Palawan State University
College of Sciences
Computer Studies Department

DIGITAL IMAGE PROCESSING LABORATORY EXERCISE #11

Implementation of image restoring techniques

Implementation of image restoring techniques in digital image processing involves using various algorithms and methods to recover or enhance the quality of images that have been degraded by factors such as noise, blur, or other distortions. The goal is to reconstruct the original image as accurately as possible. Here are the key steps and techniques involved in this process:

1. **Identifying the Degradation Model:** Before restoration, it's essential to understand the nature of the degradation. Common models include:
   - **Additive Noise:** Random noise added to the image, often modeled as Gaussian noise.
   - **Blur:** Caused by motion, defocus, or atmospheric turbulence, often modeled by a Point Spread Function (PSF).
   - **Compression Artifacts:** Degradation due to lossy compression algorithms like JPEG.

2. **Noise Reduction Techniques:** Noise can be reduced using several techniques:
   - **Spatial Domain Filters:**
     - **Mean Filter:** Averages pixel values within a neighborhood.
     - **Median Filter:** Replaces each pixel's value with the median value of the neighboring pixels.
     - **Gaussian Filter:** Uses a Gaussian function to smooth the image.
   - **Frequency Domain Filters:**
     - **Fourier Transform:** Applies a low-pass filter to remove high-frequency noise.
     - **Wavelet Transform:** Decomposes the image into different frequency components and processes each component separately.
3. **Deconvolution:** aims to reverse the effects of blurring by using the known PSF.
   - **Wiener Filter:** A linear filter that minimizes the mean square error between the estimated and the true image.
   - **Lucy-Richardson Algorithm:** An iterative method that applies maximum likelihood estimation to restore the image.
   - **Blind Deconvolution:** Used when the PSF is unknown; it simultaneously estimates the PSF and restores the image.
4. **Regularization Techniques:** To deal with ill-posed problems where direct inversion is unstable:
   - **Tikhonov Regularization:** Adds a constraint to stabilize the inversion process.
   - **Total Variation Regularization:** Preserves edges while smoothing the image.
5. **Machine Learning-Based Techniques:** Modern approaches often use machine learning and deep learning models:
   - **Denoising Autoencoders:** Neural networks trained to remove noise from images.
   - **Generative Adversarial Networks (GANs):** Used for various restoration tasks by generating high-quality images from degraded inputs.
   - **Convolutional Neural Networks (CNNs):** Specifically designed architectures like U-Net or DnCNN for image denoising and deblurring.
6. **Inpainting:** Restoring missing parts of an image:
   - **Exemplar-Based Methods:** Copying similar patches from the surrounding areas.
   - **Diffusion-Based Methods:** Propagating the known information into the missing areas using PDEs.
   - **Deep Learning Methods:** Using CNNs to predict and fill in the missing parts.
7. **Evaluation Metrics:** Assessing the quality of the restored images using:
   - **Peak Signal-to-Noise Ratio (PSNR):** Measures the ratio between the maximum possible power of a signal and the power of corrupting noise.

- **Structural Similarity Index (SSIM):** Assesses the perceived quality by comparing local patterns of pixel intensities.
- **Mean Squared Error (MSE):** Measures the average squared difference between the original and restored images.

```
import cv2
import numpy as np

# Load an image from file
image = cv2.imread('path_to_your_image.jpg')

# Check if the image was successfully loaded
if image is None:
    print("Error: Could not open or find the image.")
else:
    # Resize the image
    width = 800
    height = 600
    resized_image = cv2.resize(image, (width, height))

    # Rotate the image
    # Get the dimensions of the image
    (h, w) = image.shape[:2]
    # Calculate the center of the image
    center = (w // 2, h // 2)
    # Rotate the image by 45 degrees
    M = cv2.getRotationMatrix2D(center, 45, 1.0)
    rotated_image = cv2.warpAffine(image, M, (w, h))

    # Translate the image
    # Define the translation matrix
    tx, ty = 100, 50  # Shift right by 100 pixels and down by 50 pixels
    T = np.float32([[1, 0, tx], [0, 1, ty]])
    translated_image = cv2.warpAffine(image, T, (w, h))

    # Display the original and transformed images
    cv2.imshow('Original Image', image)
    cv2.imshow('Resized Image', resized_image)
    cv2.imshow('Rotated Image', rotated_image)
    cv2.imshow('Translated Image', translated_image)

    # Wait for a key press and close the windows
    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

Exercise #11

Implementation of image restoring techniques

Name:
Year/Block:
Application/Software:

1. Codes
2. Output
3. Answer the following questions:

   A. What are the key differences between spatial domain filters and frequency domain filters in noise reduction techniques, and how do they affect the quality of the restored image?

   B. How does the Wiener filter differ from the Lucy-Richardson algorithm in deconvolution, and in what scenarios would one be preferred over the other for image restoration?

   C. In the context of using machine learning-based techniques for image restoration, what are the advantages and limitations of using Generative Adversarial Networks (GANs) compared to Convolutional Neural Networks (CNNs)?