Package 'terrainmeshr'

October 14, 2022

Type Package

Title Triangulate and Simplify 3D Terrain Meshes

Version 0.1.0

Description
Provides triangulations of regular height fields, based on the methods described in ``Fast Polygonal Approximation of Terrains and Height Fields" Michael Garland and Paul S. Heckbert (1995) <https://www.mgarland.org/files/papers/scape.pdf> using code from the 'hmm' library written by Michael Fogleman <https://www.github.com/fogleman/hmm>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.0.2

Imports Rcpp (>= 1.0.0)

LinkingTo Rcpp

URL https://www.github.com/tylermorganwall/terrainmeshr

BugReports https://www.github.com/tylermorganwall/terrainmeshr/issues

SystemRequirements C++11

NeedsCompilation yes

Author Tyler Morgan-Wall [aut, cph, cre]
(<https://orcid.org/0000-0002-3131-3814>),
Michael Fogleman [ctb, cph]

Maintainer Tyler Morgan-Wall <tylermw@gmail.com>

Repository CRAN

Date/Publication 2020-04-29 15:20:02 UTC

R topics documented:

`triangulate_matrix` *Triangulate a Height Map*

Description

Uses Delaney triangulation to approximate a rectangular height field (in matrix form) with constraints (either maximum allowable error, or a maximum number of triangles). Increasing the error limit will result in a courser approximation, but fewer triangles in the model. For many models (particularly those with large, flat regions or smooth changes in height), this can result in significant reductions in model size with no perceptual loss in terrain surface quality.

Usage

```
triangulate_matrix(
  heightmap,
  maxError = 1e-04,
  maxTriangles = 0,
  y_up = TRUE,
  start_index = 1,
  verbose = FALSE
)
```

Arguments

`heightmap`	A two-dimensional matrix, where each entry in the matrix is the elevation at that point. All points are assumed to be evenly spaced.
`maxError`	Default ‘0.0001‘. Maximum error allowed in triangulating the height map.
`maxTriangles`	Default ‘0‘, which turns off this setting (and only uses the ‘max_error‘ arg). Otherwise, specifies the maximum number of triangles when triangulating the height map.
`y_up`	Default ‘TRUE‘. Which axis is "upwards" in the return matrix. If ‘FALSE‘, ‘z‘ is up.
`start_index`	Default ‘1‘. The offset to the first ‘x‘ and ‘z‘ indices.
`verbose`	Default ‘FALSE‘. Prints reduction in number of triangles/max error.

Value

Returns a matrix of vertices and IDs for each triangle.

Examples

```
#Let's triangulate the built-in `volcano` dataset.

#Helper function to plot polygons over an `image()` plot.
plot_polys = function(tri_matrix) {
  #reverse orienation for `image`
```

```
  tri_matrix[,3] = max(tri_matrix[,3])-tri_matrix[,3]+1
  for(i in seq_len(nrow(tri_matrix)/3)) {
    polypath(tri_matrix[(3*(i-1)+1):(3*i), c(1,3)])
  }
}

#Here, we don't accept any error, but still triangulate
tris = triangulate_matrix(volcano, maxError = 0, verbose = TRUE)
image(x=1:nrow(volcano), y = 1:ncol(volcano), volcano)
plot_polys(tris)

#Let's increase the allowable error:
tris = triangulate_matrix(volcano, maxError = 1, verbose = TRUE)
image(x=1:nrow(volcano), y = 1:ncol(volcano), volcano)
plot_polys(tris)

#Increase it again
tris = triangulate_matrix(volcano, maxError = 10, verbose = TRUE)
image(x=1:nrow(volcano), y = 1:ncol(volcano), volcano)
plot_polys(tris)

#Here, we set an allowable number of triangles instead, using exactly 20 triangles:
tris = triangulate_matrix(volcano, maxTriangles = 20, verbose = TRUE)
image(x=1:nrow(volcano), y = 1:ncol(volcano), volcano)
plot_polys(tris)

#The output of this function can be passed directly to `rgl::triangles3d()` for plotting in 3D.
```

Index