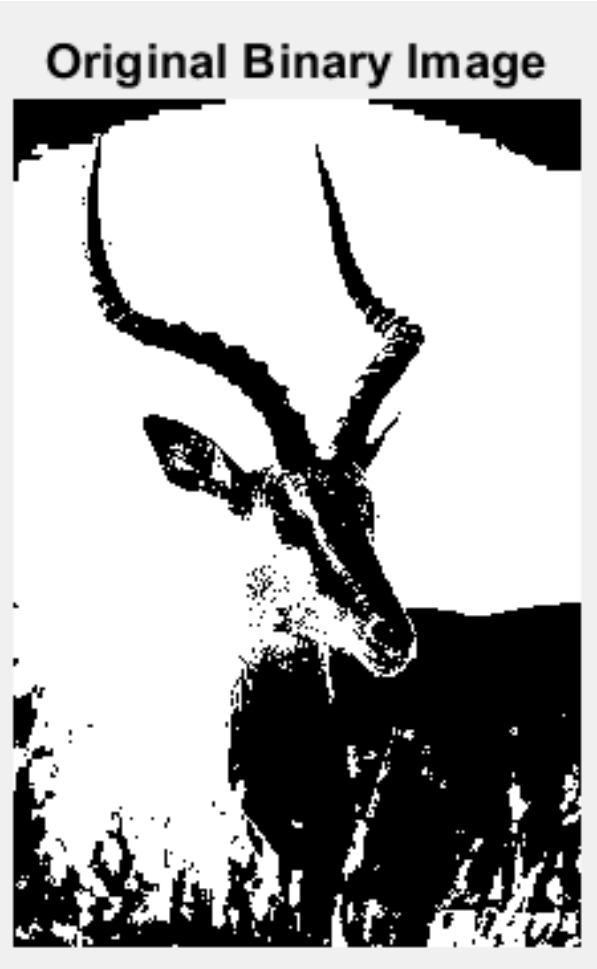
1)

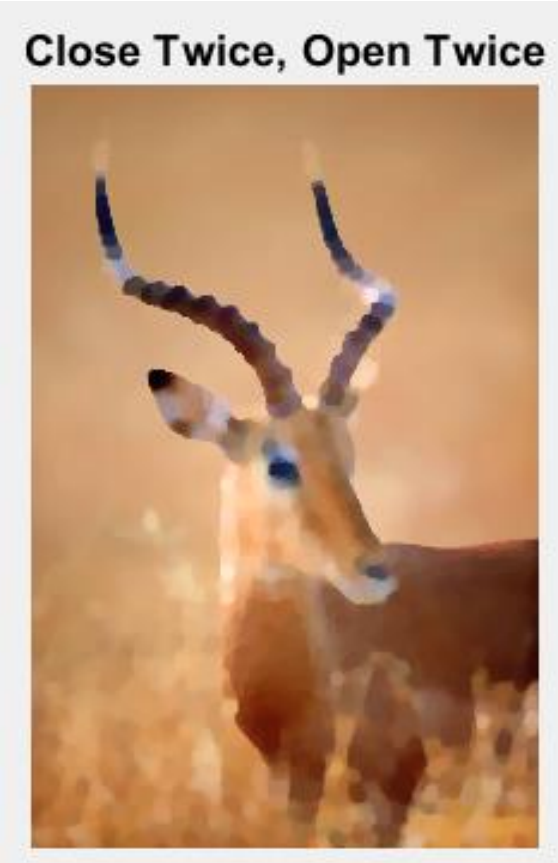


% Load a grayscale image (or convert a color image to grayscale)

```
originalImage = imread('imG.jpg');
grayImage = rgb2gray(originalImage);

% Convert the grayscale image to binary using a mean-intensity threshold
threshold = mean(grayImage(:)); % You can adjust this threshold
binaryImage = grayImage > threshold;

% Apply erosion twice and then dilation twice with a structuring element
se = strel('square', 3); % You can choose a different structuring element
erodedImage1 = imerode(binaryImage, se);
erodedImage2 = imerode(erodedImage1, se);
dilatedImage1 = imdilate(erodedImage2, se);
dilatedImage2 = imdilate(dilatedImage1, se);

% Starting with the original binary image, dilate twice and erode twice
dilatedImage3 = imdilate(binaryImage, se);
dilatedImage4 = imdilate(dilatedImage3, se);
erodedImage3 = imerode(dilatedImage4, se);
erodedImage4 = imerode(erodedImage3, se);

% Display the results
subplot(2, 2, 1); imshow(binaryImage); title('Original Binary Image');
subplot(2, 2, 2); imshow(dilatedImage2); title('Dilate Twice, Erode Twice');
subplot(2, 2, 3); imshow(erodedImage4); title('Erode Twice, Dilate Twice');
```

Original Binary Image

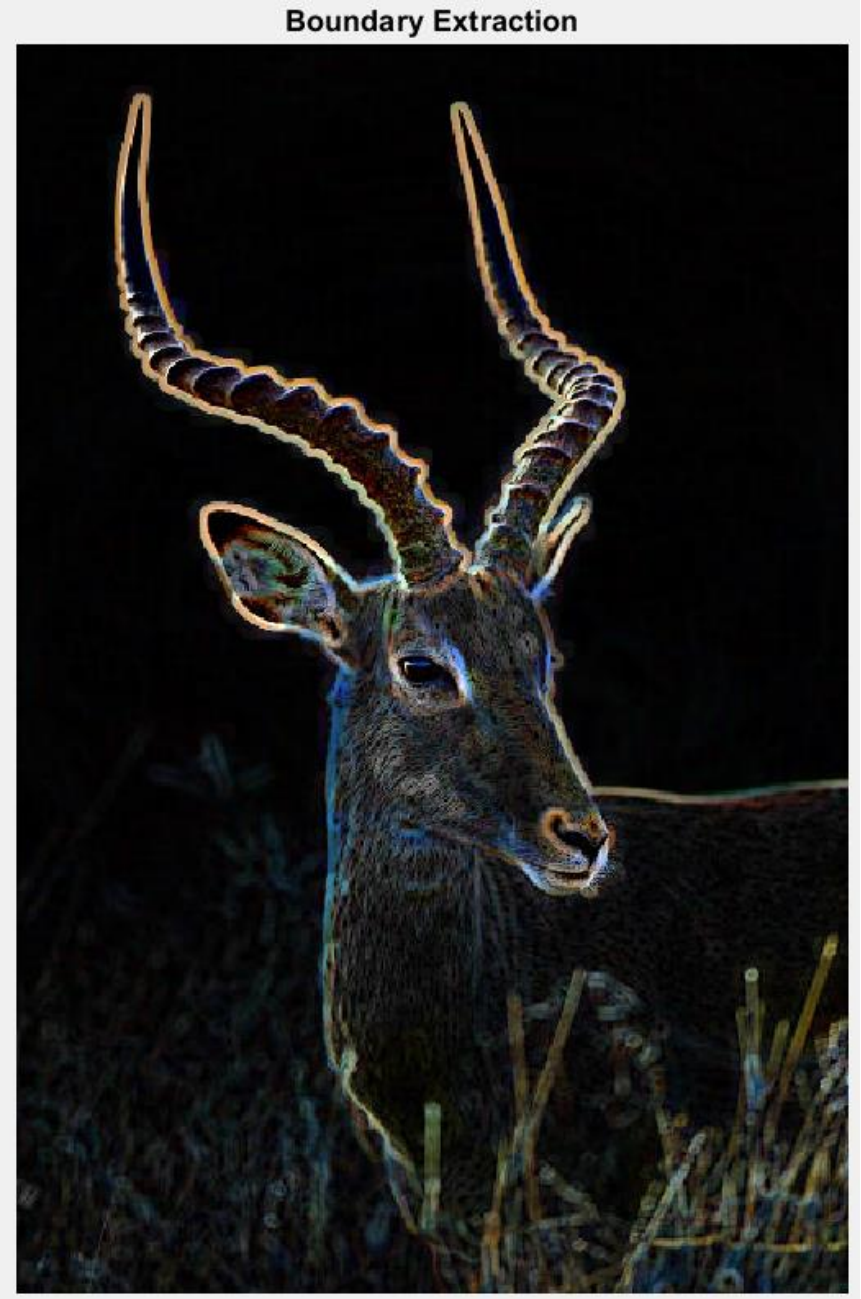
Open Twice, Close Twice



1. "Closed Twice, Opened Twice"
    - This sequence may appear to emphasize object separation and reduce noise.
    - Objects in the image is more distinctly separated from each other.
    - Small gaps or irregularities within objects are often minimized.
    - The result can look more structured and less noisy.
2. "Opened Twice, Closed Twice"
    - This sequence may emphasize preserving finer details and creating smoother boundaries.
    - Smaller objects and fine details might be better retained.
    - Object boundaries tend to be smoother and continuous.
    - The image appear cleaner and smoother.

```
binaryImage = imread('imG.jpg');

% Apply opening twice and then closing twice with a structuring element
se = strel('disk', 5); % You can choose a different structuring element
openedOnce = imopen(binaryImage, se);
openedTwice = imopen(openedOnce, se);
closedOnce = imclose(openedTwice, se);
closedTwice = imclose(closedOnce, se);

% Show the resulting images
subplot(2, 2, 1); imshow(binaryImage); title('Original Binary Image');
subplot(2, 2, 2); imshow(openedTwice); title('Opening Twice');
subplot(2, 2, 3); imshow(closedTwice); title('Closing Twice');



% Starting with the original binary image, close twice and open twice
closedOnce = imclose(binaryImage, se);
closedTwice = imclose(closedOnce, se);
openedOnce = imopen(closedTwice, se);
openedTwice = imopen(openedOnce, se);

% Show the resulting images
subplot(2, 2, 1); imshow(binaryImage); title('Original Binary Image');
subplot(2, 2, 2); imshow(closedTwice); title('Closing Twice');
subplot(2, 2, 3); imshow(openedTwice); title('Opening Twice');
```

Problem 3: Boundary Extraction



- Boundary extraction is ideal for morphology analysis, while Canny edges are better for object recognition.
- Boundary extraction with erosion is less sensitive to noise, helping to smooth small irregularities. Canny edge detection can be sensitive to noise, potentially leading to false positives and noise-induced artifacts.
- Boundary extraction tends to produce smoother, cleaner boundaries, while Canny edges are more sensitive and capture finer details, even noise.

```
binaryImage = imread('problem1_binary_image.png'); % Replace with the actual filename

% Define a structuring element B for boundary extraction
se = strel('disk', 5); % You can choose a different structuring element

% Perform boundary extraction
erodedImage = imerode(binaryImage, se);
boundaryImage = binaryImage - erodedImage;

% Load the original color image
originalImage = imread('imG.jpg'); % Replace with the actual filename

% Convert the color image to grayscale
grayImage = rgb2gray(originalImage);

% Apply Canny edge detection to the grayscale image
cannyEdges = edge(grayImage, 'Canny');

% Show the boundary image and Canny edges
subplot(1, 2, 1); imshow(boundaryImage); title('Boundary Extraction');
subplot(1, 2, 2); imshow(cannyEdges); title('Canny Edges');
```