Package 'centerline'

September 26, 2024

Title Extract Centerline from Closed Polygons

Version 0.1

Maintainer Anatoly Tsyplenkov <atsyplenkov@fastmail.com>

Description Generates skeletons of closed 2D polygons using Voronoi diagrams. It provides methods for 'sf', 'terra', and 'geos' objects to compute polygon centerlines based on the generated skeletons. Voronoi, G. (1908) <doi:10.1515/crll.1908.134.198>.

License MIT + file LICENSE

URL https://github.com/atsyplenkov/centerline

BugReports https://github.com/atsyplenkov/centerline/issues

Suggests smoothr (>= 1.0.0), testthat (>= 3.0.0), geomtextpath (>= 0.1.0), terra (>= 1.7), igraph (>= 2.0.0), ggplot2 (>= 3.1.0),

Imports wk (>= 0.9), sf (>= 1.0), geos (>= 0.2.4), sfnetworks (>= 0.6)

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

Language en-US

NeedsCompilation no

Author Anatoly Tsyplenkov [aut, cre, cph] (<https://orcid.org/0000-0003-4144-8402>)

Repository CRAN

Date/Publication 2024-09-26 13:40:06 UTC

Contents

cnt_path *Find the shortest path between start and end points within a polygon*

Description

Find the shortest path between start and end points within a polygon

Usage

```
cnt_path(skeleton, start_point, end_point)
```

Arguments

skeleton	an output from cnt_skeleton() function
start_point	one or more starting points. It should be of the same class as the skeleton parameter
end_point	one ending point of the same class as skeleton and start_point parameters.

Details

The following function uses the sfnetworks::st_network_paths() approach to connect start_point with end_point by using the skeleton of a closed polygon as potential routes.

It is important to note that multiple starting points are permissible, but there can only be **one ending point**. Should there be two or more ending points, the algorithm will return an error.

Neither starting nor ending points are required to be located on the edges of a polygon (i.e., snapped to the boundary); they can be positioned wherever possible inside the polygon.

The algorithm identifies the closest nodes of the polygon's skeleton to the starting and ending points and then connects them using the shortest path possible along the skeleton. Therefore, if more precise placement of start and end points is necessary, consider executing the cnt_skeleton() function with the keep = 1 option. In doing so, the resulting skeleton may be more detailed, increasing the likelihood that the starting and ending points are already situated on the skeleton paths.

Value

a list of sf, sfc, SpatVector or geos_geometry class objects of a LINESTRING geometry

Examples

```
library(sf)
library(geos)
# Load Polygon and points data
polygon <-
  sf::st_read(
    system.file("extdata/example.gpkg", package = "centerline"),
    layer = "polygon",
    quiet = TRUE
  ) |>
```

```
  geos::as_geos_geometry()

points <-
  sf::st_read(
    system.file("extdata/example.gpkg", package = "centerline"),
    layer = "polygon_points",
    quiet = TRUE
  ) |>
  geos::as_geos_geometry()

# Find polygon's skeleton
pol_skeleton <- cnt_skeleton(polygon)

# Connect points
pol_path <-
  cnt_path(
    skeleton = pol_skeleton,
    start_point = points[2],
    end_point = points[1]
  )

# Plot
plot(polygon)
plot(pol_skeleton, col = "blue", add = TRUE)
plot(points[1:2], col = "red", add = TRUE)
plot(pol_path, lwd = 3, add = TRUE)
```

`cnt_path_guess` *Guess polygon's centerline*

Description

This function, as follows from the title, tries to guess the polygon centerline by connecting the most distant points from each other. First, it finds the point most distant from the polygon's centroid, then it searches for a second point, which is most distant from the first. The line connecting these two points will be the desired centerline.

Usage

```
cnt_path_guess(input, skeleton = NULL, ...)
```

Arguments

`input`	`sf`, `sfc` or `SpatVector` polygons object
`skeleton`	`NULL` (default) or `cnt_skeleton()` output. If `NULL` then polygon's skeleton would be estimated in the background using specified parameters (see inherit params below).
`...`	Arguments passed on to `cnt_skeleton`
	`keep` numeric, proportion of points to retain (0.05-Inf; default 0.5). See Details.

Value

An sf, sfc or SpatVector class object of a LINESTRING geometry

Examples

```
library(sf)
library(geos)
lake <-
  sf::st_read(
    system.file("extdata/example.gpkg", package = "centerline"),
    layer = "lake",
    quiet = TRUE
  ) |>
  geos::as_geos_geometry()
# Find lake's centerline
lake_centerline <- cnt_path_guess(input = lake, keep = 1)
# Plot
plot(lake)
plot(lake_centerline, col = "firebrick", lwd = 2, add = TRUE)
```

---

cnt_skeleton *Create a skeleton of a closed polygon object*

Description

This function generates skeletons (centerlines) of closed polygon objects by applying Voronoi diagrams (Voronoi, 1908). A Voronoi diagram partitions space into regions based on the distance to the polygon's vertices. The edges of these cells form a network of lines (skeletons) that represent the structure of the polygon while preserving its overall shape.

Usage

```
cnt_skeleton(input, keep = 0.5)
```

Arguments

input	sf, sfc, SpatVector, or geos_geometry polygons object
keep	numeric, proportion of points to retain (0.05-Inf; default 0.5). See Details.

Details

- If keep equals 1, no transformation will occur. The function will use the original geometry to find the skeleton.
- If the keep parameter is below 1, then the geos::geos_simplify() function will be used. So the original input geometry would be simplified, and the resulting skeleton will be cleaner but maybe more edgy. The current realisation of simplification is similar (*but not identical*) to rmapshaper::ms_simplify() one with Douglas-Peuker algorithm. However, due to

geos superpower, it performs several times faster. If you find that the built-in simplification algorithm performs poorly, try `rmapshaper::ms_simplify()` first and then find the polygon skeleton with keep = 1, i.e. `cnt_skeleton(rmapshaper::ms_simplify(polygon_sf), keep = 1)`

- If the keep is above 1, then the densification algorithm is applied using the `geos::geos_densify()` function. This may produce a very large object if keep is set more than 2. However, the resulting skeleton would potentially be more accurate.

Value

a sf, sfc, SpatVector or geos_geometry class object of a MULTILINESTRING geometry

References

Voronoi, G. (1908). Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Journal für die reine und angewandte Mathematik, 134, 198-287. doi:10.1515/crll.1908.134.198

Examples

```
library(sf)

polygon <-
  sf::st_read(system.file("extdata/example.gpkg", package = "centerline"),
    layer = "polygon",
    quiet = TRUE
  )

plot(polygon)

pol_skeleton <- cnt_skeleton(polygon)

plot(pol_skeleton)
```

Index