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.

Digital Image Processing Lab Manual

Chapter 7: Eigenimages (GNU Octave)

Objective

To understand and implement the concept of **eigenimages** using Principal Component Analysis (PCA) in GNU Octave. This chapter demonstrates how PCA reduces the dimensionality of image data and captures its principal features.

1. What are Eigenimages?

Description: Eigenimages are a set of basis images derived using PCA that represent the most significant variations in a dataset of similar images (e.g., face datasets). They are used in compression, recognition, and visualization.

2. Preparing the Dataset

2.1 Image Dataset Format

Description: All images must be: - Grayscale - Same resolution (e.g., 64x64 or 100x100) - Flattened into vectors

2.2 Loading Multiple Images from a Folder

Code Snippet:

```
pkg load image;
img_dir = 'faces/';  % Folder with images
img_files = dir(fullfile(img_dir, '*.pgm'));

% Preallocate matrix
img_size = [64, 64];
num_imgs = length(img_files);
data = zeros(prod(img_size), num_imgs);

for i = 1:num_imgs
  img = im2double(imread(fullfile(img_dir, img_files(i).name)));
  img = imresize(img, img_size);
  data(:, i) = img(:);  % Flatten and store as column
end
```

Output Description: Loads and flattens each grayscale image into a column vector. The result is a matrix of size (pixels × images).

3. Mean Face and Data Centering

Code Snippet:

```
mean_face = mean(data, 2);
centered_data = data - mean_face;
```

Output Description: Calculates the average image (mean face) and centers each image vector by subtracting the mean.

Visualize Mean Face

```
imshow(reshape(mean_face, img_size), []);
title('Mean Face');
```

Output Description: Displays the mean appearance across all images.

4. Computing Eigenimages using PCA

4.1 Covariance Matrix and Eigen Decomposition

Code Snippet:

```
C = centered_data' * centered_data;  % Smaller covariance matrix
[Vec, Val] = eig(C);

% Project back to image space
eigenfaces = centered_data * Vec;

% Normalize eigenfaces
for i = 1:size(eigenfaces, 2)
   eigenfaces(:, i) = eigenfaces(:, i) / norm(eigenfaces(:, i));
end
```

Output Description: Computes the eigenfaces in the original image space.

4.2 Displaying Top 5 Eigenfaces

Code Snippet:

```
figure;
for i = 1:5
   subplot(1,5,i);
   imshow(reshape(eigenfaces(:, end-i+1), img_size), []);
   title(['Eigenface ', num2str(i)]);
end
```

Output Description: Visualizes the top 5 eigenimages, which capture the most variation in the dataset.

5. Image Reconstruction with Eigenfaces

5.1 Using Top k Eigenfaces

Code Snippet:

```
k = 20;
U_k = eigenfaces(:, end-k+1:end);

proj = U_k' * centered_data(:, 1);  % Projection of first image
reconstructed = U_k * proj + mean_face;

imshow(reshape(reconstructed, img_size), []);
title(['Reconstructed Image with ', num2str(k), ' Eigenfaces']);
```

Output Description: Shows the reconstructed version of the first image using limited eigenfaces.

6. Summary

- **Eigenimages** reveal directions of highest variance in the data.
- PCA allows **compression** and **noise reduction**.
- Reconstruction quality increases with more eigenfaces.

Suggested Exercises

1. Vary number of eigenfaces (k = 5, 10, 50) and compare reconstructions.
2. Classify test images using nearest neighbor in PCA space.
3. Create a GUI to slide through eigenfaces.