of 4 means that each block of 4 x 4 input pixels will be aggregated.
`group`	Each block of input pixels is aggregated to 1 output pixel; a georeference factor is created with the same name as the output map.
`nogroup`	Each block of input pixels is aggregated and the output value is stored in all pixels of the block that correspond to the considered block of input pixels; the output map uses the same georeference as the input map.
rowoffset	Optional parameter to start the aggregation from this row onward.
coloffset	Optional parameter to start the aggregation from this column onward.
newgeoref	When using option group, optional parameter to specify the name of the output georeference; if not specified, the output georeference obtains the same name as the output map.

Distance

`MapDistance`(*source rasmap* [,*weight rasmap*| `1`])
`MapDistance`(*source rasmap*, [*weight rasmap*| `1`], *output rasmap Thiessen*)
`MapThiessen`(*source rasmap*[, *weight rasmap*| `1`], *output rasmap Distance*)

source rasmap	Input raster map of any domain type; for all pixels with the undefined value, a distance value is calculated.
[*weight rasmap*| `1`]	Weight map is an optional parameter to specify a map with weight factors; raster map of domain type Value. When a 1 is specified or when the parameter is not used, weight factor 1 is used for all pixels.
output rasmap Thiessen	Name of output Thiessen raster map.
output rasmap Distance	Name of output Distance raster map.

Iteration

`MapIter`(*StartMap, IterExpr* [, *nr of iterations*])
`MapIterProp`(*StartMap, IterExpr* [, *nr of iterations*])

StartMap	Raster map that is used in the *IterExpr*.
IterExpr	An expression for neighbourhood operations.
nr of iterations	Optional parameter to specify the maximum number of iterations; if not specified, the operation continues until no more changes occur.

Area Numbering

`MapAreaNumbering`(rasmap, *`8|4`* [, *newdomain*])

`8|4`	Distinguish 8-connected or 4-connected areas; default is 8.
newdomain	Optional parameter to specify a name for the output ID domain; if not specified, the output domain will be stored by the output raster map (internal domain).

Sub Map

`MapSubMap`(rasmap, first row, first col [, nr rows, nr cols] [, *newgeoref*])

newgeoref	Optional parameter to specify a name for the output georeference; if not specified, then the output georeference obtains the same name as the output map.

Glue Maps

MapGlue(*rasmap1*, *rasmap2* [, *newdomain*] [, *`replace`*])

rasmap1	Input raster map which georeference will be used for the output raster map.
rasmap2	Input raster map that will be resampled if needed.
newdomain	Optional parameter, when merging two class or two ID maps, to specify the name of the output domain into which all items of the two input domains are merged; if not specified, the output domain will be stored by the output raster map (internal domain).
`replace`	Optional parameter to use the values of rasmap2 for overlapping pixels; if not specified, the values of rasmap1 are used for overlapping pixels.

Mirror Rotate

`MapMirrorRotate`(rasmap, *rotate type*)

rotate type	`mirrhor|mirrvert|mirrdiag|transpose|rotate90|rotate180|rotate270|normal`

IMAGE PROCESSING

Filter

`MapFilter`(*rasmap, filter*)

rasmap	All filters use input raster maps with a value domain; the Majority and the UndefMajority filters also work on other domain types.
filter	*filter on disk*| `Average`(*rows,cols*) | `RankOrder`(*rows,cols,rank*[,*threshold*]) | `Median`(*rows,cols*[,*threshold*]) | `Majority`(*rows,cols*) | `ZeroMajority`(*rows,cols*) | `UndefMajority`(*rows,cols*) | `Pattern`(*threshold*) | `FilterStandardDev`(*rows,cols*) | `FilterLinear`(*rows,cols,expression*)
filter on disk	User-defined linear filter or a system filter.
rows,cols	Size of filter in rows and columns; value >= 1; maximum size of filter (rows*cols) <= 8000.
threshold	If the difference between the resulting value and the original pixel value is smaller than or equal to the threshold, the calculated value is used. If the difference between the resulting value and the original pixel value is larger than the threshold, the original pixel value is retained.
rank	The rank number of which the pixel value is assigned to the central pixel.
expression	Fill the values in the filter by an expression in which you can use the parameters `x`, `y`, and `r`.

Stretch

`MapStretch`[Linear](*rasmap, range from, domain*)

`MapStretch`[Linear](*rasmap, range from, domain, range to*)

`MapStretchHistEq`(*rasmap, range from, intervals*)

rasmap	Input raster map using a value domain.
range from	Min:max | *perc.*
perc	Real value > 0.
intervals	Number of output intervals.
domain	Output value domain.
range to	Value range of output map *as min:max* | *min:max:prec*| *::prec.*

Slicing

`MapSlicing`(*rasmap*, domain group)

rasmap	Input raster map using a value domain.

Color Separation

`MapColorSep[aration]`(*rasmap, color*)| *rasmap.color*

rasmap Input raster map using a Picture domain or the `Color` domain.

color `red|green|blue|yellow|magenta|cyan|grey|gray|hue|saturation|sat|intensity|intens`

Cluster

`MapCluster`(*map list, nr of clusters*)

map list A map list which contains 1, 2, 3, or 4 raster maps that use the Image domain.

nr of clusters An integer value between 2 and 60 for the number of clusters in the output map.

Classify

`MapClassify`(sample set, *classifier*)

classifier `ClassifierBox`(factor) |
`ClassifierMinDist`([threshold]) |
`ClassifierMinMahaDist`([threshold]) |
`ClassifierMaxLikelihood`([threshold])

Resample

`MapResample`(rasmap, georef, *resamp meth*[, `Patch| NoPatch`])

resamp meth `NearestNeighbour| BiLinear| BiCubic`

STATISTICS

Histogram

`TableHistogram`(*rasmap*)

rasmap Input raster map of any domain type. Mind: the output raster histogram table will always have the same name as the input raster map.

`TableHistogramPnt`(*pntmap*)

pntmap Input point map of any domain type. Mind: the output point histogram table will always have the same name as the input point map.

`TableHistogramPol`(*polmap*)

polmap Input polygon map of any domain type. Mind: the output polygon histogram table will always have the same name as the input polygon map.

`TableHistogramSeg`(*segmap*)

segmap Input segment map of any domain type. Mind: the output segment histogram table will always have the same name as the input segment map.

RASTER
Autocorrelation
`TableAutoCorrSemiVar`(rasmap, *max shift*)
max shift Maximum pixel shift; integer value > 0.

MAP LIST
Principal Components
`MatrixPrincComp`(*map list*)
map list Map list containing raster maps with a value domain.

Factor Analysis
`MatrixFactorAnal`(*map list*)
map list Map list containing raster maps with a value domain.

POLYGONS
Neighbour Polygons
`TableNeighbourPol`(*polmap*)
polmap Iput polygon map with a Class, ID or Group domain.

SEGMENTS
Direction Histogram
`TableSegDir`(segmap)

POINTS
Spatial correlation
`TableSpatCorr`(*pntmap*)
pntmap Input point map with a value domain.

Pattern Analysis
`TablePattAnal`(*pntmap*)
pntmap Input point map with more than two points.

INTERPOLATION

Densify map
`MapDensify`*(rasmap, factor, interpol meth*)
rasmap Raster map with a value domain for a BiLinear or Bicubic interpolation; raster map with any domain for Nearest Neighbour.
factor Real value > 1.
interpol meth `BiLinear|BiCubic|NearestNeighbour`

Contour Interpolation
`MapInterpolContour`(*segmap*, georef)
segmap Input segment map with a value domain.
`MapInterpolContour`(*rasmap*)
rasmap Input raster map with a value domain; mind: algorithm only works well for rasterized contour lines.

POINT INTERPOLATION

Nearest Point

`MapNearestPoint`(pntmap, georef)

Trend Surface

`MapTrendSurface`(*pntmap*, georef, *surface type*)

pntmap	Input point map with a value domain.
surface type	`Plane| Linear2| Parabolic2| 2| 3| 4| 5| 6`

Moving Average

`MapMovingAverage`(*pntmap*, georef, *weight func*)

pntmap	Input point map with a value domain.
weight func	`InvDist`(*Exp*,*LimDist*) | `Linear`(*Exp*,*LimDist*)
Exp	Weight exponent.
LimDist	Limiting distance.

Moving Surface

`MapMovingSurface`(*pntmap*, georef, *surface type*, *weight func*)

pntmap	input point map with a value domain
surface type	`Plane| Linear2| Parabolic2| 2| 3| 4| 5| 6`
weight func	`InvDist`(*Exp*,*LimDist*) | `Linear`(*Exp*,*LimDist*)
Exp	Weight exponent.
LimDist	Limiting distance.

VECTOR OPERATIONS

Unique ID

`PolygonMapNumbering`(polmap [, *newdomain*])
`SegmentMapNumbering`(segmap [, *newdomain*])
`PointMapNumbering`(pntmap [, *newdomain*])

newdomain	Optional parameter to specify a name for the output ID domain; if not specified, the output domain will be stored by the output map (internal domain).

POLYGONS

Attribute Map

`PolygonMapAttribute`(*polmap*, *column*)

polmap	Polygon map with a Class, ID, or Group domain.
column	Column with a Value, Class, ID, or Group domain; by default a column from the attribute table of the polygon map.

Mask Polygons

`PolygonMapMask`(polmap, *"mask")*

"mask"	A mask consists of (multiple) search strings; asterisks and question marks can be used as wild cards; on the command line, the total mask needs to be surrounded by double quotes.

Assign Labels
`PolygonMapLabels`(polmap, pntmap)

Transform Polygons
`PolygonMapTransform`(polmap, coord sys)

SEGMENTS
Attribute Map
`SegmentMapAttribute`(*segmap, column*)

segmap	Segment map with a Class, ID, or Group domain.
column	Column with a Value, Class, ID, or Group domain; by default a column from the attribute table of the segment map.

Mask Segments
`SegmentMapMask`(segmap, *"mask")*

"mask"	A mask consists of (multiple) search strings; asterisks and question marks can be used as wild cards; on the command line, the total mask needs to be surrounded by double quotes.

Assign Labels
`SegmentMapLabels`(segmap, pntmap [, *set domain*])

set domain	`yes` | `no` ; optional parameter to set the domain of the output segment map to the domain of the input segment map or to the domain of the input point map; if this parameter is not specified, the output segment map will use the same domain as the input segment map.
`no`	Domain of output segment map is domain of input segment map.
`yes`	Domain of output segment map is domain of input point map.

Sub Map
`SegmentMapSubMap`(segmap, *minX, minY, maxX, maxY*)

minX	Minimum X-coordinate of output map.
minY	Minimum Y-coordinate of output map.
maxX	Maximum X-coordinate of output map.
maxY	Maximum Y-coordinate of output map.

Glue Segment Maps
SegmentMapGlue(*segmap1, "mask1", segmap2, "mask2"* [,...] [, *newdomain*])
SegmentMapGlue(*minX, minY, maxX, maxY, segmap1,"mask1", segmap2, "mask2"* [, ...] [, *newdomain*])

segmap1,2, ...	Are the input segment map names to be combined into one output segment map.

"mask1"	A mask consists of (multiple) search strings; asterisks and question marks can be used as wild cards; on the command line, the total mask needs to be surrounded by double quotes.
minX	Minimum X-coordinate of output map.
minY	Minimum Y-coordinate of output map.
maxX	Maximum X-coordinate of output map.
maxY	Maximum Y- coordinate of output map.
newdomain	Optional parameter, when merging Class or ID maps, to specify a name for the output domain into which all items of the input domains are merged; if not specified, the output domain will be stored by the output map (internal domain).

Densify Coords

`SegmentMapDensifyCoords`(segmap, *distance*)

distance	Is the distance in meters at which extra intermediate points should be inserted into segments; real > 0.

Tunneling

`SegmentMapTunneling`(segmap, *tunnel width*, *remove node*)

tunnel width	Tunnel width in meters; real value >= 0.
remove node	Yes| no ; remove superfluous nodes or not.

Transform Segments

`SegmentMapTransform`(segmap, coord sys)

POINTS

Attribute Map

`PointMapAttribute`(*pntmap, column*)

pntmap	Point map with a Class, ID, or Group domain.
column	Column with a Value, Class, ID, or Group domain; by default a column from the attribute table of the point map.

Mask Points

`PointMapMask`(pntmap, *"mask"*)

"mask"	A mask consists of (multiple) search strings; asterisks and question marks can be used as wild cards; on the command line, the total mask needs to be surrounded by double quotes.

Sub Map

`PointMapSubMap`(pntmap, *minX, minY, maxX, maxY*)

minX	Minimum X-coordinate of output map.
minY	Minimum Y-coordinate of output map.
maxX	Maximum X-coordinate of output map.
maxY	Maximum Y-coordinate of output map.

Glue Point Maps

`PointMapGlue`(*pntmap1, "mask1",pntmap2, "mask2"* [,...] [, *newdomain*])
`PointMapGlue`(*minX, minY, maxX, maxY, pntmap1,"mask1", pntmap2, "mask2"* [, ...] [, *newdomain*])

pntmap1,2, ...	Are the input point maps to be combined into one output point map.
"mask1"	A mask consists of (multiple) search strings; asterisks and question marks can be used as wild cards; on the command line, the total mask needs to be surrounded by double quotes.
minX	Minimum X-coordinate of output map.
minY	Minimum Y-coordinate of output map.
maxX	Maximum X-coordinate of output map.
maxY	Maximum Y- coordinate of output map.
newdomain	Optional parameter, when merging Class or ID maps, to specify a name for the output domain into which all items of the input domains are merged; if not specified, the output domain will be stored by the output map (internal domain).

Transform Points

`PointMapTransform`(pntmap, coord sys)

RASTERIZE

Polygon to Raster

`MapRasterizePolygon`(polmap, georef)

Segment to Raster

`MapRasterizeSegment`(segmap, georef)

Segment Density

`MapSegmentDensity`(segmap, *"mask"*, georef)

"mask"	A mask consists of (multiple) search strings; asterisks and question marks can be used as wild cards; on the command line, the total mask needs to be surrounded by double quotes.

Point to Raster

`MapRasterizePoint`(pntmap, georef, *point size*)

point size	Size in pixels; integer value > 0.

Point Density

`MapRasterizePointCount`(pntmap, georef, *point size*)
`MapRasterizePointSum`(pntmap, georef, *point size*)

point size	Size in pixels; integer value > 0.

VECTORIZE

Raster to Polygon

`PolygonMapFromRas`(*rasmap* [, *8|4* [, *smooth|nosmooth*]])

rasmap	Input raster map cannot have georef `None`.
8|4	Distinguish 8-connected or 4-connected areas; default `8`.
smooth	Smooth polygon boundaries; default.
nosmooth	Do not smooth polygon boundaries.

Raster to Segment

`SegmentMapFromRasAreaBnd`(*rasmap, 8|4, smooth|nosmooth, single|composite*)

rasmap	Input raster map cannot have georef `None`.
8|4	Distinguish 8-connected or 4-connected areas.
smooth	Smooth segments.
nosmooth	Do not smooth segments.
single	Assign the name 'Segments' to all output segments (internal output domain).
composite	Use the names of the pixels on both sides of the output segment and construct composite names for the output segment like `Agri|Forest` (internal output domain).

Raster to Point

`PointMapFromRas`(*rasmap*)

rasmap	Input raster map cannot have georef `None.`

Polygon to Segment

`SegmentMapPolBoundaries`(polmap, *"mask"*, *single|composite*)

"mask"	A mask consists of (multiple) search strings; asterisks and question marks can be used as wild cards; on the command line, the total mask needs to be surrounded by double quotes.
single	Assign the name 'Segments' to all output segments (internal output domain).
composite	Use the names of the polygons on both sides of each output segment and construct composite names for the segments like Agri|Forest (internal output domain).

Polygon to Point

`PointMapPolLabels`(polmap)

Segment to Polygon

`PolygonMapFromSegment`(segmap [, *"mask"* [, `auto`]])
`PolygonMapFromSegment`(segmap [, *"mask"*, *domain*|*labelpntmap* [, `auto`])

"mask"	A mask consists of (multiple) search strings; asterisks and question marks can be used as wild cards; on the command line, the total mask needs to be surrounded by double quotes; if not specified all segments are used.
domain	Optional parameter to polygonize the segments to an existing domain; after polygonization, you can break the dependencies and edit the polygon map.
labelpntmap	Optional parameter to polygonize the segments and use a point map with label points to assign names to the output polygons. If both the *domain* and *labelpntmap* parameters are not specified, the segments are polygonized and are assigned default names such as Pol 1, Pol 2, etc. (internal domain)
`auto`	Optional parameter to automatically correct segments; deletes false polygons, deletes segments with dead ends, insert nodes when needed; if not specified and an error is found, the program stops and no polygon map is calculated.

Segment to Point

`PointMapSegCoords`(segmap)
`PointMapSegDist`(segmap, *distance*)

distance	Distance interval in meters; real value > 0.

`PointMapSegNodes`(segmap)

TABLE OPERATIONS

Transpose Table

`TableTranspose`(*table*, *col domain*)
`TableTranspose`(*table*, *col domain*, *valuerange*)

table	Input table with a domain none, class or ID and maximum 1000 records.
col domain	The domain that will be used for all columns in the transposed table.
valuerange	If the *column domain* is a value domain, specify the value range as *min:max : prec* that will be used for all columns in the transposed table.

E.2.2 ILWIS expressions (alphabetic)

In the listing below, all ILWIS expressions are ordered by output object type, and then alphabetical. For more information, refer to Help topic Appendices : ILWIS expressions.

Operations resulting in a raster map

`MapAggregate`*`AggFunc`*(rasmap, *groupfactor*, `group` [,*rowoffset*, *coloffset*] [,*newgeoref*])
`MapAggregate`*`AggFunc`*(rasmap, *groupfactor*, `nogroup` [,*rowoffset*, *coloffset*])
`MapApply3D`(rasmap, georef3D)
`MapAreaNumbering`(rasmap, `8|4` [, *newdomain*])
`MapAttribute`(rasmap, attribute column)
`MapCalculate`(expression)
`MapClassify`(sample set, *classifier*)
`MapCluster`(*map list*, *nr of clusters*)
`MapColorComp24[Linear]`(*map list* [,*range1*, *range2*, *range3*])
`MapColorComp24HistEq`(*map list*, *range1*, *range2*, *range3*)
`MapColorComp24HSI`(*map list*)
`MapColorComp[Linear]`(*map list*, *range1*, *range2*, *range3*)
`MapColorCompHistEq`(*map list*, *range1*, *range2*, *range3*)
`MapColorSep[aration]`(*rasmap*,*color*)
`MapCross`(rasmap1, rasmap2, output cross table)
`MapDensify`(rasmap, factor, interpolation method)
`MapDistance`(source rasmap [, weight rasmap`|1` [, output Thiessen rasmap]])
`MapFilter`(rasmap, filter)
`MapGlue`(rasmap1, rasmap2 [,*newdomain*] [, `replace`])
`MapHeckbert`(map list, *nr of colors*)
`MapInterpolContour`(segmap, georef)
`MapInterpolContour`(rasmap)
`MapIter[Prop]`(*StartMap*, IterExpr [, *nr of iterations*])
`MapNearestPoint`(pntmap, georef)
`MapMirrorRotate`(rasmap, *rotate type*)
`MapMovingAverage`(pntmap, georef, *weight function*)
`MapMovingSurface`(pntmap, georef, *surface type*, *weight function*)
`MapRasterizePoint`[Count|Sum](pntmap, georef, *pointsize*)
`MapRasterizePolygon`(polmap, georef)
`MapRasterizeSegment`(segmap, georef)
`MapResample`(rasmap, georef, resample method[, `Patch|NoPatch`])
`MapSegmentDensity`(segmap [, *"mask"*] , georef)
`MapSlicing`(rasmap, *domain* group)
`MapStretch`[Linear](rasmap, *range from*, *domain* [, *range to*])
`MapStretchHistEq`(rasmap, *range from*, *intervals*)

`MapSubMap`(rasmap, first row, first col [, nr rows, nr cols] [, *newgeoref*])
`MapThiessen`(source rasmap [, weight rasmap|`1`], output Distance rasmap)
`MapTrendSurface`(pntmap, georef, *surface type*)

Operations resulting in a polygon map
`PolygonMapAttribute`(polmap, attribute column)
`PolygonMapFromRas`(rasmap [, `8|4` [, `smooth|nosmooth`]])
`PolygonMapFromSegment`(segmap, *"mask"* [, *domain*|*labelpntmap*] [, `auto`])
`PolygonMapLabels`(polmap, pntmap)
`PolygonMapMask`(polmap, *"mask"*)
`PolygonMapNumbering`(polmap [, *newdomain*])
`PolygonMapTransform`(polmap, coordsys)

Operations resulting in a segment map
`SegmentMapAttribute`(segmap, attribute column)
`SegmentMapDensifyCoords`(segmap, *distance*)
`SegmentMapFromRasAreaBnd`(rasmap`, 8|4, smooth|nosmooth, single|composite`)
`SegmentMapGlue`(segmap1, *"mask1"*, segmap2, *"mask2"* [, ...] [,*newdomain*])
`SegmentMapGlue`(*minX, minY, maxX, maxY*,segmap1, *"mask1"*, segmap2, *"mask2"* [, ...] [,*newdomain*])
`SegmentMapLabels`(segmap, pntmap, *set domain*)
`SegmentMapMask`(segmap, *"mask"*)
`SegmentMapNumbering`(segmap [, *newdomain*])
`SegmentMapPolBoundaries`(polmap, *"mask"*, `single|composite`)
`SegmentMapSubMap`(segmap, *minX, minY, maxX, maxY*)
`SegmentMapTransform`(segmap, coord sys)
`SegmentMapTunneling`(segmap, *tunnel width*, *remove nodes*)

Operations resulting in a point map
`PointMapAttribute`(pntmap, attribute column)
`PointMapFromRas`(rasmap)
`PointMapGlue`(pntmap1, *"mask1"*, pntmap2, *"mask2"* [, ...] [,*newdomain*])
`PointMapGlue`(*minX, minY, maxX, maxY*,pntmap1, *"mask1"*, pntmap2, *"mask2"* [, ...] [,*newdomain*])
`PointMapMask`(pntmap, *"mask"*)
`PointMapNumbering`(pntmap [, *newdomain*])
`PointMapPolLabels`(polmap)
`PointMapSegCoords`(segmap)
`PointMapSegDist`(segmap, *distance*)
`PointMapSegNodes`(segmap)
`PointMapSubMap`(pntmap, *minX, minY, maxX, maxY*)
`PointMapTransform`(pntmap, coord sys)

Operations resulting in a table
`TableAutoCorrSemiVar`(rasmap, *pixel shift*)
`TableCross`(rasmap1, rasmap2)
`TableCross`(rasmap1, rasmap2, output cross rasmap)
`TableHistogram`(rasmap)
`TableHistogramPnt`(pntmap)
`TableHistogramPol`(polmap)
`TableHistogramSeg`(segmap)
`TableNeighbourPol`(polmap)
`TablePattAnal`(pntmap)
`TableSegDir`(segmap)
`TableSpatCorr`(pntmap)
`TableTranspose`(table, column *domain* [,*value range*])

Operations resulting in a matrix
`MatrixFactorAnal`(map list)
`MatrixPrincComp`(map list)

Special syntax to create attribute maps
Map.column
Polygonmap.mpa.column
Segmentmap.mps.column
Pointmap.mpp.column

Special syntax to perform color separation
Map.color

See also: Appendices : ILWIS expressions

E.2.3 Construction of expressions

Expressions on the command line and in scripts may have additional parameters to the left of the definition symbol or the assignment symbol. Furthermore, on the command line, expressions may be followed by a semicolon. ILWIS expressions are not case-sensitive.

For more information on expressions of operations, see Appendices : ILWIS expressions. For more information on MapCalc and TabCalc expressions, see Appendices : ILWIS operators and functions (MapCalc/TabCalc).

Expressions cannot be performed if the output object is currently open or is marked read only.

In the expressions and explanation below:

- `Courier` is used for obliged parts in expressions;
- *Italics* is used for parameters with special requirements;
- 'domain' an existing domain; the domain will be used for the output object;
- a vertical bar | represents a choice;
- anything in square brackets [ ] represents an optional parameter.

Expressions on command line without semicolon

```
[$] [objtype] OUTOBJ [{domvr}]  = expression
[$] [objtype] OUTOBJ [{domvr}] := expression
```

- First it is checked whether the output object already exists. If so, a question is asked whether to overwrite the object or not. If you answer the question with 'No', no new object is created.
- When the output object is a raster map, then a Raster Map Definition dialog box appears. In this dialog box, you can select or create a domain for the output map, in case of a value domain you can specify the value range and precision for the output map, and a description for the output map. When the expression does not contain any maps but does contain predefined variables %C, %L , then you also need to specify a geroeference.
- When the expression starts with a $ sign, the output object will be displayed in a window.

Expression on command line with semicolon

```
OUTOBJ [{domvr}]  = expression;
OUTOBJ [{domvr}]  := expression;
```

There is no check on existing output objects; an existing output object is simply overwritten.
You can specify a domain and value range for the output map in a pair of curly brackets after the output object name.
These type of expressions are executed by the script processor.

Expressions in scripts

```
OUTOBJ [{domvr}]  = expression
OUTOBJ [{domvr}]  := expression
```

There is no check on existing output objects; an existing output object is simply overwritten.
You can specify a domain and value range for the output map in a pair of curly brackets after the output object name.
There numerous script commands to handle object properties (set domain, set value range, set georef, break dependencies, calc, release disk space, etc.). For more information, see ILWIS script language.

Explanation of syntax

$	to directly calculate and display the output object in a new window
objtype	to specify the output object type as: `map | pol | seg | pnt | tbl | mat`
OUTOBJ[.ext]	the name of the output object with optional extension
{domvr}	to specify a domain and value range for an output map as: {dom=domain} | {dom=domain;vr=valuerange} | {vr=valuerange;dom=domain} | {vr=valuerange}
valuerange	as min:max:prec | min:max | ::prec
=	to create a dependent output object
:=	to create an editable object
expression	expression of an operation or a MapCalc expression.

Input objects

- For each part of an expression which requires a raster map as input, you can also use:
  - if the raster map is linked to an attribute table, you can directly access attributes:
    map.column
  - if you have a table with the same domain as the raster map, you can access attributes in the table as:
    map.table.col
    map.table.ext.col
    When you do not specify a table name, it is assumed that the attribute table is linked to the raster map.
    When you do not specify an extension for the table, it is assumed that the table has extension .TBT.
- For each part of an expression which requires a table as input, you can also specify:
  table.ext
  where *table.ext* is a table name and its extension. Extension .TBT can be omitted when the table has extension .TBT.

- For each part of an expression which requires a column as input, you can also use
  table.col
  table.ext.col
  in which *table.ext* is a table name and its extension and *col* is a column name in that table. Extension .TBT can be omitted when the table has extension .TBT.
- For each part of an expression which requires a map list, you can also use:
  `mlist`(*rasmap1, rasmap2, rasmap3, ...*)

Domain and value range
In MapCalc and TabCalc expressions and in expressions of operations you can optionally specify an existing domain and/or a value range and/or the precision for the output map or column in a pair of curly brackets between the name of the output object and the definition or assignment symbol.

Examples:

```
map1{dom=domainname;vr=min:max:precision} = expression
map2{dom=domainname;vr=min:max} = expression
map3{dom=domainname;vr=::precision} = expression
map4{dom=domainname} = expression
map5{vr=min:max:precision;dom=domainname} = expression
map6{vr=min:max;dom=domainname} = expression
map7{vr=min:max:precision} = expression
map8{vr=::precision} = expression
```

The same applies for columns.

Appendix F

ILWIS script language (syntax)

A script is a sequenced list of ILWIS expressions. By creating a script, you can build a complete GIS or Remote Sensing analysis for your own research discipline. Scripts are more or less equivalent to batch files in ILWIS version 1.4.

General information

In a script, you can use any MapCalc or TabCalc expression, any expression for an operation, you can use parameters, you can call other scripts, you can display ILWIS objects on the screen, and further you can use a number of commands for file management, to handle object properties, to break dependencies and release disk space, to edit class or ID domains, etc.

When you run a script, no dialog boxes appear and no questions are asked; all lines in the script are simply performed. Error messages appear in case syntax errors are detected in a MapCalc expression, in a TabCalc expression, or in an expression for another operation, or in a script command. Further error messages appear when a script command is not recognized, or when required objects are not found. A script line is ignored when the syntax is correct and necessary objects are found but the command cannot be performed otherwise (e.g. creating objects that already exist, missing or wrong extensions during a copy).

Parameters in scripts

Parameters in a script can replace (parts of) object names, operations, etc. Parameters in scripts work as DOS replaceable parameters in DOS batch files, and must be written in the script as `%1, %2, %3,` up to `%9`. The parameters of a script have to be filled out on the command line when you run the script. The first text string found after the script name will replace `%1` in the script, etc. For more information, see How to use parameters in scripts.

To run a script

To run a script from the command line of the Main window, type:
Run scriptname parameter parameter.
If a script has no parameters, you can directly double-click the script in the Catalog. For more information, see How to run scripts.

Example
An example of a script is presented in Map and Table calculation : creating and running scripts.

☞ Single text lines of a script, i.e. the commands and expressions described below, can also be typed on the command line of the Main window. To avoid any dialog boxes, it is advised to use a semicolon ; at the end of such a line. In a script, semicolons are not allowed.

Expressions for calculations and other operations

Most text lines in a script will consist of MapCalc and TabCalc expressions and expressions of operations that you can also type on the command line of the Main window or on the command line of a table window. You should be familiar with these expressions. For an overview of MapCalc and TabCalc operators and functions, refer to Appendices : MapCalc and TabCalc operators and functions. For an overview of expressions for other operations, refer to Appendices : ILWIS expressions. For more information on the creating of expressions, see Appendices : construction of expressions.

MapCalc and Tabcalc

For MapCalc expressions, no special syntax is required: you can simply type the MapCalc expression as you would type it on the command line of the Main window.
For example, to `sum` maps `map1` and `map2` to create `map3`, type in the script:

```
map3=map1+map2
```

For TabCalc expressions, it is necessary that you type `tabcalc` *`tablename`* in front of the tabcalc expression. For example, to `sum` columns `col1` and `col2` in table `MyTable` and to store the results in column `col3`, type in the script:

```
tabcalc MyTable col3=col1+col2
```

If you like, you can also perform table calculations on other objects that can be opened as a table, e.g. histograms, point maps, class representations. Then, specify the extension of the object after the table name:

`tabcalc` *`tablename.ext`* `a=b+c`

If you want to perform a series of table calculations in one table, it is advised to use the following script commands:
`opentbl` *tablename.ext* Keep the table *tablename.ext* open as the first line before a series of TabCalc expressions on one table.
`closetbl` *tablename.ext* Close the open table *tablename.ext* as the last line after a series of TabCalc expressions on one table.

ILWIS operations

To perform other ILWIS operations, you can use any ILWIS expression as described in Appendices : ILWIS expressions. For example to cross two maps (`map1` and `map2`) and obtain a cross table (`m1m2cr`), type in the script:

```
m1m2cr = TableCross(map1,map2)
```

Additional script commands

A number of additional script commands is available for file management, to show objects, handle object properties, edit object properties, create objects, calling other scripts, etc.

In many of the following script commands, object names and extensions of their object definition files must be specified (see also Appendices : object definition files). In some script commands, you are allowed to use wildcards `*` and `?` to specify object names and their extensions (*object.ext* and *table.ext.col*).

When using a script command that works on a column in a table (*table.ext.col*), you can ignore the extension when the table has extension .TBT. Table extensions only need to be specified when the column is stored in a histogram, a point map, a class representation, a georeference tiepoints, etc.

Further, in the list below, optional parameters of script commands are shown between square brackets. Omit these square brackets when writing a script. Square brackets are only recognized for TabCalc expressions to indicate a specific record in a table.

Remarks

`rem` This is a remark	All text on this line after `rem` is ignored by the script. In this manner, you can document your script expressions.
`//` This is a remark	All text on this line after `//` is ignored by the script. In this manner, you can document your script expressions.

Open/Show an object

`open` *object.ext*	Open/show object *object.ext.*
`show` *object.ext*	Open/show object *object.ext.*

File management

`cd` *path*	Change directory to directory *path.*
`cd` *d:path*	Change drive to *drive d*: and change directory to directory *path.*
`md` [*drive*:]*path*	Make directory *path*. Optionally make directory *path* on drive *drive*
`mkdir` [*drive*:]*path*	Make directory *path*. Optionally make directory on *path* on drive *drive.*
`rd` [*drive*:]*path*	Remove directory *path*. Optionally remove directory *path* from drive *drive.*
`rmdir` [*drive*:]*path*	Remove directory *path*. Optionally remove directory *path* from drive *drive.*

`rd` [*drive*:]*path* `-force`	Remove directory *path* while deleting all files in that directory. Optionally remove directory *path* and all files in that directory from drive *drive*.
`rmdir` [*drive*:]*path* `-force`	Remove directory *path* while deleting all files in that directory. Optionally remove directory *path* and all files in that directory from drive *drive*.

An error message appears when changing to a directory that does not exist, or when removing a directory that does not exist. The script line is ignored, when making a directory that already exists, or when deleting files that do not exist.

`copy` *object.ext objname*	Copy object *object.ext* to new name *objname*.
`copy` *object.ext objname* `-breakdep`	Copy object *object.ext* to new name *objname* while breaking the dependency links of *object.ext*.
`copy` *object.ext path*	Copy object *object.ext* to existing directory *path*. Wildcards are allowed.
`copy` *object.ext path* `-breakdep`	Copy object *object.ext* to existing directory *path* while *breaking the dependencies of object.ext*. Wildcards are allowed.
`copyfile` *file.ext filename.ext*	Copy file *file.ext* to new file *filename.ext*.
`copyfile` *file.ext path*	Copy file *file.ext* to existing directory *path*. Wildcards are allowed.

When copying objects (or files), you cannot copy objects to another directory and give the object another name at the same time.
When copying an object to another directory, existing objects in that directory are not overwritten. In the same way, when copying an object in the current directory to an object name which already exists, the script line is ignored.

`del` *object.ext*	Delete object *object.ext*. Wildcards are allowed.
`del` *object.ext* `-force`	Tries to delete object *object.ext* which is not completely valid (i.e. an error occurs when the object is opened). Wildcards are allowed.
`delcol` *table.ext.column*	Delete column *table.ext.column*.
`delfile` *file.ext*	Delete file *file.ext* as if this file was deleted in DOS or the File Manager. Wildcards are allowed.

The `del` and `delcol` commands check whether the object is not read-only or whether a column is not table-owned; these commands do not take into account whether an object is still used by other objects. The script line is ignored when objects, columns or files do not exist.

Handling properties of dependent objects

`update` *object.ext*	Make the dependent map or table *object.ext* up-to-date. Wildcards are allowed.
`updatecol` *table.ext.column*	Make dependent column *table.ext.column* up-to-date.
`breakdep` *object.ext*	Break the dependency links of dependent map or table *object.ext*. Wildcards are allowed.
`breakdep` *object.ext* `-force`	Tries to break the dependency links of dependent map *object.ext* which is not completely valid (i.e. an error occurs when the object is opened). Wildcards are allowed.
`breakdepcol` *table.ext.column*	Break the dependency links of dependent column *column* in table *table.ext*.
`reldisksp` *object.ext*	Delete the data file(s) of dependent object *object.ext*.
`calc` *object.ext*	Recalculate the data files of dependent map or table *object.ext*. Wildcards are allowed.
`calccol` *table.ext.col*	Recalculate dependent column *table.ext.col*.

Editing properties of editable source objects (advanced)

`changedom` *object.ext* *domname* [*valuerange*]	Change the domain of raster, polygon, segment or point map *object.ext* to existing domain *domname*, while converting the class names, IDs, or values of the original domain into new domain *domname*. Optionally, in case of a value domain *domname*, set the value range of the object to valuerange. Specify the value range as *min:max:precision*, as *min:max*, or as *::precision*. This is a rather safe way to change the domain of a map into another domain or to change the precision of a map.
`setdom` object.ext *domname* [*valuerange*] [`-force`]	Set the domain of *object.ext* to *domname*. Wildcards are allowed for *object.ext*. Optionally, in case of a value domain *domname*, set the value range of the object to *valuerange*. Specify the value range as *min:max:precision*, as *min:max*, or as ::precision. When you also specify the `-force` flag, no checks are performed.

`setvr` *object.ext valuerange*	Set the value range of *object.ext* to *valuerange*. Wildcards are allowed for *object.ext*. Specify the value range as *min:max:precision*, as *min:max*, or as *::precision*.
`setgrf` *rasmap georef*	Set the georeference of raster map *rasmap* to *georef*. Wildcards are allowed for *rasmap*.
`setcsy` *map.ext coordsys*	Set the coordinate system of point, segment or polygon map *map.ext* to *coordsys*. Wildcards are allowed for *map.ext*.
`setcsy` *georef coordsys*	Set the coordinate system of georeference *georef* to *coordsys*. Wildcards are allowed for *georef*.
`setreadonly` *object.ext*	Mark object *object.ext* as read only. Read only objects cannot be edited or deleted. Wildcards are allowed.
`setreadwrite` *object.ext*	Remove the read only flag for object *object.ext*: the objects are editable and deleteable. Wildcards are allowed.
`setatttable` *map.ext atttable*	Set the attribute table of class or ID map *map.ext* to *atttable*. Wildcards are allowed for *map.ext*.
`setatttable` *map.ext*	Remove the link between class or ID map *map.ext* and its attribute table. Wildcards are allowed.

The `setdom`, `setvr`, `setgrf`, `setscy` and `setatttable` commands are only performed on objects that are not read only.

Creating domains

`crdom` *domname* [`-type=class|ID|group`] `-items`=*number* [`-prefix`=prefix]

Create domain *domname* with a number of items specified as `-items`=*number*. Optionally, specify `-type=class` to create a class domain, or `-type=ID` to create an ID domain, or `-type=Group` to create a group domain. If parameter `-type` is omitted, then a class domain is created. Optionally specify `-prefix=`*prefix* to obtain classes or IDs with a certain *prefix*. If parameter `-prefix` is omitted when creating a class domain, classes will obtain prefix `class`, thus `class 1`, `class 2`, etc. If parameter `-prefix` is omitted when creating an ID domain, the IDs will obtain prefix `ID`, thus `ID 1`, `ID 2`, etc.

`crdom` *domname* `-type=value min`=*number* max=*number* [`-precision`=*value*]

Create value domain *domname* with a specified value range between `min=`*number* and `max=`*number*. Optionally, specify a precision for the value domain as `-precision`=value.

Examples:

`crdom` *domname* `-type=class -items=10 -prefix=cl`
Create class domain *domname* and add ten items to this domain with class names "cl 1", "cl 2", .. "cl 10".

`crdom` *domname* `-type=id -items=100 -prefix=prov`
Create ID domain *domname* and add hundred items to this domain with IDs "prov 1", "prov 2", .. "prov 100".

`crdom` *domname* `-items=0`
Create class domain *domname* without any items. You can add items with the `additemtodomain` command.

`crdom` *domname* `-type=value -min=100 -max=200`
Create value domain *domname* with a value range between 100 and 200 (precision is 1).

`crdom` *domname* `-type=value -min=10 -max=20 -prec=0.01`
Create value domain *domname* with a value range between 10.00 and 20.00 and a precision of 0.01.

The `crdom` command is ignored when domain domname already exists.

Editing a class or ID domain

`additemtodomain` *domname class* [*classcode*]
Add item class, optionally with code *classcode*, to class or ID domain *domname*. Class names which contain spaces must be enclosed by single or double quotes.

`mergedom` *domname1 domname2*
Merge the items of class or ID domain *domname2* into class or ID domain domname1.

Create representations

`crrpr` *rprname domname*
Create representation *rprname* for class or value domain *domname*.

The `crrpr` command is ignored when representation *rprname* already exists.

Create georeference corners

`crgrf` *grfname* `500 1000 -crdsys=cs -lowleft=(0,0) -upright=(5000,10000)`

Create georeference corners *grfname* with 500 rows and 1000 columns and using coordinate system `cs`. The coordinate boundaries are defined by the lower left coordinate (0,0) and the upper right coordinate (5000,10000).

`crgrf` *grfname* `500 1000 -crdsys=cs -lowleft=(0,0) -pixsize=10`

Create georeference corners *grfname* with 500 rows and 1000 columns and using coordinate system `cs`. The georeference has as lower left coordinate (0,0) and as pixel size 10 m.

In the two expressions above, the part `-crdsys=`*coordsysname* can be left out; then system coordinate system unknown will be used. Further, by default, the georeference will take corners of corner pixels for the coordinate boundaries. When you specify as well -centercorners+, the centers of corner pixels will be used. The crgrf command is ignored when georeference grfname already exists.

Creating a two-dimensional table

`cr2dim` *2dimtablename indomname1 indomname2 outdomname3 [valuerange]*

Create two-dimensional table *2dimtablename* using existing input domains *indomname1* and *indomname2* and use existing domain *outdomname3* as the domain of the fields in the table. If *outdomname3* is a value domain, you can optionally specify the value range of this domain as *min:max* or as *min:max:precision*.

Converting domains

`domclasstoid` *domname[.ext]*	Convert class domain *domname* into an ID domain.
`domidtoclass` *domname[.ext]*	Convert ID domain *domname* into a class domain.
`dompictoclass` *domname[.ext]*	Convert Picture domain *domname.ext* into a class domain.

These are rather safe ways to convert one domain into another. When the domain you want to convert is an internal domain which is stored in a map, you need to specify the extension of the map after the domain name.

Calling other scripts

`run` *script2*	Run another script named *script2* (without parameters).
`run` *script2 parameter parameter*	Run another script name *script2*; fill out parameters.

If *script2* is not found, an error message appears.

Start other Windows applications

`!`*Command line*	Performs *Command line* as if entered in the Windows (File) Run dialog box. Starts any Windows application, batch file, or DOS application (with a .PIF file available). Applications should have one of the following extensions: `.exe, .com, .bat, .pif`. Type the application name directly after the exclamation mark (no spaces allowed).Example: to start Word and open document MyDoc, type: `!Winword MyDoc`

Import files from ILWIS 1.4 to ILWIS 2.1

`Import14` *file14.ext* [*outputdir*]	Batch-wise import of *file14.ext* in the current directory, or optionally to the specified output directory *outputdir*. Wildcards are allowed. The ILWIS 2 object(s) keep the name(s) of the 1.4 file(s); new extensions are created during import). Domains, representations, georeferences etc. are created using defaults.
`Import14` *file14.ext* [*ilwis2name*|*outputdir*] [`-dmt`=*domtype*] [`-dom`=*domainname*] [`-grf`=*georef*] [`-att`=*tablename*]	Import an ILWIS 1.4 file *file14.ext* according to your wishes. All parameters shown above between square brackets can but do not have to be used.
ilwis2name|*outputdir*	Either specify an ILWIS 2 object name as *ilwis2name* for the 1.4 file to be imported or specify an output directory as *outputdir* in which the imported object should appear. If this parameter is omitted, then *file14.ext* is imported in the current directory and the ILWIS 2 object(s) keep the name(s) of the 1.4 file(s).

`-dmt`=*domtype*	Specify the domain type *domtype* for the output object as: `Picture`, `Image`, `Value`, `Class`, or `ID`. If you specify domain type Class or ID, you also have to use the `-dom` option to specify the name of that domain.
`-dom`=*domname*	If you specified domain type `Class` or `ID` in the previous option, then also specify a new or existing *domname* for the output object.
`-grf`=*georef*	When importing a raster map, you can specify the name of a new or existing georeference *georef*.
`-att`=*tablename*	When importing a 1.4 point table which contains besides the `X!`, `Y!` and `Name$` columns also attribute columns, you can specify the name of the ILWIS 2 attribute table as *tablename*. The point table is then imported as an ILWIS 2 map and the other columns (i.e. columns other than `X!, Y!` and `Name$`) are imported as an attribute table of the point map.

`Import14 orotm4.mpd -dmt=image -grf=tmgeoref`

Import 1.4 raster map `Orotm4` as an image using existing georeference tiepoints `tmgeoref`.

Export files from ILWIS 2 to ILWIS 1.4

`export14` *object2.ext name14*	Export ILWIS 2 raster map, polygon map, segment map, point map or table *object2.ext* to an ILWIS 1.4 file *name14*.

See also: ILWIS objects : scripts

Appendix G

ILWIS Software

G.1 File extensions

In ILWIS 2.1, all ILWIS objects have an ASCII object definition file which refers to the object's data file, domain, store type, projection, georeference, representation (color, line width etc.), etc. Thus, object definition files may refer to further object definition files and to binary data files. The extension of a binary data file is mostly #. All ILWIS 1.4 file extensions are obsolete.

ILWIS 2.1 file extensions

.CD#	Binary data file for segment maps. Stores intermediate coordinates of segments (i.e. the non-nodes). Always together with a .MPS, .SC#, and .SG# file.
.CSY	Object definition and data file for a coordinate system. ASCII file. Coordinate systems always store the possible minimum and maximum X and Y values that can be used in point, segment, and polygon maps and in georeferences of raster maps. Coordinate systems may contain information on a projection.
.DOM	Object definition file for a domain. Stores the set of valid IDs, Class names or values for a data object. ASCII file. Domains of type Class, ID and Group also have a binary data file (.DM#). A domain is used by one or more raster, polygon, segment and/or point maps, tables and/or columns in tables. A domain Class, Value, and Picture use a representation.
.DM!	Binary data file for an internal class, ID or group domain of a point map. If a point map has an internal domain, the object definition file of the point map (.MPP) contains a reference to this .DM! file. A .DM! file is comparable to a .DM# file.
.DM@	Binary data file for an internal class, ID or group domain of a polygon map. If a polygon map has an internal domain, the object definition file of the polygon map (.MPA) contains a reference to this .DM@ file. A .DM@ file is comparable to a .DM# file.
.DM#	Binary data file for a domain ID, Class, or Group. Always together with a .DOM file.

.DM$	Binary data file for an internal class, ID or group domain of a segment map. If a segment map has an internal domain, the object definition file of the segment map (.MPS) contains a reference to this .DM$ file. A .DM$ file is comparable to a .DM# file.
.DM%	Binary data file for an internal class, ID or group domain of a raster map. If a raster map has an internal domain, the object definition file of the raster map (.MPR) contains a reference to this .DM% file. A .DM% file is comparable to a .DM# file.
.FIL	Object definition file for filters which includes its ASCII data file. Standard filters are available in the \SYSTEM directory.
.FND	ASCII data file for functions. Always together with a .FUN file.
.FUN	Object definition file for functions. ASCII file. Always together with a .FND file. Standard functions are available in the \SYSTEM directory.
.GRF	Object definition file for the georeference of raster maps. Stores the relation between rows and columns and real-world coordinates. ASCII file. A georeference tiepoints also has a .GR# file. A georeference can be used by one or more raster maps. A georeference uses a coordinate system; a georeference tiepoints also uses a background raster map on which tiepoints are positioned.
.GR#	Binary data file for a georeference tiepoints. Stores the tiepoints (row, col, X, Y, drow, and dcol), uses a background raster map. Always together with a .GRF file.
.HA#	Binary data file for a histogram of a polygon map. Always together with a .HSA file.
.HI#	Binary data file for a histogram of a raster map. Always together with a .HIS file.
.HIS	Object definition file for histograms of a raster map. ASCII file. Always together with a .HI# file. A histogram of a raster map belongs to the raster map with the same name.
.HP#	Binary data file for a histogram of a point map. Always together with a .HSP file.
.HS#	Binary data file for a histogram of a segment map. Always together with a .HSS file.
.HSA	Object definition file for histograms of a polygon map. ASCII file. Always together with a .HA# file. A histogram of a polygon map belongs to the polygon map with the same name.
.HSP	Object definition file for histograms of a point map. ASCII file. Always together with a .HP# file. A histogram of a point map belongs to the point map with the same name.

.HSS	Object definition file for histograms of a segment map. ASCII file. Always together with a .HS# file. A histogram of a segment map belongs to the segment map with the same name.
.ISF	ILWIS Script File. ASCII data file for a script, containing the commands and expressions. Always together with a .ISL file.
.ISL	ILWIS Script Language. Object definition file for a script. ASCII file. Always together with a .ISF file.
.MAT	Object definition and data file for a matrix. ASCII file. A matrix is for instance produced by the operations Principle Components and Factor Analysis.
.MP#	Binary data file for a raster map. Stores the actual values of a raster map. Always together with an .MPR file.
.MPA	Object definition file for polygon maps (A for Area). Together with a .PC#, .PD#, .PL#, .PS#, and .TP# file, unless a dependent polygon map was not calculated yet. A polygon map uses a domain and a coordinate system. Maps of domain Class, ID or Group can have an attribute table.
.MPL	Object definition and data file for map lists. Stores the names of raster maps in the map list. ASCII file. A map list thus has a reference to a number of raster maps, and to a georeference. If a variance-covariance or correlation matrix is calculated, then the values are stored in the .MPL.
.MPP	Object definition file for point maps. ASCII file. Together with a .PN# file, unless a dependent point map was not calculated yet. A point map uses a domain and a coordinate system. Maps of domain Class, ID or Group can have an attribute table.
.MPR	Object definition file for raster maps. ASCII file. Together with an .MP# file, unless a dependent raster map was not calculated yet. A raster map uses a domain and a georeference. Maps of domain Class, ID or Group can have an attribute table.
.MPS	Object definition file for segment maps. ASCII file. Together with a .CD#, .SC# and .SG#, file, unless a dependent segment map was not calculated yet. A segment map uses a domain and a coordinate system. Maps of domain Class, ID or Group can have an attribute table.
.MPV	Object definition file for map views. ASCII file. Stores the names and display options of the data and annotation layers that were saved as a map view.
.PC#	Binary data file for polygon maps. Stores codes of polygon areas. Always together with an .MPA, .PD#, .PL#, .PS# and .TP# file.

.PD#	Binary data file for polygon maps. Stores coordinates of polygon boundaries. Always together with an .MPA, .PC#, .PL#, .PS# and .TP# file.
.PL#	Binary data file for polygon maps. Stores links to polygon boundaries (.TP#), area and length of polygons. Always together with an .MPA, .PC#, .PD#, .PS# and .TP#.
.PN#	Binary data file for point maps. Always together with an .MPP file.
.PS#	Binary data file for polygon maps. Stores nodes of polygon boundaries. Always together with an .MPA, .PC#, .PD#, .PL# and .TP# file.
.RP!	Binary data file for an internal representation class of a point map. When a point map has an internal representation, the object definition file of the point map (.MPP) contains a reference to this .RP! file. A .RP! file is comparable to a .PR# file.
.RP@	Binary data file for an internal representation class of a polygon map. When a polygon map has an internal representation, the object definition file of the polygon map (.MPP) contains a reference to this .RP@ file. A .RP@ file is comparable to a .PR# file.
.RP#	Binary data file for a representation belonging to a domain Class, Group, or Picture. Always together with a .RPR file.
.RP$	Binary data file for an internal representation class of a segment map. When a segment map has an internal representation, the object definition file of the segment map (.MPS) contains a reference to this .RP$ file. A .RP$ file is comparable to a .PR# file.
.RP%	Binary data file for an internal representation class of a raster map. When a raster map has an internal representation, the object definition file of the raster map (.MPR) contains a reference to this .RP% file. A .RP% file is comparable to a .PR# file.
.RP^	Binary data file for an internal representation class of a picture domain. When a picture domain has an internal representation, the object definition file of the domain (.DOM) contains a reference to this .RP^ file. A .RP^ file is comparable to a .PR# file.
.RPR	Object definition file for representation. Stores the representation of raster, polygon, segment or point maps: colors, line width, etc. ASCII file. A representation for a domain type Class, Group, or Picture also has a binary data file (.RP#). A representation uses a domain.
.SC#	Binary data file for segment maps. Stores segment codes. Always together with an .MPS, .CD# and .SG# file.

.SG#	Binary data file for segment maps; stores the begin and end points of segments (i.e. the nodes) and contains links to the intermediate points. Always together with an .MPS, .CD# and .SC# file.
.SMS	Object definition file for sample sets. ASCII file. Stores references to a sample map with positions of training pixels, a map list and a background map.
.T2#	Binary data file for a two-dimensional table. Always together with a .TA2 file.
.TA2	Object definition file for two-dimensional tables. ASCII file. Always together with a T2# file.
.TB#	Binary data file for a table. Always together with a .TBT file.
.TBT	Object definition file for tables. Stores attribute information belonging to raster, polygon, segment, and/or point maps. ASCII file. Together with a .TB# file, unless a dependent table was not calculated yet. A table uses a domain.
.TP#	Binary data file for topology of polygon maps. In sequence of the segment numbers it gives forward and backward links and index numbers for the left and right polygon. Always together with an .MPA, .PC#, .PD#, .PL# and .PS# file.

G.2 Files overview

Initialization files (*.ini)

The **win.ini** file (WINDOWS directory) primarily contains settings that Windows maintains to customise your Windows environment according to your preferences (see winini.wri).

```
[Extensions]
```

The extension of the ILWIS objects are added by the installation program.

```
[Ports]
```

The communication settings for the digitizer are stored here.

The **system.ini** file primarily contains settings that customise Windows to meet your system's hardware needs (see sysini.wri).

```
[386Enh]
device=hardlock.vxd
```

The **ilwis.ini** file (WINDOWS directory) defines:
general options, how the system works, independent of user. Adjustments on e.g. the status line or the font type or -size.

```
[Digitizer]
```

Contains information about the Digitizer Set up parameters.

```
[DigRef]
[RefPoint]
```

Contains information about the Map Reference parameters.

Optional:

```
[status line]
nohelp=1
```

Disable "Press F1 for Help" on the status line.

```
[ilwis]
fontsize=10
fonttype=arial
fontweight=500
```

Adjusts the Dialog Boxes font type and/or size.

```
[Table]
fontsize=10
fonttype=courier
fontweight=500
```

Adjusts the Table Window font type and/or size. The fonttype must be fixed.

The **ilwiscfg.ini** file (in your start up directory) contains information about how ILWIS will show on screen. The file changes automatically, when the options described below are changed in ILWIS. The file consists of the following:

`[Main Window]`
Contains information about the size and position of the Main Window as shown on screen.

`[Map Window]`
Contains information about whether the Status Line and Button Bar are shown or hidden on screen. It also contains information about whether the size of the Map Window should be automatically adjusted to the area selected with Zoom In.

`[Pixel Info Window]`
Contains information about how the Pixel Info Window is shown on screen and whether the Status Line is shown or hidden.

`[Digitizer Info]`
Contains information about where the Digitizer Window is shown on screen and the color of the digitizer cursor.

`[Point Editor]`
Contains information about the customization of the Point Editor.

`[Catalog Filter]`
Contains information about which ILWIS objects are shown or hidden in the Catalog.
Etc.

Log files (*.log)

The **ilwis.log** file (in your start up directory) lists:

- The date and time ILWIS was opened and closed;
- The commands and expressions processed on the command line of the Main window and Table window. The text can be used directly in a script.

For instance:

```
10:25:26 Fri Jun 6, 1997  Started ILWIS
10:26:46 Fri Jun 6, 1997  prop geol.tbt
10:28:13 Fri Jun 6, 1997  open tmb4.mpr
10:30:37 Fri Jun 6, 1997  export
10:31:35 Fri Jun 6, 1997  Closed ILWIS
```

Definition files (*.def)

The **datum.def** file (ILWIS system directory) defines:
the datum name followed by an equal sign, followed by the name of the ellipsoid and the estimated shifts in x,y and z position in meters, averaged for the specific region. This is described in two different ways:

1. [Datums]
   Datum name = Ellipsoid name dx,dy,dz,region
2. [Datums]
   Datum name = Ellipsoid name
   [datum name]
   region =dx,dy,dz,remark

Extract 1:
```
[Datums]
Rijks Driehoeksmeting=Bessel 1841,-593,-26,-
478,Netherlands
```

Extract 2:
```
[Datums]
Provisional South American 1956=International 1924

[Provisional South American 1956]
Bolivia=-270,188,-388,Bolivia
```

The **ellips.def** file (ILWIS system directory)defines:
the shape of the defined ellipsoids, expressed in the length of the equatorial axis in meters and the inverse flattening.

Extract:
```
[Ellipsoids]
Airy 1830=6377563.396,299.3249646
Modified Airy=6377340.189,299.3249646
Australian National=6378160,298.25
Bessel 1841 (Namibia)=6377483.865,299.1528128
WGS 84=6378137,298.257223563
```

The **projs.def** file (ILWIS system directory) lists:
the names of the projections used in ILWIS. The sequence in the file defines the sequence presented to the user. After a ‘=‘ character a remark might be added.

Extract:
```
[Projections]
Azimuthal Equidistant=
Albers EqualArea Conic=
Central Cylindrical=
Equidistant Conic=
Gnomonic=
```

The **action.def** file (ILWIS directory) defines:
1. the menu structure of the Operations menu in the Main window as:
   1a. Menu, 1b. Submenu, 1c. Menucommand,
   Menu = "-" means not in Operations menu,
   Submenu or Menucommand = "" means no sub-menu or not in menu,
2. the Operations that appear in the Operation-list,
3. the icon of each Operation in the Operation-list;
   icons used are internal 16x16 icons,
4. the extensions determine on which context-sensitive menus a menu command will appear,
5. the ILWIS command that is generated on the command line when an operation is double-clicked in the Operation-list, or a menu command is selected, and
6. the popup help topic number

7. the descriptions that appear on the status line when a menu command is highlighted, or when moving with the mouse over an operation in the Operation-list.
8. the descriptions as they appear on the status line when a menu command of the context-sensitive menu is highlighted. %S is replaced by the base filename.

For instance:

```
"Visuali&zation" "" "&Show Map" "" "" "" open 7010 "Show
a map in a new map window"
"&Raster Operations" "" "&Aggregate Map" "Aggregate"
ExeMap16Ico ".mpr" aggregate 4192 "Aggregate a raster
map"
```

The **actpair.def** file (ILWIS directory) defines:

1. Which objects can be dragged from the Catalog to an operation in the Operation-list, for instance:
   ```
   open   ".mpr" open   "Show raster map %S in a new map
   window"
   filter ".fil" filter "Filter a raster map with filter
   %S"
   ```
   It shows the type of the object that can be dragged, the ILWIS command that is generated on the command line when dragging, and the description that appears on the status line.
2. Which objects can be dragged in the Catalog to another object in the Catalog, for instance:
   ```
   ".mpr" ".fil" filter      "Filter raster map %1 with
   filter %2"
   ".mpr" ".grf" resample    "Resample raster map %1 to
   georeference %2"
   ```
   It shows the object types that can be dragged to each other, the ILWIS command that is generated on the command line when dragging, and the description that appears on the status line.
3. Double-click action, for instance:
   ```
   ""   ".mpl"  edit
   ""   ".dom"  open
   ```
   It shows the object types that can be dubbelclicked and the command that is generated on the command line.
4. Some rules are not activated. These can be recognized by the ‘ ;’ character at the beginning of the sentence. By removing this character, the command is activated. If you want to deactivate a line, add a ‘;’ character at the beginning of the sentence. For example:
   ```
   ;".mpr" ".mpr" cross    "Cross raster maps %1 and %2"
   ;".mpa" ".mpp" labelpol "Recode polygons in polygon
   map %1 according to label points in point map %2"
   ```

The **exp*.def** files (ILWIS directory) consist of lists of different export formats, to which ILWIS 2.1 files can be exported. Each file type has its own list, hence every type can only be exported to specific programs because of the different structures of the files.
The sequence in the exp*.def files defines the sequence as presented to the user; examples:

Code	User text
ilwis14	"ILWIS version 1.4 format .ilw"
Shapefile	"ArcView Shapefile .SHP .xxx"
DXF	"AutoCad .DXF .xxx"
ArcGen	"Arc/Info Generate .PTS .xxx"
Infocam	"Infocam sequential file .SEQ .xxx"

The **import.def** file (ILWIS directory) consists of one list of different import formats, that can be imported in ILWIS 2.1. The sequence in the import.def files defines the sequence as presented to the user. Examples:

Code	Extension	User text
bmp	".bmp"	"Windows bitmap .BMP .xxx"
arcinfo	".nas"	"Arc/Info non-comp ASCII raster .xxx"
lin	".lin.pts"	"Arc/Info ASCII vector (ungen) .xxx"
atlas	".img"	"Atlas image .IMG .xxx"

Executable files (*.exe)

The ILWIS directory contains the following executables:
The **ilw14cnv.exe** file contains import and export possibilities from and to ILWIS 1.4.
The **ilwis.exe** starts ILWIS.
The **regcode.exe** registration program is used to enter the hardlock code.
The **ilw141cnv.pif** file is needed to run ilw14cnv.exe.
The **hardlock.vxd** (WINDOWS system directory).

Dynamic link library files (*.dll)

The ILWIS directory contains the following DLLs:

bc453rtl.dll	ilwisedt.dll	ilwispxi.dll
ilwis.dll	ilwisfnc.dll	ilwisres.dll
ilwisann.dll	ilwisfrm.dll	ilwisrpr.dll
ilwisapp.dll	ilwisgre.dll	ilwisseg.dll
ilwisclc.dll	ilwisgrf.dll	ilwissms.dll
ilwiscnv.dll	ilwisgrw.dll	ilwissrv.dll
ilwiscol.dll	ilwisimp.dll	ilwistbl.dll
ilwiscsy.dll	ilwismap.dll	ilwistbw.dll
ilwisdat.dll	ilwismat.dll	ilwistls.dll
ilwisdig.dll	ilwismpw.dll	ilwisuiw.dll
ilwisdiv.dll	ilwismst.dll	ilwiswnd.dll
ilwisdrw.dll	ilwispnt.dll	ilwiszap.dll
ilwisdsp.dll	ilwispol.dll	ilwiszpw.dll

Help files (*.hlp)

The **ilwis.hlp** file contains help on general information; the **ilwisapp.hlp** file help on operations; and the **ilwismen.hlp** file help on dialog boxes. The ILWIS Help files are located in the ILWIS directory.

MS-Write files (*.wri)

The **readme.wri** file contains information about installation and the latest information on ILWIS. The readme.wri file is located in the ILWIS directory.

Button files (*.but)

annedit.but	map.but	poledit.but
gr3dedit.but	pixedit.but	rprclass.but
grfedit.but	pntedit.but	smpledit.but

The **.but** files in your ILWIS directory contain all buttons that appear in a certain button bar. These buttons are described as the menu item and the icon name. An empty lines gives a small space between the buttons on the bar. Example:

Menu item no.	Icon name
1101	EntireMap16Ico
1102	ZoomIn16Ico
1103	ZoomOut16Ico
1106	SizeLarger16Ico
1107	SizeSmaller16Ico
1108	Redraw16Ico
2302	EndEdit16Ico

English language files (*.eng)

appforms.eng	filter.eng	otherobj.eng
colors.eng	georef.eng	pixinfo.eng
conv.eng	georef3d.eng	represen.eng
coordsys.eng	graph.eng	sample.eng
digitizr.eng	ilwisgen.eng	symbol.eng
domain.eng	impexp.eng	table.eng
drwforms.eng	mainwind.eng	userint.eng
dsc.eng	mapwind.eng	
editor.eng	men.eng	

The English language files in your ILWIS directory contain Internal numbers linked to the English text in the program.

The **dsc.eng** file contains menu item numbers, linked to the descriptions of the Menu Commands.
The **men.eng** file contains menu item numbers, linked to the menu items texts.

G.3 System objects overview

Coordinate systems

latlon.csy
unknown.csy

Domains

binary.dom	count.dom	perc.dom
bit.dom	distance.dom	real.dom
bool.dom	image.dom	string.dom
color.dom	min1to1.dom	value.dom
colorcmp.dom	none.dom	yesno.dom

Filters

avg3x3.fil	edgesenh.fil	majundef.fil
binmajor.fil	frost.fil	majzero.fil
conn8to4.fil	frostenh.fil	med3x3.fil
d2fdx2.fil	gammamap.fil	med5x5.fil
d2fdxdy.fil	inbnd4.fil	outbnd4.fil
d2fdy2.fil	inbnd8.fil	outbnd8.fil
dfddn.fil	kuan.fil	pattern.fil
dfdup.fil	laplace.fil	peppsalt.fil
dfdx.fil	lee.fil	shadow.fil
dfdy.fil	leeenh.fil	shrink4.fil
dilate4.fil	lifegame.fil	shrink8.fil
dilate8.fil	majority.fil	

Georeference

none.grf

Representations

blue.rpr	colorcmp.rpr	inverse.rpr
clrstp10.rpr	cyan.rpr	magenta.rpr
clrstp12.rpr	finegray.rpr	pseudo.rpr
clrstp6.rpr	gray.rpr	red.rpr
clrstp8.rpr	green.rpr	

G.4 General structure of object definition files

ILWIS objects have an ASCII object definition file which contains all information on the object.

Object definition files consist of sections (see also the Windows win.ini and system.ini files). Each section contains further references to e.g. the properties of the object: domain, georeference, representation, and store type, etc.
Object definition files of dependent data objects, also store the definition of how the dependent object was created.

For data objects, the following object definition files exist

.MPL	for map lists
.MPR	for raster maps
.MPA	for polygon maps (area)
.MPS	for segment maps
.MPP	for point maps
.TBT	for tables
.DOM	for domains
.RPR	for representation
.GRF	for georeference
.CSY	for coordinate system

For special objects, the following object definition files exist

.MPV	for map views
.HIS	for histograms of raster maps
.HSA	for histograms of polygon maps
.HSS	for histograms of segment maps
.HSP	for histograms of point maps
.SMS	for sample sets
.TA2	for two-dimensional tables
.MAT	for matrices
.FIL	for user-defined filters
.FUN	for user-defined functions
.ISL	for user-written scripts (ILWIS Script Language)

Below, the general structure of object definition files is described

- each section starts with [SectionName]
- each entry in a section is followed by the character =
- the | means that each of the listed values is possible

The text below merely gives an overview of all possible sections and of all possible entries of sections. In reality, only sections relevant to the object appear in the object's wrapper file and each entry will only have one answer.

```
[Ilwis]
Description=...
Time=...
Type= BaseMap | MapList | Table | Domain | Representation | GeoRef | CoordSys |
        MapView | SampleSet | Matrix | Function | Script |

[BaseMap]
Type= Map | PolygonMap | SegmentMap | PointMap
CoordSystem=name.csy
CoordBounds=minX minY maxX maxY
Domain=name.dom
DomainInfo=

etc.
```

Example .MPR of raster map TMB1

```
[Ilwis]
Description=Landsat TM Band 1
Time=854859600
DataReadOnly=no
PropertiesReadOnly=no
Type=BaseMap

[BaseMap]
CoordSystem=cochabam.csy
CoordBounds=795306.847615274251 8072108.2638787888
809171.542633942328 8090259.36385272816
Domain=Image
Type=Map
DomainInfo=image.dom;Byte;image;0;;
MinMax=25:162
Perc1=29:80

[Map]
GeoRef=tmgeo.grf
Size=550 340
Type=MapStore
MinMax=25:162
Perc1=29:80

[MapStore]
Data=tmb1.mp#
Structure=Line
StartOffset=0
RowLength=340
Type=Byte
StoreTime=854859600
```

Example .MPA of polygon map Landuse

```
[Ilwis]
Description=Polygon map of land use units
Time=872522138
Type=BaseMap

[BaseMap]
CoordSystem=cochabam.csy
CoordBounds=795499.22359436797 8071897.89040035102
808202.078751474153 8090500.68225380685
DomainInfo=landuse.dom;Byte;class;12;;
Type=PolygonMap
Domain=landuse.dom
AttributeTable=landuse.tbt

[PolygonMap]
Type=PolygonMapStore

[PolygonMapStore]
DataPol=landuse.pl#
DataPolCode=landuse.pc#
DataTop=landuse.tp#
Polygons=93
DeletedPolygons=0

[IlwisObjectVirtual]
Expression=PolygonMapFromSegment(landuse,"",lan1.mpp)
DefOnlyPossible=No

[ObjectDependency]
NrDepObjects=2
Object0=landuse.mps
Object1=lan1.mpp

[PolygonMapVirtual]
Expression=PolygonMapFromSegment(landuse,"",lan1.mpp)
Type=PolygonMapFromSegment

[PolygonMapFromSegment]
SegmentMap=landuse.mps
Mask=
AutoCorrect=No
Labels=lan1.mpp

[SegmentMap]
Alfa=0.427346
Beta1=801850.437500
Beta2=8081199.500000
Snap Distance=124.018612
Tunnel Width=12.401861
Type=SegmentMapStore
```

```
[SegmentMapStore]
DataSeg=landuse.ps#
DataCrd=landuse.pd#
Status=0
Format=1
Segments=231
DeletedSegments=0
Coordinates=2554
```

Example .TBT of table Landuse

```
[Ilwis]
Description=Table with land use information
Time=866482824
Type=Table
DataReadOnly=no
PropertiesReadOnly=no

[Table]
Domain=landuse.dom
DomainInfo=landuse.dom;Byte;class;12;;
Columns=1
Records=12
Type=TableStore

[TableStore]
Data=landuse.tb#
Type=TableBinary
StoreTime=866481982
Col0=Landvalue

[Col:Landvalue]
Description=Average value of the land per hectare
Time=866482116
Domain=value.dom
DomainInfo=value.dom;Long;value;0;-9999999.9:9999999.9:0.1:offset=0;
Range=0:10000:offset=0
ReadOnly=No
OwnedByTable=No
StoreType=Int
Stored=Yes
Type=ColumnStore
DataReadOnly=no
PropertiesReadOnly=no
MinMax=50:1000
NrDepObjects=0
```

Appendix H

Keyboard shortcuts

General

F1	Open context-sensitive Help about the currently active window or dialog box.
F11	Display the contents of ILWIS on-line Help.
Alt+F4	Close the active window.
Alt+Tab	Show all opened Windows applications one by one.
Alt+Esc	Activate all windows and minimized windows on the desktop one by one (next applications).
Shift+Alt+Esc	Activate all windows and minimized windows on the desktop one by one (previous applications).
Ctrl+Esc	Open the Task List.

To open menus and to choose commands on menus

Alt or F10	Activate the menu bar of the current window. Use the Down arrow or type an underlined letter to open a menu, and use the Left and Right Arrow to move to another menu.
Alt+char	To open a menu in a window, press Alt and simultaneously the key of an underlined letter in a menu name. For instance, to open the File menu, press Alt+F.
char	To choose a menu command when a menu is opened, press the key of an underlined character of the command. For instance, to choose Exit on the opened File menu, press X.
Esc	Stop the activation of a menu.
Alt+Space	Open the Control Menu of a window.

Cut, Copy, Paste, Clear

Ctrl+X	Cut: clear the (selected) contents from the active window and copy it to the clipboard.
Ctrl+C	Copy the (selected) contents of the active window to the clipboard.
Ctrl+V	Paste the contents of the clipboard into the active window.
Del	Clear the (selected) contents from the active window.
Shift+Del	Cut: clear the (selected) contents from the active window and copy it to the clipboard.
Ctrl+Ins	Copy the (selected) contents of the active window to clipboard.
Shift+Ins	Paste the contents of the clipboard into the active window.

PrintScrn	Copy the entire screen/desktop to the clipboard.
Alt+PrintScrn	Copy an entire active window to clipboard including title bar, menu bar, window borders etc.

Main window

Help	F1	Open Help on Main window.
	F11	Display the contents of ILWIS on-line Help.
Move cursor	Arrows	When an item is selected in the Operation-list or in the Catalog, the cursor moves up and down in the Operation-list or up, down, to the left or to the right in the Catalog.
	PgUp,PgDn	When an item is selected in the Operation-list or in the Catalog, the cursor moves up or down in the Operation-list or in the Catalog.
	Character	In the Operation-list and in the Catalog, the cursor moves to the next item beginning with that character.
Close	Alt+F4	Close the Main window and thus leave ILWIS.

Map window

Help	F1	Open Help on map window.
	F11	Display the contents of ILWIS on-line Help.
Redraw map	F5	Redraw the map; the color lookup table is re-initialized.
	Ctrl+R	Redraw the map; the color lookup table is re-initialized.
Stop drawing	Esc	Stop the (re)drawing of all map windows.
Copy	Ctrl+C	Copy the contents of the map window to the clipboard.
	Ctrl+Ins	Copy the contents of the map window to the clipboard.
Paste	Ctrl+V	Paste the contents of the clipboard into a map window as a new annotation layer.
	Shift+Ins	Paste the contents of the clipboard into a map window as a new annotation layer.
Zoom	Ctrl+I	Zoom in on the center of the map window with a factor 2.
	Ctrl+O	Zoom out with a factor 2.
	Ctrl+E	Display the entire map.
Move cursor	Arrows	Move the cursor 1 screen pixel up, down, to the left or to the right.
	Ctrl+Arrow	Move the cursor 10 screen pixels up, down, to the left or to the right.
Scrolling	PgUp	Scroll upwards (1 page).
	PgDn	Scroll downwards (1 page).

	Home	Scroll to the left (1 page).
	End	Scroll to the right (1 page).
	Ctrl+PgUp	Scroll to the top of the map.
	Ctrl+PgDn	Scroll to the bottom of the map.
	Ctrl+Home	Scroll to the extreme left of the map.
	Ctrl+End	Scroll to the extreme right of the map.
Print	Ctrl+P	Print the contents of the map window.
Exit	Alt+F4	Close a map window.

All editors

F1	Help on the current editor.
F11	Display the contents of ILWIS on-line Help.
Enter	Edit code of selection.
Exit / Alt+F4	Close a map window.

Pixel editor/Sample set editor

Arrows	Move the pixel selection cursor 1 pixel up, down, to the left, or to the right.
Shift+arrows	Move the pixel selection cursor 1 pixel up, down, to the left, or to the right, and switch the selection of this pixel: deselect it when it was selected, and select it when it was not yet selected.
Ctrl+arrows	Move the pixel selection cursor 1 pixel up, down, to the left, or to the right, and add this pixel to the selection.
Shift+Ctrl+arrows	Move the pixel selection cursor 1 pixel up, down, to the left, or to the right, and deselect this pixel from the selection.
Space	Switch the selection of this pixel: deselect it when it was selected, and select it when it was not yet selected.
Esc	Deselect all pixels.
Del	Set all selected pixels to undefined.

Copy and Paste in the pixel editor:

Cut	Copy the selected pixels into the clipboard, and clear the selection. The clipboard contains the pixels in point format as well as in table format; so you may paste the pixels as points into a point map or into any table.
Copy	Copy the currently selected pixels into the clipboard. The clipboard contains the pixels in point format as well as in table format; so you may paste the pixels as points into a point map or into any table.
Paste	On the coordinates of the points in the clipboard: paste the point codes into the current map.

Polygon editor

Copy in the polygon editor:

Copy	Copy the currently selected polygons into the clipboard.

Segment editor

Esc	Switch to Select Mode, or when entering a segment: stop entering the segment, and return to the situation before starting to enter this segment; or when moving a point: stop moving the point, and return to the situation before starting to move this point.
Ins	Switch between Select Mode and Insert Mode.
Del	Delete selected segments.
Space	End a segment with a loose end.

Copy and Paste in the segment editor:

Cut	Copy the currently selected segments into the clipboard and delete the selection.
Copy	Copy the currently selected points into the clipboard.
Paste	Pastes data from the clipboard into the current map.

Point editor

Esc	Switch to Select Mode.
Ins	Switch between Select Mode and Insert Mode.
Del	Delete selected points.

Copy and Paste in the point editor:

Cut	Copy the currently selected points into the clipboard and delete the selection.
Copy	Copy the currently selected points into the clipboard.
Paste	Pastes data from the clipboard into the current map.

Table window

Ctrl+G	Goto a specific record.	
Ctrl+R	Show the table in Record View (record by record).	
Ctrl+T	Show the table in Table View (as a whole table).	
Ctrl+F	Start editing fields.	
Esc	Clear the current selection, or, when in Record View: return to Table View.	
Scrolling	PgUp	scroll 10 records up.
	PgDn	scroll 10 records down.
	Ctrl+Home	scroll vertically to the top field of the table.
	Ctrl+End	scroll vertically to the bottom field of the table.
	Ctrl+PgDn	scroll 4 columns to the right.
	Ctrl+PgUp	scroll 4 columns to the left.

When editing a table:

Left arrow	Move one character to the left within the active field, or when editing a field of a domain type Class column: select the previous class in the class list.
Right arrow	Move one character to the right within the active field, or when editing a field of a domain type Class column: select the next class in the class list.
Ctrl+Left	Move to the previous word in the active field.
Ctrl+Right	Move to the next word in the active field.
Home	Move to the beginning of the active field.
Ctrl+Home	Move to the beginning of the active field.
End	Move to the end of the active field.
Ctrl+End	Move to the end of the active field.
Esc	Stop editing, do not save the contents current field.
Enter	Accept the contents of the current field.
Up Arrow	Accept the contents of the current field and move one field up.
Down Arrow	Accept the contents of the current field and move one field down.
Tab	Accept the contents of the current field and move one field to the right.
Shift+Tab	Accept the contents of the current field and move one field to the left.
Shift+Up	Accept the contents of the current field and move 10 fields up.
Shift+Down	Accept the contents of the current field and move 10 fields down.
Ctrl+Tab	Accept the contents of the current field and move 5 fields to the right.
Ctrl+Up	Accept the contents of the current field and move 100 fields up.
Ctrl+Down	Accept the contents of the current field and move 100 fields down.
Alt+F4	Close a table window.

Copy and Paste in a table window:

Copy	Copies the current selection to the clipboard.
Paste	Pastes the contents of the clipboard into the currently selected fields.

Dialog boxes

F1	Open Help for this dialog box.
F11	Display the contents of ILWIS on-line Help.
Tab	Move to the next option or area in the dialog box.
Shift+Tab	Move to the previous option or area in the dialog box.
Alt+char	To move to certain dialog box item, press Alt and simultaneously the key of an underlined letter in the dialog box.
Space	When the currently selected item in the dialog box is a check box, press the space bar to select or clear this check box. When the currently selected item in the dialog box are option buttons, press the space bar to select the next option.
Enter	Close the dialog box and perform the action according to the parameters used in the dialog box (OK). If a parameter is not correctly selected, the dialog box remains open and the incorrect entry is highlighted.
Esc or Alt+F4	Close the dialog box without performing any action (Cancel).

Appendix I

Mouse functions

Objects in Catalog

Move	Shows the description of the object on the status line.
Click	Shows the double-click action on the command line.
Double-click	Opens the object in a new window or shows it in a dialog box. Exceptions: double-clicking a filter returns the Filter operation dialog box; and double-clicking a script runs the script.
Right mouse click	A context-sensitive menu appears which lists the operations and other commands that can be performed on the object.
Drag to other object	Performs the operation as described on the status line. The operation's dialog box is displayed. The dragged object is already selected in this dialog box.
Drag to operation	Performs the operation as described on the status line. The operation's dialog box is displayed. The dragged object is already selected in this dialog box.
Drag to map window	Adds the dragged map to the map window. Also, georeferences and coordinate systems may be dragged to a map window.
Drag to pixel info window	Adds this map (and its attribute table) to the pixel info window.
Drag to drop-down list box	You can drag an ILWIS object from the Catalog to a drop-down list box in a dialog box: the dragged object is selected.

Operations in the Operation-list

Move	Shows the description of the operation on the status line.
Click	Shows the double-click action on command line.
Double-click	The operation's dialog box is displayed.
Right mouse click	A context-sensitive menu appears, which lists options on the operations.

Map window

Move	The status line displays the current Column Row number, XY-coordinates, geographic coordinates.
Left mouse down	Shows the code of the element located at the mouse pointer, of the top most layer in the map window which has option Info active.
Right mouse click	A context-sensitive menu appears which gives some short cuts to select an area in a map, to go to Layer Management, or to change the Double-click action. Further, all data and/or annotation layers used in that map window are listed: select a layer to change its display options.
Double-click	Depending on how you defined the double-click action: Edit Attributes, Edit Representation, or Execute Action of the top most layer in the map window which has option Info active.

All editors

Click	Clear the current selection and select this element.
Shift+click	Switch selection: when you click on already selected elements, then these are deselected; when you click on non-selected elements, then these are selected.
Ctrl+click	Add element to selection.
Ctrl+Shift+click	Remove element from selection.
Double-click	Edit the code of the element that was double-clicked.
Right mouse click	A context-sensitive menu appears which lists actions related to the editor.

Pixel editor and Sample set editor

Move	The code of the element at the mouse pointer is displayed on the status line.
Click	Select the pixel; use Shift and Ctrl modifiers as described above.
Drag rectangle	Select all pixels in the area.

Point editor - Select mode

Click	Select the point; use Shift and Ctrl modifiers as described above.
Drag rectangle	Select all points in the area; use Shift and Ctrl modifiers as described above.
Double-click	Edit the code of the point.

Point editor - Insert mode

Click	Add a point and immediately edit its code.

Point editor - Move Points mode

Drag	Take a point and move it to a new position.
Release left mouse	Drop the point at this position.

Segment editor - Select mode

Click	Select the segment; use the Shift and Ctrl modifiers as described above.
Double-click	Edit the code of the segment.

Segment editor - Insert mode

Click	Enter a segment coordinate. If this is the first coordinate of a new segment, a begin node is created. If AutoSnap is not active, this side of the segment is a loose end. If AutoSnap is active (default) and you start a new segment on top of an existing segment, then the new segment is automatically snapped to the existing one.
Shift+click	Snap to an existing node (to begin or to end a segment).
Ctrl+click	Split and Snap: create a node in an existing segment and snap to it (to begin or to end a segment).
Left Mouse + move	Enter segment coordinates.
Right Mouse	While entering segment coordinates: delete the last coordinate. When the begin node of a segment is reached, this is also removed.
Double-click	While entering segment coordinates: create an end node for this segment. This side of the segment is a loose end.
Shift+release left mouse	While entering segment coordinates: snap to an existing node (to end a segment).
Ctrl+release left mouse	While entering segment coordinates: split and snap: create a node on an existing segment and snap to it (to end a segment).
AutoSnap active:	The option AutoSnap makes the use of Shift and Ctrl keys redundant: ▪ When starting a segment on top of an existing one, a node to connect both segments is automatically created and ▪ when entering segment coordinates and releasing the left mouse button on top of an existing segment, a node to connect both segments is automatically created.

Segment editor - Move Points mode

Click	Click a node to pick up this node (the segment attached to it changes color). Click again to drop the node. When clicking a node with more segments attached to it, keep on clicking until the node with the segment that you want to move changes color.
Drag	Take a node or an intermediate segment coordinate and move it to a new position.
Release left mouse	Drop the node or intermediate segment coordinate at this position.
Shift+release left mouse	When moving nodes: snap the moved node to an existing node.

Segment editor - Split/Merge mode

Click	Click on an existing segment to split (create a node). Click a node with two segments attached to it to merge these segments (remove node).

Table window

Drag rectangle on fields	Select fields.
Drag column button	Move the column to another position in the table (i.e. change the order of columns).
Drag column button edge	Change the width of the column.
Drag on row buttons	Select the records.
Click field	Edit the field.
Shift+click field	After selecting a field, use Shift+click on another field to select all fields in between.
Click column button	Select the column.
Shift+click column button	After selecting a column, use Shift+click on another column button to select all columns in between.
Click row button	Switch to record view.
Shift+click row button	After selecting records, use Shift+click on another row button to selects all records in between.
Click upper left button	Switch to record view.
Double-click column button	Change the properties of the column.

Drop-down list box in a dialog box

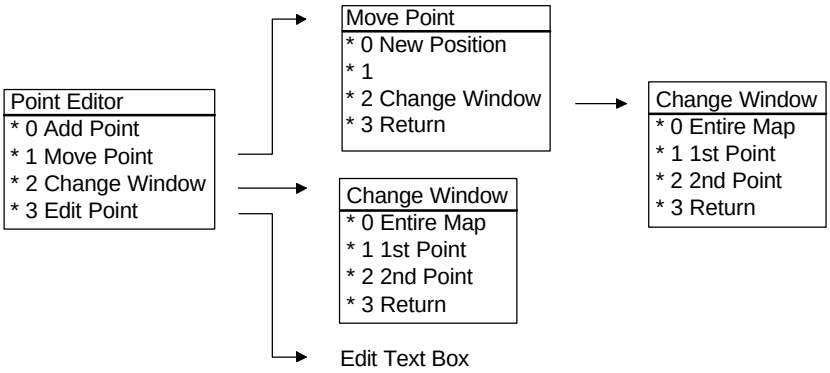
In a drop-down list box in a dialog box, directories and drives are listed besides the ILWIS objects.

Click directory	Go to that directory, and show the objects in that directory.
Click drive	Go to that drive, and show the objects and directories on that drive.

Appendix J

Digitizer menus

J.1 Point Editor



In the Point Editor menu, you can add, move or edit (change class name, ID or value) a point. You can select:

- *0 Add point*; move the cursor to the position where you want to add a point. Press the button to open a text box. Enter the class, ID or value corresponding with the point you want to add (depending on the Domain of the point map that you are editing). Click OK in the box or press ENTER on the keyboard to add, click cancel in the box or press ESCAPE on the keyboard to return without adding.
- *1 Move Point*; move the digitizer cursor to the point you want to move. Press the button to switch to the Move Point menu.
- *2 Change Window*; press the button to switch to the Change Window menu.
- *3 Edit Point*; move the cursor to the point you want to edit. Press the button to open a text box. Enter the new class, ID or value (depending on the Domain of the point map that you are editing). Press OK in the window or ENTER on the keyboard to edit the point, or ESCAPE to return without changing anything.

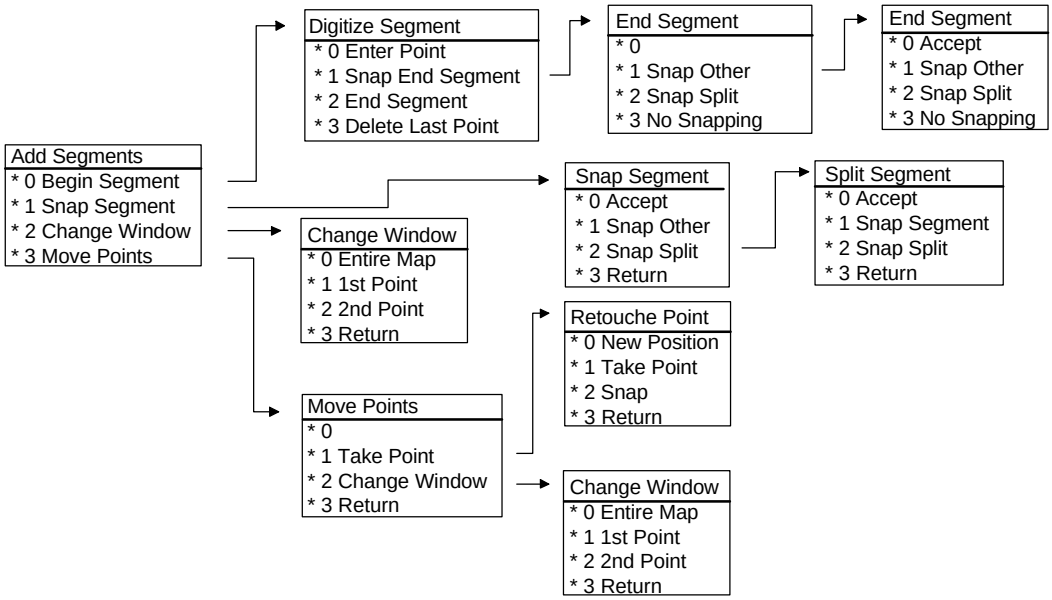
In the Move Point menu, you can move a point to another position. You can select:

- *0 New position*; move the cursor to the position you want to move the point to. Press the button to put the point on the desired position.
- *1* This button has no function.
- *2 Change Window*; press the button to switch to the Change Window menu.
- *3 Return*; press the button to return.

In the Change Window menu, you can zoom in or out on the map. You can select:

- *0 Entire Map*; press the button to display the entire map after a zoom in command and return.
- *1 1st Point*; move the cursor to the first corner of the area you want to zoom in. Press the button, you now see the zoom in box tied to the first point.
- *2 2nd Point*; after you have selected the first point (see above), move the cursor to the 2nd position of the area you want to zoom in. Press the button, to zoom in the area.
- *3 Return*; press the button to return.

J.2 Segment Editor



In the Add Segments menu, you can begin a new segment or snap another segment. You can select:

- *0 Begin Segment*; move the cursor to the first point of the new segment. Press the button to switch to the Digitize Segment menu.
- *1 Snap Segment*; move the cursor to the node or the position on a segment you want to snap to. Press the button to switch to Snap Segment menu.
- *2 Change Window*; press the button to switch to the Change Window menu.
- *3 Move Points*; move the cursor to the point you want to move. Press the button to switch to the Move Points menu.

In the Digitize Segment menu, you can enter new segment points, snap to another segment, end the segment and delete the last entered points. You can select:

- *0 Enter Point*; move the cursor to the next position of the new segment. Press the button one time, a new point is now added, etc. When you keep the button pressed down, the points are automatically added when you move the cursor over the new segment.
- *1 Snap End Segment*; move the cursor to the position or a node of the segment you want to merge with. Press the button to switch to the End Segment menu.
- *2 End Segment*; move the cursor to the last point/position of the segment. Press the button to end the segment and to open a text box. Enter the class, ID or value corresponding with the segment you want to add (depending on the Domain of the map that you are editing). Click OK in the box or press ENTER on the keyboard to add, click cancel in the box or press ESCAPE on the keyboard to return without adding.

- *3 Delete last point*; press the button to delete the last point entered. Press again or press and hold down the button and move the cursor to delete the previous entered point(s).

In the End Segment menu, you can accept to snap another segment, select a segment to snap to and split a segment so it can be snapped. You can select:
- *0 Accept*; press the button to snap.
- *1 Snap Other*; move the cursor to a node you want to snap to. Press the button to continue in the same menu.
- *2 Snap Split*; move the cursor to the position on the segment you want to snap to. Press the button to split the segment and continue in the same menu.
- *3 No Snapping*; press the button to return.

In the Snap Segment menu, you can accept to snap another segment, select a segment to snap to and split a segment so it can be snapped. You can select:
- *0 Accept*; when snapping to a node, press the button to snap.
- *1 Snap Other*; move the cursor to another node you want to snap to. Press the button to continue in the same menu.
- *2 Snap Split*; move the cursor to the position on a segment you want to snap to. Press the button to split the segment and continue in Split Segment menu.
- *3 No Snapping*; press the button to return.

In the Split Segment menu, you can accept to merge with another segment, select a segment to merge with and split a segment so it can be merged. You can select:
- *0 Accept*; press the button to merge.
- *1 Snap Segment*; move the cursor to a node you want to merge with. Press the button to continue in Split Segment menu.
- *2 Snap Split*; move the cursor to the position on the segment you want to merge with. Press the button to split the segment and continue in the same menu.
- *3 Return*; press the button to return.

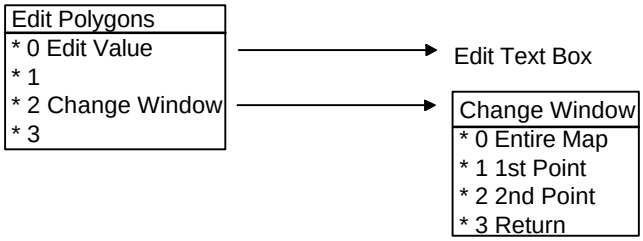
In the Move Points menu, you can move a point to another position. You can select:
- *0* This button has no function.
- *1 Take Point*; select a point you want to move on the segment. Press the button to switch to Retouche Point menu.
- *2 Change Window*; press the button to switch to the Change Window menu.
- *3 Return*; press the button to return.

In the Retouche Point Menu, you can move a point to a new position or select another point to move. You can select:
- *0 New position*; move the cursor to the position you want to move the point to. Press the button to drop the point on the desired position.
- *1 Take Point*; move the cursor to another point that you want to move. Press the button to pick up the new point, without changing the position of the previous picked up point, and continue in Retouche Point menu.
- *2 Change Window*; press the button to switch to the Change Window menu.
- *3 Return*; press the button to return.

In the Change Window menu, you can zoom in or out on the map. You can select:

- *0 Entire Map*; press the button to display the entire map after a zoom in command and return.
- *1 1st Point*; move the cursor to the first corner of the area you want to zoom in. Press the button, you now see the zoom in box tied to the first point.
- *2 2nd Point*; after you have selected the first point (see above), move the cursor to the 2nd position of the area you want to zoom in. Press the button to zoom in the area.
- *3 Return*; press the button to return.

J.3 Polygon Editor



In the Edit Polygons menu, you can edit the value (change class name, ID or value) of a polygon and zoom in or out on the map.

- *0 Edit Value*; move the cursor to the polygon that you want to edit. Press the button to open a text box. In the text box, the area and length of the polygon are displayed, and a new class, ID or value (depending on the Domain of the map) can be entered. Click OK in the box or press ENTER on the keyboard to edit, click cancel in the box or press ESCAPE on the keyboard to return.
- *1* This button has no function.
- *2 Change Window*; press the button to switch to the Change Window menu.
- *3* This button has no function.

In the Change Window menu, you can zoom in or out on the map. You can select:

- *0 Entire Map*; press the button to display the entire map after a zoom in command and return.
- *1 1st Point*; move the cursor to the first corner of the area you want to zoom in. Press the button, you now see the zoom in box tied to the first point.
- *2 2nd Point*; after you have selected the first point (see above), move the cursor to the 2nd position of the area you want to zoom in. Press the button, to zoom in the area.
- *3 Return*; press the button to return.

Appendix K

ILWIS as DDE server

For application programmers it can be useful to know that ILWIS operates as DDE-server. You can make your own forms in e.g. Visual Basic or Delphi based on user input call scripts, create new dependent data or show some maps. Also Pixel Info can be used from e.g. Paradox, enabling you to make a more direct link to your database with special designed information forms. For general information about DDE, see the documentation of your application program.

The server name is "ilwis"
The topic names are "system", "calc" and "coord".

Topic "system"

Behaves as the standard system topic.
Request "topics" to get a list of all current topics.
Request "sysitems" to get a list of all system items.
Request "status" to test if the system is ready. (yes always)
Request "format" to get a list of all supported formats. (text)
Execute follows the standards:
Syntax: [command1][command2][command3]
The brackets are mandatory.
Normally one command would be given, but more are possible. A command can be anything that can be typed in on the ILWIS command line; optionally the parameters can be placed in parenthesis, as shown in the example:
[show][show band1.mpr][show(band1.mpr)][show("band1.mpr")]
calls **show** several times with the same map to display, utilizing a different syntax each time.

Topic "calc"

Request a formula "command" as if typed in on the ILWIS command line as "?command". ILWIS will calculate the result and pass is to the DDE client as text.

Topic "coord"

Request ".x" for the current X-coordinate
Request ".y" for the current Y-coordinate
Request ".xy" for the whole current XY-coordinate
Request any basemap for the pixel value at the current coordinate. All requests are also available as advise-link. Every time the user presses the left mouse button a new value is sent.

Example

The example show how an ILWIS command, as normally entered through the ILWIS command line, can be executed from within a Visual Basic application.

```
Private Sub Exec_Click()
    Const NONE = 0, LINK_MANUAL = 2 ' Declare
constants.'
    If IlwCmd.LinkMode = NONE Then   ' Test link mode.'
        IlwCmd.LinkTopic = "ilwis|system"    ' Set link
        topic.
        IlwCmd.LinkItem = "execute" ' Set link item.
        IlwCmd.LinkMode = LINK_MANUAL    ' Set link
        mode.
        IlwCmd.LinkTimeout = -1
        IlwCmd.LinkExecute "[" + IlwCmd.Text + "]" '
        Execute the Ilwcmd
        Text1.Text = IlwCmd.Text
    End If
End Sub
```