**Lab Assignment 3: Image and Audio Data Processing**

You are tasked with developing a comprehensive Python program that reads and manipulates both image and audio data. The goal is to create a tool that processes images and audio waveforms, allowing users to perform various operations on both types of data. This exercise aims to test your proficiency in handling different data formats and applying appropriate algorithms for manipulation.

**Part 1: Image Data Processing**

1. Image Loading and Display: Your program should allow users to load an image file and display it. Ensure you use an image processing library like Pillow (PIL) to handle image data.

2. Image Manipulation: Implement at least two image manipulation operations, such as:

• Applying filters (e.g., Gaussian blur, edge detection).

• Changing image dimensions or cropping.

• Adjusting brightness, contrast, or saturation.

• Converting to grayscale or other color spaces.

3. Histogram Analysis: Implement a feature that calculates and displays histograms for different color channels of the loaded image. Allow users to analyze and manipulate histogram data.

**Input Files:**

****

**Code:**

from PIL import Image, ImageFilter

import matplotlib.pyplot as plot

def displayImage(path):

    image = Image.open(path)

    plot.imshow(image)

    plot.axis(False)

    plot.show()

    return image

def toGrayscale(path):

    image = Image.open(path)

    grayImage = image.convert('L')

    return grayImage

def gaussianBlur(path,radius=2):

    image = Image.open(path)

    blurImage = image.filter(ImageFilter.GaussianBlur(radius))

    return blurImage

def displayManipulatedImages(path):

    blur = gaussianBlur(path,5)

    gray = toGrayscale(path)

    plot.subplot(1,2,1)

    plot.title("Gaussian Blur")

    plot.imshow(blur)

    plot.axis('off')

    plot.subplot(1, 2, 2)

    plot.title("Grayscale")

    plot.imshow(gray,cmap='gray')

    plot.axis('off')

    plot.show()

import numpy as np

def display\_histogram(image):

        img=Image.open(image)

        image\_array=np.array(img)

        color\_channels=('Red', 'Green', 'Blue')

        for i, color in enumerate(color\_channels):

            histogram, bin\_edges = np.histogram(image\_array[:, :, i].flatten(), bins=256, range=(0, 256))

            plot.plot(bin\_edges[0:-1], histogram, color=color.lower())

        plot.show()

if \_\_name\_\_=="\_\_main\_\_":

    pth = 'pic.jpg'

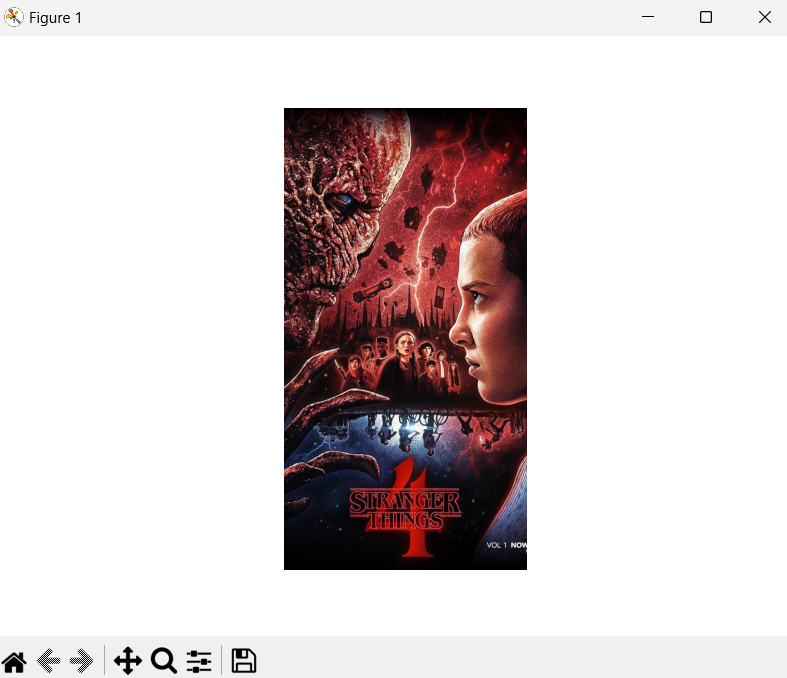
    displayImage(pth)

    displayManipulatedImages(pth)

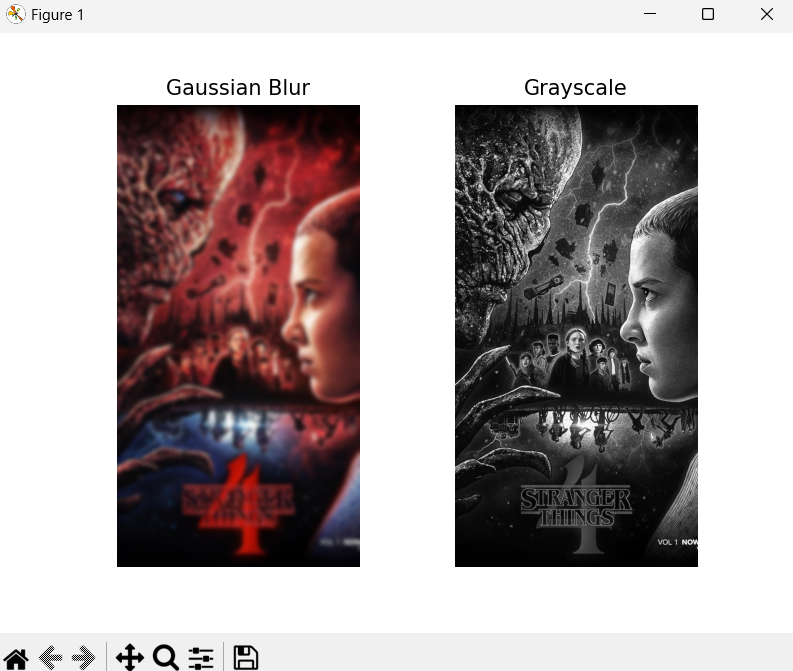
    display\_histogram('pic.jpg')

**Output:**

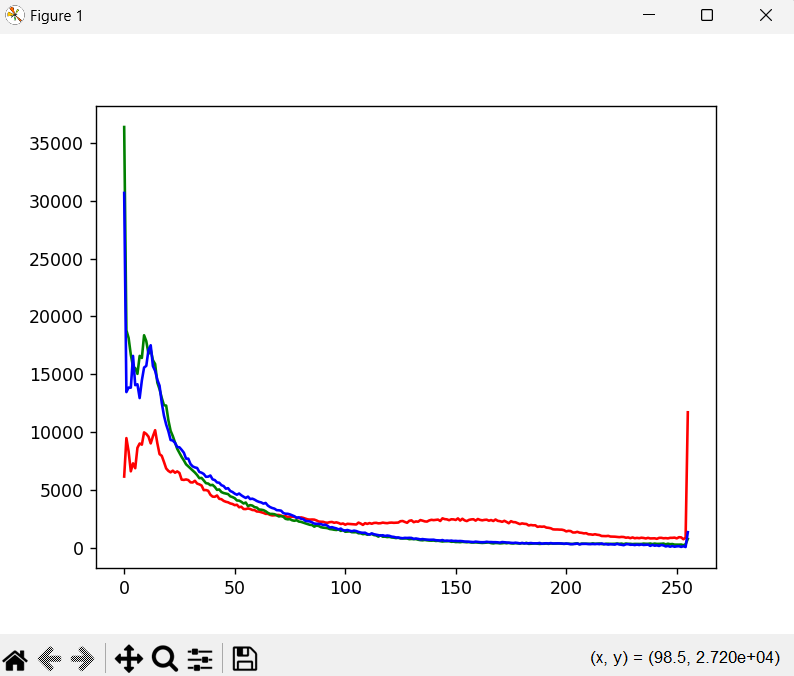
* **Image Loading and Display**

****

* **Image Manipulation**

****

* **Histogram Analysis**

****