

Package: sphereTessellation (via r-universe)

October 24, 2024

Title Delaunay and Voronoï Tessellations on the Sphere

Version 1.2.0

Description Performs Delaunay and Voronoï tessellations on spheres and provides some functions to plot them. The algorithms are mainly performed by the 'C++' library 'CGAL' (<<https://www.cgal.org/>>).

License GPL-3

URL <https://github.com/stla/sphereTessellation>

BugReports <https://github.com/stla/sphereTessellation/issues>

Imports colorsGen, grDevices, Polychrome, Rcpp, rgl

LinkingTo BH, Rcpp, RcppCGAL, RcppEigen

Suggests cooltools, uniformly

SystemRequirements C++ 17, gmp, mpfr

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation yes

Author Stéphane Laurent [aut, cre]

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Repository CRAN

Date/Publication 2024-01-08 13:50:02 UTC

Contents

DelaunayOnSphere	2
icosphereMesh	3
plotDelaunayOnSphere	4
plotVoronoiOnSphere	5
VoronoiOnSphere	6
Index	8

DelaunayOnSphere *Spherical Delaunay triangulation*

Description

Computes a spherical Delaunay triangulation.

Usage

```
DelaunayOnSphere(vertices, radius = 1, center = c(0, 0, 0), iterations = 5L)
```

Arguments

vertices	vertices, a numeric matrix with three columns
radius	radius of the sphere, a positive number; the vertices will be projected on this sphere
center	center of the sphere, a numeric vector of length three; the vertices will be projected on this sphere
iterations	positive integer, the number of iterations used to construct the meshes of the spherical faces

Details

See [2D Triangulations on the Sphere](#).

Value

A named list with four fields:

- vertices, the matrix of vertices obtained by projecting the original vertices to the sphere;
- faces, an integer matrix providing by row the indices of the faces of the triangulation;
- solidFaces, an integer vector providing the indices of the solid faces; faces are either solid faces or ghost faces, see details
- meshes, a list of meshes of the solid faces used for plotting in [plotDelaunayOnSphere](#).

See Also

[plotDelaunayOnSphere](#)

Examples

```
library(sphereTessellation)
library(rgl)

if(require(cooltools)) {
  vertices <- fibonaccisphere(30L)
  del <- DelaunayOnSphere(vertices)
```

```
open3d(windowRect = 50 + c(0, 0, 512, 512), zoom = 0.8)
plotDelaunayOnSphere(del)
}

if(require(uniformly)) {
# sample vertices on a hemisphere, so there will be some ghost faces
set.seed(421L)
vertices <- rphong_on_hemisphere(6L)
del <- DelaunayOnSphere(vertices)
# the ghost faces are not plotted
open3d(windowRect = 50 + c(0, 0, 512, 512), zoom = 0.8)
plotDelaunayOnSphere(del)
}
```

icosphereMesh

Icosphere

Description

Returns the mesh of an icosphere.

Usage

```
icosphereMesh(x = 0, y = 0, z = 0, r = 1, iterations = 3L)
```

Arguments

x, y, z	coordinates of the center
r	radius
iterations	number of iterations (the icosphere is obtained by iteratively subdividing the faces of an icosahedron)

Value

A **rgl** mesh (class mesh3d).

Examples

```
library(sphereTessellation)
library(rgl)
mesh <- icosphereMesh()
open3d(windowRect = 50 + c(0, 0, 512, 512))
shade3d(mesh, color = "navy")
```

plotDelaunayOnSphere *Plot spherical Delaunay triangulation*

Description

Plot a spherical Delaunay triangulation.

Usage

```
plotDelaunayOnSphere(
  del,
  colors = "random",
  distinctArgs = list(seedcolors = c("#ff0000", "#00ff00", "#0000ff")),
  randomArgs = list(hue = "random", luminosity = "bright"),
  edges = FALSE,
  vertices = FALSE,
  ecol = "black",
  lwd = 3,
  vcolor = "black",
  vradius = NA,
  ...
)
```

Arguments

del	an output of DelaunayOnSphere
colors	controls the filling colors of the triangles, either NA for no color, or a single color, or "random" to get multiple colors with randomColor , or "distinct" to get multiple colors with createPalette
distinctArgs	if colors = "distinct", a list of arguments passed to createPalette
randomArgs	if colors = "random", a list of arguments passed to randomColor
edges	Boolean, whether to plot the edges
vertices	Boolean, whether to plot the vertices
ecolor	a color for the edges
lwd	line width for the edges, if they are plotted
vcolor	a color for the vertices
vradius	a radius for the vertices, which are plotted as spheres (if they are plotted); NA for a default value
...	arguments passed to shade3d to plot the spherical triangles

Value

No value is returned.

Examples

```
library(sphereTessellation)
library(rgl)

vertices <- t(cuboctahedron3d())$vb[-4L, ]
del <- DelaunayOnSphere(vertices, radius = sqrt(2))

open3d(windowRect = 50 + c(0, 0, 512, 512), zoom = 0.8)
plotDelaunayOnSphere(del)
```

plotVoronoiOnSphere *Plot spherical Voronoï tessellation*

Description

Plot a spherical Voronoï tessellation.

Usage

```
plotVoronoiOnSphere(
  vor,
  colors = "gradient",
  distinctArgs = list(seedcolors = c("#ff0000", "#00ff00", "#0000ff")),
  randomArgs = list(hue = "random", luminosity = "bright"),
  palette = "Rocket",
  bias = 1,
  edges = FALSE,
  sites = FALSE,
  ecolor = "black",
  lwd = 3,
  scolor = "black",
  sradius = NA,
  ...
)
```

Arguments

vor	an output of VoronoiOnSphere
colors	controls the filling colors of the triangles, either NA for no color, or a single color, or "random" to get multiple colors with randomColor , or "distinct" to get multiple colors with createPalette , or "gradient"
distinctArgs	if colors = "distinct", a list of arguments passed to createPalette
randomArgs	if colors = "random", a list of arguments passed to randomColor
palette	this argument is used only when colors="gradient"; it can be either a character vector of colors, or the name of a palette which will be passed to the palette argument of the function hcl.colors

<code>bias</code>	this argument is used only when <code>colors="gradient"</code> ; it is passed to the <code>bias</code> argument of the function <code>colorRamp</code>
<code>edges</code>	Boolean, whether to plot the edges
<code>sites</code>	Boolean, whether to plot the Voronoi sites
<code>ecolor</code>	a color for the edges
<code>lwd</code>	graphical parameter for the edges, if they are plotted
<code>scolor</code>	a color for the sites
<code>sradius</code>	a radius for the sites, which are plotted as spheres (if they are plotted); NA for a default value
<code>...</code>	arguments passed to <code>shade3d</code> to plot the spherical faces

Value

No value is returned.

Examples

```
library(sphereTessellation)
library(rgl)
# take the vertices of the cuboctahedron and Voronoiize
vertices <- t(cuboctahedron3d())$vb[-4L, ]
vor <- VoronoiOnSphere(vertices)
# plot
open3d(windowRect = 50 + c(0, 0, 512, 512), zoom = 0.8)
plotVoronoiOnSphere(vor, specular = "black", edges = TRUE)

# effect of the `bias` argument ###
library(sphereTessellation)
library(rgl)
vertices <- t(cuboctahedron3d())$vb[-4L, ]
vor <- VoronoiOnSphere(vertices)
open3d(windowRect = 50 + c(0, 0, 900, 300), zoom = 0.8)
mfrow3d(1, 3)
plotVoronoiOnSphere(vor, palette = "Viridis", bias = 0.5)
next3d()
plotVoronoiOnSphere(vor, palette = "Viridis", bias = 0.8)
next3d()
plotVoronoiOnSphere(vor, palette = "Viridis", bias = 1.1)
```

VoronoiOnSphere

Spherical Voronoi tessellation

Description

Computes a spherical Voronoi tessellation.

Usage

```
VoronoiOnSphere(vertices, radius = 1, center = c(0, 0, 0), iterations = 5L)
```

Arguments

vertices	vertices, a numeric matrix with three columns
radius	radius of the sphere, a positive number; the vertices will be projected on this sphere
center	center of the sphere, a numeric vector of length three; the vertices will be projected on this sphere
iterations	positive integer, the number of iterations used to construct the meshes of the spherical faces

Details

First the Delaunay triangulation is computed, then the Voronoï tessellation is obtained by duality.

Value

An unnamed list whose each element corresponds to a Voronoï face and is a named list with three fields:

- site, the coordinates of the Voronoï site of the face;
- cell, a numeric matrix providing the coordinates of the vertices of the face;
- mesh, a mesh of the face used for plotting in the function [plotVoronoiOnSphere](#).

See Also

[plotVoronoiOnSphere](#)

Examples

```
library(sphereTessellation)
library(rgl)
if(require(cooltools)) {
  vertices <- fibonaccisphere(150L)
  vor <- VoronoiOnSphere(vertices)
  open3d(windowRect = 50 + c(0, 0, 512, 512), zoom = 0.8)
  plotVoronoiOnSphere(vor, colors = "random")
}
```

Index

`colorRamp`, [6](#)

`createPalette`, [4](#), [5](#)

`DelaunayOnSphere`, [2](#), [4](#)

`hcl.colors`, [5](#)

`icosphereMesh`, [3](#)

`plotDelaunayOnSphere`, [2](#), [4](#)

`plotVoronoiOnSphere`, [5](#), [7](#)

`randomColor`, [4](#), [5](#)

`shade3d`, [4](#), [6](#)

`VoronoiOnSphere`, [5](#), [6](#)