

Package ‘cmcR’

December 16, 2020

Type Package

Title An Implementation of the 'Congruent Matching Cells' Method

Version 0.1.3

Maintainer Joe Zemmels <jzemmels@iastate.edu>

Description An open-source implementation of the 'Congruent Matching Cells' method for cartridge case identification as proposed by Song (2013) <https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=911193> as well as an extension of the method proposed by Tong et al. (2015) <[doi:10.6028/jres.120.008](https://doi.org/10.6028/jres.120.008)>. Provides a wide range of pre, inter, and post-processing options when working with cartridge case scan data and their associated comparisons. See the cmcR package website for more details and examples.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports magrittr, x3ptools, dplyr, ggplot2, imager, purrr, zoo, stringr, assertthat, stats, utils, scales, ggnewscale, quantreg, tibble, tidyr, rlang

Suggests knitr, rmarkdown, markdown, testthat, DT, magick, rgl, covr, gridExtra, cowplot

VignetteBuilder knitr

Depends R (>= 3.5.0)

NeedsCompilation no

Author Joe Zemmels [aut, cre],
Heike Hofmann [aut],
Susan VanderPlas [aut]

Repository CRAN

Date/Publication 2020-12-16 10:30:03 UTC

R topics documented:

cmcPlot	2
comparison_allTogether	4
comparison_calcPropMissing	5
comparison_cellDivision	6
comparison_cor	7
comparison_fft_ccf	8
comparison_getTargetRegions	9
comparison_replaceMissing	10
comparison_standardizeHeights	11
decision_CMC	13
decision_combineDirections	15
decision_highCMC_cmcThetaDistrib	17
decision_highCMC_identifyHighCMCThetas	18
fadulData_processed	19
preProcess_crop	20
preProcess_gaussFilter	22
preProcess_ransacLevel	23
preProcess_removeFPCircle	25
preProcess_removeTrend	27
x3pListPlot	28
Index	30

cmcPlot	<i>Visualize initial and high CMCs for a cartridge case pair comparison</i>
---------	---

Description

Constructs either a single faceted plot or a list of plots depicting the CMCs/non-CMCs under the initially proposed and High CMC methods for a pair of cartridge case scans

Usage

```
cmcPlot(
  reference,
  target,
  reference_v_target_CMCs,
  target_v_reference_CMCs = reference_v_target_CMCs,
  corColName = "pairwiseCompCor",
  type = "faceted",
  x3pNames = c("reference", "target"),
  legend.quantiles = c(0, 0.01, 0.25, 0.5, 0.75, 0.99, 1),
  height.colors = c("#1B1B1B", "#404040", "#7B7B7B", "#B0B0B0", "#DBDBDB", "#F7F7F7",
    "#E4E4E4", "#C5C5C5", "#999999", "#717171", "#4E4E4E"),
  cell.colors = c("#a50026", "#313695"),
  cell.alpha = 0.2,
```

```
  na.value = "gray80"
)
```

Arguments

reference	an x3p object
target	a different x3p object
reference_v_target_CMCs	CMCs for the comparison between the reference scan and the target scan.
target_v_reference_CMCs	(optional) CMCs for the comparison between the target scan and the reference scan. If this is missing, then only the original method CMCs will be plotted
corColName	name of correlation similarity score column used to identify the CMCs in the two comparison_*_df data frames (e.g., pairwiseCompCor)
type	argument to be passed to cmcR::x3pListPlot function
x3pNames	(Optional) Names of x3p objects to be included in x3pListPlot function
legend.quantiles	vector of quantiles to be shown as tick marks on legend plot
height.colors	vector of colors to be passed to scale_fill_gradientn that dictates the height value colorscale
cell.colors	vector of 2 colors for plotting non-matching and matching (in that order) cells
cell.alpha	sets alpha of cells (passed to geom_polygon)
na.value	color to be used for NA values (passed to scale_fill_gradientn)

Value

A list of 4 ggplot objects showing the CMCs identified under both decision rules and in both comparison directions.

Examples

```
#Takes > 5 seconds to run

data(fadul1.1_processed, fadul1.2_processed)

comparisonDF_1to2 <- purrr::map_dfr(seq(-30, 30, by = 3),
  ~ comparison_allTogether(fadul1.1_processed,
    fadul1.2_processed,
    theta = .))

comparisonDF_2to1 <- purrr::map_dfr(seq(-30, 30, by = 3),
  ~ comparison_allTogether(fadul1.2_processed,
    fadul1.1_processed,
    theta = .))

comparisonDF_1to2 <- comparisonDF_1to2 %>%
dplyr::mutate(originalMethodClassif = decision_CMC(cellIndex = cellIndex,
  x = x,
```

```

                                y = y,
                                theta = theta,
                                corr = pairwiseCompCor),
  highCMCClassif = decision_CMC(cellIndex = cellIndex,
                                x = x,
                                y = y,
                                theta = theta,
                                corr = pairwiseCompCor,
                                tau = 1))

comparisonDF_2to1 <- comparisonDF_2to1 %>%
  dplyr::mutate(originalMethodClassif = decision_CMC(cellIndex = cellIndex,
                                                    x = x,
                                                    y = y,
                                                    theta = theta,
                                                    corr = pairwiseCompCor),
                highCMCClassif = decision_CMC(cellIndex = cellIndex,
                                                    x = x,
                                                    y = y,
                                                    theta = theta,
                                                    corr = pairwiseCompCor,
                                                    tau = 1))

cmcPlot(fadul1.1_processed,
        fadul1.2_processed,
        comparisonDF_1to2,
        comparisonDF_2to1,
        corColName = "pairwiseCompCor")

```

comparison_allTogether

Performs all steps in the cell-based comparison procedure.

Description

Performs all steps in the cell-based comparison procedure.

Usage

```

comparison_allTogether(
  reference,
  target,
  theta = 0,
  numCells = 64,
  maxMissingProp = 0.85
)

```

Arguments

reference	an x3p object containing a breech face scan to be treated as the "reference scan" partitioned into a grid of cells
target	an x3p object containing a breech face scan to be treated as the "target scan" that the reference scan's cells are compared to
theta	degrees that the target scan is to be rotated prior extracting regions.
numCells	number of cells to partition the breech face scan into. Must be a perfect square (49, 64, 81, etc.)
maxMissingProp	maximum proportion of missing values allowed for each cell/region.

```

data(fadul1.1_processed,fadul1.2_processed)
comparisonDF <- comparison_allTogether(reference = fadul1.1_processed, target = fadul1.2_processed)
head(comparisonDF)

```

Value

a tibble object containing cell indices and the x, y, FFT-based CCF, and pairwise-complete correlation associated with the comparison between each cell and its associated target scan region (after rotating the target scan by theta degrees)

Examples

```

data(fadul1.1_processed, fadul1.2_processed)

cellTibble <- comparison_allTogether(reference = fadul1.1_processed, target = fadul1.2_processed)

head(cellTibble)

```

```
comparison_calcPropMissing
```

Calculate the proportion of missing values in a breech face scan

Description

Calculate the proportion of missing values in a breech face scan

Usage

```
comparison_calcPropMissing(heightValues)
```

Arguments

heightValues list/tibble column of x3p objects

Value

a vector of the same length as the input containing the proportion of missing values in each x3p object's breech face scan.

Examples

```
data(fadul1.1_processed)

cellTibble <- fadul1.1_processed %>%
  comparison_cellDivision(numCells = 64) %>%
  dplyr::mutate(cellPropMissing = comparison_calcPropMissing(heightValues = cellHeightValues))

head(cellTibble)
```

comparison_cellDivision

Split a reference scan into a grid of cells

Description

Split a reference scan into a grid of cells

Arguments

x3p	an x3p object containing a breech face scan
numCells	number of cells to partition the breech face scan into. Must be a perfect square (49, 64, 81, etc.)

Value

A tibble containing a numCells number of rows. Each row contains a single cell's index of the form (row #, col #) and an x3p object containing the breech face scan of that cell.

Examples

```
data(fadul1.1_processed)

cellTibble <- fadul1.1_processed %>%
  comparison_cellDivision(numCells = 64)

head(cellTibble)
```

comparison_cor	<i>Calculates correlation between a cell and a matrix of the same dimensions extracted from the cell's associated region.</i>
----------------	---

Description

Calculates correlation between a cell and a matrix of the same dimensions extracted from the cell's associated region.

Usage

```
comparison_cor(
  cellHeightValues,
  regionHeightValues,
  fft_ccf_df,
  use = "pairwise.complete.obs"
)
```

Arguments

cellHeightValues	list/tibble column of x3p objects containing a reference scan's cells (as returned by comparison_cellDivision)
regionHeightValues	list/tibble column of x3p objects containing a target scan's regions (as returned by comparison_getTargetRegions)
fft_ccf_df	data frame/tibble column containing the data frame of (x,y) and CCF values returned by comparison_fft_ccf
use	argument for stats::cor

Value

A vector of the same length as the input containing correlation values at the estimated alignment between each reference cell and its associated target region

Examples

```
data(fadul1.1_processed, fadul1.2_processed)

cellTibble <- fadul1.1_processed %>%
  comparison_cellDivision(numCells = 64) %>%
  dplyr::mutate(regionHeightValues =
    comparison_getTargetRegions(cellHeightValues = cellHeightValues,
                               target = fadul1.2_processed)) %>%
  dplyr::mutate(cellPropMissing =
    comparison_calcPropMissing(heightValues = cellHeightValues),
               regionPropMissing =
```

```

      comparison_calcPropMissing(heightValues = regionHeightValues)) %>%
dplyr::filter(cellPropMissing <= .85 & regionPropMissing <= .85) %>%
dplyr::mutate(cellHeightValues =
  comparison_standardizeHeights(heightValues = cellHeightValues),
  regionHeightValues =
  comparison_standardizeHeights(heightValues = regionHeightValues)) %>%
dplyr::mutate(cellHeightValues =
  comparison_replaceMissing(heightValues = cellHeightValues),
  regionHeightValues =
  comparison_replaceMissing(heightValues = regionHeightValues)) %>%
dplyr::mutate(fft_ccf_df = comparison_fft_ccf(cellHeightValues,
  regionHeightValues)) %>%
dplyr::mutate(pairwiseCompCor = comparison_cor(cellHeightValues,
  regionHeightValues,
  fft_ccf_df))

head(cellTibble)

```

comparison_fft_ccf	<i>Estimate translation alignment between a cell/region pair based on the Cross-Correlation Theorem.</i>
--------------------	--

Description

Estimate translation alignment between a cell/region pair based on the Cross-Correlation Theorem.

Usage

```
comparison_fft_ccf(cellHeightValues, regionHeightValues)
```

Arguments

cellHeightValues

list/tibble column of x3p objects containing a reference scan's cells (as returned by comparison_cellDivision)

regionHeightValues

list/tibble column of x3p objects containing a target scan's regions (as returned by comparison_getTargetRegions)

Value

A list of the same length as the input containing data frames of the translation (x,y) values at which each reference cell is estimated to align in its associated target region and the CCF value at this alignment.

a data frame containing the translation (x,y) at which the CCF was maximized in aligning a target scan region to its associated reference scan cell.

Note

The FFT is not defined for matrices containing missing values. The missing values in the cell and region need to be replaced before using this function. See the [comparison_replaceMissing](#) function to replace missing values after standardization.

See Also

<https://mathworld.wolfram.com/Cross-CorrelationTheorem.html>

Examples

```
data(fadul1.1_processed, fadul1.2_processed)

cellTibble <- fadul1.1_processed %>%
  comparison_cellDivision(numCells = 64) %>%
  dplyr::mutate(regionHeightValues =
    comparison_getTargetRegions(cellHeightValues = cellHeightValues,
                                target = fadul1.2_processed)) %>%
  dplyr::mutate(cellPropMissing =
    comparison_calcPropMissing(heightValues = cellHeightValues),
    regionPropMissing =
    comparison_calcPropMissing(heightValues = regionHeightValues)) %>%
  dplyr::filter(cellPropMissing <= .85 & regionPropMissing <= .85) %>%
  dplyr::mutate(cellHeightValues =
    comparison_standardizeHeights(heightValues = cellHeightValues),
    regionHeightValues =
    comparison_standardizeHeights(heightValues = regionHeightValues)) %>%
  dplyr::mutate(cellHeightValues =
    comparison_replaceMissing(heightValues = cellHeightValues),
    regionHeightValues =
    comparison_replaceMissing(heightValues = regionHeightValues)) %>%
  dplyr::mutate(fft_ccf_df = comparison_fft_ccf(cellHeightValues,
    regionHeightValues))

cellTibble %>%
  tidyr::unnest(cols = fft_ccf_df) %>%
  head()
```

comparison_getTargetRegions

Extract regions from a target scan based on associated cells in reference scan

Description

Extract regions from a target scan based on associated cells in reference scan

Usage

```
comparison_getTargetRegions(
  cellHeightValues,
  target,
  theta = 0,
  regionSizeMultiplier = 9
)
```

Arguments

cellHeightValues list/tibble column of x3p objects containing a reference scan's cells (as returned by comparison_cellDivision)

target x3p object containing a breach face scan to be compared to the reference cell.

theta degrees that the target scan is to be rotated prior extracting regions.

regionSizeMultiplier ratio between the area of each target scan regions and the reference scan cells (e.g., 9 means that the regions' surface matrices will have thrice the number of rows and columns as the cells' surface matrices, 4 means twice the number rows and columns, etc.)

Value

A list of the same length as the input containing x3p objects from the target scan.

Examples

```
data(fadul1.1_processed, fadul1.2_processed)

cellTibble <- fadul1.1_processed %>%
  comparison_cellDivision(numCells = 64) %>%
  dplyr::mutate(regionHeightValues = comparison_getTargetRegions(cellHeightValues = cellHeightValues,
                                                                target = fadul1.2_processed)) %>%
  dplyr::mutate(cellPropMissing = comparison_calcPropMissing(heightValues = cellHeightValues),
               regionPropMissing = comparison_calcPropMissing(heightValues = regionHeightValues)) %>%
  dplyr::filter(cellPropMissing <= .85 & regionPropMissing <= .85)

head(cellTibble)
```

comparison_replaceMissing
Replace missing values in a scan

Description

Replace missing values in a scan

Usage

```
comparison_replaceMissing(heightValues, replacement = 0)
```

Arguments

```
heightValues    list/tibble column of x3p objects
replacement     value to replace NAs
```

Value

A list of the same length as the input containing x3p objects for which NA values have been replaced.

Examples

```
data(fadul1.1_processed, fadul1.2_processed)

cellTibble <- fadul1.1_processed %>%
  comparison_cellDivision(numCells = 64) %>%
  dplyr::mutate(regionHeightValues =
    comparison_getTargetRegions(cellHeightValues = cellHeightValues,
                                target = fadul1.2_processed)) %>%
  dplyr::mutate(cellPropMissing =
    comparison_calcPropMissing(heightValues = cellHeightValues),
    regionPropMissing =
    comparison_calcPropMissing(heightValues = regionHeightValues)) %>%
  dplyr::filter(cellPropMissing <= .85 & regionPropMissing <= .85) %>%
  dplyr::mutate(cellHeightValues =
    comparison_standardizeHeights(heightValues = cellHeightValues),
    regionHeightValues =
    comparison_standardizeHeights(heightValues = regionHeightValues)) %>%
  dplyr::mutate(cellHeightValues =
    comparison_replaceMissing(heightValues = cellHeightValues),
    regionHeightValues =
    comparison_replaceMissing(heightValues = regionHeightValues))

head(cellTibble)
```

```
comparison_standardizeHeights
```

Standardize height values of a scan by centering/scaling by desired statistics and replacing missing values

Description

Standardize height values of a scan by centering/scaling by desired statistics and replacing missing values

Usage

```
comparison_standardizeHeights(
  heightValues,
  withRespectTo = "individualCell",
  centerBy = mean,
  scaleBy = sd
)
```

Arguments

heightValues	list/tibble column of x3p objects
withRespectTo	currently ignored
centerBy	statistic by which to center (i.e., subtract from) the height values
scaleBy	statistic by which to scale (i.e., divide) the height values

Value

A list of the same length as the input containing x3p objects with standardized surface matrices

Note

this function adds information to the metainformation of the x3p scan it is given that is required for calculating, for example, the pairwise-complete correlation using the comparison_cor function.

Examples

```
data(fadul1.1_processed, fadul1.2_processed)

cellTibble <- fadul1.1_processed %>%
  comparison_cellDivision(numCells = 64) %>%
  dplyr::mutate(regionHeightValues = comparison_getTargetRegions(cellHeightValues = cellHeightValues,
                                                                target = fadul1.2_processed)) %>%
  dplyr::mutate(cellPropMissing = comparison_calcPropMissing(heightValues = cellHeightValues),
               regionPropMissing = comparison_calcPropMissing(heightValues = regionHeightValues)) %>%
  dplyr::filter(cellPropMissing <= .85 & regionPropMissing <= .85) %>%
  dplyr::mutate(cellHeightValues = comparison_standardizeHeights(heightValues = cellHeightValues),
               regionHeightValues = comparison_standardizeHeights(heightValues = regionHeightValues))

head(cellTibble)
```

decision_CMC	<i>Applies the decision rules of the original method of Song (2013) or the High CMC method of Tong et al. (2015)</i>
--------------	--

Description

Applies the decision rules of the original method of Song (2013) or the High CMC method of Tong et al. (2015)

Usage

```
decision_CMC(
  cellIndex,
  x,
  y,
  theta,
  corr,
  xThresh = 20,
  yThresh = xThresh,
  thetaThresh = 6,
  corrThresh = 0.5,
  tau = NULL
)
```

Arguments

cellIndex	vector/tibble column containing cell indices corresponding to a reference cell
x	vector/tibble column containing x horizontal translation values
y	vector/tibble column containing y vertical translation values
theta	vector/tibble column containing theta rotation values
corr	vector/tibble column containing correlation similarity scores between a reference cell and its associated target region
xThresh	used to classify particular x values "congruent" (conditional on a particular theta value) if they are within xThresh of the theta-specific median x value
yThresh	used to classify particular y values "congruent" (conditional on a particular theta value) if they are within yThresh of the theta-specific median y value
thetaThresh	(original method of Song (2013)) used to classify particular theta values "congruent" if they are within thetaThresh of the median theta value. (High CMC) defines how wide a High CMC mode is allowed to be in the CMC-theta distribution before it's considered too diffuse
corrThresh	to classify particular correlation values "congruent" (conditional on a particular theta value) if they are at least corrThresh

`tau` (optional) parameter required to apply the High CMC method of Tong et al. (2015). If not given, then the decision rule of the original method of Song (2013) is applied. This number is subtracted from the maximum CMC count achieved in the CMC-theta distribution. Theta values with CMC counts above this value are considered to have "high" CMC counts.

Value

A vector of the same length as the input containing the CMC classification under one of the two decision rules.

See Also

https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=911193

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4730689/pdf/jres.120.008.pdf>

Examples

```
## Not run:
data(fadul1.1_processed, fadul1.2_processed)

comparisonDF <- purrr::map_dfr(seq(-30, 30, by = 3),
                             ~ comparison_allTogether(fadul1.1_processed,
                                                       fadul1.2_processed,
                                                       theta = .))

comparisonDF <- comparisonDF %>%
dplyr::mutate(originalMethodClassif = decision_CMC(cellIndex = cellIndex,
                                                  x = x,
                                                  y = y,
                                                  theta = theta,
                                                  corr = pairwiseCompCor),
              highCMCClassif = decision_CMC(cellIndex = cellIndex,
                                             x = x,
                                             y = y,
                                             theta = theta,
                                             corr = pairwiseCompCor,
                                             tau = 1))

comparisonDF %>%
dplyr::filter(originalMethodClassif == "CMC" | highCMCClassif == "CMC")

## End(Not run)
```

 decision_combineDirections

Combine data frames containing CMC results from 2 comparison directions

Description

Combines CMC results from two comparison directions of a single cartridge case pair (i.e., where each cartridge case scan has been treated as both the reference and target scan). This function assumes that the CMC results are data frames with columns called "originalMethodClassif" and "highCMCClassif" containing CMCs identified under the original method of Song (2013) and the High CMC method of Tong et al. (2015) (see example).

Usage

```
decision_combineDirections(
  reference_v_target_CMCs,
  target_v_reference_CMCs,
  corColName = "pairwiseCompCor",
  missingThetaDecision = "fail",
  compareThetas = TRUE,
  thetaThresh = 6
)
```

Arguments

reference_v_target_CMCs	CMCs for the comparison between the reference scan and the target scan.
target_v_reference_CMCs	(optional) CMCs for the comparison between the target scan and the reference scan. If this is missing, then only the original method CMCs will be plotted
corColName	name of correlation similarity score column used to identify the CMCs in the two comparison_*_df data frames (e.g., pairwiseCompCor)
missingThetaDecision	dictates how function should handle situations in which one direction passes the high CMC criterion while another direction does not. "dismiss": only counts the initial CMCs in failed direction and high CMCs in successful direction. "fail": only counts the initial CMCs in either direction and returns the minimum of these two numbers.
compareThetas	dictates if the consensus theta values determined under the initially proposed method should be compared to the consensus theta values determined under the High CMC method. In particular, determines for each direction whether the consensus theta values determined under the two methods are within theta_thresh of each other. It is often the case that non-matching cartridge cases, even if they pass the High CMC criterion, will have differing consensus theta values under the two methods. If this isn't taken into account, non-matches tend to be assigned a lot of false positive CMCs under the High CMC method.

thetaThresh (original method of Song (2013)) used to classify particular theta values "congruent" if they are within thetaThresh of the median theta value. (High CMC) defines how wide a High CMC mode is allowed to be in the CMC-theta distribution before it's considered too diffuse. This is also used in this function to determine whether the estimated alignment theta values from the two comparison directions are "approximately" opposite (i.e., within thetaThresh of each other in absolute value), which they should be if the cartridge case pair is a known match.

Value

a list of 2 elements: (1) the CMCs identified under the original method of Song (2013) for both comparison directions since Song (2013) does not indicate whether/how results are combined and (2) the combined CMC results under the High CMC method.

Examples

```
## Not run:
data(fadul1.1_processed, fadul1.2_processed)

comparisonDF_1to2 <- purrr::map_dfr(seq(-30,30,by = 3),
  ~ comparison_allTogether(fadul1.1_processed,
    fadul1.2_processed,
    theta = .))

comparisonDF_2to1 <- purrr::map_dfr(seq(-30,30,by = 3),
  ~ comparison_allTogether(fadul1.2_processed,
    fadul1.1_processed,
    theta = .))

comparisonDF_1to2 <- comparisonDF_1to2 %>%
dplyr::mutate(originalMethodClassif = decision_CMC(cellIndex = cellIndex,
  x = x,
  y = y,
  theta = theta,
  corr = pairwiseCompCor),
  highCMCClassif = decision_CMC(cellIndex = cellIndex,
  x = x,
  y = y,
  theta = theta,
  corr = pairwiseCompCor,
  tau = 1))

comparisonDF_2to1 <- comparisonDF_2to1 %>%
dplyr::mutate(originalMethodClassif = decision_CMC(cellIndex = cellIndex,
  x = x,
  y = y,
  theta = theta,
  corr = pairwiseCompCor),
  highCMCClassif = decision_CMC(cellIndex = cellIndex,
  x = x,
  y = y,
```



```

                                theta = theta,
                                corr = pairwiseCompCor,
                                tau = 1))

decision_combineDirections(comparisonDF_1to2,comparisonDF_2to1)

## End(Not run)

```

```
decision_highCMC_cmcThetaDistrib
```

Compute CMC-theta distribution for a set of comparison features

Description

Compute CMC-theta distribution for a set of comparison features

Usage

```

decision_highCMC_cmcThetaDistrib(
  cellIndex,
  x,
  y,
  theta,
  corr,
  xThresh = 20,
  yThresh = xThresh,
  corrThresh = 0.5
)

```

Arguments

cellIndex	vector/tibble column containing cell indices corresponding to a reference cell
x	vector/tibble column containing x horizontal translation values
y	vector/tibble column containing y vertical translation values
theta	vector/tibble column containing theta rotation values
corr	vector/tibble column containing correlation similarity scores between a reference cell and its associated target region
xThresh	used to classify particular x values "congruent" (conditional on a particular theta value) if they are within xThresh of the theta-specific median x value
yThresh	used to classify particular y values "congruent" (conditional on a particular theta value) if they are within yThresh of the theta-specific median y value
corrThresh	to classify particular correlation values "congruent" (conditional on a particular theta value) if they are at least corrThresh

Value

a vector of the same length as the input containing a "CMC Candidate" or "Non-CMC Candidate" classification based on whether the particular cellIndex has congruent x,y, and theta features.

Note

This function is a helper internally called in the decision_CMC function. It is exported to be used as a diagnostic tool for the High CMC method

Examples

```
## Not run:
data(fadul1.1_processed, fadul1.2_processed)

comparisonDF <- purrr::map_dfr(seq(-30,30,by = 3),
                             ~ comparison_allTogether(fadul1.1_processed,
                                                       fadul1.2_processed,
                                                       theta = .))

comparisonDF <- comparisonDF %>%
dplyr::mutate(cmcThetaDistribClassif = decision_highCMC_cmcThetaDistrib(cellIndex = cellIndex,
                                                                    x = x,
                                                                    y = y,
                                                                    theta = theta,
                                                                    corr = pairwiseCompCor))

comparisonDF %>%
dplyr::filter(cmcThetaDistribClassif == "CMC Candidate") %>%
ggplot2::ggplot(ggplot2::aes(x = theta)) +
ggplot2::geom_bar(stat = "count")

## End(Not run)
```

```
decision_highCMC_identifyHighCMCThetas
```

Classify theta values in CMC-theta distribution as having "High" or "Low" CMC candidate counts

Description

Classify theta values in CMC-theta distribution as having "High" or "Low" CMC candidate counts

Usage

```
decision_highCMC_identifyHighCMCThetas(cmcThetaDistrib, tau = 1)
```

Arguments

`cmcThetaDistrib` output of the `decision_highCMC_cmcThetaDistrib` function

`tau` constant used to define a "high" CMC count. This number is subtracted from the maximum CMC count achieved in the CMC-theta distribution. Theta values with CMC counts above this value are considered to have "high" CMC counts.

Value

A vector of the same length as the input containing "High" or "Low" classification based on whether the associated theta value has a High CMC Candidate count.

Note

This function is a helper internally called in the `decision_CMC` function. It is exported to be used as a diagnostic tool for the High CMC method

Examples

```
## Not run:
data(fadul1.1_processed, fadul1.2_processed)

comparisonDF <- purrr::map_dfr(seq(-30,30,by = 3),
                             ~ comparison_allTogether(fadul1.1_processed,
                                                       fadul1.2_processed,
                                                       theta = .))

highCMCthetas <- comparisonDF %>%
dplyr::mutate(cmcThetaDistribClassif = decision_highCMC_cmcThetaDistrib(cellIndex = cellIndex,
                                                                    x = x,
                                                                    y = y,
                                                                    theta = theta,
                                                                    corr = pairwiseCompCor)) %>%

decision_highCMC_identifyHighCMCThetas(tau = 1)

highCMCthetas %>%
dplyr::filter(cmcThetaDistribClassif == "CMC Candidate") %>%
ggplot2::ggplot(ggplot2::aes(x = theta, fill = thetaCMCIdentif)) +
ggplot2::geom_bar(stat = "count")

## End(Not run)
```

`fadulData_processed` *Processed versions of the `fadul1.1_raw` and `fadul1.2_raw` datasets using `preProcess_*` functions from the `cmcR` package*

Description

"Fadul 1-1" and "Fadul 1-2" cartridge cases from Fadul et al. (2011). The scans have been down-sampled by a factor of 8 and processed using functions from the cmcR package.

Usage

```
fadul1.1_processed
```

```
fadul1.2_processed
```

Format

An x3p object containing a surface matrix and meta-information concerning the conditions under which the scan was taken

header.info size and resolution of scan

surface.matrix spatially-ordered matrix of elements representing the height values of the processed cartridge case surface at particular locations

feature.info provides structure for storing surface data

general.info information concerning the author of the scan and capturing device

matrix.info provides link to surface measurements in binary format

An object of class x3p of length 5.

Source

<https://tsapps.nist.gov/NRBTD/Studies/CartridgeMeasurement/Details/2d9cc51f-6f66-40a0-973a-a9292db>

See Also

T. Fadul, G. Hernandez, S. Stoiloff, and G. Sneh. An Empirical Study to Improve the Scientific Foundation of Forensic Firearm and Tool Mark Identification Utilizing 10 Consecutively Manufactured Slides, 2011.

<https://github.com/heike/x3ptools>

preProcess_crop

Remove observations from the exterior of interior of a breech face scan

Description

Remove observations from the exterior of interior of a breech face scan

Usage

```
preProcess_crop(
  x3p,
  region = "exterior",
  radiusOffset = 0,
  croppingThresh = 1,
  agg_function = median,
  scheme = 3,
  high_connectivity = FALSE,
  tolerance = 0
)
```

Arguments

x3p	an x3p object containing the surface matrix of a cartridge case scan
region	dictates whether the observations on the "exterior" or "interior" of the scan are removed
radiusOffset	number of pixels to add to estimated breech face radius. This is commonly a negative value (e.g., -30 for region = "exterior") to trim the cartridge case primer roll-off from the returned, cropped surface matrix or a positive value (e.g., 200 for region = "interior") to remove observations around the firing pin impression hole.
croppingThresh	minimum number of non-NA pixels that need to be in a row/column for it to not be cropped out of the breech face scan exterior
agg_function	the breech face radius estimation procedure returns a number of radius estimates. This argument dictates the function used to aggregate these into a final estimate.
scheme	argument for imager::imgradient
high_connectivity	argument for imager::label
tolerance	argument for imager::label

Value

An x3p object containing the surface matrix of a breech face impression scan where the observations on the exterior/interior of the breech face scan surface.

Note

The radius estimation procedure tends to over-estimate the desired radius values. As such, a lot of the breech face impression "roll-off" is included in the final scan. Excessive roll-off can bias the calculation of the CCF. As such, we can manually shrink the radius estimate (-30 or -30 seems to work well for the Fadul cartridge cases) so that little to no roll-off is included in the final processed scan.

The radius estimation procedure is effective at estimating the radius of the firing pin hole. Unfortunately, it is often desired that more than just observations in firing pin hole are removed. In particular, the plateaued region surrounding the firing pin impression hole does not come into contact

with the breech face of a firearm and is thus unwanted in the final, processed scan. The radiusOffset argument must be tuned (around 200 seems to work well for the Fadul cartridge cases) to remove these unwanted observations.

Examples

```
#Process fadul1.1 "from scratch" (takes > 5 seconds to run)
## Not run:
nbtrd_link <- "https://tsapps.nist.gov/NRBTD/Studies/CartridgeMeasurement/"
fadul1.1_link <- "DownloadMeasurement/2d9cc51f-6f66-40a0-973a-a9292dbec36d"

fadul1.1 <- x3ptools::read_x3p(paste0(nbtrd_link,fadul1.1_link))

fadul1.1_extCropped <- preProcess_crop(x3p = fadul1.1,
                                     radiusOffset = -30,
                                     region = "exterior")

fadul1.1_extIntCropped <- preProcess_crop(x3p = fadul1.1_extCropped,
                                          radiusOffset = 200,
                                          region = "interior")

x3pListPlot(list("Original" = fadul1.1,
                "Exterior Cropped" = fadul1.1_extCropped,
                "Exterior & Interior Cropped" = fadul1.1_extIntCropped ))

## End(Not run)
```

```
preProcess_gaussFilter
```

Performs a low, high, or bandpass Gaussian filter on a surface matrix with a particular cut-off wavelength.

Description

Performs a low, high, or bandpass Gaussian filter on a surface matrix with a particular cut-off wavelength.

Usage

```
preProcess_gaussFilter(x3p, wavelength = c(16, 500), filtertype = "bp")
```

Arguments

x3p	an x3p object containing a surface matrix
wavelength	cut-off wavelength
filtertype	specifies whether a low pass, "lp", high pass, "hp", or bandpass, "bp" filter is to be used. Note that setting filtertype = "bp" means that wavelength should be a vector of two numbers. In this case, the max of these two number will be used for the high pass filter and the min for the low pass filter.

Value

An x3p object containing the Gaussian-filtered surface matrix.

See Also

<https://www.mathworks.com/matlabcentral/fileexchange/61003-filt2-2d-geospatial-data-filter?focused=7181587&tab=exam>

Examples

```
data(fadul1.1_processed)

#Applying the function to fadul1.1_processed (note that this scan has already
# been Gaussian filtered)
cmCR::preProcess_gaussFilter(fadul1.1_processed)

#As a part of the recommended preprocessing pipeline (take > 5 sec to run):
## Not run:
nbtrd_link <- "https://tsapps.nist.gov/NRBD/Studies/CartridgeMeasurement/"
fadul1.1_link <- "DownloadMeasurement/2d9cc51f-6f66-40a0-973a-a9292dbec36d"

fadul1.1 <- x3ptools::read_x3p(paste0(nbtrd_link,fadul1.1_link))
fadul1.1_extCropped <- preProcess_crop(x3p = fadul1.1,
                                     region = "exterior",
                                     radiusOffset = -30)

fadul1.1_intCropped <- preProcess_crop(x3p = fadul1.1_extCropped,
                                     region = "interior",
                                     radiusOffset = 200)

fadul1.1_levelled <- preProcess_removeTrend(x3p = fadul1.1_intCropped,
                                           statistic = "quantile",
                                           tau = .5,
                                           method = "fn")
fadul1.1_filtered <- preProcess_gaussFilter(x3p = fadul1.1_levelled,
                                           wavelength = c(16,500),
                                           filtertype = "bp")

x3pListPlot(list("Original" = fadul1.1,
                "Ext. & Int. Cropped" = fadul1.1_intCropped,
                "Cropped and Levelled" = fadul1.1_levelled,
                "Filtered" = fadul1.1_filtered),type = "list")

## End(Not run)
```

```
preProcess_ransacLevel
```

Finds plane of breachface marks using the RANSAC method

Description

Finds plane of breechface marks using the RANSAC method

Usage

```
preProcess_ransacLevel(  
  x3p,  
  ransacInlierThresh = 1e-06,  
  ransacFinalSelectThresh = 2e-05,  
  iters = 300,  
  returnResiduals = TRUE  
)
```

Arguments

<code>x3p</code>	an <code>x3p</code> object containing a surface matrix
<code>ransacInlierThresh</code>	threshold to declare an observed value close to the fitted plane an "inlier". A smaller value will yield a more stable estimate.
<code>ransacFinalSelectThresh</code>	once the RANSAC plane is fitted based on the <code>ransacInlierThresh</code> , this argument dictates which observations are selected as the final breech face estimate.
<code>iters</code>	number of candidate planes to fit (higher value yields more stable breech face estimate)
<code>returnResiduals</code>	dictates whether the difference between the estimated breech face and fitted plane are returned (residuals) or if the estimates breech face is simply shifted down by its mean value

Value

an `x3p` object containing the leveled surface matrix.

Note

Given input depths (in microns), find best-fitting plane using RANSAC. This should be the plane that the breechface marks are on. Adapted from `cartridges3D::findPlaneRansac` function. This a modified version of the `findPlaneRansac` function available in the `cartridges3D` package on GitHub.

The `preProcess_ransacLevel` function will throw an error if the final plane estimate is rank-deficient (which is relatively unlikely, but theoretically possible). Re-run the function (possibly setting a different seed) if this occurs.

See Also

<https://github.com/xhtai/cartridges3D>

Examples

```
## Not run:
nbtrd_link <- "https://tsapps.nist.gov/NRBTD/Studies/CartridgeMeasurement/"
fadul1.1_link <- "DownloadMeasurement/2d9cc51f-6f66-40a0-973a-a9292dbec36d"

fadul1.1 <- x3ptools::read_x3p(paste0(nbtrd_link, fadul1.1_link))

fadul1.1_ransacLeveled <- fadul1.1 %>%
  preProcess_crop(region = "exterior",
                  radiusOffset = -30) %>%
  preProcess_crop(region = "interior",
                  radiusOffset = 200) %>%
  preProcess_removeTrend(statistic = "quantile",
                        tau = .5,
                        method = "fn")

x3pListPlot(list("Original" = fadul1.1,
                 "RANSAC Leveled" = fadul1.1_ransacLeveled), type = "list")

## End(Not run)
```

```
preProcess_removeFPCircle
```

Given a surface matrix, estimates and filters any pixels within the estimated firing pin impression circle

Description

Given a surface matrix, estimates and filters any pixels within the estimated firing pin impression circle

Usage

```
preProcess_removeFPCircle(
  x3p,
  aggregationFunction = mean,
  smootherSize = 2 * round((0.1 * nrow(surfaceMat)/2)) + 1,
  gridSize = 40,
  gridGranularity = 1,
  houghScoreQuant = 0.9
)
```

Arguments

x3p an x3p object containing a surface matrix

aggregationFunction function to select initial radius estimate from those calculated using fpRadius-GridSearch

smootherSize size of average smoother (to be passed to `zoo::roll_mean`)
gridSize size of grid, centered on the initial radius estimate, to be used to determine the best fitting circle to the surface matrix via the Hough transform method
gridGranularity granularity of radius grid used to determine the best fitting circle to the surface matrix via the Hough transform method
houghScoreQuant quantile cut-off to be used when determining a final radius estimate using the score values returned by the `imager::hough_circle`

Value

An `x3p` object containing a surface matrix with the estimated firing pin circle pixels replaced with NAs.

Note

`imager` treats a matrix as its transpose (i.e., x and y axes are swapped). As such, relative to the original surface matrix, the x and y columns in the data frame `fpImpressionCircle` actually correspond to the row and column indices at which the center of the firing pin impression circle is estimated to be.

Examples

```

## Not run:
nbtrd_link <- "https://tsapps.nist.gov/NRBTD/Studies/CartridgeMeasurement/"
fadul1.1_link <- "DownloadMeasurement/2d9cc51f-6f66-40a0-973a-a9292dbec36d"

fadul1.1 <- x3ptools::read_x3p(paste0(nbtrd_link, fadul1.1_link))

fadul1.1_labelCropped <- fadul1.1 %>%
  preProcess_crop(region = "exterior",
                  radiusOffset = -30) %>%
  preProcess_crop(region = "interior",
                  radiusOffset = 200) %>%
  preProcess_removeTrend(statistic = "quantile",
                          tau = .5,
                          method = "fn")

fadul1.1_houghCropped <- fadul1.1 %>%
  x3ptools::x3p_sample() %>%
  preProcess_ransacLevel() %>%
  preProcess_crop(region = "exterior",
                  radiusOffset = -30) %>%
  preProcess_removeFPCircle()

x3pListPlot(list("Original" = fadul1.1,
                 "Cropped by Labeling" = fadul1.1_labelCropped,
                 "Cropped by Hough" = fadul1.1_houghCropped), type = "list")

## End(Not run)

```

```
preProcess_removeTrend
```

Level a breech face impression surface matrix by a conditional statistic

Description

Level a breech face impression surface matrix by a conditional statistic

Usage

```
preProcess_removeTrend(x3p, statistic = "mean", ...)
```

Arguments

x3p	an x3p object containing the surface matrix of a cartridge case scan
statistic	either "mean" or "quantile"
...	arguments to be set in the <code>quantreg::rq</code> function if <code>statistic = "quantile"</code> is set. In this case, <code>tau = .5</code> and <code>method = "fn"</code> are recommended

Value

an x3p object containing the leveled cartridge case scan surface matrix.

Examples

```
#Process fadul1.1 "from scratch" (takes > 5 seconds to run)
## Not run:
nbtrd_link <- "https://tsapps.nist.gov/NRBD/Studies/CartridgeMeasurement/"
fadul1.1_link <- "DownloadMeasurement/2d9cc51f-6f66-40a0-973a-a9292dbec36d"

fadul1.1 <- x3ptools::read_x3p(paste0(nbtrd_link, fadul1.1_link))
fadul1.1_extCropped <- preProcess_crop(x3p = fadul1.1,
                                     region = "exterior",
                                     radiusOffset = -30)

fadul1.1_intCropped <- preProcess_crop(x3p = fadul1.1_extCropped,
                                     region = "interior",
                                     radiusOffset = 200)

fadul1.1_leveled <- preProcess_removeTrend(x3p = fadul1.1_intCropped,
                                          statistic = "quantile",
                                          tau = .5,
                                          method = "fn")

x3pListPlot(list("Original" = fadul1.1,
                "Ext. Cropped" = fadul1.1_extCropped,
                "Ext. & Int. Cropped" = fadul1.1_intCropped,
                "Cropped and Leveled" = fadul1.1_leveled))
```

```
## End(Not run)
```

x3pListPlot

Plot a list of x3ps

Description

Plots the surface matrices in a list of x3p objects. Either creates one plot faceted by surface matrix or creates individual plots per surface matrix and returns them in a list.

Usage

```
x3pListPlot(
  x3pList,
  type = "faceted",
  rotate = 0,
  legend.quantiles = c(0, 0.01, 0.25, 0.5, 0.75, 0.99, 1),
  height.colors = rev(c("#7f3b08", "#b35806", "#e08214", "#fdb863", "#fee0b6",
    "#f7f7f7", "#d8daeb", "#b2abd2", "#8073ac", "#542788", "#2d004b")),
  na.value = "gray80",
  guide = "colorbar"
)
```

Arguments

x3pList	a list of x3p objects. If the x3p objects are named in the list, then these names will be included in the title of their respective plot
type	dictates whether one plot faceted by surface matrix or a list of plots per surface matrix is returned. The faceted plot will have a consistent height scale across all surface matrices.
rotate	angle (in degrees) to rotate all surface matrices plotted
legend.quantiles	vector of quantiles to be shown as tick marks on legend plot
height.colors	vector of colors to be passed to <code>scale_fill_gradientn</code> that dictates the height value colorscale
na.value	color to be used for NA values (passed to <code>scale_fill_gradientn</code>)
guide	internal usage

Value

A ggplot object or list of ggplot objects showing the surface matrix height values.

Examples

```
data(fadul1.1_processed, fadul1.2_processed)

x3pListPlot(list("Fadul 1-1" = fadul1.1_processed,
                 "Fadul 1-2" = fadul1.2_processed))
```

Index

* datasets

- fadulData_processed, [19](#)

- cmcPlot, [2](#)
- comparison_allTogether, [4](#)
- comparison_calcPropMissing, [5](#)
- comparison_cellDivision, [6](#)
- comparison_cor, [7](#)
- comparison_fft_ccf, [8](#)
- comparison_getTargetRegions, [9](#)
- comparison_replaceMissing, [9](#), [10](#)
- comparison_standardizeHeights, [11](#)

- decision_CMC, [13](#)
- decision_combinedDirections, [15](#)
- decision_highCMC_cmcThetaDistrib, [17](#)
- decision_highCMC_identifyHighCMCThetas, [18](#)

- fadul1.1_processed
(fadulData_processed), [19](#)
- fadul1.2_processed
(fadulData_processed), [19](#)
- fadulData_processed, [19](#)

- preProcess_crop, [20](#)
- preProcess_gaussFilter, [22](#)
- preProcess_ransacLevel, [23](#)
- preProcess_removeFPcircle, [25](#)
- preProcess_removeTrend, [27](#)

- x3pListPlot, [28](#)