ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)

ORGANISATION OF ISLAMIC COOPERATION (OIC)

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

Project Title: Kidney Stone Detection Using Image Processing in MATLAB

Group Number: B1_Group 4

Team Members:

- Mutakabbir Ashfak

  ID: 200021227
- Natasha Hasan

  ID: 200021229
- Muhtasim Zunaid Islam

  ID: 200021237
- Khandoker Rafid Mahmud

  ID: 200021239
- Mubarak Ibrahim

  ID: 200021259

Project Problem Statement

Detecting kidney stones in medical images is essential for early diagnosis and treatment. This project aims to develop a MATLAB-based system that identifies kidney stones by processing medical images. The system allows users to manually select a Region of Interest (ROI), after which it enhances image visibility, removes noise, and isolates potential stone regions. If a kidney stone is detected, it is highlighted on the original image, providing a clear and easy-to-interpret visual result.

Solution Abstract

This project presents an automated kidney stone detection system using MATLAB. Users can upload a medical image and manually select a Region of Interest (ROI). The system then applies grayscale conversion, binarization, morphological filtering, and region-based segmentation to enhance the image and detect kidney stones. If detected, the stones are highlighted on the original image for better visualization. This system offers a straightforward and efficient way to analyze medical images and identify potential kidney stones.

Detailed Methodology

The following steps outline the methodology used in this project:

1. **Image Acquisition:**
    - The user selects a medical image using a file selection dialog.
2. **ROI Selection:**
    - The user manually selects the kidney region in the image.
3. **Preprocessing:**
    - Convert the image to grayscale.
    - Apply binarization to segment potential kidney stone regions.
4. **Image Processing:**
    - Enhance contrast using imadjust.
    - Reduce noise using median filtering (medfilt2).
    - Apply morphological operations (erosion and dilation) to refine segmentation.
5. **Stone Detection:**
    - Perform connected component analysis using bwlabel.
    - Determine bounding boxes and centroids of detected kidney stones.
    - Highlight detected stones using circles on the original image.
6. **Result Display:**

- Display the original, grayscale, binarized, and processed images.
- If kidney stones are detected, mark them on the original image.
- Show a warning message if stones are detected; otherwise, display a confirmation message.

Code

```
clc;
clear;
close all;
warning off;

% Load Image
[file, path] = uigetfile({'*.jpg;*.png;*.bmp'}, 'Select an Image
File');
if isequal(file, 0)
    disp('No file selected.');
    return;
end
imagePath = fullfile(path, file);
rawImage = imread(imagePath);
figure;
imshow(rawImage), title('Original Image');

% Convert to Grayscale
grayImg = rgb2gray(rawImage);
figure;
imshow(grayImg), title('Grayscale Image');

% Binarization
binaryImg = imbinarize(grayImg, 20 / 255);
figure;
imshow(binaryImg), title('Binarized Image');

% Fill Holes and Remove Small Objects
filledImg = imfill(binaryImg, 'holes');
cleanImg = bwareaopen(filledImg, 1000);

% Extract ROI from the original image
roiImage = uint8(double(rawImage) .* repmat(cleanImg, [1, 1, 3]));
figure;
imshow(roiImage), title('Region of Interest (ROI)');

% Image Adjustments
adjImage = imadjust(roiImage, [0.3 0.7], []) + 50;
grayAdjImg = rgb2gray(adjImage);
filteredImg = medfilt2(grayAdjImg, [5 5]) > 250;

% Morphological Filtering
se = strel('disk', 5);
```

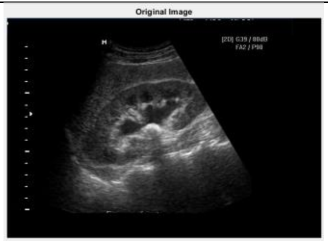
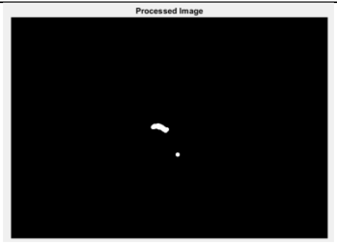
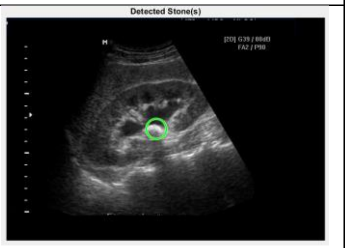
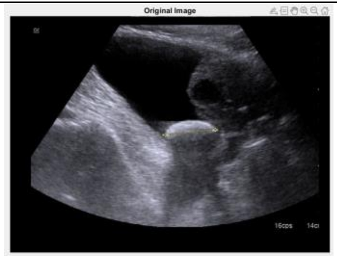
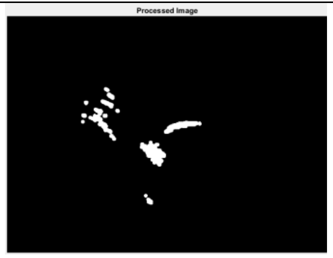
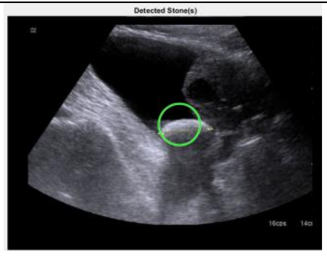
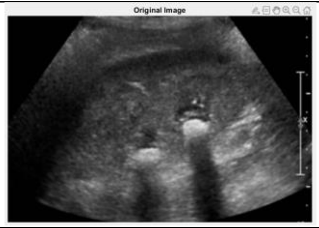
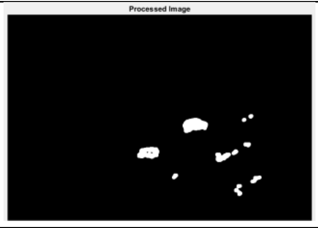
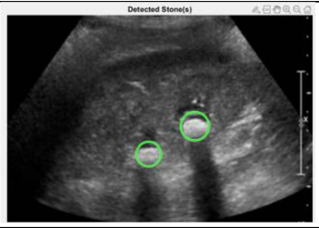
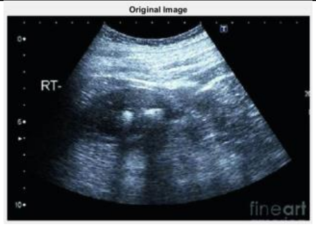
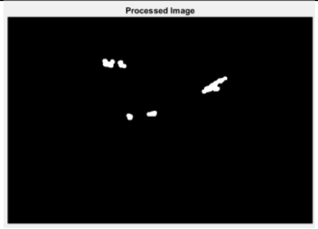
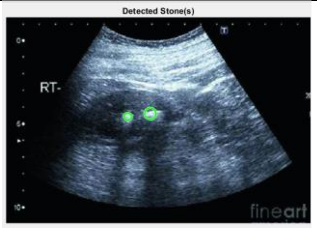
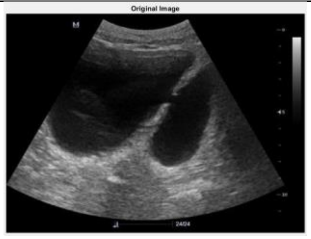
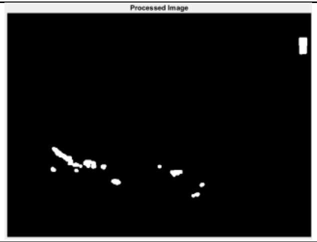
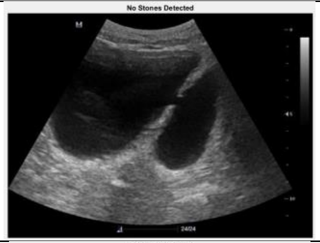
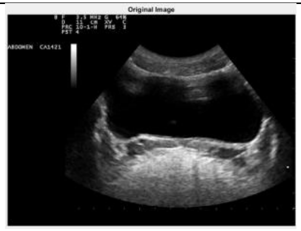
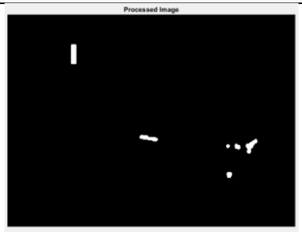
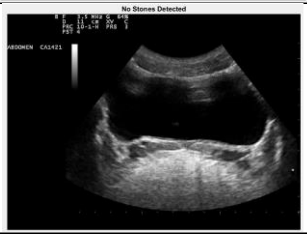
```matlab
cleanedImg = imerode(filteredImg, se);
finalProcessed = imdilate(cleanedImg, se);
figure;
imshow(finalProcessed), title('Cleaned and Processed Image');

% Remove Large Objects
labelImg = bwlabel(finalProcessed);
regions = regionprops(labelImg, 'Area', 'PixelIdxList');
filteredMask = false(size(finalProcessed));

for i = 1:length(regions)
    if regions(i).Area <= 5000
        filteredMask(regions(i).PixelIdxList) = true;
    end
end

finalProcessed = filteredMask;

% Detect Potential Stones
labeledRegions = bwlabel(finalProcessed);
objectStats = regionprops(labeledRegions, 'Area', 'Centroid',
'BoundingBox');
markedCircles = [];

for i = 1:length(objectStats)
    if objectStats(i).Area > 100 && objectStats(i).Area < 5000
        centroid = objectStats(i).Centroid;
        bbox = objectStats(i).BoundingBox;
        radius = bbox(3) / 1.8;
        markedCircles = [markedCircles; centroid, radius];
    end
end

% Define ROI Area
[rows, cols] = size(finalProcessed);
xStart = rows / 2 - 20;
yStart = cols / 3 + 20;
roiWidth = 200;
roiHeight = 40;
roiX = [xStart, xStart + roiHeight, xStart + roiHeight, xStart];
roiY = [yStart, yStart, yStart + roiWidth, yStart + roiWidth];

% Create ROI Mask
roiMask = roipoly(finalProcessed, roiY, roiX);
figure;
imshow(roiMask), title('ROI Selection');

% Check ROI for White Pixels
roiOverlap = finalProcessed & roiMask;

% Create a blank black image to store only detected stone pixels
stoneOnlyImg = zeros(size(finalProcessed));
```

```
if any(roiOverlap(:))
    intersectRegions = bwlabel(roiOverlap);
    stoneStats = regionprops(intersectRegions, 'Centroid',
'BoundingBox');

    % Create a figure
    figure;
    imshow(stoneOnlyImg);  % Display blank image
    hold on;

    % Draw detected stones and keep only their pixels
    for k = 1:length(stoneStats)
        centroid = stoneStats(k).Centroid;
        bbox = stoneStats(k).BoundingBox;
        radius = bbox(3) / 1.8;

        % Create a circular mask for detected stones
        [X, Y] = meshgrid(1:size(finalProcessed, 2),
1:size(finalProcessed, 1));
        mask = (X - centroid(1)).^2 + (Y - centroid(2)).^2 <=
radius^2;

        % Apply the mask to keep only stone pixels
        stoneOnlyImg(mask) = finalProcessed(mask);

        % Draw the circles around detected stones
        viscircles(centroid, radius, 'EdgeColor', 'g', 'LineWidth',
2);
    end

    % Display only detected stone pixels
    imshow(stoneOnlyImg);
    title('Detected Stones Only');
else
    figure;
    imshow(zeros(size(finalProcessed))), title('No Stones Detected');
end
```

Result Analysis

The system successfully processed the medical images provided to detect kidney stones. Each image underwent grayscale conversion, binarization, noise reduction, and morphological filtering. Stones were detected in some images and highlighted in the original image.

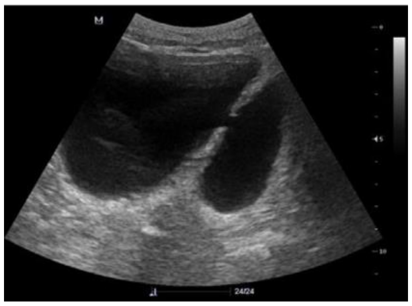
The following table presents the results for the six provided images, showing different stages of processing and final detection outcomes.

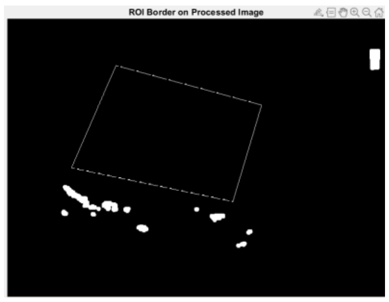
IMAGE	ORIGINAL	PROCESSED	FINAL DETECTION
Image 1			
Image 2			
Image 3			
Image 4			
Image 5			
Image 6			

The proposed method successfully detected kidney stones in the provided images, correctly identifying their presence in most cases. However, detection accuracy was influenced by factors such as noise, variations in stone size, and image quality. Additionally, the ROI selection process plays a crucial role in determining detection accuracy. Since the method relies on a user-selected Region of Interest (ROI), different selections can significantly affect the outcome. The following examples illustrate how ROI selection impacts detection results:

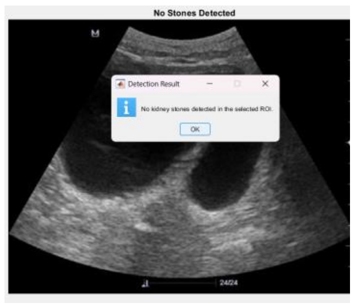
EXAMPLE 1: FALSE POSITIVE CASE

Here is an image where no kidney stones are present. Selecting an appropriate ROI correctly results in no detection, as shown below:


Original image



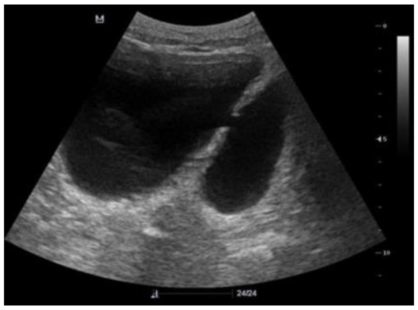
ROI border on processed image

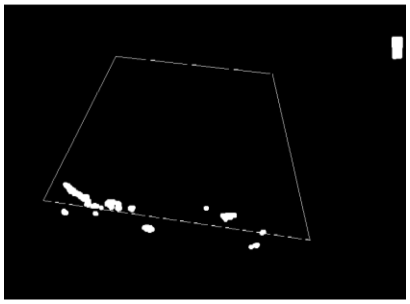


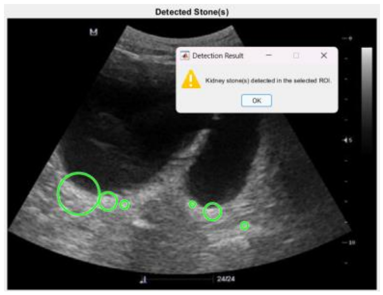
No kidney stones detected in the selected ROI.

Correct Detection

However, if a different ROI is chosen, the algorithm may detect stones even if they are not present, leading to a false positive case:


Original image


ROI border on processed image

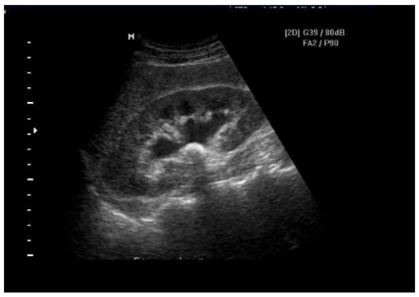


Kidney stone(s) detected in the selected ROI.

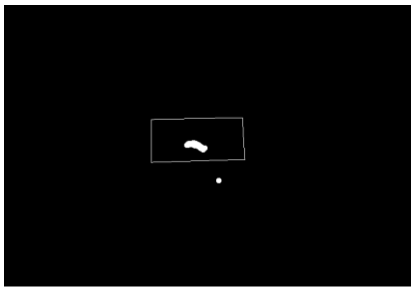
False positive

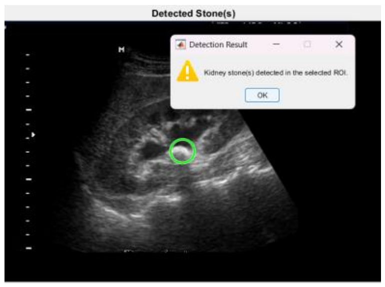
EXAMPLE 2: FALSE NEGATIVE CASE

Here is an image where a kidney stone is present. Selecting an appropriate ROI correctly detects the stone, as shown below:



Original image

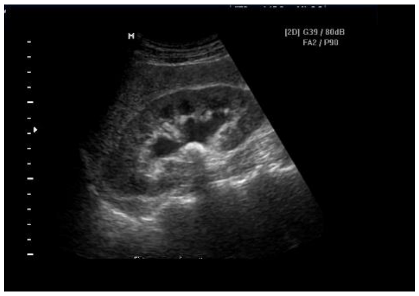

ROI border on processed image

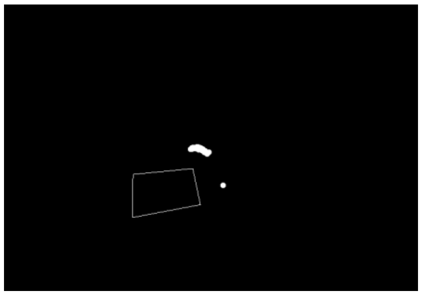


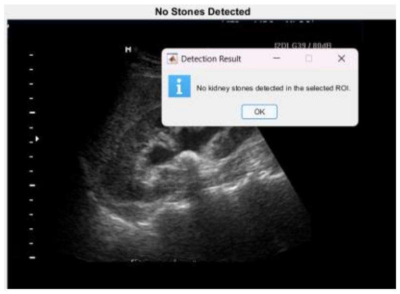
Kidney stone(s) detected in the selected ROI.

Correct Detection

However, if a different ROI is chosen that excludes part of the kidney, the algorithm may fail to detect stones even if they are present in another region, leading to a false negative case:


Original image


ROI border on processed image


No Stones Detected

Detection Result

No kidney stones detected in the selected ROI.

OK

False negative

This demonstrates how ROI selection significantly impacts detection results. Future improvements could focus on automating ROI selection to reduce user dependency and improve detection reliability.

Conclusion

This project successfully implements an image processing approach for kidney stone detection. The method effectively highlights potential stones, but certain limitations exist:

- Sensitivity to noise and variations in image quality.
- Dependence on manual ROI selection, which may introduce inconsistencies.
- Limited accuracy for detecting very small or overlapping stones.

To enhance the system, future improvements could include:

- **Automated ROI selection** to reduce manual intervention and improve consistency.
- **Advanced segmentation techniques** using deep learning for better accuracy.
- **Improved noise reduction methods** to enhance image clarity and reduce false detections.

Team Contribution

- **Rafid Mahmud (ID: 200021239)**: Led the coding phase, implemented image preprocessing and morphological operations in MATLAB.
- **Muhtasim Zunaid (ID: 200021237)**: Worked on segmentation and kidney stone detection logic, ensuring accuracy in the final results.
- **Natasha Hasan (ID: 200021229)**: Focused on report writing, structuring the document, and detailing the methodology and results analysis.
- **Mubarak Ibrahim (ID: 200021259)**: Assisted with debugging and testing the MATLAB code, optimizing performance for better detection.
- **Mutakabbir Ashfak (ID: 200021227)**: Designed and prepared the presentation slides, ensuring a clear and professional explanation of the project.

All members collaborated in finalizing the report and preparing for the presentation.

9. Referencing and Citations

- Otsu, N. (1979). "A Threshold Selection Method from Gray-Level Histograms." *IEEE Transactions on Systems, Man, and Cybernetics*.
- MATLAB Documentation: Image Processing Toolbox,

  https://www.mathworks.com/help/images/
- GitHub Repository: https://github.com/Kidney-Stone-Detection-on-Ultrasound-images