Assignment Number: 04

Name: Mihir Unmesh Patil
Roll No: TYCOC213
Batch: C/C-3

Title

Implementation of Edge Detection Techniques Using Roberts, Prewitt, Sobel, and Laplacian Operators

---

Aim

To develop a Python program that applies various edge detection techniques (Roberts, Prewitt, Sobel, and Laplacian) to a grayscale image and visualizes the results for analysis.

Objectives

1. To understand and implement edge detection algorithms using convolution with specific kernels.
2. To compare the effectiveness of Roberts, Prewitt, Sobel, and Laplacian operators in identifying edges in an image.
3. To display the original image and the edge-detected outputs for X-axis, Y-axis, and combined edge magnitude.
4. To handle errors and edge cases, such as invalid image loading or no detectable edges.

Theory

Edge detection is a fundamental technique in image processing used to identify boundaries or significant changes in pixel intensity within an image. These boundaries often correspond to object outlines, making edge detection crucial for tasks like object recognition, segmentation, and feature extraction. The process typically involves convolving an image with specific kernels designed to detect intensity gradients.

1. **Roberts Operator**
    - A simple 2x2 kernel pair that detects edges by approximating the gradient in diagonal directions.

Kernels:

- $G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
- $G_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$

The edge magnitude is computed as $\sqrt{G_x^2 + G_y^2}$ (or using `np.hypot` for efficiency).

2. **Prewitt Operator**
    - A 3x3 kernel pair that detects horizontal and vertical edges by emphasizing differences in neighboring pixels.

Kernels:

- $G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$
- $G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

    - Combines edge responses similarly to Roberts.
3. **Sobel Operator**
    - An enhanced 3x3 kernel pair that weights central pixels more heavily, providing smoother edge detection.

Kernels:

- $G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$
- $G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

    - Known for better noise suppression compared to Prewitt.
4. **Laplacian Operator**
    - A single 3x3 kernel that detects edges by computing the second derivative, highlighting areas of rapid intensity change.

Kernel:

- $\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ (or variations like -3 in the center as in the code snippet)

    - Unlike gradient-based methods, it’s isotropic but sensitive to noise.

In this assignment, the image "girl.jpg" is loaded in grayscale, and each operator is applied using the convolve function from scipy.ndimage. The results are visualized using cv2_imshow from Google Colab, showing X-axis edges, Y-axis edges, and the combined edge magnitude.

Code Implementation

Assignment_04_IVP

Conclusion

The implementation successfully applied four edge detection techniques—Roberts, Prewitt, Sobel, and Laplacian—to the image "girl.jpg". Each method highlighted edges differently:

- **Roberts** provided basic diagonal edge detection with minimal computation.
- **Prewitt** improved edge visibility with a larger kernel, capturing horizontal and vertical gradients.
- **Sobel** offered smoother and more pronounced edges due to its weighted kernel, making it effective for noisy images.
- **Laplacian** detected edges isotropically but was more sensitive to noise, as seen in the output.

The program handled errors (e.g., invalid image loading) and normalized outputs for consistent visualization. The results, displayed via cv2_imshow, allowed for a clear comparison of edge detection strengths, fulfilling the objectives of understanding and implementing these techniques in Python.