Height-field file

Introduction

The default file format for L3DT heightfields is HFF v1.0. This file format supports single precision floating-point data, 2-byte unsigned integer data and unsigned byte data. In the case of integer data, the vertical scaling and offset are used to stretch the byte range across the minimum and maximum altitudes in the heightfield, giving maximum fidelity.

Example code

To view some example C++ code for loading a HFF file, follow this link. This code is a snippet from the L3DT source, and does not include all the class or function definitions/implementations to which it refers. To use this code, you will need make appropriate modifications as described in the code comments.

Header structure

The height data in the HFF v1.0 file is preceded by a header containing the following information:

Field name	Contents	Length	Offset	Purpose
ASCII file marker	“L3DT”	4	0	Tells L3DT and humans that this is an L3DT file. Note string is not null terminated.
Binary map-type marker	USHORT	2	4	Tells L3DT what type of map file this is. This must be 300 for HFF v1.0
ASCII map-type marker	“HFF_v1.0”	8	6	Lets humans know what type of file this is. Note string is not null terminated.
Data offset	USHORT	2	14	Indicates the address of the first map pixel in this file, relative to the start of the file. See comments below.
Map width	UINT	4	16	East-west extent of the map (number of pixels).
Map height	UINT	4	20	North-south extent of the map (number of pixels).
Data size	BYTE	1	24	Bytes per heightfield pixel, either 1, 2 or 4.
Floating point flag	BYTE	1	25	1 if floating-point, 0 if integers.
Vertical scale	float	4	26	Vertical scale (in metres) of the heightfield.
Vertical offset	float	4	30	Vertical offset (in metres) of the heightfield.
Horizontal scale	float	4	34	Horizontal scale (in metres) of the heightfield.
Tile size	USHORT	2	38	See comments below.
Wrap flag	BYTE	1	40	See comments below.
Reserved	0	see below	41	See comments below.

Header length & reserved section

Typically the *data offset* is set to 64, meaning the reserved section is 64 - 41 = 23 bytes long. The reserved section contains no information, but may be used in later versions to store additional parameters, flags, etc.

Pixel order

a) Conventional order

If the *tile size* value is less than or equal to 1, the map data that follows the header is stored conventionally as a set of rows in order from south-to-north, and the data within the rows is ordered west-to-east. For example, a 256 × 512 map will consist of 512 rows of length 256. The first 256 values will be the southern-most row of heightfield values, starting at the left going to the right. The next 256 values will be the next row to the north, and so on.

b) Tiled order

If the *tile size* is greater than 1, the map within the file is stored as a series of square 'tiles' with a side-length of *tile size*. The data within the tiles is stored in the same fashion as above, and the ordering of the tiles is also west-to-east rows in order of south-to-north. L3DT does not, by default, use tiling within the HFF.

Data types

HFF data may be either:

- Single-precision floating point values (32-bit).
- Unsigned short integers (16-bit).
- Unsigned characters (8-bit).

Reading a HFF

To calculate the heights from a HFF, use the following formula:

height = VertScale × val + VertOffset

(where *val* is the height value in the file).

Note this formula holds true whether the datum are integers or floating point values.

The maximum and minimum altitudes will be:

minalt = VertOffset

maxalt = VertOffset + VertScale × FileRange

Where *FileRange* is:

- 256 for 8-bit mode.
- 65536 for 16-bit mode.
- Undefined for 32-bit float mode (to get the range, you will need to parse the file).

Writing a HFF

When writing a HFF, the values for each pixel that are written to the file should be:

val = (height - VertOffset) / VertScale

Where *val* is the value written to the file, *height* is the altitude value in the heightfield, and

VertOffset and *VertScale* are calculated from the formulas provided below.

Floating-point data (32-bit)

Easy, set *VertOffset* to 0 and *VertScale* to 1.

Unsigned short integer data (16-bit)

Set *VertOffset* to *minalt* and set *VertScale* to:

(maxalt - minalt) / 65535

(where *minalt* and *maxalt* are the lowest and highest altitudes in the map, respectively).

Unsigned character data (8-bit)

Set *VertOffset* to *minalt* and set *VertScale* to:

(maxalt - minalt) / 255

Wrap flag

The *wrap flag* indicates that the north-south and east-west edges of the map join in a continuous fashion. This option is obviously only useful for games and similar non-serious applications.

Backwards compatibility

Note that versions of L3DT older than v2.1 (Nov 2003) did not use the HFF v1.0 format. Those ancient HFF formats have since been relegated to the dust-bin of history, and can no-longer be loaded by L3DT.