Department of Computer Science and Engineering (Data Science)
Image Processing and Computer Vision I (DJ19DSL603)

Experiment 7

NAME: Aaditya Malani **SAP ID:60009220192** **BATCH:D1-1**

Aim: To perform morphological operations on image (erosion, dilation, opening, closing and hit and miss transform)

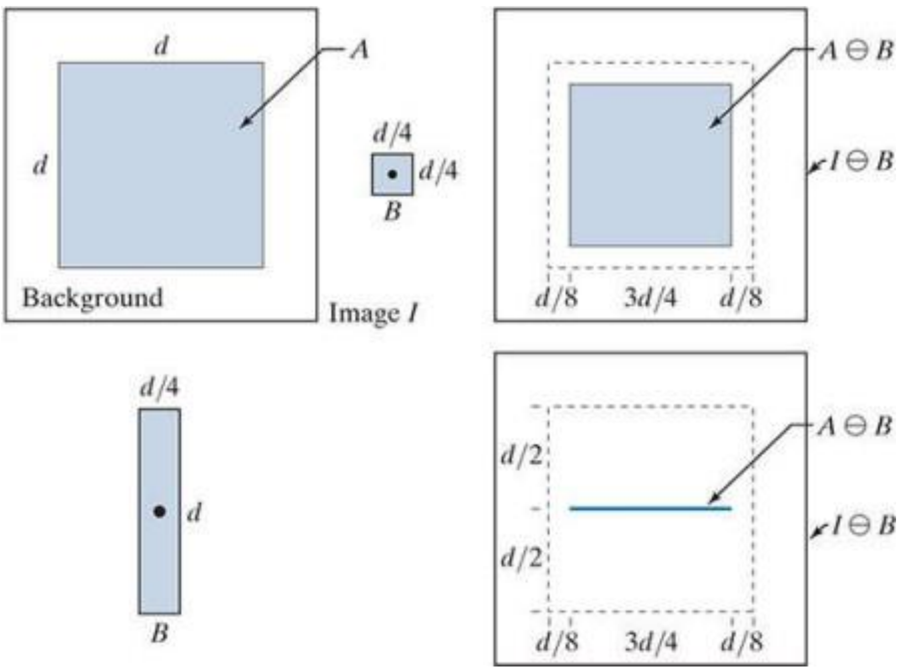
Theory:

1. **Erosion:**

Morphological expressions are written in terms of structuring elements and a set, A, of foreground pixels, or in terms of structuring elements and an image, I, that contains A. We consider the former approach first. Erosion of A by B is defined as:

$$A \ominus B = \{z \mid (B)_z \subseteq A\}$$

In words, this equation indicates that the erosion of A by B is the set of all points z such that B, translated by z, is contained in A. (Remember, displacement is defined with respect to the origin of B. The result of erosion is controlled by the shape of the structuring element. The image is eroded by two different structuring elements (B) giving the outputs as seen:

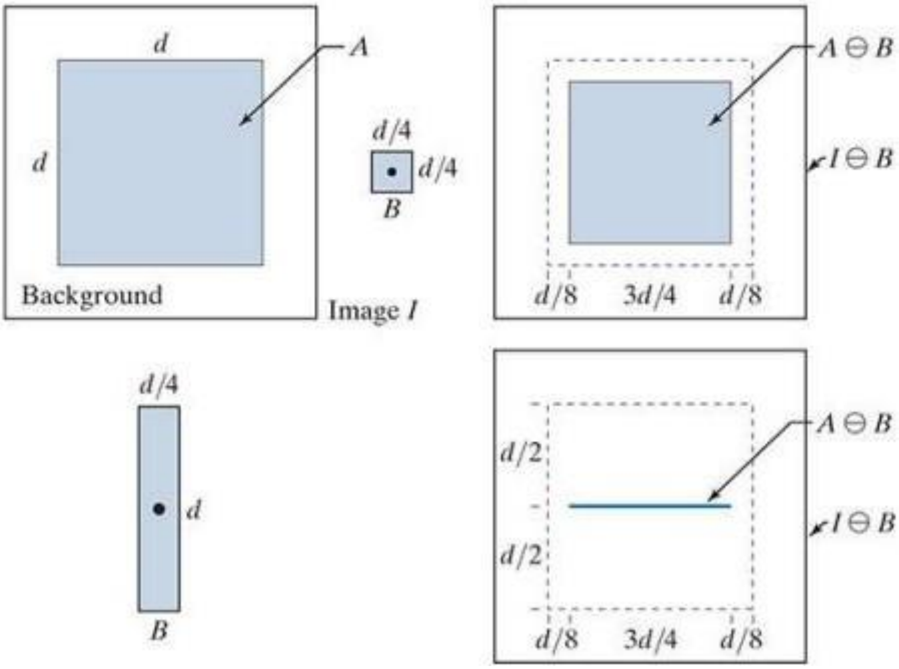


2. **Dilation:**

Dilation of A by B is defined as:

$$A \oplus B = \left\{ z \,\middle|\, [(\hat{B})_z \cap A] \subseteq A \right\}$$

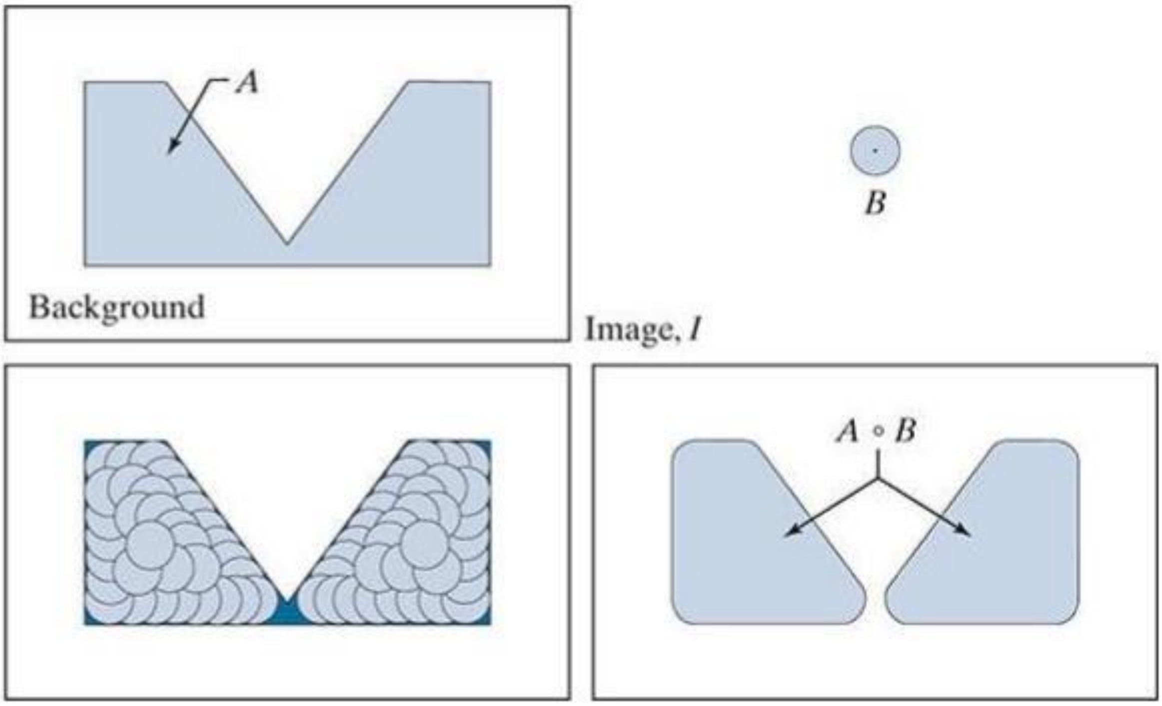
The dilation of A by B then is the set of all displacements, z, such that the foreground elements of overlap at least one element of A. (Remember, z is the displacement of the origin of $\hat{B}$. The image is dilated by two different structuring elements (B) giving the outputs as seen:



3. **Opening:**
   Opening generally smoothens the contour of an object, breaks narrow isthmuses, and eliminates thin protrusions. The opening of set A by structuring element B is defined as:

   $$A \circ B = (A \ominus B) \oplus B$$

   Thus, the opening A by B is the erosion of A by B, followed by a dilation of the result by B. The opening of A by B is the union of all the translations of B so that B fits entirely in A

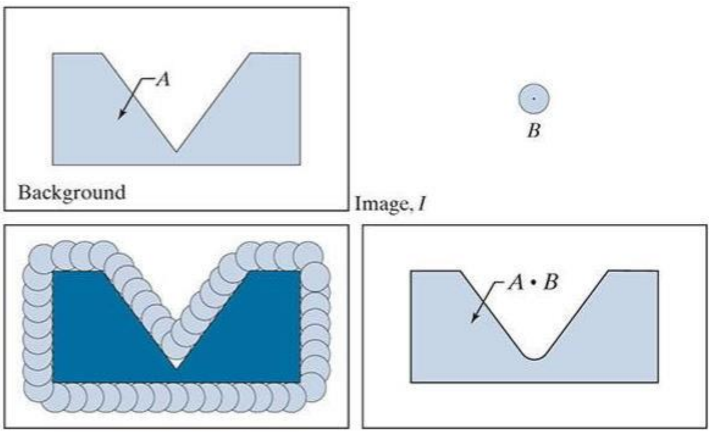


4. **Closing:**

   The closing of set A by structuring element B is defined as:

   $A \bullet B = (A \oplus B) \ominus B$

   The closing of A by B is simply the dilation of A by B, followed by erosion of the result by B. The opening of A by B is the union of all the translations of B so that B fits entirely in A.



5. **Hit and miss transform:**

   The morphological hit-or-miss transform (HMT) is a basic tool for shape detection. Let I be a binary image composed of foreground (A) and background pixels, respectively. Unlike the morphological methods discussed thus far, the HMT utilizes two structuring elements: B_1 for detecting shapes in the foreground, and B_2 for detecting shapes in the background. The HMT of image I is defined as

$$I \circledast B_{1,2} = \{z \mid (B_1)_z \subseteq A \text{ and } (B_2)_z \subseteq A^c\}$$
$$= (A \ominus B_1) \cap (A^c \ominus B_2)$$

In words, this equation says that the morphological HMT is the set of translations, z, of structuring elements B_1 and B_2 such that, simultaneously, B_1 found a match in the foreground (i.e., is contained in A) and B_2 found a match in the background (i.e., is contained in A^c).

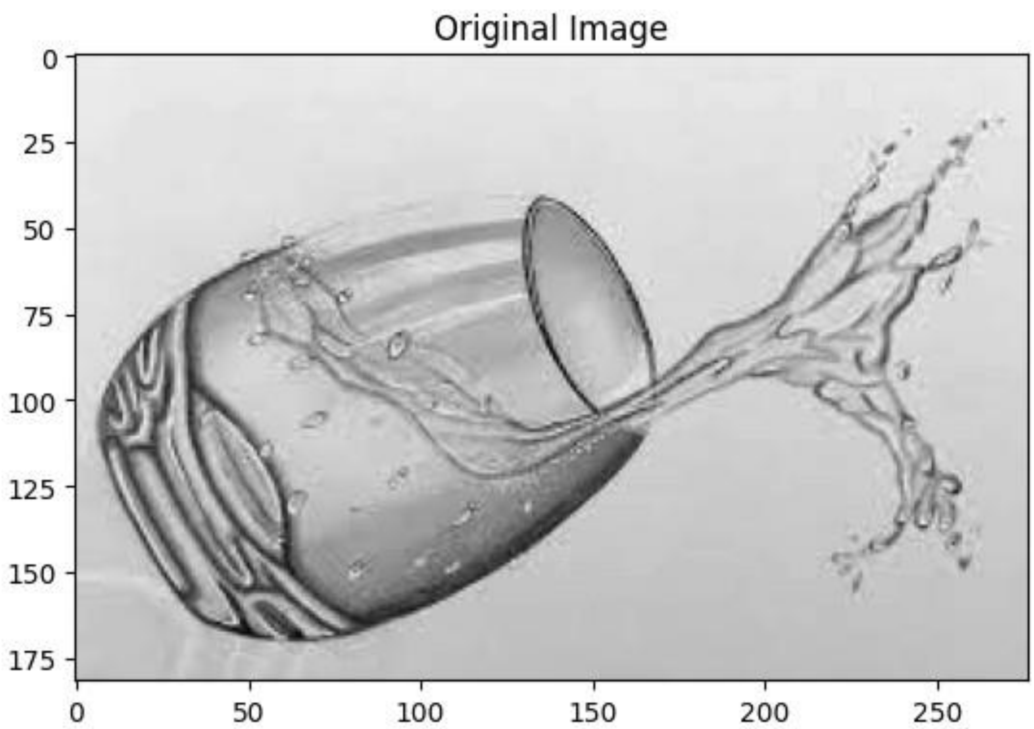
Lab Assignments to complete in this session

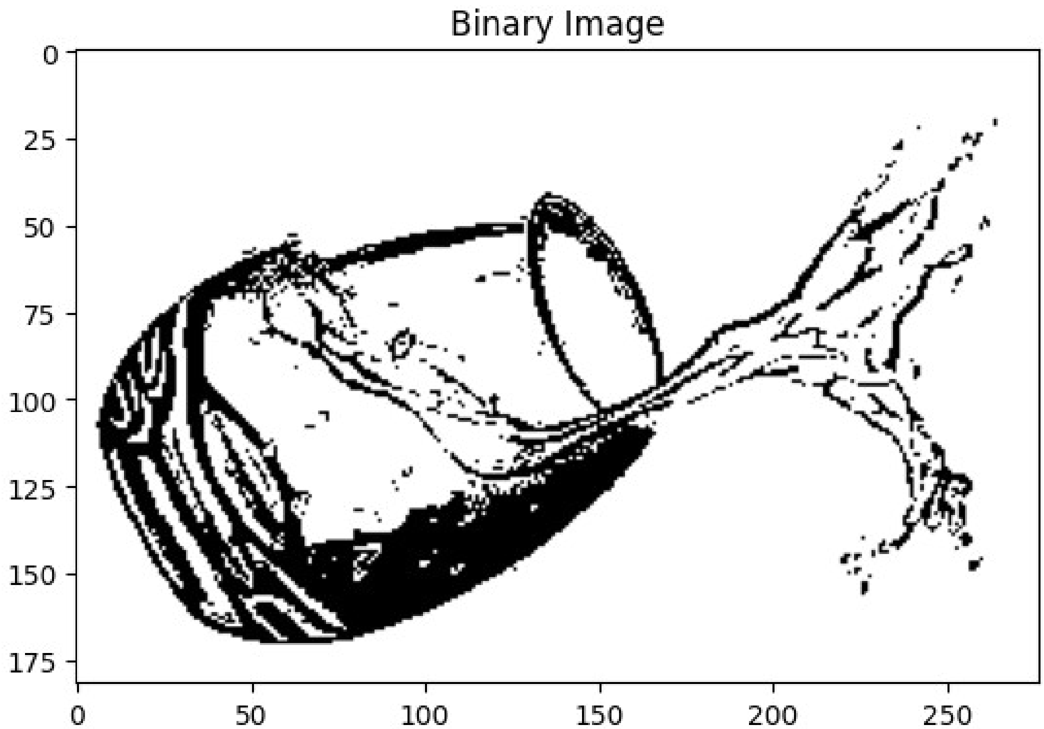
Problem Statement: Develop a Python program utilizing the OpenCV library to morph the images in spatial domain using the morphological operations explained. The program should address the following tasks:

1. Read any image from COVID 19 Image Dataset. Take it as object **Dataset Link:** **[Covid-19 Image Dataset]**
2. Create a structuring element of any form that is much smaller in size than the image considered in step 1.
3. Display the before & after image(s) used in every task below:
   a. Apply erosion and show the before and after image
   b. Apply dilation and show the before and after image
   c. Apply opening and show the before and after image
   d. Apply closing and show the before and after image
   e. Apply hit and miss transformation and show the before and after image. Take the structuring element to be negation of the one used till now.

The solution to the operations performed must be produced by scratch coding without the use of built in OpenCV methods.

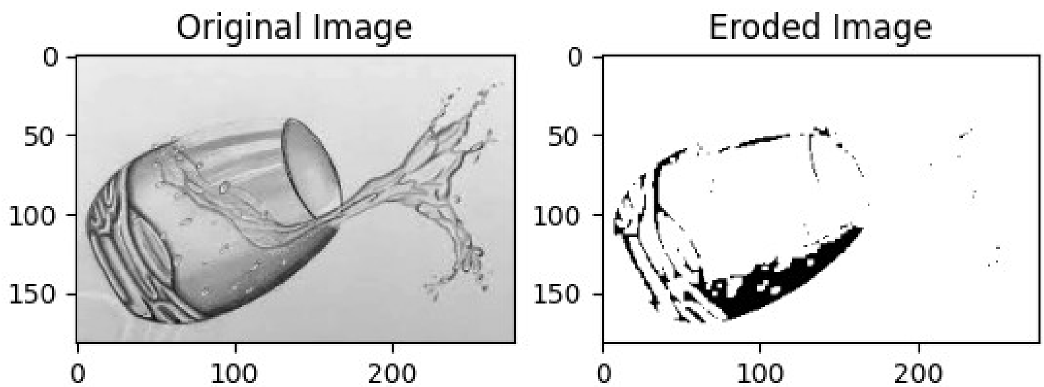
```
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image


image_path = "/content/pencil_drawing.jpeg"
image = Image.open(image_path).convert('L')
image = np.array(image)


plt.imshow(image, cmap='gray')
plt.title("Original Image")
plt.show()


threshold = 128
binary_image = np.where(image > threshold, 1, 0)

plt.imshow(binary_image, cmap='gray')
plt.title("Binary Image")
plt.show()
```



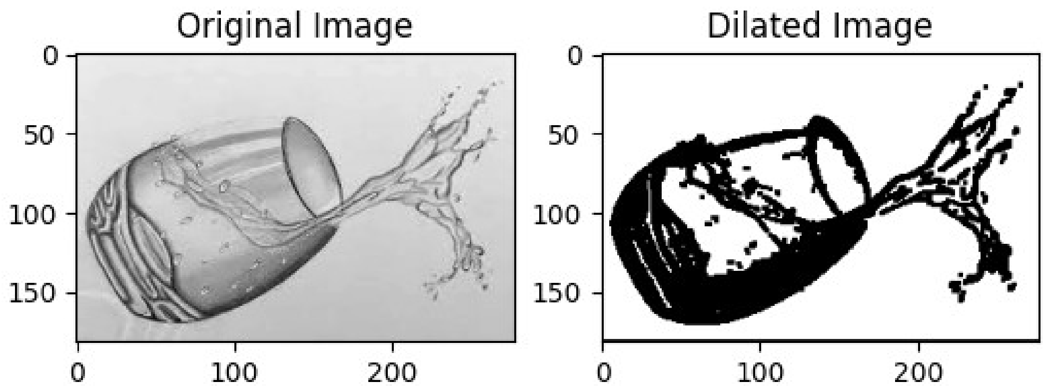



```
def pad_image(image, kernel_size):
    pad = kernel_size // 2
    return np.pad(image, pad, mode='constant', constant_values=0)

def apply_kernel(image, kernel, operation):
    kernel_size = kernel.shape[0]
    padded_image = pad_image(image, kernel_size)
    result_image = np.zeros_like(image)

    for i in range(image.shape[0]):
        for j in range(image.shape[1]):

            region = padded_image[i:i+kernel_size, j:j+kernel_size]

            if operation == 'dilation':
                result_image[i, j] = 1 if np.array_equal(region &
kernel, kernel) else 0
            elif operation == 'erosion':
                result_image[i, j] = 1 if np.any(region & kernel) else
0

    return result_image


kernel = np.ones((3, 3), dtype=np.uint8)
```

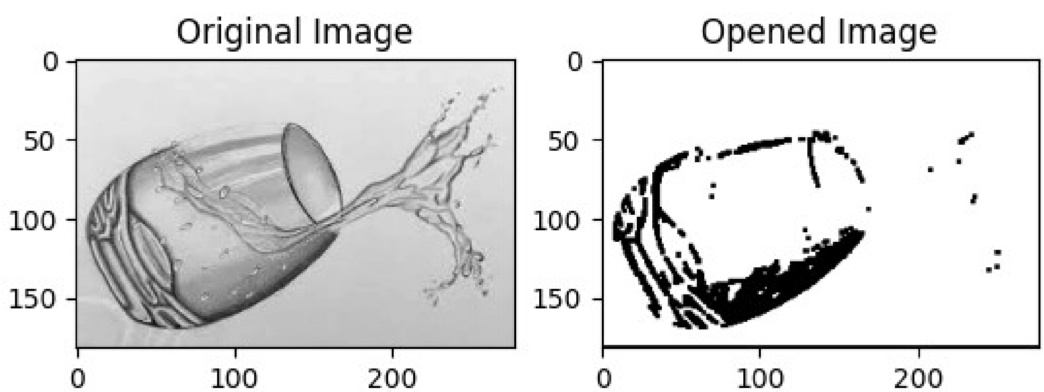
```
eroded_image = apply_kernel(binary_image, kernel, 'erosion')



plt.subplot(1, 2, 1)
plt.imshow(image, cmap='gray')
plt.title("Original Image")

plt.subplot(1, 2, 2)
plt.imshow(eroded_image, cmap='gray')
plt.title("Eroded Image")
plt.show()
```




```

dilated_image = apply_kernel(binary_image, kernel, 'dilation')


plt.subplot(1, 2, 1)
plt.imshow(image, cmap='gray')
plt.title("Original Image")

plt.subplot(1, 2, 2)
plt.imshow(dilated_image, cmap='gray')
plt.title("Dilated Image")
plt.show()
```

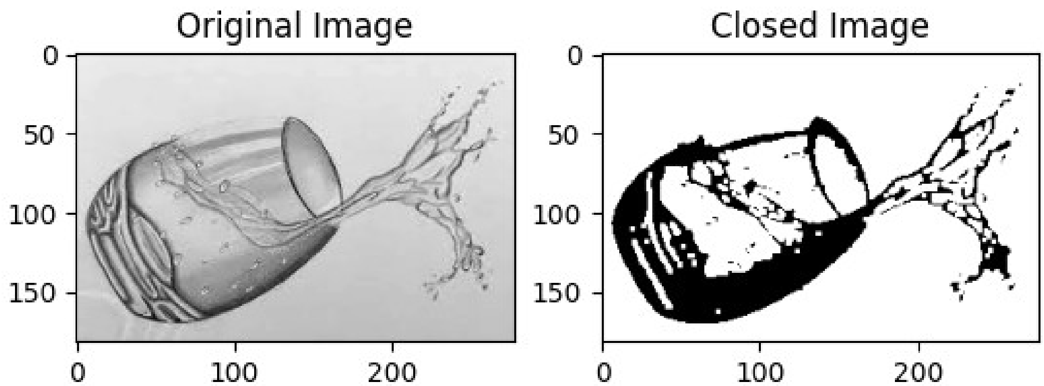



```
plt.subplot(1, 2, 1)
plt.imshow(image, cmap='gray')
plt.title("Original Image")

opened_image = apply_kernel(eroded_image, kernel, 'dilation')


plt.subplot(1, 2, 2)
plt.imshow(opened_image, cmap='gray')
plt.title("Opened Image")
plt.show()
```

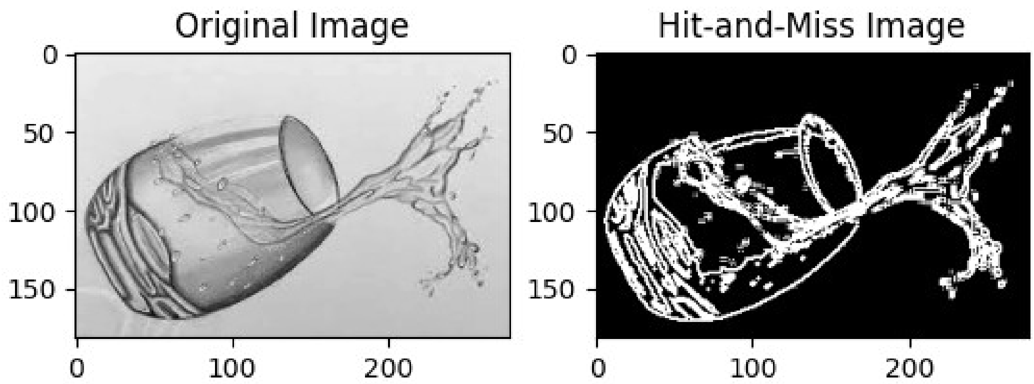



```
closed_image = apply_kernel(dilated_image, kernel, 'erosion')


plt.subplot(1, 2, 1)
plt.imshow(image, cmap='gray')
plt.title("Original Image")
```

```
plt.subplot(1, 2, 2)
plt.imshow(closed_image, cmap='gray')
plt.title("Closed Image")
plt.show()
```




```
def hit_and_miss(image, kernel_fg, kernel_bg):
    eroded_fg = apply_kernel(image, kernel_fg, 'erosion')
    eroded_bg = apply_kernel(1 - image, kernel_bg, 'erosion')
    return eroded_fg & eroded_bg


kernel_fg = np.array([[0, 1, 0], [1, 1, 1], [0, 1, 0]],
dtype=np.uint8)
kernel_bg = np.array([[1, 0, 1], [0, 0, 0], [1, 0, 1]],
dtype=np.uint8)


hit_miss_image = hit_and_miss(binary_image, kernel_fg, kernel_bg)


plt.subplot(1, 2, 1)
plt.imshow(image, cmap='gray')
plt.title("Original Image")

plt.subplot(1, 2, 2)
plt.imshow(hit_miss_image, cmap='gray')
plt.title("Hit-and-Miss Image")
plt.show()
```



Conclusion:

In this experiment, we manually implemented the **morphological operations** using NumPy arrays instead of relying on OpenCV's built-in methods. Each operation:

- **Erosion** reduced white areas by shrinking object boundaries.
- **Dilation** expanded white areas by growing object boundaries.
- **Opening** removed small white noise from the foreground.
- **Closing** filled small black holes in the foreground.
- **Hit-and-Miss** transformation detected specific patterns by combining erosion on the foreground and the complement of the background.

Applications:

- **Image Preprocessing**: These operations are used in image preprocessing tasks to prepare images for object detection and edge detection.
- **Medical Imaging**: Morphological operations are used to analyze structures in medical images like MRI scans.
- **Object Detection**: Operations like hit-and-miss are used for shape recognition in object detection systems.
- **Document Image Analysis**: These techniques are used to remove noise, highlight text, and segment document images.