Noise Addition in Images using Python

Your Name

April 7, 2025

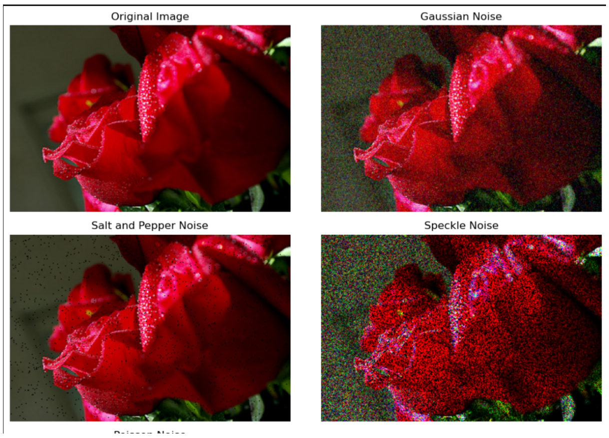
Introduction

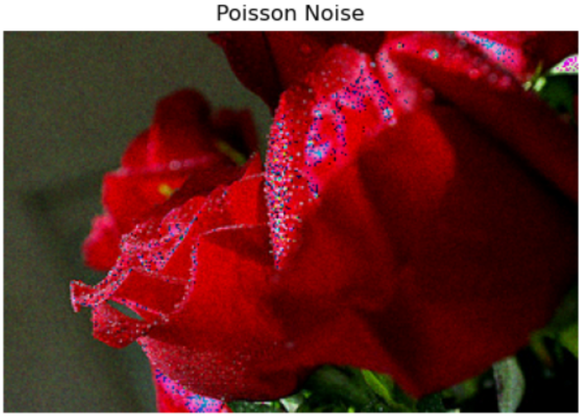
This document demonstrates the addition of different types of noise to an image using Python and OpenCV. The noise types include Gaussian noise, Salt-and-Pepper noise, Speckle noise, and Poisson noise.

Python Code

```
import numpy as np
import cv2
import matplotlib.pyplot as plt

# Function to add Gaussian Noise
def add_gaussian_noise(image, mean=0, sigma=25):
    row, col, ch = image.shape
    gauss = np.random.normal(mean, sigma, (row, col, ch))
    noisy = np.clip(image + gauss, 0, 255)
    return noisy.astype(np.uint8)

# Function to add Salt-and-Pepper Noise
def add_salt_and_pepper_noise(image, prob=0.02):
    noisy = image.copy()
    row, col, ch = image.shape
    s_vs_p = 0.5
    amount = prob
    # Salt noise
    num_salt = int(amount * row * col * s_vs_p)
    salt_coords = [np.random.randint(0, i-1, num_salt) for i in image.shape]
    noisy[salt_coords[0], salt_coords[1]] = 1
    # Pepper noise
    num_pepper = int(amount * row * col * (1.0 - s_vs_p))
    pepper_coords = [np.random.randint(0, i-1, num_pepper) for i in image.shape]
    noisy[pepper_coords[0], pepper_coords[1]] = 0
    return noisy

# Function to add Speckle Noise
def add_speckle_noise(image):
    row, col, ch = image.shape
    gauss = np.random.randn(row, col, ch)
    noisy = np.clip(image + image * gauss, 0, 255)
    return noisy.astype(np.uint8)

# Function to add Poisson Noise
def add_poisson_noise(image):
    noisy = np.random.poisson(image / 255.0 * 256).astype(np.uint8)
    return noisy

# Read an example image
image = cv2.imread('/home/ai-ds3/Downloads/6.jpg')  # Replace with your image path
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
```

```python
# Adding different noises
gaussian_noisy_image = add_gaussian_noise(image)
# Adding different noises
salt_and_pepper_noisy_image = add_salt_and_pepper_noise(image)
speckle_noisy_image = add_speckle_noise(image)
poisson_noisy_image = add_poisson_noise(image)

# Display the original and noisy images
plt.figure(figsize=(10, 10))

# Original Image
plt.subplot(3, 2, 1)
plt.imshow(image)
plt.title("Original Image")
plt.axis('off')

# Gaussian Noise Image
plt.subplot(3, 2, 2)
plt.imshow(gaussian_noisy_image)
plt.title("Gaussian Noise")
plt.axis('off')

# Salt-and-Pepper Noise Image
plt.subplot(3, 2, 3)
plt.imshow(salt_and_pepper_noisy_image)
plt.title("Salt and Pepper Noise")
plt.axis('off')

# Speckle Noise Image
plt.subplot(3, 2, 4)
plt.imshow(speckle_noisy_image)
plt.title("Speckle Noise")
plt.axis('off')

# Poisson Noise Image
plt.subplot(3, 2, 5)
plt.imshow(poisson_noisy_image)
plt.title("Poisson Noise")
plt.axis('off')

# Display the images
plt.tight_layout()
plt.show()
```





Output Images

Conclusion

Noise simulation is useful in image processing research to test and develop denoising algorithms. This script helps understand the visual effects of various types of noise on a clean image.