Package 'RImagePalette'

August 29, 2016

Type Package

Title Extract the Colors from Images

Version 0.1.1

Date 2016-01-05

Author Joel Carlson

Maintainer Joel Carlson <jnkcarlson@gmail.com>

Description A pure R implementation of the median cut algorithm. Extracts the dominant colors from an image, and turns them into a scale for use in plots or for fun!

License MIT + file LICENSE

LazyData TRUE

Depends R (>= 2.1.0)

Imports ggplot2

Suggests testthat, scales, jpeg, png

NeedsCompilation no

Repository CRAN

Date/Publication 2016-01-05 21:01:52

R topics documented:

display_image *Display color image*

Description

Convenience wrapper to create a raster image of the image you wish to extract the palette from.

Usage

```
display_image(image)
```

Arguments

image	Matrix The image from which the palette will be extracted from. Should be a 3 (or more) dimensional matrix. The output of functions such as `readJPG()` are suitable as `image`.

Value

A raster image in the plot window.

Examples

```
img <- jpeg::readJPEG(system.file("img", "Rlogo.jpg", package="jpeg"))
display_image(img)
```

display_palette *Display color palette*

Description

Displays the created palette as a barchart with axis labels representing hex values of the colors. A more attractive method for doing so would be to use `show_cols()` from `library(scales)`.

Usage

```
display_palette(palette)
```

Arguments

palette	Vector The output of `image_palette`.

Value

A plot of the colors extracted from the image

See Also

```
scales::show_cols()
```

Examples

```
img <- jpeg::readJPEG(system.file("img", "Rlogo.jpg", package="jpeg"))
display_palette(image_palette(img, n=5))
```

`image_pal` *Image palette*

Description

Image palette function

Usage

```
image_pal(image, choice = mean, volume = FALSE)
```

Arguments

`image`	Matrix The image from which the palette will be extracted from. Should be a 3 (or more) dimensional matrix. The output of a function such as `readJPG()` or `readPNG()` are suitable as `image`.
`choice`	Function Defines how the color will be chosen from the final color cubes. The default choice is to take the `mean` value of the image cube, but other choices may return a subjectively superior scale. Try `median`, or `min`, or `max`, or whatever summary statistic suits your fancy.
`volume`	Logical volume controls the method for choosing which color cube to split at each iteration of the algorithm. The default choice (when `volume = FALSE`) is to choose the cube based on which cube contains the largest extent (that is, the largest range of some color). When `volume = TRUE`, the cube with the largest volume is chosen to split. Occasionally, setting to `TRUE` returns a better palette.

Examples

```
img <- jpeg::readJPEG(system.file("img", "Rlogo.jpg", package="jpeg"))
display_image(img)
scales::show_col(image_pal(img)(10))
```

image_palette *Create image palette*

Description

Image palette function

Usage

```
image_palette(image, n, choice = mean, volume = FALSE)
```

Arguments

image	Matrix The image from which the palette will be extracted from. Should be a 3 (or more) dimensional matrix. The output of a function such as `readJPG()` or `readPNG()` are suitable as `image`.
n	Integer The number of discrete colors to be extracted from the image.
choice	Function Defines how the color will be chosen from the final color cubes. The default choice is to take the `mean` value of the image cube, but other choices may return a subjectively superior scale. Try `median`, or `min`, or `max`, or whatever summary statistic suits your fancy.
volume	Logical volume controls the method for choosing which color cube to split at each iteration of the algorithm. The default choice (when `volume = FALSE`) is to choose the cube based on which cube contains the largest extent (that is, the largest range of some color). When `volume = TRUE`, the cube with the largest volume is chosen to split. Occasionally, setting to `TRUE` returns a better palette.

Details

Uses the median cut algorithm to create `n` discrete colors based on colors present in an image. See `median_cut` for more details.

See Also

`median_cut`

Examples

```
img <- jpeg::readJPEG(system.file("img", "Rlogo.jpg", package="jpeg"))
display_image(img)
scales::show_col(image_palette(img, n=5))
```

median_cut *The median cut algorithm*

Description

Cut an rgb cube into two color cubes, each with as imilar number of elements.

Usage

```
median_cut(image, vbox, iter = 1)
```

Arguments

image	List An image in list form, with three components: red, green, blue
vbox	List The output of vbox() for the given image. A list of image parameters ("min", "max", "med", "ext" and "volume")
iter	Integer The number attached to the names of the two new images.

Details

Represents the rgb colorspace as a cube, with side lengths based on the red, green, and blue extents (difference between maximum and minimum within-color values).

The algorithm takes the side with the largest extent (extent information is passed in via the vbox() parameter), and splits the cube along the median value.

Both halves of the cube are then returned.

Value

Two new images in a list, each separated into rgb components

See Also

vbox image_palette

quantize_image *Quantize image*

Description

Quantize image into discrete colors using the median cut algorithm

Usage

```
quantize_image(image, n, ...)
```

Arguments

image	Matrix The image from which the palette will be extracted from. Should be a 3 (or more) dimensional matrix. The output of a function such as `readJPG()` or `readPNG()` are suitable as `image`.
n	Integer The number of discrete colors to be extracted from the image.
...	Pass any of the arguments for `image_palette`

Details

Note: This function is extremely slow for large images. Takes up to 20 seconds for 500x500 image on a desktop with 2.7GHz processor and 4Gb ram.

See Also

`image_palette`

Examples

```
img <- jpeg::readJPEG(system.file("img", "Rlogo.jpg", package="jpeg"))
quant_img <- quantize_image(img, n=3)
display_image(img)
display_image(quant_img)
```

---

scale_color_image *Image color scales*

Description

Uses the image color scale.

Usage

```
scale_color_image(..., image, n = 3, choice = mean, volume = FALSE,
  discrete = TRUE)

scale_colour_image(..., image, n = 3, choice = mean, volume = FALSE,
  discrete = TRUE)

scale_fill_image(..., image, n = 3, choice = mean, volume = FALSE,
  discrete = TRUE)
```

Arguments

...	parameters to discrete_scale or scale_fill_gradientn
image	Matrix The image from which the palette will be extracted from. Should be a 3 (or more) dimensional matrix. The output of a function such as readJPG() or readPNG() are suitable as image.
n	For continuous color scales, you may optionally pass in an integer, n. This allows some control over the scale, if n is too large the scale has too many colors and ceases to be meaningul. n = 3 to n = 5 is recommended.
choice	Function Defines how the color will be chosen from the final color cubes. The default choice is to take the mean value of the image cube, but other choices may return a subjectively superior scale. Try median, or min, or max, or whatever summary statistic suits your fancy.
volume	Logical volume controls the method for choosing which color cube to split at each iteration of the algorithm. The default choice (when volume = FALSE) is to choose the cube which contains the largest extent (that is, the largest range of some color). When volume = TRUE, the cube with the largest volume is chosen to split. Occasionally, setting to TRUE returns a better palette.
discrete	generate a discrete palette? (default: FALSE - generate continuous palette)

Details

For discrete == TRUE (the default) the function will return a discrete_scale with the plot-computed number of colors. All other arguments are as to scale_fill_gradientn or scale_color_gradientn.

See image_palette for more information on the color scale.

See Also

median_cut image_palette vbox display_image

Examples

```
library(ggplot2)

# ripped from the pages of ggplot2
your_image <- jpeg::readJPEG(system.file("img", "Rlogo.jpg", package="jpeg"))
display_image(your_image)

#Discrete scale example
p <- ggplot(mtcars, aes(wt, mpg))
p + geom_point(size=4, aes(colour = factor(cyl))) +
    scale_color_image(image = your_image) +
    theme_bw()

#Continuous scale example
dsub <- subset(diamonds, x > 5 & x < 6 & y > 5 & y < 6)
dsub$diff <- with(dsub, sqrt(abs(x-y))* sign(x-y))
d <- ggplot(dsub, aes(x, y, colour=diff)) + geom_point()
d + scale_color_image(image = your_image, discrete=FALSE) + theme_bw()
```

switch_colors *Swap Colors in an Image*

Description

Swap the palette of an image!

Usage

```
switch_colors(target_image, source_image, source_colors = 3,
  smoothness = 100, ...)
```

Arguments

target_image	Matrix The image you wish to transfer colors into. The output from readJPEG is of suitable format.
source_image	Matrix The image you wish to transfer colors from.
source_colors	Integer The number of colors you wish to extract from the source image.
smoothness	Integer The source colors are interpolated such that the image doesn't appear blocky. The value of smoothness determines how many values are interpoloated between the source_colors. That is, smoothness determines the length of the palette used. Higher values return smoother images.
...	Pass any of the arguments for image_palette

Value

The image, but with swapped colors!

Examples

```
#Trivial example of using only 5 dominant colors
# from an image to recolor itself
img1 <- jpeg::readJPEG(system.file("img", "Rlogo.jpg", package="jpeg"))
img2 <- jpeg::readJPEG(system.file("img", "Rlogo.jpg", package="jpeg"))
switch_colors(img1, img2, source_colors=5, smoothness=20)
```

vbox *Volume box*

Description

Extract minimum, maximum, median, extent, and volume information from red, green, and blue color channels.

Usage

```
vbox(im)
```

Arguments

im	List An image in list form, with three components: red, green, blue

Details

For passing to `median_cut()`.

Value

A list containing the minimum, maximum, median, extent, and volume of each component of the image

See Also

`median_cut` `image_palette`

Index