Package 'ForestTools'

October 2, 2023

Type Package

Title Tools for Analyzing Remote Sensing Forest Data

Version 1.0.1

Date 2023-09-27

Description Tools for analyzing remote sensing forest data, including functions for detecting tree-tops from canopy models, outlining tree crowns, and calculating textural metrics.

Depends R (>= 4.2)

License GPL-3

Encoding UTF-8

LazyData true

LinkingTo Rcpp

Imports terra, sf, Matrix, plyr, imager, Rcpp

Suggests testthat (>= 3.0.0), knitr, rmarkdown

RoxygenNote 7.2.3

URL https://github.com/andrew-plowright/ForestTools

BugReports https://github.com/andrew-plowright/ForestTools/issues

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation yes

Author Andrew Plowright [aut, cre],
Jean-Romain Roussel [ctb] (Contributed to segment-based GLCM segmentation)

Maintainer Andrew Plowright <andrew.plowright@alumni.ubc.ca>

Repository CRAN

Date/Publication 2023-10-02 17:00:02 UTC

R topics documented:

`glcm` *Grey-Level Co-Occurrence Matrix*

Description

Generate textural metrics using Grey-Level Co-Occurrence Matrices (GLCM). Can be applied to an entire or image or, if a coterminous raster of segments is provided, GLCM can be calculated for each segment.

Usage

```
glcm(image, segs = NULL, n_grey = 32, angle = 0)
```

Arguments

`image`	SpatRaster. A single-band raster layer from which texture is measured
`segs`	SpatRaster. A segmented raster. Cell values should be equal to segment numbers. If `segs` are not provided, GLCM will be calculated for the entire image.
`n_grey`	integer. Number of grey levels into which the image will be discretized
`angle`	integer. Angle at which GLCM will be calculated. Valid inputs are 0, 45, 90, or 135

Details

The underlying C++ code for computing GLCMs and their statistics was originally written by Joel Carlson for the defunct [radiomics](https://github.com/cran/radiomics) library. It has been reused here with permission from the author.

Value

data.frame

References

Parmar, C., Velazquez, E.R., Leijenaar, R., Jermoumi, M., Carvalho, S., Mak, R.H., Mitra, S., Shankar, B.U., Kikinis, R., Haibe-Kains, B. and Lambin, P. (2014). *Robust radiomics feature quantification using semiautomatic volumetric segmentation. PloS one, 9*(7)

See Also

mcws

Examples

```
## Not run:
library(terra)
library(ForestTools)

chm <- rast(kootenayCHM)
image <- rast(kootenayOrtho)[[1]]

# Generate raster segments
segs <- mcws(kootenayTrees, chm, minHeight = 0.2, format = "raster")

# Get textural metrics for ortho's red band
tex <- glcm(image, segs)

## End(Not run)
```

---

glcm0 *Create a 0 degree GLCM*

Description

Used internally by glcm()

Usage

```
glcm0(x, n_grey, d)
```

Arguments

x	A Numeric matrix, integer values only
n_grey	Number of grey levels
d	distance from reference pixel to neighbour pixel

glcm135 *Create a 135 degree GLCM*

Description

Used internally by glcm()

Usage

```
glcm135(x, n_grey, d)
```

Arguments

x	A Numeric matrix, integer values only
n_grey	Number of grey levels
d	distance from reference pixel to neighbour pixel

glcm45 *Create a 45 degree GLCM*

Description

Used internally by glcm()

Usage

```
glcm45(x, n_grey, d)
```

Arguments

x	A Numeric matrix, integer values only
n_grey	Number of grey levels
d	distance from reference pixel to neighbour pixel

glcm90 *Create a 90 degree GLCM*

Description

Used internally by glcm()

Usage

```
glcm90(x, n_grey, d)
```

Arguments

x	A Numeric matrix, integer values only
n_grey	Number of grey levels
d	distance from reference pixel to neighbour pixel

glcm_features *GLCM Features*

Description

GLCM Features

Usage

```
glcm_mean(glcm)

glcm_variance(glcm)

glcm_autoCorrelation(glcm)

glcm_cProminence(glcm)

glcm_cShade(glcm)

glcm_cTendency(glcm)

glcm_contrast(glcm)

glcm_correlation(glcm)

glcm_differenceEntropy(glcm, base = 2)

glcm_dissimilarity(glcm)
```

```
glcm_energy(glcm)

glcm_entropy(glcm, base = 2)

glcm_homogeneity1(glcm)

glcm_homogeneity2(glcm)

glcm_IDMN(glcm)

glcm_IDN(glcm)

glcm_inverseVariance(glcm)

glcm_maxProb(glcm)

glcm_sumAverage(glcm)

glcm_sumEntropy(glcm, base = 2)

glcm_sumVariance(glcm)
```

Arguments

glcm	A matrix of class "glcm" produced by glcm.
base	Base of the logarithm in differenceEntropy.

Functions

- glcm_mean(): Mean
- glcm_variance(): Variance
- glcm_autoCorrelation(): Autocorrelation
- glcm_cProminence(): Cluster Prominence
- glcm_cShade(): Cluster Shade
- glcm_cTendency(): Cluster Tendency
- glcm_contrast(): Contrast
- glcm_correlation(): Correlation
- glcm_differenceEntropy(): Difference Entropy
- glcm_dissimilarity(): Dissimilarity
- glcm_energy(): Energy
- glcm_entropy(): Entropy
- glcm_homogeneity1(): Homogeneity
- glcm_homogeneity2(): Homogeneity 2

- `glcm_IDMN()`: Inverse Difference Moment (Normalized)
- `glcm_IDN()`: Inverse Difference (Normalized)
- `glcm_inverseVariance()`: Inverse Variance
- `glcm_maxProb()`: Maximum Probability
- `glcm_sumAverage()`: Sum Average
- `glcm_sumEntropy()`: Sum Entropy
- `glcm_sumVariance()`: Sum Variance

References

https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0102107

`kootenayBlocks` *Kootenay forest - Cut blocks*

Description

Boundaries of cut blocks within a 1.5 hectare section of forest in the Kootenay mountains, in British Columbia, Canada. Each block contains trees of different levels of maturity. Overlaps with kootenayTrees, kootenayCrowns, kootenayOrtho and kootenayCHM.

Usage

```
kootenayBlocks
```

Format

Simple polygon feature collection with the following attributes:

BlockID numerical identifier for each block

Shape_Leng length of polygon on meters

Shape_Area area of polygon in square meters

See Also

kootenayTrees kootenayCHM kootenayCrowns kootenayOrtho

kootenayCHM *Kootenay forest - Canopy height model*

Description

A canopy height model of a 1.5 hectare section of forest in the Kootenay mountains, in British Columbia, Canada.

Usage

```
kootenayCHM
```

Format

PackedSpatRaster object

Cell values are equal to canopy height above ground (in meters)

Source

Data acquired from a photogrammetric drone survey performed by Spire Aerobotics on June 16th, 2016.

See Also

kootenayTrees kootenayBlocks kootenayCrowns kootenayOrtho

kootenayCrowns *Kootenay forest - Tree crowns*

Description

Outlines of tree crowns corresponding to the kootenayTrees treetops. Generated using mcws.

Usage

```
kootenayCrowns
```

Format

Simple polygon feature collection with the following attributes:

height height of the tree's apex, in meters above ground. Inherited from kootenayTrees.

winRadius radius of the moving window at the treetop's location. Inherited from kootenayTrees.

crownArea area of crown outline in square meters

See Also

kootenayTrees kootenayCHM kootenayBlocks kootenayOrtho

kootenayOrtho *Kootenay forest - Orthomosaic*

Description

An orthomosaic of a 1.5 hectare section of forest in the Kootenay mountains, in British Columbia, Canada.

Usage

```
kootenayOrtho
```

Format

PackedSpatRaster object

Cell values are equal to canopy height above ground (in meters)

Source

Data acquired from a photogrammetric drone survey performed by Spire Aerobotics on June 16th, 2016.

See Also

kootenayTrees kootenayBlocks kootenayCrowns kootenayCHM

kootenayTrees *Kootenay forest - Dominant trees over 2 m*

Description

Dominant trees from a 1.5 hectare section of forest in the Kootenay mountains, in British Columbia, Canada. Trees were detected by applying the vwf function to the kootenayCHM raster dataset. Only trees over 2 m above ground were detected.

Usage

```
kootenayTrees
```

Format

Simple point feature collection with the following attributes:

height height of the tree's apex, in meters above ground

winRadius radius of the moving window (see vwf) at the treetop's location

See Also

kootenayCHM kootenayBlocks kootenayCrowns kootenayOrtho

`mcws` *Marker-Controlled Watershed Segmentation*

Description

This function implements the watershed function to segment (i.e.: outline) crowns from a CHM (canopy height model). Segmentation is guided by the point locations of treetops, typically detected using the vwf function. See Meyer & Beucher (1990) for details on watershed segmentation.

Usage

```
mcws(
  treetops,
  CHM,
  minHeight = 0,
  format = "raster",
  OSGeoPath = NULL,
  IDfield = "treeID"
)
```

Arguments

`treetops`	sf. The point locations of treetops in `sf` format.
`CHM`	SpatRaster. Canopy height model in `SpatRaster` format. This should be the same CHM that was used to the detect the `treetops`.
`minHeight`	numeric. The minimum height value for a `CHM` pixel to be considered as part of a crown segment. All `CHM` pixels beneath this value will be masked out. Note that this value should be lower than the minimum height of `treetops`.
`format`	string. Format of the function's output. Can be set to either 'raster' or 'polygons'.
`OSGeoPath`	character. Obsolete. Will be removed next version
`IDfield`	character. Name of the field for storing the unique tree identifier

Details

Crown segments are returned as either a `SpatRaster` or a `sf` (Simple Feature) class object, as defined using the `format` argument. For many analytic purposes, it is preferable to have crown outlines as polygons. However, polygonal crown maps take up significantly more disk space, and take longer to process. It is advisable to run this function using a raster output first to review results and adjust parameters.

NOTE: when setting `format` to 'polygons', orphaned segments (i.e.: outlines without an associated treetop) will be removed. This will NOT occur using 'raster' format. This issue will be resolved eventually but requires the watershed function to be rewritten.

Value

Depending on the setting for `format`, this function will return a map of outlined crowns as either a `SpatRaster` class object, in which distinct crowns are given a unique cell value, or a `sf` class object, in which each crown is represented by a polygon.

References

Meyer, F., & Beucher, S. (1990). Morphological segmentation. *Journal of visual communication and image representation, 1*(1), 21-46.

See Also

`vwf`

Examples

```
## Not run:
library(terra)
library(ForestTools)

chm <- rast(kootenayCHM)

# Use variable window filter to detect treetops
ttops <- vwf(chm, winFun = function(x){x * 0.06 + 0.5}, minHeight = 2)

# Segment tree crowns
segs <- mcws(ttops, chm, minHeight = 1)

## End(Not run)
```

---

quesnelBlocks *Quesnel forest - Cut blocks*

Description

Boundaries of cut blocks within a 125 hectare section of forest in the Quesnel Timber Supply Area, in British Columbia, Canada. Each block contains trees of different levels of maturity. Overlaps with quesnelTrees and quesnelCHM.

Usage

```
quesnelBlocks
```

Format

Simple polygon feature collection with the following attributes:

BlockID numerical identifier for each block

Shape_Leng length of polygon on meters

Shape_Area area of polygon in square meters

See Also

quesnelTrees quesnelCHM

`quesnelCHM` *Quesnel forest - Canopy height model*

Description

A canopy height model of a 125 hectare section of forest in the Quesnel Timber Supply Area, in British Columbia, Canada.

Usage

```
quesnelCHM
```

Format

PackedSpatRaster object

Cell values are equal to canopy height above ground (in meters)

Source

Data acquired from a photogrammetric drone survey performed by Spire Aerobotics on September 15th, 2016.

See Also

quesnelTrees quesnelBlocks

quesnelTrees *Quesnel forest - Dominant trees over 2 m*

Description

Dominant trees from a 125 hectare section of forest in the Quesnel Timber Supply Area, in British Columbia, Canada. Trees were detected by applying the vwf function to the quesnelCHM raster dataset. Only trees over 2 m above ground were detected.

Usage

```
quesnelTrees
```

Format

Simple point feature collection with the following attributes:

height height of the tree's apex, in meters above ground

winRadius radius of the moving window (see vwf) at the treetop's location

See Also

quesnelCHM quesnelBlocks

vwf *Variable Window Filter*

Description

Implements the variable window filter algorithm (Popescu & Wynne, 2004) for detecting treetops from a canopy height model.

Usage

```
vwf(
  CHM,
  winFun,
  minHeight = NULL,
  warnings = TRUE,
  minWinNeib = "queen",
  IDfield = "treeID"
)
```

Arguments

CHM	SpatRaster. Canopy height model in SpatRaster format.
winFun	function. The function that determines the size of the window at any given location on the canopy. It should take the value of a given CHM pixel as its only argument, and return the desired *radius* of the circular search window when centered on that pixel. Size of the window is in map units.
minHeight	numeric. The minimum height value for a CHM pixel to be considered as a potential treetop. All CHM pixels beneath this value will be masked out.
warnings	logical. If set to FALSE, this function will not emit warnings related to inputs.
minWinNeib	character. Define whether the smallest possible search window (3x3) should use a queen or a rook neighborhood.
IDfield	character. Name of field for unique tree identifier

Details

This function uses the resolution of the raster to figure out how many cells the window needs to cover. This means that the raster value (representing height above ground) and the map unit (represented by the raster's resolution), need to be in the _same unit_. This can cause issues if the raster is in lat/lon, whereby its resolution is in decimal degrees.

Value

Simple feature collection of POINT type. The point locations of detected treetops. The object contains two fields in its data table: *height* is the height of the tree, as extracted from the CHM, and *winRadius* is the radius of the search window when the treetop was detected. Note that *winRadius* does not necessarily correspond to the radius of the tree's crown.

References

Popescu, S. C., & Wynne, R. H. (2004). Seeing the trees in the forest. *Photogrammetric Engineering & Remote Sensing, 70*(5), 589-604.

See Also

mcws

Examples

```
## Not run:
library(terra)
library(ForestTools)

chm <- rast(kootenayCHM)

# Set function for determining variable window radius
winFunction <- function(x){x * 0.06 + 0.5}

# Set minimum tree height (treetops below this height will not be detected)
```

```
minHgt <- 2

# Detect treetops in demo canopy height model
ttops <- vwf(chm, winFunction, minHgt)

## End(Not run)
```

Index