KENNESAW STATE
UNIVERSITY

CS 4732

MACHINE VISION

PROJECT 3

IMAGE RESOLUTION

INSTRUCTOR

Dr. Mahmut KARAKAYA

Jose Rodriguez

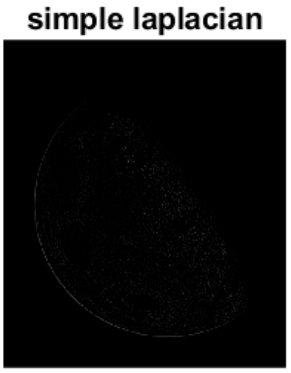
000976962

1. Abstract

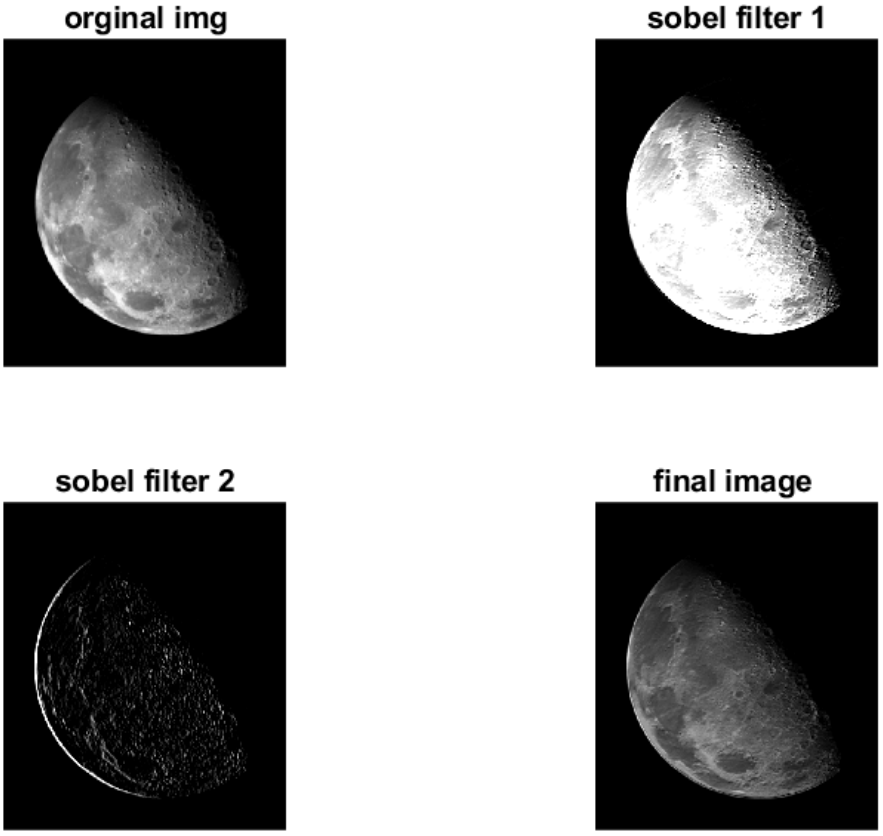
In our third project, we use morphological filter and spatial filtering techniques to edit an image as desired. In this project, some techniques used were the Laplacian filter, Sobel filter, erosion, dilation, opening, closing, and connected components. Upon working on this project and completing it I have learned and discovered many of these techniques in MATLAB already have built-in functions, but while that is true using the function for its purpose properly was a task harder than expected. That being said, working on the project and researching the official MATLAB webpage for functions and examples have led me to a greater understanding of the techniques used and how they work along with how they are implemented in code.

2. Test Results of Images

2.1 Test Results for Laplacian filter



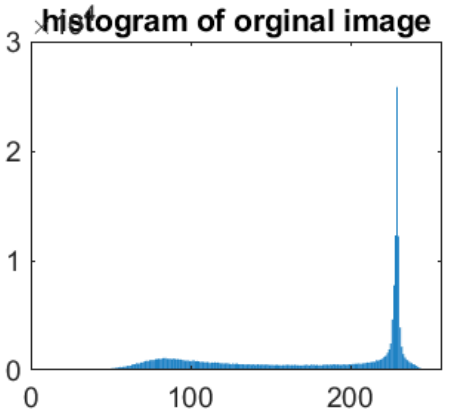
2.2 Test Results for sobel filter



2.3 Test Results for grayscale fingerprint image

orginal image

histogram of orginal image



orginal image with threshold applied

first dilation

second erosion

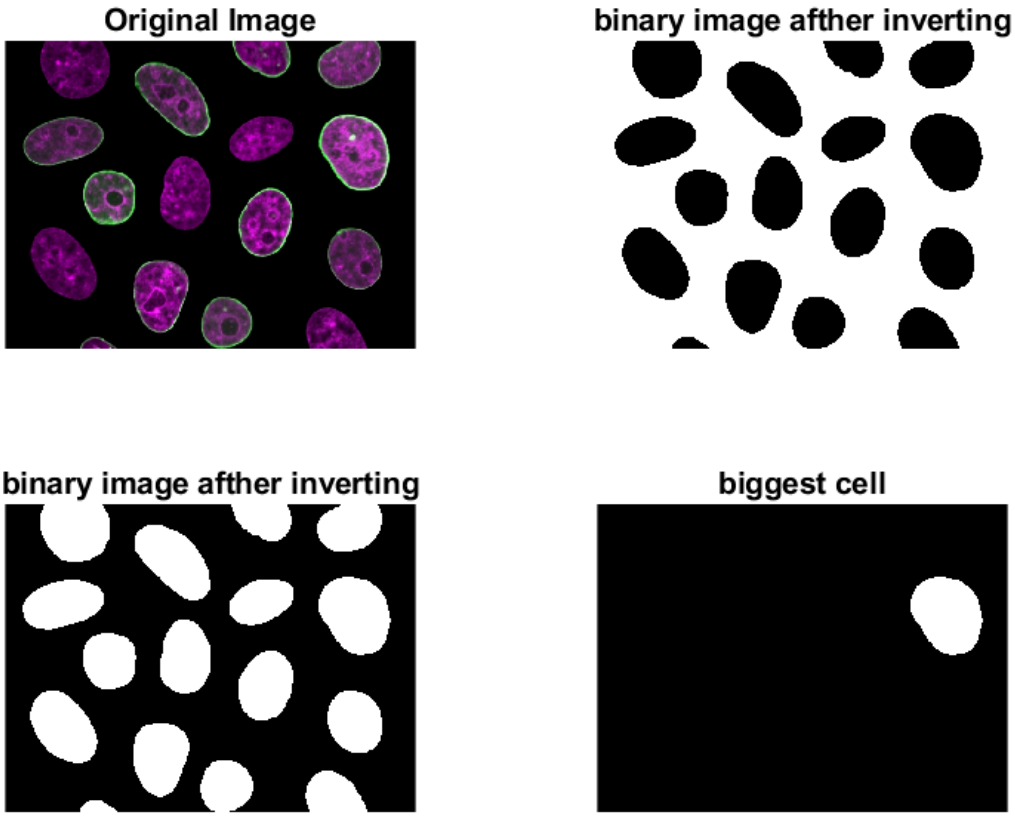
third dilation

fourth erosion

5th dilation

When looking at the threshold value selected, I chose to go with 100 as other numbers left too much of the outer thumbprint white or took too much of the thumbprint itself out and caused many issues. As for the structuring elements, the lecture slides provide several best examples and upon looking at them, I created 5×5 versions of them to test to see which would give the best results to use. Overall, the selected threshold and structuring elements used were the ones that gave the best results, as there weren't specific ones used except for the threshold.

2.4 Test Results for cell.jpg



4 Discussion

In completing this project I found myself researching endlessly and on the final day, I found myself revisiting certain questions to improve my work. While I am satisfied with the work I have done, I can't help but feel as if granted more time to complete part two of the project, I could have created and improved my current image using other techniques but given the short time to complete the section and having others to do I was left with no choice but to accept what I have gotten to and continue my research to complete the project. Overall, my experience with this project has been great and left me with a desire to better improve myself and my coding skills to achieve greater results that properly display the effort I have put in.

5 CODES

5.1 code for Laplace filter

```
%reads in users image
img = imread("moon.jpg");
%creates 3 different types of filters to be used on the original image
laplacian_filter = [0 1 0; 1 -4 1; 0 1 0];
laplacian_filter_2 = [0 -1 0; -1 5 -1; 0 -1 0];
laplacian_filter_3 = [1 1 1; 1 -8 1; 1 1 1];
%applies all 3 filters on image
laplacian_img = imfilter(img,laplacian_filter);
laplacian_img_2 = imfilter(img,laplacian_filter_2);
laplacian_img_3 = imfilter(img,laplacian_filter_3);
% creates a new figure to display the images
figure
%creats a 2x2 tile to place the images
tiledlayout(2,2)
%display the orginal image in the first tile
nexttile
imshow(img);
title('orginal img');
%display the simple laplacian filter image  in the second tile
nexttile
imshow(laplacian_img);
title('simple laplacian');
%display the sharpened image using laplacian in the third tile
nexttile
imshow(laplacian_img_2);
title('sharpened image');
%display another variant of Laplacian in the fourth tile
nexttile
imshow(laplacian_img_2);
title('variant of laplacian');
%saves the figure as a png file
saveas(gcf, 'Laplacian.png');
```

5.2 code for Sobel filter

```
%reads in requested image
img = imread("moon.jpg");
%turns the image into a double
img_2 = im2double(img);
%creates the 2 filters needed for sobel
sobel_filter = [-1 -2 1; 0 0 0; 1 2 1];
sobel_filter_2 = [-1 0 1; -2 0 2; -1 0 1];
%applies the two filters onto the original image
sobel_img = imfilter(img_2,sobel_filter);
sobel_img_2 = imfilter(img_2,sobel_filter_2);
%applies the formula for sobels onto the image
sobel_img_3 = sqrt(sobel_img.^2 + sobel_img_2.^2);
%apply mat2gray to help with coloring issue
sobel_img_3_scale = mat2gray(sobel_img_3);
% creates a new figure to display the images
figure
%creats a 2x2 tile to place the images
tiledlayout(2,2)
%display the original image in the first tile
nexttile
imshow(img);
title('orginal img');
%display the first part of the sobel filter in the second tile
nexttile
imshow(sobel_img);
title('sobel filter 1');
```

```
%display second part of the filter in the third title
nexttile
imshow(sobel_img_2);
title('sobel filter 2');
%display the final image in the fourth tile
nexttile
imshow(sobel_img_3_scale);
title('final image ');
%saves the displayed images
saveas(gcf, 'Sobel.png');
```

5.3 code for grayscale fingerprint image

```
% reads in requested image
img = imread("fingerprint.jpg");
%creates a histogram of the image
img_histo = imhist(img);
%creates a threshold value to be used
threshold_value = 100;
%applies threshold to the image
img_2 = img < threshold_value;
%creates a set of structuring elements to be used
dilation_filter = [0 1 0; 1 1 1; 0 1 0];
dilation_filter_2 = [1 1 1; 1 1 1; 1 1 1];
dilation_filter_3 = [0 0 1 0 0; 0 1 1 1 0; 1 1 1 1 1; 0 1 1 1 0; 0 0 1 0 0];
dilation_filter_4 = [1 1 1 1 1; 1 1 1 1 1; 1 1 1 1 1; 1 1 1 1 1; 1 1 1 1 1];
%creates a set of structuring elements to be used
erosion_filter = [1 1 1; 1 1 1; 1 1 1];
erosion_filter_2 = [0 1 0; 1 1 1; 0 1 0];
%different test ran and picking the best image out of each test below
%test below apply the filters created above to each image below and the
%images after
%A = imopen(img_2,dilation_filter);
%B = imopen(img_2,dilation_filter_2);
%C = imopen(img_2,dilation_filter_3);
%D = imopen(img_2,dilation_filter_4);
%E = imdilate(img_2,dilation_filter);
F = imdilate(img_2,dilation_filter_2);
%G = imdilate(img_2,dilation_filter_3);
%H= imerode(img_2,erosion_filter);
%I = imerode(img_2,erosion_filter_2);
J= imclose(img_2,erosion_filter);
%K = imclose(img_2,erosion_filter_2);
L = imdilate(J,dilation_filter);
%M = imdilate(J,dilation_filter_2);
%N = imdilate(J,dilation_filter_3);
%O = imdilate(J,dilation_filter_4);
%P = imopen(J,dilation_filter);
%Q = imopen(J,dilation_filter_2);
%R = imopen(J,dilation_filter_3);
%S = imopen(J,dilation_filter_4);
%T = imclose(J,erosion_filter);
U = imclose(J,erosion_filter_2);
%V = imerode(J,erosion_filter);
%W = imerode(J,erosion_filter_2);
%X = imerase(J,erosion_filter);
%Y = imerase(J,erosion_filter_2);
%Z =imdilate(J,dilation_filter);
AA = imdilate(J,dilation_filter_2);
%AB = imdilate(J,dilation_filter_3);
%AC = imdilate(J,dilation_filter_4);
%creats a 2x2 tile to place the images
figure
tiledlayout(2,2)
%display the original image in the first tile
nexttile
imshow(img);
```

```
title('original image');
nexttile
bar(img_histo);
title('histogram of original image');
nexttile
imshow(img_2);
title('original image with threshold applied');
nexttile
imshow(F);
title('first dilation');
pause;
%saves the figure as a png file
saveas(gcf, 'Question2_orginal.png');
%creates a 2x2 tile to place the images
figure
tiledlayout(2,2)
nexttile
imshow(J);
title('second erosion');
nexttile
imshow(L);
title('third dilation');
nexttile
imshow(U);
title('fourth erosion');
nexttile
imshow(AA);
title('5th dilation');
%saves the figure as a png file
saveas(gcf, 'Question2_Test.png');
```

5.4 code for cell.jpg

```
% reads in requested image
img = imread("cell.jpg");
%turns the image into grayscale
gray_img = rgb2gray(img);
%creates the threshold value we will use later
threshold_value = 10;
%applies threshold value to image
img_2 = gray_img < threshold_value;
%inverts the image to let bwconncomp function count cells properly
img_3 = ~img_2;
%creates the function we will use to count each cell with a 8 connectivity
connected_componets = bwconncomp(img_3,8);
%counts the total number of cells
total_cells = connected_componets.NumObjects;
%displays the total number of cells
disp(['Total number of cells:', num2str(total_cells)]);
%grabs the area and number of pixels from each cell
stats = regionprops(connected_componets,'area','PixelIdxList');
%displays each cell and their area in pixels
for i = 1:length(stats)
   disp(['Area of cell ' , num2str(i),': ',num2str(stats(i).Area), ' pixels']);
end
%displays the biggest cell
disp(['Biggest cell ' , num2str(15),': ',num2str(stats(15).Area), ' pixels']);
%creates a binary image filled with false values
cell_15 = false(size(img_3));
%fills the binary image created with only truth values equal to the pixel
%values of the biggest cell
cell_15(stats(15).PixelIdxList) = true;
% creates a new figure to display the images
figure
%creats a 2x2 tile to place the images
tiledlayout(2,2)
%display the original image in the first tile
```

```
nexttile
imshow(img);
title('Original Image');
%display the binary image  in the second tile
nexttile
imshow(img_2);
title('binary image after inverting');
%display the binary image after inverting image in the third tile
nexttile
imshow(img_3);
title('binary image after inverting');
%display the biggest cell in the fourth tile
nexttile
imshow(cell_15);
title('biggest cell');
%saves the figure as a png file
saveas(gcf,'Question3.png');
```