Package ‘VoxR’

February 19, 2015

Type Package
Title Metrics extraction of trees from T-LiDAR data
Version 0.5.1
Date 2013-05-28
Author Bastien Lecigne, Sylvain Delagrange and Christian Messier
Maintainer Bastien Lecigne <lecionebastien@gmail.com>
Depends R (>= 2.10)
Suggests rgl, raster, sp
Description Tools for tree crown structure description based on T-LiDAR data voxelisation
License GPL (>= 2.0)
NeedsCompilation no
Repository CRAN
Date/Publication 2014-01-29 19:30:02

R topics documented:

VoxR-package ................................................................. 2
axis.angle ................................................................. 3
axis.distance .............................................................. 4
data1 ................................................................. 6
data2 ................................................................. 6
data_part ............................................................... 7
level ................................................................. 8
obj.rec ............................................................... 9
point.distance ....................................................... 10
project ............................................................... 11
raster.proj .......................................................... 13
sub.obj ............................................................. 15
surface ............................................................ 16
treecloud .......................................................... 17
treecloud_vox ..................................................... 18
vox ................................................................. 19
Description

Tools for a geometric description of tree crown, tree growth and spatially differentiated objects recognition based on point cloud voxelisation.

Details

VoxR offers tools for 3d point cloud voxelisation (see vox), 3d point cloud projection and visualization of projections (see project, raster.proj, level and surface), geometrical description of 3d point cloud (see point.distance, axis.distance and axis.angle), tree growth analysis based on voxelised LiDAR point clouds comparison (see sub.obj) and spatially differentiated objects recognition (see obj.rec).

Author(s)

Bastien Lecigne, Sylvain Delagrange and Christian Messier
Maintainer : Bastien Lecigne <lecignebastien@gmail.com>

References

This package uses previous work from the following packages :


These packages must be installed to use the function raster.proj.

The examples of 3d functions are plotted using the plot3d function from the rgl package : Daniel Adler and Duncan Murdoch (2013). rgl: 3D visualization device system (OpenGL). R package version 0.93.932. http://CRAN.R-project.org/package=rgl

Examples

data(treecloud)

#-voxelisation using the vox function

treecloud_vox <- vox(treecloud,res=0.02)

#-visualisation

require(rgl)
library(rgl)
Open 3D object.

plot3d(treecloud_vox, size=0.1)

---

### axis.angle

#### Computing angle of points with an axis

**Description**

Computing angles of points with an axis (x, y or z) and the origin of the 3D Cartesian coordinate system.

**Usage**

```r
axis.angle(data, axis, projected, plan)
```

**Arguments**

- `data` : a data frame containing the x, y, z, ... coordinates of a point cloud
- `axis` : character string specifying the reference axis to compute the angles : "X", "Y" or "Z"
- `projected` : logical : if TRUE the original point cloud will be projected in a 2D coordinate system
- `plan` : character string specifying the plan of projection (only if projected = TRUE), see details for more information.

**Details**

Complementary information for the plan parameter : if axis = "X", projected should be "xz" or "xy" ; if axis = "Y", projected should be "xy" or "xz" ; if axis = "Z", projected should be "xz" or "yz".

Default : projected = FALSE.

**Value**

A vector containing the angle values of the points.

**Author(s)**

Bastien Lecigne, Sylvain Delagrange and Christian Messier

**See Also**

- `point.distance`
- `axis.distance`
Examples

data(treecloud_vox)

########################
#- using projection
#- computing angles

dist <- axis.angle(treecloud_vox, axis="X", projected=TRUE, plan="xy")
treecloud_vox[,4] <- dist

#- density plot
plot(density(dist, na.rm=TRUE))

#- visualisation
z <- c(sort(unique(round(treecloud_vox[,4], digits=0)), decreasing=TRUE))
col <- rainbow(n=length(z), start=0, end=2/6)
library(rgl)
open3d()
for(i in 1:length(z)){
a <- subset(treecloud_vox, round(treecloud_vox[,4], digits=0)==z[i])
plot3d(a, col=col[i], add=TRUE)}

########################
#- without projection
#- computing angles

dist <- axis.angle(treecloud_vox, axis="X", projected=FALSE)
treecloud_vox[,4] <- dist

#- density plot
plot(density(dist, na.rm=TRUE))

#- visualisation
z <- c(sort(unique(round(treecloud_vox[,4], digits=0)), decreasing=TRUE))
col <- rainbow(n=length(z), start=0, end=2/6)
open3d()
for(i in 1:length(z)){
a <- subset(treecloud_vox, round(treecloud_vox[,4], digits=0)==z[i])
plot3d(a, col=col[i], add=TRUE)
}

---

axis.distance  Computing distance of points from an axis
Description
Computing distance of points with an axis (x, y or z) and the origin of the 3d Cartesian coordinate system.

Usage
axis.distance(data, axis)

Arguments
- data: a data frame containing the x, y, z, ... coordinates of a point cloud
- axis: character string specifying the reference axis to compute the distances: "X", "Y" or "Z"

Details
Default: proportion = FALSE

Value
A vector containing the distance values of the points.

Author(s)
Bastien Lecigne, Sylvain Delagrange and Christian Messier

See Also
point.distance and axis.angle

Examples
```R
data(treecloud_vox)

#- computing distances
dist <- axis.distance(treecloud_vox,axis="Z")
treecloud_vox[,4] <- dist

#-density plot
plot(density(dist, na.rm=TRUE))

#-visualisation
z <- c(sort(unique(round(treecloud_vox[,4], digits=2)), decreasing=TRUE))
col <- rainbow(n=length(z), start=0, end=2/6)
require(rgl)
library(rgl)
open3d()
for(i in 1:length(z)) {
  # code for visualization
```
a <- subset(treecloud_vox, round(treecloud_vox[,4], digits=2) == z[i])
plot3d(a, col=col[i], add=TRUE)}

---

**data1**  
*Cubic point cloud*

**Description**

Cubic point cloud used as an example for the `sub.obj` function complementary toward `data2`.

**Usage**

```r
data(data1)
```

**Format**

A data frame with 9261 observations on the following 4 variables.

- **X**: a numeric vector
- **Y**: a numeric vector
- **Z**: a numeric vector
- **NBpts**: a numeric vector

**Examples**

```r
data(data1)
require(rgl)
library(rgl)
plot3d(data1)
```

---

**data2**  
*Cubic point cloud*

**Description**

Cubic point cloud used as an example for the `sub.obj` function complementary toward `data1`.

**Usage**

```r
data(data2)
```
**data_part**

**Format**

A data frame with 7140 observations on the following 4 variables.

- **X** a numeric vector
- **Y** a numeric vector
- **Z** a numeric vector
- **NBpts** a numeric vector

**Examples**

```r
data(dataR)
require(rgl)
library(rgl)
plot3d(dataR)
```

---

**data_part**  
**Geometrics objects spatially differentiated**

**Description**

Point cloud presenting spatially differentiated geometric objects used as an example for the `obj.rec` function.

**Usage**

```r
data(data_part)
```

**Format**

A data frame with 2382 observations on the following 4 variables.

- **X** a numeric vector
- **Y** a numeric vector
- **Z** a numeric vector
- **NBpts** a numeric vector

**Examples**

```r
data(data_part)
require(rgl)
library(rgl)
plot3d(data_part)
```
level

Density levels definition

Description

This function creates density levels from point clouds projected using the function *project*. It can be used as a sub function of the *raster.proj* function to discriminate voxels, point or ratio npts/nvox density.

Usage

level(datas, by, levels)

Arguments

datas a vector containing the values of a variable

by character string specifying which kind of discrimination will be used to define the density levels: "quantiles" or "percents".

levels a vector containing the values of discretisation (see details for more information).

Details

Details on levels parameter: if by = "quantiles" indicate the proportion of the variable contained in each level (if level=0.25 the quantiles will be returned, if level=0.2 the quintiles will be returned). If by = "percents" the levels are defined as a percentage of the variable (if level=c(0.25,0.5,0.75) the discretisation values will be respectively 25, 50, 75 and 100% of the maximum value of the variable).

Defaults: by = "quantiles" and if by = "quantiles", level = 0.25 ; if by = "percents", levels = c(25, 50, 75)

Value

A vector containing the values of the discretisation levels

Author(s)

Bastien Lecigne, Sylvain Delagrange and Christian Messier

See Also

surface
Examples

#- projection

data(treecloud_vox)
proj <- project(treecloud_vox, dim="xy")
nvox <- c(proj[,3])
npts <- c(proj[,4])
ratio <- c(proj[,5])

#- computing discretisation level

    # number of voxels by quantiles
lev_vox <- level(nvox, by="quantiles", levels=c(0, 0.2))

    # number of points by percents
lev_pts <- level(npts, by="percents", levels=c(0, 0.2, 0.4, 0.6, 0.8))

    # ratio npts/nvox by quantiles
lev_ratio <- level(ratio, by="quantiles", levels=c(0, 0.25))

# to see the levels
lev_vox
lev_pts
lev_ratio

obj.rec        Spatially differentiated objects recognition

Description

Recognition of spatially differentiated objects within a point cloud. Two points located under within distance of research from each other are considered as the parts of a unique object.

Usage

obj.rec(data, fac)

Arguments

data       a data frame containing the x, y, z, ... coordinates of a voxel cloud
fac         numeric specifying the distance of research (in the scale of the original coordinate system)

Value

A data frame containing the x, y, z coordinates and object ID of the input data

Note

This function can be time consuming if used on big data sets
**point.distance**

Computing distance of points from a unique point

**Description**

Computing distance of points from a unique point within a point cloud.

**Usage**

```r
point.distance(data, point)
```

**Arguments**

- `data`: a data frame containing the x, y, z, ... coordinates of a point cloud
- `point`: a vector containing the x, y, z coordinates of the reference point
**project**  

_Projection of voxels within a plan_

**Value**  

A vector containing the distance values of the points

**Author(s)**  

Bastien Lecigne, Sylvain Delagrange and Christian Messier

**See Also**  

axis.angle and axis.distance

**Examples**

```r
data(treecloud_vox)

#- computing distances
dist <- point.distance(treecloud_vox, point=c(0,0,0))
treecloud_vox[,4] <- dist

#-density plot
plot(density(dist, na.rm=TRUE))

#-3d visualisation
z <- c(sort(unique(round(treecloud_vox[,4],digits=2)),decreasing=TRUE))
col <- rainbow(n=length(z),start=0,end=2/6)
require(rgl)
library(rgl)
open3d()
for(i in 1:length(z)){
a <- subset(treecloud_vox,round(treecloud_vox[,4],digits=2)==z[i])
plot3d(a, col=col[i], add=TRUE)
}
```

**Description**  

Projection of voxels within a 2d coordinate system formed by two axes of the original coordinate system.

**Usage**

```r
project(data, dim)
```
Arguments

data a data frame containing the x, y, z, ... coordinates of a point cloud
dim a character string specifying the projection plan: "xy", "xz" or "yz"

Details

Default: dim = "xy"

Value

A data frame of a 2D point cloud containing: x, y coordinates of the pixels and the number of voxels (nvox), number of points (npts), ratio npts/nvox contained in each pixel.

Author(s)

Bastien Lecigne, Sylvain Delagrange and Christian Messier

Examples

data(treecloud_vox)

#-projection in "xy" plan
proj <- project(treecloud_vox, dim="xy")

#-creating vectors with interests variables
npts <- c(proj[,4])
nvox <- c(proj[,3])
ratio <- c(proj[,5])

#-discretisation level

  # by quantiles (20% quantiles or 25% quantiles)
lev_vox <- level(nvox, by="quantiles", levels=c(0.2))
lev_ratio <- level(ratio, by="quantiles", levels=c(0.25))

  # by percents
lev_pts <- level(npts, by="percents", levels=c(0.2, 0.4, 0.6, 0.8))

#- computing surfaces

  # surface in proportion
surf_nvox <- surface(proj, method="nvox", levels=lev_vox, res=0.02, proportion=TRUE)
  # surface in absolute
surf_ratio <- surface(proj, method="ratio", levels=lev_ratio, res=1, proportion=FALSE)
  # surface in absolute
surf_npts <- surface(proj, method="npts", levels=lev_pts, res=0.02, proportion=FALSE)

#- ploting raster images
raster.proj

Create a raster image from project, level and surface

Description

Easy tool for raster image creation from projections created by the project function using level function for density discretisation with the possibility of integrating an automatic caption.

Usage

raster.proj(data, title, res, method, levels, colors, contour, classlegend, surf, dim)

Arguments

data 2d point cloud x, y, npts, nvox, npts/nvox

title the raster’s title

res pixel resolution in the scale of the original orthonormed system

method a character string specifying the variable of interest from the projection ("nvox", "npts" ou "ratio")

levels a vector containing the density levels of discretisation as created by level

colors a vector containing the colors of the density classes
contour logical: if TRUE the contours of the density classes are drawn
classlegend a vector of characters string containing the names of the density classes for the caption
surf a vector containing the surface of each density class for the caption as created by the surface function
dim a character string specifying the dimension of projection ("xy", "yz", "xz")

Details

Defaults:
- method = "nvox"
- res = 1
- levels = quantiles 0.25
- title = ""
- colors = c(grey, green, yellow, red)
- contour = TRUE
- classlegend = ""
- surf = ""
- dim = "xy"

Value

Draw the raster image in the default R graphic device

Author(s)

Bastien Lecigne, Sylvain Delagrange and Christian Messier

References

This function use previous work from the following packages:

Examples

data(treecloud_vox)

#- projection in xy plan
proj <- project(treecloud_vox, dim="xy")

#- computing parameters for levels and surf
### Description
Isolation of unique voxels to one of two voxel clouds based on the detection of isolated voxels within a voxels of bigger dimensions.

### Usage
```
sub.obj(data1, data2, res, nvox.reaserch)
```

### Arguments
- **data1**: a data frame containing the x, y, z, ... coordinates of a point cloud from which "data2" will be substracted
- **data2**: a data frame containing the x, y, z, ... coordinates of a point cloud to substract to "data1"
- **res**: numeric specification of the resolution of the voxels (in the scale of the original coordinate system)
- **nvox.reaserch**: numeric specifying as a factor the resolution of the research’s voxel (nvox.reasearch * res)

### Value
A data frame containing the x, y, z coordinates of the voxels unique to data1

### Note
This function can be time consuming if used on big data sets

### Author(s)
Bastien Lecigne, Sylvain Delagrange and Christian Messier
Examples

```r
#- importing 2 data sets
data(data1)
data(data2)

#- subtraction of data2 to data1
sub <- sub.obj(data1, data2, res=0.5, vox.research=1)

#- visualisation
require(rgl)
library(rgl)
open3d()
plot3d(sub, col="red", add=TRUE)
plot3d(data2, add=TRUE)
```

---

**surface**

*Compute the surface of density classes of a projection*

Description

Compute the surface of density classes within a projection created by the `project` function and discretised by `level`. The surface is calculated using the following relationship: \( \text{level}_n \leq \text{surface} < \text{level}_{n+1} \)

Usage

```r
surface(data, method, levels, res, proportion)
```

Arguments

- `data`: a data frame of a 2D point cloud containing: x, y coordinates of the pixels and the number of voxels (nvox), number of points (npts), ratio npts/nvox contained in each pixel
- `method`: character string specifying the variable of interest of the projection: "nvox", "npts" or "ratio"
- `levels`: a vector containing the discretisation levels as created by the `level` function
- `res`: numeric definition of the resolution of the pixels. If `res = 1`, the calculated surface corresponds to the number of pixels.
- `proportion`: logical: if TRUE the surfaces are expressed in proportion to the total surface

Details

Defaults:

- `levels`: quantiles 0.25
- `method`: "nvox"
- `res`: 1
- `proportion`: FALSE
Value

A vector containing the surface of each density class

Author(s)

Bastien Lecigne, Sylvain Delagrange and Christian Messier

Examples

```r
#- projection
data(treecloud_vox)
proj <- project(treecloud_vox,dim="xy")

#- creating vectors with interests variables
npts <- c(proj[,4])
nvox <- c(proj[,3])
ratio <- c(proj[,5])

#-creating level of discretisation
lev_vox <- level(nvox,by="quantiles",levels=c(0.2))
lev_pts <- level(npts,by="percents",levels=c(0.2,0.4,0.6,0.8))
lev_ratio <- c(1,2)# <- level(ratio,by="quantiles",level=c(0.25))

#- computing surfaces
surf_nvox <- surface(proj,method="nvox",levels=lev_vox,res=0.02,proportion=TRUE)
surf_npts <- surface(proj,method="npts",levels=lev_pts,res=0.02,proportion=FALSE)
surf_ratio <- surface(proj,method="ratio",levels=lev_ratio,res=1,proportion=FALSE)
surf_nvox
surf_npts
surf_ratio
```

data(treecloud)

**treecloud**

*LiDAR scene of a tree*

Description

LiDAR scene of a tree digitized using an Iliris 3d (opech) T-LiDAR.

Usage

data(treecloud)
Format

A data frame with 680710 observations on the following 3 variables.

X a numeric vector
Y a numeric vector
Z a numeric vector

Examples

data(treecloud)
require(rgl)
library(rgl)
plot3d(treecloud)

treecloud_vox

Description

treecloud LiDAR scene voxelised using vox function with a 0.02m resolution

Usage

data(treecloud_vox)

Format

A data frame with 373939 observations on the following 4 variables.

X a numeric vector
Y a numeric vector
Z a numeric vector
NBpts a numeric vector

Examples

data(treecloud_vox)
require(rgl)
library(rgl)
plot3d(treecloud_vox)
Description

Voxelisation algorithm of 3d point cloud recording the number of points within each voxels

Usage

vox(data, res)

Arguments

data  a data frame containing the x, y, z, ... coordinates of a point cloud
res   numeric specification of the voxels resolution in the scale of the original coordinate system

Details

Default : res = 1

Value

A data frame containing the x, y, z coordinates and the number of points within each voxel of a voxel cloud.

Author(s)

Bastien Lecigne, Sylvain Delagrange and Christian Messier

Examples

#-import data
data(treecloud)
#-voxelisation
treecloud_vox <- vox(treecloud, res=0.02)
#-visualisation
require(rgl)
library(rgl)
open3d()
plot3d(treecloud_vox, size=0.1)
Index

*Topic 2d points cloud
   project, 11
*Topic 3d angle
   axis.angle, 3
*Topic 3d distance
   axis.distance, 4
   point.distance, 10
*Topic 3d objects
   obj.rec, 9
*Topic 3d point cloud
   VoxR-package, 2
*Topic 3d points clouds subtraction
   sub.obj, 15
*Topic 3d points clouds
   sub.obj, 15
*Topic 3d points cloud
   axis.angle, 3
   axis.distance, 4
   obj.rec, 9
   point.distance, 10
   project, 11
   vox, 19
*Topic LiDAR
   VoxR-package, 2
*Topic datasets
   data1, 6
   data2, 6
   data_part, 7
   treecloud, 17
   treecloud_vox, 18
*Topic package
   VoxR-package, 2
*Topic percentages
   level, 8
*Topic pixels surface
   surface, 16
*Topic pixels
   surface, 16
*Topic projection
   project, 11
*Topic quantiles
   level, 8
*Topic raster plotting
   raster.proj, 13
*Topic variable discretisation
   level, 8
*Topic voxelisation
   vox, 19

axis.angle, 2, 3, 5, 11
axis.distance, 2, 3, 4, 11
data1, 6, 6
data2, 6, 6
data_part, 7
level, 2, 8, 13, 16
obj.rec, 2, 7, 9
point.distance, 2, 3, 5, 10
project, 2, 8, 11, 13, 16
raster.proj, 2, 8, 13
sub.obj, 2, 6, 10, 15
surface, 2, 8, 13, 14, 16
treecloud, 17, 18
treecloud_vox, 18
vox, 2, 18, 19
VoxR (VoxR-package), 2
VoxR-package, 2