Package 'Arothron'

March 19, 2018

Type Package

Title R functions for Geometric Morphometrics Analyses

Version 1.0.0

Author Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

Maintainer Antonio Profico <antonio.profico@uniroma1.it>

Description Tools for Geometric Morphometrics. The package includes tools of Virtual Anthropology to build virtual cavities as endocasts, to align two disarticulated models belonging to the same specimen, and functions to import and export format files into another format.

Depends R (>= 3.4.3)

Imports compositions (>= 1.40-1), doParallel (>= 1.0.11), foreach (>= 1.4.4), geometry (>= 0.3-6), graphics(>= 3.4.3), grDevices(>= 3.4.3), Morpho (>= 2.5.1), rgl (>= 0.93.963), Rvcg (>= 0.17), stats (>= 3.4.3), stringr (>= 1.3.0), utils (>= 3.4.3), vegan (>= 2.4-5)

License GPL-2

Encoding UTF-8

LazyLoad yes

RoxygenNote 6.0.1.9000

NeedsCompilation no

R topics documented:

`Arothron-package` *R functions for Geometric Morphometrics Analyses*

Description

Tools for Geometric Morphometrics. The package includes tools of Virtual Anthropology to build virtual cavities as endocasts, to align two disarticulated models belonging to the same specimen, and functions to import and export format files into another format.

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

`aro.clo.points` *aro.clo.points*

Description

Find the closest matches between a reference (2D or 3D matrix) and a target matrix (2D/3D) or mesh returning row indices and distances

Usage

```
aro.clo.points(target, reference)
```

Arguments

`target`	kxm matrix or object of class "mesh3d"
`reference`	numeric: a kxm matrix (coordinates)

Value

position numeric: a vector of the row indices

distances numeric: a vector of the coordinates distances

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

Examples

```
## Not run:
#load an example: mesh, and L set
data(yoda_sur)
data(yoda_set)
sur<-yoda_sur
set<-yoda_set
example<-aro.clo.points(target=sur,reference=set)

## End(Not run)
```

`bary.mesh` *bary.mesh*

Description

This function calculates the barycenter of a matrix or a 3D mesh

Usage

```
bary.mesh(mesh)
```

Arguments

`mesh`	matrix mesh vertex

Value

barycenter numeric: x,y,z coordinates of the barycenter of the mesh

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

Examples

```
## Not run:
#load an example: mesh, and L set
data(SCP1.mesh)
sur<-SCP1.mesh
bary<-bary.mesh(mesh=sur)

## End(Not run)
```

compare_check.set *compare_check.set*

Description

This function applyes the Digital Alignment Tool (DTA) on a disarticulated model using a reference landmark configuration

Usage

```
compare_check.set(RM_set_1, RM_set_2, DM_set_1, DM_set_2, DM_mesh_1, DM_mesh_2)
```

Arguments

RM_set_1	matrix: 3D landmark set of the first module acquired on the reference model
RM_set_2	matrix: 3D landmark set of the second module acquired on the reference model
DM_set_1	matrix: 3D landmark set of the first module acquired on the disarticulated model
DM_set_2	matrix: 3D landmark set of the second module acquired on the disarticulated model
DM_mesh_1	mesh3d: mesh of the disarticulated model (first module)
DM_mesh_2	mesh3d: mesh of the disarticulated model (second module)

Value

SF1 numeric: scale factor used to scale the reference set (first module)

SF2 numeric: scale factor used to scale the reference set (second module)

RM_set_1_sc matrix: scaled 3D reference set (first module)

RM_set_2_sc matrix: scaled 3D reference set (second module)

AM_model list: output of the Morpho::rotmesh.onto function

dist_from_mesh numeric: mesh distance between the aligned model and the scaled reference set

eucl_dist_1 numeric: euclidean distance between the landmark configuration of the disarticulated and reference model (first module)

eucl_dist_2 numeric: euclidean distance between the landmark configuration of the disarticulated and reference model (second module)

procr_dist numeric: procrustes distance between the landmark configuration of the aligned and reference model

procr_dist_1 numeric: procrustes distance between the landmark configuration of the disarticulated and reference model (first module)

procr_dist_2 numeric: procrustes distance between the landmark configuration of the disarticulated and reference model (second module)

eucl_dist numeric: euclidean distance between the landmark configuration of the aligned and reference model

single_l_1 numeric: euclidean distance between the landmark configuration of the disarticulated and reference model (first module)

single_l_2 numeric: euclidean distance between the landmark configuration of the disarticulated and reference model (second module)

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

`dec.curve` *dec.curve*

Description

This function computes the order of points on a open 3D curve and finds intermediate points

Usage

```
dec.curve(mat_input, mag, plot = TRUE)
```

Arguments

`mat_input`	numeric: a kx3 matrix
`mag`	numeric: how many times will be divided by the number of initial points
`plot`	logical: if TRUE will be plotted the starting and final point matrices

Value

matt numeric: a kx3 matrix with points coordinates

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

`dta` *dta*

Description

This function applyes the Digital Alignment Tool (DTA) on a disarticulated model using a reference sample

Usage

```
dta(RM_sample, mod_1, mod_2, pairs_1, pairs_2, DM_mesh_1, DM_mesh_2, DM_set_1,
  DM_set_2, method = c("euclidean"))
```

Arguments

RM_sample	3D array: 3D landmark configurations of the reference sample
mod_1	numeric vector: vector containing the position of which landmarks belong to the first module
mod_2	numeric vector: vector containing the position of which landmarks belong to the second module
pairs_1	matrix: a X x 2 matrix containing the indices of right and left landmarks of the first module
pairs_2	matrix: a X x 2 matrix containing the indices of right and left landmarks of the second module
DM_mesh_1	mesh3d: mesh of the disarticulated model (first module)
DM_mesh_2	mesh3d: mesh of the disarticulated model (second module)
DM_set_1	matrix: 3D landmark set of the first module acquired on the disarticulated model
DM_set_2	matrix: 3D landmark set of the second module acquired on the disarticulated model
method	character: specify method to be used to individuate the best DTA ("euclidean" or "procrustes")

Value

AM_mesh mesh3d: mesh of the aligned model

AM_set matrix: landmark configuration of the aligned model

AM_id character: name of the item of the reference sample resulted as best DTA

AM_SF_1 numeric: scale factor used to scale the reference set (first module)

AM_SF_2 numeric: scale factor used to scale the reference set (second module)

distance numeric: distance between the landmark configuration of the aligned and the reference model

tot_proc numeric vector: procrustes distances between aligned and reference models (all DTAs)

tot_eucl numeric vector: euclidean distances between aligned and reference models (all DTAs)

setarray 3D array: landmark configurations of the disarticulated model aligned on each item of the reference sample

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

endo_set *example dataset*

Description

POVs defined inside the endocranial cavity

Usage

```
data(endo_set)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

`export.amira.path` *export.amira.path*

Description

Convert and save a 3D matrix into a AmiraMesh ASCII Lineset (.am) object

Usage

```
## S3 method for class 'amira.path'
export(vertices, filename, Lines = c(1:(dim(vertices)[1]
  - 1) - 1, -1), path = getwd())
```

Arguments

`vertices`	numeric: a kx3 matrix
`filename`	character: name of the requested output
`Lines`	numeric: sequence of the vertices that defines the line
`path`	character: folder path

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

Examples

```
## Not run:
x<-c(1:20)
y<-seq(1,3,length=20)
z<-rnorm(20,0.01)
vertices<-cbind(x,y,z)
example<-export.amira.path(vertices=vertices,filename="example_line")

## End(Not run)
```

`export_amira` *export_amira*

Description

This function exports a list of 3D landmark set in separate files (format landmarkAscii)

Usage

```
export_amira(lista, path = getwd())
```

Arguments

lista	list containing 3D landmark sets
path	character: path of the folder where saving the Amira landmark sets

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

ext.int.mesh *ext.int.mesh*

Description

This function finds the vertices visible from a set of points of view

Usage

```
ext.int.mesh(mesh, views = 20, dist.sphere = 3, param1 = 2.5,
  param2 = 10, default = TRUE, import_pov, matrix_pov, expand = 1,
  scale.factor, method = "ast3d", start.points = 250, num.cores = NULL)
```

Arguments

mesh	object of class mesh3d
views	numeric: number of points of view
dist.sphere	numeric: scale factor. This parameter the distance betweem the barycenter of the mesh and the radius of the sphere used to define set of points of view
param1	numeric: first parameter for spherical flipping (usually ranged from 0.5 to 5, try!)
param2	numeric second paramter for spherical flipping (don't change it!)
default	logical: if TRUE the points of views are defined automatically, if FALSE define the matrix_pov
import_pov	logical: if FALSE an interactive 3D plot for the definition of the points of view is returned
matrix_pov	matrix: external set of points of view
expand	numeric: scale factor for the grid for the interactive 3D plot
scale.factor	numeric: scale factor for
method	character: select "a" or "c"
start.points	numeric: number of POVs available
num.cores	numeric: number of cores

Value

position numeric: a vector with vertex number nearest the landmark set

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

`ext.mesh.rai` *ext.mesh.rai*

Description

This function returns a 3D mesh with colours based on the vertices visibile from each point of view

Usage

```
ext.mesh.rai(scans, mesh)
```

Arguments

`scans`	an ext.int.mesh
`mesh`	matrix mesh vertex (the same of the ext.int.mesh object)

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

`grid_pov` *grid_pov*

Description

This function creates a grid for an interactive way to define the set of the points of view

Usage

```
grid_pov(mesh, expand = 1)
```

Arguments

`mesh`	object of class mesh3d
`expand`	numeric: scale factor for the grid for the interactive 3D plot

Value

matrice matrix: matrix with the x,y,z coordinates of the points of view

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

human_skull *example dataset*

Description

3D mesh of a human skull

Usage

```
data(human_skull)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

krd1_tooth *example dataset*

Description

3D mesh of a decidous Neanderthal tooth

Usage

```
data(krd1_tooth)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

landmark_frm2amira *landmark_frm2amira*

Description

This function converts the .frm files, from Evan Toolbox, stored in a folder into the format landmarkAscii

Usage

```
landmark_frm2amira(path_folder_frm, path_amira_folder)
```

Arguments

`path_folder_frm`
: character: path of the folder where the .frm files are stored

`path_amira_folder`
: character: path folder to store the landmarkAscii configurations

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

malleus_bone *example dataset*

Description

3D mesh of a human malleus

Usage

```
data(malleus_bone)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

noise.mesh *noise.mesh*

Description

This function adds noise to a mesh

Usage

```
noise.mesh(mesh, noise = 0.025, seed = 123)
```

Arguments

mesh	triangular mesh stored as object of class "mesh3d"
noise	sd deviation to define vertex noise
seed	seed for random number generator

Value

mesh_n a 3D model of class "mesh3d" with noise

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

`out.inn.mesh` *out.inn.mesh*

Description

This function separates a 3D mesh subjected to the ext.int.mesh into two 3D models: the visible mesh and the not visible one

Usage

```
out.inn.mesh(scans, mesh, plot = TRUE)
```

Arguments

`scans`	an ext.int.mesh
`mesh`	matrix mesh vertex (the same of the ext.int.mesh object)
`plot`	logical: if TRUE the wireframe of the mesh with the visible vertices is plotted

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

Examples

```
## Not run:
#CA-LSE tool on Neanderthal tooth
#load a mesh
data(krd1_tooth)
ca_lse_krd1<-ext.int.mesh(mesh= krd1_tooth, views=50, param1=3, default=TRUE,
import_pov = NULL,expand=1, scale.factor=1,num.cores = 4)
vis_inv_krd1<-out.inn.mesh(ca_lse_krd1, krd1_tooth, plot=TRUE)
inv_mesh<-vcgIsolated(vis_inv_krd1$invisible)
open3d()
shade3d(inv_mesh,col=2)
open3d()
shade3d(vis_inv_krd1$visible, col=3)
#CA-LSE tool on human malleus
#load a mesh
data(malleus_bone)
ca_lse_malleus<-ext.int.mesh(mesh= malleus_bone, views=50, param1=3,
default=TRUE, import_pov = NULL, expand=1, scale.factor=1)
vis_inv_malleus<-out.inn.mesh(ca_lse_malleus, malleus_bone, plot=TRUE)
inv_mesh<- vis_inv_malleus$invisible
inv_mesh<-ca_lse_malleus$invisible

#AST-3D tool
#load a mesh
data(human_skull)
data(endo_set)
ast3d_endocast<-ext.int.mesh(mesh=human_skull, views=50, param1=1.0, default=FALSE,
import_pov = TRUE,expand=1, matrix_pov =endo_set, scale.factor=1,num.cores = 4)
vis_inv_endo<-out.inn.mesh(ast3d_endocast,human_skull,plot=TRUE)
vis_mesh<-vcgIsolated(vis_inv_endo$visible)
```

```
open3d()
shade3d(vis_mesh,col=3)
open3d()
shade3d(vis_inv_endo$invisible, col=2)

## End(Not run)
```

`patches_frm2amira` *patches_frm2amira*

Description

This function converts the .frm files, from Evan Toolbox, stored in a folder into the format landmarkAscii (semilandmark patches)

Usage

```
patches_frm2amira(path_folder_frm, path_amira_folder)
```

Arguments

`path_folder_frm`
: character: path of the folder where the .frm files are stored

`path_amira_folder`
: character: path folder to store the landmarkAscii configurations

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

`pov_selecter` *pov_selecter*

Description

Internal function to define the points of view

Usage

```
pov_selecter(mesh, grid, start.points = 250, method = "ast3d")
```

Arguments

`mesh`	object of class mesh3d
`grid`	matrix: a 3D grid
`start.points`	numeric: number of center to be found
`method`	character: select "a" or "c" for respectively AST-3D and CA-LSE method

Value

selection numeric: positioning vector of the selected points of the grid

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

read.amira.dir *read.amira.dir*

Description

This function reads and stores in an array the coordinated allocated in a folder in separate files (format landmarkAscii)

Usage

```
read.amira.dir(path.dir, nland)
```

Arguments

path.dir	character: path of the folder
nland	numeric: number of landmark sampled in Amira

Value

array.set numeric: a kx3xn array with landmark coordinates

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

read.amira.set *read.amira.set*

Description

This function converts a landmarkAscii file set in a kx3x1 array

Usage

```
read.amira.set(name.file, nland)
```

Arguments

name.file	character: path of a landmarkAscii file
nland	numeric: number of landmark sampled in Amira, if is set on "auto" it will be automatically recognized

Value

array.set numeric: a kx3x1 array with landmark coordinates

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

read.path.amira *read.path.amira*

Description

This function extracts and orders the coordinate matrix from a surface path file from Amira

Usage

```
read.path.amira(path.name)
```

Arguments

path.name	character: path of surface path .ascii extension file

Value

data numeric: a kxd matrix with xyz coordinates

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

repmat *repmat*

Description

This function repeats copies of a matrix

Usage

```
repmat(X, m, n)
```

Arguments

X	numeric: a matrix
m	numeric: number of times to repeat the X matrix in row and column dimension
n	numeric: repetition factor for each dimesion

Value

matrice: repeated matrix

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

SCP1.mesh *example dataset*

Description

Mesh of the Saccopastore 1 Neanderthal skull

Usage

```
data(SCP1.mesh)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

---

sinus_set *example dataset*

Description

POVs sampled inside the maxillary sinus cavity

Usage

```
data(sinus_set)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

spherical.flipping *spherical.flipping*

Description

Internal spherical flippping function

Usage

```
spherical.flipping(C, mesh, param1, param2)
```

Arguments

C	numeric: coordinates of the point of view
mesh	object of class mesh3d
param1	numeric: first parameter for spherical flipping (usually ranged from 0.1 to 3, try!)
param2	numeric second paramter for spherical flipping (don't change it!)

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

trasf.mesh *trasf.mesh*

Description

This function centers a mesh on the barycenter coordinates

Usage

```
trasf.mesh(mesh, barycenter)
```

Arguments

mesh	a 3D mesh of class "mesh3d"
barycenter	numeric: coordinates of the center

Value

mesh a 3D mesh of class "mesh3d"

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

yoda_set *example dataset*

Description

Landmark set on Yoda

Usage

```
data(yoda_set)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia

yoda_sur *example dataset*

Description

Mesh of Yoda

Usage

```
data(yoda_sur)
```

Author(s)

Antonio Profico, Alessio Veneziano, Marina Melchionna, Pasquale Raia