Package 'imagerExtra'

January 25, 2019

Type Package

Title Extra Image Processing Library Based on 'imager'

Version 1.3.2

Maintainer Shota Ochi <shotaochi1990@gmail.com>

Description Provides advanced functions for image processing based on the package 'imager'.

License GPL-3

Depends R (>= 2.10.0), imager (>= 0.40.2)

Imports fftwtools, magrittr, Rcpp (>= 0.12.14)

Suggests testthat (>= 2.0.0), knitr, rmarkdown, tesseract

URL https://github.com/ShotaOchi/imagerExtra

BugReports https://github.com/ShotaOchi/imagerExtra/issues

LinkingTo Rcpp

LazyData true

RoxygenNote 6.1.1

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation yes

Author Shota Ochi [aut, cre],
Guoshen Yu [ctb, cph],
Guillermo Sapiro [ctb, cph],
Catalina Sbert [ctb, cph],
Image Processing On Line [cph],
Pascal Getreuer [ctb, cph]

Repository CRAN

Date/Publication 2019-01-25 13:50:02 UTC

R topics documented:

`BalanceSimplest` *Balance color of image by Simplest Color Balance*

Description

Balance color of image by Simplest Color Balance

Usage

```
BalanceSimplest(im, sleft, sright, range = c(0, 255))
```

Arguments

`im`	a grayscale image of class cimg
`sleft`	left saturation percentage. sleft can be specified by numeric or string, e.g. 1 and "1%". note that sleft is a percentile.
`sright`	right saturation percentage. sright can be specified by numeric or string. note that sright is a percentile.
`range`	this function assumes that the range of pixel values of of input image is [0,255] by default. you may prefer [0,1].

Value

a grayscale image of class cimg

Author(s)

Shota Ochi

References

Nicolas Limare, Jose-Luis Lisani, Jean-Michel Morel, Ana Belen Petro, and Catalina Sbert, Simplest Color Balance, Image Processing On Line, 1 (2011), pp. 297-315. https://doi.org/10.5201/ipol.2011.llmps-scb

Examples

```
dev.new()
par(mfcol = c(1,2))
boats_g <- grayscale(boats)
plot(boats_g, main = "Original")
BalanceSimplest(boats_g, 1, 1) %>% plot(., main = "Simplest Color Balance")
```

DCT *Two Dimensional Discrete Cosine Transformation and Inverse Cosine Transformation*

Description

DCT2D computes two dimensional discrete cosine transformation. IDCT2D computes two dimensional inverse discrete cosine transformation.

Usage

```
DCT2D(imormat, returnmat = FALSE)

IDCT2D(imormat, returnmat = FALSE)
```

Arguments

imormat	a grayscale image of class cimg or a numeric matrix
returnmat	if returnmat is TRUE, returns numeric matrix. if FALSE, returns a grayscale image of class cimg.

Value

a grayscale image of class cimg or a numeric matrix

Author(s)

Shota Ochi

References

Makhoul, J. (1980). A fast cosine transform in one and two dimensions. IEEE Transactions on Acoustics, Speech, and Signal Processing. 28 (1): 27-34.

Examples

```
g <- grayscale(boats)
layout(matrix(1:2, 1, 2))
plot(g, main = "Original")
gg <- DCT2D(g) %>% IDCT2D() %>% plot(main = "Transformed")
mean((g - gg)^2)
```

`DenoiseDCT` *denoise image by DCT denoising*

Description

denoise image by DCT denoising

Usage

```
DenoiseDCT(im, sdn, flag_dct16x16 = FALSE)
```

Arguments

`im`	a grayscale image of class cimg
`sdn`	standard deviation of Gaussian white noise
`flag_dct16x16`	flag_dct16x16 determines the size of patches. if TRUE, the size of patches is 16x16. if FALSE, the size if patches is 8x8.

Value

a grayscale image of class cimg

Author(s)

Shota Ochi

References

Guoshen Yu, and Guillermo Sapiro, DCT Image Denoising: a Simple and Effective Image Denoising Algorithm, Image Processing On Line, 1 (2011), pp. 292-296. https://doi.org/10.5201/ipol.2011.ys-dct

Examples

```
dev.new()
par(mfcol = c(1,2))
boats_g <- grayscale(boats)
boats_noisy <- imnoise(dim = dim(boats_g), sd = 0.05) + boats_g
plot(boats_noisy, main = "Noisy Boats")
DenoiseDCT(boats_g, 0.05) %>% plot(., main = "Denoised Boats")
```

---

dogs *Photograph of a dog from GAHAG*

Description

This photograph was downloaded from http://gahag.net/img/201603/03s/gahag-0062116383-1.jpg. Its size was reduced by half to speed up loading and save space.

Usage

```
dogs
```

Format

an image of class cimg

Source

http://gahag.net/img/201603/03s/gahag-0062116383-1.jpg

EqualizeADP *Adaptive Double Plateaus Histogram Equalization*

Description

compute the paramters, t_down and t_up, and then apply double plateaus histogram equalization.

Usage

```
EqualizeADP(im, n = 5, N = 1000, range = c(0, 255),
  returnparam = FALSE)
```

Arguments

im	a grayscale image of class cimg
n	window size to determine local maximum
N	the number of subintervals of histogram
range	range of the pixel values of image. this function assumes that the range of pixel values of of an input image is [0,255] by default. you may prefer [0,1].
returnparam	if returnparam is TRUE, returns the computed parameters: t_down and t_up.

Value

a grayscale image of class cimg or a numericvector

Author(s)

Shota Ochi

References

Kun Liang, Yong Ma, Yue Xie, Bo Zhou ,Rui Wang (2012). A new adaptive contrast enhancement algorithm for infrared images based on double plateaus histogram equalization. Infrared Phys. Technol. 55, 309-315.

Examples

```
g <- grayscale(dogs)
layout(matrix(1:2, 1, 2))
plot(g, main = "Original")
EqualizeADP(g) %>% plot(main = "Contrast Enhanced")
```

EqualizeDP *Double Plateaus Histogram Equalization*

Description

enhance contrast of image by double plateaus histogram equalization.

Usage

```
EqualizeDP(im, t_down, t_up, N = 1000, range = c(0, 255))
```

Arguments

im	a grayscale image of class cimg
t_down	lower threshold
t_up	upper threshold
N	the number of subintervals of histogram
range	range of the pixel values of image. this function assumes that the range of pixel values of of an input image is [0,255] by default. you may prefer [0,1].

Value

a grayscale image of class cimg

Author(s)

Shota Ochi

References

Kun Liang, Yong Ma, Yue Xie, Bo Zhou ,Rui Wang (2012). A new adaptive contrast enhancement algorithm for infrared images based on double plateaus histogram equalization. Infrared Phys. Technol. 55, 309-315.

Examples

```
g <- grayscale(dogs)
layout(matrix(1:2, 1, 2))
plot(g, main = "Original")
EqualizeDP(g, 20, 186) %>% plot(main = "Contrast Enhanced")
```

`EqualizePiecewise` *Piecewise Affine Histogram Equalization*

Description

enhance contrast of image by piecewise affine histogram equalization

Usage

```
EqualizePiecewise(im, N, smax = 255, smin = 0, range = c(0, 255))
```

Arguments

`im`	a grayscale image of class cimg
`N`	number of subintervals of partition. N controls how the input gray levels will be mapped in the output image. if N is large, Piecewise Affine Equalization and Histogram Equalization are very similar.
`smax`	maximum value of slopes. if smax is small, contrast enhancement is suppressed.
`smin`	minimum value of slopes. if smin is large, contrast enhancement is propelled, and saturations occur excessively.
`range`	range of the pixel values of image. this function assumes that the range of pixel values of of an input image is [0,255] by default. you may prefer [0,1]. if you change range, you should change smax. one example is this (smax = range[2] - range[1]).

Value

a grayscale image of class cimg

Author(s)

Shota Ochi

References

Jose-Luis Lisani, Ana-Belen Petro, and Catalina Sbert, Color and Contrast Enhancement by Controlled Piecewise Affine Histogram Equalization, Image Processing On Line, 2 (2012), pp. 243-265. https://doi.org/10.5201/ipol.2012.lps-pae

Examples

```
dev.new()
par(mfcol = c(1,2))
boats_g <- grayscale(boats)
plot(boats_g, main = "Original")
EqualizePiecewise(boats_g, 10) %>% plot(., main = "Piecewise Affine Equalization")
```

`GetHue` *store hue of color image*

Description

store hue of color image

Usage

```
GetHue(imcol)
```

Arguments

`imcol`	a color image of class cimg

Value

a color image of class cimg

Author(s)

Shota Ochi

Examples

```
GetHue(boats)
```

Grayscale *compute average of RGB channels*

Description

compute average of RGB channels

Usage

```
Grayscale(imcol)
```

Arguments

imcol	a color image of class cimg

Value

a grayscale image of class cimg

Author(s)

Shota Ochi

Examples

```
Grayscale(boats) %>% plot
```

imagerExtra *imagerExtra: Extra Image Processing Library Based on Imager*

Description

imagerExtra is built on imager. imager by Simon Simon Barthelme provides an interface with CImg that is a C++ library for image processing. imager makes functions of CImg accessible from R and adds many utilities for accessing and working with image data from R. imagerExtra provides advanced functions for image processing based on imager.

OCR *Optical Character Recognition with tesseract*

Description

OCR and OCR_data are wrappers for ocr and ocr_data of tesseract package. You need to install tesseract package to use these functions.

Usage

```
OCR(imorpx, engine = tesseract::tesseract("eng"), HOCR = FALSE)

OCR_data(imorpx, engine = tesseract::tesseract("eng"))
```

Arguments

imorpx	a grayscale image of class cimg or a pixel set
engine	a tesseract engine. See the reference manual of tesseract for detail.
HOCR	if TRUE return results as HOCR xml instead of plain text

Author(s)

Shota Ochi

Examples

```
hello <- DenoiseDCT(papers, 0.01) %>% ThresholdAdaptive(., 0.1, range = c(0,1))
if (requireNamespace("tesseract", quietly = TRUE))
{
  OCR(hello) %>% cat
  OCR_data(hello)
}
```

papers *Photograph of a paper*

Description

This photograph was filmed by Shota Ochi.

Usage

```
papers
```

Format

an image of class cimg

RestoreHue *restore hue of color image*

Description

restore hue of color image

Usage

```
RestoreHue(im, hueim)
```

Arguments

im	a grayscale image of class cimg
hueim	a color image of class cimg

Value

a color image of class cimg

Author(s)

Shota Ochi

Examples

```
g <- Grayscale(boats)
hue <- GetHue(boats)
layout(matrix(1:2, 1, 2))
plot(g, main = "Original")
RestoreHue(g, hue) %>% plot(main="Resotred")
```

SegmentCV *Chan-Vese segmentation*

Description

iterative image segmentation with Chan-Vese model

Usage

```
SegmentCV(im, mu = 0.25, nu = 0, lambda1 = 1, lambda2 = 1,
  tol = 1e-04, maxiter = 500, dt = 0.5, initial, returnstep)
```

Arguments

im	a grayscale image of class cimg
mu	length penalty
nu	area penalty
lambda1	fit weight inside the cuve
lambda2	fit weight outside the curve
tol	convergence tolerance
maxiter	maximum number of iterations
dt	time step
initial	"interactive" or a grayscale image of class cimg. you can define initial condition as a rectangle shape interactively if initial is "interactive". If initial is a grayscale image of class cimg, pixels whose values are negative will be treated as outside of contour. pixels whose values are non-negative will be treated as inside of contour. checker board condition will be used if initial is not specified.
returnstep	a numeric vector that determines which result will be returned. 0 means initial condition, and 1 means the result after 1 iteration. final result will be returned if returnstep is not specified.

Value

a pixel set or a list of lists of numeric and pixel set

Author(s)

Shota Ochi

References

Pascal Getreuer (2012). Chan-Vese Segmentation. Image Processing On Line 2, 214-224.

Examples

```
layout(matrix(1:2, 1, 2))
g <- grayscale(dogs)
plot(g, main = "Original")
SegmentCV(g, lambda2 = 15) %>% plot(main = "Binarized")
```

SPE *Correct inhomogeneous background of image by solving Screened Poisson Equation*

Description

Correct inhomogeneous background of image by solving Screened Poisson Equation

Usage

```
SPE(im, lamda, s = 0.1, range = c(0, 255))
```

Arguments

im	a grayscale image of class cimg
lamda	this function corrects inhomogeneous background while preserving image details. lamda controls the trade-off. when lamda is too large, this function acts as an edge detector.
s	saturation percentage. this function uses BalanceSimplest. s is used as both sleft and sright. that's why s can not be over 50%.
range	this function assumes that the range of pixel values of of an input image is [0,255] by default. you may prefer [0,1].

Value

a grayscale image of class cimg

Author(s)

Shota Ochi

References

Jean-Michel Morel, Ana-Belen Petro, and Catalina Sbert, Screened Poisson Equation for Image Contrast Enhancement, Image Processing On Line, 4 (2014), pp. 16-29. https://doi.org/10.5201/ipol.2014.84

Examples

```
dev.new()
par(mfcol = c(1,2))
boats_g <- grayscale(boats)
plot(boats_g, main = "Original")
SPE(boats_g, 0.1) %>% plot(main = "Screened Poisson Equation")
```

ThresholdAdaptive *Local Adaptive Thresholding*

Description

Local Adaptive Thresholding

Usage

```
ThresholdAdaptive(im, k, windowsize = 17, range = c(0, 255))
```

Arguments

im	a grayscale image of class cimg
k	a numeric in the range [0,1]. when k is high, local threshold values tend to be lower. when k is low, local threshold value tend to be higher.
windowsize	windowsize controls the number of local neighborhood
range	this function assumes that the range of pixel values of of input image is [0,255] by default. you may prefer [0,1]. Note that range determines the max standard deviation. The max standard deviation plays an important role in this function.

Value

a pixel set

Author(s)

Shota Ochi

References

Faisal Shafait, Daniel Keysers, Thomas M. Breuel, "Efficient implementation of local adaptive thresholding techniques using integral images", Proc. SPIE 6815, Document Recognition and Retrieval XV, 681510 (28 January 2008)

Examples

```
layout(matrix(1:4, 2, 2))
plot(papers, main = "Original")
threshold(papers) %>% plot(main = "A variant of Otsu")
ThresholdAdaptive(papers, 0, range = c(0,1)) %>% plot(main = "local adaptive (k = 0)")
ThresholdAdaptive(papers, 0.2, range = c(0,1)) %>% plot(main = "local adaptive (k = 0.2)")
```

ThresholdFuzzy *Fuzzy Entropy Image Segmentation*

Description

automatic fuzzy thresholding based on particle swarm optimization

Usage

```
ThresholdFuzzy(im, n = 50, maxiter = 100, omegamax = 0.9,
  omegamin = 0.1, c1 = 2, c2 = 2, mutrate = 0.2, vmaxcoef = 0.1,
  intervalnumber = 1000, returnvalue = FALSE)
```

Arguments

im	a grayscale image of class cimg
n	swarm size
maxiter	maximum iterative time
omegamax	maximum inertia weight
omegamin	minimum inertia weight
c1	acceleration coefficient
c2	acceleration coefficient
mutrate	rate of gaussian mutation
vmaxcoef	coefficient of maximum velocity
intervalnumber	interval number of histogram
returnvalue	if returnvalue is TRUE, returns a threshold value. if FALSE, returns a pixel set.

Value

a pixsel set or a numeric

Author(s)

Shota Ochi

References

Linyi Li, Deren Li (2008). Fuzzy entropy image segmentation based on particle swarm optimization. Progress in Natural Science.

Examples

```
g <- grayscale(boats)
layout(matrix(1:2, 1, 2))
plot(g, main = "Original")
ThresholdFuzzy(g) %>% plot(main = "Fuzzy Thresholding")
```

`ThresholdML` *Multilevel Thresholding*

Description

Segments a grayscale image into several gray levels. Multilevel thresholding selection based on the artificial bee colony algorithm is used when thr is not a numeric vector. Preset parameters for fast computing is used when thr is "fast". Preset parameters for precise computing is used when thr is "precise". You can tune the parameters if thr is "manual". Also you can specify the values of thresholds by setting thr as a numeric vector.

Usage

```
ThresholdML(im, k, thr = "fast", sn = 30, mcn = 100, limit = 100,
  intervalnumber = 1000, returnvalue = FALSE)
```

Arguments

`im`	a grayscale image of class cimg
`k`	level of thresholding. k is ignored when thr is a numeric vector.
`thr`	thresholds, either numeric vector, or "fast", or "precise", or "manual".
`sn`	population size. sn is ignored except when thr is "manual".
`mcn`	maximum cycle number. mcn is ignored except when thr is "manual".
`limit`	abandonment criteria. limit is ignored except when thr is "manual".
`intervalnumber`	interval number of histogram. intervalnumber is ignored except when thr is "manual".
`returnvalue`	if returnvalue is TRUE, returns threshold values. if FALSE, returns a grayscale image of class cimg.

Value

a grayscale image of class cimg or a numeric vector

Author(s)

Shota Ochi

References

Ming-HuwiHorng (2011). Multilevel thresholding selection based on the artificial bee colony algorithm for image segmentation. Expert Systems with Applications.

Examples

```
g <- grayscale(boats)
ThresholdML(g, k = 2) %>% plot
```

ThresholdTriclass *Iterative Triclass Thresholding*

Description

compute threshold value by Iterative Triclass Threshold Technique

Usage

```
ThresholdTriclass(im, stopval = 0.01, repeatnum, intervalnumber = 1000,
  returnvalue = FALSE)
```

Arguments

im	a grayscale image of class cimg
stopval	value to determine whether stop iteration of triclass thresholding or not. Note that if repeat is set, stop is ignored.
repeatnum	number of repetition of triclass thresholding
intervalnumber	interval number of histogram
returnvalue	if returnvalue is TRUE, ThresholdTriclass returns threshold value. if FALSE, ThresholdTriclass returns pixset.

Value

a pixel set or a numeric

Author(s)

Shota Ochi

References

Cai HM, Yang Z, Cao XH, Xia WM, Xu XY (2014). A New Iterative Triclass Thresholding Technique in Image Segmentation. IEEE TRANSACTIONS ON IMAGE PROCESSING.

Examples

```
g <- grayscale(boats)
layout(matrix(1:4, 2, 2))
plot(boats, main = "Original")
plot(g, main = "Grayscale")
threshold(g) %>% plot(main = "A Variant of Otsu")
ThresholdTriclass(g) %>% plot(main = "Triclass")
```

Index