Package 'triplclust'

November 4, 2026

Type Package

Title Algorithm for Curve Detection in 3D Point Clouds

Version 2.0.1

Date 2026-10-18

Maintainer Christoph Dalitz <christoph.dalitz@hs-niederrhein.de>

Description The TriplClust algorithm assigns each point in a 3D or 2D point cloud a cluster label representing curves or noise. It can be used to detect curves in point clouds. TriplClust is based on hierarchical clustering of point triplets and is explained in Dalitz et al. (2019) <doi:10.5201/ipol.2019.234>.

Depends R (>= 3.5.0)

License BSD_2_clause + file LICENSE

URL https://github.com/cdalitz/triplclust/

LazyData true

Imports Rcpp (>= 1.0.0)

LinkingTo Rcpp

Encoding UTF-8

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Christoph Dalitz [aut, cre],
Jens Wilberg [aut, cph],
Lukas Aymans [aut, cph],
Daniel Müllner [aut, cph],
Vitali Krieger [aut]

Repository CRAN

Date/Publication 2026-11-04 10:10:02 UTC

Contents

attpc 2
label_colors 3

lidar 4
plot_threshold 4
save_csv 5
save_gnuplot 6
synthetic_clean 6
synthetic_noise 7
tennis 7
triplclust 8

Index **11**

`attpc` *AT-TPC Point Cloud*

Description

Particle tracks from an active target time projection chamber (AT-TPC) measurement at the AT-TPC at NSCL, Michigan State University.

Format

A numeric matrix with 121 rows and three columns: `x`, `y`, and `z`.

Details

This is an example that works with the default settings of triplclust.

Source

AT-TPC at the NSCL, Michigan State University (courtesy of Yassid Ayyad)

References

Dalitz C., Ayyad Y., Wilberg J., Aymans L., Bazin D., Mittig W.: Automatic trajectory recognition in Active Target Time Projection Chambers data by means of hierarchical clustering. *Computer Physics Communications* 235:159-168 (2019) doi:10.1016/j.cpc.2018.09.010

Examples

```
tc <- triplclust(attpc)
```

label_colors *Convert triplclust labels to colors.*

Description

For noise points (no cluster label), red is returned, for intersection points (more than one cluster label), black is returned. For all other cluster labels, a unique color is computed with modulo arithmetic.

Usage

```
label_colors(labels)
```

Arguments

labels	List containing the labels as returned by `triplclust()`. Alternatively a `triplclust` object can be provided as returned by `triplclust()`.

Value

Character vector of RGB colors with the same length as `labels`.

References

Dalitz C., Wilberg J., Aymans L. (2019) TriplClust: An Algorithm for Curve Detection in 3D Point Clouds. *Image Processing On Line* 9:26-46, doi:10.5201/ipol.2019.234

See Also

triplclust

Examples

```
tc <- triplclust(attpc)
colors <- label_colors(tc)

## Not run: # visualization with rgl
library(rgl)
plot3d(tc$points, col=colors)

# visualization with scatterplot3d
library(scatterplot3d)
scatterplot3d(tc$points, color=colors)

# visualization with plot3D
library(plot3D)
points3D(tc$points[,1], tc$points[,2], tc$points[,3],
         colvar=1:length(colors), col=colors, colkey=F)

## End(Not run)
```

lidar *LIDAR Point Cloud*

Description

Airborne LIDAR scan of a high voltage pole with about seven dots per square meter. The original data in this excerpt from "Geobasisdaten der Kommunen und des Landes NRW 2014" had 5813 points, from which we have randomly sampled 1000 points.

Format

A numeric matrix with 1000 rows and three columns: `x`, `y`, and `z`.

Details

This is an example with a lower point density along the curves, so that the triplclust parameter k should be set to k=12. Moreover, the automatic threshold detection does not work in this case and it should be manually set to t=12.

Source

Geobasisdaten der Kommunen und des Landes NRW (2014), courtesy of Katasteramt Krefeld.

References

Dalitz C., Wilberg J., Aymans L. (2019) TriplClust: An Algorithm for Curve Detection in 3D Point Clouds. *Image Processing On Line* 9:26-46, doi:10.5201/ipol.2019.234

Examples

```
tc <- triplclust(lidar, k=12, t=12)
```

plot_threshold *Plots triplclust threshold amongst the clustering distances.*

Description

The distance cdist between clusters in each clustering step is plotted as a function of the number of clusters. Additionally the maximium "expected next cdist" mean(cdits) + 2*sd(cdist) is plotted, which is used by the automatic thresold selection.

Usage

```
plot_threshold(triplclust)
```

Arguments

`triplclust`	An object of class `triplclust` as returned by `triplclust()`.

Details

The actually chosen threshold t_real is shown, too.

Value

No return value, displays a plot.

See Also

triplclust

Examples

```
tc <- triplclust(attpc)
plot_threshold(tc)
```

`save_csv` *Save triplclust result as CSV file.*

Description

The CSV is comma separated with four columns, the x, y, z coordinates, and the cluster label. For noise, the special label `-1` is used and, for points belonging to multiple clusters, all cluster labels are given as a semicolon-separated list.

Usage

```
save_csv(tc, file)
```

Arguments

`tc`	An object of class `triplclust` as returned by `triplclust()`.
`file`	File name.

Value

No return value, writes a file.

See Also

triplclust

save_gnuplot *Save triplclust result to gnuplot command file.*

Description

Noise is plotted in red, points with multiple cluster memberships are plotted as a separate overlap series. The plot can be shown from the command line with `gnuplot -persist`.

Usage

```
save_gnuplot(tc, file)
```

Arguments

`tc`	An object of class `triplclust` as returned by `triplclust()`.
`file`	File name.

Value

No return value, writes a file with gnuplot commands.

See Also

triplclust

Examples

```
tc <- triplclust(attpc)
save_gnuplot(tc, file="/tmp/bla.gnuplot")
system("gnuplot -persist /tmp/bla.gnuplot")
```

synthetic_clean *Synthetic Clean Point Cloud*

Description

Synthetically generated point cloud with four curves and no added noise.

Format

A numeric matrix with 225 rows and three columns: `x`, `y`, and `z`.

Details

In this example, triplclust merges two curves because they meet tangential.

References

Dalitz C., Wilberg J., Aymans L. (2019) TriplClust: An Algorithm for Curve Detection in 3D Point Clouds. *Image Processing On Line* 9:26-46, doi:10.5201/ipol.2019.234

Examples

```
tc <- triplclust(synthetic_clean)
```

---

synthetic_noise *Synthetic Noisy Point Cloud*

Description

Synthetically generated point cloud with four curves and added noise.

Format

A numeric matrix with 325 rows and three columns: `x`, `y`, and `z`.

Details

In this example, the noise confuses the default smoothing of triplclust and the smoothing radius r should be set to r=1.

References

Dalitz C., Wilberg J., Aymans L. (2019) TriplClust: An Algorithm for Curve Detection in 3D Point Clouds. *Image Processing On Line* 9:26-46, doi:10.5201/ipol.2019.234

Examples

```
tc <- triplclust(synthetic_noise, r=1)
```

---

tennis *Tennis Ball Point Cloud*

Description

Candidate points for tennis ball tracking extracted by by Vito Renò et al. from video recordings of a tennis match. The resulting data contained candidate points for tracking the tennis ball. From this data we have selected a contiguous sequence of 701 points.

Format

A numeric matrix with 701 rows and three columns: `x`, `y`, and `z`.

Details

This is an example for a low point density so that the triplclust parameter k should be set to k=12.

Source

courtesy of V. Renò

References

Renò V., Mosca N., Nitti M., Guaragnella C., D'Orazio T., Stella E. (2016) Real-time tracking of a tennis ball by combining 3D data and domain knowledge. *International Conference on Technology and Innovation in Sports, Health and Wellbeing (TISHW)* 2016:1-7 doi:10.1109/TISHW.2016.7847774

Examples

```
tc <- triplclust(tennis, k=12)
```

`triplclust` *Cluster 3D point cloud into curves and noise.*

Description

The TriplClust algorithm assigns each point in a 3D or 2D point cloud a cluster label representing curves or noise. It can be used to detect curves in point clouds. TriplClust is based on hierarchical clustering of point triplets and is explained in Dalitz et al. (2019).

Usage

```
triplclust(
  points,
  r = NULL,
  k = 19L,
  n = 2L,
  a = 0.03,
  s = NULL,
  t = "auto",
  dmax = NULL,
  linkage = "single",
  m = 5L,
  ordered = FALSE
)
```

Arguments

points	Numeric matrix or data.frame with the point coordinates, one point per line. If it has more than three columns, the columns named "x", "y", and "z" are used or, if there are no columns with these names, the first three are used. If it only has two columns, a third column is padded with zeros.
r	Smoothing radius. Accepts a finite numeric scalar, or a string such as "2", "2dNN", or "0.5dNN"; defaults to 2dNN.
k	Number of nearest neighbors used to generate candidate triplets.
n	Minimum number of neighbors used by triplet generation.
a	Collinearity tolerance in (0, pi).
s	Distance scale for hierarchical clustering. Accepts a finite numeric scalar, or a string such as "0.33", "0.33dNN", or "1.5dNN"; defaults to 0.33dNN.
t	Clustering threshold. Accepts a finite non-negative numeric value, or the case-insensitive strings "auto" / "automatic" to let the algorithm choose it.
dmax	Optional maximum gap for splitting clusters. Accepts a finite numeric scalar, a dNN-scaled string such as "2.5dNN" (case-insensitive suffix), or "none".
linkage	Linkage method for the hierarchical clustering step: "single", "complete", or "average".
m	Minimum cluster size retained by pruning.
ordered	Logical flag. If TRUE, treat the input as an ordered point sequence, matching the CLI -ordered option.

Value

The returned value is an object of the class "triplclust" containing the following components:

- points: Matrix of input points.
- labels: List with the assigned cluster labels encoded as one integer vector per input point. Each vector contains the IDs of the clusters containing that point. Unassigned points have an empty vector. This means that points can be assigned to a single cluster (curve points), several clusters (intersection points of curves) or no cluster (noise).
- params: Named list with the parameters of the run. A parameter given as a multiple of dNN has three entries: the plain name holds the requested value and its _dnn flag states that it is a multiple of dNN (for example r is 2 and r_dnn is TRUE for "2dNN"), whereas the _real suffix holds the absolute value that was actually used. The _real entries are resolved during the run and are only reported for debugging and reproducibility; the algorithm does not use them.

  r smoothing radius as requested

  r_dnn TRUE if r was given as a multiple of dNN

  r_real absolute smoothing radius used

  k neighbours considered for triplet generation

  n best triplets kept per midpoint

  a max angle between the triplet branches

  s distance scale as requested

`s_dnn` `TRUE` if `s` was given as a multiple of dNN

`s_real` absolute distance scale used

`t` cluster distance threshold as requested

`tauto` `TRUE` if `t` was determined automatically

`t_real` the absolute cluster distance threshold that was actually applied, i.e. the automatically determined one when `tauto` is `TRUE` and otherwise simply `t`; `NaN` if no triplets were generated, so that no threshold was applied at all

`dmax` max gap within a triplet as requested

`dmax_dnn` `TRUE` if `dmax` was given as a multiple of dNN

`is_dmax` `TRUE` if splitting of clusters at gaps is enabled

`dmax_real` absolute max gap used

`dnn_real` the absolute characteristic length that was used for scaling, `0` if no parameter depended on dNN

`linkage` linkage method

`m` minimum cluster size

Note that `k`, `n`, `a`, `linkage` and `m` are never rescaled and therefore have no `_real` counterpart.

- cdist: Numeric vector with the distance of each of the n-1 merge steps of the triplet clustering, where n is the number of generated triplets. The values are the heights of the dendrogram, in ascending order. It is empty if no triplets were generated.

References

Dalitz C., Wilberg J., Aymans L. (2019) TriplClust: An Algorithm for Curve Detection in 3D Point Clouds. *Image Processing On Line* 9:26-46, doi:10.5201/ipol.2019.234

See Also

label_colors plot_threshold save_csv save_gnuplot

Examples

```
tc <- triplclust(attpc)

# points not assigned to any cluster (noise)
noise <- tc$points[sapply(tc$labels, function(x) length(x)==0),]
# points assigned to more than one cluster (intersections)
intersections <- tc$points[sapply(tc$labels, function(x) length(x)>1),]

## Not run: # visualize result
library(rgl)
plot3d(tc$points, col=label_colors(tc))

## End(Not run)

# input values or threhsold and actual value used (t_real)
tc$params[c("t", "tauto", "t_real")]
# visualization of threshold among the clustering distances
plot_threshold(tc)
```

Index

∗ **datasets**
attpc, 2
lidar, 4
synthetic_clean, 6
synthetic_noise, 7
tennis, 7

attpc, 2

label_colors, 3
lidar, 4

plot_threshold, 4

save_csv, 5
save_gnuplot, 6
synthetic_clean, 6
synthetic_noise, 7

tennis, 7
triplclust, 8