Team Project 2 : Posterize a Photo

Team name : Imposter(#03)
Members : Joonghyun Cho(20160036) SeungHun you(20175429)
Sungmin Park(20164256) Seokwon Yoon(20174089)

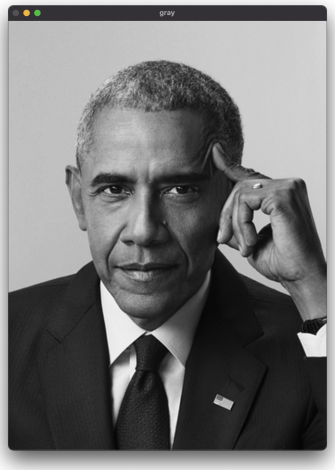
① Outline of program

The goal of our program is to change an image to a pop-art style poster. The input will be any kind of image with a person inside. After the conversion, the output result will become a good-looking poster. With a various pool of color palettes, there will be different colored posters.

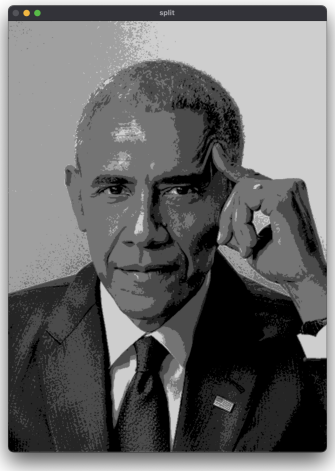
② Design of program

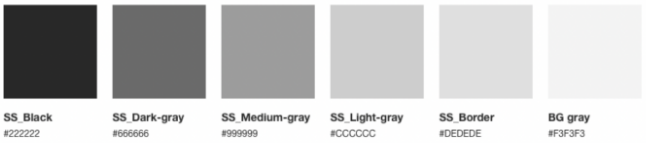
The implementation of the program consists of 4 steps.

1. Change the input image into a gray-scale image. In order to make the job easier, the input image will be turned into gray-scale and be resized into a certain size. We will color all segements with a new color so original rgb values are useless. So turn it into grayscale image which only contains brightness information



2. Split the given image into 6 distinct regions by the brightness level. The brightness level will be calculated by the cumulative number of pixels. By doing so, images which are bright or dark on average can be converted nicely. The splitted regions will categorize the image pixels and will make the image ready to be recolored as a poster on the last step.





3. Soften Image with the blurring mask. The edge parts of regions are a little bit massy in splitted image above. To make the edge line clear, we use a blur mask to get rid of stains in the middle of each region.

4. The classified regions will be colored with a diverse pool of palettes. With a colorful set of palettes the image will turn into a nice looking art.

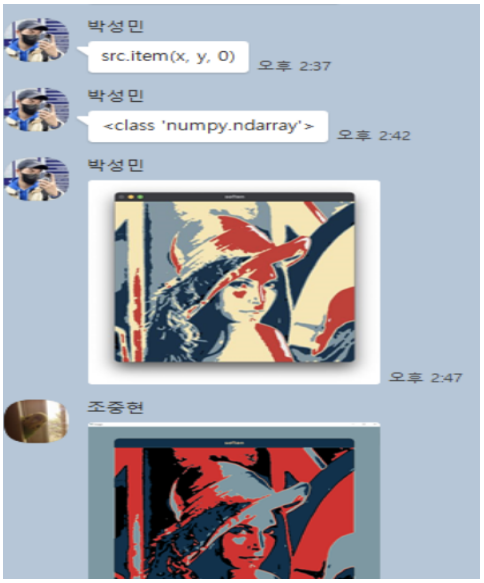
③ The spending time

Our team passionately participated with the team project. Thanks to that, we efficiently used our time following amazing results. We spent a total of 20 hours in 4days.

④ Design process of program and minutes(Picture and Capture image of Kakao Talk, etc.)

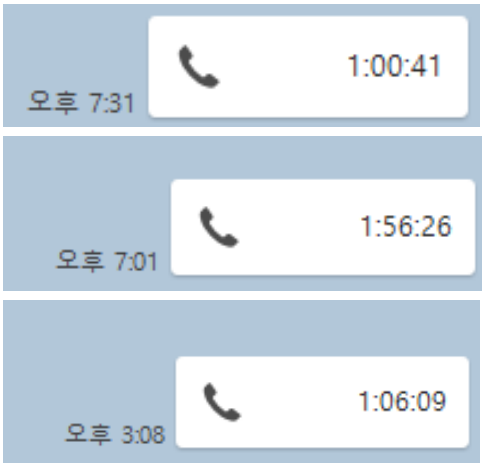
Decide on a Topic

merge personal result



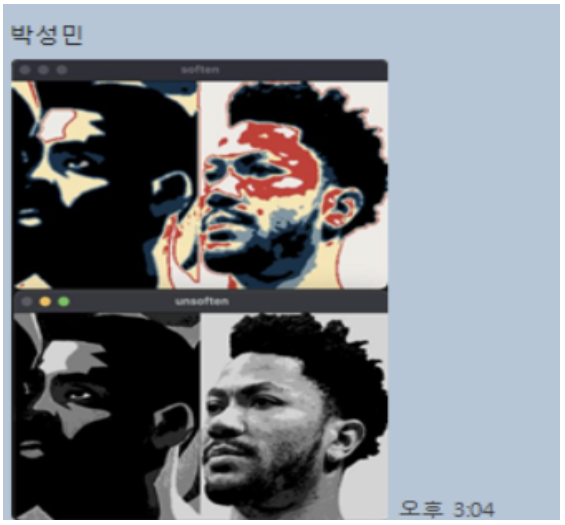
discussing prototype of color palette

every meeting was held on voice talk



discussion of result Image

merging sub projects



prototype of converted Image

⑤ Implementation

```
import cv2
import numpy as np
import funct

img = cv2.imread("./image/shirtman.png")
img = funct.image2gray(img)
img = cv2.resize(img,(int(((800*img.shape[1])/img.shape[0])),800), interpolation = cv2.INTER_CUBIC)
img = funct.devideLevel(img)
img = funct.softenImage(img)
```

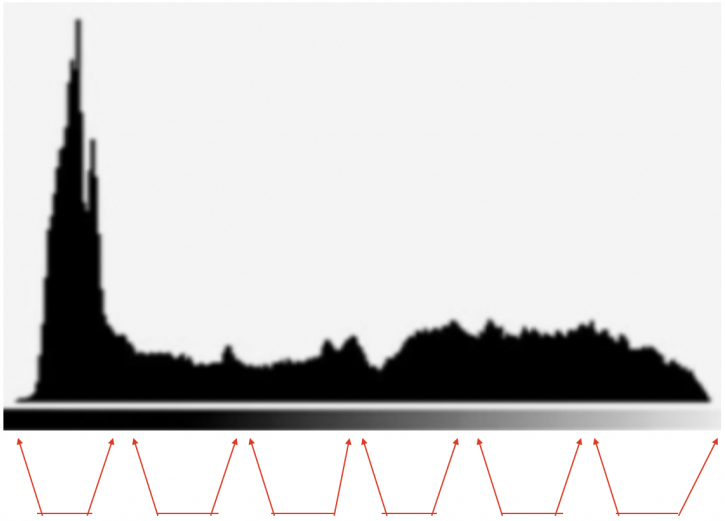
The sequence of processing images starts with reading the image file. We used the opencv library. After that, we change the rgb image into a grayscale image. To make fixed mask size works generally on various images, resize it into optimal size.
Divide all the segments of the image by their bright level and color them. Finally, we smooth them to erase tiny fractions of colored area. As a result, a new posterized image will be created.

input grayscale divide segments

```
def image2gray(src):

    dst = src.copy()

    for y in range(0, height):
        for x in range(0, width):
            grayColor = (src.item(y, x,0)+src.item(y, x,1)+src.item(y, x,2))/3
            dst[y, x] = round(grayColor)

    return dst
```

This function converts original images into grayscale images.
input is a colored image and the output will be a grayscale image of the input.

```
def setBoundaryByHistogram(src):
    height, width, color = src.shape

    pixelNumOfEachBoundary = height * width / 6

    stack = [0 for i in range(256)]
    maxValueOfEachBoundary = []

    for y in range(0, height):
        for x in range(0, width):
            stack[src.item(y, x, 0)] += 1

    countNumb = 0
    for i in range(0, 255):
        countNumb += stack[i]
        if countNumb >= pixelNumOfEachBoundary :
            countNumb -= pixelNumOfEachBoundary
            maxValueOfEachBoundary.append(i)
            print(i)

    return maxValueOfEachBoundary
```



This function returns the array of each region's max intensity value. By skimming through the image and counting the number of each intensity, the stack array is filled. While cumulating the array's elements from the low intensity pixel, if the sum value is over ⅙ of the number of whole pixels at certain intensity level, the level becomes the max intensity of its region. Repeat until finding 5 max intensities to divide into 6 regions.

```python
def devideLevel(src):

    height, width, color = src.shape

    maxValueOfEachBoundary = setBoundaryByHistogram(src)

    dst = src.copy();

    for y in range(0, height):
        for x in range(0, width):
            if src.item(y, x, 0) < maxValueOfEachBoundary[0]:
                dst[y, x] = 0

            elif src.item(y, x, 0) < maxValueOfEachBoundary[1]:
                dst[y, x] = 1

            elif src.item(y, x,0) < maxValueOfEachBoundary[2]:
                dst[y, x] = 2

            elif src.item(y, x, 0) < maxValueOfEachBoundary[3]:
                dst[y, x] = 3

            elif src.item(y, x, 0) < maxValueOfEachBoundary[4]:
                dst[y, x] = 4

            else :
                dst[y, x] = 5

    return dst
```

This function assigns every pixel with a certain level.
Each level's intensity boundary is determined by the cumulative number of pixels.

```python
def softenImage(src):
    height, width, color = src.shape
    dst = src.copy()

    mask55 = np.ones((5,5), np.float64) / 25
    cv2.filter2D(src, -1, mask55, dst)

    return dst
```

soften unsoften

soften unsoften

The last step for image conversion will be smoothing the image.
It returns a soften input image.

```
palette1 = [
    (0, 0, 0), #검은색
    (164, 46, 105), #보라색
    (13, 100, 210), #귤색
    (50, 170, 240), #덜연한귤색
    (164,  227, 255),#연한귤색
    (242, 250, 253) #흰색
]

palette2 = [
    (0, 0, 0), #블랙
    (40, 41, 38), #검은색
    (125, 99, 22), #심해색
    (44, 189, 250), #하늘색
    (251, 194, 156), #살구색
    (255,  248, 178) #연노랑
]

palette3 = [
    (148, 58, 108), #보라
    (58, 19, 189), #목젖색
    (233,  151, 35), #뽕따꼭따리
    (60, 200, 255), #레몬껍질색
    (141, 243, 197), #메로나
    (208, 246, 250) #귤피색
    #여기에 팔레트 색상 추가
]
```

sample palette, we can make user's own palette.

⑥ Team’s thoughts on this project

The lecture of our class has effectively influenced us to choose an interesting topic and to strengthen our imagination.
Our team efficiently splitted the tasks to each individual resulting in great performance. Since the tasks were nicely divided, merging the tasks was not a problem either. Every team member did their best and they made good use of their personal strengths and acted as a security point for each other.

The concept of our project solution was simple but its result was fairly significant.
We made a very good converter with small effort.

We faced many obstacles while doing our project like finding perfect edges or sharpening them to divide several segments. To solve the problems, we underwent many tests repeatedly. Afterwards we struggled to find out several solutions and it was a great chance to implement and understand about image processing.

⑦ More Results