Package 'brainR'

April 1, 2025

Type Package

Title Helper Functions to 'misc3d' and 'rgl' Packages for Brain Imaging

Version 1.7.0

Maintainer John Muschelli <muschellij2@gmail.com>

Description This includes functions for creating 3D and 4D images using 'WebGL', 'rgl', and 'JavaScript' commands. This package relies on the X toolkit ('XTK', <https://github.com/xtk/X#readme>).

License GPL-2

Depends rgl, misc3d, oro.nifti

Imports grDevices

RoxygenNote 7.3.2

Encoding UTF-8

Suggests servr

NeedsCompilation no

Author John Muschelli [aut, cre]

Repository CRAN

Date/Publication 2025-04-01 16:00:09 UTC

Contents

brainR-package 2
makeScene 2
scene4d 3
write4D 4
write4D.file 6
writeTrianglesSTL 7
writeWebGL_split 8

Index **10**

brainR-package *Functions to render 3D brain images in html*

Description

Brain Template from Copyright (C) 1993-2009 Louis Collins, McConnell Brain Imaging Centre, Montreal Neurological Institute, McGill University 6th generation non-linear symmetric brain

Author(s)

John Muschelli <muschellij2@gmail.com>

References

G. Grabner, A. L. Janke, M. M. Budge, D. Smith, J. Pruessner, and D. L. Collins, "Symmetric atlasing and model based segmentation: an application to the hippocampus in older adults", Med Image Comput Comput Assist Interv Int Conf Med Image Comput Comput Assist Interv, vol. 9, pp. 58-66, 2006.

See Also

contour3d, rgl

makeScene *Make Leveled Scene*

Description

Make scene returns a list of levels - but makes them mutually distinct. So if cutoff 0.1, 0.2, then 0.1<= x < 0.2 is an roi, not > 0.1 and > 0.2. Different than contour3d as these are mutually exclusive levels.

Usage

```
makeScene(data, cutoffs, alpha, cols)
```

Arguments

data	- 3D array of values (can be nifti-class)
cutoffs	- series of levels to be created
alpha	- alpha levels for each contour
cols	- colors for each contour

Value

scene with multiple objects - can be passed to write4D

scene4d *Wrapper to write a 4D scene*

Description

This function takes in filenames, levels, and creates an output html file, with 4D elements. The html is based on XTK (https://github.com/xtk/X#readme)

Usage

```
scene4d(
  files,
  fnames = NULL,
  outfile = "index_4D_stl.html",
  levels = NULL,
  alpha = NULL,
  color = "white",
  useTemp = FALSE,
  MNITemp = c("1mm", "2mm"),
  objtype = "stl",
  ...
)
```

Arguments

files	(character) vector of filenames (first being a brain file if useTemp=FALSE)
fnames	(character) filenames for the 3D surfaces in the scene - needs to be the same length as files
outfile	(character) html filename
levels	(numeric/list) levels to make contours/surfaces for each file. Either a numeric vector may be passed, one level for each file. Or a list of numeric vectors of multiple levels for each file. Will be coerced to a list.
alpha	(numeric/list) alpha opacities for each contours/surface for each file. Will be coerced to list similarly as levels
color	(character/list) colors for each contours/surface for each file. Will be coerced to list similarly as levels
useTemp	(logical) whether to use template from brainR as the brain figure
MNITemp	(character) if (useTemp = TRUE) either "1mm" or "2mm" denoting the resolution of the template used
objtype	(character) object type to write the files to. Either "stl", "obj", or "ply" to write.
...	other options to be passed to write4D

Examples

```
### Faster - 8mm resampled but very coarse
imgs <- paste("Visit_", 1:5, "_8mm.nii.gz", sep="")
ifiles <- sapply(imgs, system.file, package='brainR')
files = file.path(tempdir(), basename(ifiles))
file.copy(ifiles, files)
outfile <-  file.path(tempdir(), "index_4D_stl.html")
scene4d(files, levels=rep(0.99, length(files)),
outfile = outfile, color= rep("blue", length(files)), useTemp=TRUE,
MNITemp = "8mm", alpha = rep(1, length(files)), rescale=TRUE  )
## Not run:
imgs <- paste("Visit_", 1:5, ".nii.gz", sep="")
ifiles <- sapply(imgs, system.file, package='brainR')
files = file.path(tempdir(), basename(ifiles))
file.copy(ifiles, files)
scene4d(files, levels=rep(0.99, length(files)),
outfile = outfile,
color= rep("blue", length(files)), useTemp=TRUEge,
MNITemp = "8mm", alpha = rep(1, length(files)), rescale=TRUE  )

## End(Not run)
```

`write4D` *Wrapper to write a 4D scene*

Description

This function takes in a scene and writes it out to a series of files either with the stl format or obj format (see writeOBJ and writeSTL)

Usage

```
write4D(
  scene,
  outfile,
  fnames = NULL,
  captions = NULL,
  writefiles = TRUE,
  reprint = TRUE,
  ...
)
```

Arguments

`scene`	list of 3D triangles (see contour3d). If a multicolored object is to be rendered (multiple contours with one control) - it must be in a list
`outfile`	html filename that is to be exported
`fnames`	filenames for the 3D surfaces in the scene - needs to be the same length as scene

captions	labels for checkboxes on html webpage
writefiles	(experimental) simply run the code to create the html and not write the .obj or .stl files
reprint	(logical, experimental) do you want to reprint the rgl before saving (common use by rgl functions)
...	other options to be passed to write4D.file

Examples

```
#Brain Template from Copyright (C) 1993-2009 Louis Collins,
#McConnell Brain Imaging Centre,
#Montreal Neurological Institute, McGill University
#6th generation non-linear symmetric brain
##Downsampled to 8mm using FSL fslmaths -subsamp2

template <- readNIfTI(system.file("MNI152_T1_8mm_brain.nii.gz", package="brainR")
, reorient=FALSE)
dtemp <- dim(template)
### 4500 - value that empirically value that presented a brain with gyri
### lower values result in a smoother surface
brain <- contour3d(template, x=1:dtemp[1], y=1:dtemp[2],
z=1:dtemp[3], level = 4500, alpha = 0.8, draw = FALSE)

### Example data courtesy of Daniel Reich
### Each visit is a binary mask of lesions in the brain
imgs <- paste("Visit_", 1:5, "_8mm.nii.gz", sep="")
files <- sapply(imgs, system.file, package='brainR')
scene <- list(brain)
## loop through images and thresh
nimgs <- length(imgs)
cols <- rainbow(nimgs)
for (iimg in 1:nimgs) {
mask <- readNIfTI(files[iimg], reorient=FALSE)
if (length(dim(mask)) > 3) mask <- mask[,,,1]
### use 0.99 for level of mask - binary
  activation <- contour3d(mask, level = c(0.99), alpha = 1,
  add = TRUE, color=cols[iimg], draw=FALSE)
## add these triangles to the list
scene <- c(scene, list(activation))
}
## make output image names from image names
fnames <- c("brain.stl", gsub(".nii.gz", ".stl", imgs, fixed=TRUE))
fnames = file.path(tempdir(), fnames)
outfile <-  file.path(tempdir(), "index.html")
write4D(scene=scene, fnames=fnames, outfile=outfile, standalone=TRUE,
rescale=TRUE)
if (interactive()) {
if (requireNamespace("servr", quietly = TRUE)) {
   servr::httd(tempdir())
}
}
```

```
unlink(outfile)
```

`write4D.file` *Write a 4D scene*

Description

This function takes in a scene and writes it out to a series of files either with the stl format or obj format

Usage

```
write4D.file(
  scene = NULL,
  outfile = "index_4D.html",
  fnames,
  visible = TRUE,
  opacity = 1,
  colors = NULL,
  captions = "",
  standalone = FALSE,
  rescale = FALSE,
  index.file = system.file("index_template.html", package = "brainR"),
  toggle = "checkbox",
  xtkgui = FALSE
)
```

Arguments

`scene`	- list of 3D triangles (see contour3d). If a multicolored object is to be rendered (multiple contours with one control) - it must be in a list
`outfile`	- html filename that is to be exported
`fnames`	- filenames for the 3D surfaces in the scene - needs to be the same length as scene
`visible`	- logical vector indicating which structures are visible in html file
`opacity`	- list of alpha values - same length as scene; if sub-structures are present, then the each list element has length the number of structures
`colors`	- character vector of colors (col2rgb is applied)
`captions`	- labels for checkboxes on html webpage
`standalone`	- logical - should this be able to be rendered offline?
`rescale`	- rescale the scene? - in beta
`index.file`	- template html file used
`toggle`	- (experimental) "checkbox" (default) or "radio" for radio or checkboxes to switch thing
`xtkgui`	- (experimental) Logical to use xtkgui for objects

See Also

writeOBJ, writeSTL, contour3d

Examples

```
template <- readNIfTI(system.file("MNI152_T1_8mm_brain.nii.gz", package="brainR")
, reorient=FALSE)
dtemp <- dim(template)
### 4500 - value that empirically value that presented a brain with gyri
### lower values result in a smoother surface
brain <- contour3d(template, x=1:dtemp[1], y=1:dtemp[2],
z=1:dtemp[3], level = 4500, alpha = 0.8, draw = FALSE)

### Example data courtesy of Daniel Reich
### Each visit is a binary mask of lesions in the brain
imgs <- paste("Visit_", 1:5, "_8mm.nii.gz", sep="")
files <- sapply(imgs, system.file, package='brainR')
scene <- list(brain)
## loop through images and thresh
nimgs <- length(imgs)
cols <- rainbow(nimgs)
for (iimg in 1:nimgs) {
mask <- readNIfTI(files[iimg], reorient=FALSE)
if (length(dim(mask)) > 3) mask <- mask[,,,1]
### use 0.99 for level of mask - binary
  activation <- contour3d(mask, level = c(0.99), alpha = 1,
  add = TRUE, color=cols[iimg], draw=FALSE)
## add these triangles to the list
scene <- c(scene, list(activation))
}
## make output image names from image names
fnames <- c("brain.stl", gsub(".nii.gz", ".stl", imgs, fixed=TRUE))
fnames = file.path(tempdir(), fnames)
outfile <-  file.path(tempdir(), "index.html")
write4D.file(
scene=scene, fnames=fnames,
visible = FALSE,
outfile=outfile, standalone=TRUE, rescale=TRUE)



unlink(outfile)
unlink(fnames)
```

---

writeTrianglesSTL *Write STL triangles (without recalling the ids)*

Description

This is code extracted from writeSTL in rgl. This allows users to write the triangles in STL without reprinting the rgl (which takes time)

Usage

```
writeTrianglesSTL(scene, con, ascii = FALSE)
```

Arguments

scene	list of triangles (that have class Triangles3D)
con	filename or connection of stl file to write
ascii	indicator if the file should be written in ascii or binary

Value

filename (invisible) of stl object

---

writeWebGL_split *Write WebGL with split triangles*

Description

Adapted writeWebGL function that splits the triangles into 65535 vertices

Usage

```
writeWebGL_split(ids = rgl.ids()$id, writeIt = TRUE, verb = FALSE, ...)
```

Arguments

ids	- rgl ids (see rgl.ids)
writeIt	- (logical) write the file out
verb	- verbose output
...	- further arguments passed to writeWebGL

Value

if writeIt is TRUE, then returns the value from writeWebGL. Otherwise, returns the split triangles from the rgl objects

Examples

```
## Not run:
#Brain Template from Copyright (C) 1993-2009 Louis Collins,
#McConnell Brain Imaging Centre,
#Montreal Neurological Institute, McGill University
#6th generation non-linear symmetric brain
template <- readNIfTI(system.file("MNI152_T1_2mm_brain.nii.gz", package="brainR")
, reorient=FALSE)
dtemp <- dim(template)
### 4500 - value that empirically value that presented a brain with gyri
```

```
### lower values result in a smoother surface
brain <- contour3d(template, x=1:dtemp[1], y=1:dtemp[2],
z=1:dtemp[3], level = 4500, alpha = 0.1, draw = FALSE)
drawScene.rgl(brain)
### this would be the ``activation'' or surface you want to render -
# hyper-intense white matter
contour3d(template, level = c(8200, 8250),
alpha = c(0.5, 0.8), add = TRUE, color=c("yellow", "red"))
### add text
text3d(x=dtemp[1]/2, y=dtemp[2]/2, z = dtemp[3]*0.98, text="Top")
text3d(x=-0.98, y=dtemp[2]/2, z = dtemp[3]/2, text="Right")
fname <- "knitted_webGL.html"
writeWebGL_split(dir=getwd(), filename =fname,
template = system.file("my_template.html", package="brainR"), width=500,
writeIt=TRUE)
browseURL(fname)

## End(Not run)
```

Index

∗ **brainR**
brainR-package, 2
∗ **package**
brainR-package, 2

brainR *(*brainR-package*)*, 2
brainR-package, 2

contour3d, *2*, *4*, *6, 7*

makeScene, 2

nifti-class, *2*

rgl, *2*
rgl.ids, *8*

scene4d, 3

write4D, *2, 3*, 4
write4D.file, *5*, 6
writeOBJ, *4*, *7*
writeSTL, *4*, *7*
writeTrianglesSTL, 7
writeWebGL, *8*
writeWebGL_split, 8