Package 'patternize'

July 23, 2025

Title Quantification of Color Pattern Variation

Version 0.0.5

Maintainer Steven Van Belleghem <vanbelleghemsteven@hotmail.com>

Description Quantification of variation in organismal color patterns as obtained from image data. Patternize defines homology between pattern positions across images either through fixed landmarks or image registration. Pattern identification is performed by categorizing the distribution of colors using RGB thresholds or image segmentation.

BugReports https://github.com/StevenVB12/patternize/issues

URL https://github.com/StevenVB12/patternize

Depends R (>= 3.5.0)

Imports raster, sp, sf, abind, Morpho, dplyr, imager, magrittr, methods, purrr, vegan, RNiftyReg, geomorph, ClusterR

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

NeedsCompilation no

Author Steven Van Belleghem [aut, cre]

Repository CRAN

Date/Publication 2023-08-22 14:40:05 UTC

Contents

alignLan *Align images using landmarks*

Description

Align images using landmarks

Usage

```
alignLan(
  imageList,
  landList,
  IDlist = NULL,
  adjustCoords = FALSE,
  resampleFactor = NULL,
  res = c(300, 300),
  transformRef = "meanshape",
  transformType = "tps",
  maskOutline = NULL,
  removebg = NULL,
  removebgColOffset = 0.1,
  inverse = FALSE,
  cartoonID = NULL,
  refImage = NULL,
  plotTransformed = FALSE,
  format = "imageJ"
)
```

Arguments

imageList	List of RasterStack objects.
landList	Landmark list as returned by makeList.
IDlist	List of sample IDs should be specified when masking outline and transformRef is 'meanshape'.
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).
resampleFactor	Integer for downsampling used by redRes.
res	Resolution vector c(x,y) for output rasters (default = c(300,300)). This should be reduced if the number of pixels in the image is lower than th raster.
transformRef	ID or landmark matrix of reference sample for shape to which color patterns will be transformed to. Can be 'meanshape' for transforming to mean shape of Procrustes analysis.
transformType	Transformation type as used by computeTransform (default ='tps').
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL). This can be a list of multiple outlines.

removebg	Integer or RGB vector indicating the range of RGB threshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL).
removebgColOffset	Color offset for color background extraction (default = 0.10).
inverse	If TRUE, areas withing the outline will be masked. If maskOutline is a list, this should also be a list.
cartoonID	ID of the sample for which the cartoon was drawn and will be used for masking (should be set when transformRef = 'meanShape').
refImage	Image (RasterStack) used for target. Use raster::stack('filename').
plotTransformed	Plot transformed image (default = FALSE).
format	ImageJ (Fiji) or tps format (default = 'imageJ').

Value

List of aligned RasterStack objects.

---

alignReg *Aligns images using* niftyreg *utilities for automated image registration..*

Description

Aligns images using niftyreg utilities for automated image registration..

Usage

```
alignReg(
  sampleList,
  target,
  resampleFactor = NULL,
  useBlockPercentage = 75,
  crop = c(0, 0, 0, 0),
  removebgR = NULL,
  maskOutline = NULL,
  plotTransformed = FALSE
)
```

Arguments

sampleList	List of RasterStack objects.
target	Image imported as RasterStack used as target for registration.
resampleFactor	Integer for downsampling used by redRes (default = NULL).
useBlockPercentage	Block percentage as used in niftyreg (default = 75).

crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image.
removebgR	Integer indicating the range RGB treshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL) for registration analysis. This works only to remove a white background.
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
plotTransformed	Whether to plot transformed images while processing (default = FALSE).

Value

List of raster objects.

`colorChecker` *Calibrate images using ColorChecker.*

Description

Calibrate images using ColorChecker.

Usage

```
colorChecker(
  IDlist,
  prepath = NULL,
  extension = NULL,
  colorCheckerType = "X-Rite",
  fixedCorners = FALSE,
  patchSize = 0.6,
  colorCheckerXY = NULL,
  fixedModel = NULL,
  resampleFactor = NULL,
  adjustCoords = FALSE
)
```

Arguments

IDlist	List of sample IDs.
prepath	Prepath (default = NULL).
extension	Extension (default = NULL).
colorCheckerType	Type of colorChecker. Options are 'X-Rite ' and 'ColorGauge Micro Analyzer' (default = 'X-Rite ').
fixedCorners	Specify whether to set the coordinates of the colorChecker corners for every image (default = FALSE).

patchSize	Proportion of ColorChecker patch that will be used for observed RGB values (default = 0.6).
colorCheckerXY	Landmark list of colorChecker corners as returned by makeList. The image will not be plotted.
fixedModel	Precalculated model to adjust colors. Should be a listof a model for R, G and B (the colorChecker function gives as output such a list obtained from the last image in the analysis).
resampleFactor	Integer for downsampling used by redRes.
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).

Value

Calibrated image(s) ('filename_calibrated.jpg')

`colorChecker_customGray` *Calibrate images using ColorChecker.*

Description

Calibrate images using ColorChecker.

Usage

```
colorChecker_customGray(
  IDlist,
  prepath = NULL,
  extension = NULL,
  colorCheckerType = "X-Rite",
  fixedCorners = FALSE,
  patchSize = 0.6,
  colorCheckerXY = NULL,
  fixedModel = NULL,
  resampleFactor = NULL
)
```

Arguments

IDlist	List of sample IDs.
prepath	Prepath (default = NULL).
extension	Extension (default = NULL).
colorCheckerType	Type of colorChecker. Options are 'X-Rite ' and 'ColorGauge Micro Analyzer' (default = 'X-Rite ').

fixedCorners	Specify whether to set the coordinates of the colorChecker corners for every image (default = FALSE).
patchSize	Proportion of ColorChecker patch that will be used for observed RGB values (default = 0.6).
colorCheckerXY	Landmark list of colorChecker corners as returned by makeList. The image will not be plotted.
fixedModel	Precalculated model to adjust colors. Should be a listof a model for R, G and B (the colorChecker function gives as output such a list obtained from the last image in the analysis).
resampleFactor	Integer for downsampling used by redRes.

Value

Calibrated image(s) ('filename_calibrated.jpg')

`colorChecker_half` *Calibrate images using (right) half of ColorChecker. Only works for X-Rite.*

Description

Calibrate images using (right) half of ColorChecker. Only works for X-Rite.

Usage

```
colorChecker_half(
  IDlist,
  prepath = NULL,
  extension = NULL,
  colorCheckerType = "X-Rite",
  fixedCorners = FALSE,
  patchSize = 0.6,
  colorCheckerXY = NULL,
  fixedModel = NULL,
  resampleFactor = NULL,
  adjustCoords = FALSE
)
```

Arguments

IDlist	List of sample IDs.
prepath	Prepath (default = NULL).
extension	Extension (default = NULL).
colorCheckerType	Type of colorChecker. Options are 'X-Rite ' and 'ColorGauge Micro Analyzer' (default = 'X-Rite').

fixedCorners	Specify whether to set the coordinates of the colorChecker corners for every image (default = FALSE).
patchSize	Proportion of ColorChecker patch that will be used for observed RGB values (default = 0.6).
colorCheckerXY	Landmark list of colorChecker corners as returned by makeList. The image will not be plotted.
fixedModel	Precalculated model to adjust colors. Should be a listof a model for R, G and B (the colorChecker function gives as output such a list obtained from the last image in the analysis).
resampleFactor	Integer for downsampling used by redRes.
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).

Value

Calibrated image(s) ('filename_calibrated.jpg')

---

createPhenotype *Plot color pattern prediction for specified PCA values*

Description

Plot color pattern prediction for specified PCA values

Usage

```
createPhenotype(
  PCAdata,
  PCApredict,
  IDlist,
  rasterList,
  colpalette = NULL,
  plotCartoon = FALSE,
  refShape = NULL,
  outline = NULL,
  lines = NULL,
  landList = NULL,
  adjustCoords = FALSE,
  cartoonID = NULL,
  normalized = TRUE,
  crop = c(0, 0, 0, 0),
  flipRaster = NULL,
  flipOutline = NULL,
  imageList = NULL,
  cartoonOrder = "above",
  lineOrder = "above",
```

```
  cartoonCol = "gray",
  cartoonFill = NULL,
  legendTitle = "Proportion",
  zlim = NULL
)
```

Arguments

PCAdata	Output of PCA analysis. List item 3 of patPCA.
PCApredict	A vector with the PCA values for which to predict the phenotype. This vector only needs to include the values upto the last PCA axis to predict along, other values are set to zero.
IDlist	List of sample IDs.
rasterList	rasterList used for PCA.
colpalette	Vector of colors for color palette (default = c("white","lightblue","blue","green", "yellow","red"))
plotCartoon	Whether to plot a cartoon. This cartoon should be drawn on one of the samples used in the analysis.
refShape	This can be 'target' in case the reference shape is a single sample (for registration analysis) or 'mean' if the images were transformed to a mean shape (only for meanshape when using landmark transformation)
outline	xy coordinates that define outline.
lines	list of files with xy coordinates of line objects to be added to cartoon.
landList	Landmark landmarkList.
adjustCoords	Adjust landmark coordinates.
cartoonID	ID of the sample for which the cartoon was drawn.
normalized	Set this to true in case the summed rasters are already devided by the sample number.
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image used in landmark or registration analysis.
flipRaster	Whether to flip raster along xy axis (in case there is an inconsistency between raster and outline coordinates).
flipOutline	Whether to flip plot along x, y or xy axis.
imageList	List of images should be given if one wants to flip the outline or adjust landmark coordinates.
cartoonOrder	Whether to plot the cartoon outline 'above' or 'under' the pattern raster (default = 'above'). Set to 'under' for filled outlines.
lineOrder	Whether to plot the cartoon lines 'above' or 'under' the pattern raster (default = 'above').
cartoonCol	Outline and line color for cartoon (deafault = 'gray').
cartoonFill	Fill color for outline of cartoon (default = NULL).
legendTitle	Title of the raster legend (default = 'Proportion').
zlim	zlim values for predicted pattern.

createTarget *Create a target image (RasterStack) from a polygon.*

Description

Create a target image (RasterStack) from a polygon.

Usage

```
createTarget(
  outline,
  image,
  res = 300,
  colorFill = "black",
  colorBG = "white",
  sigma = 10,
  plot = FALSE
)
```

Arguments

outline	xy coordinates that define outline.
image	Image imported as RasterStack used in the analysis. This is used to extract the extent and dimensions for the raster layers.
res	Resolution for RasterStack (default = 300).
colorFill	Color for the fill of the polygon (default = 'black').
colorBG	Color for the background (default = 'white').
sigma	Size of sigma for Gaussian blurring (default = 10).
plot	Whether to plot the created target image (default = FALSE).

Value

RasterStack

Examples

```
## Not run:
outline_BC0077 <- read.table(paste(system.file("extdata",  package = 'patternize'),
'/BC0077_outline.txt', sep=''), header = FALSE)

data(imageList)

target <- createTarget(outline_BC0077, imageList[[1]], plot =  TRUE)

## End(Not run)
```

`extdata` *External patternize data*

Description

Raw image, landmark and cartoon data of Heliconius erato hydara wings.

Format

Raw JPG images, landmark and cartoon data.

BC0077.JPG jpeg image

BC0071.JPG jpeg image

BC0050.JPG jpeg image

BC0049.JPG jpeg image

BC0004.JPG jpeg image

BC0077_landmarks_LFW.Txt xy landmark coordinates

BC0071_landmarks_LFW.Txt xy landmark coordinates

BC0050_landmarks_LFW.Txt xy landmark coordinates

BC0049_landmarks_LFW.Txt xy landmark coordinates

BC0004_landmarks_LFW.Txt xy landmark coordinates

BC0077_outline.txt xy outline coordinates

BC0077_vein1.txt xy vein coordinates

BC0077_vein2.txt xy vein coordinates

BC0077_vein3.txt xy vein coordinates

BC0077_vein4.txt xy vein coordinates

BC0077_vein5.txt xy vein coordinates

BC0077_vein6.txt xy vein coordinates

BC0077_vein7.txt xy vein coordinates

BC0077_vein8.txt xy vein coordinates

BC0077_vein9.txt xy vein coordinates

BC0077_vein10.txt xy vein coordinates

BC0077_vein11.txt xy vein coordinates

GMMImage *GMM clustering of image imported as a RasterStack.*

Description

GMM clustering of image imported as a RasterStack.

Usage

```
GMMImage(image, k = 5, maskToNA = NULL, kmeansOnAll = FALSE)
```

Arguments

image	Image imported as a RasterStack for clustering.
k	Integer for number of k clusters (default = 3).
maskToNA	Replace the color value used for masking (i.e. 0 or 255) with NA.
kmeansOnAll	Whether to perform the kmeans clusters on the combined set of pixels of all images first (default = FALSE).

Value

List including the clustered RasterSatck returned as an array and object of class "GMM".

imageList *imageList*

Description

List of RasterStacks as returned by makeList.

Usage

```
imageList
```

Format

A list of 5 RasterStack objects of Heliconius erato hydara dorsal forewings.

Examples

```
## Not run:
data(imageList)
summary(imageList)

## End(Not run)
```

kImage — *kmeans clustering of image imported as a RasterStack. This function is used by* patLanK *and* patRegK.

Description

kmeans clustering of image imported as a RasterStack. This function is used by patLanK and patRegK.

Usage

```
kImage(image, k = 5, startCenter = NULL, maskToNA = NULL, kmeansOnAll = FALSE)
```

Arguments

image	Image imported as a RasterStack for k-means clustering.
k	Integer for number of k-means clusters (default = 3).
startCenter	A matrix of cluster centres to start k-means clustering from (default = NULL).
maskToNA	Replace the color value used for masking (i.e. 0 or 255) with NA.
kmeansOnAll	Whether to perform the kmeans clusters on the combined set of pixels of all images first (default = FALSE).

Value

List including the k-means clustered RasterSatck returned as an array and object of class "kmeans".

Examples

```
image <- raster::stack(system.file("extdata", "BC0077.jpg", package = "patternize"))
out <- kImage(image, 6)
```

kImageHSV — *kmeans clustering of image imported as a RasterStack. This function is used by* patLanK *and* patRegK.

Description

kmeans clustering of image imported as a RasterStack. This function is used by patLanK and patRegK.

Usage

```
kImageHSV(
  image,
  k = 5,
  startCenter = NULL,
  maskToNA = NULL,
  kmeansOnAll = FALSE,
  ignoreHSVvalue = FALSE
)
```

Arguments

image	HSV image imported as a RasterStack for k-means clustering.
k	Integer for number of k-means clusters (default = 3).
startCenter	A matrix of cluster centres to start k-means clustering from (default = NULL).
maskToNA	Replace the color value used for masking (i.e. 0 or 255) with NA.
kmeansOnAll	Whether to perform the kmeans clusters on the combined set of pixels of all images first (default = FALSE).
ignoreHSVvalue	Whether to ignore the HSV value (~darkness).

Value

List including the k-means clustered RasterSatck returned as an array and object of class "kmeans".

Examples

```
image <- raster::stack(system.file("extdata", "BC0077.jpg", package = "patternize"))
out <- kImage(image, 6)
```

---

lanArray *Build landmark array for* Morpho.

Description

Build landmark array for Morpho.

Usage

```
lanArray(sampleList, adjustCoords = FALSE, imageList = NULL, imageIDs = NULL)
```

Arguments

sampleList	List of landmark matrices as returned by makeList.
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).
imageList	List of RasterStacks as returned by makeList should be given when adjustCoords = TRUE.
imageIDs	A list of IDs to match landmarks to images if landmarkList and imageList don't have the same length.

Value

X x Y x n array, where X and Y define the coordinates of the landmark points and n is the sample size.

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')

prepath <- system.file("extdata",  package = 'patternize')
extension <- '_landmarks_LFW.txt'

landmarkList <- makeList(IDlist, 'landmark', prepath, extension)

landmarkArray <- lanArray(landmarkList)

## End(Not run)
```

---

landmarkArray *landmarkArray*

Description

Array of landmarks as returned by lanArray and used by link[Morpho]{procsym}.

Usage

```
landmarkArray
```

Format

An array of landmarks for 5 Heliconius erato hydara dorsal forewings.

Examples

```
## Not run:
data(landmarkArray)
summary(landmarkArray)

## End(Not run)
```

`landmarkList` *landmarkList*

Description

List of landmarks as returned by `makeList`.

Usage

```
landmarkList
```

Format

A list of landmarks for 5 Heliconius erato hydara dorsal forewings.

Examples

```
## Not run:
data(landmarkList)
summary(landmarkList)

## End(Not run)
```

`makeList` *Build list of landmarks or RasterStacks from images using filepath and file extension.*

Description

Build list of landmarks or RasterStacks from images using filepath and file extension.

Usage

```
makeList(
  IDlist,
  type,
  prepath = NULL,
  extension = NULL,
  format = "imageJ",
  tpsFile = NULL,
  skipLandmark = NULL
)
```

Arguments

IDlist	List of sample IDs.
type	'landmark' or 'image' depending on what type of list to make.
prepath	Prepath (default = NULL).
extension	Extension (default = NULL).
format	ImageJ (Fiji) or tps format (default = 'imageJ').
tpsFile	Provide filename of tps file ff format is 'tps'.
skipLandmark	Vector of rownumbers of landmarks to skip.

Value

Landmark or RasterStack list.

Examples

```
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')

prepath <- system.file("extdata",  package = 'patternize')
extension <- '_landmarks_LFW.txt'

landmarkList <- makeList(IDlist, 'landmark', prepath, extension)

extension <- '.jpg'
imageList <- makeList(IDlist, 'image', prepath, extension)
```

---

maskOutline *Intersects a RasterStack with an outline. Everything outside of the outline will be removed from the raster.*

Description

Intersects a RasterStack with an outline. Everything outside of the outline will be removed from the raster.

Usage

```
maskOutline(
  RasterStack,
  outline,
  refShape,
  landList = NULL,
  adjustCoords = FALSE,
  cartoonID = NULL,
  IDlist = NULL,
  crop = c(0, 0, 0, 0),
```

```
  flipRaster = NULL,
  flipOutline = NULL,
  imageList = NULL,
  maskColor = 0,
  inverse = FALSE
)
```

Arguments

RasterStack	RasterStack to be masked.
outline	xy coordinates that define outline.
refShape	This can be 'target' in case the reference shape is a single sample (for registration analysis) or 'mean' if the images were transformed to a mean shape (only for meanshape when using landmark transformation)
landList	Landmark list to be given when type = 'mean'.
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).
cartoonID	ID of the sample for which the cartoon was drawn. Only has to be given when refShape is 'mean'.
IDlist	List of sample IDs should be specified when refShape is 'mean'.
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image used in landmark or registration analysis.
flipRaster	Whether to flip raster along xy axis (in case there is an inconsistency between raster and outline coordinates).
flipOutline	Whether to flip plot along x, y or xy axis.
imageList	List of image as obtained from makeList should be given if one wants to flip the outline or adjust landmark coordinates.
maskColor	Color the masked area gets. Set to 0 for black (default) or 255 for white.
inverse	If TRUE, areas withing the outline will be masked.

Examples

```
## Not run:
data(imageList)
outline_BC0077 <- read.table(paste(system.file("extdata",  package = 'patternize'),
'/BC0077_outline.txt', sep=''), header = FALSE)

masked <- maskOutline(imageList[[1]], outline_BC0077, refShape = 'target', flipOutline = 'y')

## End(Not run)
```

patArea *This fucntion calculates the area in which the color pattern is expressed in each sample as the relative proportion using the provided outline of the considered trait or structure.*

Description

This fucntion calculates the area in which the color pattern is expressed in each sample as the relative proportion using the provided outline of the considered trait or structure.

Usage

```
patArea(
  rList,
  IDlist,
  refShape,
  type,
  outline = NULL,
  landList = NULL,
  adjustCoords = FALSE,
  cartoonID = NULL,
  crop = c(0, 0, 0, 0),
  flipRaster = NULL,
  flipOutline = NULL,
  imageList = NULL
)
```

Arguments

rList	List of RasterLayers as obtained from the main patternize functions.
IDlist	List of sample IDs.
refShape	This can be 'target' in case the reference shape is a single sample (for registration analysis) or 'mean' if the images were transformed to a mean shape using landmark transformation.
type	Type of rasterlist; 'RGB' or 'k' (result from RGB or k-means analysis, respectively).
outline	xy coordinates that define outline.
landList	Landmark list as returned by makeList.
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).
cartoonID	ID of the sample for which the cartoon was drawn.
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image used in landmark or registration analysis.
flipRaster	Whether to flip raster along xy axis (in case there is an inconsistency between raster and outline coordinates).

flipOutline	Whether to flip plot along x, y or xy axis.
imageList	List of images as obtained from makeList should be given if one wants to flip the outline or adjust landmark coordinates.

Value

Table or list of tables with sample IDs and relative area of color pattern or kmeans cluster.

Examples

```
data(rasterList_lanRGB)
#data(rasterList_regRGB)
#data(rasterList_lanK)
#data(rasterList_regK)

data(imageList)

IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')

outline_BC0077 <- read.table(paste(system.file("extdata",  package = 'patternize'),
'/BC0077_outline.txt', sep=''), header = FALSE)

prepath <- system.file("extdata", package = 'patternize')
extension <- '_landmarks_LFW.txt'

landmarkList <- makeList(IDlist, 'landmark', prepath, extension)

## Not run:
area_lanRGB <- patArea(rasterList_lanRGB, IDlist, refShape = 'mean', type = 'RGB',
outline = outline_BC0077, landList = landmarkList, adjustCoords = TRUE,
imageList = imageList, cartoonID = 'BC0077')

area_regRGB <- patArea(rasterList_regRGB, IDlist, refShape = 'target', type = 'RGB',
outline = outline_BC0077, crop = c(100,400,40,250), adjustCoords = TRUE,
imageList = imageList, cartoonID = 'BC0077', flipRaster = 'xy')

areaList_lanK <- patArea(rasterList_lanK, IDlist, refShape = 'mean', type = 'k',
outline = outline_BC0077, landList = landmarkList, adjustCoords = TRUE,
imageList = imageList, cartoonID = 'BC0077')

areaList_regK <- patArea(rasterList_regK, IDlist, refShape = 'target', type = 'k',
outline = outline_BC0077, crop = c(100,400,40,250), adjustCoords = TRUE,
imageList = imageList, cartoonID = 'BC0077', flipRaster = 'xy')

## End(Not run)
```

patGMM *Extract colors using GMM clustering (for pre-aligned images).*

Description

Extract colors using GMM clustering (for pre-aligned images).

Usage

```
patGMM(
  sampleList,
  k = 3,
  resampleFactor = NULL,
  maskOutline = NULL,
  plot = FALSE,
  focal = FALSE,
  sigma = 3,
  maskToNA = NULL,
  kmeansOnAll = FALSE
)
```

Arguments

sampleList	List of RasterStack objects.
k	Integere for defining number of clusters (default = 3).
resampleFactor	Integer for downsampling used by redRes.
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
plot	Whether to plot transformed color patterns while processing (default = FALSE).
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).
maskToNA	Replace the color value used for masking (i.e. 0 or 255) with NA.
kmeansOnAll	Whether to perform the kmeans clusters on the combined set of pixels of all images first (default = FALSE).

Value

List of summed raster for each k-means cluster objects.

patK *Extract colors using k-means clustering (for pre-aligned images).*

Description

Extract colors using k-means clustering (for pre-aligned images).

Usage

```
patK(
  sampleList,
  k = 3,
  fixedStartCenter = NULL,
  resampleFactor = NULL,
  maskOutline = NULL,
  plot = FALSE,
  focal = FALSE,
  sigma = 3,
  maskToNA = NULL,
  kmeansOnAll = FALSE
)
```

Arguments

sampleList	List of RasterStack objects.
k	Integere for defining number of k-means clusters (default = 3).
fixedStartCenter	Specify a dataframe with start centers for k-means clustering.
resampleFactor	Integer for downsampling used by redRes.
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
plot	Whether to plot transformed color patterns while processing (default = FALSE).
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).
maskToNA	Replace the color value used for masking (i.e. 0 or 255) with NA.
kmeansOnAll	Whether to perform the kmeans clusters on the combined set of pixels of all images first (default = FALSE).

Value

List of summed raster for each k-means cluster objects.

patK_HSV *Extract colors using k-means clustering (for pre-aligned images).*

Description

Extract colors using k-means clustering (for pre-aligned images).

Usage

```
patK_HSV(
  sampleList,
  k = 3,
  fixedStartCenter = NULL,
  resampleFactor = NULL,
  maskOutline = NULL,
  plot = FALSE,
  focal = FALSE,
  sigma = 3,
  maskToNA = NULL,
  kmeansOnAll = FALSE,
  ignoreHSVvalue = FALSE
)
```

Arguments

sampleList	List of RasterStack objects.
k	Integere for defining number of k-means clusters (default = 3).
fixedStartCenter	Specify a dataframe with start centers for k-means clustering.
resampleFactor	Integer for downsampling used by redRes.
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
plot	Whether to plot transformed color patterns while processing (default = FALSE).
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).
maskToNA	Replace the color value used for masking (i.e. 0 or 255) with NA.
kmeansOnAll	Whether to perform the kmeans clusters on the combined set of pixels of all images first (default = FALSE).
ignoreHSVvalue	Whether to ignore the HSV value (~darkness).

Value

List of summed raster for each k-means cluster objects.

patLanHSV *Aligns images usings transformations obtained from fixed landmarks and extracts colors using a predefined RGB values and cutoff value.*

Description

Aligns images usings transformations obtained from fixed landmarks and extracts colors using a predefined RGB values and cutoff value.

Usage

```
patLanHSV(
  sampleList,
  landList,
  HSV,
  resampleFactor = NULL,
  colOffset = 0.1,
  crop = FALSE,
  cropOffset = c(0, 0, 0, 0),
  res = 300,
  transformRef = "meanshape",
  transformType = "tps",
  adjustCoords = FALSE,
  plot = NULL,
  focal = FALSE,
  sigma = 3,
  iterations = 0,
  ignoreHSVvalue = FALSE,
  patternsToFile = NULL
)
```

Arguments

sampleList	List of RasterStack objects.
landList	Landmark list as returned by makeList.
HSV	HSV values for color pattern extraction specified as vector.
resampleFactor	Integer for downsampling used by redRes.
colOffset	Color offset for color pattern extraction (default = 0.10).
crop	Whether to use the landmarks range to crop the image. This can speed up the analysis (default = FALSE).
cropOffset	Vector c(xmin, xmax, ymin, ymax) that specifies the number of pixels you want the cropping to be offset from the landmarks (in case the landmarks do not surround the entire color pattern). The values specified should present the percentage of the maximum landmark value along the x and y axis.

res	Resolution for color pattern raster (default = 300). This should be reduced if the number of pixels in the image is lower than th raster.
transformRef	ID of reference sample for shape to which color patterns will be transformed to. Can be 'meanshape' for transforming to mean shape of Procrustes analysis.
transformType	Transformation type as used by computeTransform (default ='tps').
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).
plot	Whether to plot transformed color patterns while processing (default = NULL). Transformed color patterns can be plot on top of each other ('stack') or next to the original image for each sample ('compare').
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).
iterations	Number of iterations for recalculating average color.
ignoreHSVvalue	Whether to ignore the HSV value (~darkness).
patternsToFile	Name of directory to which the color pattern of each individual will be outputted (default = NULL).

Value

List of raster objects.

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '_landmarks_LFW.txt'

landmarkList <- makeList(IDlist, 'landmark', prepath, extension)

extension <- '.jpg'
imageList <- makeList(IDlist, 'image', prepath, extension)

HSV <- c(0.025,1,0.45)
rasterList_lanHSV <- patLanRGB(imageList, landmarkList, HSV,
colOffset = 0.15, crop = TRUE, res = 100, adjustCoords = TRUE, plot = 'stack')

## End(Not run)
```

---

patLanK *Aligns images usings transformations obtained from fixed landmarks and extracts colors using k-means clustering.*

Description

Aligns images usings transformations obtained from fixed landmarks and extracts colors using k-means clustering.

Usage

```
patLanK(
  sampleList,
  landList,
  k = 3,
  fixedStartCenter = NULL,
  resampleFactor = NULL,
  crop = FALSE,
  cropOffset = c(0, 0, 0, 0),
  res = 300,
  transformRef = "meanshape",
  transformType = "tps",
  removebg = NULL,
  removebgColOffset = 0.1,
  adjustCoords = FALSE,
  plot = FALSE,
  focal = FALSE,
  sigma = 3
)
```

Arguments

sampleList	List of RasterStack objects.
landList	Landmark list as returned by makeList.
k	Integere for defining number of k-means clusters (default = 3).
fixedStartCenter	Specify a dataframe with start centers for k-means clustering.
resampleFactor	Integer for downsampling used by redRes.
crop	Whether to use the landmarks range to crop the image. This can significantly speed up the analysis (default = FALSE).
cropOffset	Vector c(xmin, xmax, ymin, ymax) that specifies the number of pixels you want the cropping to be offset from the landmarks (in case the landmarks do not surround the entire color pattern). The values specified should present the percentage of the maximum landmark value along the x and y axis.
res	Resolution for color pattern raster (default = 300). This should be reduced if the number of pixels in the image is lower than th raster.
transformRef	ID of reference sample for shape to which color patterns will be transformed to. Can be 'meanshape' for transforming to mean shape of Procrustes analysis.
transformType	Transformation type as used by computeTransform (default ='tps').
removebg	Integer or RGB vector indicating the range of RGB threshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL).

removebgColOffset
: Color offset for color background extraction (default = 0.10).

adjustCoords
: Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).

plot
: Whether to plot transformed color patterns while processing (default = FALSE).

focal
: Whether to perform Gaussian blurring (default = FALSE).

sigma
: Size of sigma for Gaussian blurring (default = 3).

Value

List of summed raster for each k-means cluster objects.

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '_landmarks_LFW.txt'
landmarkList <- makeList(IDlist, 'landmark', prepath, extension)

extension <- '.jpg'
imageList <- makeList(IDlist, 'image', prepath, extension)
# Note that this example only aligns two images with the target,
# remove [1:2] to run a full examples.
rasterList_lanK <- patLanK(imageList[1:2], landmarkList[1:2], k = 4, crop = TRUE,
res = 100, removebg = 100, adjustCoords = TRUE, plot = TRUE)

## End(Not run)
```

---

patLanK_HSV *Aligns images usings transformations obtained from fixed landmarks and extracts colors using k-means clustering.*

Description

Aligns images usings transformations obtained from fixed landmarks and extracts colors using k-means clustering.

Usage

```
patLanK_HSV(
  sampleList,
  landList,
  k = 3,
  fixedStartCenter = NULL,
  resampleFactor = NULL,
  crop = FALSE,
```

```
  cropOffset = c(0, 0, 0, 0),
  res = 300,
  transformRef = "meanshape",
  transformType = "tps",
  removebgK = NULL,
  adjustCoords = FALSE,
  plot = FALSE,
  focal = FALSE,
  sigma = 3,
  ignoreHSVvalue = FALSE
)
```

Arguments

sampleList	List of RasterStack objects.
landList	Landmark list as returned by makeList.
k	Integere for defining number of k-means clusters (default = 3).
fixedStartCenter	Specify a dataframe with start centers for k-means clustering.
resampleFactor	Integer for downsampling used by redRes.
crop	Whether to use the landmarks range to crop the image. This can significantly speed up the analysis (default = FALSE).
cropOffset	Vector c(xmin, xmax, ymin, ymax) that specifies the number of pixels you want the cropping to be offset from the landmarks (in case the landmarks do not surround the entire color pattern). The values specified should present the percentage of the maximum landmark value along the x and y axis.
res	Resolution for color pattern raster (default = 300). This should be reduced if the number of pixels in the image is lower than th raster.
transformRef	ID of reference sample for shape to which color patterns will be transformed to. Can be 'meanshape' for transforming to mean shape of Procrustes analysis.
transformType	Transformation type as used by computeTransform (default ='tps').
removebgK	Integer indicating the range RGB treshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL) for k-means analysis. This works only to remove a white background.
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).
plot	Whether to plot transformed color patterns while processing (default = FALSE).
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).
ignoreHSVvalue	Whether to ignore the HSV value (~darkness).

Value

List of summed raster for each k-means cluster objects.

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '_landmarks_LFW.txt'
landmarkList <- makeList(IDlist, 'landmark', prepath, extension)

extension <- '.jpg'
imageList <- makeList(IDlist, 'image', prepath, extension)
# Note that this example only aligns two images with the target,
# remove [1:2] to run a full examples.
rasterList_lanK <- patLanK(imageList[1:2], landmarkList[1:2], k = 4, crop = TRUE,
res = 100, removebgK = 100, adjustCoords = TRUE, plot = TRUE)

## End(Not run)
```

---

patLanRGB *Aligns images usings transformations obtained from fixed landmarks and extracts colors using a predefined RGB values and cutoff value.*

Description

Aligns images usings transformations obtained from fixed landmarks and extracts colors using a predefined RGB values and cutoff value.

Usage

```
patLanRGB(
  sampleList,
  landList,
  RGB = NULL,
  sampleRGB = FALSE,
  sampleRGBtype = "point",
  resampleFactor = NULL,
  colOffset = 0.1,
  crop = FALSE,
  cropOffset = c(0, 0, 0, 0),
  res = 300,
  transformRef = "meanshape",
  transformType = "tps",
  adjustCoords = FALSE,
  plot = NULL,
  focal = FALSE,
  sigma = 3,
  iterations = 0,
  imageIDs = NULL,
  patternsToFile = NULL
)
```

Arguments

sampleList	List of RasterStack objects.
landList	Landmark list as returned by makeList.
RGB	RGB values for color pattern extraction specified as vector.
sampleRGB	Whether to set RGB for each image manually.
sampleRGBtype	Whether to pick a point or area (defined by left bottom and top right) for sampleRGB.
resampleFactor	Integer for downsampling used by redRes.
colOffset	Color offset for color pattern extraction (default = 0.10).
crop	Whether to use the landmarks range to crop the image. This can speed up the analysis (default = FALSE).
cropOffset	Vector c(xmin, xmax, ymin, ymax) that specifies the number of pixels you want the cropping to be offset from the landmarks (in case the landmarks do not surround the entire color pattern). The values specified should present the percentage of the maximum landmark value along the x and y axis.
res	Resolution for color pattern raster (default = 300). This should be reduced if the number of pixels in the image is lower than th raster.
transformRef	ID of reference sample for shape to which color patterns will be transformed to. Can be 'meanshape' for transforming to mean shape of Procrustes analysis.
transformType	Transformation type as used by computeTransform (default ='tps').
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).
plot	Whether to plot transformed color patterns while processing (default = NULL). Transformed color patterns can be plot on top of each other ('stack') or next to the original image for each sample ('compare').
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).
iterations	Number of iterations for recalculating average color.
imageIDs	A list of IDs to match landmarks to images if landmarkList and imageList don't have the same length.
patternsToFile	Name of directory to which the color pattern of each individual will be outputted (default = NULL).

Value

List of raster objects.

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '_landmarks_LFW.txt'
```

```
landmarkList <- makeList(IDlist, 'landmark', prepath, extension)

extension <- '.jpg'
imageList <- makeList(IDlist, 'image', prepath, extension)

RGB <- c(114,17,0)
rasterList_lanRGB <- patLanRGB(imageList, landmarkList, RGB,
colOffset = 0.15, crop = TRUE, res = 100, adjustCoords = TRUE, plot = 'stack')

## End(Not run)
```

---

patLanW *Extracts color pattern from landmark transformed image using watershed segmentation. This function works interactively by allowing to pick a starting pixel within each pattern element from which the watershed will extract the pattern. This function works best for patterns with sharp boundaries.*

Description

Extracts color pattern from landmark transformed image using watershed segmentation. This function works interactively by allowing to pick a starting pixel within each pattern element from which the watershed will extract the pattern. This function works best for patterns with sharp boundaries.

Usage

```
patLanW(
  sampleList,
  landList,
  IDlist = NULL,
  adjustCoords = FALSE,
  transformRef = "meanshape",
  resampleFactor = NULL,
  transformType = "tps",
  maskOutline = NULL,
  cartoonID = NULL,
  correct = FALSE,
  blur = TRUE,
  sigma = 3,
  bucketfill = TRUE,
  cleanP = NULL,
  splitC = NULL,
  plotTransformed = FALSE,
  plotCorrect = FALSE,
  plotEdges = FALSE,
  plotPriority = FALSE,
```

```
  plotWS = FALSE,
  plotBF = FALSE,
  plotFinal = FALSE
)
```

Arguments

sampleList	List of RasterStack objects.
landList	Landmark list as returned by makeList.
IDlist	List of sample IDs should be specified when masking outline and transformRef is 'meanshape'.
adjustCoords	Adjust landmark coordinates in case they are reversed compared to pixel coordinates (default = FALSE).
transformRef	ID or landmark matrix of reference sample for shape to which color patterns will be transformed to. Can be 'meanshape' for transforming to mean shape of Procrustes analysis.
resampleFactor	Integer for downsampling image used by redRes.
transformType	Transformation type as used by computeTransform (default ='tps').
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
cartoonID	ID of the sample for which the cartoon was drawn and will be used for masking (should be set when transformRef = 'meanShape').
correct	Correct image illumination using a linear model (default = FALSE).
blur	Blur image for priority map extraction (default = TRUE).
sigma	Size of sigma for Gaussian blurring (default = 5).
bucketfill	Use a bucket fill on the background to fill holes (default = TRUE).
cleanP	Integer to remove spurious areas with width smaller than cleanP (default = NULL).
splitC	Integer to split selected patterns into connected components and remove ones with areas smaller than splitC (default = NULL).
plotTransformed	Plot transformed image (default = FALSE).
plotCorrect	Plot corrected image, corrected for illumination using a linear model (default = FALSE).
plotEdges	Plot image gradient (default = FALSE).
plotPriority	Plot priority map (default = FALSE).
plotWS	Plot watershed result (default = FALSE).
plotBF	Plot bucketfill (default = FALSE).
plotFinal	Plot extracted patterns (default = FALSE).

Value

List of raster objects.

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '_landmarks_LFW.txt'

landmarkList <- makeList(IDlist, 'landmark', prepath, extension)

extension <- '.jpg'
imageList <- makeList(IDlist, 'image', prepath, extension)

outline_BC0077 <- read.table(paste(system.file("extdata",  package = 'patternize'),
'/BC0077_outline.txt', sep=''), header = FALSE)

rasterList_W <- patLanW(imageList, landmarkList, IDlist, transformRef = 'meanshape',
adjustCoords = TRUE, plotTransformed = FALSE, correct = TRUE, plotCorrect = FALSE, blur = FALSE,
sigma = 2, bucketfill = FALSE, cleanP = 0, splitC = 10, plotPriority = TRUE, plotWS = TRUE,
plotBF = TRUE, plotFinal = TRUE, maskOutline = outline_BC0077, cartoonID = 'BC0077')

## End(Not run)
```

---

patPCA *This function transforms the individual color pattern rasters as obtained by the main patternize functions to a dataframe of 0 and 1 values that can be used for Principal Component Analysis (prcomp). This function also allows to plot the analysis including a visualization of the shape changes along the axis. Pixel values are predicted by multiplying the rotation matrix (eigenvectors) with a vector that has the same length as the number of rows in the rotation matrix and in which all values are set to zero except for the PC value for which we want to predict the pixel values.*

Description

This function transforms the individual color pattern rasters as obtained by the main patternize functions to a dataframe of 0 and 1 values that can be used for Principal Component Analysis (prcomp). This function also allows to plot the analysis including a visualization of the shape changes along the axis. Pixel values are predicted by multiplying the rotation matrix (eigenvectors) with a vector that has the same length as the number of rows in the rotation matrix and in which all values are set to zero except for the PC value for which we want to predict the pixel values.

Usage

```
patPCA(
  rList,
  popList,
  colList,
```

```
  symbolList = NULL,
  rListPredict = NULL,
  popListPredict = NULL,
  colListPredict = NULL,
  pcaListPredict = NULL,
  pcaPopListPredict = NULL,
  pcaColPredict = "red",
  symbolListPredict = NULL,
  plot = FALSE,
  plotType = "points",
  plotChanges = FALSE,
  PCx = 1,
  PCy = 2,
  plotCartoon = FALSE,
  refShape = NULL,
  outline = NULL,
  lines = NULL,
  landList = NULL,
  adjustCoords = FALSE,
  crop = c(0, 0, 0, 0),
  flipRaster = NULL,
  flipOutline = NULL,
  imageList = NULL,
  cartoonID = NULL,
  refImage = NULL,
  colpalette = NULL,
  normalized = NULL,
  cartoonOrder = "above",
  lineOrder = "above",
  cartoonCol = "gray",
  cartoonFill = NULL,
  plotLandmarks = FALSE,
  landCol = "black",
  zlim = c(-1, 1),
  legendTitle = "Predicted",
  xlab = "",
  ylab = "",
  main = "",
  ...
)
```

Arguments

rList	List of raster objects.
popList	List of vectors including sampleIDs for each population.
colList	List of colors for each population.
symbolList	List with graphical plotting symbols (default = NULL).
rListPredict	List of raster objects to predict into PCA space (default = NULL).

popListPredict List of vectors including sampleIDs for each set of predict samples (default = NULL). Note to that this also has to be a list if only one population is included.

colListPredict List of colors for each set of predict samples (default = NULL).

pcaListPredict Points to plot within PCA space.

pcaPopListPredict
List of population symbols for plotting additional PCA values.

pcaColPredict Color for additional PCA values.

symbolListPredict
List with graphical plotting symbols for predict sets (default = NULL).

plot Whether to plot the PCA analysis (default = FALSE).

plotType Plot 'points' or sample 'labels' (default = 'points')

plotChanges Wether to include plots of the changes along the PC axis (default = FALSE).

PCx PC axis to be presented for x-axis (default PC1).

PCy PC axis to be presented for y-axis (default PC2).

plotCartoon Whether to plot a cartoon. This cartoon should be drawn on one of the samples used in the analysis.

refShape This can be 'target' in case the reference shape is a single sample (for registration analysis) or 'mean' if the images were transformed to a mean shape (only for meanshape when using landmark transformation)

outline xy coordinates that define outline.

lines list of files with xy coordinates of line objects to be added to cartoon.

landList Landmark landmarkList.

adjustCoords Adjust landmark coordinates.

crop Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image used in landmark or registration analysis.

flipRaster Whether to flip raster along xy axis (in case there is an inconsistency between raster and outline coordinates).

flipOutline Whether to flip plot along x, y or xy axis.

imageList List of image should be given if one wants to flip the outline or adjust landmark coordinates.

cartoonID ID of the sample for which the cartoon was drawn.

refImage Image (RasterStack) used for target. Use raster::stack('filename').

colpalette Vector of colors for color palette (default = c("white","lightblue","blue","green", "yellow","red"))

normalized Set this to true in case the summed rasters are already devided by the sample number.

cartoonOrder Whether to plot the cartoon outline 'above' or 'under' the pattern raster (default = 'above'). Set to 'under' for filled outlines.

lineOrder Whether to plot the cartoon lines 'above' or 'under' the pattern raster (default = 'above').

cartoonCol	Outline and line color for cartoon (deafault = 'gray').
cartoonFill	Fill color for outline of cartoon (default = NULL).
plotLandmarks	Whether to plot the landmarks from the target image or mean shape landmarks (default = FALSE).
landCol	Color for plotting landmarks (default = 'black').
zlim	z-axis limit (default = c(0,1))
legendTitle	Title of the raster legend (default = 'Proportion')
xlab	Optional x-axis label.
ylab	Optional y-axis label.
main	Optional main title.
...	additional arguments for PCA plot function.

Value

If plot = TRUE: List including a [1] dataframe of the binary raster values that can be used for principle component analysis, [2] a dataframe of sample IDs and specified population colors and [3] prcomp results. If plot = FALSE: prcomp result.

See Also

prcomp

Examples

```
data(rasterList_lanRGB)

pop1 <- c('BC0077','BC0071')
pop2 <- c('BC0050','BC0049','BC0004')
popList <- list(pop1, pop2)
colList <- c("red", "blue")

pcaOut <- patPCA(rasterList_lanRGB, popList, colList, plot = TRUE)
```

---

patRDA *This function transforms the individual color pattern rasters as obtained by the main patternize functions to a dataframe of 0 and 1 values that can be used for constrained Redundancy Analysis (RDA) (rda). This function also allows to plot the analysis including a visualization of the shape changes along the axis.*

Description

This function transforms the individual color pattern rasters as obtained by the main patternize functions to a dataframe of 0 and 1 values that can be used for constrained Redundancy Analysis (RDA) (rda). This function also allows to plot the analysis including a visualization of the shape changes along the axis.

Usage

```
patRDA(
  rList,
  popList,
  colList,
  symbolList = NULL,
  rListPredict = NULL,
  popListPredict = NULL,
  colListPredict = NULL,
  symbolListPredict = NULL,
  plot = FALSE,
  plotType = "points",
  plotChanges = FALSE,
  PCx = 1,
  PCy = 2,
  plotCartoon = FALSE,
  refShape = NULL,
  outline = NULL,
  lines = NULL,
  landList = NULL,
  adjustCoords = FALSE,
  crop = c(0, 0, 0, 0),
  flipRaster = NULL,
  flipOutline = NULL,
  imageList = NULL,
  cartoonID = NULL,
  colpalette = NULL,
  normalized = NULL,
  cartoonOrder = "above",
  lineOrder = "above",
  cartoonCol = "gray",
  cartoonFill = NULL,
  plotLandmarks = FALSE,
  landCol = "black",
  zlim = c(-1, 1),
  legendTitle = "Predicted",
  xlab = "",
  ylab = "",
  main = ""
)
```

Arguments

rList	List of raster objects.
popList	List of vectors including sampleIDs for each population.
colList	List of colors for each population.
symbolList	List with graphical plotting symbols (default = NULL).

rListPredict	List of raster objects to predict into DFA space (default = NULL).
popListPredict	List of vectors including sampleIDs for each set of predict samples (default = NULL). Note to that this also has to be a list if only one population is included.
colListPredict	List of colors for each set of predict samples (default = NULL).
symbolListPredict	List with graphical plotting symbols for predict sets (default = NULL).
plot	Whether to plot the PCA analysis (default = FALSE).
plotType	Plot 'points' or sample 'labels' (default = 'points')
plotChanges	Wether to include plots of the changes along the PC axis (default = FALSE).
PCx	PC axis to be presented for x-axis (default PC1).
PCy	PC axis to be presented for y-axis (default PC2).
plotCartoon	Whether to plot a cartoon. This cartoon should be drawn on one of the samples used in the analysis.
refShape	This can be 'target' in case the reference shape is a single sample (for registration analysis) or 'mean' if the images were transformed to a mean shape (only for meanshape when using landmark transformation)
outline	xy coordinates that define outline.
lines	list of files with xy coordinates of line objects to be added to cartoon.
landList	Landmark landmarkList.
adjustCoords	Adjust landmark coordinates.
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image used in landmark or registration analysis.
flipRaster	Whether to flip raster along xy axis (in case there is an inconsistency between raster and outline coordinates).
flipOutline	Whether to flip plot along x, y or xy axis.
imageList	List of image should be given if one wants to flip the outline or adjust landmark coordinates.
cartoonID	ID of the sample for which the cartoon was drawn.
colpalette	Vector of colors for color palette (default = c("white","lightblue","blue","green", "yellow","red"))
normalized	Set this to true in case the summed rasters are already devided by the sample number.
cartoonOrder	Whether to plot the cartoon outline 'above' or 'under' the pattern raster (default = 'above'). Set to 'under' for filled outlines.
lineOrder	Whether to plot the cartoon lines 'above' or 'under' the pattern raster (default = 'above').
cartoonCol	Outline and line color for cartoon (deafault = 'gray').
cartoonFill	Fill color for outline of cartoon (default = NULL).
plotLandmarks	Whether to plot the landmarks from the target image or mean shape landmarks (default = FALSE).

landCol	Color for plotting landmarks (default = 'black').
zlim	z-axis limit (default = c(0,1))
legendTitle	Title of the raster legend (default = 'Proportion')
xlab	Optional x-axis label.
ylab	Optional y-axis label.
main	Optional main title.

Value

If plot = TRUE: List including a [1] dataframe of the binary raster values that can be used for discriminant function analysis, [2] a dataframe of sample IDs and specified population colors and [3] lda results. if rListPredict not empty: [4] class prediction of samples. If plot = FALSE: lda result only.

See Also

lda

Examples

```
data(rasterList_lanRGB)

pop1 <- c('BC0077','BC0071')
pop2 <- c('BC0050','BC0049','BC0004')
popList <- list(pop1, pop2)
colList <- c("red", "blue")

pcaOut <- patRDA(rasterList_lanRGB, popList, colList, plot = TRUE)
```

---

patRegHSV *Aligns images using niftyreg utilities for automated image registration and extracts colors using a predefined HSV values and cutoff value.*

Description

Aligns images using niftyreg utilities for automated image registration and extracts colors using a predefined HSV values and cutoff value.

Usage

```
patRegHSV(
  sampleList,
  target,
  HSV,
  resampleFactor = NULL,
```

```
  useBlockPercentage = 75,
  colOffset = 0.1,
  crop = c(0, 0, 0, 0),
  removebgR = NULL,
  maskOutline = NULL,
  plot = FALSE,
  focal = FALSE,
  sigma = 3,
  iterations = 0,
  ignoreHSVvalue = FALSE,
  patternsToFile = NULL
)
```

Arguments

sampleList	List of RasterStack objects.
target	Image imported as RasterStack used as target for registration.
HSV	Values for color pattern extraction specified as HSV vector.
resampleFactor	Integer for downsampling used by redRes (default = NULL).
useBlockPercentage	Block percentage as used in niftyreg (default = 75).
colOffset	Color offset for color pattern extraction (default = 0.10).
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image.
removebgR	Integer indicating the range RGB treshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL) for registration analysis. This works only to remove a white background.
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
plot	Whether to plot transformed color patterns while processing (default = FALSE). Transformed color patterns can be plot on top of each other ('stack') or next to the original image for each sample ('compare').
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).
iterations	Number of iterations for recalculating average color (default = 0). If set, the RGB value for pattern extraction will be iteratively recalculated to be the average of the extracted area. This may improve extraction of distinct color pattern, but fail for more gradually distributed (in color space) patterns.
ignoreHSVvalue	Whether to ignore the HSV value (~darkness).
patternsToFile	Name of directory to which the color pattern of each individual will be outputted (default = NULL).

Value

List of raster objects.

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '.jpg'

imageList <- makeList(IDlist, 'image', prepath, extension)

target <- imageList[[1]]

HSV <- c(0.025,1,0.45)

# Note that this example only aligns one image with the target,
# remove [2] to run a full examples.
rasterList_regHSV <- patRegRGB(imageList[2], target, HSV,
colOffset= 0.15, crop = c(100,400,40,250), removebgR = 100, plot = 'stack')

## End(Not run)
```

---

patRegK *Aligns images using* niftyreg *utilities for automated image registration and extracts colors using k-means clustering.*

Description

Aligns images using niftyreg utilities for automated image registration and extracts colors using k-means clustering.

Usage

```
patRegK(
  sampleList,
  target,
  k = 3,
  fixedStartCenter = NULL,
  resampleFactor = NULL,
  useBlockPercentage = 75,
  crop = c(0, 0, 0, 0),
  removebgR = NULL,
  removebgK = NULL,
  maskOutline = NULL,
  maskColor = 0,
  plot = FALSE,
  focal = FALSE,
  sigma = 3
)
```

Arguments

sampleList	List of RasterStack objects.
target	Image imported as RasterStack used as target for registration.
k	Integere for defining number of k-means clusters (default = 3).
fixedStartCenter	Specify a dataframe with start centers for k-means clustering.
resampleFactor	Integer for downsampling used by redRes (default = NULL).
useBlockPercentage	Block percentage as used in niftyreg (default = 75).
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image.
removebgR	Integer indicating the range RGB treshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL) for registration analysis. This works only to remove a white background.
removebgK	Integer indicating the range RGB treshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL) for k-means analysis. This works only to remove a white background.
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
maskColor	Color the masked area gets. Set to 0 for black (default) or 255 for white.
plot	Whether to plot k-means clustered image while processing (default = FALSE).
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).

Value

List of rasters for each k-means cluster objects.

Examples

```
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '.jpg'

imageList <- makeList(IDlist, 'image', prepath, extension)

target <- imageList[[1]]

## Not run:
rasterList_regK <- patRegK(imageList[3], target, k = 5,
crop = c(100,400,40,250), removebgR = 100, plot = TRUE)

## End(Not run)
```

patRegK_HSV *Aligns images using* niftyreg *utilities for automated image registration and extracts colors using k-means clustering.*

Description

Aligns images using niftyreg utilities for automated image registration and extracts colors using k-means clustering.

Usage

```
patRegK_HSV(
  sampleList,
  target,
  k = 3,
  fixedStartCenter = NULL,
  resampleFactor = NULL,
  useBlockPercentage = 75,
  crop = c(0, 0, 0, 0),
  removebgR = NULL,
  removebgK = NULL,
  maskOutline = NULL,
  maskColor = 0,
  plot = FALSE,
  focal = FALSE,
  sigma = 3,
  ignoreHSVvalue = FALSE
)
```

Arguments

sampleList	List of RasterStack objects.
target	Image imported as RasterStack used as target for registration.
k	Integere for defining number of k-means clusters (default = 3).
fixedStartCenter	Specify a dataframe with start centers for k-means clustering.
resampleFactor	Integer for downsampling used by redRes (default = NULL).
useBlockPercentage	Block percentage as used in niftyreg (default = 75).
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image.
removebgR	Integer indicating the range RGB treshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL) for registration analysis. This works only to remove a white background.

removebgK	Integer indicating the range RGB treshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL) for k-means analysis. This works only to remove a white background.
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
maskColor	Color the masked area gets. Set to 0 for black (default) or 255 for white.
plot	Whether to plot k-means clustered image while processing (default = FALSE).
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).
ignoreHSVvalue	Whether to ignore the HSV value (~darkness).

Value

List of rasters for each k-means cluster objects.

Examples

```
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '.jpg'

imageList <- makeList(IDlist, 'image', prepath, extension)

target <- imageList[[1]]

## Not run:
rasterList_regK <- patRegK(imageList[3], target, k = 5,
crop = c(100,400,40,250), removebgR = 100, plot = TRUE)

## End(Not run)
```

---

patRegRGB *Aligns images using* niftyreg *utilities for automated image registration and extracts colors using a predefined RGB values and cutoff value.*

Description

Aligns images using niftyreg utilities for automated image registration and extracts colors using a predefined RGB values and cutoff value.

Usage

```
patRegRGB(
  sampleList,
  target,
  RGB,
  resampleFactor = NULL,
  useBlockPercentage = 75,
  colOffset = 0.1,
  crop = c(0, 0, 0, 0),
  removebgR = NULL,
  maskOutline = NULL,
  plot = FALSE,
  focal = FALSE,
  sigma = 3,
  iterations = 0,
  patternsToFile = NULL
)
```

Arguments

sampleList	List of RasterStack objects.
target	Image imported as RasterStack used as target for registration.
RGB	Values for color pattern extraction specified as RGB vector.
resampleFactor	Integer for downsampling used by redRes (default = NULL).
useBlockPercentage	Block percentage as used in niftyreg (default = 75).
colOffset	Color offset for color pattern extraction (default = 0.10).
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image.
removebgR	Integer indicating the range RGB treshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL) for registration analysis. This works only to remove a white background.
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
plot	Whether to plot transformed color patterns while processing (default = FALSE). Transformed color patterns can be plot on top of each other ('stack') or next to the original image for each sample ('compare').
focal	Whether to perform Gaussian blurring (default = FALSE).
sigma	Size of sigma for Gaussian blurring (default = 3).
iterations	Number of iterations for recalculating average color (default = 0). If set, the RGB value for pattern extraction will be iteratively recalculated to be the average of the extracted area. This may improve extraction of distinct color pattern, but fail for more gradually distributed (in color space) patterns.
patternsToFile	Name of directory to which the color pattern of each individual will be outputted (default = NULL).

Value

List of raster objects.

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '.jpg'

imageList <- makeList(IDlist, 'image', prepath, extension)

target <- imageList[[1]]

RGB <- c(114,17,0)

# Note that this example only aligns one image with the target,
# remove [2] to run a full examples.
rasterList_regRGB <- patRegRGB(imageList[2], target, RGB,
colOffset= 0.15, crop = c(100,400,40,250), removebgR = 100, plot = 'stack')

## End(Not run)
```

---

patRegW *Aligns images using* niftyreg *utilities for automated image registration and extracts color pattern using watershed segmentation. This function works interactively by allowing to pick a starting pixel within each pattern element from which the watershed will extract the pattern. This function works best for patterns with sharp boundaries.*

Description

Aligns images using niftyreg utilities for automated image registration and extracts color pattern using watershed segmentation. This function works interactively by allowing to pick a starting pixel within each pattern element from which the watershed will extract the pattern. This function works best for patterns with sharp boundaries.

Usage

```
patRegW(
  sampleList,
  target,
  resampleFactor = NULL,
  useBlockPercentage = 75,
  crop = c(0, 0, 0, 0),
  removebgR = NULL,
  maskOutline = NULL,
```

```
  cartoonID = NULL,
  correct = FALSE,
  blur = TRUE,
  sigma = 3,
  bucketfill = TRUE,
  cleanP = NULL,
  splitC = NULL,
  plotTransformed = FALSE,
  plotCorrect = FALSE,
  plotEdges = FALSE,
  plotPriority = FALSE,
  plotWS = FALSE,
  plotBF = FALSE,
  plotFinal = FALSE
)
```

Arguments

sampleList	List of RasterStack objects.
target	Image imported as RasterStack used as target for registration.
resampleFactor	Integer for downsampling image used by redRes.
useBlockPercentage	Block percentage as used in niftyreg (default = 75).
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image.
removebgR	Integer indicating the range RGB treshold to remove from image (e.g. 100 removes pixels with average RGB > 100; default = NULL) for registration analysis. This works only to remove a white background.
maskOutline	When outline is specified, everything outside of the outline will be masked for the color extraction (default = NULL).
cartoonID	ID of the sample for which the cartoon was drawn and will be used for masking.
correct	Correct image illumination using a linear model (default = FALSE).
blur	Blur image for priority map extraction (default = TRUE).
sigma	Size of sigma for Gaussian blurring (default = 5).
bucketfill	Use a bucket fill on the background to fill holes (default = TRUE).
cleanP	Integer to remove spurious areas with width smaller than cleanP (default = NULL).
splitC	Integer to split selected patterns into connected components and remove ones with areas smaller than splitC (default = NULL).
plotTransformed	Plot transformed image (default = FALSE).
plotCorrect	Plot corrected image, corrected for illumination using a linear model (default = FALSE).
plotEdges	Plot image gradient (default = FALSE).

plotPriority	Plot priority map (default = FALSE).
plotWS	Plot watershed result (default = FALSE).
plotBF	Plot bucketfill (default = FALSE).
plotFinal	Plot extracted patterns (default = FALSE).

Value

List of raster objects.

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '.jpg'

imageList <- makeList(IDlist, 'image', prepath, extension)

target <- imageList[[1]]

outline_BC0077 <- read.table(paste(system.file("extdata",  package = 'patternize'),
'/BC0077_outline.txt', sep=''), header = FALSE)

rasterList_regW <- patRegW(imageList, target, plotTransformed = FALSE, cartoonID = 'BC0077',
                           correct = TRUE, plotCorrect = FALSE, blur = FALSE, sigma = 2,
                         bucketfill = FALSE, cleanP = 0, splitC = 10, plotPriority = TRUE,
                        plotWS = FALSE, plotBF = FALSE, plotFinal = TRUE, removebgR = 100,
                           maskOutline = outline_BC0077)

## End(Not run)
```

---

patternize *patternize - An R package for quantifying color pattern variation.*

Description

Quantifying variation in color patterns to study and compare the consistency of their expression necessitates the homologous alignment and color-based segmentation of images. Patternize is an R package that quantifies variation in color patterns as obtained from image data. Patternize defines homology between pattern positions across specimens either through fixed landmarks or image registration. Pattern identification is performed by categorizing the distribution of colors using either an RGB threshold or an unsupervised image segmentation. The quantification of the color patterns can be visualized as heat maps and compared between sets of samples.

patternize main functions

The package has six main functions depending on how you want the alignment of the iamges and the color extraction to be performed.

`patLanRGB`
Aligns images by transformations obtained from fixed landmarks and extracts colors using a predefined RGB values and cutoff value.

`patLanK`
Aligns images by transformations obtained from fixed landmarks and extracts colors using k-means clustering.

`patLanW`
Aligns images by transformations obtained from fixed landmarks and extracts color patterns by watershed segmentation using `imager` utilities.

`patRegRGB`
Aligns images using `niftyreg` utilities for automated image registration and extracts colors using a predefined RGB values and cutoff value.

`patRegK`
Aligns images using `niftyreg` utilities for automated image registration and extracts colors using k-means clustering.

`patRegW`
Aligns images using `niftyreg` utilities for automated image registration and extracts color patterns by watershed segmentation using `imager` utilities.

patternize preprocessing functions

The input for the main patternize functions are `RasterStack` objects and when landmark transformation is used, landmark arrays.

`makeList`
This function returns a list of RasterStacks or a list of landmarks depending on the input provided.

`sampleLandmarks`
Sample landmarks in an image.

`lanArray`
This function creates a landmark array as used by `procSym` in the package `Morpho`.

patternize postprocessing functions

`sumRaster`
This function sums the individual color pattern rasters as obtained by the main patternize functions.

`plotHeat`
Plots the color pattern heatmaps from `sumRaster` output.

`patPCA`
This function transforms the individual color pattern rasters as obtained by the main patternize functions to a dataframe of 0 and 1 values that can be used for Principal Component Analysis (`prcomp`). This function also allows to plot the analysis including a visualization of the shape changes along the axis.

`patRDA`
This function transforms the individual color pattern rasters as obtained by the main patternize functions to a dataframe of 0 and 1 values that can be used for constrained Redundancy Analysis (`rda`). This function also allows to plot the analysis including a visualization of the shape changes along the axis.

`patArea`
This fucntion calculates the area in which the color pattern is expressed in each sample as the relative proportion using the provided outline of the considered trait or structure.

patternize miscellaneous functions

`redRes`
Reduces the resolution of the `RasterStack` objects to speed up analysis.

`kImage`
Performs k-means clustering of images.

`sampleRGB`
Interactive function to sample RGB value from pixel or area in an image.

`createTarget`
Creates an artificial target images using a provided outline that can be used for image registration (experimantal).

`maskOutline`
Intersects a RasterStack with an outline. Everything outside of the outline will be removed from the raster.

`colorChecker`
Calibrate images using ColorChecker.

Author(s)

Steven M. Van Belleghem

See Also

`raster`, `stack`, `procSym`, `computeTransform`, `niftyreg` `imager`

Jon Clayden, Marc Modat, Benoit Presles, Thanasis Anthopoulos and Pankaj Daga (2017). RNiftyReg: Image Registration Using the 'NiftyReg' Library. R package version 2.5.0. https://CRAN.R-project.org/package=RNiftyReg

Stefan Schlager (2016). Morpho: Calculations and Visualisations Related to Geometric Morphometrics. R package version 2.4.1.1. https://github.com/zarquon42b/Morpho

Simon Barthelmé (2017). imager: Image processing library based on 'CImg'. R package version 0.40.2. https://CRAN.R-project.org/package=imager

plotHeat *Plots the color pattern heatmaps from* sumRaster *output.*

Description

Plots the color pattern heatmaps from sumRaster output.

Usage

```
plotHeat(
  summedRaster,
  IDlist,
  colpalette = NULL,
  plotCartoon = FALSE,
  refShape = NULL,
  outline = NULL,
  lines = NULL,
  landList = NULL,
  adjustCoords = FALSE,
  cartoonID = NULL,
  normalized = FALSE,
  crop = c(0, 0, 0, 0),
  flipRaster = NULL,
  flipOutline = NULL,
  imageList = NULL,
  refImage = NULL,
  cartoonOrder = "above",
  lineOrder = "above",
  cartoonCol = "gray",
  cartoonFill = NULL,
  plotLandmarks = FALSE,
  landCol = "black",
  zlim = c(0, 1),
  legend = TRUE,
  legendTitle = "Proportion",
  legend.side = 4,
  xlab = "",
  ylab = "",
  main = "",
  plotType = "multi",
  imageIDs = NULL,
  format = "imageJ"
)
```

Arguments

summedRaster Summed raster or summedRasterList.

IDlist	List of sample IDs.
colpalette	Vector of colors for color palette (default = c("white","lightblue","blue","green", "yellow","red"))
plotCartoon	Whether to plot a cartoon. This cartoon should be drawn on one of the samples used in the analysis.
refShape	This can be 'target' in case the reference shape is a single sample (for registration analysis) or 'mean' if the images were transformed to a mean shape (only for meanshape when using landmark transformation)
outline	xy coordinates that define outline.
lines	list of files with xy coordinates of line objects to be added to cartoon.
landList	Landmark landmarkList.
adjustCoords	Adjust landmark coordinates.
cartoonID	ID of the sample for which the cartoon was drawn.
normalized	Set this to true in case the summed rasters are already devided by the sample number.
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image used in landmark or registration analysis.
flipRaster	Whether to flip raster along xy axis (in case there is an inconsistency between raster and outline coordinates).
flipOutline	Whether to flip plot along x, y or xy axis.
imageList	List of images should be given if one wants to flip the outline or adjust landmark coordinates.
refImage	Image (RasterStack) used for target. Use raster::stack('filename').
cartoonOrder	Whether to plot the cartoon outline 'above' or 'under' the pattern raster (default = 'above'). Set to 'under' for filled outlines.
lineOrder	Whether to plot the cartoon lines 'above' or 'under' the pattern raster (default = 'above').
cartoonCol	Outline and line color for cartoon (deafault = 'gray').
cartoonFill	Fill color for outline of cartoon (default = NULL).
plotLandmarks	Whether to plot the landmarks from the target image or mean shape landmarks (default = FALSE).
landCol	Color for ploting landmarks (default = 'black').
zlim	z-axis limit (default = c(0,1))
legend	Whether to plot legend with heatmaps.
legendTitle	Title of the raster legend (default = 'Proportion')
legend.side	Side to plot legend (default = 4)
xlab	Optional x-axis label.
ylab	Optional y-axis label.
main	Optional main title.

plotType	Set as 'PCA' when visualizing shape changes along PCA axis in \ codepatPCA, as 'one' when visualizing single image or as 'multi' for multi plotting or when setting customized margins (default = 'multi').
imageIDs	A list of IDs to match landmarks to images if landmarkList and imageList don't have the same length.
format	ImageJ (Fiji) or tps format (default = 'imageJ').

Examples

```
data(rasterList_lanRGB)
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
outline_BC0077 <- read.table(paste(system.file("extdata",  package = 'patternize'),
'/BC0077_outline.txt', sep=''), header = FALSE)
lines_BC0077 <- list.files(path=paste(system.file("extdata",  package = 'patternize')),
pattern='vein', full.names = TRUE)

summedRaster_regRGB <- sumRaster(rasterList_regRGB, IDlist, type = 'RGB')
data(imageList)

plotHeat(summedRaster_regRGB, IDlist, plotCartoon = TRUE, refShape = 'target',
outline = outline_BC0077, lines = lines_BC0077, crop = c(100,400,40,250),
flipRaster = 'xy', imageList = imageList, cartoonOrder = 'under', cartoonID = 'BC0077',
cartoonFill = 'black', main = 'registration_example')

## Not run:
data(rasterList_lanK)
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
summedRasterList <- sumRaster(rasterList_lanK, IDlist, type = 'k')
plotHeat(summedRasterList, IDlist)

summedRasterList_regK <- sumRaster(rasterList_regK, IDlist, type = 'k')
plotHeat(summedRasterList_regK, IDlist, plotCartoon = TRUE, refShape = 'target',
outline = outline_BC0077, lines = lines_BC0077, crop = c(100,400,40,250),
flipRaster = 'y', imageList = imageList, cartoonOrder = 'under',
cartoonFill = 'black', main = 'kmeans_example')

plotHeat(summedRasterList_regK[[1]], IDlist, plotCartoon = TRUE, refShape = 'target',
outline = outline_BC0077, lines = lines_BC0077, crop = c(100,400,40,250),
flipRaster = 'y', imageList = imageList, cartoonOrder = 'under',
cartoonFill = 'black', main = 'kmeans_example')


prepath <- system.file("extdata", package = 'patternize')
extension <- '_landmarks_LFW.txt'
landmarkList <- makeList(IDlist, 'landmark', prepath, extension)

summedRaster_lanRGB <- sumRaster(rasterList_lanRGB, IDlist, type = 'RGB')

plotHeat(summedRaster_lanRGB, IDlist, plotCartoon = TRUE, refShape = 'mean',
outline = outline_BC0077, lines = lines_BC0077, landList = landmarkList,
adjustCoords = TRUE, imageList = imageList, cartoonID = 'BC0077',
cartoonOrder = 'under', cartoonFill= 'black', main = 'Landmark_example')
```

```
summedRaster_lanK <- sumRaster(rasterList_lanK, IDlist, type = 'k')

plotHeat(summedRaster_lanK, IDlist, plotCartoon = TRUE, refShape = 'mean',
outline = outline_BC0077, lines = lines_BC0077, landList = landmarkList,
adjustCoords = TRUE, imageList = imageList, cartoonID = 'BC0077',
cartoonOrder = 'under', cartoonFill= 'black', main = 'Landmark_example')

plotHeat(summedRaster_lanK[[2]], IDlist, plotCartoon = TRUE, refShape = 'mean',
outline = outline_BC0077, lines = lines_BC0077, landList = landmarkList,
adjustCoords = TRUE, imageList = imageList, cartoonID = 'BC0077',
cartoonOrder = 'under', cartoonFill= 'black', main = 'Landmark_example')

## End(Not run)
```

`plotRasterstackAsImage` *Plot rasterStack as image.*

Description

Plot rasterStack as image.

Usage

```
plotRasterstackAsImage(rasterStack, flipY = FALSE)
```

Arguments

`rasterStack`	A single rasterStack.
`flipY`	Whether to flip the raster along the Y-axis.

`rasterList_lanK` *rasterList_lanK*

Description

List of RasterLayers as returned by `patLanK`.

Usage

```
rasterList_lanK
```

Format

A list of RasterLayers including the red color pattern extracted from 5 Heliconius erato hydara dorsal forewings using `patLanK`.

Examples

```
## Not run:
data(rasterList_lanK)
summary(rasterList_lanL)

## End(Not run)
```

`rasterList_lanRGB` *rasterList_lanRGB*

Description

List of RasterLayers as returned by `patLanRGB`.

Usage

```
rasterList_lanRGB
```

Format

A list of RasterLayers including the red color pattern extracted from 5 Heliconius erato hydara dorsal forewings using `patLanRGB`.

Examples

```
## Not run:
data(rasterList_lanRGB)
summary(rasterList_lanRGB)

## End(Not run)
```

`rasterList_regK` *rasterList_regK*

Description

List of RasterLayers as returned by `patRegK`.

Usage

```
rasterList_regK
```

Format

A list of RasterLayers including the red color pattern extracted from 5 Heliconius erato hydara dorsal forewings using `patRegK`.

Examples

```
## Not run:
data(rasterList_regK)
summary(rasterList_regK)

## End(Not run)
```

`rasterList_regRGB` *rasterList_regRGB*

Description

List of RasterLayers as returned by `patRegRGB`.

Usage

```
rasterList_regRGB
```

Format

A list of RasterLayers including the red color pattern extracted from 5 Heliconius erato hydara dorsal forewings using `patRegRGB`.

Examples

```
## Not run:
data(rasterList_regRGB)
summary(rasterList_regRGB)

## End(Not run)
```

`redRes` *Reduce the resolution of an image imported as a RasterStack by downsampling.*

Description

Reduce the resolution of an image imported as a RasterStack by downsampling.

Usage

```
redRes(image, resampleFactor)
```

Arguments

`image`	RasterStack for downsampling.
`resampleFactor`	Integer for downsampling.

Value

Downsampled RasterStack

Examples

```
image <- raster::stack(system.file("extdata", "BC0077.jpg", package = "patternize"))
image_reduced <- redRes(image, 5)
```

`sampleLandmarks` *Sample landmarks in an image.*

Description

Sample landmarks in an image.

Usage

```
sampleLandmarks(sampleList, resampleFactor = NULL, crop = c(0, 0, 0, 0))
```

Arguments

`sampleList`	RasterStack or list of RasterStack objects as obtained by `makeList`.
`resampleFactor`	Integer for downsampling the image(s) used by `redRes`.
`crop`	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image.

Value

landmark matrix or landmark list

Examples

```
## Not run:
IDlist <- c('BC0077','BC0071')
prepath <- system.file("extdata",  package = 'patternize')
extension <- '.jpg'
imageList <- makeList(IDlist, 'image', prepath, extension)

landmarkList <- sampleLandmarks(imageList)

## End(Not run)
```

sampleRGB *Interactive function to sample RGB value from pixel or square area in an image.*

Description

Interactive function to sample RGB value from pixel or square area in an image.

Usage

```
sampleRGB(image, resampleFactor = NULL, crop = c(0, 0, 0, 0), type = "point")
```

Arguments

image	Image imported as a RasterStack.
resampleFactor	Integer for downsampling used by redRes.
crop	Vector c(xmin, xmax, ymin, ymax) that specifies the pixel coordinates to crop the original image.
type	Set 'point' to extract RGB from a single point or 'area' to extract from a square area defined by setting two points (default = 'point').

Value

RGB vector

Examples

```
image <- raster::stack(system.file("extdata", "BC0077.jpg", package = "patternize"))
RGB <- sampleRGB(image, resampleFactor = 1)
```

setMask *Interactive function to to draw an outline for masking.*

Description

Interactive function to to draw an outline for masking.

Usage

```
setMask(summedRaster, IDlist, filename, ...)
```

Arguments

summedRaster	Summed raster of extracted patterns.
IDlist	List of sample IDs.
filename	Name of file to which mask will be written.
...	additional arguments for plotHeat function.

Value

file

---

sumRaster *This function sums the individual color pattern RasterLayes as obtained by the main patternize functions.*

Description

This function sums the individual color pattern RasterLayes as obtained by the main patternize functions.

Usage

```
sumRaster(rList, IDlist, type)
```

Arguments

rList	List of RasterLayers or list of RasterLayers for each k-means cluster.
IDlist	List of sample IDs.
type	Type of rasterlist; 'RGB' or 'k' (result from RGB or k-means analysis, respectively).

Examples

```
data(rasterList_lanRGB)
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
summedRaster <- sumRaster(rasterList_lanRGB, IDlist, type = 'RGB')

data(rasterList_lanK)
IDlist <- c('BC0077','BC0071','BC0050','BC0049','BC0004')
summedRasterList <- sumRaster(rasterList_lanK, IDlist, type = 'k')
```

Index