Addis Ababa University

Master's in Artificial Intelligence

Digital Image Processing (DIP) Laboratory Manual

Prepared by: Dr. Natnael Argaw Wondimu

Environment: GNU Octave

Preface

This manual is prepared in a workbook style for AI Master's students at Addis Ababa University. It provides a hands-on introduction to Digital Image Processing using GNU Octave, covering basic to advanced topics. Each lab includes objectives, theoretical background, procedures, Octave code, checkpoints, try-it-yourself prompts, and collaborative assignments. The manual is intended to build foundational knowledge in Digital Image Processing and serve as a base for advanced studies in Computer Vision.

Chapter 2: Color Science and Point Operations

Objective

To understand the fundamentals of color image representation and perform point-based image operations using GNU Octave.

1. Color Image Representation

1.1 What is a Color Image?

Description: A color image consists of three channels: Red, Green, and Blue (RGB). Each channel represents the intensity of that primary color per pixel.

1.2 Reading and Displaying a Color Image

Code Snippet:

```
color_img = imread('peppers.png');
imshow(color_img);
title('Original Color Image');
```

Output Description: Displays the full-color image of “peppers” in RGB format.

1.3 Separating RGB Channels

Code Snippet:

```
R = color_img(:,:,1);
G = color_img(:,:,2);
B = color_img(:,:,3);

subplot(1,3,1); imshow(R); title('Red Channel');
subplot(1,3,2); imshow(G); title('Green Channel');
subplot(1,3,3); imshow(B); title('Blue Channel');
```

Output Description: Displays the red, green, and blue channels separately. Each appears as a grayscale image showing intensities.

2. Point Operations

2.1 What are Point Operations?

Description: Point operations manipulate pixel values independently of neighboring pixels. Examples include contrast adjustment, brightness scaling, and negative transformation.

2.2 Brightness Increase

Code Snippet:

```
bright_img = color_img + 50;
imshow(bright_img);
title('Brightness Increased');
```

Output Description: The image appears brighter. Caution: values beyond 255 are saturated.

2.3 Brightness Decrease

Code Snippet:

```
dark_img = color_img - 50;
imshow(dark_img);
title('Brightness Decreased');
```

Output Description: Image appears darker. Values below 0 are clipped.

2.4 Image Negative

Code Snippet:

```
negative_img = 255 - color_img;
imshow(negative_img);
title('Negative Image');
```

Output Description: Inverts color intensities. Light areas become dark and vice versa.

2.5 Contrast Stretching

Code Snippet:

```
double_img = im2double(color_img);
contrast_img = imadjust(double_img, stretchlim(double_img), []);
imshow(contrast_img);
title('Contrast Stretched Image');
```

Output Description: Enhances contrast by redistributing intensity levels across a wider range.

3. Summary

- RGB images consist of three 2D matrices.
- Each point operation affects pixel values directly and independently.
- Octave supports point operations with simple arithmetic and built-in functions.

Suggested Exercises

1. Convert the RGB image to grayscale using the luminosity method.
2. Create a function for dynamic brightness adjustment using a slider input.
3. Perform histogram equalization on each channel separately.