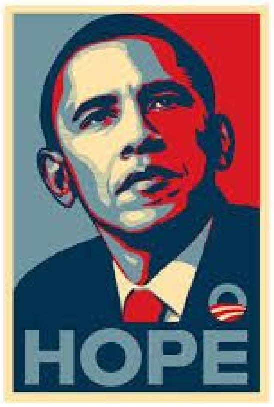
Obamicon

Language: Python
Topics: Lists, Functions, Tuples, Libraries, Loops

Overview



In this project you are going to learn how to create a colorization filter similar to the picture on the right. This filter re-colorizes a photo by pixel intensity. Complete this project in groups of two or three. Use a photo of yourself or a photo from the internet.

History

Artist Shepard Fairey created the image shown on the left during the 2008 Presidential Campaign. It is one of the most recognizable images of the campaign.

Instructions

You can get a similar effect if you look at each pixel's RGB, or red green blue, values. RGB values work like light. If a color has high values for all three colors, it is close to white. If there are low values, it's close to black. If you add all the values together, you get a number that correlates to how "light" the color is. Lighter colors get colorized yellow. Medium/light colors get coded as light blue. Medium/dark colors are red and very dark colors get coded as dark blue.

Start with the starter code.

Summary

- Get the intensity of each pixel by adding the red, green and blue values.
- If the pixel has low intensity (<182), color it dark blue.
- If the pixel is medium/low intensity (between 182 and 364), color it red.
- If the pixel is medium/high intensity (between 364 and 546), color it light blue.
- If the pixel is high intensity (>546) color it yellow.

- The RGB codes for the colors are as follows:
    -
    - `darkBlue = (0, 51, 76)`
    - `red = (217, 26, 33)`
    - `lightBlue = (112, 150, 158)`
    - `yellow = (252, 227, 166)`

Requirements

- Load in a picture to your program using PIL
- Extract pixel data from the image
- Create a new image that re-colorizes each pixel
- Use at least one function
- Save the resulting image

Extensions

If you finish early, try one of these extension activities.

- Recolor your image using a different color pallet.
- Take user input to select the photo that should be recolored.
- Take user input to select the color pallet that should be used to recolor the image.
- Define a function that will take a the file name for a photo and a specific color pallet and recolor the photo.
- Watch this video about Blur Filters and try to code this.
- Innovate in any way you'd like to!
- Take a selfie and share the filtered version on Loop

Reference

To accomplish this task, you will need an image library. We will use Pillow (PIL or Python Image Library).

Example:

```
from PIL import Image
im = Image.open("puppy.jpg")
im.rotate(45).show()
new_image = Image.new(im.mode, im.size)
```

```
            new_image.save("output.jpg", "jpeg")
```

This example imports the library, opens a picture and shows a rotated version of that picture then saves the original picture.

> **Note**: you must run this program from the same directory (folder) as the image file OR call the file using an absolute path. The output.jpg file will be stored in the same folder.

Selected Documentation

Below are a couple of functions that will help you complete this project. If you are interested, feel free to check out the complete documentation.

getdata

```
im.getdata() ⇒ sequence
```

Returns the contents of an image as a sequence object containing pixel values. The sequence object is flattened, so that values for line one follow directly after the values of line zero, and so on.

Note that the sequence object returned by this method is an internal PIL data type, which only supports certain sequence operations, including iteration and basic sequence access. To convert it to an ordinary sequence (e.g. for printing), use list(im.getdata()).

putdata

```
im.putdata(data)
im.putdata(data, scale, offset)
```

Copy pixel values from a sequence object into the image, starting at the upper left corner (0, 0). The scale and offset values are used to adjust the sequence values:

```
 pixel = value * scale + offset
```

If the scale is omitted, it defaults to 1.0. If the offset is omitted, it defaults to 0.0.

new

```
Image.new(mode, size) ⇒ image
Image.new(mode, size, color) ⇒ image
```

Creates a new image with the given mode and size. Size is given as a (width, height)-tuple, in pixels. The color is given as a single value for single-band images, and a tuple for multi-band images (with one value for each band). In 1.1.4 and later, you can also use color names (see the **ImageColor** module documentation for details) If the color argument is omitted, the image is filled with zero (this usually corresponds to black). If the color is None, the image is not initialised. This can be useful if you're going to paste or draw things in the image.

```
from PIL import Image
im = Image.new("RGB", (512, 512), "white")
```

size #

```
im.size ⇒ (width, height)
```