Package: MtreeRing (via r-universe)

July 17, 2024

Type Package

Title A Shiny Application for Automatic Measurements of Tree-Ring Widths on Digital Images

Version 1.4.5

Author Jingning Shi [aut, cre], Wei Xiang [aut]

Maintainer Jingning Shi <snow940220@bjfu.edu.cn>

Suggests testthat, knitr, shinytest, mockery, spelling

Imports magrittr (>= 1.5), png, jpeg, tiff, bmp, magick, imager, dplR, spatstat.geom, measuRing, shiny, dplyr, shinydashboard, shinyWidgets

Description Use morphological image processing and edge detection algorithms to automatically measure tree ring widths on digital images. Users can also manually mark tree rings on species with complex anatomical structures. The arcs of inner-rings and angles of successive inclined ring boundaries are used to correct ring-width series. The package provides a Shiny-based application, allowing R beginners to easily analyze tree ring images and export ring-width series in standard file formats.

License GPL-3

NeedsCompilation no

Encoding UTF-8

URL https://docs.ropensci.org/MtreeRing, https://github.com/ropensci/MtreeRing

BugReports https://github.com/ropensci/MtreeRing/issues

RoxygenNote 6.1.1

Language en-US

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

RemoteUrl https://github.com/ropensci/MtreeRing

RemoteRef master

RemoteSha c3f44754fffe2e696522a40a2b71a0013cb41f1d

Contents

`MtreeRing-package` *A Shiny Application for Automatic Measurements of Tree-Ring Widths on Digital Images*

Description

Use morphological image processing and edge detection algorithms to automatically measure tree ring widths on digital images. Users can also manually mark tree rings on species with complex anatomical structures. The arcs of inner-rings and angles of successive inclined ring boundaries are used to correct ring-width series. The package provides a Shiny-based application, allowing R beginners to easily analyze tree ring images and export ring-width series in standard file formats.

Details

Package:	MtreeRing
Type:	Package
License:	GPL-3
Maintainer:	Jingning Shi <snow940220@bjfu.edu.cn>
LazyData:	TRUE

Author(s)

Jingning Shi, Wei Xiang

pith_measure *Calibrate ring-width series*

Description

This function can calibrate the ring-width series using arcs of inner rings.

Usage

```
pith_measure(ring.data, inner.arc = TRUE, last.yr = NULL,
  color = "black", border.type = 16, label.cex = 1.5)
```

Arguments

ring.data	A magick image object produced by ring_read.
inner.arc	A logical value indicating whether to calibrate the ring-width series using the arcs of inner rings. See details below.
last.yr	NULL or an integer giving the year of formation of the left-most ring. If NULL, border numbers (starting from 1) are used instead of years.
color	Color for labels.
border.type	Symbol for ring borders. See pch in points for possible values and shapes.
label.cex	The magnification to be used for years or border numbers.

Details

This function allows the user to create a path, and manually mark ring borders by clicking on the graphical window.

An example demonstrated with pictures can be found in the package vignette. Type vignette('pith-MtreeRing') to see this example.

- If inner.arc = TRUE, the ring-width series is calibrated using arcs of inner rings (Duncan, 1989).

  Step1. You can click the left mouse button to add a horizontal path. The path should traverse an appropriate arc (read the reference below for more details).

  Step2. You can add three points to the selected arc by left-clicking. The first point should be placed on the left endpoint of the arc, and the second point is placed on the right endpoint.

  After adding these two points, a vertical dashed line will be plotted automatically according to the (x,y) positions of endpoints you just added. The third points should be placed on the intersection of the vertical dashed line and the selected arc.

  Step3. you are prompted to mark tree rings along the path by left-clicking on the image. Every click draws a point. Note that the left endpoint of the arc will be considered as the last ring border without the need to mark it.

  After marking tree rings, the identification process does not automatically stop by itself. On the Windows platform, the identification process can be terminated by clicking the second

button and selecting **Stop** from the menu. On the MacOS system, you can press the **Escape** key to terminate this process.

The ring-width series are corrected using formulas proposed by Duncan (1989).

- If `inner.arc = FALSE`, the user can create a path which matches the direction of wood growth.

  Step1. You can add two points by left-clicking on the image. Every click draws a point. A path passing through these two points will be plotted. The path should follow the rays from bark to pith.

  Step2. You can mark tree rings along the path by left-clicking on the image. The termination of identification process is similar.

Value

A data frame of the calibrated ring-width series. The measurements units are millimeters (mm)

Author(s)

Jingning Shi

References

Duncan R. (1989) An evaluation of errors in tree age estimates based on increment cores in Kahikatea (Dacrycarpus dacrydiodes). *New Zealand Natural Sciences* **16(4)**, 1-37.

Examples

```
img.path <- system.file("missing_pith.png", package = "MtreeRing")

## Read the image:
t1 <- ring_read(img = img.path, dpi = 1200, plot = FALSE)

## Use the arcs of inner rings to calibrate ring-width series:
t2 <- pith_measure(t1, inner.arc = TRUE, last.yr = 2016)

## Try another method to measure ring widths:
t3 <- pith_measure(t1, inner.arc = FALSE, last.yr = 2016)
```

---

ring_app_launch *Run Shiny-based Application*

Description

Run a Shiny-based application within the system's default web browser.

Usage

```
ring_app_launch(launch.browser = TRUE)
```

Arguments

launch.browser	A logical value. If FALSE, a built-in browser will be launched automatically after the app is started. If TRUE, the system's default web browser is used instead. This argument only works for RStudio. See details below.

Details

launch.browser = FALSE is not recommended, as the file renaming does not work on the RStudio built-in browser when saving the data.

A workflow for the Shiny app can be found here: https://ropensci.github.io/MtreeRing/articles/app-MtreeRing.html. Most steps are demonstrated with a gif to make the workflow more understandable.

To stop the app, go to the R console and press the Escape key. You can also click the stop sign icon in the upper right corner of the RStudio console.

Author(s)

Jingning Shi, Wei Xiang

---

ring_calculate *Generate a ring-width series*

Description

This function can calculate ring widths according to coordinates of detected ring borders.

Usage

```
ring_calculate(ring.data, seriesID)
```

Arguments

ring.data	A matrix or array produced by ring_detect or ring_modify.
seriesID	A character string specifying the column name of the ring-width series.

Value

A data frame. The series ID is the column name and years are row names. The measurements units are millimeters (mm).

Author(s)

Jingning Shi

Examples

```
img.path <- system.file("001.png", package = "MtreeRing")

## Read a tree ring image:
t1 <- ring_read(img = img.path, dpi = 1200)

## Split a long core sample into 3 pieces to
## get better display performance and use the
## watershed algorithm to detect ring borders:
t2 <- ring_detect(ring.data = t1, seg = 3, method = 'watershed')

## Calculate ring widths from the attribute list of t2:
rw.df <- ring_calculate(ring.data = t2, seriesID = "940220")
```

`ring_detect` *Automatic detection of tree-ring borders*

Description

This function is used to automatically detect tree ring borders along the user-defined path.

Usage

```
ring_detect(ring.data, seg = 1, auto.path = TRUE, manual = FALSE,
  method = "canny", incline = FALSE, sample.yr = NULL,
  watershed.threshold = "auto", watershed.adjust = 0.8,
  struc.ele1 = NULL, struc.ele2 = NULL, marker.correction = FALSE,
  default.canny = TRUE, canny.t1, canny.t2, canny.smoothing = 2,
  canny.adjust = 1.4, path.dis = 1, origin = 0,
  border.color = "black", border.type = 16, label.color = "black",
  label.cex = 1.2)
```

Arguments

`ring.data`	A magick image object produced by `ring_read`.
`seg`	An integer specifying the number of image segments.
`auto.path`	A logical value. If `TRUE`, a path is automatically created at the center of the image. If `FALSE`, the function allows the user to create a sub-image and a path by interactive clickings. See details below.
`manual`	A logical value indicating whether to skip the automatic detection. If `TRUE`, ring borders are visually identified after creating the path. See `ring_modify` to learn how to mark tree rings by clicking on the image.
`method`	A character string specifying how ring borders are detected. It requires one of the following characters: `"watershed"`, `"canny"`, or `"lineardetect"`. See details below.

`incline` A logical value indicating whether to correct ring widths. If `TRUE`, two horizontal paths are added to the image.

`sample.yr` `NULL` or an integer giving the year of formation of the left-most ring. If `NULL`, use the current year.

`watershed.threshold`
The threshold used for producing the marker image, either a numeric from 0 to 1, or the character "auto" (using the Otsu algorithm), or a character of the form "XX%" (e.g., "58%").

`watershed.adjust`
A numeric used to adjust the Otsu threshold. The default is 1 which means that the threshold will not be adjusted. The sizes of early-wood regions in the marker image will reduce along with the decrease of `watershed.adjust`.

`struc.ele1` `NULL` or a vector of length two specifying the width and height of the first structuring element. If `NULL`, the size of the structuring element is determined by the argument `dpi`.

`struc.ele2` `NULL` or a vector of length two specifying the width and height of the second structuring element. If `NULL`, the size of the structuring element is determined by the argument `dpi`.

`marker.correction`
A logical value indicating whether to relabel early-wood regions by comparing the values of their left-side neighbours.

`default.canny` A logical value. If `TRUE`, upper and lower Canny thresholds are determined automatically.

`canny.t1` A numeric giving the threshold for weak edges.

`canny.t2` A numeric giving the threshold for strong edges.

`canny.smoothing`
An integer specifying the degree of smoothing.

`canny.adjust` A numeric used as a sensitivity control factor for the Canny edge detector. The default is 1 which means that the sensitivity will not be adjusted. The number of detected borders will reduce along with the increase of this value.

`path.dis` A numeric specifying the perpendicular distance between two paths when the argument `incline = TRUE`. The unit is in mm.

`origin` A numeric specifying the origin in smoothed gray to find ring borders. See `ringBorders` from the package `measuRing`.

`border.color` Color for ring borders.

`border.type` Symbol for ring borders. See `pch` in `points` for possible values and their shapes.

`label.color` Color for years and border numbers.

`label.cex` The magnification to be used for years and border numbers.

Details

If `auto.path = FALSE`, the user can create a rectangular sub-image and a horizontal path by interactively clicking on the tree ring image. The automatic detection will be performed within this rectangular sub-image. To create a sub-image and a path, follow these steps.

- Step 1. Select the left and right edges of the rectangle

  The user can point the mouse at any desired locations and click the left mouse button to add each edge.

- Step 2. Select the top and bottom edges of the rectangle

  The user can point the mouse at any desired locations and click the left mouse button to add each edge. The width of the rectangle is defined as the distance between the top and bottom edges, and should not be unnecessarily large to reduce time consumption and memory usage. Creating a long and narrow rectangle if possible.

- Step 3. Create a path

  After creating the rectangular sub-image, the user can add a horizontal path by left-clicking on the sub-image (generally at the center of the sub-image, try to choose a clean defect-free area). Ring borders and other markers are plotted along this path. If `incline = TRUE`, two paths are added simultaneously.

After creating the sub-image and the path, this function will open several graphics windows and plot detected ring borders on image segments. The number of image segments is controlled by the argument `seg`.

Argument `method` determines how ring borders are identified.

- If `method = "watershed"`, this function uses the watershed algorithm to obtain ring borders (Soille and Misson, 2001).
- If `method = "canny"`, this function uses the Canny algorithm to detect borders.
- If `method = "lineardetect"`, a linear detection algorithm from the package `measuRing` is used to identify ring borders (Lara et al., 2015). Note that `incline = TRUE` is not supported in this mode, and path will be automatically created at the center of the image.

If the argument `method = "watershed"` or `"canny"`, the original image is processed by morphological openings and closings using rectangular structuring elements of increasing size before detecting borders. The first small structuring element is used to remove smaller dark spots in early wood regions, and the second large structuring element is used to remove light strips in late wood regions. More details about morphological processing can be found at Soille and Misson (2001).

Value

A matrix (grayscale image) or array (color image) representing the tree-ring image.

Note

This function uses `locator` to record mouse positions so it only works on "X11", "windows" and "quartz" devices.

Author(s)

Jingning Shi

References

Soille, P., Misson, L. (2001) Tree ring area measurements using morphological image analysis. *Canadian Journal of Forest Research* **31**, 1074-1083. doi: 10.1139/cjfr-31-6-1074

Lara, W., Bravo, F., Sierra, C.A. (2015) measuRing: An R package to measure tree-ring widths from scanned images. *Dendrochronologia* **34**, 43-50. doi: 10.1016/j.dendro.2015.04.002

Examples

```
img.path <- system.file("001.png", package = "MtreeRing")

## Read a tree ring image:
t1 <- ring_read(img = img.path, dpi = 1200, plot = FALSE)

## Split a long core sample into 3 pieces to
## get better display performance and use the
## watershed algorithm to detect ring borders:
t2 <- ring_detect(t1, seg = 3, method = 'watershed', border.color = 'green')
```

---

ring_modify *Edit ring borders visually*

Description

This function can remove existing ring borders or add new borders.

Usage

```
ring_modify(ring.data, del = NULL, del.u = NULL, del.l = NULL,
  add = FALSE)
```

Arguments

ring.data	A matrix or array produced by ring_detect.
del	A numeric vector giving the border numbers to be removed.
del.u	A numeric vector giving the border numbers to be removed on the upper path.
del.l	A numeric vector giving the border numbers to be removed on the lower path.
add	A logical value indicating whether to add new ring borders.

Details

This function is used to remove existing ring borders, or to add new borders by interactively clicking on the image segments.

If the user creates one path (incline = FALSE), the argument del is used to remove ring borders. If the user creates two paths (incline = TRUE), arguments del.u and del.l are used to remove ring borders.

If add = TRUE, graphics windows opened by ring_detect will be activated sequentially. When a graphics window is activated, the user can add new borders by left-clicking the mouse along the path. Every click draws a point representing the ring border. Type vignette('detection-MtreeRing') to see an example of adding ring borders.

The identification process does not automatically stop by itself.

- On the Windows system, the identification process can be terminated by pressing the right mouse button and selecting **Stop** from the menu.
- On the MacOS system, for a X11 device the identification process is terminated by pressing any mouse button other than the first, and for a quartz device this process is terminated by pressing the **ESC** key.

Once the user terminates the identification process, the current graphics window will be closed automatically, and the graphics window of the following segment is activated. When all graphics windows are closed, ring_modify will re-open graphics windows and plot new borders.

This function can perform both deletion and addition in one call. The removal of ring borders takes precedence over addition.

Value

A matrix (grayscale image) or array (color image) representing the tree ring image.

Author(s)

Jingning Shi

Examples

```
img.path <- system.file("001.png", package = "MtreeRing")

## Read a tree ring image:
t1 <- ring_read(img = img.path, dpi = 1200)

## Split a long core sample into 3 pieces to
## get better display performance and use the
## watershed algorithm to detect ring borders:
t2 <- ring_detect(ring.data = t1, seg = 3, method = 'watershed')

## Do not modify t2, but create a new array object t3.
## Remove some borders without adding new borders:
t3 <- ring_modify(ring.data = t2, del = c(1, 3, 5, 19:21), add = FALSE)
```

`ring_read` *Read and plot a tree-ring image file*

Description

This function can read an image file from the hard disk and plot it in a newly-opened graphics device.

Usage

```
ring_read(img, dpi = NULL, RGB = c(0.299, 0.587, 0.114),
  plot = FALSE, rotate = 0, magick = TRUE)
```

Arguments

`img`	A character string indicating the path of the image file. Supported formats include png, tiff, jpg and bmp.
`dpi`	An integer specifying the dpi of the image file. A minimum of 300 dpi is required when running automatic detection.
`RGB`	A numeric vector of length 3 giving the weight of RGB channels.
`plot`	A logical value indicating whether to plot the tree ring image when reading it. If `FALSE`, the image is not plotted until function `ring_detect` or `pith_measure` is called.
`rotate`	An integer specifying how many degrees to rotate (clockwise). It requires one of the following values: `0`, `90`, `180` or `270`.
`magick`	A logical value. If `TRUE`, `magick` is used to read the tree ring image. If `FALSE`, packages `png`, `jpg` and `tiff` are used instead. See details below.

Details

Proper image preparation has a great influence on the measurement of ring widths. A tree-ring image should not contain irrelevant or redundant features, such as wooden mounts where cores are glued. The larger the file size of an image, the slower the image processing operation will be.

Pith side of a wood sample should be placed on the **right side** of a graphics window. Use `rotate` to change its position.

It is highly recommended to use the default value `magick = TRUE`, because `magick` can significantly reduce the memory usage when reading a large file. If image data is stored in a non-standard format, image reading may fail. In that case you can set `magick = FALSE` to avoid the use of `magick`.

Value

A magick image object containing the image data.

Author(s)

Jingning Shi

Examples

```
img.path <- system.file("001.png", package = "MtreeRing")

## Read and plot the image:
t1 <- ring_read(img = img.path, dpi = 1200, plot = TRUE)
```

Index