Implementation of Basic Image Editor-Image Processing (EE 610)

Mohit Kumar Meena
Department of Electrical Engineering
Indian Institute of Technology, Bombay
Powai, Mumbai, Maharashtra, 400076
Email: 213070021@iitb.ac.in

***Abstract*—In this project an Basic Image editor is implemented that can perform basic Image Processing operations like Histogram Equalization, Power-law transformations, Logarithmic Transformations, Sharpening, Smoothing and Negative of an image. An GUI (Graphical User Interface) is also implemented to perform all these operations using Tkinter (Python) framework. This report will provide an basic outline on how these operations are performed with the snippet of the code and the mathematics involved behind these operations.**
Index Terms

***Index Terms*—GUI, Python, Operations, Image, Basic.**

I. Introduction

This section emphasize on the brief understanding of some of the image processing operations which are used for image enhancement in various fields like medical imaging, Satellite imaging, Remote sensing, Microscopic imaging, Pattern recognition,etc. Image processing is not just limited to adjust the spatial resolution of the everyday images captured by the image, It is not just limited to increase the brightness of the photo, etc. Rather it is far more than that[1]. **Histogram Equalization** is an image processing technique that adjusts the contrast of an image by using its histogram. To enhance the image's contrast, it spreads out the most frequent pixel intensity values or stretches out the intensity range of the image. By accomplishing this, histogram equalization allows the image's areas with lower contrast to gain a higher contrast [2]. A variety of devices for image capture, printing, and display respond according to a power law. The exponent in power law equation is referred to as gamma Þ process used to correct this power law response phenomena is called **gamma correction**. [3] Gamma correction is extremely important as use of digital images for commercial purposes over the internet has increased. **Log transformation** of an image means replacing all pixel values, present in the image, with its logarithmic values. Log transformation is used for image enhancement as it expands dark pixels of the image as compared to higher pixel values. **Image negatives** is the reversing the intensity levels of an digital image in this manner produces the equivalent of a photographic negative. **Smoothing** of an image is done to reduce sharp transitions in the intensity since the noise generally contains sharp transitions so ultimately it results in noise reduction. **Sharpening** of an image is done to highlight transitions in intensity. Use of image sharpening range from electronic printing and medical imaging to industrial inspection and automomous guidance on military systems[4]. All these operations are discussed thoroughly in forthcoming sections.

II. GUI Design

GUI design of this project is implemented using Tkinter framework in python.Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task [5]. There are a number of widgets which you can put in your tkinter application. Some of the major widgets are listed below

- **Frame**- It acts as an container to hold the widgets.
- **Button**-Perform specified operation on click.
- **RadioButton**- Select the operations.
- **Canvas**- Display input and output.
- **Label**- Write text on the interface.
- **Entry**- Takes input from the user.
- **Text**-To edit a multi-line text and format the way it has to be displayed.

The Basic GUI approach involves three main steps

- Select the File
- Select the Operation
- Display the Output

In the first step an image is taken as an input from the user, the image can be any one of .png, .jpg, .jpeg and .tif format. the input selected by the user is stored in the global variable that is used for further operations.

```
global my_image
def open():
    global path,image_path,type,list
    path=filedialog.askopenfilename()
    image_path=Image.open(pathh)
    image_path.thumbnail((1000,700))
```

In the second step the operation needs to be selected among the all given choices and based on the selected choice the input image is manipulated.

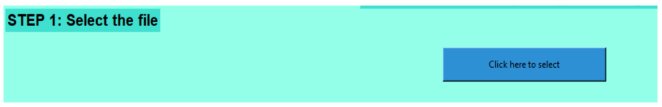

Figure 1. Snippet of GUI for selecting image

```
# Code snippet for step 2
label1=Label(root,text='STEP 2:
Select the Operation',font=
('arial',15,'bold'),bg='#40E0D0')
label1.place(x=5,y=170)
r1=Radiobutton(root, text="Equalize
Histogram",bg='#93FFE8',font=('arial',13),
variable=r, value=1,
command=lambda: clicked(r.get()))
r1.place(x=5,y=200)
```

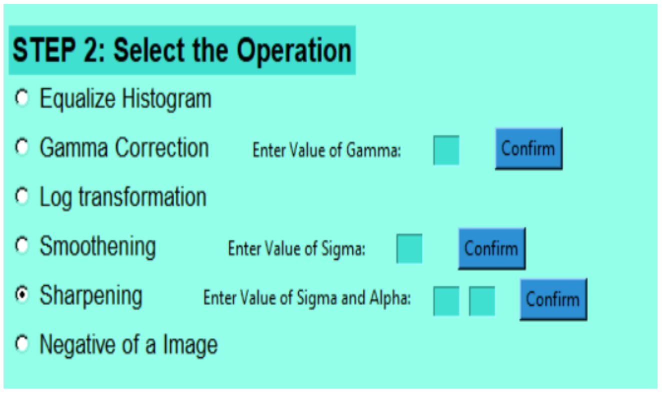

Figure 2. Snippet of GUI for selecting the operation

In the third step the output of transformed image is displayed, An canvas of size 900*300 is taken which initially displays the original image that is selected in step 1 and then displays the transformed image based on the selection in step 2.

```
# Code snippet for step 3
canvas1=Canvas(root,width="900",
height="300",relief=RIDGE,
bd=5,bg='#87dec3')
canvas1.place(x=280,y=400)
```

The overall GUI design also include some additional button that are not listed above such as Undo operation , saving an transformed image and exit the window as the name suggests the undo operation button will revert an operation that had done recently on clicking once and revert the second previous operation on clicking twice and so on. the various command to import Tkinter libraries used are

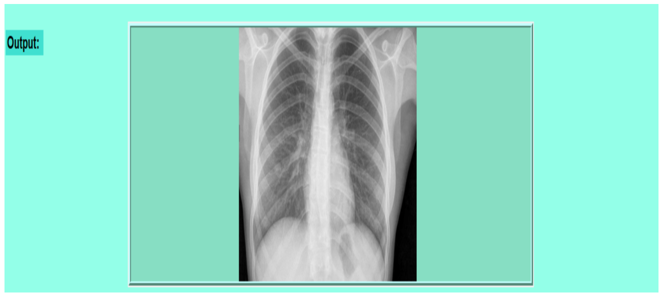

Figure 3. Snippet of GUI for Image display area

```
# Importing Tkinter libraries
from tkinter import *
from PIL import ImageTk, Image
from tkinter import filedialog
```

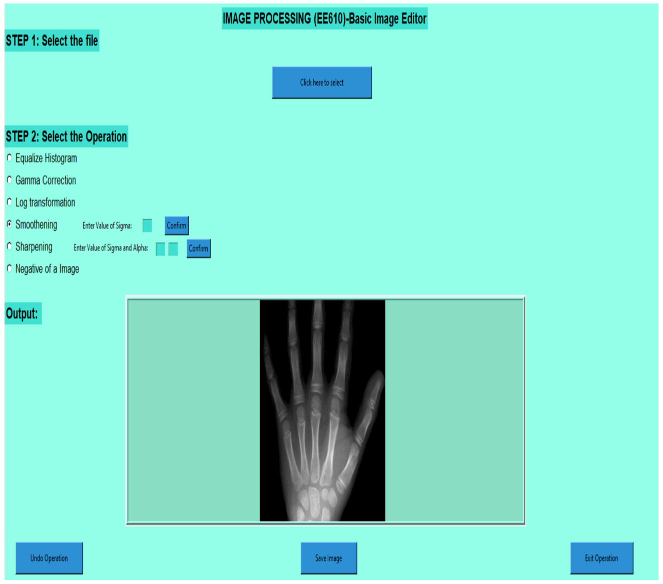

Figure 4. Overall GUI design

III. Image Processing Operations

As discussed in Section I the various Image enhancement techniques used in this project are

- Histogram equalization
- Power- law transformations
- Log transformations
- Smoothing an Image
- Sharpening an Image
- Negative of an Image

A. *Histogram Equalization*

Histogram equalization is a method to process images in order to adjust the contrast of an image by modifying the intensity distribution of the histogram. The objective of this technique is to give a linear trend to the cumulative probability function associated to the image.

The processing of histogram equalization relies on the use of the cumulative probability function (cdf)[6]. The cdf is a cumulative sum of all the probabilities lying in its domain and defined by

$$CDF(x) = \sum_{k=-\infty}^{k=\infty} P(k)$$

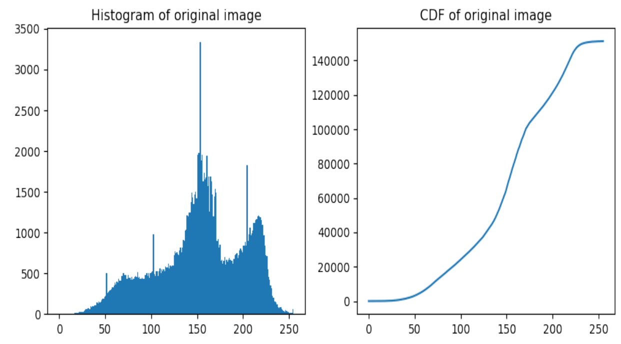

Figure 5. Histogram of original image [left] and it's CDF [right]

The idea of this processing is to give to the resulting image a linear cumulative distribution function. Indeed, a linear cdf is associated to the uniform histogram that we want the resulting image to have.

$$S_k = (L-1)CDF(x)$$

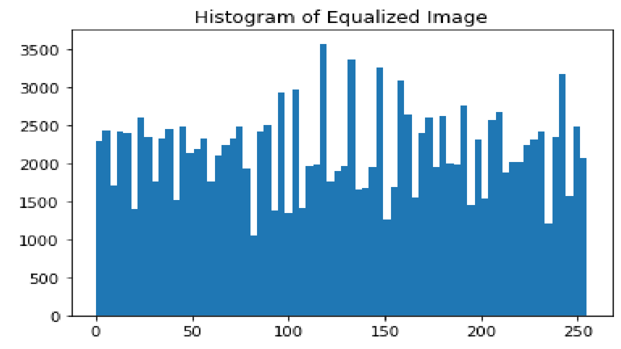

Figure 6. Equalized Histogram

Here L is the number of intensity levels. and intensity values before and after the transformations remains in $[0, L-1]$. Because a histogram is an approximation to a PDF, and no new allowed intensity levels are created in the process, As shown in figure 6, perfectly flat histograms are rare in practical applications. the net result of histogram equalization is the contrast enhancement as shown in figure 7.

The reason for choosing the image in figure 7 is to visualize the effect of histogram equalization clearly since the image contains blur background and various ranges of intensity values are present in the image.

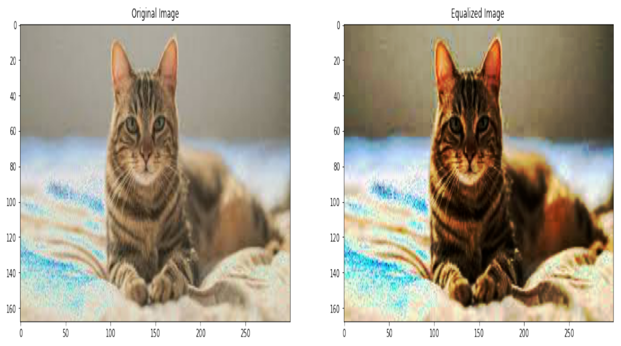

Figure 7. Original Image [left] and equalized image [right]

B. Power-law transformations

The power-law transformation also called Gamma transformation or Gamma correction is a technique used for enhancing images for different type of display devices. There are further two transformation in power law transformations, that include nth power and nth root transformation. The power-law transformations have the form

$$s = cr^{\gamma}$$

where c and γ are positive constants and transformations is performed by varying simply varying γ. Curves generated with the value of γ >1 has exactly the opposite effect as those genereated with values of γ <1 [4].

```
# Gamma transformations
c = 255 / math.pow(255,gamma)
a = c * (np.power(a,gamma))
a = np.array(np.round(a))
```

the above code is just an part of main algorithm while in the actual program some other steps needs to be carried out e.g Flattening, unflattening, etc.

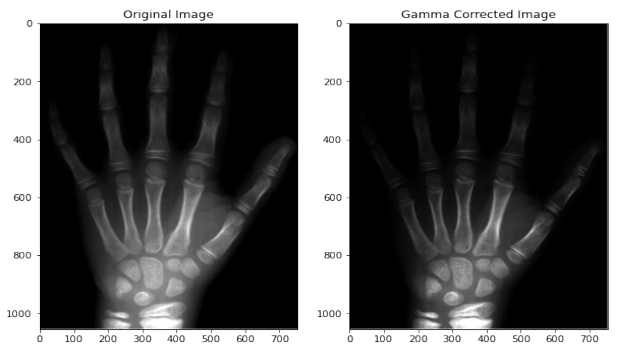

Figure 8. Original Image [left] and Gamma transformed image [right]

The result shown in figure 8 is for γ= 1.5, the image becomes darker on increasing the value of γ and becomes brighter on decreasing γ and it totally depends upon the image and it's application.

C. Log transformations

During log transformation, the dark pixels in an image are expanded as compare to the higher pixel values. The higher pixel values are kind of compressed in log transformation. The general form of the log transformations is

$$s = clog(1 + r)$$

where c is a constant and it is assumed that r $\geq$ 0. the opposite of log transformations is the inverse log (exponential) transformations[4]. Any curve that has general shape of log transformations accomplish this spreading/compressing of intensity levels in an image, but the power- law transformations discussed in section III-B are much more versatile for this purpose.

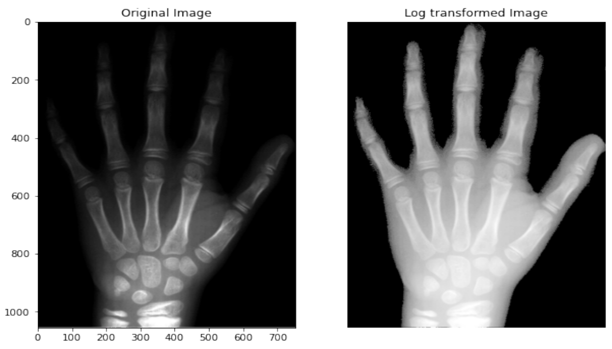

Figure 9. Original Image [left] and log transformed image [right]

The image in figure 9 is an X-Ray image of a hand of a human and is chosen because some part of the X-Ray is not clearly visible in original so after applying transformations the intended part becomes more visible.

D. Smoothing of an Image

Smoothing (also called averaging) spatial filters are used to reduce sharp transitions in intensity. Because random noise typically consist of sharp transitions in intensity, an obvious application of smoothing is noise reduction. Smoothing is used to reduce irrelevant details in an image where "irrelevant" refer to pixel regions that are small with respect to the size of the filter kernel[4]. In this project the Gaussian filtering is implemented since it has various advantage when compared to box filtering. The basic expression for gausssian filter in 2-D is

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

```
#Gaussian filter algorithm
for x in range(-m, m+1):
    for y in range(-n, n+1):
        x1 = 2*np.pi*(sigma**2)
        x2 = np.exp(-(x**2 + y**2)/
              (2* sigma**2))
        gaussian[x+m, y+n] = (1/x1)*x2
```

The Gaussian outputs a ‘weighted average’ of each pixel’s neighborhood, with the average weighted more towards the value of the central pixels. This is in contrast to the mean filter’s uniformly weighted average. Because of this, a Gaussian provides gentler smoothing and preserves edges better than a similarly sized mean filter[7].

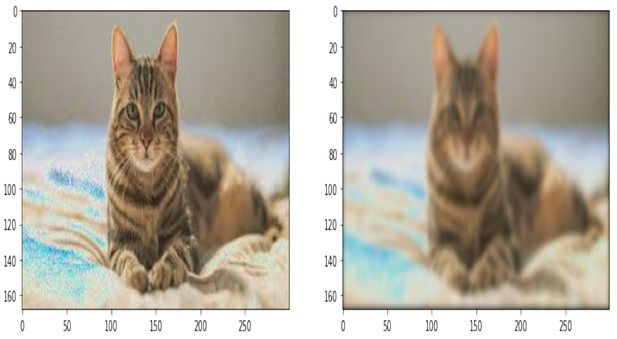

Figure 10. Original Image [left] and Gaussian filtered image [right]

The results shown in figure 10 is obtained for σ=2 and blurring increases with the increasing value of σ.

E. Sharpening of an Image

Sharpening filters also called Highpass spatial filters are used to highlight transitions in intensity. Uses of image sharpening ranges from electronic printing and medical imaging to industrial inspection and autonomous guidance in military systems. Sharpening can be accomplished by spatial differentiation and the strength of the response of a derivative operator is proportional to the magnitude of the intensity discontinuity at the point at which the operator is applied [4].A basic first order derivative in 1-D is given by

$$\frac{\partial f}{\partial x} = f(x + 1) - f(x)$$

whereas the second order derivative is the difference given below

$$\frac{\partial^2 f}{\partial x^2} = f(x + 1) + f(x - 1) - 2f(x)$$

```
#Sharpening filter algorithm
 for c in range(3):
   im_filtered[:, :, c] = convolution
   (image_path[:, :, c], firstDerivative_X
   (sigma)*firstDerivative_Y(sigma)*
   gaussianFilter(sigma))
    output=np.clip((image_path -
    (alpha * im_filtered)),0,255).astype
     (np.uint8)
output1=np.array(output
```

The key point in the effective sharpening process lies in the choice of the high-pass filtering operation. Traditionally, linear filters have been used to implement the high-pass filter, however, linear techniques can lead to unacceptable results if the original image is corrupted with noise [3]. A basic 3*3 laplacian kernel has the form

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

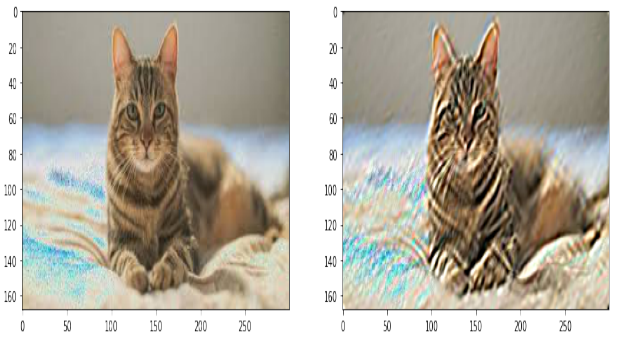

Figure 11. Original Image [left] and Sharpened image [right]

The results shown in figure 11 is obtained for the values of σ=1.5 and $\alpha = 5$.

F. Negative of an Image

mage negatives, most of you might have heard this term, in good old days were used to produce images. Film Photography has not yet become obsolete as some wedding photographers are still shooting film. Because one has to pay for the film rolls and processing fees, most people have now switched to digital. Image negative is produced by subtracting each pixel from the maximum intensity value. e.g. for an 8-bit image, the max intensity value is $2^8 - 1 = 255$, thus each pixel is subtracted from 255 to produce the output image [8]. Thus, the transformation function used in image negative is

$$s = T(r) = L - 1 - r$$

Where L-1 is the max intensity value and s, and r are the output and input pixel values respectively. The image used in figure 12 is an X-Ray image of human skull and the results of negating the image can be clearly visualized.The black background in original image has turned white and skull which was bright in original image has turned dark in transformed image.

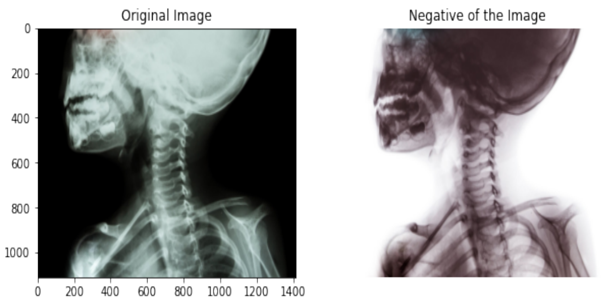

Figure 12. Original Image [left] and Negative of Image [right]

IV. Experiments and Results

The results obtained in various operations performed throughout the project is shown in Section III in their respective subsection, the result obtained matches with their intended theoretical intuition. The difference between the original image and transformed image can be clearly visualized. The various steps involved in making of an GUI is listed and illustrated clearly in Section II and the overall GUI design can be seen in figure 4. The various operations listed in the statement of Assignment 1 has been carried out and for the additional part Image negative has been implemented.

V. Conclusion and Discussion

The main challenge I faced is in the GUI design since Tkinter is totally new framework for me. And in stipulated time along with writing the code for all those operations listed in Section III implementing the GUI from scratch is really challenging initially but after looking various resources over internet and after going through several articles some understanding of the framework is developed that is somewhat sufficient for implementing the given project. Given more time definitely I would like to design more professional GUI and want to perform various other operations taught during lectures like Sobel Operation, mirror padding, median filtering, bandreject filtering, bandpass filtering,etc.

Acknowledgment

I would like to thank Prof. Amit Sethi, Dept. of Electrical Engineering, IIT Bombay for his keen efforts during teaching that helped throughout the project and also Students of Image processing (EE 610) whose discussion helped to avoid and correct several mistakes while implementing the project.

References

[1] Tutorial Point, “Image processing and it’s applications,” https://www.tutorialspoint.com/dip/applications_and_usage.htm, accessed 18-08-2021.

[2] Medium , “Histogram equalization in image processing,” "https://medium.com/@kyawsawhtoon/a-tutorial-to-histogram-equalization-497600f270e2", accessed 18-08-2021.

[3] NPTEL, “Power law transformations,” "https://nptel.ac.in/content/storage2/courses/117104069/chapter_8/8_14.html", accessed 18-08-2021.

[4] R. C. Gonzalez and R. E. Woods, *Digital Image Processing 4th edition*. Pearson, 2008.

[5] GeeksForGeeks, “Python gui,” "https://www.geeksforgeeks.org/python-gui-tkinter/", accessed 18-08-2021.

[6] U. of Utah, “Histogram equalization,” "http://www.sci.utah.edu/~acoste/uou/Image/project1/Arthur_COSTE_Project_1_report.html", accessed 16-08-2021.

[7] inf.ac.ed.uk, “Gaussian filtering,” "https://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm", accessed 19-08-2021.

[8] T. A. Learner, “Image negatives,” "https://theailearner.com/2019/01/01/image-negatives/", accessed 19-08-2021.