Image Editor Using Tkinter (Assignment 1)

Durgesh Ahire (18D070004)
Electrical Engineering
Department,

IIT Bombay

***Abstract*—This document serves as a report for the overall work done as part of EE610: Image Processing assignment 1. The topic was to create a basic image editor with a few added filters and image enhancement techniques.**

Keywords*— *Image Editor, Tkinter, Image Enhancement, filters(*key words*)

I. Overall Summary

The primary library used for creating the GUI is Tkinter due to its overall ease of implementation. The editor created can handle both grey scale as well as RGB images. It has an image display area where the image loaded from the current directory is shown. The enhancement techniques available in the editor include, histogram equalization, gamma transform, log transform, blurring, sharpening, color reduction. Undo, Undo all and save are a few features present in the editor. For, the RGB images first the image is converted into HSV format, and then the value of V channel is used to perform transformations. The image processing libraries used for the editor are Tkinter, opencv, PIL, numpy. All the functions used to implement the specific filters are self coded without use of any inbuilt function. A special function is used to reduce the color intensity of the image named color reduction.

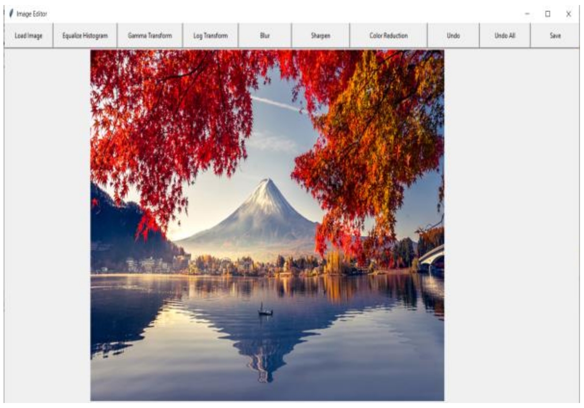


Fig.1. Image Editor Main Window View

II. Introduction

The image is first loaded into the image editor by using the open button. All the images used (opened) are rescaled to a size of 800x600 pixels in order to fit the window. The main parts or the features of the image editor are explained below:

A. *Functions Body and its applications:*

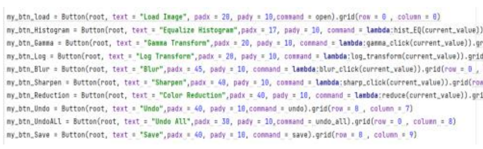
The first function built was the load function. It has a variety of global variables declaration in its body. The filedialog function is used to restrict the allowed types of files that could be opened. We have allowed only two types of image files to be edited using our editor which are ‘jpg’ and ‘png’. Next the open function also converts the loaded image into an BGR image for performing operations and saving the initial values into the undo and undo all variables. It is then displayed onto the main window using the PIL library functions PhotoImage function.

Then, the log and gamma transformation functions are used upon the BGR value converted from the initial load operation. For log transformation, a normalization constant is used to round off the intensity values to less than or equal to 255. Similarly, the input gamma value is also taken from user for the gamma transformation which is to be applied to the initial image. Repeated application of the same function on the image leads to operations on the updated images of each iteration. The undo and undo all functions when applied lead to the last updated image and original image respectively. It performs this function by accessing the values stored in the variable ‘u’ and ‘ul’ respectively.

The blur and sharpening functions defined use the convolve2d function to calculate the convolution of the images with the given kernels. For blurring, average pooling is used to convolve with original image and also the padding used is adjusted depending upon the dimension of the kernel size. For sharpening, the specified kernel is used of size 3x3 with padding =1. The image is saved with the help of cv2.write function from the opencv library used in the current working directory. Equalize histogram function uses the cumsum() function in order to calculate the cumulative sum and perform the equal distribution of intensity across the pixels.

B. *The GUI Design:*

The GUI design revolves around the use of Tkinter library and the various functionalities is provides in creating the



```
my_btn_load = Button(root, text = "Load Image", padx = 20, pady = 10,command = open).grid(row = 0 , column = 0)
my_btn_Histogram = Button(root, text = "Equalize Histogram",padx = 17, pady = 10, command = lambda:hist_EQ(current_value)).grid(row = 0 , column = 1)
my_btn_Gamma = Button(root, text = "Gamma Transform",padx = 20, pady = 10, command = lambda:gamma_click(current_value)).grid(row = 0 , column = 2)
my_btn_Log = Button(root, text = "Log Transform",padx = 20, pady = 10, command = lambda:log_transform(current_value)).grid(row = 0 , column = 3)
my_btn_Blur = Button(root, text = "Blur",padx = 45, pady = 10, command = lambda:blur_click(current_value)).grid(row = 0 , column = 4)
my_btn_Sharpen = Button(root, text = "Sharpen",padx = 40, pady = 10, command = lambda:sharp_click(current_value)).grid(row = 0 , column = 5)
my_btn_Reduction = Button(root, text = "Color Reduction",padx = 40, pady = 10, command = lambda:reduce(current_value)).grid(row = 0 , column = 6)
my_btn_Undo = Button(root, text = "Undo",padx = 40, pady = 10,command = undo).grid(row = 0 , column = 7)
my_btn_UndoALL = Button(root, text = "Undo All",padx = 30, pady = 10,command = undo_all).grid(row = 0 , column = 8)
my_btn_Save = Button(root, text = "Save",padx = 40, pady = 10, command = save).grid(row = 0 , column = 9)
```

Fig.2. Buttons Defined for various functions

GUI. The main function is creation of widgets and buttons by using the library and packing them onto the window of the GUI. Then the integration of functions along with the use of buttons follows where the decision regarding which function to be called upon is decided on the basis of the button clicked. Using grid the buttons are arranged in the form of a matrix elements where each button can be positioned using the row and column reference like in the matrix.

III. Results and Experiments:

A) Gamma Transformation:

For gamma transformations a darker image was considered

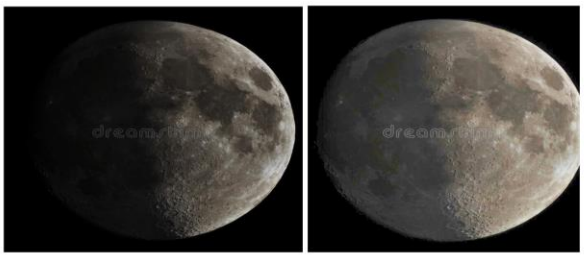


Fig.3. Original v/s Gamma Transformed (y = 2)

since the lower intensity pixels that contained information could be mapped to a higher intensity in order to get the desired information. Hence, the choice of image here was a darker image.

B) Histogram Equalization:

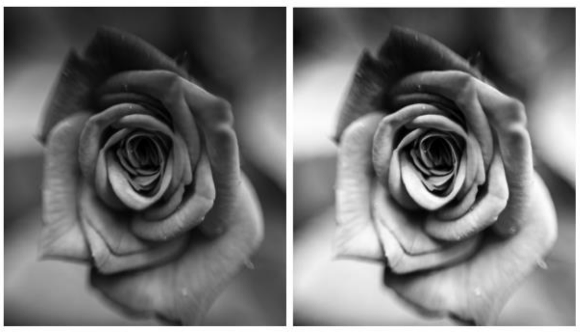


Fig.3. Original v/s Histogram Equalized

The above image shows the histogram equalized image of the original left hand side rose image. We can see from the above figure that the intensity values in the right hand side image is equally distributed.

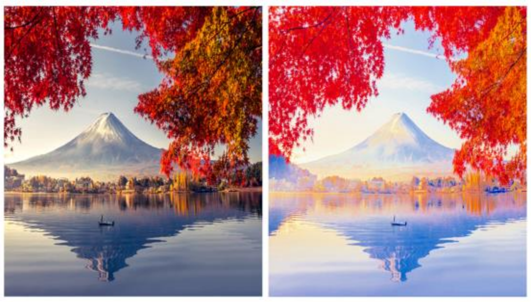


Fig.4. Original v/s Log Transformed Image

C) Log Transform:

The image for log transformation is chosen to be a bit darker since post application of log transform the image becomes more brighter. The darker features which are not visible in the original image also become brighter and hence some unique characteristics of image can be obtained.

D) Blurred Image:

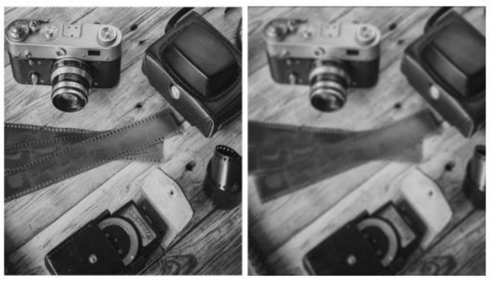


Fig.5. Original v/s Blurred Image

Average pooling filter is used to blurr the image. The extent of blurring is dependent upon the size of the filter. As seen from the above image, the right hand sided image is somewhat smoother as compared to the original left hand side image.

E) Sharpened Image:

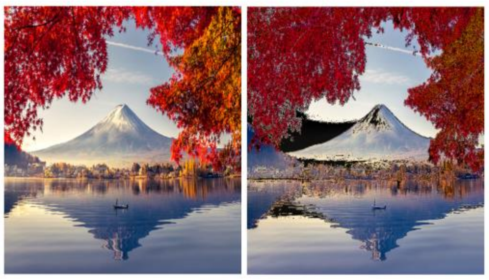


Fig.5. Original v/s Blurred Image

The above image shows the sharpened version of the original image. The some pixels get mapped to 0 intensity due to sharpening and hence the black portion is visible in the image.

F) Color Reduction or Discretization(Unique Enhancement):

The pixel intensities of the image are discretized by performing the floor operation on the intensity values by dividing by 32 and the resulting color discretized version of the image is obtained. The image shown below is the example of discretized version.

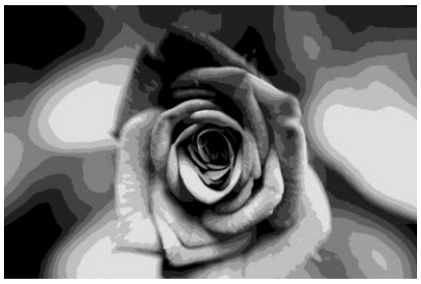


Fig.6. Intensity Value Discretized Image

Now, multiple operations like gamma transform, blurring, sharpening are operated on the image and resulting image obtained is as shown below:

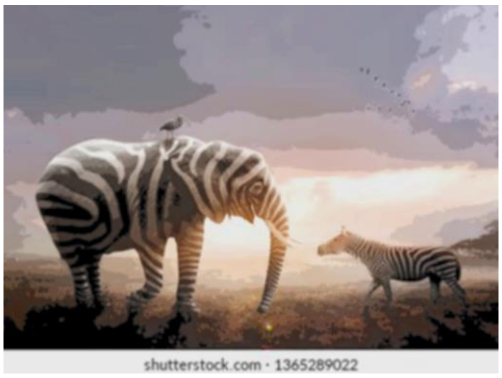


Fig.7. Multiple Operations Processed Image

IV. Challenges Faced and Improvements:

The main challenges include the development of the GUI using tkinter. Understanding the basic functionalities of the library, positioning, displaying images take a lot of time to get implemented. Apart from that the function for convolutional2d used for the kernels in case of blurring and sharpening of image using Laplacian operator is also quite cumbersome. Padding needs to be taken into account and then finally extracting the original image from the new padded image also took some time. I also faced the problem of int overflow while implementing log transform which I then rectified by conversion in 64 bit int and then applying the transform.

The improvements can be made into the sharpening function. It can be optimized by the use unsharp masking and high boost filtering. The layout of the editor can be made more attractive by using some specific color combinations and designs.

V. References

1. https://datacarpentry.org/image-processing/06-blurring/#:~:text=Blurring%20is%20an%20example%20of,tasks%20such%20as%20edge%20detection. – Blurring using Python for Image Processing
2. https://towardsdatascience.com/image-processing-with-python-blurring-and-sharpening-for-beginners-3bcebec0583a - Sharpening using Python
3. https://medium.com/analytics-vidhya/2d-convolution-using-python-numpy-43442ff5f381 - 2D convolution adapted from the above code
4. https://www.youtube.com/watch?v=YXPyB4XeYLA - Tutorials on Tkinter in python
5. https://pillow.readthedocs.io/en/stable/ - For display of image in tkinter
6. https://medium.com/@kyawsawhtoon/a-tutorial-to-histogram-equalization-497600f270e2 - Used for Histogram..Equalization