Assignment 2

Passing marks for the assignment is 65 points

Objective

The objective of this assignment is to understand and apply image processing techniques on a dataset.

You have image datasets on blackboard. You are also allowed to use any other dataset from outside sources (with the basic description).

For each task in this assignment:

1. Clarity of Explanation: 30%
2. Quality of Results: 30%
3. Insight in Discussion: 40%

Fourier Transform (20 points)

1. Load a grayscale image and apply the 2D Discrete Fourier Transform (DFT) to it. Visualize the original image and its frequency spectrum (magnitude). Submit the images, and explanation.
2. Implement a low-pass filter in the frequency domain to remove high-frequency noise from an image. Compare the filtered image with the original image. Submit images, and analysis of the results.
3. Implement a high-pass filter to enhance the edges in an image. Visualize the filtered image and discuss the effects observed. Submit images, and explanation.
4. Implement an image compression technique using Fourier Transform by selectively keeping only a certain percentage of the Fourier coefficients. Evaluate the quality of the reconstructed image as you vary the percentage of coefficients used. Submit the images, and your observations on image quality and compression ratio.

PCA (25 points)

Select a set of grayscale images (e.g., faces, landscapes, or any other category).

Normalize the images by scaling pixel values to a range [0, 1].

PCA Implementation:

Write a Python function to perform PCA on the images.

Implement the following steps:

1. Convert the image into a 2D matrix where each row represents an image and each column represents a pixel value.
2. Compute the covariance matrix of the image data.
3. Calculate the eigenvalues and eigenvectors of the covariance matrix.
4. Sort the eigenvectors based on the eigenvalues in descending order.
5. Select the top k eigenvectors to form the principal components.
6. Project the original images onto the lower-dimensional subspace defined by the selected principal components.

Reconstruction of Images:

1. Using the selected principal components, reconstruct the images.
2. Compare the reconstructed images with the original images to observe the effects of dimensionality reduction.

Experimentation:

1. Vary the number of principal components (k) and observe the impact on the quality of the reconstructed images.
2. Plot the variance explained by the principal components and determine the optimal number of components that balances compression and quality.

Visual Analysis:

1. Display the original images alongside the reconstructed images for different values of k.
2. Comment on the visual quality of the images and how much information is lost during compression.

Error Analysis:

1. Compute the Mean Squared Error (MSE) between the original and reconstructed images.
2. Analyze the trade-off between compression and reconstruction error.

Image processing (55 points)

HOG Features (12 points)

1. Write a Python script to compute the HOG features of a given image using a library such as OpenCV or scikit-image.
2. Apply your implementation to at least three different images, including both simple and complex scenes.
3. Visualize the original image, the gradient image, and the HOG feature image.
4. Compare the HOG features extracted from different images. Discuss the impact of varying parameters like cell size, block size, and the number of bins on the resulting HOG descriptors.

Local Binary Patterns (LBP) (13 points)

1. Write a Python function to compute the LBP of a given grayscale image. Use the basic 8-neighbor LBP method.
2. Your function should output the LBP image, where each pixel is replaced by its corresponding LBP value.
3. Write a Python function to compute the histogram of the LBP image.
4. Plot the histogram and explain what it represents in terms of the texture features of the image.
5. Apply your LBP function to at least three different grayscale images (e.g., a natural scene, a texture, and a face image).
6. Generate and compare the histograms of the LBP images.
7. Discuss the differences in the histograms and what they tell you about the textures of the different images.

Implement a blob detection algorithm (15 points).

1. Apply the blob detection algorithm to one of the provided image datasets on blackborad.
2. Visualize the detected blobs on the original images, marking each detected blob with a circle or bounding box.
3. Calculate and display relevant statistics for each image, such as the number of blobs detected, their sizes, and positions.
4. Evaluate and discuss the effect of different parameters in the algorithms on the detection of different blobs.

Implement a contour detection algorithm. (15 points)

1. Apply the contour detection algorithm to the same image dataset.
2. Visualize the detected contours on the original images, marking each contour with a different color.
3. Calculate and display relevant statistics for each image, such as the number of contours detected, contour area, and perimeter.
4. Compare the results of blob detection and contour detection for the chosen dataset.
5. Discuss the advantages and limitations of each technique.
6. Analyze the impact of different parameters (e.g., threshold values, filter sizes) on the detection results.
7. Provide examples where one technique might be more suitable than the other.