Package 'concom'

July 22, 2025

Type Package

Title Connected Components of an Undirected Graph

Version 1.0.0

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Description Provides a function for fast computation of the connected components of an undirected graph (though not faster than the components() function of the 'igraph' package) from the edges or the adjacency matrix of the graph. Based on this one, a function to compute the connected components of a triangle 'rgl' mesh is also provided.

License GPL-3

URL https://github.com/stla/concom

BugReports https://github.com/stla/concom/issues

Imports english, Rcpp (>= 1.0.8), rgl, Rvcg

Suggests rmarchingcubes

LinkingTo BH, Rcpp

Encoding UTF-8

RoxygenNote 7.2.0

NeedsCompilation yes

Author Stéphane Laurent [cre, aut]

Repository CRAN

Date/Publication 2022-06-15 07:30:05 UTC

Contents

concom-package *Connected Components*

Description

Provides a function which computes the connected components of an undirected graph.

Details

There are three functions in this package: `concom`, the main function, which returns the connected components from the edges of the graph; `concomFromMatAdj`, which returns the connected components from the adjacency matrix of the graph; `concom3d`, which returns the connected components of a triangle **rgl** mesh.

Author(s)

Stéphane Laurent.

Maintainer: Stéphane Laurent <laurent_step@outlook.fr>

concom *Connected components*

Description

Fast computation of the connected components of an undirected graph.

Usage

```
concom(edges)
```

Arguments

`edges`	a matrix with two columns, whose rows represent the edges of the graph; each edge is given by two vertex indices, and it is assumed that the vertex indices are 1, 2, 3, ...

Value

A list with four elements: `indices`, an integer vector whose `i`-th element gives the label of the connected component of vertex `i`; `sizes`, an integer vector giving the number of elements of each connected component; `ncomponents`, the number of connected components; `components`, a list of length `ncomponents`, whose `j`-th element is the integer vector made of the labels of the `j`-th connected component.

Examples

```
library(concom)
edges <- cbind(
  1:7,
  c(2, 3, 1, 5, 6, 7, 4)
)
concom(edges)
```

---

concom3d *Connected components of a 'rgl' mesh*

Description

Computes the connected components of a triangular 'rgl' mesh.

Usage

```
concom3d(tmesh)
```

Arguments

tmesh	a triangular **rgl** mesh (of class **mesh3d**)

Value

A list of **rgl** meshes, each one corresponding to a connected component.

Examples

```
library(concom)
library(rgl)
library(rmarchingcubes)

# credit to 'ICN5D' for this isosurface
f <- function(x, y, z, a, cosb, sinb){
    (sqrt((sqrt(x*x + (y*sinb + a*cosb)^2) - 2)^2) - 1)^2 +
      (sqrt((sqrt(z*z + (y*cosb - a*sinb)^2) - 2)^2) - 1)^2
}
a <- 0.6
b <- 0.785
cosb <- cos(b)
sinb <- sin(b)


x <- z <- seq(-3.5, 3.5, len = 150L)
y <- seq(-4.2, 4.2, len = 150L)
g <- expand.grid(X = x, Y = y, Z = z)
voxel <- array(
  with(g, f(X, Y, Z, a, cosb, sinb)),
```

```
  dim = c(150L, 150L, 150L)
)

contour_shape <- contour3d(
  griddata = voxel,
  level = 0.1,
  x = x,
  y = y,
  z = z
)

tmesh <- tmesh3d(
  vertices = t(contour_shape[["vertices"]]),
  indices = t(contour_shape[["triangles"]]),
  normals = contour_shape[["normals"]],
  homogeneous = FALSE
)

components <- concom3d(tmesh)
colors <- hcl.colors(length(components))
open3d(windowRect = c(50, 50, 562, 562), zoom = 0.9)
lapply(1:length(components), function(i){
  shade3d(components[[i]], color = colors[i])
})
```

`concomFromMatAdj` *Connected components from adjacency matrix*

Description

Connected components of an undirected graph from its adjacency matrix.

Usage

```
concomFromMatAdj(M)
```

Arguments

`M`	adjacency matrix; it must be a square symmetric matrix with numeric or Boolean entries, whose non-zero or `TRUE` entries indicate the connections (connection between `i`-th vertex and `j`-th vertex if the entry is located at row `i` and column `j`)

Value

The output is the same as the one of the `concom` function.

Examples

```
matAdj <- rbind(
  c(0, 1, 0, 0),
  c(1, 0, 0, 0),
  c(0, 0, 0, 0),
  c(0, 0, 0, 1)
)
concomFromMatAdj(matAdj)
```

Index