Abstract

This manual was automatically generated from the source code of version 0.93.1666 (20090610) of the rasp (Routines for Airborne Scanner Processing) scanner data processing system.

rasp is currently available for Linux and Windows XP systems, and can operate in a script-driven mode, using a command line interpreter, from a graphical user interface, or a combination thereof. As great care has been taken to keep the code as portable as possible, this description of rasp should be applicable to any computing platform where the FreePascal Compiler (http://www.freepascal.org) and OpenGL (http://www.opengl.org/) are available.

Copyright

Copyright © 2007-2009
Airborne Research Australia/Wolfgang Lieff
Hangar 60, Parafield Airport, South Australia

As this document is generated automatically, the (re-)usage conditions of the underlying source code apply. Please refer to the header of the main source file ‘rasp.pas’ for more information.
Contents

1 General Concepts 7
2 Running rasp 8
   2.1 Invocation ........................................ 8
   2.2 Command Line Options ............................. 8
3 Command Interpreter 9
   3.1 Infrastructure Commands ......................... 9
      3.1.1 Quit ......................................... 9
      3.1.2 Prompt ....................................... 10
      3.1.3 Print ......................................... 10
      3.1.4 Set.PrintFile .................................. 10
      3.1.5 About ......................................... 10
      3.1.6 Execute ....................................... 10
      3.1.7 Delay ......................................... 11
      3.1.8 CD ............................................. 11
      3.1.9 Repeat ........................................ 11
      3.1.10 Wait .......................................... 11
      3.1.11 Help .......................................... 11
      3.1.12 DEMpath ...................................... 11
      3.1.13 DEMgroup .................................... 12
      3.1.14 UTMzone ..................................... 12
      3.1.15 SwitchTo ..................................... 12
   3.2 Evaluation Operators ............................... 13
      3.2.1 Concepts ...................................... 13
      3.2.2 Evaluators .................................... 13
4 Processing Loop 14
   4.1 Processing Commands .............................. 14
      4.1.1 Data Assignment ................................. 14
      4.1.2 List ............................................ 14
      4.1.3 Details ........................................ 15
      4.1.4 Process ........................................ 15
      4.1.5 Select ......................................... 15
      4.1.6 Remove ........................................ 15
      4.1.7 Clear ........................................... 15
      4.1.8 Export ......................................... 16
5 Filtering

5.1 Description ........................................ 17

5.2 Framework Commands ............................... 17
   5.2.1 Filter.Help ................................. 18
   5.2.2 Filter.Add ................................. 18
   5.2.3 Filter.List ................................. 18
   5.2.4 Filter.Report ............................... 18
   5.2.5 Filter.Reset ............................... 18
   5.2.6 Filter.Clear ............................... 18

5.3 Actions ........................................ 18
   5.3.1 Keep ....................................... 18
   5.3.2 EndData .................................... 19
   5.3.3 Skip ....................................... 19
   5.3.4 All ......................................... 19
   5.3.5 Invert ...................................... 19
   5.3.6 Set ......................................... 19
   5.3.7 TimeCut .................................... 19
   5.3.8 Translate ................................... 20
   5.3.9 Scale ...................................... 20
   5.3.10 DEMheight ................................ 20
   5.3.11 GeoRange ................................ 20
   5.3.12 GeoRay ................................... 21
   5.3.13 FieldPoint ................................ 21
   5.3.14 FieldLine ................................ 21

5.4 Filters .......................................... 21
   5.4.1 Area ....................................... 21
   5.4.2 Return ..................................... 22
   5.4.3 Class ...................................... 22
   5.4.4 ScanLine ................................... 22
   5.4.5 Thinout .................................... 22
   5.4.6 Cell ........................................ 24
   5.4.8 FieldMath .................................. 26
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.4.8 FieldMath</td>
<td>26</td>
</tr>
<tr>
<td>5.4.9 Gate</td>
<td>27</td>
</tr>
<tr>
<td>5.4.10 GeoXYZ</td>
<td>28</td>
</tr>
<tr>
<td>5.4.11 GridMath</td>
<td>29</td>
</tr>
<tr>
<td>5.4.12 Jump</td>
<td>30</td>
</tr>
<tr>
<td>5.4.13 Slope</td>
<td>31</td>
</tr>
<tr>
<td>5.4.14 Spline</td>
<td>32</td>
</tr>
</tbody>
</table>

## 6 Graphics System

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Concepts</td>
<td>33</td>
</tr>
<tr>
<td>6.2 Palettes</td>
<td>33</td>
</tr>
<tr>
<td>6.2.1 Concept</td>
<td>33</td>
</tr>
<tr>
<td>6.2.2 Palette.Load</td>
<td>33</td>
</tr>
<tr>
<td>6.3 Graphics Commands</td>
<td>33</td>
</tr>
<tr>
<td>6.3.1 Plot</td>
<td>33</td>
</tr>
<tr>
<td>6.3.2 Plot.List</td>
<td>33</td>
</tr>
<tr>
<td>6.3.3 Plot.Remove</td>
<td>33</td>
</tr>
<tr>
<td>6.3.4 Plot.Clear</td>
<td>33</td>
</tr>
<tr>
<td>6.3.5 Plot.Projection</td>
<td>34</td>
</tr>
<tr>
<td>6.3.6 Plot.Show</td>
<td>34</td>
</tr>
<tr>
<td>6.3.7 Plot.Hide</td>
<td>34</td>
</tr>
<tr>
<td>6.3.8 Plot.Palette</td>
<td>34</td>
</tr>
<tr>
<td>6.3.9 Plot.Colour</td>
<td>34</td>
</tr>
<tr>
<td>6.3.10 Plot.FillMode</td>
<td>35</td>
</tr>
<tr>
<td>6.3.11 Plot.Export</td>
<td>35</td>
</tr>
<tr>
<td>6.3.12 Plot.Stretch</td>
<td>35</td>
</tr>
<tr>
<td>6.3.13 Plot.Rotate</td>
<td>35</td>
</tr>
</tbody>
</table>

## 7 Graphical User Interface

## 8 Data

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1 Concepts</td>
<td>37</td>
</tr>
<tr>
<td>8.2 Timing</td>
<td>37</td>
</tr>
<tr>
<td>8.2.1 DeltaT</td>
<td>37</td>
</tr>
<tr>
<td>8.3 Georeferencing</td>
<td>37</td>
</tr>
<tr>
<td>8.3.1 DeltaXYZ</td>
<td>37</td>
</tr>
<tr>
<td>8.3.2 DeltaPRH</td>
<td>38</td>
</tr>
<tr>
<td>8.4 Data Points</td>
<td>39</td>
</tr>
</tbody>
</table>
8.5 Point Cloud (Memory Buffer) . . . . . . . . . . . . . . . . 40
  8.5.1 Description . . . . . . . . . . . . . . . . . . . . . 40
  8.5.2 Definition . . . . . . . . . . . . . . . . . . . . . . 40
8.6 Point Grid (Memory Buffer) . . . . . . . . . . . . . . . . . 41
  8.6.1 Assignment . . . . . . . . . . . . . . . . . . . . . . 41
  8.6.2 Grid.Align . . . . . . . . . . . . . . . . . . . . . . . 41
  8.6.3 Filter . . . . . . . . . . . . . . . . . . . . . . . . . 41
  8.6.4 Export . . . . . . . . . . . . . . . . . . . . . . . . . 41
  8.6.5 Clone . . . . . . . . . . . . . . . . . . . . . . . . . . 42
  8.6.6 Resample . . . . . . . . . . . . . . . . . . . . . . . . 42
  8.6.7 Combine . . . . . . . . . . . . . . . . . . . . . . . . 42
  8.6.8 Init . . . . . . . . . . . . . . . . . . . . . . . . . . . 42
  8.6.9 FillDEM . . . . . . . . . . . . . . . . . . . . . . . . 43
  8.6.10 Math . . . . . . . . . . . . . . . . . . . . . . . . . . 43
  8.6.11 Set.GridMode . . . . . . . . . . . . . . . . . . . . . 43

9 Data Files 44
  9.1 Concepts . . . . . . . . . . . . . . . . . . . . . . . . . . . 44
  9.2 Definition . . . . . . . . . . . . . . . . . . . . . . . . . . 44
  9.3 Infrastructure Commands . . . . . . . . . . . . . . . . . . 44
    9.3.1 WriteMode . . . . . . . . . . . . . . . . . . . . . . 44
    9.3.2 PathMode . . . . . . . . . . . . . . . . . . . . . . . 44
    9.3.3 GPSoffset . . . . . . . . . . . . . . . . . . . . . . . 44
  9.4 BIL/BSQ data . . . . . . . . . . . . . . . . . . . . . . . . . 46
    9.4.1 Description . . . . . . . . . . . . . . . . . . . . . . 46
    9.4.2 Definition . . . . . . . . . . . . . . . . . . . . . . . 47
    9.4.3 Examples . . . . . . . . . . . . . . . . . . . . . . . . 47
    9.4.4 Export . . . . . . . . . . . . . . . . . . . . . . . . . 48
  9.5 BLN files . . . . . . . . . . . . . . . . . . . . . . . . . . . 49
    9.5.1 Description . . . . . . . . . . . . . . . . . . . . . . 49
    9.5.2 Definition . . . . . . . . . . . . . . . . . . . . . . . 49
    9.5.3 Specific Commands . . . . . . . . . . . . . . . . . . 49
  9.6 BT files . . . . . . . . . . . . . . . . . . . . . . . . . . . . 50
    9.6.1 Description . . . . . . . . . . . . . . . . . . . . . . 50
    9.6.2 Definition . . . . . . . . . . . . . . . . . . . . . . . 50
  9.7 CSV files . . . . . . . . . . . . . . . . . . . . . . . . . . . . 51
    9.7.1 Description . . . . . . . . . . . . . . . . . . . . . . 51
    9.7.2 Definition . . . . . . . . . . . . . . . . . . . . . . . 51


1 General Concepts

rasp (Routines for Airborne Sensor Processing) is a multi-purpose data processing system trying to integrate a number of airborne sensor systems (remote sensing sensors and inertial data systems as well as in-situ measurements) into a common workflow.

As the volume of data generated by some of these systems far exceed the amount which can be held in the main memory of common computing platforms of today (and most probably will for quite some time), the central concept of rasp is that of a 'streaming' processor: One or more input data streams are being processed into an output stream without the necessity to hold more data in core memory than is absolutely necessary at a given time.

In order to realize this concept, rasp implements a number of such streams (either as file import/export facilities or as memory buffers) in conjunction with a highly configurable filter chain. Optionally an OpenGL-based graphics sub-system can be used for visualization and interaction with the data (as it is dynamically allocated, it does not take up any system resources while inactive). In order to compile/use rasp on platforms without the necessary windowing (MS Windows or X11) or graphics (hardware-accelerated OpenGL) support, a non-graphical version can easily be generated.

\[ \text{Table 1: Overview} \]

rasp can operate in one or any combination of three modes: Batch (non-interactive), TUI (text-based user interface), or GUI (graphical user interface). Each level is designed in a way that it operates as a driver for the next lower level - i.e. mouse operations in the GUI generate command lines which are passed to the TUI and then evaluated by the batch processor.
2 Running rasp

2.1 Invocation

`rasp` is typically invoked directly from the command prompt of a text terminal or indirectly by a preprocessor or process management script. Several options can be passed to `rasp` from the command line, e.g. allowing to influence verbosity and fault tolerance.

2.2 Command Line Options

Syntax: `rasp {options} {filename} {parameters}`

Parameters:

- `{options}`
  - `-h` show help message
  - `-v` increase verbosity (can be used multiple times)
  - `-q` decrease verbosity (can be used multiple times)
  - `-g` activate the GUI at startup
  - `-t` increase error tolerance (can be used multiple times)
  - `-l` filename write a log file

- `{filename}` (optional) Name of a `rasp` script file to process. If no file is specified on the command line `rasp` will start up in interactive text interface mode.

- `{parameters}` (optional) Any additional command line parameters are passed on to the script file for which they are accessible as `'#1', '#2',..., '#n'`. 
3 Command Interpreter

The various sub-systems of rasp are accessed through a command interpreter, which evaluates lines of text, which can be read from a script file, typed in at the TUI-prompt or generated by GUI actions. In addition to literal interpretation of the command line, the command interpreter has the following features:

Integrated Preprocessor

Script files read by rasp are 'on-the-fly' pre-processed by PPREP - please refer to its manual for details about its capabilities and syntax.

Partial Match

Commands can be accessed through the first partial match (in alphabetical order), e.g. ‘Q’, ‘QU’ and ‘QUIT’ will all execute the ‘QUIT’ command.

Dot Expansion

Many commands have the format '{prefix}.{postfix}'. A when the command interpreter encounters a command of the form '.{postfix}', it assumes the '{prefix}' part to be identical to the previous command. A singular dot ('.') as a command replicates the previous command.

Constant Expansion

When the command interpreter encounters expressions of the form '@{name}', it searches its constant database (usually pre-loaded from the file 'rasp_constants.txt', if it can be found in the search path (Default: '{current directory};{rasp binary directory}/lib')) for an entry of the form '{name} = {replacement}' and replaces '@{name}' with '{replacement}'. The standard constants file contains shorthand definition for common colour values, LAS class IDs and common filter matrices. User defined values can be added using the 'Set.Constant' command.

Percent Evaluation

Percent expressions of the form 'and replaced by their equivalent expression. For details, refer to the respective section of this manual.

Parameter Expansion

Parameters passed to scripts, either from the command line or in a subroutine call, are available through the codes '{#1, #2, ... #9}', indicating the first to ninth parameter, respectively.

Macro Expansion

When a command '{name} {parameters}' can not be interpreted internally, rasp attempts to find a file '{name}.rsp' in the search path. If a match is found, processing at the current level will be suspended and the new file processed at the next level (i.e. as a subroutine). The '{parameters}' are accessible to the subroutine through parameter expansion.

3.1 Infrastructure Commands

3.1.1 Quit

Syntax: Quit

Description: Terminate processing on the current level and return to the previous level.
If the current level is the base level, 'QUIT' will terminate the program.
If the command prompt was invoked from a script, 'QUIT' will continue
the script where it left off. When a script reaches its end without and
explicit 'QUIT' command, it is implicitly assumed.

3.1.2 Prompt

Syntax: Prompt
Description: Suspend processing at the current level and issue a command line prompt
at the next level. The level of a command prompt is indicated by the
number of '>' prompt characters written to the console, i.e. a base level
prompt will be '{current directory}>', whereas a secondary prompt
invoked from a script will be '{current directory}>>'.

3.1.3 Print

Syntax: Print {string}
Description: Evaluate all expressions and constant definitions in string and print the
result to the assigned output (Default = console).
Parameters:
{string} (string) The text to print

3.1.4 Set.PrintFile

Syntax: Set.PrintFile {filename}
Description: Redirect output of the 'Print' command to a file. If the '{filename}'
field is empty, the console will be used.
Parameters:
{filename} (filename, optional) Name of text output file

3.1.5 About

Syntax: About
Description: Display the version/copyright notice.

3.1.6 Execute

Syntax: Execute {command_line}
Description: Submits a command line to the operating system for execution in the
same processing thread as rasp (i.e. rasp will be suspended until the
external command has finished).
Parameters:
{command_line} (string) Command line to pass to the operating system
Note: rasp does not interpret environment settings (e.g. default path) or in-
build commands usually evaluated by command shells. For example,
the 'DIR' command under Windows needs to be submitted as 'Execute
c:\windows \system 32\cmd .exe dir' in order to work as expected.
3.1.7 Delay

**Syntax:** Delay \{n\}

**Description:** Suspend execution for a number of milliseconds.

**Parameters:**

\(\{n\}\) (numeric, optional) Number of milliseconds to hesitate for (Default=100ms)

3.1.8 CD

**Syntax:** CD \{path\}

**Description:** Change the current directory.

**Parameters:**

\{path\} (string) Absolute or relative path of an existing directory

3.1.9 Repeat

**Syntax:** Repeat \{n\} \{command_line\}

**Description:** Repeat a command line multiple times.

**Parameters:**

\{n\} (numeric) Number of times to repeat the following command  
\{command_line\} (string) The command line to repeat

3.1.10 Wait

**Syntax:** Wait

**Description:** Suspend execution of a script and wait for a key to be pressed.

3.1.11 Help

**Syntax:** Help

**Syntax:** Help \{command\}

**Description:** Help without any specific \{command\} as a parameter lists all the commands the command line interpreter knows about in alphabetical order. If a \{command\} parameter is supplied, a brief usage help for that command is printed to the console.

\{command\} (string, optional) – The command to print help for

3.1.12 DEMpath

**Syntax:** Set.DEMpath \{path\}

**Description:** Set the DEM base path to a new value.

**Parameters:**

\{path\} (string) DEM base path (i.e. where to find the ‘dem_conf.txt’ file)
3.1.13 DEMgroup

Syntax:  
Set.DEMgroup {Group}

Description:  
Set the DEM group to a new value

Parameters:

{Group}  
DEM group (i.e. which section of the 'dem_conf.txt' file to use (Default='*'))

3.1.14 UTMzone

Syntax:  
Set.UTMzone {zone} {N/S}

Description:  
Set the UTM zone for lat/lon operations. This can either be used to prevent zone autodetection (e.g. to ensure a consistent UTM zone for 'border' cases), to manually set a zone where autodetection is not possible (e.g. if the first operation the UTM subsystem has to do is a UTM-Lat/Lon conversion and not the other way round), or to reset autodetection by setting the zone to '0'. Per default rasp assumes the southern hemisphere convention of adding a 'false northing' of 10 million metres as an offset to the northing. For the northern hemisphere this can be overriden by using the 'N' option explicitly.

Parameters:

{zone}  
(numeric,optional) – UTM zone numer ('0' or blank for autodetection)

{N/S}  
(keyword,optional) – Northen/southern hemisphere identifier (Default='S')

3.1.15 SwitchTo

Syntax:  
SwitchTo {filename}

Description:  
Terminate execution of the current file and continue processing at the beginning of another file.

Parameters:

{filename}  
(string) Name of the new file
3.2 Evaluation Operators

3.2.1 Concepts

Evaluators of the form ‘%{evaluator}({data})’ are available in order to use the runtime parameters of one data object during creation and/or processing of another. For example, the line

NewGrid : grid 5 0 %bounds(Cloud1)

would define an empty new grid of 5m cell size using the same East / West / North / South boundaries as the existing ‘Cloud1’. Evaluators can be nested using the common arithmetic bracketing rules.

3.2.2 Evaluators

The following evaluators are available:

<table>
<thead>
<tr>
<th>Evaluator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%minX({data})</td>
<td>minimum easting in UTM metres</td>
</tr>
<tr>
<td>%midX ({data})</td>
<td>central easting in UTM metres</td>
</tr>
<tr>
<td>%maxX ({data})</td>
<td>maximum easting in UTM metres</td>
</tr>
<tr>
<td>%minY ({data})</td>
<td>minimum northing in UTM metres</td>
</tr>
<tr>
<td>%midY ({data})</td>
<td>central northing in UTM metres</td>
</tr>
<tr>
<td>%maxY ({data})</td>
<td>maximum northing in UTM metres</td>
</tr>
<tr>
<td>%Bounds ({data})</td>
<td>Horizontal boundaries in UTM metres</td>
</tr>
<tr>
<td>%minZ ({data})</td>
<td>minimum elevation in metres above WGS84 ellipsoid</td>
</tr>
<tr>
<td>%midZ ({data})</td>
<td>central elevation in metres above WGS84 ellipsoid</td>
</tr>
<tr>
<td>%maxZ ({data})</td>
<td>maximum elevation in metres above WGS84 ellipsoid</td>
</tr>
<tr>
<td>%UTMzone</td>
<td>current rasp UTMzone setting</td>
</tr>
<tr>
<td>%minLon ({data})</td>
<td>minimum longitude in WGS84 degrees</td>
</tr>
<tr>
<td>%maxLon ({data})</td>
<td>maximum longitude in WGS84 degrees</td>
</tr>
<tr>
<td>%minLat ({data})</td>
<td>minimum latitude in WGS84 degrees</td>
</tr>
<tr>
<td>%maxLat ({data})</td>
<td>maximum latitude in WGS84 degrees</td>
</tr>
<tr>
<td>%nPoints ({data})</td>
<td>number of data points</td>
</tr>
<tr>
<td>%nCols ({data})</td>
<td>number of columns of a grid data object</td>
</tr>
<tr>
<td>%nRows ({data})</td>
<td>number of rows of a grid data object</td>
</tr>
<tr>
<td>%dCols ({data})</td>
<td>column spacing of a grid data object</td>
</tr>
<tr>
<td>%dRows ({data})</td>
<td>row spacing of a grid data object</td>
</tr>
<tr>
<td>%rspName</td>
<td>name of the current script file</td>
</tr>
</tbody>
</table>
4 Processing Loop

The processing loop is the central element of *rasp*: data is read from a data source (files or memory buffers), piped through the (potentially empty) filter chain, and written to a data target (which again can be memory buffers as well as files).

In order to process data in this way, *rasp* needs to be aware of the various parameters of the input/output data streams it works with. This is achieved through 'assignment' of a group of attributes (e.g. file name format identifiers, etc.) to a named entry in a lookup table. This assignment can either be done explicitly through the assignment operator ':=' or implicitly as part of the processing operators '=' and '<'. Additionally, the associated names can either be supplied explicitly or automatically generated by the system (i.e. when they are omitted, or through the '*' default identifier in those places where omission is not possible).

4.1 Processing Commands

4.1.1 Data Assignment

**Syntax:** `{Name} : {Data Definition}`

**Description:** Creates or overwrites an entry in the data lookup table. The data definition can be supplied in full or in an abbreviated format. If the latter is chosen, *rasp* attempts to infer the missing information from the available facts (e.g. the file format of an existing file from its extension).

{Name} *(string,optional) – Name to use for the assignment. Although theoretically any character is possible, it is recommended to use an alphanumeric name with a letter as the leading character (Default = Automatically generated as a string of the letter 'D' followed by a uniquely identifying four digit number with leading zeros). The name has to be unique amongst data names, plot and keywords.*

{Data Definition} *(Possible Formats):*

- Full Definition `{Data Type} {File Name} {Type Specific Definitions}`
- or `{File Name} {Type Specific Definitions}`
- or `{File Name} {Type Specific Definitions}`

**Parameters:**

{Data Type} *(keyword,optional) – One of the supported data type identifiers (see next section for details, Default= 'CLOUD'). If no data type is stated for an existing file, *rasp* will infer the data type from the file names extension. For new files, the data type always has to be stated explicitly.*

{File Name} *(string,optional) – The file name of those data types associated with a file, omitted for memory buffers.*

{Type Specific Definitions} *(string,optional) – Any further information is not evaluated by the ASSIGN command, but passed on to the sub-module for that specific data type. Please refer to the format-specific sections of the appendix for details.*

4.1.2 List

**Syntax:** `List`
4 Processing Loop

Description: Lists an overview of all current data assignments. The current default object is indicated by an ‘*’. The default object usually is the last one accessed by any object-specific command. It can be selected by using the data name ‘*’ or in most cases by omitting the data name altogether.

4.1.3 Details

Syntax: Details {Data Name}

Description: Lists the details for a specific data object.

{Data Name} (string,optional) – Name of the data object to provide details for (Default=Default object).

4.1.4 Process

Syntax: Process {Source} {Target} {Additional Definitions}

Description: Start the main processing loop, reading data points from {Source}, passing them through the filter chain (see section "Filter Chain" for details), and writing them to {Target}. {Source} and {Target} can either be referenced by data names assigned through ASSIGN (see above) or as an implicit assignment following the same rules as described for ASSIGN. Incomplete implicit assignments need to be terminated by a semicolon (‘;’) in order to ensure correct default values to be assigned to missing parameters.

{Source} (string) – Name or definition of the data source (‘*’ can be used to select the default object).

{Target} (string) – Name or definition of the data target (‘*’ can be used to generate a new auto-named cloud buffer). At the end of the processing loop, the target will be the new default object. This allows to build processing chains using auto-naming without having to worry about the names of the data objects at intermediate stages.

4.1.5 Select

Syntax: Select {Data Name}

Description: Select a specific data object to be the default object.

{Data Name} (string) – Name of the data object to select as new default object.

4.1.6 Remove

Syntax: Remove {Data Name}

Description: Remove a specific data object from the lookup table and release its associated memory (i.e. Remove will not change/delete any files, but only remove the internal reference to them).

{Data Name} (string,optional) – Name of the data object to clear (Default=Default object).

4.1.7 Clear

Syntax: Clear

Description: Remove a all data objects from the lookup table and release all associated memory.
4.1.8 Export

Syntax: Export {Data} {Export Definition}
Description: Export a data object to a file.

4.1.9 Import

Syntax: Import {Target} {Source} {Options}
Description: This command performs a block based transfer (as opposed to the normal stream based operation of \texttt{rasp}) from one data object to another, i.e. the state of all objects involved has to be static before this command is invoked, and no filtering/processing of individual data points takes place during the transfer.

The normal purpose of this command would be to import data from a file \texttt{Source} into a compatible (e.g. gridded objects can only be imported from gridded file formats) memory-based object \texttt{Target}. Format-specific Options are evaluated by the specific handler for each.

Note: Only certain combinations of data objects allows successful import/export. ThReport system/memory status.

4.1.10 Status

Syntax: Status
Description: Report system/memory status.

4.1.11 Inspect

Syntax: Inspect {name} {start_index} {n_points} {format}
Description: Inspect the properties of one or more data point(s).

Parameters:

{name} (numeric,optional) Index of the first point to inspect (A negative values denotes that this index is relative to the end of a data object. Default=1)

{start_index} (numeric,optional) Number of points to inspect (Default=1)

{n_points} (string,optional) Format to use for the output, using the same nomenclature as the ASCII data file output (Default=all fields by ID in alphabetical sequence, separated by commas)

4.1.12 Constant.Set

Syntax: Constant.Set {name} {value}
Description: Create/modify the named constant '{name}' with a content of '{value}'.

Parameters:

{name} (filename,optional) – name of the constant

4.1.13 Constant.List

Syntax: Constant.List
Description: List all named constants.
5 Filtering

5.1 Description

If the filter operator '|/' is present in a data transfer statement, e.g. `out =| in'`, all data points have to pass through the raspi filtering system on their way from the data source to the target. Apart from filtering out unwanted data points, this system can perform a number of actions based on the characteristics of the data points being processed.

The filtering system is implemented as a freely configurable 'filter chain', similar in philosophy to a distillery column in chemistry. Using the `Filter.Add` command, a custom chain of individually configured filters can be designed to suit a particular purpose.

At processing time, a data point enters the filter chain with its 'state' flag set and is passed sequentially from one filter stage to the next. Each stage performs actions ('Actions') depending on the current state flag and/or modifies the state flag depending on a processing result generated from the data point ('Filters').

As memory and data structures for the filter chain are dynamically allocated at runtime, there is no limitation for the number and complexity of the filters which can be added to the chain (apart from system memory resources).

![Diagram of filter classes](image)

**Table 2:** Available Classes of Filter "Building Blocks"

5.2 Framework Commands

The filter chain is maintained by the means of three framework commands, allowing to add stages to the chain, and to inspect or clear the chain. As the filter chain remains in place as defined until cleared, a common cause of unexpected processing results is forgetting to clear the filter chain when it is no longer needed.
5.2.1 Filter.Help

Syntax: Filter.Help
Description: Generate a short list of the available filters.

5.2.2 Filter.Add

Syntax: Filter.Add {-}{Filter}
Description: Add a filter or action to the end of the chain. See below for details on specific elements. If the filter definition is preceded by a `-`, the interpretation of the state flag by that stage is inverted.

Parameters:

{-} (optional) Inversion indicator
{Filter} Filter or Action definition

5.2.3 Filter.List

Syntax: Filter.List
Description: Lists the current filter chain.

5.2.4 Filter.Report

Syntax: Filter.Report
Description: Report what effect the filter chain had on the last processing command.

5.2.5 Filter.Reset

Syntax: Filter.Reset
Description: Reset the filter statistics and put adaptive filters back into their initial state.

5.2.6 Filter.Clear

Syntax: Filter.Clear
Description: Removes all filters from the chain.

5.3 Actions

This section describes a category of filter stages named 'Actions', because they do evaluate the individual data points, but perform actions depending on the state flag.

5.3.1 Keep

Syntax: Filter.Add Keep
Description: Terminates the chain for all points with a set filter flag. This action indicates to the system that all points which still have their state flag set should not be processed any further in the chain (i.e. kept). In normal usage, this action would be followed by an 'All' action in order to continue filtering with different criteria. `-Keep` can be used to shortcut the filter chain for all points which have their state flag cleared.
5.3.2 EndData

Syntax:  Filter.Add EndData

Description: Terminates the processing loop when it encounters a sample with a set filter flag. Normally this action is most useful in it’s inverted form ‘-EndData’ as a termination trigger behind e.g. time or scanline gate filter in order to speed up processing of long data files.

5.3.3 Skip

Syntax:  Filter.Add Skip {nSkip}

Description: Skips the next nSkip stages for all points with a set state flag. This action allows to create multi-pathway filter chains, which apply different secondary filtering to pre-selected data points.

Parameters:
{nSkip} (numeric,optional) – Number of stages to skip (default=1)

5.3.4 All

Syntax:  Filter.Add All

Description: Sets the filter flag of all points still in the chain. This action allows filtering ‘to start from scratch’, e.g. after pre-selected points have already been discarded by a ‘-Keep’ action.

5.3.5 Invert

Syntax:  Filter.Add Invert

Description: Inverts the filter flag of all points still in the chain. The inversion modifier ‘-’ is ignored by this action, as ‘-Invert’ would be a ‘no operation’.

5.3.6 Set

Syntax:  Filter.Add Set {what} {value}

Description: Sets a field of all point samples with a set state flag to a new value. The available field identifiers are described in the ’Data’ section of this manual. A usage example of this action would be ‘Filter.Add Set Class *ground’ to set the class ID of all pre-filtered points to ‘Ground’ (the constant ‘*ground’ is defined in the standard ’constants.txt’ file as the value ‘2’).

Parameters:
{what} (keyword) – Identifier of the field to set (default=’Class’
{value} (numeric) – Value to set it to

5.3.7 TimeCut

Syntax:  Filter.Add TimeCut {cut_1} {cut_2} ... {cut_n}

Description:

Parameters:
{start_n} (numeric,optional) – Start time
5.3.8 Translate

Syntax: Filter.Add Translate {dx} {dy} {dz}

Description: Translates all points in the chain with a set filter flag by a linear offset.

Parameters:
- {dx} (numeric, optional) – offset in X direction [m] (default=0)
- {dy} (numeric, optional) – offset in Y direction [m] (default=0)
- {dz} (numeric, optional) – offset in Z direction [m] (default=0)

5.3.9 Scale

Syntax: Filter.Add Scale {fx} {fy} {fz}

Description: Scales all points in the chain with a set filter flag by a factor.

Parameters:
- {fx} (numeric, optional) – scale factor in X direction (default=1)
- {fy} (numeric, optional) – scale factor in Y direction (default={fx})
- {fz} (numeric, optional) – scale factor in Z direction (default={fy})

5.3.10 DEMheight

Syntax: Filter.Add DEMheight {clipvalue}

Description: The height (z) field of the data point is replaced with DEM-derived values. In order to select a suitable DEM, RASP will parse a list of available DEMs (‘dem_conf.txt’) in the directory defined by ‘SET.DEMbase’ (Default=’/ara/dem’). The first matching DEM will be selected unless search is limited to a certain DEM group, e.g. ‘srtm30’.

Parameters:
- {clipvalue} (numeric, optional) – threshold level underneath which the DEM is clipped to zero (e.g. use a clip value of ‘0’ to exclude the SRTM30 bathymetry data (Default=-19999, i.e. no clipping)).
- {group} (keyword, optional) – limit DEM search to a named group of DEM files.

Note: The UTMzone (see SET.UTMzone command) needs to be set explicitly for the DEMheight action to work properly if no previous lat/lon operations (implicit or explicit) in the same zone have been performed.

5.3.11 GeoRange

Syntax: Filter.Add GeoRange {INUdata}

Description: Converts the position information of each data point from polar aircraft coordinates (Range, Theta) to earth coordinates using the reference point and attitude information contained in an inertial navigation data stream.

Synchronization between data points and INU data is achieved using the UTC second-of-day field, assuming a strictly monotonous increase in both. Intermediate values for position and attitude are calculated using linear interpolation.

Parameters:
- {INUdata} (keyword) – Identifier defining and/or referencing inertial navigation data
5.3.12 GeoRay

Syntax:
Filter.Add GeoRay {-}{INUdata} {DEMsource}

Description:
Estimate position information in earth coordinates for each data point
by ray-casting a vector in aircraft coordinates (Theta) onto a suitable
DEM (using the reference point and attitude information contained in
an inertial navigation data stream).

Synchronization between data points and INU data is achieved using
the UTC second-of-day field, assuming a strictly monotonous increase in
both. Intermediate values for position and attitude are calculated using
linear interpolation.

Parameters:
{-} (keyword,optional) – If present, this indicates that the sensor flew 'back-
wards'
{INUdata} (keyword) – Identifier defining and/or referencing inertial navigation data
{DEMsource} (keyword) – Identifier defining and/or referencing digital elevation data

5.3.13 FieldPoint

Syntax: Filter.Add FieldPoint {Data} {FieldFrom} {FieldTo}

Description:
Set a field of each data point passing through this action to the value of a
selected field of another data stream. If the end of this stream is reached,
the stream is rewound and reading resumed from the start.

Parameters:
{INUdata} (keyword) – Identifier defining and/or referencing inertial navigation data
{DEMdata} (keyword) – Identifier defining and/or referencing digital elevation data

5.3.14 FieldLine

Syntax: Filter.Add FieldLine {Data} {FieldFrom} {FieldTo}

Description:
Set a field of each data point passing through this action to the value of a
selected field of another data stream. If the end of this stream is reached,
the stream is rewound and reading resumed from the start.

Parameters:
{INUdata} (keyword) – Identifier defining and/or referencing inertial navigation data
{DEMdata} (keyword) – Identifier defining and/or referencing digital elevation data

5.4 Filters

5.4.1 Area

Syntax: Filter.Add Area {method} {definition}

Description:
Filters out points enclosed in a polygonal region. This region can be
defined either interactively (in GUI mode) using the mouse digitizer buffer,
read from a BLN point list file, read from a named data object, or defined
on the command line. rasp uses a fast 'winding number' algorithm to
determine if a point is inside our outside the polygon. While for most
polymgons, this algorithm gives the same result as the 'edge intersection'
method, classification is slightly different for self-intersecting outlines.

Parameters:

{method} (keyword) – 'MOUSE' reads the polygon from the mouse digitizer buffer
(if active), 'BLN' interprets the {'definition'} field as the file and
section names of a BLN file. Any other values are checked against the list
of currently assigned data objects, and if a match is found the polygon is
initialized from that object (interpreting 'point clouds' as vector lists and
'grids' as their rectangular extent). If none the above has succeeded, the
corners of the polygon are read as a space separated list of (x/y) pairs
from the command line.

5.4.2 Return

Syntax: Filter.Add Return {which}
Description: Filters out points which are corresponding to the first or last return, or
are single returns, from any given LiDAR shot.

Parameters:

{which} (keyword,optional) – 'FIRST', 'LAST' or 'SINGLE' (Default='LAST')

5.4.3 Class

Syntax: Filter.Add Class {class list}
Description: Filters out points which are in a given list of class IDs.

Parameters:

{class list} (numeric) – comma-separated list of class IDs (a text representation of
common class IDs is defined in constants.txt)

5.4.4 ScanLine

Syntax: Filter.Add ScanLine {start line} {end line}
Description: Filters out points which are within a given range of scan lines (relative to
the start of the data). As the scan angle backtrace is used to determine
the scan line count, this filter will only work reliably on data which a)
contains scan angle values and b) is still in the original chronological
sequence.

Parameters:

{start line} (numeric,optional) – first scan line to let through (Default=0)
{end line} (numeric,optional) – last scan line to let through (Default=MaxInt)

5.4.5 Thinout

Syntax: Filter.Add Thinout {nline} {npoint} {line offset} {point offset}
Description: Thin out scanner data to each nth line/point, using an optional start
offset.

Parameters:

{nline} (numeric,optional) – line count interval (Default=1, i.e. use each line)
{npoint} (numeric,optional) – point count interval (Default=1, i.e. use each point)
{line offset} (numeric,optional) – start offset for the first line to use (Default=0)
{point offset} (numeric,optional) – start offset for the first point to use (Default=0)
5.4.6 Cell

Syntax: Filter.Add Cell {Grid}

Description: Filters out points which do not fall into a valid cell of {Grid}.

Parameters:

{Grid} The grid to filter against
5.4.7 FieldMath

**Syntax:** Filter.Add FieldMath `{target_field}` `{rpn_expression}`

**Description:** Calculates an RPN (reverse polish notation) expression using space-separated field identifiers, constants, and operators (`'+','-','*','/`). The result is stored in the field specified by `{target_field}`. RPN notation is being used because it can easily be converted into a linear reference list at definition time, which can be 'replayed' very efficiently at execution time. As a side-effect, errors (e.g. under/overflows of the RPN stack will only be detected at execution time). The maximum allowed stack depth is 8 elements, which should be sufficient for all practical calculations (i.e. any expressions requiring a higher stack depth could/should be split up into smaller sub-expressions).

**Parameters:**

- `{target_field}` Field identifier of the target field
- `{rpn_expression}` The expression to evaluate, e.g. `red green - 256 * red green + /` to generate an 8-bit differential index value from the 8-bit channels 'red' and 'green'

**Note:** 'FieldMath' uses 32bit signed integer maths and does not check for potential rounding errors (in particular when using division) or overflows (e.g. when using multiplication) - it is the user's responsibility to formulate `{rpn_expression}` in an appropriate way. Please be aware that multiplication and division generally do not commute when using integer maths, e.g. the expressions 'red 100 * 20 /' and 'red 20 / 100 *' do have different results.
5.4.8 FieldMath

**Syntax:** Filter.Add FieldMath {target_field} {rpn_expression}

**Description:** Calculates an RPN (reverse polish notation) expression using space-separated field identifiers, constants, and operators (‘+’, ‘-’, ‘*’, ‘/’). The result is stored in the field specified by ‘{target_field}’. RPN notation is being used because it can easily be converted into a linear reference list at definition time, which can be 'replayed' very efficiently at execution time. As a side-effect, errors (e.g. under/overflows of the RPN stack will only be detected at execution time). The maximum allowed stack depth is 8 elements, which should be sufficient for all practical calculations (i.e. any expressions requiring a higher stack depth could/should be split up into smaller sub-expressions).

**Parameters:**

- `{target_field}`: Field identifier of the target field
- `{rpn_expression}`: The expression to evaluate, e.g. `red green - 256 * red green + /` to generate an 8-bit differential index value from the 8-bit channels 'red' and 'green'

**Note:** 'FieldMath' uses 32bit signed integer maths and does not check for potential rounding errors (in particular when using division) or overflows (e.g. when using multiplication) - it is the user's responsibility to formulate `{rpn_expression}` in an appropriate way. Please be aware that multiplication and division generally do not commute when using integer maths, e.g. the expressions 'red 100 * 20 /' and 'red 20 / 100 *' do have different results.
5.4.9 Gate

**Syntax:**  Filter.Add Gate {min} {max} {What} {Grid} {Method}

**Description:** Filters out points with a parameter in a given value range, optionally relative to an offset grid.

**Parameters:**

- `{min}` (numeric, optional) – minimum value (Default=0)
- `{max}` (numeric, optional) – maximum value (Default=99999)
- `{What}` (keyword, optional) – Field identifier which point component to analyze (Default=ALTITUDE)
- `{Grid}` if present, the min/max values are interpreted as relative to interpolated grid values for the data point location.
- `{Mode}` (keyword, optional) – Interpolation algorithm to use: FLAT, LINEAR or CUBIC (Default=CUBIC)
5.4.10 GeoXYZ

**Syntax:** Filter.Add GeoXYZ {INUdata}

**Description:** Converts the position information of each data point from linear aircraft coordinates (X,Y,Z) to earth coordinates using the reference point and attitude information contained in an inertial navigation data stream.

Synchronization between data points and INU data is achieved using the UTC second-of-day field, assuming a strictly monotonous increase in both. Intermediate values for position and attitude are calculated using linear interpolation.

**Parameters:**

{INUdata} (keyword) – Identifier defining and/or referencing inertial navigation data
5.4.11 GridMath

**Syntax:** \texttt{Filter.Add Gridmath \{Grid\} \{Sign\} \{What\} \{Mode\}}

**Description:** This action modifies a selected field of the data point by adding or subtracting a value interpolated according to its \((x,y)\) position within a grid.

**Parameters:**

\{Grid\} (data) – The grid to use

\{Sign\} (keyword) – Sign of the operation: (+) or (-)

\{What\} (keyword,optional) – Field identifier on which point component to operate (Default=\texttt{ALTITUDE})

\{Mode\} (keyword,optional) – Interpolation algorithm to use: \texttt{FLAT}, \texttt{LINEAR} or \texttt{CUBIC} (Default=\texttt{CUBIC})
5.4.12 Jump

**Syntax:**  Filter.Add Jump \{maxJump\} \{What\}

**Description:** Filters outlier pixels which differ more than a jump threshold from their predecessor. This filter will only work reliably (or at least, as designed) when used on data which are still in their original chronological sequence.

**Parameters:**

\{maxJump\} (numeric,optional) – maximum tolerable jump from one pixel to the next
( Default=50)

\{What\} (keyword,optional) – Field identifier which point component to analyze
( Default=\texttt{RANGE})
5.4.13 Slope

Syntax: \texttt{Filter.Add Slope \{max\_slope\} \{n\_reset\}}

Description: Calculates the vector between the current point and the previous point and compares it with vector between the last two valid points. If the angle between the two exceeds a maximum slope, the point is filtered out. A count \'{n\_reset}’ can be specified in order to reset the filter after a certain number of consecutive points have been filtered out.

Parameters:

\begin{itemize}
  \item \texttt{\{max\_slope\}} (numeric, optional) maximum tolerable slope in degrees (default=25)
  \item \texttt{\{n\_reset\}} (numeric, optional) number of consecutively discarded points after which the filter will reset (default=5)
\end{itemize}
5.4.14 Spline

Syntax: \texttt{Filter.Add Spline \{tension\} \{min\} \{max\} \{field\}}

Description: Calculates the vector direction between the current point and the previous point and compares it either with the horizontal plane (absolute) or the vector between the last two valid points (relative).

Parameters:
\{min\_slope\}
\{field\} (keyword,optional) – Field identifier which point component to analyze (Default=\texttt{RANGE})
6 Graphics System

6.1 Concepts

This section describes the *rasp* graphics subsystem.

6.2 Palettes

6.2.1 Concept

6.2.2 Palette.Load

**Syntax:** Palette.Load {name} {filename}

6.3 Graphics Commands

6.3.1 Plot

**Syntax:** Plot {plot}

or

**Syntax:** Plot {plot definition}

**Description:** Select an existing plot to be the work plot, or create a new plot as the work plot.

**Parameters:**

{plot} Name of the plot to select as work plot

{plot definition} Definition for a new plot

6.3.2 Plot.List

**Syntax:** Plot.List

**Description:** List all currently defined plots.

6.3.3 Plot.Remove

**Syntax:** Plot.Remove {plot}

**Description:** Permanently remove a plot from the display subsystem. If the cleared plot is the current work plot, the current work plot is set to ‘undefined’ by this command.

**Parameters:**

{plot} (keyword,optional) – Name of the plot to remove (Default: current work plot)

6.3.4 Plot.Clear

**Syntax:** Plot.Clear

**Description:** Permanently remove all plots from the display subsystem.
6.3.5 Plot.Projection

**Syntax:** `Plot.Projection {projection}

**Description:** Show a previously hidden plot again.

**Parameters:**

{projection} (keyword,optional) – Selects the projection style of the display window (Default: 3D perspective view). The available values are:

- **3D**: 3D perspective view
- **TOP**: 2D orthographic view from above
- **EAST**: 2D orthographic view in X-direction towards the east
- **NORTH**: 2D orthographic view in Y-direction towards the north

6.3.6 Plot.Show

**Syntax:** `Plot.Show {plot}

**Description:** Show a previously hidden plot again.

**Parameters:**

{plot} (keyword,optional) – Name of the plot to show (Default: current work plot)

6.3.7 Plot.Hide

**Syntax:** `Plot.Hide {plot}

**Description:** Hide a plot from the display (but don’t delete it).

**Parameters:**

{plot} (keyword,optional) – Name of the plot to hide (Default: current work plot)

6.3.8 Plot.Palette

**Syntax:** `Plot.Palette {palette} {plot}

**Description:** Set the palette associated with a plot.

**Parameters:**

{palette} (keyword,optional) – Name of the palette to use (Default: current work palette)

{plot} (keyword,optional) – Name of the plot to apply the palette to (Default: current work plot)

6.3.9 Plot.Colour

**Syntax:** `Plot.Colour {Source} {Scale} {Offset} {plot}

**Description:** Set the colouring mode associated with a plot.

**Parameters:**

{palette} (keyword,optional) – Name of the palette to use (Default: current work palette)

{plot} (keyword,optional) – Name of the plot to apply the palette to (Default: current work plot)
6.3.10 Plot.FillMode

**Syntax:** Plot.FillMode {mode} {plot}

**Description:** Set the fill mode associated with a plot.

**Parameters:**

- `{mode}` (keyword, optional) – The fill mode can be either ’MESH’ or ’SURFACE’ (Default: ’SURFACE’)
- `{plot}` (keyword, optional) – Name of the plot to apply the fill mode to (Default: current work plot)

6.3.11 Plot.Export

**Syntax:** Plot.Export {filename} {format}

**Description:** Save the current graphics display to a BMD (raster) or PostScript (Vector) file.

- `{filename}` (keyword, optional) – Name of file to write to (Default=’rasp_export.bmp’)
- `{format}` (keyword, optional) – One of ’BMP’ or ’PS’ (Default=implied from filename extension)

6.3.12 Plot.Stretch

**Syntax:** Plot.Stretch {+/-}  

or

**Syntax:** Plot.Stretch {factor}

**Description:** Apply/modify a stretch factor to the Z scaling of the display.

- `{+/-}` Doubles ’+’ or halves ’-’ the stretch factor
- `{factor}` (numeric, optional) – Explicitely set the strech factor to a value (Default=1)

6.3.13 Plot.Rotate

**Syntax:** Plot.Rotate {+/-} {ax} {ay} {az}

**Description:** Rotate the current graphics view about the X,Y,Z axis to given angles, or increment decrement current angles.

- `{+/-}` (optional) if present, the current rotation is incremented/decremented instead of set
- `{ax}` (numeric, optional) – angle to rotate around the x-axis in degrees (Default=0)
- `{ay}` (numeric, optional) – angle to rotate around the y-axis in degrees (Default=0)
- `{az}` (numeric, optional) – angle to rotate around the z-axis in degrees (Default=0)
7 Graphical User Interface
8 Data

8.1 Concepts
The ‘atomic’ data element in rasp is the ‘DataPoint’, an internal data record representing an envelope format for the available information from airborne scanning instruments.

8.2 Timing
All times in rasp are usually referenced to the previous UTC midnight of the source file in the processing loop. For combined processing with auxiliary files (e.g. attitude/navigation) files which start on different UTC days a time offset can be defined (see also ‘Set.TimeOffset’ command), e.g. to let rasp know that it should assume that an attitude/navigation file starts on the day before the file it is used to process, the command ‘Set.TimeOffset *daybefore’ would be appropriate.

8.2.1 DeltaT

Syntax:
Set.DeltaT {value} Add.DeltaT {value}

Description:
Set/Add to the time offset for position/attitude interpolation. This offset is applied whenever attitude and position data are accessed in this capacity (i.e. it is not applied when position points are accessed as discrete points. The 'Translate' filter action can be used to permanently offset discrete point values). A time offset of 86400 seconds (or the named constant ‘*DayBefore’ needs to be specified for data beginning on different GPS days.

Parameters:
{value} (numeric,optional) – Time offset in seconds (Default=0)

8.3 Georeferencing
Internally rasp maintains all geographic values as UTM coordinates in integer centimetres. The horizontal reference datum is AGD94; the vertical values are relative to the WGS84 ellipsoid.

8.3.1 DeltaXYZ

Syntax:
Set.DeltaXYZ {dx} {dy} {dz} Add.DeltaXYZ {dx} {dy} {dz}

Description:
Set/Add to the position offset for position/attitude interpolation. This offset is applied whenever attitude and position data are accessed in this capacity (i.e. it is not applied when position points are accessed as discrete points. The 'Translate' filter action can be used to permanently offset discrete point values).

Parameters:
{dx} (numeric,optional) – offset in positive X direction in metres (East; default=0)
{dy} (numeric,optional) – offset in positive Y direction in metres (North; default=0)
{dz} (numeric,optional) – vertical offset in metres (Up; default=0)
8.3.2 DeltaPRH

**Syntax:**
Set.DeltaPRH \{dPitch\} \{dRoll\} \{dHeading\}  
Add.DeltaPRH \{dPitch\} \{dRoll\} \{dHeading\}

**Description:** Set/Add to the offset angles for position/attitude interpolation. This offset is applied whenever attitude and position data are accessed.

**Parameters:**

- \{dPitch\} (numeric, optional) – pitch offset in degrees (Default=0)
- \{dRoll\} (numeric, optional) – roll offset in degrees (Default=0)
- \{dHeading\} (numeric, optional) – heading offset in degrees (Default=0)
### 8.4 Data Points

The following identifiers are available:

<table>
<thead>
<tr>
<th>Evaluator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EASTING</td>
<td>Easting in UTM metres</td>
</tr>
<tr>
<td>NORTHING</td>
<td>Northing in UTM metres</td>
</tr>
<tr>
<td>ALTITUDE</td>
<td>Altitude in metres above WGS84 ellipsoid</td>
</tr>
<tr>
<td>RANGE</td>
<td>Range from instrument in metres</td>
</tr>
<tr>
<td>RED</td>
<td>Amplitude of 'Red' component</td>
</tr>
<tr>
<td>GREEN</td>
<td>Amplitude of 'Green' component</td>
</tr>
<tr>
<td>BLUE</td>
<td>Amplitude of 'Blue' component</td>
</tr>
<tr>
<td>AMPLITUDE</td>
<td>Amplitude of main component (i.e. usually the Lidar pulse)</td>
</tr>
<tr>
<td>WIDTH</td>
<td>Half-peak width of the Lidar pulse in 1/10 nanoseconds</td>
</tr>
<tr>
<td>ITARGET</td>
<td></td>
</tr>
<tr>
<td>CLASS</td>
<td></td>
</tr>
<tr>
<td>NTARGETS</td>
<td></td>
</tr>
<tr>
<td>SOURCEID</td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td></td>
</tr>
<tr>
<td>PITCH</td>
<td></td>
</tr>
<tr>
<td>ROLL</td>
<td></td>
</tr>
<tr>
<td>HEADING</td>
<td></td>
</tr>
<tr>
<td>RGB</td>
<td></td>
</tr>
<tr>
<td>NDVI</td>
<td></td>
</tr>
<tr>
<td>INDEX</td>
<td></td>
</tr>
<tr>
<td>ROW</td>
<td></td>
</tr>
<tr>
<td>COLUMN</td>
<td></td>
</tr>
<tr>
<td>SET</td>
<td></td>
</tr>
<tr>
<td>NADIR</td>
<td></td>
</tr>
</tbody>
</table>
8.5 Point Cloud (Memory Buffer)

8.5.1 Description

8.5.2 Definition
8.6 Point Grid (Memory Buffer)

8.6.1 Assignment

Syntax:  {Assign} {name} grid {nx} {ny} {west} {east} {south} {north}

Description: Declares a named data buffer to be of a 'grid' type

Parameters:

- {nx} (numeric,optional) – number of grid columns or desired grid spacing in metres (if no other parameters are present)
- {ny} (numeric,optional) – number of grid rows
- {west} (numeric,optional) – minimum UTM easting of grid
- {east} (numeric,optional) – maximum UTM easting of grid
- {south} (numeric,optional) – minimum UTM northing of grid
- {north} (numeric,optional) – maximum UTM northing of grid

Note: All parameters are optional - they default to 'reasonable' values derived from the data source (if available) during the 'prepare' phase of the processing loop.

8.6.2 Grid.Align

Syntax: Grid.Align {grid}

Description: Aligns the x/y information stored in the individual grid pixels with the actual location of the grid cell centres.

Parameters:

- {Grid} (data) – The grid to align

8.6.3 Filter

Syntax: Grid.Filter {Grid} {Field} {Mode} {Matrix}

Description: Modify a grid by applying a linear convolution filter, defined by a centered matrix of filter weights. Filter matrices for common tasks (e.g. gap filling and smoothing) are defined as constants in 'rasp_constants.txt'.

Parameters:

- {Grid} (data) – The grid to work on
- {Field} (keyword,optional) – Field identifier on which point component to operate (Default=ALTITUDE)
- {Mode} (keyword,optional) – Select the command to work on 'GAPS' only, or 'ALL' data (Default=GAPS)
- {Matrix} (numeric) – The filter matrix as a sequence of space-separated values, with rows separated by ';

8.6.4 Export

Syntax: Grid.Export {Grid} {Filename} {What} {Format}

Description: Export gridded data to a Golden Software Surfer GRD file (either binary or ASCII).

Parameters:
8.6.5 Clone

Syntax: Grid.Clone {Grid1} {Grid2}

Description: Create an identical copy of a grid.

Parameters:

{Grid1}
{Grid2}

8.6.6 Resample

Syntax: Grid.Resample {Grid1} {Grid2} {Field} {mode}

Description:

Parameters:

{Grid1}
{Grid2}

8.6.7 Combine

Syntax: Grid.Combine {Grid1} {Grid2} {What} {Condition}

Description: Overwrites cells in Grid1 with those from Grid2 if they have a higher 'nHits' count, i.e. if the number of original data points for that cell was higher. If set to 'IGNORE', only points which are not defined in Grid1 will be filled with the equivalent ones from Grid2, if they are defined.

Parameters:

{Grid1}
{Grid2}

8.6.8 Init

Syntax: Grid.Init {Grid} {Data}

Description:

Parameters:

{Grid}
{Data}
8.6.9 FillDEM

Syntax: Grid.FillDEM {Grid}

Description:

Parameters:

{Grid}

{Data}

8.6.10 Math

Syntax: Grid.Math {Grid1} {Grid2} {Factor} {Field}

Description:

Parameters:

{Grid1}

{Grid2}

8.6.11 Set(GridMode)

Syntax: Set.GridMode {Mode}

Description: Override the gridding mode for data points written to the grid (Default=MinZ)

Parameters:

{Mode} (keyword, optional) – One of (’First, Last, MinZ, MaxZ, Mean, SDev’) (Default=MinZ)
9 Data Files

9.1 Concepts

Data files in *rasp* are treated as a specific subset of general data objects, i.e. like the internal buffers they are accessible through a handle which can either be explicitly named or is automatically assigned a unique name as an identifier.

9.2 Definition

**Syntax:**
```
{name} : {ID} {filename} {format} {block size} {start index} {end index}
```

**Parameters:**
- `{name}` (keyword,optional) – The name to use for the file handle
- `{ID}` (keyword,optional) – A file type identifier can be used to override automatic file type deduction
- `{filename}` (filename) – name of the data file
- `{format}` (keyword,optional) – a textual descriptor which is passed to the handlers of the individual file formats for detailed setup
- `{block size}` (numeric,optional) – block size (in data points) of individual files (Default = 0 = no blocking)
- `{start index}` (numeric,optional) – first index to use for blocked files (Default = 1)
- `{end index}` (numeric,optional) – first index to use for blocked files (Default = 1)

9.3 Infrastructure Commands

9.3.1 WriteMode

**Syntax:**
```
Set.WriteMode {Mode}
```

**Description:** This parameter controls the behaviour of *rasp* when it attempts to write to an existing file.

**Parameters:**
- `{Mode}` (keyword,optional) – One of ‘ASK,OVERWRITE,APPEND,FAIL’ (Default=’ASK’)

9.3.2 PathMode

**Syntax:**
```
Set.PathMode {Mode}
```

**Description:** This parameter controls the behaviour of *rasp* when it attempts to write a file into a non-existent directory.

**Parameters:**
- `{Mode}` (keyword,optional) – One of ‘ASK,CREATE,FAIL’ (Default=’ASK’)

9.3.3 GPSoffset

**Syntax:**
```
Set.GPSoffset {seconds}
```

**Description:** Override the default offset value (15 seconds) between UTC and GPS time.

**Parameters:**
\{seconds\} (keyword, optional) – Set the UTC/GPS offset in seconds (Default = 15)
9.4 BIL/BSQ data

9.4.1 Description

The BIL (Band Interleaved Lines) and BSQ (Band Sequential Lines) formats can be used by rasp to import/export gridded georeferenced data as well as non-georeferenced data from 'pushbroom' type scanners (i.e. instruments registering complete lines of data at a discrete time). rasp will only process/generate BIL/BSQ files containing a single band - making both file formats identical for all practical purposes.

A minimum BIL dataset consists of two files (additional files may be present, but are not processed/generated by rasp - if an ESRI 'world file' is required, this can be generated by a separate 'export' command):

a) the main data file is a binary file containing a integer pixel grid in row-major order.

b) a header file describing the properties of the data file.

The header file is a plain text file containing lines with keyword/value pairs. The following keywords are recognized/generated by rasp (any unknown keywords encountered are ignored; if no header file can be found on import, all values will be set to the 'keyword absent' case, i.e. assuming a non-georeferenced BIL with accompanying BIA and BIT files):

- **BYTEORDER** Can be 'I' ('Intel', least significant byte first) or 'M' ('Motorola', most significant byte first). If the keyword is absent, or the value set to an unknown value, 'I' is assumed.

- **LAYOUT** The values 'BIL' and 'BSQ' are accepted by rasp. Any other value will cause rasp to terminate with an error. If the keyword is absent, 'BIL' is assumed.

- **NBANDS** If the keyword is present, the value has to be '1', or rasp will terminate with an error message.

- **NBITS** Permitted values for this field are '8', '16', or '32'. 8-bit values are treated as unsigned integers (e.g. intensities), whereas 16 and 32 bit values are considered to be signed. If an unknown value is encountered, rasp terminates with an error message. If the keyword is absent, 8-bit values are assumed.

- **NCOLS** A numerical value with this keyword is interpreted as the number of data columns of a rectangular pixel grid. As an extension over the commonly used BIL format, rasp uses any non-numerical value specified with this keyword as the file name of the pixel angle file (BIA) associated with non-georeferenced BIL data. For each column of the BIL file, the BIA file contains a single text line with the central sensing angle of the associated data pixel. If the keyword is absent, the string '{filename}.BIA' is assumed.

- **NROWS** A numerical value with this keyword is interpreted as the number of data rows of a rectangular pixel grid. As an extension over the commonly used BIL format, rasp uses any non-numerical value specified with this keyword as the file name of the line timing file (BIT) associated with non-georeferenced BIL data. For each row of the BIL file, the BIT file contains a single text line with the timing information for this line as seconds since the last UTC midnight. If the keyword is absent, the string '{filename}.BIT' is assumed.

- **ULXMAP** Easting/Longitude value associated with the first column of georeferenced...
data

**ULYMAP**  Northing/Latitude value associated with the first row of georeferenced data

**XDIM**  East-west size of a data pixel in metres/degrees (georeferenced data only)

**YDIM**  North-south size of a data pixel in metres/degrees (georeferenced data only)

**Note:** As BIL files only contain integer values, and (unlike e.g. LAS files) do not contain any scaling factors internally or in the header, appropriate pre/post-scaling has to be applied to resolve/preserve any fractional detail. Rather attempting to evaluate any projection file associated with a georeferenced BIL dataset, **rasp** will assume the data to be in UTM coordinates of the currently set zone (see 'Set.UTMzone') unless directed otherwise with the 'WGS84' option in its definition.

### 9.4.2 Definition

**Syntax:**  
{bil_file} : {bil/bsq} {filename} {field},{band_index},{data_type},{scale},{wgs84},{envi/esri}

**Parameters:**

- `{name}` (keyword,optional) – The name to use for the BIL file handle
- `{ID}` (keyword,optional) – The identifier ‘BIL’ can be used to override automatic file type deduction
- `{filename}` (filename) – name of the BIL data file
- `{field}` (keyword,optional) – point data field to read the BIL data to (Default ‘Altitude’)
- `{scale}` (numeric,optional) – scale factor for the BIL data (multiplication on import, division on export, default = 1.0)
- `{wgs84}` (keyword,optional) – The presence of 'WGS84' makes **rasp** use geographic coordinates in degrees instead of UTM metres for horizontal coordinates

### 9.4.3 Examples

**bildata:/data/tile1.bil wgs84**

Exports the grid 'grid1' in WGS84 coordinates to the file 'tile1.bil' and writes the georeferencing information to 'tile1.hdr'.

**red:/data/tslsred.bil red**

**inu:/data/inu.csv**

**out1:/data/out1.las**

**project out1 red inu 'grid1;/data/local/dem_conf.txt;*’**

**nir:/data/tslsnir.bil blue**

**out2:/data/out2.las**

**merge out2 out1 nir blue**

First georeferences/projects the data in ‘tslsred.bil’, using the angle information in ‘tslsred.bia’ and timing from ‘tslsred.bit’ in conjunction with navigation data from ‘inu.csv’. Elevation information for the projection is looked up in sequence in the local memory grid ‘grid1’,
a DEM file tree described in ‘/data/local/dem_conf.txt’, and finally (if no prior success) in the system-wide DEM tree.

9.4.4 Export

**Syntax:** Export {bil} {in} {options}

**Description:** Export a grid as a BIL file.

**Parameters:**

- `{bil}` (data) – output data in earth coordinates (UTM metres with elevation relative to WGS84 ellipsoid)
- `{in}` (data) – input
- `{inu}` (string) – The command to repeat
- `{Parameters}` (string) – Parameters for the command
9 Data Files

9.5 BLN files

9.5.1 Description

_rasp_ uses a very simple ASCII file format for storage of 2D/3D point and/or vector lists. The format used is very similar to the 'blanking file' format used by the Golden Software range of programs. Actually, the _rasp_.bln files represent a superset of the Golden Software .bln files, i.e. _rasp_ can read .bln files generated by Surfer, Grapher etc., but these programs can not necessarily interpret the various extensions to the format used by _rasp_.

In General, .bln files can be used anywhere in _rasp_ where point lists are required - if the first character of a point list is numeric, _rasp_ tries interpret it literally (i.e. as a sequence of point coordinates and associated properties), if not, it is interpreted as the name of a .bln file which is then searched for using the general file search rules. If a matching .bln file is found, a second text parameter is evaluated as the name of a sub-entity within the .bln file (this parameter defaults to '*', i.e. the first entity encountered).

```plaintext
npoints,flag,name x1,y1 (z1,string1) . . . . . . xn,yn (zn,stringn)
```

9.5.2 Definition

**Syntax:** `{name} : {ID} {filename} {section}

**Parameters:**

- `{name}` (keyword, optional) – The name to use for the BLN file handle
- `{ID}` (keyword, optional) – The identifier 'BLN' can be used to override automatic file type deduction
- `{filename}` (filename) – name of the BLN data file

9.5.3 Specific Commands

(none)
9.6 BT files

9.6.1 Description

*rasp* can export gridded data to BT (Binary Terrain) files. A limited (no external projection files) import capability is also available.

9.6.2 Definition
9.7 CSV files

9.7.1 Description

(ASCII text data from OXTS RTnK INU units; source only)

9.7.2 Definition

**Syntax:** \{name\} : \{ID\} \{filename\}

**Parameters:**

- \{name\} (keyword,optional) – The name to use for the CSV file handle
- \{ID\} (keyword,optional) – The identifier 'CSV' can be used to override automatic file type deduction
- \{filename\} (filename) – name of the CSV data file

9.7.3 Specific Commands

(none)
9.8 DAT files

9.8.1 Description

(Generic ASCII text data files

9.8.2 Definition

Syntax: \{name\} : \{ID\} \{filename\} \{format\} \{blocking specification\}

Parameters:

\{name\}  (keyword,optional) – The name to use for the DAT file handle
\{ID\}  (keyword,optional) – The identifier ‘DAT’ can be used to override automatic file type deduction
\{filename\}  (filename) – name of the DAT data file
\{blocking specification\}  (numeric,optional) – specification of the block size and start/end indices as decribed above

Available format identifiers:

- Amplitude of channel ‘blue’ in receiver units
- D  Range (distance) in metres
- G  Amplitude of channel ‘green’ in receiver units
- R  Amplitude of channel ‘red’ in receiver units
- X  Longitude in decimal degrees (WGS84)
- Y  Latitude in decimal degrees (WGS84)
- a  Amplitude of primary channel (i.e. normally echo intensity) in receiver units
- c  Class ID (as specified for the LAS file format)
- h  Heading angle in decimal degrees
- i  Index of target for a given laser shot
- k  Index of line within file
- n  Total number of targets for a given laser shot
- p  Pitch angle in decimal degrees
- r  Roll angle in decimal degrees (for georeferenced data, this field contains the scan angle)
- t  Time in microseconds since UTC midnight of the file start day
- w  Echo half-peak width in integer tenth of nanoseconds
- x  Easting in UTM metres (WGS84)
- y  Northing in UTM metres (WGS84)
- z  Altitude in metres (relative to WGS84 ellipsoid)
- T  TAB delimiter

All other characters represent themselves and can be used as delimiters between data fields, e.g. the format ‘Y,X,z’ would generate a comma-delimited list of latitude/longitude/altitude data.
9.8.3 Specific Commands

(none)
9.9 GRD files

9.9.1 Description

This data object handles Golden Software grid files (‘GRDA’ ASCII text and ‘GRD6’ binary, ‘GRD7’ and later formats are not supported).

9.9.2 Capabilities

<table>
<thead>
<tr>
<th>Operation</th>
<th>Cloud</th>
<th>Grid</th>
<th>Mesh</th>
<th>Bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Streaming Read</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td>x</td>
</tr>
<tr>
<td>Streaming Write</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Import</td>
<td>-</td>
<td>x</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Export</td>
<td>-</td>
<td>x</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

9.9.3 Export

**Syntax:** Export {Grid Object} {GRD File Object}

**Description:** Export gridded data to a Golden Software Surfer GRD file (either binary or ASCII).

**Parameters:**

- **{Grid}** (data) – The grid to export from
- **{Filename}** (filename,optional) – name of the output file (Default=’out.grd’)
- **{What}** (keyword,optional) – Field identifier designating which point component to export (Default=ALTITUDE)
- **{Format}** (keyword,optional) – ‘GRD6’ and ‘GRDA’ select binary and ASCII formats, respectively (Default=’GRD6’)

9.9.4 Import

**Syntax:** Import {Grid Object} {GRD File Object}

**Description:** Export gridded data to a Golden Software Surfer GRD file (either binary or ASCII).

**Parameters:**

- **{Grid}** (data) – The grid to export from
- **{Filename}** (filename,optional) – name of the output file (Default=’out.grd’)
- **{What}** (keyword,optional) – Field identifier designating which point component to export (Default=ALTITUDE)
- **{Format}** (keyword,optional) – ‘GRD6’ and ‘GRDA’ select binary and ASCII formats, respectively (Default=’GRD6’)

9.10 LAS files

9.10.1 Description

9.10.2 Definition

Syntax: \{name\} : \{ID\} \{filename\} \{blocking specification\}

Parameters:

\{name\} (keyword,optional) – The name to use for the LAS file handle

\{ID\} (keyword,optional) – The identifier ‘LAS’ can be used to override automatic file type deduction

\{filename\} (filename) – name of the LAS data file

\{blocking specification\} (numeric,optional) – specification of the block size and start/end indices as described above

Class IDs defined in the LAS specification:

In *rasp* only the four lower bits of the class id are used/evaluated. The LAS specification defines bits 5,6, and 7 as flags for ‘synthetic data’, keypoint’ and ‘deleted’, respectively. For data generated by *rasp*, these bits are always set to zero. For externally generated data, these bits are preserved on import/export (i.e. unless the data has been re-classed by *rasp*).

0 Created but never classified

1 Unclassified (i.e. attempted but not resolved, or existing classification revoked)

2 Ground

3 Low vegetation

4 Medium vegetation

5 High vegetation

6 Building

7 Low point (noise)

8 Model keypoint (mass point)

9 Water

12 Overlap

9.10.3 LAS System ID

Syntax: **Set.LASsystemID** \{string\}

Description: Instructs the LAS file exporter to write a specific system ID to the header of files it generates.

Parameters:

\{string\} (optional) The system ID string to use (max. 32 characters, Default=’(unknown)’)
9.10.4 LAS Point Format

**Syntax:**  
Set.LASpointFormat {0/1}

**Description:** Instructs the LAS file exporter to write point records of a specific type (refer to LAS format documentation for details).

**Parameters:**

{0/1}  (numeric,optional) – Selects point record type 0 or 1 (Default=1)

9.10.5 LAS Project ID

**Syntax:**  
Set.LASprojectID {ID1} {ID2} {ID3} {ID4}

**Description:** Instructs the LAS file exporter to write a specific project ID to the header of files it generates.

**Parameters:**

{ID1}  (numeric,optional) – The project ID1 number to use (positive 32 bit integer; Default=0)

{ID2}  (numeric,optional) – The project ID2 number to use (positive 16 bit integer; Default=0)

{ID3}  (numeric,optional) – The project ID3 number to use (positive 16 bit integer; Default=0)

{ID4}  (string,optional) – The project ID string to use (max. 8 characters; Default='(none)')

9.10.6 LAS File Source ID

**Syntax:**  
Set.LASfileSourceID {ID}

**Description:** Instructs the LAS file exporter to write a specific file source ID to the header of files it generates.

**Parameters:**

{ID}  (numeric,optional) – The file source ID number to use (positive 16 bit integer; Default=0)
9.11 PLY Files

9.11.1 Description

`rasp` can import vertex and face data from PLY files (Stanford Polygon File Format, ASCII as well as binary). The standard PLY vertex properties 
\( (x,y,z) \) are imported as \( (\text{easting},\text{northing},\text{altitude}) \). Integer values are considered to be centimetres, floating point values are treated as decimal metres. In addition to the positioning information, `rasp` can import/export most other point properties from/to PLY files, e.g. if a PLY file contains the fields \( (\text{red},\text{green},\text{blue}) \), these are automatically imported into the respective colour values for each data point. 'Unknown' point data fields (e.g. normal vectors and diffuse/specular reflectance values) commonly found in PLY files generated by 3D modelling software are silently ignored by `rasp`. PLY handling in `rasp` is limited to a maximum of 16 properties per data point (including the positioning information).

Faces defined in the PLY file are imported as 'hidden' data points (i.e. with the attributes nTargets=0 and iTarge=1) containing the corner points of a triangle as offsets into the main point cloud. Faces with more than 3 corner points are broken up into triangle fans on import.

Any other data possibly contained in the PLY file (e.g. edge and material data) is ignored.

9.11.2 Definition

**Syntax:** \{name\} : {ID} {filename}

**Parameters:**
- `{name}` (keyword,optional) – The name to use for the PLY file handle
- `{ID}` (keyword,optional) – The identifier 'PLY' can be used to override automatic file type deduction
- `{filename}` (filename) – name of the PLY data file

9.11.3 Specific Commands

(none)
9.12 SDC Files

9.12.1 Description

SDC files are intermediate data files containing non-georeferenced scan point data, primarily from 'Riegl' laser scanners. Originating software is e.g. 'RiAnalyze' (converting full-waveform data to discrete returns) and 'Q242SDC' (which converts data from the Q240 scanner series to SDC format).

In order to import SDC files, `rasp` needs access to independently logged position/attitude data (specified as secondary data in the processing statement). `rasp` can not export data to the SDC format.

9.12.2 Definition

**Syntax:** `{name} : {ID} {filename}`

**Parameters:**

- `{name}` (keyword,optional) – The name to use for the SDC file handle
- `{ID}` (keyword,optional) – The identifier ‘SDC’ can be used to override automatic file type deduction
- `{filename}` (filename) – name of the SDC data file

9.12.3 Specific Commands

(none)
9.13 ESRI World File

9.13.1 Description

`rasp` can generate ESRI style ‘world files’ describing the geolocation of gridded/raster data as six lines of ASCII text. The common naming convention for world files is to alter the extension part of the associated image file by replacing the second letter with the last letter, and replace the third letter with the letter ‘w’. (e.g., .tif -¿ .tfw, .bsq -¿ .bqw).

World files do not specify a particular coordinate system and/or projection. `rasp` can generate world files in two basic styles:

- Metric UTM coordinates - as the UTM zone is not contained in the world file, it has to be supplied to an importing program separately
- WGS84 geographic coordinates

9.13.2 Definition

**Syntax:** \{world_file\} : world \{filename\} \{utm/wgs\}

**Parameters:**

- \{world_file\} Identifier to assign to this data object
- \{filename\} Path/name of the file to associate with the identifier
- \{utm/wgs\} Keyword selecting the coordinate system to be used (Optional; can be 'UTM' or 'WGS'; default='UTM')

9.13.3 Export

**Syntax:** Export \{grid_object\} \{world_file\}

**Description:** Write a world file describing the geolocation of a grid object.

**Parameters:**

- \{grid_object\} Identifier referencing a gridded data object
- \{world_file\} Identifier referencing a world file
10 Appendices
10.1 Predefined Colours

<table>
<thead>
<tr>
<th>Colour Number</th>
<th>Colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>grey</td>
</tr>
<tr>
<td>1</td>
<td>navy</td>
</tr>
<tr>
<td>2</td>
<td>green</td>
</tr>
<tr>
<td>3</td>
<td>teal</td>
</tr>
<tr>
<td>4</td>
<td>red</td>
</tr>
<tr>
<td>5</td>
<td>purple</td>
</tr>
<tr>
<td>6</td>
<td>maroon</td>
</tr>
<tr>
<td>7</td>
<td>white</td>
</tr>
<tr>
<td>8</td>
<td>lightgrey</td>
</tr>
<tr>
<td>9</td>
<td>aqua</td>
</tr>
<tr>
<td>10</td>
<td>lime</td>
</tr>
<tr>
<td>11</td>
<td>blue</td>
</tr>
<tr>
<td>12</td>
<td>olive</td>
</tr>
<tr>
<td>13</td>
<td>fuchsia</td>
</tr>
<tr>
<td>14</td>
<td>yellow</td>
</tr>
<tr>
<td>15</td>
<td>black</td>
</tr>
</tbody>
</table>

Table 3: Default definitions of colour numbers 0 - 15
10.2 Standard Palettes

<table>
<thead>
<tr>
<th>Name</th>
<th>Preview</th>
</tr>
</thead>
<tbody>
<tr>
<td>bgold</td>
<td></td>
</tr>
<tr>
<td>blackbody</td>
<td></td>
</tr>
<tr>
<td>blue-green</td>
<td></td>
</tr>
<tr>
<td>blues</td>
<td></td>
</tr>
<tr>
<td>borders</td>
<td></td>
</tr>
<tr>
<td>browns</td>
<td></td>
</tr>
<tr>
<td>caramel</td>
<td></td>
</tr>
<tr>
<td>caribbean-blues</td>
<td></td>
</tr>
<tr>
<td>cascade</td>
<td></td>
</tr>
<tr>
<td>china</td>
<td></td>
</tr>
<tr>
<td>coldfire</td>
<td></td>
</tr>
<tr>
<td>dark_pastels</td>
<td></td>
</tr>
<tr>
<td>deviation1</td>
<td></td>
</tr>
<tr>
<td>deviation2</td>
<td></td>
</tr>
<tr>
<td>deviation3</td>
<td></td>
</tr>
<tr>
<td>deviation4</td>
<td></td>
</tr>
<tr>
<td>deviation5</td>
<td></td>
</tr>
<tr>
<td>deviation6</td>
<td></td>
</tr>
<tr>
<td>ega</td>
<td></td>
</tr>
<tr>
<td>firecode</td>
<td></td>
</tr>
<tr>
<td>gold</td>
<td></td>
</tr>
<tr>
<td>gradient1</td>
<td></td>
</tr>
<tr>
<td>gradient2</td>
<td></td>
</tr>
<tr>
<td>gradient3</td>
<td></td>
</tr>
<tr>
<td>gradient4</td>
<td></td>
</tr>
<tr>
<td>gradient5</td>
<td></td>
</tr>
<tr>
<td>gradient6</td>
<td></td>
</tr>
<tr>
<td>gradient7</td>
<td></td>
</tr>
<tr>
<td>gradient8</td>
<td></td>
</tr>
<tr>
<td>gradient9</td>
<td></td>
</tr>
<tr>
<td>grayblue</td>
<td></td>
</tr>
<tr>
<td>grays</td>
<td></td>
</tr>
<tr>
<td>grayviolet</td>
<td></td>
</tr>
<tr>
<td>greens</td>
<td></td>
</tr>
<tr>
<td>hilite</td>
<td></td>
</tr>
<tr>
<td>intensity1</td>
<td></td>
</tr>
<tr>
<td>intensity2</td>
<td></td>
</tr>
<tr>
<td>khaki</td>
<td></td>
</tr>
<tr>
<td>land-1</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Standard Palettes
<table>
<thead>
<tr>
<th>Name</th>
<th>Preview</th>
</tr>
</thead>
<tbody>
<tr>
<td>muted</td>
<td><img src="muted.png" alt="" /></td>
</tr>
<tr>
<td>ndvi</td>
<td><img src="ndvi.png" alt="" /></td>
</tr>
<tr>
<td>news3</td>
<td><img src="news3.png" alt="" /></td>
</tr>
<tr>
<td>plasma</td>
<td><img src="plasma.png" alt="" /></td>
</tr>
<tr>
<td>rainbow1</td>
<td><img src="rainbow1.png" alt="" /></td>
</tr>
<tr>
<td>ramp_blue1</td>
<td><img src="ramp_blue1.png" alt="" /></td>
</tr>
<tr>
<td>ramp_blue2</td>
<td><img src="ramp_blue2.png" alt="" /></td>
</tr>
<tr>
<td>ramp_green1</td>
<td><img src="ramp_green1.png" alt="" /></td>
</tr>
<tr>
<td>ramp_green2</td>
<td><img src="ramp_green2.png" alt="" /></td>
</tr>
<tr>
<td>ramp_green3</td>
<td><img src="ramp_green3.png" alt="" /></td>
</tr>
<tr>
<td>ramp_grey</td>
<td><img src="ramp_grey.png" alt="" /></td>
</tr>
<tr>
<td>ramp_red</td>
<td><img src="ramp_red.png" alt="" /></td>
</tr>
<tr>
<td>ramp_six</td>
<td><img src="ramp_six.png" alt="" /></td>
</tr>
<tr>
<td>ramp_three</td>
<td><img src="ramp_three.png" alt="" /></td>
</tr>
<tr>
<td>reds</td>
<td><img src="reds.png" alt="" /></td>
</tr>
<tr>
<td>royal</td>
<td><img src="royal.png" alt="" /></td>
</tr>
<tr>
<td>sky</td>
<td><img src="sky.png" alt="" /></td>
</tr>
<tr>
<td>spectrum</td>
<td><img src="spectrum.png" alt="" /></td>
</tr>
<tr>
<td>sunrise</td>
<td><img src="sunrise.png" alt="" /></td>
</tr>
<tr>
<td>temperature1</td>
<td><img src="temperature1.png" alt="" /></td>
</tr>
<tr>
<td>temperature2</td>
<td><img src="temperature2.png" alt="" /></td>
</tr>
<tr>
<td>topo_atlas</td>
<td><img src="topo_atlas.png" alt="" /></td>
</tr>
<tr>
<td>topo_ice</td>
<td><img src="topo_ice.png" alt="" /></td>
</tr>
<tr>
<td>topo</td>
<td><img src="topo.png" alt="" /></td>
</tr>
<tr>
<td>topo_water</td>
<td><img src="topo_water.png" alt="" /></td>
</tr>
<tr>
<td>volcano</td>
<td><img src="volcano.png" alt="" /></td>
</tr>
<tr>
<td>water</td>
<td><img src="water.png" alt="" /></td>
</tr>
</tbody>
</table>

Table 5: Standard Palettes (continued)