CENG471 Introduction to Image Processing Group Project

PicArt

Çağrı Mert Kemerkaya
Computer Engineering Department
Gazi University
141180044
kemerkaya.mert@gmail.com

Hilal Ilgaz
Computer Engineering Department
Gazi University
161180038
hilalilgaz06@gmail.com

Muhterem Oğuzhan Yıldırım
Computer Engineering Department
Gazi University
171180755
muhteremoguzhanyildirim@gmail.com

ABSTRACT
Our project, using some artistic effects and filters, automatic image enhancement, basic operations such as cropping, mirroring can make some photo manipulations. First step is making basic operations. These operations are flipping, rotating, cropping, mirroring. Second step is making automatic image enhancement such as histogram normalization, contrast enhancement. Third step is making more than 20 artistic filters and effects. This paper presents these 3 steps and explains them.

Keywords
Filter, Effect, OpenCV, Photo Manipulation, Pillow.

1. INTRODUCTION
Some applications in photographs such as filters, effects, minor corrections, reflections are an indispensable part of our age. Instagram is a photo editing and sharing application. In 2020, Instagram shared its statistics. One billion people use Instagram. 510 million users are female, and the others are male. This means that every 7.8 person in the World uses Instagram. More than 50 billion photos have been uploaded to Instagram [1]. This numbers are enormous and show that image editing and sharing is a huge marketing place. Lots of people think Instagram's photo editing tool does not meet the need. This problem led to a different solution. Many people prefer to use different photoshop applications such as Adobe Photoshop, VSCO for manipulating their photos. Instagram's photo editing tool just uses for making artistic filters. According to this information, it can be easily understood that there is always a place for a new photo manipulation application in the market. Our project's goal is making a simple photo manipulation application.

Our project, using some artistic effects and filters, automatic image enhancement, basic operations such as cropping, mirroring can make some photo manipulations. The project written in Python programming language and Jupyter Notebook. First step is making basic operations. These operations are flipping, rotating, cropping, mirroring. Second step is making automatic image enhancement such as histogram normalization, contrast enhancement. Third step is making more than 20 artistic filters and effects.

2. THE APPROACH
The approach which used in the project will be examined in 3 stages. These stages are stage 1- basic operations, stage 2- automatic image enhancement, stage 3- artistic effects and filters.

2.1 Stage 1
Stage 1 is about basic operations on the images. Basic operations are common side of the all of photo manipulation tools. Because these are basics of image editing. Basic operations are rotation, flipping, mirroring, cropping and inverting.

The rotate() method is written for rotation is written to rotate the photo clockwise or to rotate in counterclockwise. In addition, there is no limit for angle. Angle and rotation way depend on user wants. We did not want to strict the user, because image editing also depends on the user's imagination. User should feel free.

The flip() method is written for flipping. Flipping is an attribute that allows user to turn an image vertical or horizontal way. In our method user gives the direction which is the way for flipping. The ways are horizontal and vertical. For flipping we use transpose() method from Pillow library. Transpose() method allows to flip an image with a specific keywords. If user does not want to flip vertical or horizontal way, user can flip in both way but firstly, vertical flip after that horizontal flip.

The mirror() method is written for mirroring. Mirroring is a mirrored duplication of an object that seems nearly identical, however is reversed within the direction vertical to the mirror surface. In other words, mirroring is a reflection of an image from surface of a mirror or water [6]. For that method, we used ImageOps Module from Pillow. This module is used to make various manipulations on L and RGB images.

The crop() method is written for cropping the image. Cropping is an operation for eliminating outer or unwanted areas. The areas only depend on the user. If user wants a 1x1 cm^2 on his image, user can crop this area. There is only two limitation in our crop() method. The first limitation is bottom right x axis value must be bigger than top left x axis value. The second limitation is bottom right y axis value must be bigger than top left y axis value. The reason of these limitations is crop() method is used bottom right x axis value and bottom left axis value as a base. Also, these limitations make controls cropping coordinates for the coordinates' location in the image. Because if a user wants to crop an image, the cropping coordinates must be in the image size. After setting x axis and y axis value for the unwanted areas, we use crop() method in Pillow library for cropping. The trick is setting the axis values.

The invert() method is written for inverting. Invert() method can switch 0 to 255 and 255 to 0 in RGB values. It means this method can take negative of an image in color tones [7].

2.2 Stage 2

Stage 2 is about automatic image enhancement. Image enhancement is that the procedure of up the standard and knowledge content of original data before processing. There are two categories in image enhancement. The first one is spatial domain methods. The spatial domain methods operate directly on pixels. The second category is frequency domain methods. The frequency domain methods operate on the Fourier transform. For automatic image enhancement we wrote four methods. These methods are enhanceBrightness(), enhanceContrast(), histogramEqualization() and clahe().

enhanceBrightness() method is used for setting brightness. Brightness values are between 0 to 5. For our GUI, we adapted these values to be between 0 and 100. Our default brightness value is 1.0. If you slide left, the image is going to be darker. If you slide right, the image is going to be lighter.

enhanceContrast() method is used for setting contrast. Contrast values are between -2 to 4. For our GUI, we adapted these values to be between 0 and 100. Our default contrast value is 20.0. I f you slide left, the image is going to be black. If you slide right, the image is going to be white. For enhanceBrightness() and enhanceContrast() method, we use enhance() method in Pillow library. This function returns enhanced image.

histogramEqualization() method is used for improving contrast in the image. Histogram equalization increases the global contrast of an image when the image's usable data is showed by close contrast values. So, it allows lower contrast to gain higher contrast [8]. Our method includes equalizeHist() method in OpenCV library. This method equalizes the histogram of a grayscale image and increases the contrast and normalizes the brightness of image [9].

clahe() method is used for Adaptive histogram equalization. CLAHE is a specific type of adaptive histogram equalization. It means contrast limited adaptive histogram equalization. CLAHE focuses small parts of an image. These small parts are called tiles. The tiles are combined using interpolation. So on CLAHE's goal is improving the contrast of images [10]. We used createCLAHE() method in OpenCV for CLAHE. This method creates pointer to CLAHE class [9].

2.3 Stage 3

Stage 3 is about artistic filters and effects. Artistic effects and filters are that can change the appearance of a photo or an image by trying to imitate art such as

oil painting or pencil. Also, effects are used for adding texture and creating blurring or changing image's color tones. Artistic filters and effects used in our project are

- _1977
- Aden
- Brannan
- Brooklyn
- Clarendon
- Earlybird
- Gingham
- Inkwell
- Kelvin
- Lark
- Casper
- PencilSketch
- Emboss
- SpongeBob
- Outline
- Sharpening
- BeginnerLuck
- Shmoo
- CorpseBride
- CartoonSketch
- Toaster
- Vintage
- Carbile
- Amaro

Making and designing effects and filters has different steps. Some of them written by kernels and making those kernels as filters. Some of them combined using different OpenCV methods and some of them are processed by converting from RGB to HSV.

Vintage, CorpseBride, Shmoo, BeginnerLuck, Outline, Sharpening and Emboss are written using kernels. Vintage filter focuses center of the image. For doing that image's columns and rows assigned with a blur value to a Gaussian kernel. After that, with these kernels' transpose value multiplied old image RGB values. These new values are vintage filter's RGB values. Vintage filter's RGB values assigned to new image and the method returned this new image. CorpseBride, Shmoo, BeginnerLuck, Outline, Sharpening and Emboss filters logic is quite different from Vintage filter. Outline, Sharpening and Emboss directly applied kernels.

The sharpen kernel emphasizes differences in adjacent pixel values. The applied kernel makes the image's look more realistic [2].

0	-1	0
-1	5	-1
0	-1	0

Figure 1. The sharpen kernel

The emboss kernel gives the illusion of depth by emphasizing the differences of pixels in each direction. According to our kernel, the direction is from the top left to the bottom right [2].

-2	-1	0
-1	1	1
0	1	2

Figure 2. The emboss kernel

The outline kernel is used for highlighting large differences between pixel values. The pixel next to neighbor pixels with close to the same intensity appears black, otherwise pixels appear white [2].

-1	-1	-1
-1	8	-1
-1	-1	-1

Figure 3. The outline kernel

The sample code is the main flow of the Shmoo, BeginnerLuck, Outline, CorpseBride Sharpening and Emboss filters.

```
def emboss(img):
    img=convert_from_image_to_cv2(img)
    emboss_kernel=np.array(([[-2,-1,0],[-1,1,1],[0,1,2]]),np.float32)
    embossfilter=cv2.filter2D(src=img,kernel=emboss_kernel,ddepth=-1)
    img=convert_from_cv2_to_image(embossfilter)
    return img
```

Figure 4. The sample code

Firstly, we converted PIL image to OpenCV format. Secondly, we transformed kernel to array and this array used for making 2D filter in OpenCV. After that our OpenCV format image converted to PIL image format. CorpseBride, Shmoo and BeginnerLuck is the combine of these filters. Shmoo takes its name from

the smoothest cartoon character. Shmoo is the combine of emboss and oil painting. BeginnerLuck is the first filter which is written by us and the result is not bad. BeginnerLuck is the combine of random kernel and sharpen. CorpseBride is the combine of sharpen and gaussian blur kernel.

Carbile, SpongeBob and Casper filters are combine of two ready-to-use methods. Carbile is combine of bilateralFilter and stylization. bilateralFilter() is used for removing noise and keeping edges sharp. Stylization() is used for producing digital imagery with a wide variety effects not focused on realism [3].

Casper takes its name from the texture. Csper is combining of edgePreservingFilter and stylization methods. edgePreservingFilter() has two aim. First aim is smoothing the image. Second aim is unsmoothing the edges. The edges are color boundaries. It looks like bilateral, but it takes different sigma values. Stylization() method used like Carbile with different parameters. . SpongeBob takes its name from shape of SpongeBob not from texture. SpongeBob edge is combine of detailEnhance and stylization methods.detailEnhance() method enhances the details and makes the image's appearance sharper. Stylization() method used exactly same as Casper filter.

In these ready-to-use methods has two important parameters. These are sigma_s and sigma_r. sigma_s is using for controlling the size of the neighborhood. Sigma_r is using for controlling the how dissimilar colors in the neighborhood is averaged [4]. These parameters changes stylizations and smoothing.

PencilSketch and CartoonSketch is very different from the others. Basically, they are a series of filters and image conversions. CartoonSketch first step is downscaling image and applying bilateral filter. After applying bilateral image is upscaling. CartoonSketch is aim get a nonrealistic result, so blurred image meets the need. Using some filter edges can recognizes. After that we used adaptive threshold. The reason behind using adaptive threshold is not known the image will be the user wants to edit. Adaptive threshold works good in different lightning conditions. PencilSketch is also about edges with a similar difference. Difference is now every bit is very important for sketch. So, we converted image as a bit image after that the bit image is blurred. Using divide() method we split the image between grayscale and smooth image. The result is the pencil sketch of the image.

Amaro, Gingham, Kelvin and Toaster filters are basic converting process. It means the logic behind them is converting RGB color to HSV color. HSV means hue, saturation and value. HSV is the closest to how humans sense color. The HSV describe colors -it means hue- in terms of the colors' shade -saturation- and the colors' brightness value [5]. In our project Amaro, Gingham, Toaster and Kelvin is processed like HSV. There are 4 method for the filters HSV settings. These methods are one for brightness settings, one for saturation settings, one for splitting RGB and one for hue settings. These methods are very basic, and they use ready-to-use methods in OpenCV and Numpy. The ready-to-use methods are very common such as cvtColor, array, inRange etc. The real trick is how to set hue, saturation and value. Because each filter's appearance is different. The toaster filter is like autumn weather color like brown, yellow, red etc. So, our focus is making yellowish, redish layer for image. The parameters assigned like that. Kelvin filter looks like Toaster with a difference. The difference is Kelvin's focus point is background. So, we just set brightness, hue, saturation and converted them to RGB. Gingham is focus on sharpening colors. Hue, saturation and brightness settings meet the need. Amaro is sharpening Gingham. It means sometimes the edges can be less visible for a darker image such as the image of a place where the screen and keyboard of a black laptop intersects. If you apply Gingham filter colors can be sharp but if you apply Amaro filter color and edges can be sharp.

Aden,_1977, Brannan, Brooklyn, Clarendon, Earlybird, Inkwell and Lark filters main flow is converting image to RGB and making series of methods. These methods are written by our group. Sepia() method takes 2 inputs. RGB image and a quantity. According to quantity sepia applies filter inside the function until quantity equals to zero. The sepia result is taken by my_contr() method. This function is used for contrast settings. The contrast result is taken by my_bright() method. My_bright() method is used for brightness settings. The brightness result is taken by my_gray method. This method applies grayscale according to hand-written matrix and returns final image which is Inkwell filter image. _1977 filter's aim is to create an old image like a movie from the 70's. The appearance is softer and in pinkish color tones. Our _1977 method fills the image pinkish color tones and sets contrast, brightness and

shades like an old movie. Aden and Lark filters are dark filters. So, for their dark settings we used extra dark method. This method is making images darker before setting brightness and contrast. The approach is _1977 filter's darker version. Brannan is lighter version of Inkwell with different sepia and contrast quantities. The approach is making lighter before setting sepia and contrast. The other filters' concept is splitting layers. We split layer the image and arrange color tones with necessary parameters.

3. RESULT

In this section, we will show you the working performance of our application by showing you the results of some of the operations we have applied.

Figure 5. The original image

Figure 6. Result of Casper effect

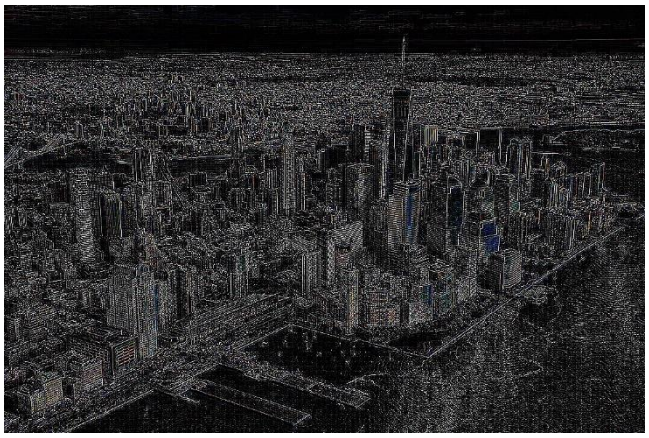

Figure 7. Result of BeginnerLuck effect

Figure 8. Result of Kelvin filter

Figure 9. Result of Amaro filter

Figure 10. Result of pencil sketch filter

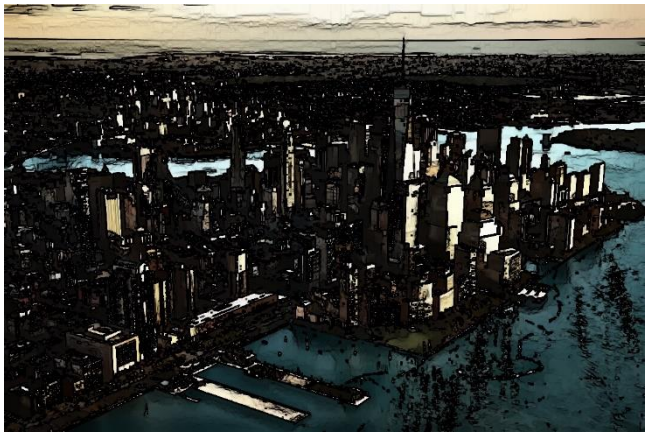

Figure 11. Result of SpongeBob effect

Figure 12. Result of Emboss effect

4. CONCLUSION

The image or photo manipulation tool can be very effective for designing and recreating images. Our image editing tool in this project works with a few lines code in various methods. The various methods are written for making basic operations on image, setting image enhancement and creating artistic effects and filters.

Basic operations on the image are rotation, cropping, flipping, mirroring and inverting. These operations work well under any circumstances. It means they do not depend on image quality, size or color tones. They depend on user's imagination.

Automatic image enhancement includes four methods. These methods can set brightness, contrast and histogram. In project's GUI is designed for controlling image enhancement by a slide bar. For automatic image enhancement, as we allow the user to make the desired setting, we also set the values we think are the best as default.

Artistic filters and effects are written method by method. Our methods use for smoothing, sharpening, blurring, setting color channels. Our goal is to make the images more beautiful or different by using artistic effects and filters. Artistic effects and filters methods give different quality for different images. For example, if you want to cartoonize a cartoon character, you can not take best result. Because this method's goal is making cartoon to an image. If you give a human selfie for applying cartoon sketch effect, you can take very good results.

ACKNOWLEDGEMENTS

This project is made for CENG471 Introduction to Image Processing final project. We thank the lecturer Dr. Duygu Sarıkaya who is the lecturer of CENG472 Introduction to Image Processing for the information she gave us about image processing and what we learned for our project.

5. REFERENCES

[1] Clement J. 2021. *Instagram by the Numbers.* Web: https://www.omnicoreagency.com/instagram-statistics/

[2] Powell V. (nd) *Image Kernels.* Web: https://setosa.io/ev/image-kernels/

[3] OpenCV Document. (nd) *Smoothing Images.* Web: https://docs.opencv.org/master/d4/d13/tutorial_py_filtering.html

[4] Mallick, S. 2015. *Non-Photorealistic Rendering using OpenCV*. Web: https://learnopencv.com/non-photorealistic-rendering-using-opencv-python-c/.

[5] Kormos J. 2019. *The HSV Color Model in Graphic Design.* Web: https://www.lifewire.com/what-is-hsv-in-design-1078068

[6] Carson A. 2009. *Mirror Reflections.* Web: https://web.archive.org/web/20100206121909/http://www.d7s.com/mirror.htm

[7] Anonim. (nd) *Image-Invert.* Web: http://www.georeference.org/doc/image_invert.htm

[8] Sudhakar S. 2017. *Histogram Equalization.* Web: https://towardsdatascience.com/histogram-equalization-5d1013626e64

[9] OpenCV document (nd) *Histograms.* Web: https://docs.opencv.org/3.4/d6/dc7/group__imgproc__hist.html#ga7e54091f0c937d49bf84152a16f76d6e

[10] Senaratne R. 2020. *CLAHE and Thresholding in Python.* Web: https://towardsdatascience.com/clahe-and-thresholding-in-python-3bf690303e40