Package: lidRalignment (via r-universe)

June 1, 2026

Type Package

Title ALS MLS and TLS forest plot alignment

Version 1.1.1

Description Alignment of forest plots point clouds (ALS, ULS, MLS, TLS), including complex cases with significant misalignments.

License GPL (>= 2)

Depends R (>= 4.0.0)

Imports lidR (>= 4.2.0), lasR (>= 0.13.7), terra, data.table, rlas, rgl, grDevices, methods, Rcpp, R6, sf

Remotes r-lidar/lidR, r-lidar/lasR

RoxygenNote 7.3.3

Encoding UTF-8

LinkingTo Rcpp, RcppEigen, RcppProgress, lidR

Collate 'cloudcompare.R' 'R6Class.R' 'RcppExports.R' 'extract_features.R' 'matrix_operations.R' 'overlap.R' 'plot.R' 'registration.R' 'zzz.R'

Suggests knitr, rmarkdown

VignetteBuilder knitr

Config/pak/sysreqs libabsl-dev cmake libfreetype6-dev libgdal-dev gdal-bin libgeos-dev libglu1-mesa-dev make texlive libpng-dev libuv1-dev libgl1-mesa-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev zlib1g-dev

Repository https://r-lidar.r-universe.dev

Date/Publication 2026-03-11 14:49:41 UTC

RemoteUrl https://github.com/r-lidar/lidRalignment

RemoteRef HEAD

RemoteSha a1c999cec147e2259fbcfb61cf26b32cc9f9de7e

Contents

AlignmentScene 2
rtm_info 5
transform_las 5

Index **6**

`AlignmentScene` *AlignmentScene R6 Class for Point Cloud Alignment*

Description

The 'AlignmentScene' R6 class provides methods for aligning two 3D point clouds plots, typically from terrestrial (MLS/TLS) or aerial (ALS/UAV) LiDAR sources. It implements a multi-step alignment process with coarse, fine, and extra-fine alignment stages.

Methods

Public methods:

- `AlignmentScene$new()`
- `AlignmentScene$print()`
- `AlignmentScene$set_radius()`
- `AlignmentScene$set_ref_is_ground_based()`
- `AlignmentScene$set_mov_is_ground_based()`
- `AlignmentScene$prepare()`
- `AlignmentScene$coarse_align()`
- `AlignmentScene$fine_align()`
- `AlignmentScene$extra_fine_align()`
- `AlignmentScene$align()`
- `AlignmentScene$plot()`
- `AlignmentScene$get_registration_matrix()`
- `AlignmentScene$clone()`

Method `new()`**:** Create a new AlignmentScene object.

Usage:

```
AlignmentScene$new(fref, fmov, radius = 20, verbose = TRUE)
```

Arguments:

`fref, fmov` character, path to the las laz file with the point cloud to align. fref is the reference point cloud, fmov is the point cloud to be aligned

`radius` clip radius. The routine does not align the full point cloud but clip before to compute the registration matrix

`verbose` logical

Returns: A new 'AlignmentScene' object.

Method `print()`**:** print

Usage:

```
AlignmentScene$print()
```

Method `set_radius()`**:** Set clip radius

Usage:

```
AlignmentScene$set_radius(radius)
```

Arguments:

`radius` numeric.

Method `set_ref_is_ground_based()`**:** Tell the pipeline if the reference point cloud is ground-based (TLS, MLS)

Usage:

```
AlignmentScene$set_ref_is_ground_based(val = TRUE)
```

Arguments:

`val` logical

Method `set_mov_is_ground_based()`**:** Tell the pipeline if the moving point cloud is ground-based (TLS, MLS)

Usage:

```
AlignmentScene$set_mov_is_ground_based(val = TRUE)
```

Arguments:

`val` logical

Method `prepare()`**:** First function to run. It reads the point cloud, extract features, and performs raw alignment

Usage:

```
AlignmentScene$prepare()
```

Method `coarse_align()`**:** Second function to run. It perform a brute force alignment

Usage:

```
AlignmentScene$coarse_align(res = 2, max_offset = 10, debug = FALSE)
```

Arguments:

`res` numeric. Subsampling resolution. Keep it as is.

`max_offset` numeric. Maximum translation possible. Increase if the point cloud are strongly missaligned on XY.

`debug` logical. Displays rendering for debugging

Method `fine_align()`**:** Third function to run. It perform an ICP alignment

Usage:

```
AlignmentScene$fine_align(overlap = "auto", ...)
```

Arguments:

`overlap` numeric trimmed ICP overlap. Can be 10, 20, 30, 40 to 100.

`...` unused

Method `extra_fine_align()`**:** Fourth function to run. It perform an ICP alignment on the trunks

Usage:

```
AlignmentScene$extra_fine_align(...)
```

Arguments:

`...` unused

Method `align()`**:** Full pipeline

Usage:

```
AlignmentScene$align(res = 2, max_offset = 8)
```

Arguments:

`res` numeric. Subsampling resolution. Keep it as is.

`max_offset` numeric. Maximum translation possible. Increase if the point cloud are strongly missaligned on XY.

Method `plot()`**:** Plot alignment

Usage:

```
AlignmentScene$plot(
  which = c("raw", "coarse", "fine", "extra"),
  compare_to = which
)
```

Arguments:

`which` string. The alignment level to plot

`compare_to` string. The alignment level to plot and to compare

Method `get_registration_matrix()`**:** Get the registration matrix. This matrix is more or less accurate as a function of the level of registration performed

Usage:

```
AlignmentScene$get_registration_matrix()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
AlignmentScene$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

rtm_info *Extract Translation and Rotation from a Transformation Matrix*

Description

Prints translation and rotation angles from a 4x4 transformation matrix.

Usage

```
rtm_info(M)
```

Arguments

M	A 4x4 transformation matrix.

Value

None (prints results).

transform_las *Apply a Transformation Matrix to a LAS Object*

Description

Applies a 4x4 transformation matrix to a LAS point cloud. The function can handle both LAS objects and LAS file paths, transforming the point cloud data accordingly.

Usage

```
transform_las(las, M, crs = NULL)
```

Arguments

las	A LAS object or a character string representing the file path to a LAS file.
M	A 4x4 transformation matrix.
crs	Optional CRS to assign to the transformed LAS. Can be the epsg code, the wkt string or a sf crs object.

Value

A transformed LAS object. If a file path is provided for las, a new LAS file is created and returned.

Index

AlignmentScene, 2

rtm_info, 5

transform_las, 5