EE604 : Homework - 4

Submitted by : Manvi Verma(220631)

Code to implement erosion and dilation operations from scratch for binary images using binary structuring elements

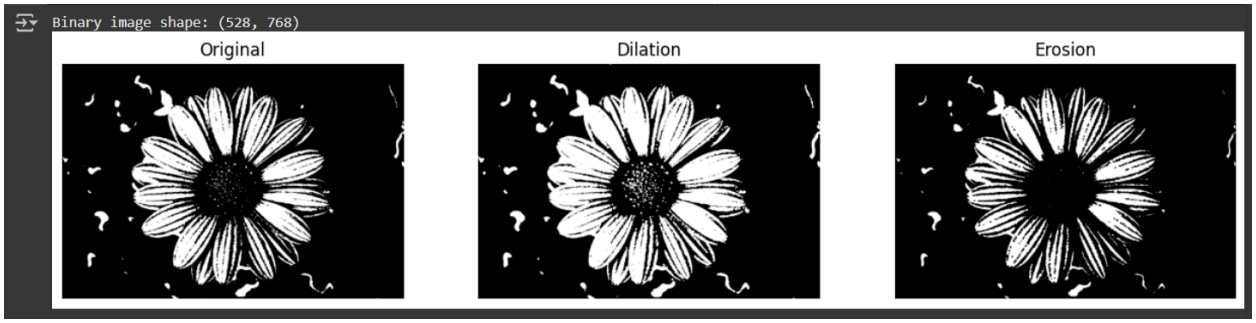
```
[15] import numpy as np
     import matplotlib.pyplot as plt
     import cv2

     # Convert Binary Image to Set
     def image_to_set(img):
         return {(i, j) for i in range(img.shape[0]) for j in range(img.shape[1]) if img[i, j] == 1}

     # Convert Set back to Image
     def set_to_image(foreground_set, shape):
         img = np.zeros(shape, dtype=np.uint8)
         for (i, j) in foreground_set:
             if 0 <= i < shape[0] and 0 <= j < shape[1]:
                 img[i, j] = 1
         return img

     # Structuring Element to Set
     def se_to_set(se):
         center = (se.shape[0] // 2, se.shape[1] // 2)
         return {(i - center[0], j - center[1]) for i in range(se.shape[0]) for j in range(se.shape[1]) if se[i, j] == 1}
```

```
     # Dilation
     def dilate(foreground, se_set):
         result = set()
         for (i, j) in foreground:
             for (a, b) in se_set:
                 result.add((i + a, j + b))
         return result

     # Erosion
     def erode(foreground, se_set):
         result = set()
         for (i, j) in foreground:
             fits = True
             for (a, b) in se_set:
                 if (i + a, j + b) not in foreground:
                     fits = False
                     break
             if fits:
                 result.add((i, j))
         return result
```

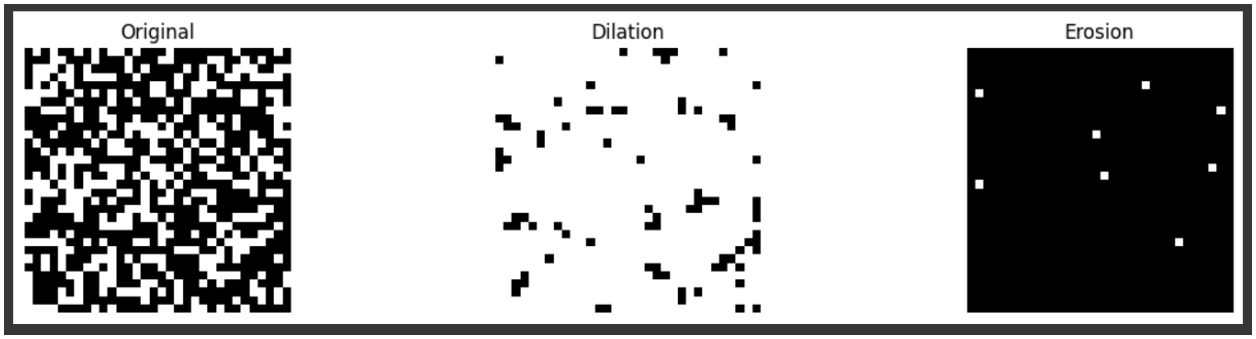
```python
if __name__ == "__main__":

    # 1. Example binary image (random noise)
    # img = (np.random.rand(32, 32) > 0.5).astype(np.uint8)

    # 2. Load grayscale image
    uploaded_img = cv2.imread("Flower_2.png", cv2.IMREAD_GRAYSCALE)

    # Convert to binary (thresholding)
    _,img = cv2.threshold(uploaded_img, 127, 1, cv2.THRESH_BINARY)

    # Now binary_img is a 2D array of 0s and 1s
    print("Binary image shape:", img.shape)

    # Example structuring element (cross)
    se = np.array([[0,1,0],
                   [1,1,1],
                   [0,1,0]], dtype=np.uint8)

    # Convert to sets
    img_set = image_to_set(img)
    se_set = se_to_set(se)

    # Perform operations
    dilated_set = dilate(img_set, se_set)
    eroded_set = erode(img_set, se_set)

    # Convert back to images for visualization
    D = set_to_image(dilated_set, img.shape)
    E = set_to_image(eroded_set, img.shape)
```

```python
    # Visualization
    fig, axes = plt.subplots(1, 3, figsize=(15, 3))
    axes[0].imshow(img, cmap="gray"); axes[0].set_title("Original")
    axes[1].imshow(D, cmap="gray"); axes[1].set_title("Dilation")
    axes[2].imshow(E, cmap="gray"); axes[2].set_title("Erosion")
    for ax in axes:
        ax.axis("off")
    plt.show()
```

(1) Results when a grayscale or binary image is explicitly loaded



(2) Results when Random Noise is taken as Example Binary Image


Original
Dilation
Erosion