Package: visor (via r-universe)

June 2, 2026

Title Geospatial Tools for Visibility Analysis

Version 0.1.1

Description Provides tools for visibility analysis in geospatial data. It offers functionality to perform isovist calculations, using arbitrary geometries as both viewpoints and occluders.

License Apache License (>= 2)

URL https://cityriverspaces.github.io/visor/, https://github.com/CityRiverSpaces/visor, https://doi.org/10.5281/zenodo.15133420

BugReports https://github.com/CityRiverSpaces/visor/issues

Imports sf, sfheaders

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Depends R (>= 4.2)

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

VignetteBuilder knitr

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev

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

Date/Publication 2025-12-04 21:37:17 UTC

RemoteUrl https://github.com/cityriverspaces/visor

RemoteRef HEAD

RemoteSha 7ddc0e6a5371718315617653d8e36b520a0f8f20

Contents

`create_occluder` *Create a polygon representing an occluder*

Description

Create a polygon representing an occluder

Usage

```
create_occluder(center_x, center_y, length, width)
```

Arguments

`center_x`	Center x coordinate
`center_y`	Center y coordinate
`length`	Length of the occluder
`width`	Width of the occluder

Value

object of class sfc_POLYGON

Examples

```
occluder <- create_occluder(0, 0, 10, 2)
```

`get_isovist` *Calculate isovist from one or multiple viewpoints*

Description

Isovists are estimated by shooting a set of rays from each viewpoint, and by constructing the envelope of the (partially occluded) rays.

Usage

```
get_isovist(
  viewpoints,
  occluders = NULL,
  ray_num = 40,
  ray_length = 100,
  remove_holes = TRUE
)
```

Arguments

viewpoints	object of class sf_POINT or sfc_POINT
occluders	object of class sf, sfc or sfg
ray_num	number of rays per viewpoint. The number of rays per quadrant needs to be a whole number, so ray_num will be rounded to the closest multiple of four
ray_length	length of rays
remove_holes	whether to remove holes from the overall isovist geometry

Value

object of class sfc_POLYGON or sfc_MULTIPOLYGON

Examples

```
# Define viewpoints and occluder geometries
viewpoints <- sf::st_sfc(
  sf::st_point(c(-1, 1)),
  sf::st_point(c(0, 0)),
  sf::st_point(c(1, -1))
)
occluder1 <- sf::st_polygon(list(sf::st_linestring(
  cbind(c(-1, -1, -0.9, -0.9, -1),
        c(-1, -0.9, -0.9, -1, -1))
)))
occluder2 <- sf::st_polygon(list(sf::st_linestring(
  cbind(c(0.4, 0.4, 0.6, 0.6, 0.4),
        c(0.5, 0.7, 0.7, 0.5, 0.5))
)))
occluders <- sf::st_sfc(occluder1, occluder2)

# Calculare isovist based on 40 rays (default)
get_isovist(viewpoints, occluders, ray_length = 1.5)

# Increase number of rays to get higher resolution
get_isovist(viewpoints, occluders, ray_num = 400, ray_length = 1.5)
```

`get_viewpoints` *Get viewpoints from an arbitrary geometry*

Description

Generate a discrete set of points on the given geometry. If the geometry is a (MULTI)POLYGON, points are generated on its boundary.

Usage

```
get_viewpoints(x, density = 1/50)
```

Arguments

`x`	object of class sf, sfc or sfg
`density`	number of points per distance unit

Value

object of class sfc_POINT

Examples

```
line <- sf::st_linestring(cbind(c(-1, 1), c(0, 0)))
get_viewpoints(line, density = 5)
```

Index