Package 'imgpalr'

November 28, 2019

Title Create Color Palettes from Images

Version 0.3.0

Description Provides ability to create color palettes from image files.
It offers control over the type of color palette to derive from an image (qualitative, sequential or divergent) and other palette properties.
Quantiles of an image color distribution can be trimmed.
Near-black or near-white colors can be trimmed in RGB color space independent of trimming brightness or saturation distributions in HSV color space.
Creating sequential palettes also offers control over the order of HSV color dimensions to sort by.
This package differs from other related packages like 'RImagePalette' in approaches to quantizing and extracting colors in images to assemble color palettes
and the level of user control over palettes construction.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/leonawicz/imgpalr

BugReports https://github.com/leonawicz/imgpalr/issues

Suggests testthat, covr, bmp, png, magick

Language en-US

RoxygenNote 7.0.1

Imports downloader, farver, tibble, magrittr, dplyr, jpeg

NeedsCompilation no

Author Matthew Leonawicz [aut, cre] (<https://orcid.org/0000-0001-9452-2771>)

Maintainer Matthew Leonawicz <matt_leonawicz@esource.com>

Repository CRAN

Date/Publication 2019-11-28 11:50:02 UTC

R topics documented:

`image_load` *Load PNG, JPG, BMP or GIF*

Description

Load PNG, JPG, BMP or GIF from disk or URL.

Usage

```
image_load(file)
```

Arguments

`file`	character, file name. A local file or URL. Extension must be one of png, jpg, jpeg, bmp or gif.

Details

The function will prompt you of the need to install a format-specific package if not installed and needed for the given `file` extension; `png`, `bmp`, `magick` (for GIF). `jpeg` is already imported for purpose of running examples.

Value

an RBG array

Examples

```
x <- paste0(system.file(package = "imgpalr"), "/blue-yellow.",
  c("jpg", "png", "bmp", "gif"))
str(image_load(x[1]))
if(require(png)) str(image_load(x[2]))
if(require(bmp)) str(image_load(x[3]))
if(require(magick)) str(image_load(x[4]))
```

image_pal *Create a color palette from an image*

Description

Derive qualitative, sequential and divergent color palettes from an image on disk or at a URL.

Usage

```
image_pal(
  file,
  n = 9,
  type = c("qual", "seq", "div"),
  k = 100,
  bw = c(0, 1),
  brightness = c(0, 1),
  saturation = c(0, 1),
  seq_by = "hsv",
  div_center = "#FFFFFF",
  seed = NULL,
  plot = FALSE,
  labels = TRUE,
  label_size = 1,
  label_color = "#000000",
  keep_asp = TRUE,
  quantize = FALSE
)
```

Arguments

file	character, file path or URL to an image.
n	integer, number of colors.
type	character, type of palette: qualitative, sequential or divergent ("qual", "seq", or "div").
k	integer, the number of k-means cluster centers to consider in the image. See details.
bw	a numeric vector of length two giving the lower and upper quantiles to trim trim near-black and near-white colors in RGB space.
brightness	as above, trim possible colors based on brightness in HSV space.
saturation	as above, trim possible colors based on saturation in HSV space.
seq_by	character, sort sequential palette by HSV dimensions in a specific order, e.g., "hsv", "svh". See details.
div_center	character, color used for divergent palette center, defaults to white.
seed	numeric, set the seed for reproducible results.

plot	logical, plot the palette.
labels	logical, show hex color values in plot.
label_size	numeric, label size in plot.
label_color	text label color.
keep_asp	logical, adjust rectangles in plot to use the image aspect ratio.
quantize	logical, quantize the reference thumbnail image in the plot using the derived color palette. See image_quantmap.

Details

Ordering colors is a challenging problem. There are many ways to do it; none are perfect. Color is a multi-dimensional property; any reduction to a a one dimensional color spectrum necessarily removes information.

Creating a sequential palette from an arbitrary image that contains several hues, at different saturation and brightness levels, and making a palette that looks sequential is particularly problematic. This function does a decent job of creating qualitative, sequential and divergent palettes from images, but additional tweaking of function arguments is needed on a case by case basis. This can include trimming the extreme values of the color distribution in terms of brightness, saturation and presence of near-black/white colors as pre-processing steps. There is also variation in possible palettes from a given image, depending on the image complexity and other properties, though you can set the random seed for reproducibility.

The number of k-means centers k defines the maximum number of unique colors to consider in the image for color binning prior to palette construction. This is different from n, the number of colors are desired in the derived palette. It is limited by the number of unique colors in the image. Larger k may allow for better palette construction under some conditions, but takes longer to run. k applies to sequential and qualitative palettes, but not divergent palettes.

Value

character vector of hex colors, optionally draws a plot

Trimming color distribution

Some pre-processing can be done to limit undesirable colors from ending up in a palette. bw specifically drops near-black and near-white colors as soon as the image is loaded by looking at the average values in RGB space. brightness and saturation trimming are applied subsequently to trim lower and upper quantiles of the HSV value and saturation, respectively. If you have already trimmed black and white, keep in mind these two arguments will trim further from what remains of the color distribution.

Choosing appropriate palette type

Keep in mind that many images simple do not make sense to try to derive sensible color palettes from. For images that do lend themselves to a useful color palette derivation, some may only make sense to consider for a divergent palette, or an increasing/decreasing sequential palette, or only a qualitative palette if there are too many colors that are difficult to order. For divergent palettes in particular, it is recommended to trim white, e.g. bw = c(0,0.9), depending on the white space of a given image, since the divergent palettes are centered on white.

Sorting sequential palettes

seq_by = "hsv" orders the final palette by hue, then saturation, then value (brightness). This default is not meant to be ideal for all images. It work better in cases where sequential palettes may contain several distinct hues, but not much variation in saturation or brightness. However, for example, palettes derived from an image with relatively little variation in hue may appear more sorted to the human eye if ordered by hue last using "svh" or "vsh", depending on whether you want the palette to appear to transition more from lower saturation or lower brightness to the predominant hue.

See Also

image_quantmap

Examples

```
set.seed(1)
x <- system.file("blue-yellow.jpg", package = "imgpalr")

# Focus on bright, saturated colors for divergent palette:
image_pal(x, n = 3, type = "div",
  saturation = c(0.75, 1), brightness = c(0.75, 1), plot = TRUE)


image_pal(x, n = 5, type = "seq", k = 2, saturation = c(0.5, 1),
  brightness = c(0.25, 1), seq_by = "hsv")
```

---

image_quantmap *Quantize an image using an existing color palette*

Description

Quantize image colors by mapping all pixels to the nearest color in RGB space, respectively, given an arbitrary palette.

Usage

```
image_quantmap(
  file,
  pal,
  pal2 = NULL,
  k = 100,
  plot = FALSE,
  show_pal = TRUE,
  labels = TRUE,
  label_size = 1,
  label_color = "#000000",
  keep_asp = TRUE
)
```

Arguments

file	if character, file path or URL to an image. You can also provide an RGB array from an already loaded image file.
pal	character, vector of hex colors, the color palette used to quantize the image colors.
pal2	character, optional vector of hex colors, same length as pal. After quantizing image to pal, you can subsequently remap pal to pal2.
k	integer, the number of k-means cluster centers to consider in the image. See details.
plot	logical, plot the palette with quantized image reference thumbnail. If FALSE, only return the RGB array.
show_pal	logical, show the palette like with image_pal. If FALSE, plot only the image; all subsequent arguments ignored.
labels	logical, show hex color values in plot.
label_size	numeric, label size in plot.
label_color	text label color.
keep_asp	logical, adjust rectangles in plot to use the image aspect ratio.

Details

The palette pal does not need to be related to the image colors. Each pixel will be assigned to whichever color in pal that it is nearest to in RGB space. You can use pal2 to remap to arbitrary colors after quantizing. This function returns the new RGB array. You can plot a preview just like with image_pal using plot = TRUE. The number of k-means centers k is for binning image colors prior to mapping the palette pal. It is limited by the number of unique colors in the image. Larger k provides more binned distances between image colors and palette colors, but takes longer to run.

Value

an RGB array with values ranging from 0 to 1

See Also

image_pal

Examples

```
x <- system.file("blue-yellow.jpg", package = "imgpalr")
pal <- c("black", "navyblue", "dodgerblue", "yellow")
pal2 <- c("darkred", "darkgreen", "tomato", "orange")

a <- image_quantmap(x, pal, k = 7, plot = TRUE)
str(a)

a <- image_quantmap(x, pal, pal2, k = 7, plot = TRUE)
```

imgpalr *imgpalr: Create color palettes from images*

Description

The imgpalr package is used for generating color palettes from image files. It offers control over the type of color palette to derive from an image (qualitative, sequential or divergent) and other palette properties. Quantiles of an image color distribution can be trimmed. Near-black or near-white colors can be trimmed in RGB color space independent of trimming brightness or saturation distributions in HSV color space. Creating sequential palettes also offers control over the order of HSV color dimensions to sort by.

Index