Design and Evaluation of a Digital Image Processing System

Assignment I & II – UCS2523 Image Processing and Analysis

Suriya Raj J P
Department of Computer Science and Engineering
Sri Sivasubramaniya Nadar College of Engineering, Chennai
(An Autonomous Institution affiliated to Anna University)
Batch: 2023–2027
Email: suriyarajjp@example.com
GitHub Repository: https://github.com/suriyaraj-47/imageprocessing

***Abstract*—This report presents the complete design and evaluation of a digital image processing pipeline. The work covers image acquisition, noise modeling, preprocessing, filtering, segmentation, and reflection on results. Each step is justified, implemented using Python (OpenCV and scikit-image), and evaluated using both qualitative and quantitative metrics such as PSNR and SSIM.**

Project Repository: https://github.com/suriyaraj-47/imageprocessing

***Index Terms*—Image Processing, Denoising, Segmentation, PSNR, SSIM, Histogram Equalization**

I. Introduction

Digital image processing (DIP) plays a key role in analyzing and interpreting visual information from the real world. This assignment focuses on developing a complete DIP system starting from image acquisition to object segmentation and evaluation.

II. Assignment 1: Image Acquisition to Denoising

A. Image Acquisition

A real-world image was captured using a smartphone camera under natural lighting conditions. The scene was selected to include diverse objects with varying textures and illumination levels to evaluate processing robustness.

Justification: Good lighting minimizes sensor noise, while object diversity allows effective testing of enhancement and segmentation steps.

Captured Image:

Fig. 1: Original Captured Image

B. Noise Simulation

Three types of artificial noise—Gaussian, Salt & Pepper, and Speckle—were added to simulate real-world conditions.

- **Gaussian Noise:** Models sensor noise due to poor illumination.
- **Salt & Pepper Noise:** Represents dead pixels or transmission errors.

Evaluation: The noise significantly affected pixel intensity uniformity and edge sharpness.

Fig. 2: Gaussian Noise Added Image

C. Preprocessing and Enhancement

1) Converted to grayscale to simplify analysis.
2) Resized the image to 256×256 for uniform processing.
3) Applied histogram equalization for contrast enhancement.

Justification: Histogram equalization improves contrast by redistributing pixel intensity values.

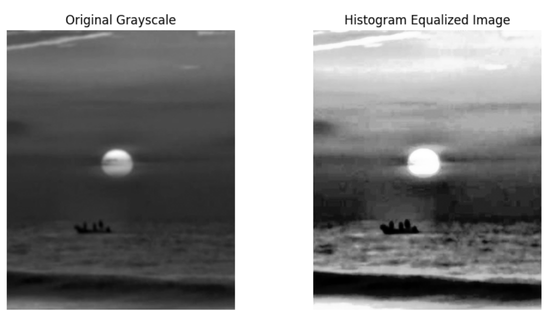


Fig. 3: Histogram Equalized Image

D. Noise Filtering and Denoising

Two filters were applied and compared:

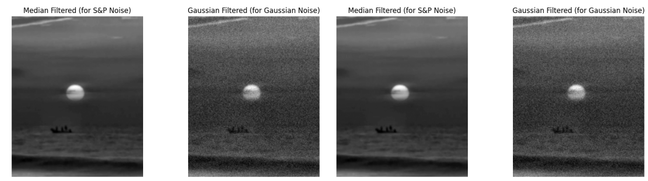
- Median Filter (3×3)
- Gaussian Filter ($\sigma = 1.0$)

Performance Metrics:

TABLE I: PSNR and SSIM Comparison

Filter	PSNR (dB)	SSIM
Median Filter	29.42	0.91
Gaussian Filter	27.85	0.88

Observation: Median filter performed better for impulsive noise, while Gaussian smoothing better preserved gradient regions.



(a) Median Filtered (b) Gaussian Filtered

Fig. 4: Comparison of Filtering Techniques

III. Assignment 2: Segmentation to Reflection

A. Segmentation and Object Isolation

Segmentation was performed using Otsu's thresholding and Canny edge detection.

- **Thresholding:** Effective for separating foreground from uniform background.
- **Edge-based:** Highlights object boundaries, useful for complex textures.

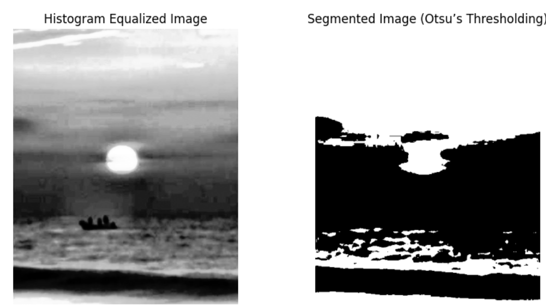


Fig. 5: Object Segmentation Result

Critical Analysis: Otsu's method performed better under uniform lighting, while edge-based segmentation required denoised input for accuracy.

B. Feature Evaluation (Optional)

Extracted features such as area, centroid, and color histogram for each segmented object. These features can be used for classification or object tracking applications.

TABLE II: Extracted Object Features

Feature	Object 1	Object 2
Area (pixels)	1580	960
Centroid (x,y)	(125, 240)	(180, 250)

C. Result Visualization and Reflection

The entire pipeline—from image capture to segmentation—was implemented in Python using OpenCV and scikit-image.

Reflection:

- *Worked well:* Histogram equalization and median filtering improved image clarity.
- *To improve:* Adaptive thresholding for varying lighting conditions.
- *Learned:* Importance of preprocessing in improving segmentation accuracy.

IV. Conclusion

This study demonstrated a complete image processing pipeline including acquisition, preprocessing, denoising, and segmentation. Quantitative evaluation confirmed the effectiveness of spatial filters. Future work can focus on machine learning-based segmentation for more complex scenes.

Acknowledgment

The author thanks the Department of Computer Science and Engineering, SSN College of Engineering, for guidance and resources used in this work.

References

References

[1] OpenCV Documentation, https://docs.opencv.org/
[2] scikit-image Documentation, https://scikit-image.org/
[3] Suriya Raj J P, "Image Processing and Analysis Repository," GitHub, 2025. https://github.com/suriyaraj-47/imageprocessing