Lab Assignment: Morphological Filtering

1. **Implementing Dilation and Erosion:**

- Write code to perform dilation and erosion operations on a noisy grayscale image using Python.
- Experiment with different structuring elements (e.g., square, disk, diamond) and compare how they affect the output of dilation and erosion.

2. **Opening Operation:**

- Implement the opening operation using OpenCV and logic.
- Apply the opening operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.
- Compare the results obtained with different structuring elements.

3. **Closing Operation:**

- Implement the closing operation using OpenCV and logic.
- Apply the closing operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.
- Compare the results obtained with different structuring elements.

4. **Morphological Gradient:**

- Implement the morphological gradient operation using OpenCV and logic.
- Apply the morphological gradient operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.
- Visualize and analyze the differences in results obtained with different structuring elements.

5. **Top Hat Operation:**

- Implement the top hat operation using OpenCV and logic.
- Apply the top hat operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.
- Examine the output images and discuss the effects of different structuring elements on the results.

6. **Black Hat Operation:**

- Implement the black hat operation using OpenCV and logic.
- Apply the black hat operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.

- Compare the results obtained with different structuring elements and discuss the variations observed.

Note:

- Experiment with different sizes and shapes of structuring elements to observe their effects on the output images.
- Provide explanations and interpretations of the results in the report.

Assignment 9

Morphological Filtering

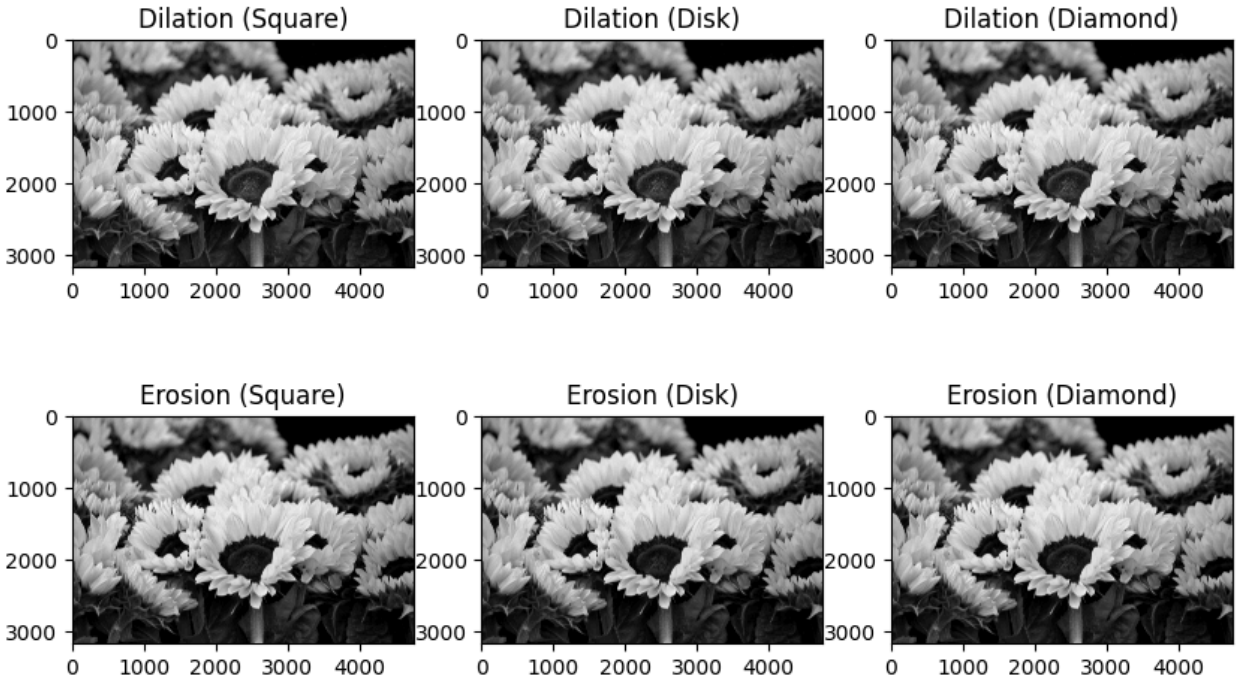
1. Implementing Dilation and Erosion:

● Write code to perform dilation and erosion operations on a noisy grayscale image using Python.
● Experiment with different structuring elements (e.g., square, disk, diamond) and compare how they affect the output of dilation and erosion.

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
# Load a noisy grayscale image (you can use your own image)
image = cv2.imread('image.jpg', cv2.IMREAD_GRAYSCALE)
# Structuring elements
square_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (5, 5))
disk_kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5, 5))
diamond_kernel = cv2.getStructuringElement(cv2.MORPH_CROSS, (5, 5))
# Dilation
dilated_square = cv2.dilate(image, square_kernel, iterations=1)
dilated_disk = cv2.dilate(image, disk_kernel, iterations=1)
dilated_diamond = cv2.dilate(image, diamond_kernel, iterations=1)
# Erosion
eroded_square = cv2.erode(image, square_kernel, iterations=1)
eroded_disk = cv2.erode(image, disk_kernel, iterations=1)
eroded_diamond = cv2.erode(image, diamond_kernel, iterations=1)
# Display results
plt.figure(figsize=(10, 10))
plt.subplot(2, 3, 1)
plt.title('Dilation (Square)')
plt.imshow(dilated_square, cmap='gray')
plt.subplot(2, 3, 2)
plt.title('Dilation (Disk)')
plt.imshow(dilated_disk, cmap='gray')
plt.subplot(2, 3, 3)
plt.title('Dilation (Diamond)')
plt.imshow(dilated_diamond, cmap='gray')
plt.subplot(2, 3, 4)
plt.title('Erosion (Square)')
plt.imshow(eroded_square, cmap='gray')
plt.subplot(2, 3, 5)
plt.title('Erosion (Disk)')
plt.imshow(eroded_disk, cmap='gray')
plt.subplot(2, 3, 6)
```

```
plt.title('Erosion (Diamond)')
plt.imshow(eroded_diamond, cmap='gray')
plt.show()
```

Output:

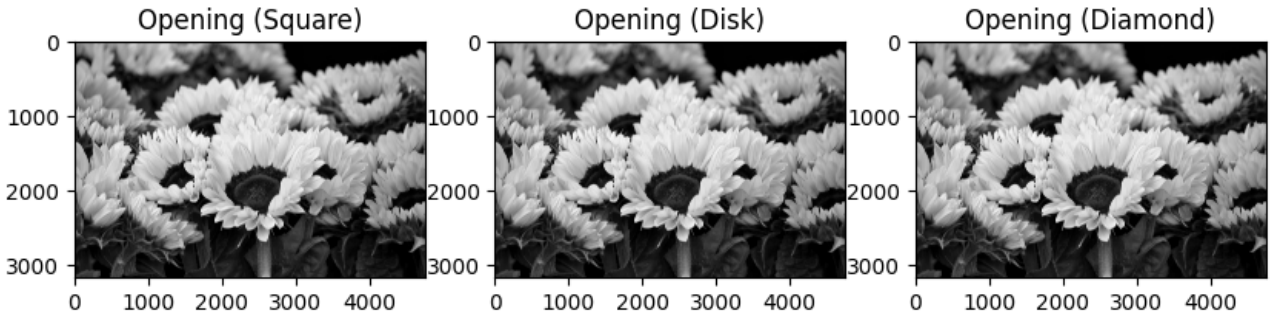


2. Opening Operation:

● Implement the opening operation using OpenCV and logic.
● Apply the opening operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.
● Compare the results obtained with different structuring elements.

```
# Opening is erosion followed by dilation
opened_square = cv2.morphologyEx(image, cv2.MORPH_OPEN, square_kernel)
opened_disk = cv2.morphologyEx(image, cv2.MORPH_OPEN, disk_kernel)
opened_diamond = cv2.morphologyEx(image, cv2.MORPH_OPEN, diamond_kernel)
# Display results
plt.figure(figsize=(10, 10))
plt.subplot(1, 3, 1)
plt.title('Opening (Square)')
plt.imshow(opened_square, cmap='gray')
plt.subplot(1, 3, 2)
plt.title('Opening (Disk)')
plt.imshow(opened_disk, cmap='gray')
plt.subplot(1, 3, 3)
plt.title('Opening (Diamond)')
plt.imshow(opened_diamond, cmap='gray')
plt.show()
```

Output:

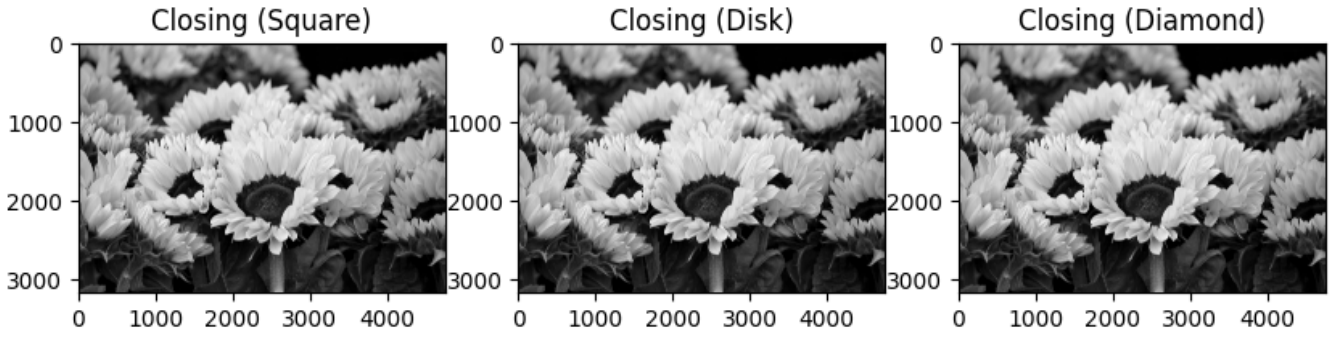


3. Closing Operation:

● Implement the closing operation using OpenCV and logic.
● Apply the closing operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.
● Compare the results obtained with different structuring elements.

```
# Closing is dilation followed by erosion
closed_square = cv2.morphologyEx(image, cv2.MORPH_CLOSE, square_kernel)
closed_disk = cv2.morphologyEx(image, cv2.MORPH_CLOSE, disk_kernel)
closed_diamond = cv2.morphologyEx(image, cv2.MORPH_CLOSE, diamond_kernel)
# Display results
plt.figure(figsize=(10, 10))
plt.subplot(1, 3, 1)
plt.title('Closing (Square)')
plt.imshow(closed_square, cmap='gray')
plt.subplot(1, 3, 2)
plt.title('Closing (Disk)')
plt.imshow(closed_disk, cmap='gray')
plt.subplot(1, 3, 3)
plt.title('Closing (Diamond)')
plt.imshow(closed_diamond, cmap='gray')
plt.show()
```

Output:

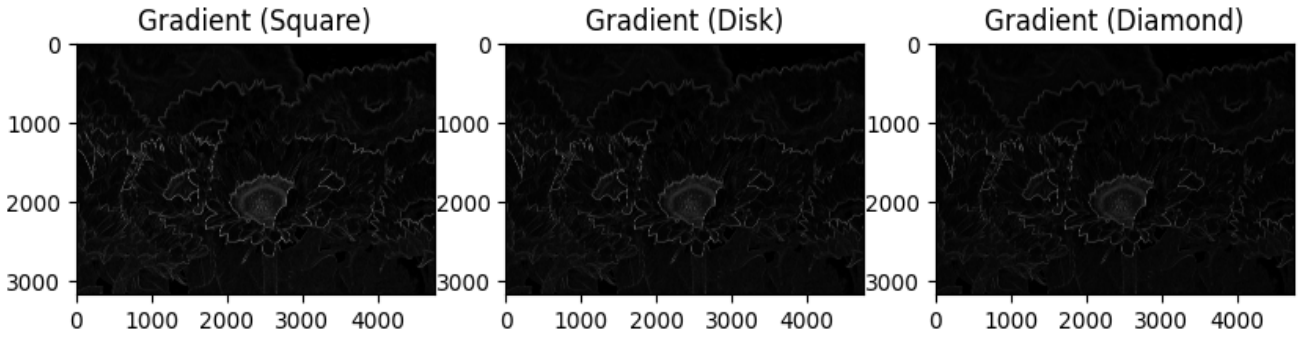


4. Morphological Gradient:

● Implement the morphological gradient operation using OpenCV and logic.
● Apply the morphological gradient operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.
● Visualize and analyze the differences in results obtained with different structuring elements.

```
# Morphological gradient = dilation - erosion
gradient_square = cv2.morphologyEx(image, cv2.MORPH_GRADIENT,
square_kernel)
gradient_disk = cv2.morphologyEx(image, cv2.MORPH_GRADIENT, disk_kernel)
gradient_diamond = cv2.morphologyEx(image, cv2.MORPH_GRADIENT,
diamond_kernel)
# Display results
plt.figure(figsize=(10, 10))
plt.subplot(1, 3, 1)
plt.title('Gradient (Square)')
plt.imshow(gradient_square, cmap='gray')
plt.subplot(1, 3, 2)
plt.title('Gradient (Disk)')
plt.imshow(gradient_disk, cmap='gray')
plt.subplot(1, 3, 3)
plt.title('Gradient (Diamond)')
plt.imshow(gradient_diamond, cmap='gray')
plt.show()
```

Output:



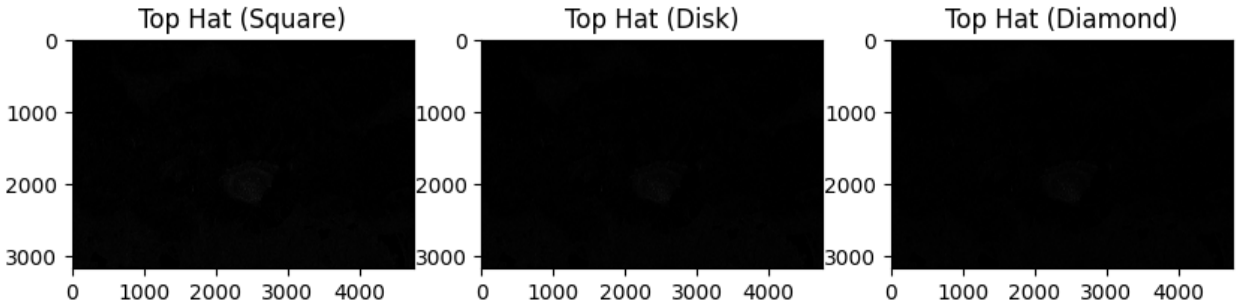
5. Top Hat Operation:

● Implement the top hat operation using OpenCV and logic.
● Apply the top hat operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.
● Examine the output images and discuss the effects of different structuring elements on the results.

```
# Top Hat = original image - opening
tophat_square = cv2.morphologyEx(image, cv2.MORPH_TOPHAT, square_kernel)
tophat_disk = cv2.morphologyEx(image, cv2.MORPH_TOPHAT, disk_kernel)
tophat_diamond = cv2.morphologyEx(image, cv2.MORPH_TOPHAT, diamond_kernel)

# Display results
plt.figure(figsize=(10, 10))
plt.subplot(1, 3, 1)
plt.title('Top Hat (Square)')
plt.imshow(tophat_square, cmap='gray')
plt.subplot(1, 3, 2)
plt.title('Top Hat (Disk)')
plt.imshow(tophat_disk, cmap='gray')
plt.subplot(1, 3, 3)
plt.title('Top Hat (Diamond)')
plt.imshow(tophat_diamond, cmap='gray')
plt.show()
```

Output:



6. Black Hat Operation:

● Implement the black hat operation using OpenCV and logic.
● Apply the black hat operation on a grayscale image using different structuring elements such as rectangles, circles, and crosses.
● Compare the results obtained with different structuring elements and discuss the variations observed.

```
# Black Hat = closing - original image
blackhat_square = cv2.morphologyEx(image, cv2.MORPH_BLACKHAT,
square_kernel)
blackhat_disk = cv2.morphologyEx(image, cv2.MORPH_BLACKHAT, disk_kernel)
blackhat_diamond = cv2.morphologyEx(image, cv2.MORPH_BLACKHAT,
diamond_kernel)
```

```
# Display results
plt.figure(figsize=(10, 10))
plt.subplot(1, 3, 1)
plt.title('Black Hat (Square)')
plt.imshow(blackhat_square, cmap='gray')

plt.subplot(1, 3, 2)
plt.title('Black Hat (Disk)')
plt.imshow(blackhat_disk, cmap='gray')
plt.subplot(1, 3, 3)
plt.title('Black Hat (Diamond)')
plt.imshow(blackhat_diamond, cmap='gray')
plt.show()
```

Output: