Lab Report: 06

Title: Convert RGB to HSI color model and Grayscale

Course title: Digital Image Processing Laboratory

Course code: CSE-406

4^{th} Year 1^{st} Semester Examination 2023

Date of Submission: 06/10/2024

Submitted to-

Dr. Md. Golam Moazzam

Professor

Department of Computer Science and Engineering

Jahangirnagar University

&

Dr. Morium Akter

Professor

Department of Computer Science and Engineering

Jahangirnagar University

Savar, Dhaka-1342

Class Roll	Exam Roll	Name
353	202165	Shanjida Alam

Experiment Number: 01

Experiment Name: Convert RGB color model to HIS color model

Objectives:

1. Understand the RGB Color Model and HIS Model.
2. Apply Normalization and Scaling Techniques.
3. Optimize for Computational Efficiency

Code-01: Python

```
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image
def rgb2hsi_components(rgb_image):
    rgb_image = np.array(rgb_image, dtype=np.float64) / 255.0
    R = rgb_image[:, :, 0]
    G = rgb_image[:, :, 1]
    B = rgb_image[:, :, 2]
    I = (R + G + B) / 3
    min_RGB = np.minimum(np.minimum(R, G), B)
    S = 1 - (3 / (R + G + B + 1e-8)) * min_RGB
    num = 0.5 * ((R - G) + (R - B))
    den = np.sqrt((R - G) ** 2 + (R - B) * (G - B)) + 1e-8
    theta = np.arccos(num / den)
    H = np.copy(theta)
    H[B > G] = 2 * np.pi - H[B > G]
    H = H / (2 * np.pi)
    return H, S, I
rgb_image = Image.open('nature.jpeg')
H, S, I = rgb2hsi_components(rgb_image)
fig, ax = plt.subplots(2, 2, figsize=(10, 8))
ax[0, 0].imshow(rgb_image)
ax[0, 0].set_title('RGB Image')
ax[0, 0].axis('off')
ax[0, 1].imshow(H, cmap='hsv')
ax[0, 1].set_title('Hue')
ax[0, 1].axis('off')
ax[1, 0].imshow(S, cmap='gray')
ax[1, 0].set_title('Saturation')
ax[1, 0].axis('off')
ax[1, 1].imshow(I, cmap='gray')
```

```
ax[1, 1].set_title('Intensity')
ax[1, 1].axis('off')
plt.tight_layout()
plt.show()
```

Output:

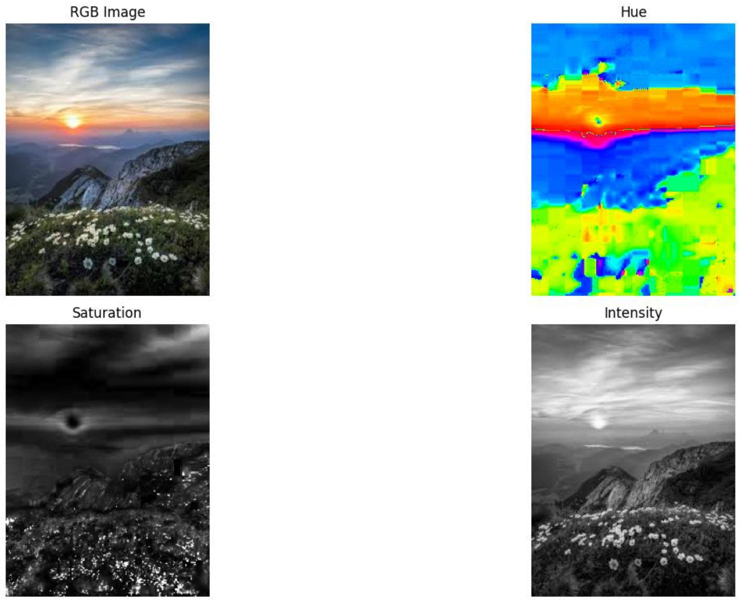

Figure 1.1: Converting RGB color model to HIS color model in python

Explanation:

1. We use the **PIL** library to load the RGB image. The image is normalized to the range [0, 1] by dividing by 255.
2. The **intensity (I)** is the average of the R, G, and B channels.
3. **Saturation (S)** is calculated based on the minimum of the R, G, and B values relative to the sum of the R, G, and B values.
4. **Hue (H)** is calculated using an arccosine formula that considers the relative differences between the R, G, and B channels.
5. We display the original RGB image, along with the Hue, Saturation, and Intensity components using **matplotlib's imshow()** function.

Code-02: MATLAB

```
function [H, S, I] = rgb2hsi_components(rgb_image)
    [rows, cols, ~] = size(rgb_image);
    H = zeros(rows, cols);
    S = zeros(rows, cols);
    I = zeros(rows, cols);
    rgb_image = im2double(rgb_image);
    R = rgb_image(:, :, 1);
    G = rgb_image(:, :, 2);
    B = rgb_image(:, :, 3);
    I = (R + G + B) / 3;
    min_RGB = min(min(R, G), B);
    S = 1 - (3 ./ (R + G + B + eps)) .* min_RGB;
    theta = acos((0.5 * ((R - G) + (R - B))) ./ sqrt((R - G).^2 + (R - B).*(G - B) + eps));
    H(B > G) = 2 * pi - theta(B > G);
    H(G >= B) = theta(G >= B);
    H = H / (2 * pi);
end
rgb_image = imread('nature.jpeg');
[H, S, I] = rgb2hsi_components(rgb_image);

figure;
subplot(2, 2, 1), imshow(rgb_image), title('RGB Image');
subplot(2, 2, 2), imshow(H), title('Hue');
subplot(2, 2, 3), imshow(S), title('Saturation');
subplot(2, 2, 4), imshow(I), title('Intensity');
```

Output:

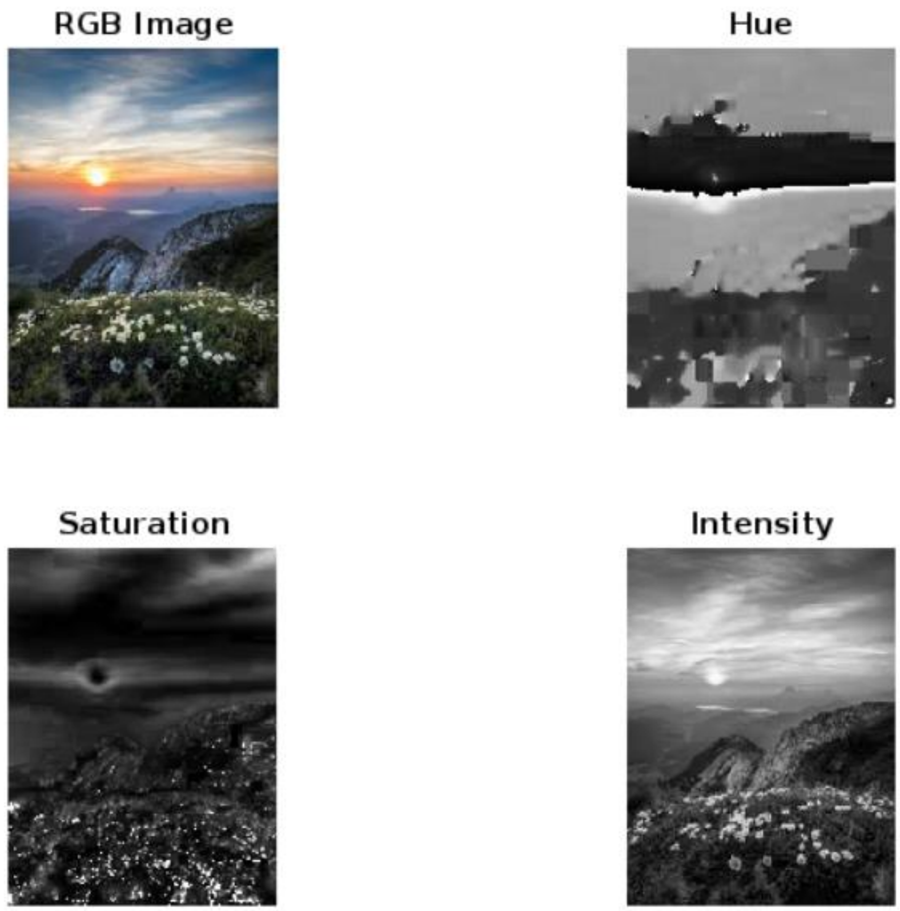

Figure 1.2: Converting RGB color model to HIS color model in MATLAB

Explanation:

1. The hue is calculated using the arccosine formula, adjusted based on the comparison of B and G values.
2. Saturation is calculated based on the minimum RGB value divided by the total sum of RGB components. It is scaled between 0 (no color saturation) and 1 (full color saturation).
3. Intensity is simply the average of the R, G, and B channels.

Experiment Number: 02

Experiment Name: Convert RGB to Grayscale

Objectives:

1. Understand the RGB Color Model.
2. Understand the concept of grayscale, where images contain only shades of gray (intensity values) ranging from black (0) to white (255), without any color information.
3. Optimize for Performance and Efficiency.

Code-01: Python

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
def rgb_components_to_grayscale(image):
    image = image.astype(np.float32) / 255.0
    R = image[:, :, 0]
    G = image[:, :, 1]
    B = image[:, :, 2]
    grayscale_image = 0.2989 * R + 0.5870 * G + 0.1140 * B
    plt.figure(figsize=(10, 8))
    plt.subplot(3, 2, 3)
    plt.imshow(R, cmap='gray')
    plt.title('Red Component')
    plt.axis('off')
    plt.subplot(3, 2, 4)
    plt.imshow(G, cmap='gray')
    plt.title('Green Component')
    plt.axis('off')
    plt.subplot(3, 2, 5)
    plt.imshow(B, cmap='gray')
    plt.title('Blue Component')
    plt.axis('off')
    plt.subplot(3, 2, 6)
    plt.imshow(grayscale_image, cmap='gray')
    plt.title('Grayscale Image')
    plt.axis('off')
    plt.tight_layout()
    plt.show()
image = cv2.imread('nature.jpeg')
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
```

rgb_components_to_grayscale(image)

Output:

Figure 2.1: Convert RGB to Grayscale python code

Explanation:

1. Importing Libraries such as cv2, numpy, matplotlib.pyplot
2. Defining the Function **rgb_components_to_grayscale(image)**
3. Plotting the Components and Grayscale Image

Code-02: MATLAB

```
function rgb_components_to_grayscale(image)

    image = double(image) / 255;
    R = image(:, :, 1);
    G = image(:, :, 2);
    B = image(:, :, 3);
    grayscale_image = 0.2989 * R + 0.5870 * G + 0.1140 * B;
    figure;

    subplot(3, 2, 2), imshow(R), title('Red Component');
    subplot(3, 2, 3), imshow(G), title('Green Component');
    subplot(3, 2, 4), imshow(B), title('Blue Component');
    subplot(3, 2, 5), imshow(grayscale_image), title('Grayscale Image');
    subplot(3, 2, 1), imshow(image), title('Original Image');
end
image = imread('nature.jpeg');
rgb_components_to_grayscale(image);
```

Output:

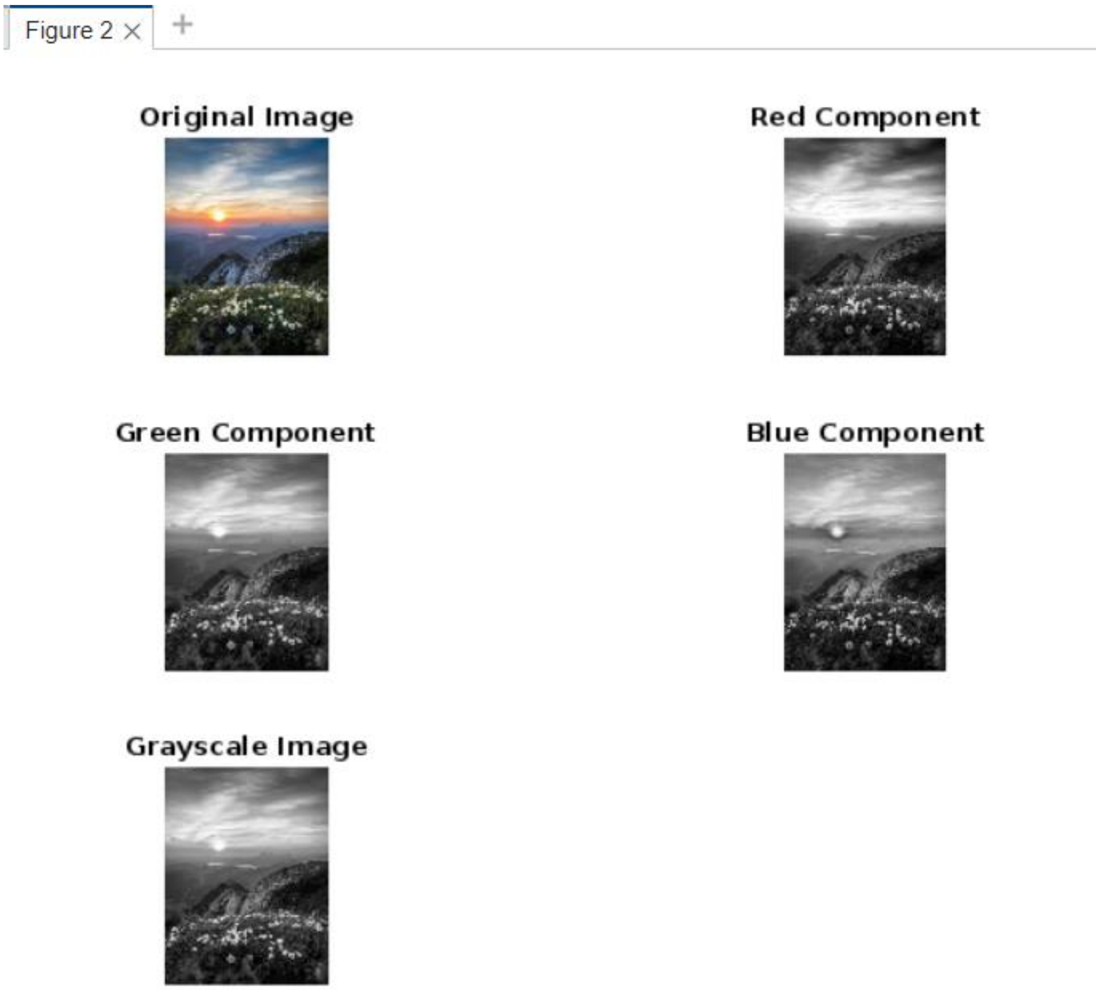

Figure 2.2: Converting the RGB model to Grayscale image in MATLAB

Explanation:

1. This defines a function called **rgb_components_to_grayscale** that takes an image (RGB image) as input.
2. The image is converted from uint8 (range 0-255) to double (range 0-1) by dividing by 255. This is done because many image processing operations work best with floating-point values between 0 and 1.
3. The image is a 3D matrix, where the third dimension contains the color channels.
4. This formula is used to convert the RGB image to grayscale. The weights are based on the human eye's sensitivity to different colors.