Package ‘tcie’

April 21, 2019

Type Package

Title Topologically Correct Isosurface Extraction

Version 0.3

Date 2019-04-19

Author Lis Custódio

Maintainer Lis Custodio <liscustodio.uerj@gmail.com>

URL http://liscustodio.github.io/

Description Isosurfaces extraction algorithms are a powerful tool in the interpretation of volumetric data. The isosurfaces play an important role in several scientific fields, such as biology, medicine, chemistry and computational fluid dynamics. And, for the data to be correctly interpreted, it is crucial that the isosurface be correctly represented. The Marching Cubes algorithm, proposed by Lorensen and Cline <doi:10.1145/37401.37422> in 1987, is clearly one of the most popular isosurface extraction algorithms, and an important tool for many visualization specialists and researchers. The generalized adoption of the Marching Cubes has resulted in many improvements in its algorithm, including, the establishment of the topological correctness of the generated mesh. In 2013, Custodio et al. <doi:10.1016/j.cag.2013.04.004> noted and corrected algorithmic inaccuracies that compromised the topological correctness of the mesh generated by the last version of the Marching Cubes Algorithm: the Marching Cubes 33 proposed by Chernyaev in 1995, implemented in 2003 by Lewiner et al. <doi:10.1080/10867651.2003.10487582>. In 2019, Custodio et al. (in the work An Extended Triangulation to the Marching Cubes 33 Algorithm) proposed an extended triangulation to the Marching Cubes 33 algorithm, in the proposed algorithm the grid vertex are labeled with ``+'', ``-'' and ``='', according to the relationship between its scalar field value and the isovalue.The inclusion of the ``='' grid vertex label naturally avoids degenerate triangles, a well-known issue in meshes generated by the Marching Cubes. The Marching Cubes algorithm has been implemented using many software programs and compilers: C++, proposed by Lewiner et al. (2003); 'MATLAB', proposed by Hammer (2011); and R, proposed by Feng and Tierney (2008). Marching Cubes is also integrated into many visualization toolkits. The complexity of an algorithm increases considerably when it aims to reproduce the topology of the trilinear interpolant correctly. This complexity can sometimes result in errors in the algorithm or in its implementation. During our experiments we observe that all the implementations mentioned have critical issues that compromise the continuity and the topological correctness of the generated mesh. The 'tcie' package is a toolkit with a topologically correct implementation of the Marching Cubes algorithm, based on the Custodio et al. work, which implements the most recent improvements of the algorithm.

LazyData TRUE

LinkingTo Rcpp,RcppArmadillo

Imports nat(>= 1.8.11), rgl(>= 0.99.9), Rvcg (>= 0.17), geomorph (>= 3.0.5)

Depends R (>= 3.2.3)

RoxygenNote 6.1.1

License GPL

Encoding UTF-8

NeedsCompilation yes

X-CRAN-Archive Archived on 2019-04-21 as update had exactly the same check problems as before.

Repository CRAN

Date/Publication 2019-04-21 15:10:03 UTC

R topics documented:

`BettiNumbers` *Returns the Betti numbers of a given mesh.*

Description

This function returns the Betti numbers b0, b1 and b2, which represents the number of connected components, the number of independent tunnelsand the number of closed regions in space, respectively. The function implementation followsthe algorithms described by Konkle, Moran, Hamann, and Joy in the work Fast Methods for Computing Isosur-face Topology with Betti Numbers.

Usage

```
BettiNumbers(file_path)
```

Arguments

file_path	A string: the path to a triangular mesh file in the ply format

Examples

```
BettiNumbers(system.file("extdata", "m3.ply", package = "tcie"))
```

Clipping — *Builds and plots the mesh representing the desired isosurface of an specific region of the dataset*

Description

This function returns the visualization of an specific region of the dataset.

Usage

```
Clipping(file_path, isovalue, verification, color_mesh, opacity,
  new_window, range_x, range_y, range_z)
```

Arguments

file_path	A string: the path to the nhdr file
isovalue	A number: the value corresponding to the desired isosurface
verification	A boolean: determines whether the grid verification will be performed
color_mesh	A string: the color to rendirind the resulting mesh
opacity	A number: the opacity-level
new_window	A boolean: determines whether a new view window will open
range_x	A vector contaning the clipping limits of the dataset in the x axis.
range_y	A vector contaning the clipping limits of the dataset in the y axis.
range_z	A vector contaning the clipping limits of the dataset in the z axis.

Value

The visualization of the generated mesh.

Examples

```
Clipping(system.file("extdata","f3.nhdr",package ="tcie"),0,FALSE,"red",1,TRUE,c(2,3),c(2,4),c(2,4))
Clipping(system.file("extdata","f3.nhdr",package ="tcie"),0,TRUE,"red",1,TRUE,c(2,3),c(2,4),c(2,4))
Clipping(system.file("extdata","f9.nhdr",package ="tcie"),0,TRUE,"blue",1,TRUE,c(1,5),c(2,4),c(3,5))
```

DataInfo — *Returns the dataset size and the isovalue range of a given dataset.*

Description

This function returns the size and isovalue range of a given dataset.

Usage

```
DataInfo(file_path)
```

Arguments

file_path	A string: the path to the nhdr file

Value

A vector with the dataset size and the minimum and maximum values of a given dataset.

Examples

```
DataInfo(system.file("extdata", "f3.nhdr", package = "tcie"))
DataInfo(system.file("extdata", "f3.nhdr", package = "tcie"))
```

ExpClipping — *Builds and exports the mesh representing the desired isosurface of an specific region of the dataset*

Description

This function returns the file (ply format) of the mesh of an specific region of the dataset.

Usage

```
ExpClipping(file_path, isovalue, verification, range_x, range_y, range_z,
  export_path)
```

Arguments

file_path	A string: the path to the nhdr file
isovalue	A number: the value corresponding to the desired isosurface
verification	A boolean: determines whether the grid verification will be performed
range_x	A vector contaning the clipping limits of the dataset in the x axis.
range_y	A vector contaning the clipping limits of the dataset in the y axis.
range_z	A vector contaning the clipping limits of the dataset in the z axis.
export_path	A string: the path to a directory to put the mesh generated, following by the file name.

Value

The file of the mesh in the ply format.

Examples

```
ExpClipping(system.file("extdata","f3.nhdr",package ="tcie"),0,FALSE,c(2,3),c(2,4),c(2,4),"m.ply")
ExpClipping(system.file("extdata","f3.nhdr",package ="tcie"),0,TRUE,c(2,3),c(2,4),c(2,4),"m.ply")
ExpClipping(system.file("extdata","f9.nhdr",package ="tcie"),0,TRUE,c(1,5),c(2,4),c(3,5),"m.ply")
```

ExpManifoldContour *Builds and exports the mesh representing the desired isosurface*

Description

This function returns the file (ply format) of the mesh generated by the Marching Cubes 33 algorithm. Optionally, the original dataset grid is preprocessed (subdivided in specifics points) to eliminate the configuration that results in non-manifold edges.

Usage

```
ExpManifoldContour(file_path, isovalue, verification, export_path)
```

Arguments

`file_path`	A string: the path to the nhdr file
`isovalue`	A number: the value corresponding to the desired isosurface
`verification`	A boolean: determines whether the grid verification will be performed
`export_path`	A string: the path to a directory to put the mesh generated, following by the file name.

Value

The file of the mesh in the ply format.

Examples

```
ExpManifoldContour(system.file("extdata", "f3.nhdr", package = "tcie"), 0.0, FALSE, "mesh.ply")
ExpManifoldContour(system.file("extdata", "f3.nhdr", package = "tcie"), 0.0, TRUE, "mesh.ply")
ExpManifoldContour(system.file("extdata", "f9.nhdr", package = "tcie"), 0.0, TRUE, "mesh.ply")
```

ManifoldContour *Builds and plots the mesh representing the desired isosurface*

Description

This function returns the visualization (rendered by the rgl package) of the mesh generated by the Marching Cubes 33 algotithm. Optionally, the original dataset grid is preprocessed (subdivided in specifics points) to eliminate some configuration which results in non-manifold edges.

Usage

```
ManifoldContour(file_path, isovalue, verification, color_mesh, opacity,
  new_window)
```

Arguments

file_path	A string: the path to the nhdr file
isovalue	A number: the value corresponding to the desired isosurface
verification	A boolean: determines whether the grid verification will be performed
color_mesh	A string: the color to rendirind the resulting mesh
opacity	A number: the opacity-level
new_window	A boolean: determines whether a new view window will be opened

Value

The visualization of the generated mesh.

Examples

```
ManifoldContour(system.file("extdata", "f3.nhdr", package = "tcie"), 0.0, FALSE,"red",1.0, TRUE)
ManifoldContour(system.file("extdata", "f3.nhdr", package = "tcie"), 0.0, TRUE,"red",1.0, TRUE)
ManifoldContour(system.file("extdata", "f9.nhdr", package = "tcie"), 0.0, TRUE, "blue",1.0, TRUE)
```

Index