pyaesthetics Documentation

Release 0.0.7

Giulio Gabrieli

Mar 30, 2023

Contents:

CHAPTER 1

Analysis

CHAPTER 2

Brightness analysis

This module contains function to evaluate the brightness of an image. It includes a converter for sRGB to RGB, evaluation of relative luminance according to BT.709 and BT.601

@author: Giulio Gabrieli

`brightness.`**`relativeLuminance_BT601`**`(`*img*`)`

This function evaluates the brightness of an image by mean of Y, where Y is evaluated as:

Y = 0.587G + 0.114B + 0.299R B = mean(Y)

Parameters **`img`** (*`numpy.ndarray`*) – image to analyze, in RGB

Returns mean brightness

Return type float

`brightness.`**`relativeLuminance_BT709`**`(`*img*`)`

This function evaluates the brightness of an image by mean of Y, where Y is evaluated as:

Y = 0.7152G + 0.0722B + 0.2126R B = mean(Y)

Parameters **`img`** (*`numpy.ndarray`*) – image to analyze, in RGB

Returns mean brightness

Return type float

`brightness.`**`sRGB2RGB`**`(`*img*`)`

this function converts a sRGB img to linear RGB values.

It loops through each pixel, and apply a conversion to pass from sRGB to linear RGB value.

Parameters **`img`** (*`numpy.ndarray`*) – image to analyze, in sRGB

Returns image to analyze, in RGB

Rtyipe numpy.ndarray

CHAPTER 3

Colorfulness analysis

This module contains function to evaluate the colorfulness of an image in both the HSV and RGB color spaces.

@author: Giulio Gabrieli

`colorfulness.`**`colorfulnessHSV`**`(`*img*`)`

This function evaluates the colorfulness of a picture using the formula described in Yendrikhovskij et al., 1998. Input image is first converted to the HSV color space, then the S values are selected. Ci is evaluated with a sum of the mean S and its std, as in:

Ci = mean(Si)+ std(Si)

Parameters **`img`** (*`numpy.ndarray`*) – image to analyze, in RGB

Returns colorfulness index

Return type float

`colorfulness.`**`colorfulnessRGB`**`(`*img*`)`

This function evaluates the colorfulness of a picture using Metric 3 described in Hasler & Suesstrunk, 2003. Ci is evaluated with as:

Ci =std(rgyb) + 0.3 mean(rgyb) [Equation Y] std(rgyb) = sqrt(std(rg)^2+std(yb)^2) mean(rgyb) = sqrt(mean(rg)^2+mean(yb)^2) rg = R - G yb = 0.5(R+G) - B

Parameters **`img`** (*`numpy.ndarray`*) – image to analyze, in RGB

Returns colorfulness index

Return type float

`colorfulness.`**`sRGB2RGB`**`(`*img*`)`

this function converts a sRGB img to linear RGB values.

Parameters **`img`** (*`numpy.ndarray`*) – image to analyze, in sRGB

Returns image to analyze, in RGB

Rtyipe numpy.ndarray

CHAPTER 4

Color Detection

This module contains function to evaluate the presence of different colors of an image. It uses the 16 basic colors defined in the W3C specifications.

@author: Giulio Gabrieli

`colorDetection.`**`getColorsW3C`**`(`*img*, *plot=False*`)`

This functions is used to get a simplified color palette (W3C siteens basic colors).

F = 255 C0 = 192 80 = 128

Parameters

- **`img`** (*`numpy.ndarray`*) – image to analyze in RGB
- **`plot`** (*`boolean`*) – whether to plot or not the results

Returns percentage distribution of colors according to the W3C sixteens basic colors

Return type list of shape 16x2, where x[0] is the color name and x[1] the percentage of pixels most similar to that color in the image

CHAPTER 5

Face Detection

This is an entrypoint for automatic analysis of a website.

Created on Mon Apr 16 22:40:46 2018

@author: giulio

`faceDetection.`**`getFaces`**`(`*img*, *plot=False*`)`

This functions uses CV2 to get the faces in a pciture.

Parameters

- **`img`** (*`numpy.ndarray`*) – image to analyze in RGB
- **`plot`** (*`boolean`*) – whether to plot or not the results

CHAPTER 6

QuadTree Decomposition analysis

This file contains class and functions to perform a Quadratic Tree decomposition of an image and to visually inspect it.

Created on Mon Apr 16 11:49:45 2018

@author: giulio

class `quadTreeDecomposition.`**`quadTree`**(*img*, *minStd*, *minSize*)

This class is used to perfrom a QuadTree decomposition of an image.

During initialization, QuadTree decomposition is done and result are store in self.blocks as a list containing [x,y,height, width,Std].

To visualize the results, use plot().

`plot`(*edgecolor='red'*, *facecolor='none'*, *linewidth=1*)

This function is used to generate a graphical representation of the QuadTree decomposition.

Parameters

- **`edgecolor`** (*`string`*) – color of the rectangles, default is red
- **`facecolor`** (*`string`*) – color used for rectangles fills. Default is none.
- **`linewidth`** – width in px of the rectangles' borders. Default is 1.

Returns plot with image and leaves of the quadTree Decomposition

`quadTreeDecomposition`(*img*, *x*, *y*, *minStd*, *minSize*)

This function evaluate the mean and std of an image, and decides Whether to perform or not other 2 splits of the leave.

Parameters

- **`img`** (*`numpy.ndarray`*) – img to analyze
- **`x`** (*`int`*) – x offset of the leaves to analyze
- **`Y`** (*`int`*) – Y offset of the leaves to analyze

MinStd Std threshold for subsequent splitting

MinSize Size threshold for subsequent splitting, in pixel

CHAPTER 7

Symmetry analysis

This module contains functions to compute the degree of symmetry of an image. - Symmetry by QuadTree Decomposition

Created on Mon Apr 16 11:49:45 2018

@author: giulio

symmetry.**getSymmetry**(*img*, *minStd*, *minSize*, *plot=False*)

This function returns the degree of symmetry (0-100) between the left and right side of an image

Parameters **img** (*numpy.ndarray*) – img to analyze

MinStd Std threshold for subsequent splitting

MinSize Size threshold for subsequent splitting, in pixel

Returns degree of vertical symmetry

Return type float

class symmetry.**quadTree**(*img*, *minStd*, *minSize*)

This class is used to perfrom a QuadTree decomposition of an image.

During initialization, QuadTree decomposition is done and result are store in self.blocks as a list containing [x,y,height, width,Std].

To visualize the results, use plot().

plot(*edgecolor='red'*, *facecolor='none'*, *linewidth=1*)

This function is used to generate a graphical representation of the QuadTree decomposition.

Parameters

- **edgecolor** (*string*) – color of the rectangles, default is red
- **facecolor** (*string*) – color used for rectangles fills. Default is none.
- **linewidth** – width in px of the rectangles' borders. Default is 1.

Returns plot with image and leaves of the quadTree Decomposition

`quadTreeDecomposition`(*img*, *x*, *y*, *minStd*, *minSize*)

This function evaluate the mean and std of an image, and decides Whether to perform or not other 2 splits of the leave.

Parameters

- **`img`** (*`numpy.ndarray`*) – img to analyze
- **`x`** (*`int`*) – x offset of the leaves to analyze
- **`Y`** (*`int`*) – Y offset of the leaves to analyze

MinStd Std threshold for subsequent splitting

MinSize Size threshold for subsequent splitting, in pixel

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index