



User Manual

**wavemap**<sup>TM</sup>

< 2.1 >

**REF : RP/WAVECALL/09-2001N01/ZDP**

**Dapeng Zhang**  
**Robert Schweikert**

**Summary:**

This document describes the how to use the GIS tool: WaveMap for converting buildings and terrain data and checking the buildings data.

## Company Information

### Address

Lausanne Wavecall SA  
Science Park of the Swiss Institute of Technology  
PSE-B / EPFL  
1015 Lausanne  
  
Phone +41 21 693 84 05 and 41 21 693 85 10  
Fax +41 21 693 84 06

Amsterdam Wavecall BV  
NZ Kolk 29  
1012 Amsterdam / NL  
  
Phone +31 20 320 8302  
Fax +31 20 528 7363

Contact Dr. Karim Rizk, CEO  
Email [info@wavecall.com](mailto:info@wavecall.com)  
Web <http://www.wavecall.com>

## Document History

Version	Revision	Date
1.0	Dapeng Zhang	Sept 05, 2001
1.1	Robert Schweikert	Sept 11, 2001
2.0	Robert Schweikert	Sept 27, 2001
2.1	Robert Schweikert	October 10, 2001

## Acknowledgments

Thank Robert Schweikert for his very helpful work on this document.

## Contents

<b>1. Introduction</b>	<b>4</b>
1.1 Structure of the document	4
<b>2 Quick Reference</b>	<b>5</b>
2.1 The Outline of Usages	5
2.2 Building and Terrain Formats	5
2.3 Modes and Overall Options	6
2.3.1 Mode Option strings	6
2.3.2 Overall Options	7
2.3.3 Checking Keywords	7
<b>3 Building and Terrain formats</b>	<b>8</b>
3.1 The Input and Output of Buildings	8
3.1.1 Asset/Planet Buildings	8
3.1.2 Odyssey Buildings	8
3.1.3 MIF Format Buildings	9
3.1.4 SIM Format Buildings	9
3.1.5 Shape2D format buildings	10
3.1.6 DXF Building	11
3.2 The Input and Output of Terrains	12
3.2.1 Asset/Planet Terrain	12
3.2.2 TRN Terrain	12
3.2.3 Odyssey Terrain	12
3.2.4 XYV Terrain	12
3.2.5 BIL Terrain	13
<b>4 Modes and Options</b>	<b>14</b>
4.1 wavemap run modes	14
4.2 Overall Options	15
<b>5 Checking Data</b>	<b>17</b>

# 1. Introduction

## 1.1 Structure of the document

2<sup>nd</sup> chapter of this document describes the function of wavemap, including a quick reference of all formats supported by wavemap, modes' options, and data checking keyword.

3<sup>rd</sup> chapter of this document gives converting between formats a brief description.

4<sup>th</sup> chapter of this document explain all modes and their usages.

5<sup>th</sup> chapter of this document gives checking data a brief description.

## 2. Quick Reference

This chapter give a quick reference to all formats, options, and keywords supported by wavemap.

### 2.1 The Outline of Usages

wavemap can convert map data from one format to another.

The modes is to satisfy relatively complex converting requirements

Checking gives a solution to correct or only report errors which are found while converting.

### 2.2 Building and Terrain Formats

All building format keywords and option strings are listed below, see chapter 3 for detail.

Format	Options
SIM	"-simb file.sim"
Asset/Planet	"-assetb vecdir"
Odyssey	"-odyb vecdir feature"
MIF	"-mifb file.mif elevation"
2D Shape	"-shape2db file.shp elevation"
DXF	"-dxfb file.dxf"(only output)

Format	Argument
TRN	"-trnt file.trn"
Asset/Planet terrain	"-assett terdir"
Odyssey format	"-odyt terdir"
XYV	"-xyvt file.txt"
BIL	"-bilt file.bil"

## 2.3 Modes and Overall Options

All modes and their option strings are listed below, see chapter 4 for detail.

### 2.3.1 Mode Option strings

Run Modes' Keywords Specification			
Mode	Keyword	input	output
wavemap -h	-h	No	No
wavemap -v	-v	No	No
wavemap B1 B2	No	B1	B2
wavemap B1 B2	No	B1	B2
wavemap -sea2ground B1 B2 R	-sea2ground	B1 R	B2
wavemap -ground2sea B1 B2 R	-ground2sea	B1 R	B2
wavemap R -frame xmin ymin xmax ymax -res res_num -tedf tdef_num	-frame -res -tedf	no	R
wavemap -queryframe B	-queryframe	B	No
wavemap -assetb strs B -atrformat strs	-atrformat	-assetb strs	B
wavemap -assetb strs1 -atrformat strs assetb strs -atrformat strs2	-atrformat atrformat	-assetb strs1	-assetb strs2
wavemap B R1 R2 -rasterize	-rasterize	B R1	R2
wavemap R1 R2 -resample xmin ymin xmax ymax res	-resample	R1	R2
wavemap B1 B2 -check keyword	-check	B1	B2
<p>Notes:</p> <p><i>B means one of building format, R means one of raster/terrain format. See 2.2 for detail.</i></p> <p><i>It's possible to exist some overall options in these modes. See 2.3.2 for overall options.</i></p>			

### 2.3.2 Overall Options

Overall options can be used in all modes, see chapter 4 for detail

Overall Options Specification	
Options	Expression
-lonlat2grid	-lonlat2grid radius (float)
-latlon2grid	-latlon2grid radius (float)
-grid2lonlat	-grid2lonlat radius (float)
-grid2latlon	-grid2latlon radius (float)
-renumber	-renumber base_num (int)
-precision	-precision decimal_num (int )
-heightfactor	-heightfactor hf (float)
-coordfactor	-coordfactor cf (float)

### 2.3.3 Checking Keywords

All keywords following -check are listed below, see chapter 5 for detail information.

Check Keywords Specification	
Keywords	Expression
all/ [none]	-check all/-check
repeatedcoords	-check repeatedcoords
overlap	-check overlap
eight	-check eight
identical	-check identical
selfintersect	-check selfintersect
open	-check open
oneortwo	-check oneortwo
negheight	-check negheight
Notes: <i>-check mode is specified in 2.3.1</i>	

### 3. Building and Terrain formats

This chapter describe the input and output of wavemap.

#### 3.1 The Input and Output of Buildings

##### 3.1.1 Asset/Planet Buildings

<b>input</b>	
index.txt	each column: vector file(string) attribute file (string) ... \n maybe more than one column, refresh the frame by buildings vectors when output.
attribute file (specified in index.txt)	each column: ID (long) name (string) height(double) Name and ID are ingnored when read.
vector file (specified in index.txt)	head: ID(long) name (string) the number of walls (long); Lost name when read.  body: x(double) y(double) ... number is specified by head.
<b>output</b>	
index.txt	Only one column: vec_bldgs.txt (string) attr_bldgs.txt (string) frame xmin xmax ymin ymax (double double double double, calculated from all buildings) buildings (string)
attr_bldgs.txt	each column: ID(long In vector file1-5) name(always "buildings" 7-19) height (double)
vec_bldgs.txt	head: ID (long) buildings(1-5) space(6-15) buildings(16-47) space(48-50) wall number (51-55)  body: x(double) y(double) two decimal, lost +1 or -1 accuracy at the last digital.

##### 3.1.2 Odyssey Buildings

<b>input</b>	
list	Recognized row: RAYTRACE filename.vec (filename is string) frame (double*4) string (feature name) \n (read the same name .atr file)  MIF filename.mif frame string\n (using Height or Elevation as height attribute )
mif mid vec	same as mif mid and asset vector file.
atr file	ID(int) "string string"(maybe space) height (double)
<b>output</b>	
list	only one row RAYTRACE filename.vec frame buildings
vector file	OdyBldgs.vec same as Asset vector file (There is no mif format vector file)
Attribute file	OdyBldgs.atr Using "buildings" as name string, other same as Asset (There is no mid format attribute file.)

### 3.1.3 MIF Format Buildings

<b>input</b>	
filename.mif	Header: ...[delimiter(string, used to search ID and Height)] \n Column(string, search ID and Height begin) [ID(string get the ID position in mid file)] Height(string specified by user(following the keyword BldgHeightAttrib) get the Height position in mid file) Data(string, body begin)\n  Body: Region(string) num (int, the number of buildings in this region.) \n <each building is> num (int, the number of walls in this building)\n <each wall is> num num (double, wall data) \n
filename.mid	[n * delimiter(char)] [ID(integer) delimiter (char)] Height(double) (delimiter)
<b>output</b>	
filename.mif	default header:  VERSION 300 CHARSET "WindowsLatin1" DELIMITER ", " Coordsys NonEarth Units "m" BOUNDS (0.0,0.0) (45.0,5.0) COLUMNS 2  ID integer  Bldg_elevation float  DATA  body: <each building is>Region 1 \n num(wall number)\n <each wall is>num num (double double)\n
filename.mid	<each row>ID(int), height(double)\n

### 3.1.4 SIM Format Buildings

<b>input/output</b>	
<b>file name</b>	<b>file format</b>
data file (*.sim)	headline "Is2Ground" number (0 or 1)  each column: string "ID" tab num(int) tab string "Elev" tab num(long) string "height" tab num(double) string "floor" num, num, ...(double) \n

### 3.1.5 Shape2D format buildings

<b>Input</b>	
<b>file name</b>	<b>file format</b>
index file (*.shx)	Only check if this file is exist, if not, report an error.
vector file (*.shp)	<p>Open as binary,</p> <p>File Header: from the 32<sup>nd</sup> byte read a integer in little-endian as shape type. Only support type=3 PolyLine and type=5 Polygon.</p> <p>Record Header: From first byte read an integer in big-endian as Building ID.</p> <p>Record Content: Read the 36-49 in little-endian as number part. Read 40-43 in little-endian as points number. Read following data according to specification. Accept only one part. Report a warning if there are more than one part(Polygon).</p>
attribute file (*.dbf)	<p>Open as binary,</p> <p>Read the length of every record, get the position of height information and length of height record. Read the heights.</p>
<b>Output</b>	
index file (*.shx)	Write every byte according to specification.
vector file (*.shp)	<p>Default header:</p> <p>Write shape type as 5 (Polygon) others are written as defined in specification.</p>
attribute file (*.dbf)	<p>Default header:</p> <p>0 03</p> <p>1-3 01 08 1F</p> <p>4-7 Building Number (int)</p> <p>8 41</p> <p>9 00</p> <p>10 10</p> <p>11-31 00</p> <p>32-42 Field Name (string)</p> <p>43 'N'</p> <p>44-47 00 00 00 00</p> <p>48 0f</p> <p>49 Decimal count (int)</p> <p>50-63 00</p> <p>64 0D</p> <p>Write height information.</p>

### 3.1.6 DXF Building

<b>Output</b>	
Vector file (* .dxf)	???
<b>Notes:</b> <i>Only support write, no height output.</i>	

## 3.2 The Input and Output of Terrains

### 3.2.1 Asset/Planet Terrain

input/output	
index.txt	each row: filename (string *.bin) frame(double*4) res(double)
*.bin	2*sizeof(char)for a float. Big-endian row → column each column big->small

### 3.2.2 TRN Terrain

*.trn	header: xmin ymin xmax ymax (frame information double.)\n res (double)\n Body: double...\n the bumper is calculated from frame and res!
filename*.trn	If there are more than one terrain data, create filename* (* is number created automatically) filename1.trn filename2.trn filename is a string defined by user.

### 3.2.3 Odyssey Terrain

Input	
list	each row: HEIGHT_FILE (string) space filename(string, bin file name) space frame(4*double) resolution (double) direction (0, little→big; 1, big→little) endian (SPARC big-endian, Intel little-endian)
bin	2 byte for a float, big-endian row->column direction defined in list
Output	
list	each row: HEIGHT_FILE (string) space filename (ody_terr.bin, add number after ody_terr if there are mutiple terrains.) frame(double*4) resolution (double) 0 SPARC
bin	ody_terr*.bin (* could be none, 1, 2, 3... ) 2 byte for a float, big-endian row→column each column mall→big

### 3.2.4 XYV Terrain

Input/Output	
ASCII (*.xyv)	The first row "x, y, v \n" Read/Write text x, y, v.

### 3.2.5 BIL Terrain

<b>Input</b>	
Header file (*.hdr)	Read layout, only support bil. Read nrows, ncols, ulxmap, ulymp, xdim, ydim to calculate x, y frame and res. Read skipbytes to skip bytes in bil file.
bin (*.bil)	Row->column big->small two bytes a height.
<b>Output</b>	
Header file (*.hdr)	Default header: nbits 16 nbands 1 layout bil skipbytes 512 Others caluate the data.
bin (*.bil)	Row->column big->small two bytes a height.

## 4. Modes and Options

### 4.1 wavemap run modes

Mode	Function	Example
wavemap -h	Output help information	wavemap -h
wavemap -v	Output version information	wavemap -v
wavemap B1 B2	Read from B1, write to B2	wavemap -assetb c:\asset\ -odyb C:\ody\ height
mapocnv R1 R2	Read from R1, write to R2	wavemap -asset c:\asset\ -odyt c:\ody\
wavemap -sea2ground B1 B2 R	Read from B1, change B1 height to ground level according to R, write to B2. When the point of data is on the boundary of gid, it's value is the average value of all grids besides it.	wavemap -assetb c:\asset -odyb c:\ody height -asset c:\asset -sea2ground
wavemap -ground2sea B1 B2 R	Same with -sea2ground mode, change to sea level. When the point of data is on the boundary of gid, it's value is the average value of all grids besides it.	wavemap -assetb c:\asset -odyb c:\ody height -asset c:\asset -ground2sea
wavemap R -frame xmin ymin xmax ymax -res res_num -tedf tdef_num	Create terrain files (R) the terrain frame is specified by -frame, height by -tedf, and resolution by -res.	wavemap -asset c:\asset -frame 0 0 3 3 -res 0.5 -tedf 9.5
wavemap -queryframe B	print the frame of B and building number of B.	wavemap -assetb c:\asset -queryframe
wavemap -assetb str B -atrformat str	read asset building attribute files using format specified by -atrformat. /I means building ID /H means building height, other string is specified by %s.	wavemap -assetb c:\asset -odyb c:\ody height -atrformat /I%s/H
wavemap -assetb str1 -atrformat str assetb str -atrformat str2	read asset building attribute files using format specified by the first -atrformat. and save them according to the second -atrformat. /I means building ID /H means building height, other string is specified by %s, or write out the string when output data.	wavemap -assetb c:\asset1 -assetb c:\asset2 -atrformat %d -atrformat %s%d
wavemap B R1 R2 -rasterize	Add B's height to R1 and output to R2	wavemap -assetb c:\asset -asset c:\asset -odyt c:\ody
wavemap R1 R2 -resample xmin ymin	read r1, change the resolution and save to r2. Maybe change	wavemap -asset c:\asset -odyt c:\ody -resample 0 0 3 3 0.5

xmax ymax res	only in the frame specified by -resample. When the point of data is on the boundary of gid, it's value is the average value of all grids besides it.	c:\ody -resample 0 0 3 3 0.5
wavemap B1 B2 -check keyword	check building data according to keyword, and write to B2.	wavemap -assetb c:\asset -odyb c:\ody height -check all

## 4.2 Overall Options

Option	Function	Example
-lonlat2grid	source building coordinate is longitude latitude (degree) order, change them to x, y plane: lat=lat/180*PAI;//convert to radian lon=lon/180*PAI;//convert to radian grid_y=Radius*cos(lat)*lon; grid_x=Radius*lon;	wavemap -assetb c:\asset -odyb c:\ody height -lonlat2grid 64000
-latlon2grid	source building coordinate is latitude longitude (degree) order, change them to x, y plane coordinate: lon=lon/180*PAI;//convert to radian lat=lat/180*PAI;//convert to radian grid_x=ConvRadius*cos(lat)*lon; grid_y=ConvRadius*lon;	wavemap -assetb c:\asset -odyb c:\ody height -latlon2grid 64000
-grid2lonlat	source building coordinate is x, y plane coordinate, change them to longitude latitude coordinate: lon=y/Radius/x; lat=acos(lon); lon=x/ConvRadius; lon=lon/PAI*180; lat=lat/PAI*180;	wavemap -assetb c:\asset -odyb c:\ody height -grid2lonlat 64000
-grid2latlon	source building coordinate is x, y plane coordinate, change them to latitude longitude coordinate: lat=x/Radius/y; lon=acos(lat); lat=y/Radius; lon=lon/PAI*180; lat=lat/PAI*180;	wavemap -assetb c:\asset -odyb c:\ody height -grid2latlon 64000

-renumber	Create the building ID according to the base number following this option, for example: if base num=1 then 1, 2, 3 ...	wavemap -assetb c:\asset - odyb c:\ody height -renumber 1
-precision	Define the the system precison and output vertices decimal. The precision value will decide whether or not two coordinates are identical, please pay extra attention to -precision when use -check.	wavemap -assetb c:\asset - odyb c:\ody height -precision 3
-heightfactor	Use the height factor to multiply height and output the new height to target.	wavemap -assetb c:\asset - odyb c:\ody height - heightfactor 10
-coordfactor	Use the coordiantes factor to multiply the coordinates and output he new coordinates to target	wavemap -assetb c:\asset - odyb c:\ody height - coordfactor 0.1

## 5. Checking Data

Keywords	Fuction	Example
all/ [none]	The order of check is: 1 negheight 2 open 3 repeatedcoords 4 line 5 oneortwo 6 eight 7 identical 8 selfintersect 9 overlap	mapconv -assetb c:\asset - odyb c:\ody height -check
repeatedcoords	When two continuous coordinates are identical, they are reported and the last one is deleted.	mapconv -assetb c:\asset - odyb c:\ody height -check repeatedcoords
overlap	Scan every wall in buildings to check if it intersect with other buidlings' wall. Report the coordinates when intersecting is found.	mapconv -assetb c:\asset - odyb c:\ody height -check overlap
eight	Scan all the walls in one building, is there are two separate walls are identical, the building is regard as eight and all wall other than walls between them are removed. Removed walls are reported.	mapconv -assetb c:\asset - odyb c:\ody height -check eight
identical	Compare among buildings, if both the walls and the order of walls are same, two buildings are regard as identical, the last one is removed from building list, all walls in it are reported.	mapconv -assetb c:\asset - odyb c:\ody height -check identical
selfintersect	When there are a wall intersect with another wall in a building, the building is regard as self-intersect, the intersecting walls are reported.	mapconv -assetb c:\asset - odyb c:\ody height -check selfintersect
open	When the begin vertex and the end vertex are not identical, the building is regard as open, mapconv will add the first vertex to end. The added vertex is reported.	mapconv -assetb c:\asset - odyb c:\ody height -check open
oneortwo	When there are less than 4 vertices in a building, the building is regarded as one-or-two, mapconv will remove the building and report all its coordinates.	mapconv -assetb c:\asset - odyb c:\ody height -check oneortwo
negheight	When the height of a building is below zero, the building is removed from building list and all deleted coordinates are reported.	mapconv -assetb c:\asset - odyb c:\ody height -check negheight

bldginsidebldg	When all coordinates of a building is inside or on the boundary of another, the building is regard as inside, all its coordinates are reported.	mapconv -assetb c:\asset -odyb c:\ody height -check bldginsidebldg
<p>Notes:</p> <p><i>Report the error coordinates and building lds to check.txt.</i></p>		