Assignment Report

Advanced Image Processing and Computer Vision

Assignment 1

Hardik Soni 20CS30023
Khushboo Singhania 19CS30023

30th January, 2024

Part 1: Contrast Enhancement

Introduction

Contrast enhancement is a fundamental image processing technique that aims to improve the visual quality of images by adjusting the distribution of pixel intensities. This can make features and details more discernible, especially in images with low contrast. This report explores a method for contrast enhancement that selectively targets the achromatic component of an image while preserving the color information.

Methodology

1. Image Reading:

- The code begins by loading the input image "contrast.jpg" using the OpenCV library's cv2.imread() function.

2. HSV Color Space Conversion:

- The image is converted to the HSV (Hue, Saturation, Value) color space using cv2.cvtColor(image, cv2.COLOR_BGR2HSV).

- HSV is advantageous for contrast enhancement as it separates color information (Hue and Saturation) from brightness (Value).

3. Contrast Enhancement:

- The Value (V) channel is isolated: v_channel = hsv_image[:,:,2].

- Contrast Limited Adaptive Histogram Equalization (CLAHE) is applied to the V channel using cv2.createCLAHE() and clahe.apply(v_channel).

- CLAHE enhances contrast by dividing the image into small tiles and applying histogram equalization locally within each tile.

- clipLimit=2.0 controls the extent of contrast enhancement to prevent over-amplification of noise.

- tileGridSize=(8, 8) sets the size of tiles for local histogram equalization.

- The enhanced V channel is replaced back into the HSV image: hsv_image[:,:,2] = enhanced_v_channel.

4. Color Space Conversion Back to BGR:

- The image is converted back to the BGR color space for display or saving: enhanced_image = cv2.cvtColor(hsv_image, cv2.COLOR_HSV2BGR).

5. Image Saving:

- The enhanced image is saved as "enhanced_contrast.jpg" using cv2.imwrite().

Key Formulae:

- HSV color space conversion:

- H = arccos((R - G) / sqrt((R - G)^2 + (R - B)(G - B)))

- S = 1 - 3 min(R, G, B) / (R + G + B)

- V = max(R, G, B)

- CLAHE algorithm (simplified):

- For each tile:

  - Compute a histogram of pixel intensities.

- Clip the histogram based on clipLimit to redistribute excessive frequencies.

- Apply histogram equalization to redistribute intensities within the tile.

- Reconstruct the tile using the equalized histogram.

Conclusion

By selectively enhancing the Value channel in the HSV color space, this method improves contrast while maintaining the original color characteristics of the image. This approach is particularly effective for images where contrast is primarily limited by brightness variations, and it can be applied to various image analysis and enhancement tasks.

Enhanced_contrast.jpg

contrast.jpg

Part 2: Saturation-Desaturation

Introduction:

The goal of this project is to implement saturation and desaturation operations on a given color image, "flower.jpg," using plausible coordinates of vertices of the gamut triangle in the CIE chromaticity chart. Additionally, the color matching functions in XYZ space for every monochromatic color are provided in the attached "ciexyz31_1.csv" file.

Methods:

1. Color Matching Functions:

The first step involves loading the color matching functions (CMFs) from the CSV file. The CMFs provide information about the tristimulus values (X, Y, Z) for each monochromatic color at different wavelengths.

```
cmf_data = pd.read_csv("ciexyz31_1.csv", header=None, names=["Wavelength",
"X", "Y", "Z"])
```

2. Conversion to XYZ Color Space:

The original image ("flower.jpg") is read, and its RGB values are converted to the XYZ color space. This conversion is crucial for subsequent operations involving chromaticity coordinates.

```
original_image = cv2.imread("flower.jpg")
original_image_rgb = cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB)
original_image_xyz = cv2.cvtColor(original_image_rgb, cv2.COLOR_RGB2XYZ)
```

3. Chromaticity Points and Plotting:

Chromaticity points for the original image are computed by converting the wavelengths to RGB values. These points are then plotted on the CIE chromaticity diagram.

```
chromaticity_points_original = np.zeros((len(wavelengths), 3))
for i, wavelength in enumerate(wavelengths):
    chromaticity_points_original[i, :] = wavelength_to_rgb(wavelength)

plt.scatter(chromaticity_points_original[:, 1] /
chromaticity_points_original.sum(axis=1),
            chromaticity_points_original[:, 2] /
chromaticity_points_original.sum(axis=1),
            c=chromaticity_points_original[:, 0],
            cmap='viridis',
            marker='o',
            s=20)
```

4. Transformation and Saturation-Desaturation Operations:

The gamut triangle is defined by three vertices, and an affine transformation is applied to the original image to simulate color manipulation. Saturation and desaturation operations are performed on the transformed image.

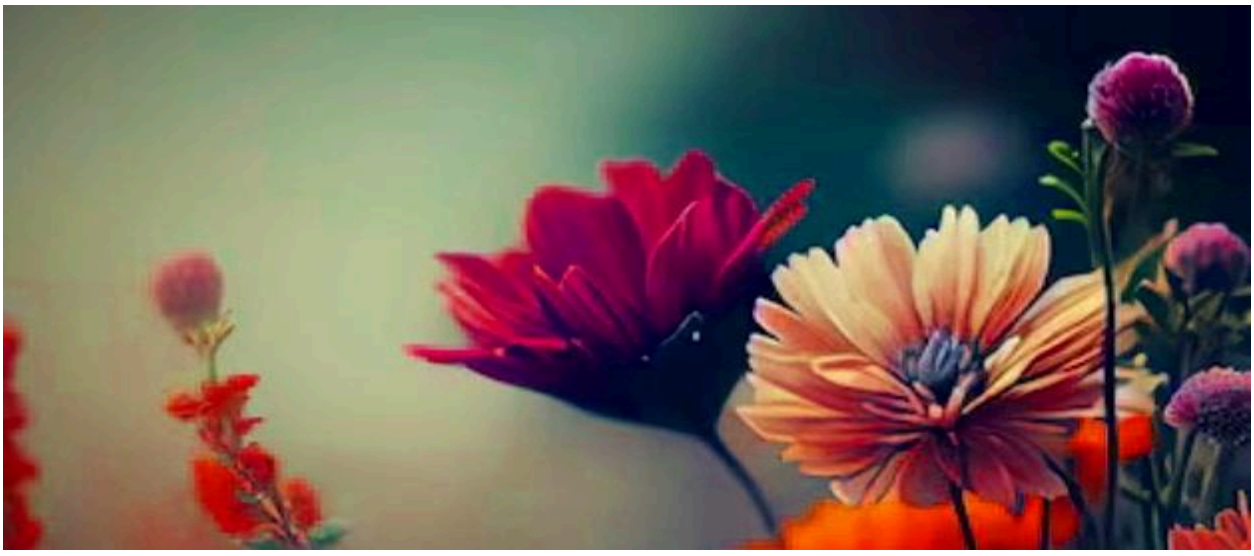
```
max_saturated_image = saturate_image(transformed_image_rgb, 2.0)
desaturated_image = desaturate_image(transformed_image_rgb, 0.5)
saturated_desaturated_image = saturate_image(desaturated_image, 2.0)
```

Results:

1. Maximally Saturated Image:

The maximally saturated image is obtained by enhancing the saturation of the transformed image.

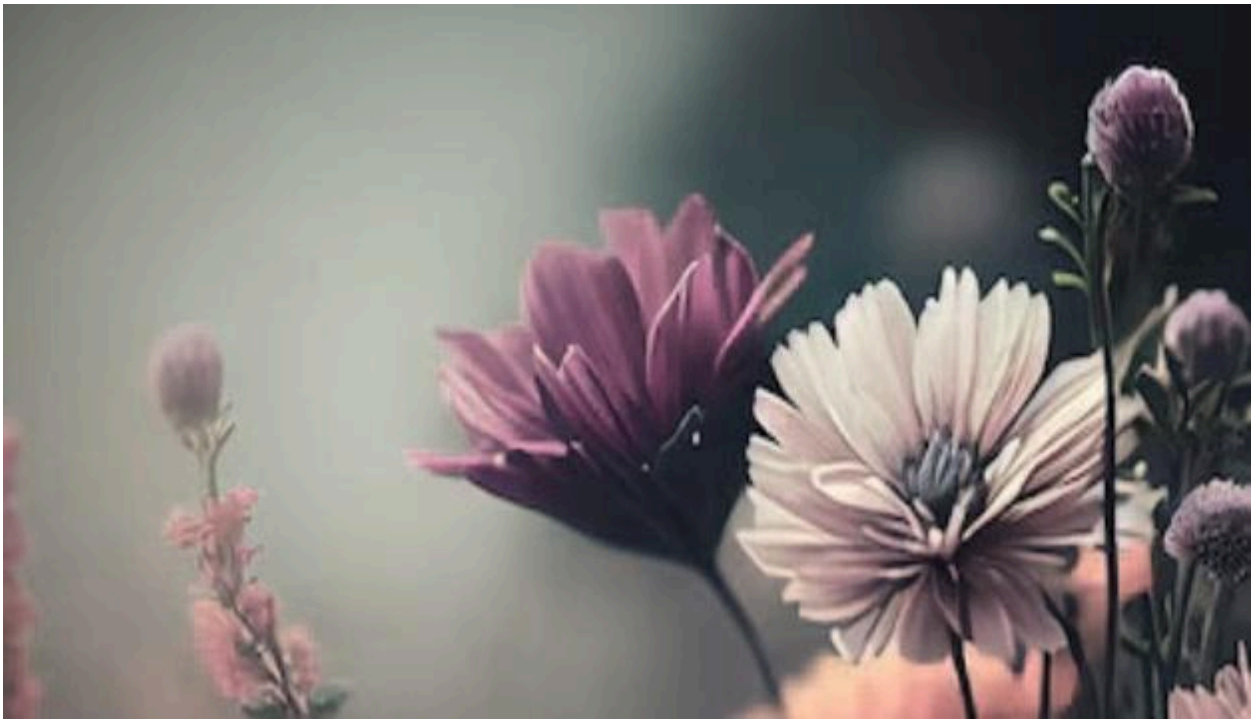
```
max_saturated_image = saturate_image(transformed_image_rgb, 2.0)
cv2.imwrite("max_saturated_image.jpg", cv2.cvtColor(max_saturated_image,
cv2.COLOR_RGB2BGR))
```



2. Desaturated Image:

The desaturated image is generated by reducing the saturation of the transformed image.

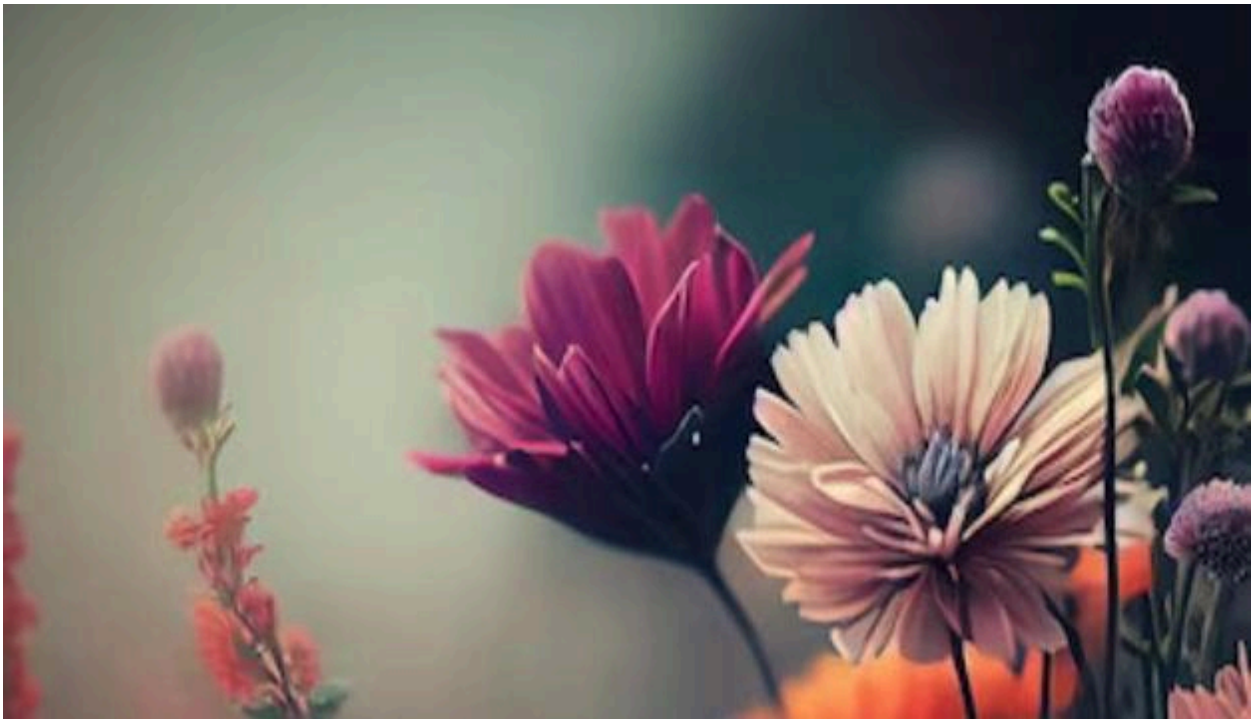
```
desaturated_image = desaturate_image(transformed_image_rgb, 0.5)
cv2.imwrite("desaturated_image.jpg", cv2.cvtColor(desaturated_image,
cv2.COLOR_RGB2BGR))
```



3. Saturated-Desaturated Image:

This image is created by first desaturating the transformed image and then saturating it.

```
saturated_desaturated_image = saturate_image(desaturated_image, 2.0)
cv2.imwrite("saturated_desaturated_image.jpg",
cv2.cvtColor(saturated_desaturated_image, cv2.COLOR_RGB2BGR))
```



Conclusion:

In conclusion, the implemented saturation and desaturation operations successfully manipulate the color characteristics of the given image based on plausible coordinates in the CIE chromaticity chart. The chromaticity points and resulting images provide insights into the impact of these operations on the color representation.

Future Work:

Future work may involve exploring different gamut triangle configurations, experimenting with alternative color spaces, and assessing the perceptual impact of saturation-desaturation operations.

Overall, this project demonstrates the integration of color science principles with image processing techniques to achieve meaningful color manipulations on digital images.