Digital Image Processing: Assignment

Vaibhav Sharma: T23156

September 17, 2024

1 Image Analysis (Brightness, Contrast, etc.)

1.1 Brightness

The brightness of an image can be calculated as the mean value of all pixel intensities.

1.2 Contrast

Contrast can be calculated using the image histogram. Specifically, it is the standard deviation of pixel intensity values.

1.3 Hue, Saturation, and Value

We convert the RGB image into HSV and extract hue, saturation, and value components of each pixel and get average of all pixels.

Figure 1: RGB Image 1

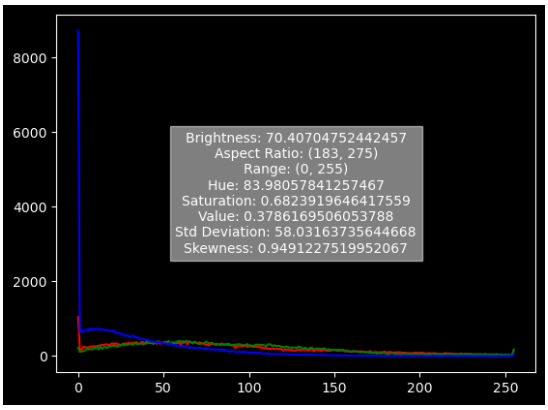


Figure 2: Properties of image 1

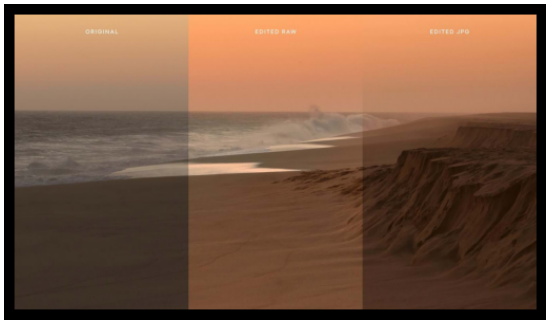


Figure 3: RGB Image 2

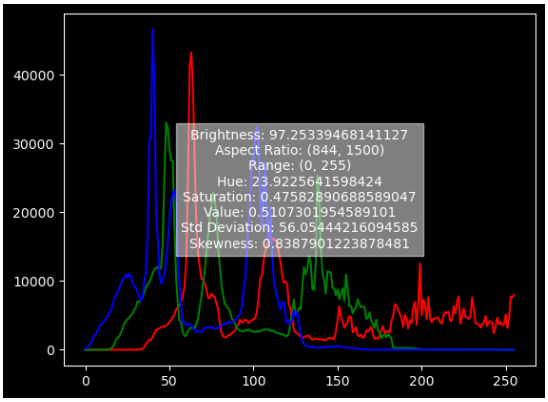


Figure 4: Properties of image 2

1.4 Conclusion

By calculating the above metrics, we gain insights into the visual and statistical properties of the image. These properties assist in tasks such as image preprocessing and filtering.

2 To apply smoothing filters, edge filters, and Laplacian filter on a given image

In this section, we apply various filters on an image to observe their effects. The filters used are:

- Average filter for smoothing
- Gaussian filter for noise reduction
- Sobel filter for edge detection
- Laplacian filter for edge enhancement

The filters are applied using convolution, and we analyze the mathematical principles behind each.

2.1 Average and Gaussian Smoothing Filters

The average filter works by averaging the pixel values in a local neighbourhood. A simple 3x3 kernel is used for this operation:

$$K_{\text{avg}} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

The Gaussian filter is used to reduce noise and blur the image. The kernel for a 3x3 Gaussian filter is:

$$K_{\text{gauss}} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

2.2 Sobel Edge Detection Filter

The Sobel filter is used to detect edges in an image by computing the gradient magnitude. It uses two 3x3 kernels to detect horizontal and vertical edges:

$$K_{\text{sobel-x}} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$K_{\text{sobel-y}} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

2.3 Laplacian Filter

The Laplacian filter detects edges by computing the second derivative of the image. The 3x3 kernel for the Laplacian filter is:

$$K_{\text{lap}} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Smoothing filters reduce noise and blur images, while edge detection filters like Sobel and Laplacian emphasize the boundaries and details. Padding options affect how the boundaries of the image are handled during convolution.

(a) Original Image

(b) Applying Average Filter

(c) Applying Gaussian Filter

(d) Applying Edge Filter

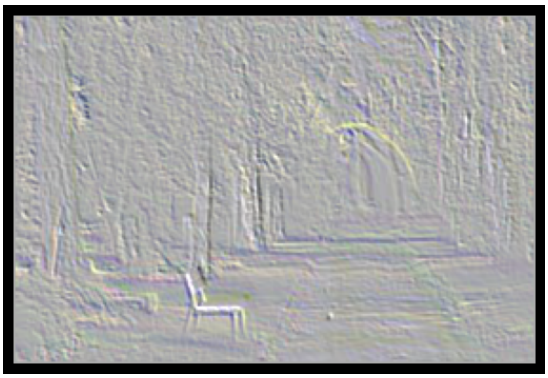
(e) Applying Laplacian Filter

Figure 5: Applying various filters to image 1

(a) Original Image

(b) Applying Average Filter

(c) Applying Gaussian Filter



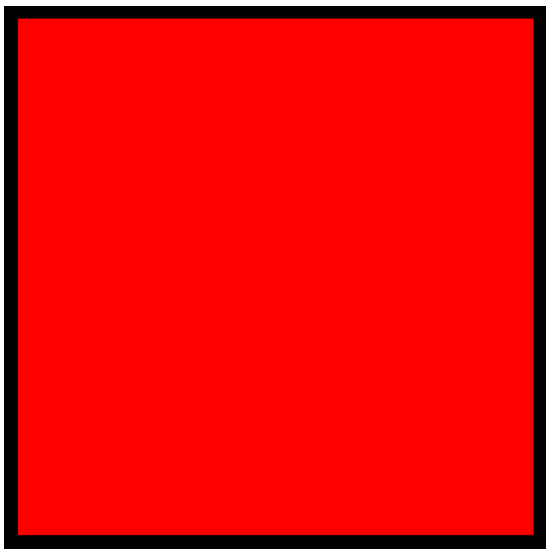
(d) Applying Edge Filter

(e) Applying Laplacian Filter

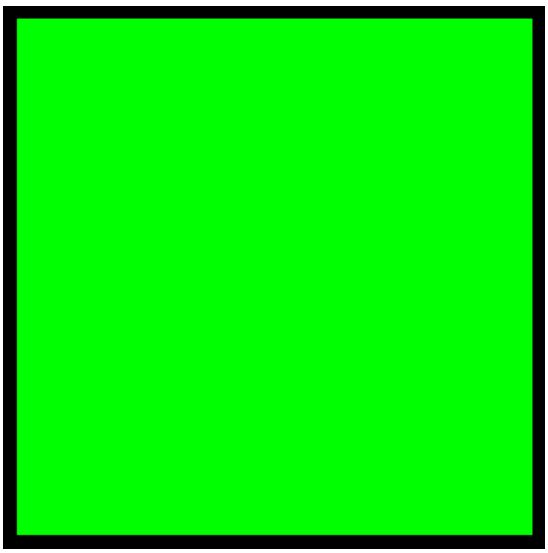
Figure 6: Applying various filters to image 2

3 To create three colour images and combine them using weighted tuples

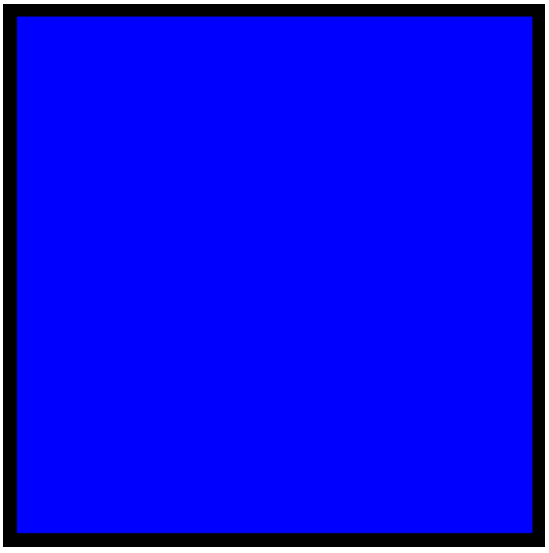
This section focuses on generating three images with primary colours (red, green, and blue) and combining them using a weighted combination. The weights determine the contribution of each colour to the final image. Each image is created by setting one colour channel to the maximum value (255) while the others are set to zero.



(a) Pure Red color

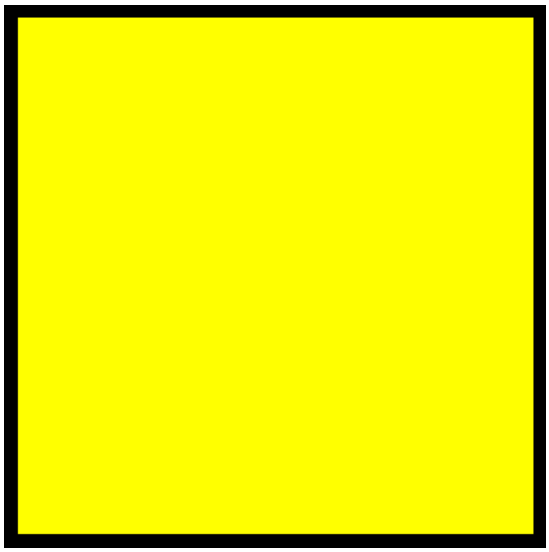


(b) Pure Green color

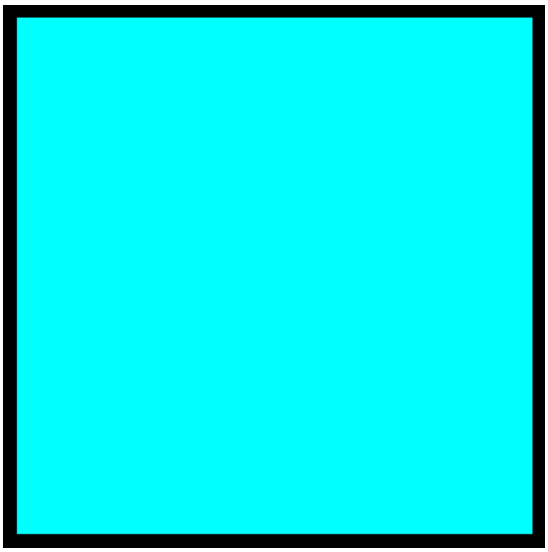


(c) Pure Blue color

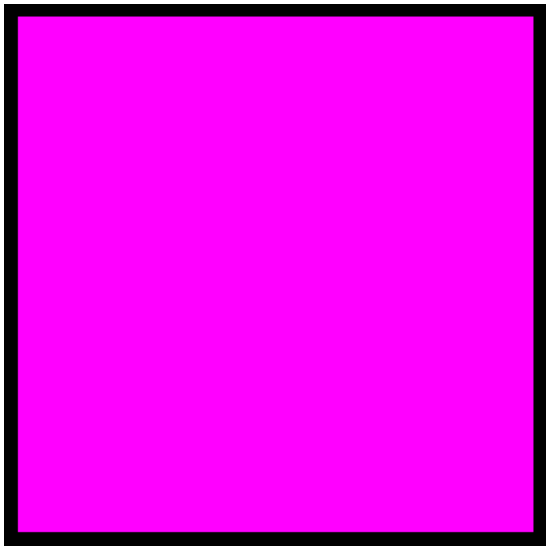
Figure 7: Three images with red, green and blue components only.



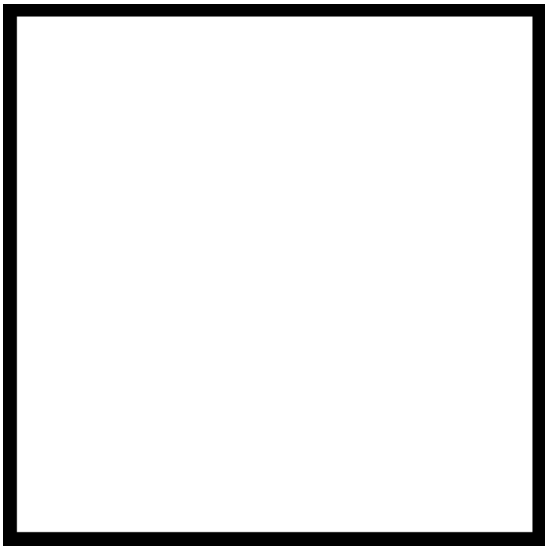
(a) weight = [1 1 0]



(b) weight = [0 1 1]



(c) weight = [1 0 1]



(d) weight = [1 1 1]

Figure 8: Images as a weighted sum of red, green and blue images (=aR + bG + cB).

By adjusting the weights of the primary colours, we can generate a wide range of colours and observe how different proportions affect the final output.

Code Availability

All the code used in this project is available in a public GitHub repository. You can access it at: https://github.com/Computer-Science-Practicum/DIP-Lab-Assignment.