CS22B1082_Assignment5

February 15, 2025

0.0.1 CS22B1082

0.0.2 Thallapally Nimisha

0.1 Assignment 5

1 Sharpening an Image using the Prewitt Filter

1.0.1 Input:

- **Image f**: The input image (in grayscale, e.g., Lena image).

1.0.2 Output:

- **Image g**: The sharpened version of the input image `f`.

[1]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

1.0.3 Load the image

[2]:
```
image_path = 'images/Lena.png'
image_bgr = cv2.imread(image_path)
f = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2GRAY)
```

`filter2D` function, which performs a 2D corelation operation.

In convolution, the filter (or kernel) is flipped by 180 degrees before applying it to the image. In correlation, the filter is directly applied without flipping.

Therefore, when using `filter2D`, we implicitly flip the kernel by 180 degrees before applying it.

1.0.4 Original Prewitt Filters:

- **Prewitt X Filter (for detecting horizontal edges)**: `python Prewitt_x = np.array([[1, 0, 1], [1, 0, 1], [1, 0, 1]])`

1.0.5 Define Prewitt Filter

[3]:
```
Prewitt_x = np.array([[-1, 0, 1],
                      [-1, 0, 1],
                      [-1, 0, 1]])

Prewitt_y = np.array([[-1, -1, -1],
                      [ 0,  0,  0],
                      [ 1,  1,  1]])
```

1.0.6 Gradient Calculation

[4]:
```
def gradient(image, kernel):
    return cv2.filter2D(image, -1, kernel)
```

[5]:
```
# Calculate the gradients in the x and y directions
grad_x = gradient(f, Prewitt_x)
grad_y = gradient(f, Prewitt_y)

# Compute the magnitude of the gradient
gradient_magnitude = np.sqrt(grad_x**2 + grad_y**2)
# Normalize and threshold the edge image
gradient_magnitude = (gradient_magnitude / gradient_magnitude.max()) * 255
```

1.0.7 Set Threshold

[6]:
```
# Threshold to detect edges
T = 200
e = np.where(gradient_magnitude > T, 1, 0)*255
```

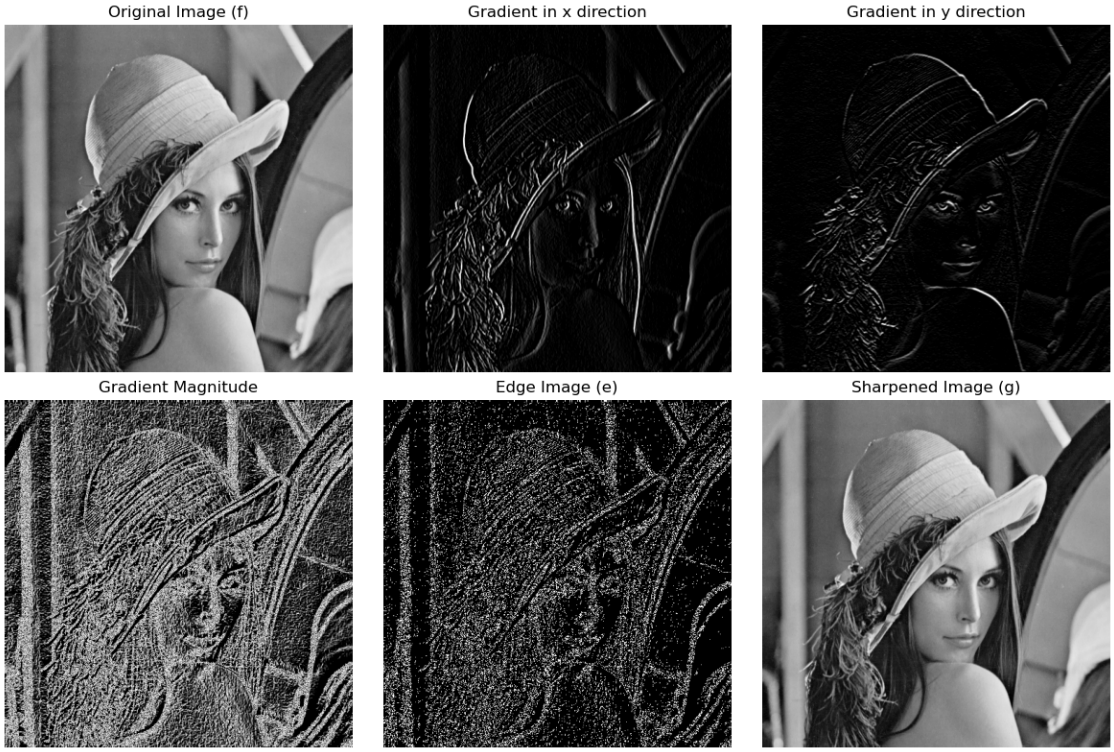
1.0.8 Image Sharpening

[7]:
```
c = 1

# Sharpen the image
g = np.clip(f + c * e.astype(np.uint8), 0, 255)
```

1.0.9 Displaying the results

[8]:
```
plt.figure(figsize=(12, 8))

plt.subplot(2, 3, 1)
plt.imshow(f, cmap='gray')
plt.title("Original Image (f)")
plt.axis('off')

plt.subplot(2, 3, 2)
plt.imshow(grad_x, cmap='gray')
```

```
plt.title("Gradient in x direction")
plt.axis('off')

plt.subplot(2, 3, 3)
plt.imshow(grad_y, cmap='gray')
plt.title("Gradient in y direction")
plt.axis('off')

plt.subplot(2, 3, 4)
plt.imshow(gradient_magnitude, cmap='gray')
plt.title("Gradient Magnitude")
plt.axis('off')

plt.subplot(2, 3, 5)
plt.imshow(e, cmap='gray')
plt.title("Edge Image (e)")
plt.axis('off')

plt.subplot(2, 3, 6)
plt.imshow(g, cmap='gray')
plt.title("Sharpened Image (g)")
plt.axis('off')

plt.tight_layout()
plt.show()
```



[ ]: