Package 'x3ptools'

January 30, 2024

Type Package

Title Tools for Working with 3D Surface Measurements

Version 0.0.4

Date 2024-01-21

Maintainer Heike Hofmann <hofmann@iastate.edu>

Description The x3p file format is specified in ISO standard 5436:2000 to describe 3d surface measurements. 'x3ptools' allows reading, writing and basic modifications to the 3D surface measurements.

Depends R (>= 4.0)

Imports MASS (>= 7.3), digest (>= 0.6), xml2 (>= 1.3.5), rgl (>= 1.2.0), zoo (>= 1.8), png (>= 0.1-7), readr (>= 2.1.0), dplyr (>= 1.1.0), pracma (>= 2.4.0), assertthat (>= 0.2.1), purrr (>= 1.0.0), yaml (>= 2.3.7), scales (>= 1.2.1), tidyr (>= 1.3.0), imager (>= 0.45.2), magrittr (>= 2.0.3), grDevices

Suggests knitr, rmarkdown, patchwork, testthat (>= 3.0.4), covr, here, magick (>= 2.0)

License MIT + file LICENSE

RoxygenNote 7.2.3

URL https://github.com/heike/x3ptools, https://heike.github.io/x3ptools/

BugReports https://github.com/heike/x3ptools/issues

Encoding UTF-8

LazyData true

NeedsCompilation no

Author Heike Hofmann [aut, cre, cph] (<https://orcid.org/0000-0001-6216-5183>), Susan Vanderplas [aut] (<https://orcid.org/0000-0002-3803-0972>), Ganesh Krishnan [aut], Eric Hare [aut] (<https://orcid.org/0000-0002-4277-3146>)

Repository CRAN

Date/Publication 2024-01-30 10:30:02 UTC

R topics documented:

`df_to_x3p` *Convert a data frame into an x3p file*

Description

Convert a data frame into an x3p file

Usage

```
df_to_x3p(dframe, var = "value")
```

Arguments

`dframe`	data frame. `dframe` must have the columns x, y, and value.
`var`	name of the variable containing the surface measurements. Defaults to "value".

Value

x3p object

`head.x3p` *Show meta information of an x3p file*

Description

`head.x3p` expands the generic head method for x3p objects. It gives a summary of the most relevant 3p meta information and returns the object invisibly.

Usage

```
## S3 method for class 'x3p'
head(x, n = 6L, ...)
```

Arguments

`x`	x3p object
`n`	number of rows/columns of the matrix
`...`	extra parameters passed to head.matrix()

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
head(logo)
```

`image.x3p` *Raster image of an x3p surface*

Description

`image.x3p` expands the generic image method for x3p objects. This image function creates a raster image to show the surface of an x3p file. Due to some inconsistency in the mapping of the origin (0,0), (choice between top left or bottom left) image functions from different packages will result in different images.

Usage

```
## S3 method for class 'x3p'
image(x, ...)
```

Arguments

`x`	an x3p object
`...`	parameters passed into image

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
image(logo)
```

`lea` *Subsampled scan of a land-engraved area*

Description

LEAs (land-engraved areas) are created on the outside of a bullet during the firing process. Depending on the rifling inside the barrel, multiple lands exist for each barrel. Striation marks in these land engraved areas are used in forensic labs to determine whether two bullets were fired from the same firearm.

Usage

```
lea
```

Format

`x3p` object

Examples

```
data(lea)
image(lea)
if (interactive()) x3p_image(lea)
```

---

print.x3p *Show meta information of an x3p file*

Description

print.x3p expands the generic print method for x3p objects. It gives a summary of the most relevant x3p meta information and returns the object invisibly.

Usage

```
## S3 method for class 'x3p'
print(x, ...)
```

Arguments

x	x3p object
...	ignored

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
print(logo)
```

---

stl_to_x3p *Convert an STL file to an x3p file*

Description

STL (STereo Lithographic) files describe 3d objects as mesh objects. Here, we assume that the 3d object consists of a 3d surface on the top of a rectangular, equi-spaced 2d grid. We further assume, that each node of the STL file describes the x-y location of an actual measurement. These measurements are then converted into the surface matrix of an x3p object. The resolution is derived from the distance between consecutive x and y nodes.

Usage

```
stl_to_x3p(stl)
```

Arguments

stl	STL file object or path to the file

Value

x3p object

Examples

```
## Not run:
# the website https://touchterrain.geol.iastate.edu/ allows a download
# of a 3d printable terrain model. For an example we suggest to download a file from there.
gc <- rgl::readSTL("<PATH TO STL FILE>", plot=FALSE)
x3p <- stl_to_x3p(gc)

## End(Not run)
```

`tmd_to_x3p` *Read (or convert) from TMD file to x3p*

Description

TMD files are used in telemetry, specifically, they are a native format used by GelSight to store 3d topographic surface scans.

Usage

```
tmd_to_x3p(tmd_path, yaml_path = NA, verbose = TRUE)
```

Arguments

`tmd_path`	path to TMD file
`yaml_path`	path to corresponding yaml file with meta information. If set to `NA` (default), path of the the tmd file will be tried. If set to NULL, meta file will be ignored.
`verbose`	boolean

Details

The algorithm is based on GelSight's MatLab routine `readtmd.m` published as part of the Github repository `gelsightinc/gsmatlab`

Value

x3p file of the scan. Some rudimentary information will be filled in, information of scanning process, and parameter settings need to be added manually.

Examples

```
#x3p <- tmd_to_x3p("~/Downloads/Sc04.Pl044.Ma4.SB.An80.Pb.DirFo.SizL.tmd") #
#x3p <- tmd_to_x3p("~/Downloads/Sc04.Pl044.Ma4.SB.An80.Pb.DirFo.SizL.tmd",
#                  yaml_path="~/Downloads/scan.yaml") #
```

wire *Subsampled wire cut scan*

Description

An example part of a wire cut in x3p format. The wire cut is part of a CSAFE study involving 1.5 mm Aluminium wires cut by Kaiweet wire-cutters.

Usage

```
wire
```

Format

x3p object

x3p_add_annotation *Add annotations to an x3p object*

Description

Annotations in an x3p object are legend entries for each color of a mask.

Usage

```
x3p_add_annotation(x3p, color, annotation)
```

Arguments

x3p	x3p object
color	name or hex value of color
annotation	character value describing the region

Value

x3p object with the added annotations

Examples

```
## Not run:
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
color_logo <- png::readPNG(system.file("csafe-color.png", package="x3ptools"))
logoplus <- x3p_add_mask(logo, as.raster(color_logo))
x3p_image(logoplus, multiply=50, size = c(741, 419),zoom = 0.5)
logoplus <- x3p_add_annotation(logoplus, "#FFFFFFFF", "background")
logoplus <- x3p_add_annotation(logoplus, "#818285FF", "text")
logoplus <- x3p_add_annotation(logoplus, "#F6BD47FF", "fingerprint")
logoplus <- x3p_add_annotation(logoplus, "#D2202FFF", "fingerprint")
logoplus <- x3p_add_annotation(logoplus, "#92278FFF", "fingerprint")

x3p_add_legend(logoplus)

## End(Not run)
```

---

x3p_add_grid *Add a grid of helper lines to the mask of an x3p object*

Description

Add a grid of lines to overlay the surface of an x3p object. Lines are added to a mask. In case no mask exists, one is created.

Usage

```
x3p_add_grid(
  x3p,
  spaces,
  size = c(1, 3, 5),
  color = c("grey50", "black", "darkred")
)
```

Arguments

x3p	x3p object
spaces	space between grid lines, doubled for x
size	width (in pixels) of the lines
color	(vector of) character values to describe color of lines

Value

x3p object with added vertical lines in the mask

Examples

```
## Not run:
logo <- x3p_read(system.file("csafe-logo.x3p", package = "x3ptools"))
# ten vertical lines across:
logoplus <- x3p_add_grid(logo,
  spaces = 50e-6, size = c(1, 3, 5),
  color = c("grey50", "black", "darkred")
)
x3p_image(logoplus, size = c(741, 419), zoom = 0.5)

## End(Not run)
```

`x3p_add_hline` *Add horizontal line to the mask of an x3p object*

Description

Add horizontal lines to overlay the surface of an x3p object. Lines are added to a mask. In case no mask exists, one is created.

Usage

```
x3p_add_hline(x3p, yintercept, size = 5, color = "#e6bf98")
```

Arguments

`x3p`	x3p object
`yintercept`	(vector of) numerical values for the position of the lines.
`size`	width (in pixels) of the line
`color`	(vector of) character values to describe color of lines

Value

x3p object with added vertical lines in the mask

Examples

```
## Not run:
logo <- x3p_read(system.file("csafe-logo.x3p", package = "x3ptools"))
color_logo <- magick::image_read(system.file("csafe-color.png", package = "x3ptools"))
logoplus <- x3p_add_mask(logo, as.raster(color_logo))
# five horizontal lines at equal intervals:
logoplus <- x3p_add_hline(logo, seq(0, 418 * 6.4500e-7, length = 5), size = 3)
x3p_image(logoplus, size = c(741, 419), zoom = 0.5)

## End(Not run)
```

x3p_add_legend *Display legend in active rgl object*

Description

Display the legend for colors and annotations in the active rgl window. In case no rgl window is opened, a new window displaying the x3p file (using default sizes and zoom) opens.

Usage

```
x3p_add_legend(x3p, colors = NULL)
```

Arguments

x3p	x3p object with a mask
colors	named character vector of colors (in hex format by default), names contain annotations

Examples

```
x3p <- x3p_read(system.file("sample-land.x3p", package="x3ptools"))
## Not run:
# run when rgl can open window on the device
x3p_image(x3p)
x3p_add_legend(x3p) # add legend

## End(Not run)
```

x3p_add_mask *Add/Exchange a mask for an x3p object*

Description

Create a mask for an x3p object in case it does not have a mask yet. Masks are used for overlaying colors on the bullets surface.

Usage

```
x3p_add_mask(x3p, mask = NULL)
```

Arguments

x3p	x3p object
mask	raster matrix of colors with the same dimensions as the x3p surface. If NULL, an object of the right size will be created.

Value

x3p object with added/changed mask

Examples

```
x3p <- x3p_read(system.file("sample-land.x3p", package="x3ptools"))
# x3p file has mask consisting color raster image:
x3p$mask[1:5,1:5]
## Not run:
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
color_logo <- png::readPNG(system.file("csafe-color.png", package="x3ptools"))
logoplus <- x3p_add_mask(logo, as.raster(color_logo))
x3p_image(logoplus, multiply=50, size = c(741, 419),zoom = 0.5)

## End(Not run)
```

---

x3p_add_mask_layer *Add a layer to the mask*

Description

Add a region a mask. The region is specfied as TRUE values in a matrix of the same dimensions as the existing mask. In case no mask exists, one is created.

Usage

```
x3p_add_mask_layer(x3p, mask, color = "red", annotation = "")
```

Arguments

x3p	x3p object
mask	logical matrix of the same dimension as the surface matrix. Values of TRUE are assumed to be added in the mask, values of FALSE are being ignored.
color	name or hex value of color
annotation	character value describing the region

Value

x3p object with changed mask

x3p_add_meta *Add/change xml meta information in x3p object*

Description

Use a specified template to overwrite the general info in the x3p object (and structure of the feature info, if needed).

Usage

```
x3p_add_meta(x3p, template = NULL)

addtemplate_x3p(x3p, template = NULL)
```

Arguments

x3p	x3p object
template	file path to xml template, use NULL for in-built package template

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
# exchange meta information for general x3p information:
logo <- x3p_add_meta(logo, template = system.file("templateXML.xml", package="x3ptools"))
logo$general.info
```

x3p_add_vline *Add vertical line to the mask of an x3p object*

Description

Add vertical lines to overlay the surface of an x3p object. Lines are added to a mask. In case no mask exists, one is created.

Usage

```
x3p_add_vline(x3p, xintercept, size = 5, color = "#e6bf98")
```

Arguments

x3p	x3p object
xintercept	(vector of) numerical values for the position of the lines.
size	width (in pixels) of the line
color	(vector of) character values to describe color of lines

Value

x3p object with added vertical lines in the mask

Examples

```
## Not run:
logo <- x3p_read(system.file("csafe-logo.x3p", package = "x3ptools"))
logo_color <- magick::image_read(system.file("csafe-color.png", package = "x3ptools"))
logoplus <- x3p_add_mask(logo, as.raster(logo_color))
# ten vertical lines across:
logoplus <- x3p_add_vline(logo, seq(0, 740 * 6.4500e-7, length = 5), size = 3)
x3p_image(logoplus, size = c(741, 419), zoom = 0.5)

## End(Not run)
```

---

x3p_average *Average an x3p object*

Description

Calculate blockwise summary statistics on the surface matrix of an x3p. If the x3p object has a mask, the mode of the mask value

Usage

```
x3p_average(x3p, b = 10, f = mean, ...)
```

Arguments

x3p	x3p object
b	positive integer value, block size
f	function aggregate function
...	parameters passed on to function f. Make sure to use na.rm = T as needed.

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
small <- x3p_average(logo)
```

x3p_bin_stripes *Add colored stripes of the surface gradient to the mask of an x3p object*

Description

Apply gradient-based color shading to the mask of a 3d topographic surface. Gradients are determined empirically based on sequentical row- (or column-)wise differences of surface values. The `direction` parameter determines the direction of differencing. If direction is "vertical", columns in the surface matrix are differenced to identify whether 'vertical' stripes exist.

Usage

```
x3p_bin_stripes(
  x3p,
  direction = "vertical",
 colors = rev(c("#b12819", "#d7301f", "#e16457", "#ffffff", "#5186a2", "#175d82",
    "#134D6B")),
  freqs = c(0, 0.05, 0.1, 0.3, 0.7, 0.9, 0.95, 1)
)
```

Arguments

`x3p`	object containing a 3d topographic surface
`direction`	in which the stripes are created: `vertical` or `horizontal`.
`colors`	vector of colors
`freqs`	vector of values corresponding to color frequency (turned into quantiles of the differenced values)

Value

x3p object with mask colored by discretized surface gradient

Examples

```
data(wire)
x3p <- wire
if (interactive()) x3p_image(x3p, size = c(400, 400), zoom=0.8)
x3p_with <- x3p_bin_stripes(x3p, direction="vertical")
x3p_with <- x3p_bin_stripes(x3p, direction="vertical",
colors=c("#b12819","#ffffff","#134D6B"), freqs=c(0, 0.3, 0.7, 1))
if (interactive()) x3p_image(x3p_with, size = c(400, 400), zoom=0.8)

data(lea)
if (interactive()) {
  lea %>% x3p_bin_stripes() %>% x3p_image() # default stripes

# three colors only
```

```
  lea %>% x3p_bin_stripes(
     colors=c("#b12819","#ffffff","#134D6B"),
     freqs=c(0, 0.3, 0.7, 1)) %>%  x3p_image()
}
```

`x3p_circle_select` *Select a circle area on the surface of an x3p file using rgl*

Description

In the active rgl window select a circle on the scan's surface by clicking on three points along the circumference. Make sure that x3p file and the rgl window match. If no rgl window is active, an rgl window opens with the scan.

Usage

```
x3p_circle_select(x3p, col = "#FF0000", update = TRUE)
```

Arguments

`x3p`	x3p file
`col`	character value of the selection color
`update`	boolean value, whether the rgl window should be updated to show the selected circle

Value

x3p file with selected circle in mask

Examples

```
## Not run:
if (interactive) {
  if (!file.exists("fadul1-1.x3p")) {
   url <- "https://tsapps.nist.gov/NRBTD/Studies/CartridgeMeasurement/DownloadMeasurement"
    file <- "2d9cc51f-6f66-40a0-973a-a9292dbee36d"
    download.file(file.path(url, file), destfile="fadul1-1.x3p")
  }
  x3p <- x3p_read("fadul1-1.x3p")
  x3p_image(x3p, size=c(500,500), zoom=.8)
  x3p <- x3p_circle_select(x3p, update=TRUE, col="#FF0000")

  logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
  x3p_image(logo, size=c(500,500), zoom = 1)
  x3p_circle_select(logo, update=TRUE, col="#00FF00")
}
## End(Not run)
```

x3p_crop *Crop an x3p object to a specified width and height*

Description

Cuts out a rectangle of size width x height from the location (x, y) of an x3p object. x and y specify the bottom right corner of the rectangle. In case the dimensions of the surface matrix do not allow for the full dimensions of the rectangle cutout the dimensions are adjusted accordingly.

Usage

```
x3p_crop(x3p, x = 1, y = 1, width = 128, height = 128)
```

Arguments

x3p	x3p object
x	integer, location (in pixels) of the leftmost side of the rectangle,
y	integer, location (in pixels) of the leftmost side of the rectangle,
width	integer, width (in pixels) of the rectangle,
height	integer, height (in pixels) of the rectangle,

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
# crop the x3p file to just the CSAFE logo
logo_only <- x3p_crop(logo, x=20, y=50, width = 255 ,height =310)
logo_only <- x3p_crop(logo, x=20, y=50, width = 255 ,height =510)
# x3p_image(logo_only, size=c(500,500), zoom = 1)
```

x3p_darker *Darken active rgl object*

Description

Makes the currently active rgl object darker by removing a light source. Once all light sources are removed the object can not be any darker.

Usage

```
x3p_darker()
```

Examples

```
x3p <- x3p_read(system.file("sample-land.x3p", package="x3ptools"))
## Not run:
x3p_image(x3p) # run when rgl can open window on the device
x3p_darker() # remove a light source

## End(Not run)
```

`x3p_delete_mask` *Delete mask from an x3p object*

Description

Deletes mask and its annotations from an x3p file.

Usage

```
x3p_delete_mask(x3p)
```

Arguments

`x3p`	x3p object

Value

x3p object without the mask

`x3p_extract` *Extract values from a surface matrix based on a mask*

Description

If a mask is present, a subset of the surface matrix is extracted based on specified value(s).

Usage

```
x3p_extract(x3p, mask_vals)
```

Arguments

`x3p`	x3p object
`mask_vals`	vector of mask value(s)

Value

x3p object

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
# add a mask
logo <- x3p_add_mask(logo)
mask <- t(logo$surface.matrix==median(logo$surface.matrix))
logo <- x3p_add_mask_layer(logo, mask, color = "red", annotation = "median")
x3p_extract(logo, "#cd7f32")
# x3p_image(logo, size=c(500,500), zoom = 1)
```

`x3p_extract_profile` *Interactively select a line on the active rgl device*

Description

In the active rgl device select a line on the 3d surface by clicking on start and end-point (order matters). These points define the beginning and end of a line segment. The line segment is drawn on the mask of the x3p object. The line object is returned as part of the x3p object, if `line_result` is set to `TRUE`

Usage

```
x3p_extract_profile(
  x3p,
  col = "#FF0000",
  update = TRUE,
  line_result = "equi-spaced",
  multiply = 5,
  linewidth = 1
)
```

Arguments

`x3p`	x3p file
`col`	character value of the selection color
`update`	boolean value, whether the rgl window should be updated to show the selected circle
`line_result`	enhance result by a data frame of the line: NULL for no, "raw" for data frame of original x and y (in the mask) and projected x onto the line, "equi-spaced" (default) returns a data frame with equispaced x values after fitting a loess smooth to the raw values. Note that variable x indicates the direction from first click (x=0) to the second click (max x).
`multiply`	integer value, factor to multiply surface values. Only applied if update is true. Defaults to 5,
`linewidth`	line width of the extracted line. Defaults to 1.

Value

x3p file with identified line in the mask. Depending on the setting of `line_result` additional information on the line is attached as a data frame.

Examples

```
## Not run:
if (interactive) {
  x3p <- x3p_read(system.file("sample-land.x3p", package="x3ptools"))
  x3p %>% image_x3p(size=dim(x3p$surface.matrix), multiply=1, zoom=.3)
  x3p <- x3p_extract_profile(x3p, update=TRUE, col="#FFFFFF")
  x3p$line_df %>%
    ggplot(aes(x = x, y = value)) + geom_line()

 x3p$line_df$y <- 1
 sigs <- bulletxtrctr::cc_get_signature(ccdata = x3p$line_df,
   grooves = list(groove=range(x3p$line_df$x)), span1 = 0.75, span2 = 0.03)
 sigs %>%
   ggplot(aes(x = x)) +
     geom_line(aes(y = raw_sig), colour = "grey50") +
     geom_line(aes(y = sig), size = 1) +
     theme_bw()
}
## End(Not run)
```

---

x3p_extract_profile_segments

Extract profiles from surface using multiple segments

Description

The 3d topographic surface is split into multiple segments of width `width` (in pixels) using an overlap of 10% between segments. For each segment, a line is extracted (with `x3p_extract_profile`). Line segments are projected onto the mask of the initial x3p object and exported as a `lines` attribute.

Usage

```
x3p_extract_profile_segments(
  x3p,
  width,
  col = "#FF0000",
  linewidth = 11,
  verbose = TRUE
)
```

Arguments

x3p	object
width	segment width
col	color
linewidth	integer value specifying the width for the profile
verbose	logical

Value

x3p object with added lines attribute.

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
logo <- x3p_m_to_mum(logo)
if(interactive())
  x3p_extract_profile_segments(logo, 850, col="#ffffff", linewidth=5)
```

---

x3p_flip_x *Flip the x coordinate of an x3p file*

Description

Flip the surface matrix of an x3p file along the x axis.

Usage

```
x3p_flip_x(x3p)

x_flip_x3p(x3p)
```

Arguments

x3p	x3p object

Value

x3p object in which the x coordinate is reversed.

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
dim(logo$surface.matrix)
## Not run:
x3p_image(logo)

## End(Not run)
# flip the y-axis for the old ISO standard:
logoflip <- x3p_flip_x(logo)
dim(logoflip$surface.matrix)
## Not run:
x3p_image(logoflip)

## End(Not run)
```

---

x3p_flip_y *Flip the y coordinate of an x3p image*

Description

One of the major changes between the previous two ISO standards is the way the y axis is defined in a scan. The entry (0,0) used to refer to the top left corner of a scan, now it refers to the bottom right corner, which means that all legacy x3p files have to flip their y axis in order to conform to the newest ISO norm.

Usage

```
x3p_flip_y(x3p)

y_flip_x3p(x3p)
```

Arguments

x3p	x3p object

Value

x3p object in which the y coordinate is reversed.

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
dim(logo$surface.matrix)
## Not run:
x3p_image(logo)

## End(Not run)
# flip the y-axis for the old ISO standard:
logoflip <- x3p_flip_y(logo)
```

```
dim(logoflip$surface.matrix)
## Not run:
x3p_image(logoflip)

## End(Not run)
```

`x3p_fuzzyselect` *Interactive selection of region of interest*

Description

Interactive selection of region of interest

Usage

```
x3p_fuzzyselect(x3p, col = "#FF0000", mad = 5, type = "plane", update = TRUE)
```

Arguments

`x3p`	x3p file
`col`	character value of the selection color
`mad`	scalar
`type`	only "plane" is implemented at the moment
`update`	boolean value, whether the rgl window should be updated to show the selected rectangle

Value

x3p file with updated mask

Examples

```
## Not run:
if (interactive) {
  if (!file.exists("fadul1-1.x3p")) {
   url <- "https://tsapps.nist.gov/NRBTD/Studies/CartridgeMeasurement/DownloadMeasurement"
    file <- "2d9cc51f-6f66-40a0-973a-a9292dbee36d"
    download.file(file.path(url, file), destfile="fadul1-1.x3p")
  }
  x3p <- x3p_read("fadul1-1.x3p")
  x3p_image(x3p, size=c(500,500), zoom=.8)
  x3p <- x3p_fuzzyselect(x3p, update=TRUE, col="#FF0000")

  logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
  x3p_image(logo, size=c(500,500), zoom = 1)
  x3p_fuzzyselect(logo, update=TRUE, col="#00FF00")
}
## End(Not run)
```

x3p_get_scale *Check resolution of a scan*

Description

Scans in x3p format capture 3d topogographic surfaces. According to ISO standard ISO5436 - 2000 scans are supposed to be captured in meters. For microscopic images capture in meters might be impractical.

Usage

```
x3p_get_scale(x3p)
```

Arguments

x3p	object

Value

numeric value of resolution per pixel

x3p_image *Plot x3p object as an image*

Description

Plot an interactive surface plot of the x3p matrix. This implementation uses the `rgl` package. In case `rgl.useNULL` is set to TRUE (i.e. no separate window will be opened), an rgl widget can be used to show the surface in the viewer window (see the example).

Usage

```
x3p_image(
  x3p,
  file = NULL,
  col = "#cd7f32",
  size = 750,
  zoom = 0.35,
  multiply = 5,
  update = FALSE,
  ...
)

image_x3p(
  x3p,
  file = NULL,
```

```
  col = "#cd7f32",
  size = c(750, 250),
  zoom = 0.35,
  multiply = 5,
  ...
)
```

Arguments

x3p	x3p object
file	file name for saving, if file is NULL the opengl device stays open. The file extension determines the type of output. Possible extensions are png, stl (suitable for 3d printing), or svg.
col	color specification
size	vector of width and height. If only one value is given, height or width will be adjusted proportionally to the dimensions of the surface matrix of the scan to reach an upper bound of size.
zoom	numeric value indicating the amount of zoom
multiply	exaggerate the relief by factor multiply
update	Boolean value indicating whether a scene should be updated (defaults to FALSE). If FALSE, a new rgl device is opened.
...	not used

Examples

```
save <- getOption("rgl.useNULL")
options(rgl.useNULL=TRUE)

logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
x3p_image(logo, size = c(741, 419), zoom=0.5)
# add crosscut:
logoplus <- x3p_add_hline(logo, yintercept = 50*.645e-6, color = "#e6bf98", size = 5)
x3p_image(logoplus, size = c(741, 419), zoom=0.5)
widget <- rgl::rglwidget()
if (interactive())
  widget

options(rgl.useNULL=save)
```

---

x3p_interpolate *Interpolate from an x3p object*

Description

An interpolated scan is created at specified resolutions `resx`, `resy` in x and y direction. The interpolation is based on `na.approx` from the `zoo` package. It is possible to create interpolations at a higher resolution than the one specified in the data itself, but it is not recommended to do so. `x3p_interpolate` can also be used as a way to linearly interpolate any missing values in an existing scan without changing the resolution.

Usage

```
x3p_interpolate(x3p, resx = 1e-06, resy = resx, maxgap = 1)

interpolate_x3p(x3p, resx = 1e-06, resy = resx, maxgap = 1)
```

Arguments

`x3p`	x3p object
`resx`	numeric value specifying the new resolution for the x axis.
`resy`	numeric value specifying the new resolution for the y axis.
`maxgap`	integer variable used in `na.approx` to specify the maximum number of NAs to be interpolated, defaults to 1.

Value

interpolated x3p object

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
# resolution:
logo$header.info$incrementX
# change resolution to 1 micron = 1e-6 meters
logo2 <- x3p_interpolate(logo, resx = 1e-6)
logo2$header.info$incrementX
```

---

x3p_lighter *Lighten active rgl object*

Description

Make the currently active rgl object lighter. Adds a light source. Up to eight light sources can be added. Alternatively, any rgl light source can be added (see `light3d`).

Usage

```
x3p_lighter()
```

Examples

```
x3p <- x3p_read(system.file("sample-land.x3p", package="x3ptools"))
## Not run:
x3p_image(x3p) # run when rgl can open window on the device
x3p_lighter() # add a light source

## End(Not run)
```

`x3p_mask_legend` *Get legend for mask colors*

Description

Retrieve color definitions and annotations from the mask. If available, results in a named vector of colors.

Usage

```
x3p_mask_legend(x3p)
```

Arguments

`x3p`	x3p object with a mask

Value

named vector of colors, names show annotations. In case no annotations exist NULL is returned.

Examples

```
x3p <- x3p_read(system.file("sample-land.x3p", package="x3ptools"))
x3p_mask_legend(x3p) # annotations and color hex definitions
```

`x3p_mask_quantile` *Draw a quantile region on the mask*

Description

For each x value of the surface matrix add a region to the mask of an x3p object corresponding to the area between two specified quantiles.

Usage

```
x3p_mask_quantile(
  x3p,
  quantiles = c(0.25, 0.75),
  color = "red",
  annotation = "quantile-region"
)
```

Arguments

x3p	x3p object
quantiles	vector of quantiles between which surface matrix values are included in the mask
color	name or hex value of color
annotation	character value describing the region

Value

x3p object with changed mask

---

x3p_modify_xml *Modify xml elements meta information in x3p object*

Description

Identify xml fields in the meta file of an x3p object by name and modify content if uniquely described.

Usage

```
x3p_modify_xml(x3p, element, value)
```

Arguments

x3p	x3p object
element	character or integer. In case of character, name of xml field in the meta file. Note that element can contain regular expressions, e.g. "*" returns all meta fields. In case of integer, element is used as an index for the meta fields.
value	character. Value to be given to the xml field in the meta file.

Value

x3p object with changed meta information

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
x3p_show_xml(logo, "creator")
x3p_modify_xml(logo, "creator", "I did that")
x3p_show_xml(logo, 20)
x3p_modify_xml(logo, 20, "I did that, too")
```

---

x3p_m_to_mum *Convert x3p header information to microns from meters*

Description

ISO standard 5436_2 asks for specification of values in meters. For topographic surfaces collected by microscopes values in microns are more readable. Besides scaling the values in the surface matrix, corresponding increments are changed to microns as well.

Usage

```
x3p_m_to_mum(x3p)
```

Arguments

x3p	x3p file with header information in meters

Value

x3p with header information in microns

---

x3p_read *Read an x3p file into an x3p object*

Description

Read file in x3p format. x3p formats describe 3d topological surface according to ISO standard ISO5436 – 2000. x3p files are a container format implemented as a zip archive of a folder consisting of an xml file of meta information and a binary matrix of numeric surface measurements.

Usage

```
x3p_read(file, size = NA, quiet = T, tmpdir = NULL)

read_x3p(file, size = NA, quiet = T, tmpdir = NULL)
```

Arguments

file	The file path to the x3p file, or an url to an x3p file
size	size in bytes to use for reading the binary file. If not specified, default is used. Will be overwritten if specified in the xml meta file.
quiet	for url downloads, show download progress?
tmpdir	temporary directory to use to extract the x3p file (default NULL uses tempdir() to set a directory).

Value

x3p object consisting of a list of the surface matrix and the four records as specified in the ISO standard

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
```

---

x3p_read_dat *Read data from an x-y-z file*

Description

Read data from an x-y-z file

Usage

```
x3p_read_dat(dat, delim = " ", col_names = FALSE)
```

Arguments

dat	path to the x-y-z file
delim	character determining delimiter
col_names	logical value - does the first line of the file contain the column names? Default is set to FALSE.

Value

x3p object

x3p_read_plux *Read information from plux file*

Description

plux files are zip containers of 3d topographic scans in a format proprietary to Sensofar™. One of the files in the container is the file `index.xml` which contains meta-information on the instrument, scan settings, date, and creator. This information is added to the x3p meta-information.

Usage

```
x3p_read_plux(plux)
```

Arguments

`plux`	path to plux file

Value

xml of general information as stored in the plux file

x3p_rotate *Rotate an x3p object*

Description

Rotate the surface matrix and mask of an x3p object. Also adjust meta information.

Usage

```
x3p_rotate(x3p, angle = 90)

rotate_x3p(x3p, angle = 90)
```

Arguments

`x3p`	x3p object
`angle`	rotate counter-clockwise by angle in degrees.

Examples

```
## Not run:
logo <- x3p_read(system.file("csafe-logo.x3p", package = "x3ptools"))
color_logo <- png::readPNG(system.file("csafe-color.png", package="x3ptools"))
logoplus <- x3p_add_mask(logo, as.raster(color_logo))
dim(logoplus$surface.matrix)
dim(logoplus$mask)
x3p_image(logoplus, multiply=50, size = c(741, 419),zoom = 0.5)

logoplus60 <- x3p_rotate(x3p = logoplus, angle = 60)
dim(logoplus60$surface.matrix)
dim(logoplus60$mask)
x3p_image(logoplus60, multiply=50, size = c(741, 419),zoom = 0.75)

## End(Not run)
```

---

x3p_sample *Sample from an x3p object*

Description

Sample from an x3p object

Usage

```
x3p_sample(x3p, m = 2, mY = m, offset = 0, offsetY = offset)

sample_x3p(x3p, m = 2, mY = m, offset = 0, offsetY = offset)
```

Arguments

x3p	x3p object
m	integer value - every mth value is included in the sample
mY	integer value - every mth value is included in the sample in x direction and every mYth value is included in y direction
offset	integer value between 0 and m-1 to specify offset of the sample
offsetY	integer value between 0 and mY-1 to specify different offsets for x and y direction

Value

down-sampled x3p object

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
dim(logo$surface.matrix)
# down-sample to one-fourth of the image:
logo4 <- x3p_sample(logo, m=4)
dim(logo4$surface.matrix)
## Not run:
x3p_image(logo)
x3p_image(logo4)

## End(Not run)
```

`x3p_scale_unit` *Scale x3p object by given unit*

Description

x3p objects can be presented in different units. ISO standard 5436_2 asks for specification of values in meters. For topographic surfaces collected by microscopes values in microns are more readable. This functions allows to convert between different units.

Usage

```
x3p_scale_unit(x3p, scale_by)
```

Arguments

`x3p`	object in x3p format, 3d topographic surface.
`scale_by`	numeric value. Value the surface to be scaled by. While not enforced, values of `scale_by` make most sense as multiples of 10 (for a metric system).

Value

x3p with header information in microns

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
logo # measurements in meters
x3p_scale_unit(logo, scale_by=10^6) # measurements in microns
```

x3p_select *Draw rectangle on the mask of an x3p file using rgl*

Description

Interactive selection of rectangular area on the mask of an x3p object. Once the function runs, the active rgl window is brought to the front. Select the window with a click, then use click & drag to select a rectangular area. On release, this area is marked in the mask and (if update is TRUE) appears in the selection color in the active rgl window.

Usage

```
x3p_select(x3p, col = "#FF0000", update = TRUE)
```

Arguments

x3p	x3p file
col	character value of the selection color
update	boolean value, whether the rgl window should be updated to show the selected rectangle

Value

x3p file with selection in mask

Examples

```
## Not run:
if (interactive) {
  if (!file.exists("fadul1-1.x3p")) {
   url <- "https://tsapps.nist.gov/NRBTD/Studies/CartridgeMeasurement/DownloadMeasurement"
    file <- "2d9cc51f-6f66-40a0-973a-a9292dbee36d"
    download.file(file.path(url, file), destfile="fadul1-1.x3p")
  }
  x3p <- x3p_read("fadul1-1.x3p")
  x3p_image(x3p, size=c(500,500), zoom=.8)
  x3p <- x3p_select(x3p, update=TRUE, col="#FF0000")

  logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
  x3p_image(logo, size=c(500,500), zoom = 1)
  x3p_select(logo, update=TRUE, col="#00FF00")
}
## End(Not run)
```

x3p_shade_mask *Shade the mask of an x3p object to reflect its surface profile*

Description

Apply color shading to the mask of a 3d topographic surface.

Usage

```
x3p_shade_mask(
  x3p,
 colors = rev(c("#b12819", "#d7301f", "#e16457", "#ffffff", "#5186a2", "#175d82",
    "#134D6B")),
  freqs = c(0, 0.05, 0.25, 0.45, 0.55, 0.75, 0.95, 1)
)
```

Arguments

x3p	object containing a 3d topographic surface
colors	vector of colors
freqs	vector of values corresponding to color frequency (turned into quantiles of the differenced values)

Value

x3p object with color-shaded mask

Examples

```
## Not run:
data(wire)
x3p <- wire
x3p_image(x3p, size = c(400, 400), zoom=0.8)
x3p_with <- x3p %>% x3p_shade_mask()
x3p_image(x3p_with, size = c(400, 400), zoom=0.8)

data(lea)
lea %>% x3p_shade_mask() %>% x3p_image()
lea %>% x3p_shade_mask(freqs = c(0, 0.05, 0.1, 0.3,0.7, 0.9, 0.95, 1)) %>% x3p_image()

## End(Not run)
```

x3p_show_xml *Show xml elements from meta information in x3p object*

Description

Identify xml fields by name and show content.

Usage

```
x3p_show_xml(x3p, element)
```

Arguments

x3p	x3p object
element	character or integer (vector). In case of character, name of xml field in the meta file. Note that element can contain regular expressions, e.g. "*" returns all meta fields. In case of integer, element is used as an index vector for the meta fields.

Value

list of exact field names and their contents

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
x3p_show_xml(logo, "creator") # all fields containing the word "creator"
x3p_show_xml(logo, "axis")
x3p_show_xml(logo, "CZ.AxisType")
# show all fields:
x3p_show_xml(logo, "*")
# show first five fields
x3p_show_xml(logo, 1:5)
```

x3p_snapshot *Take a snapshot of the active rgl device and save in a file*

Description

Make a snapshot of the current rgl device and save it to file. Options for file formats are png, svg, and stl (for 3d printing).

Usage

```
x3p_snapshot(file)
```

Arguments

file	file name for saving. The file extension determines the type of output. Possible extensions are png, stl (suitable for 3d printing), or svg.

---

x3p_to_df *Convert an x3p file into a data frame*

Description

An x3p file consists of a list with meta info and a 2d matrix with scan depths. fortify turns the matrix into a data frame, using the parameters of the header as necessary.

Usage

```
x3p_to_df(x3p)
```

Arguments

x3p	a file in x3p format as returned by function x3p_read

Value

data frame with variables x, y, and value and meta function in attribute

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
logo_df <- x3p_to_df(logo)
head(logo_df)
```

---

x3p_transpose *Transpose an x3p object*

Description

Transpose the surface matrix of an x3p object. Also adjust meta information.

Usage

```
x3p_transpose(x3p)

transpose_x3p(x3p)
```

Arguments

x3p	x3p object

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
dim(logo$surface.matrix)
## Not run:
x3p_image(logo)

## End(Not run)
#  transpose the image
logotp <- x3p_transpose(logo)
dim(logotp$surface.matrix)
## Not run:
x3p_image(logotp)

## End(Not run)
```

`x3p_trim_na` *Trim rows and columns with missing values only from an x3p*

Description

Trims rows and columns from the edges of a surface matrix that contain missing values only.

Usage

```
x3p_trim_na(x3p, ratio = 1)
```

Arguments

`x3p`	x3p object
`ratio`	ratio between zero and one, indicating the percent of values that need to be missing in each row and column, for the row or column to be removed

Value

x3p object of the same or smaller dimension where missing values are removed from the boundaries

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
logo$surface.matrix[logo$surface.matrix == median(logo$surface.matrix)] <- NA
x3p_trim_na(logo) # reduced to dimension: 668 by 268
```

`x3p_write` *Write an x3p object to a file*

Description

Write an x3p object to a file

Usage

```
x3p_write(x3p, file, size = 8, quiet = F, create_dir = T)

write_x3p(x3p, file, size = 8, quiet = F)
```

Arguments

`x3p`	x3p object
`file`	path to where the file should be written
`size`	integer. The number of bytes per element in the surface matrix used for creating the binary file. Use size = 4 for 32 bit IEEE 754 floating point numbers and size = 8 for 64 bit IEEE 754 floating point number (default).
`quiet`	suppress messages
`create_dir`	boolean. create directory for saving file, if necessary. Posts a message in case a directory is created.

Examples

```
logo <- x3p_read(system.file("csafe-logo.x3p", package="x3ptools"))
# write a copy of the file into a temporary file
x3p_write(logo, file = tempfile(fileext="x3p"))
```

Index