KENNESAW STATE
UNIVERSITY

CS 4732

MACHINE VISION

PROJECT 2

IMAGE RESOLUTION

INSTRUCTOR

Dr. Mahmut KARAKAYA

Jose Rodriguez

000976962

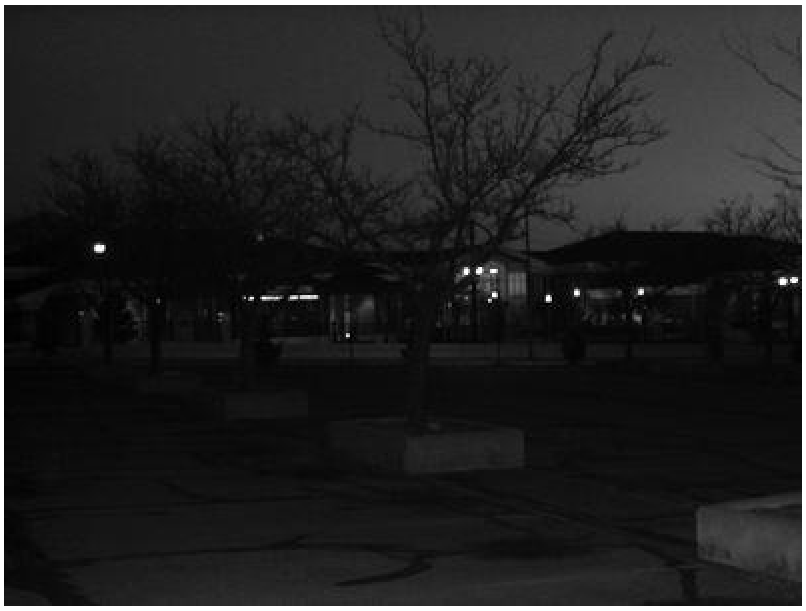
1. Abstract

In Project 2, we demonstrated some of the new image enhancement techniques we learned this week, specifically log and power transformations, along with using average and median filters. In researching this project, I found that looking back at the slides and the official MATLAB website for the functions needed helped guide me to the answers.For the second part, we were tasked with applying the histogram equalization technique. By doing so, we can see how the grayscale on the histogram chart goes from being right-skewed to a more balanced and spread form.For parts 4 and 5, we were tasked with removing noise from an image using average and median filters with different filter sizes for each one. While deciding which method was better is subjective based on the results, median filters did a better job of clearing the noise.

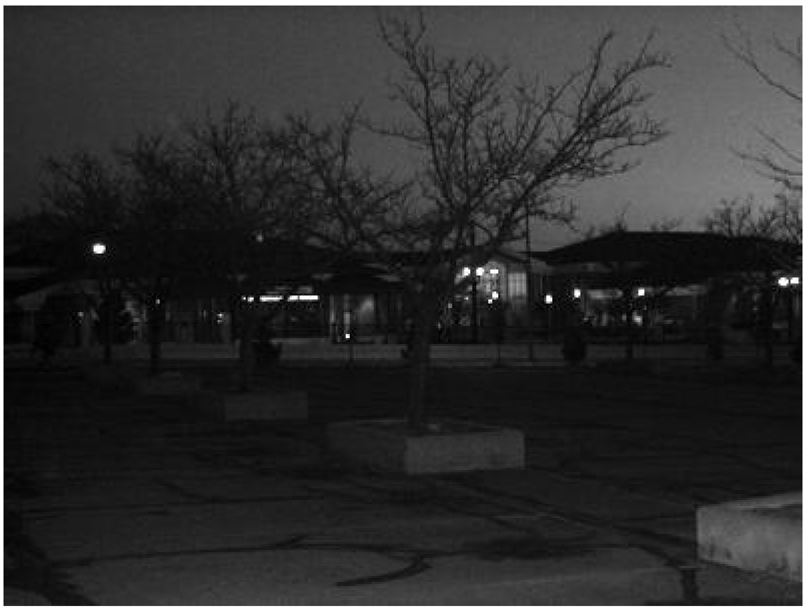
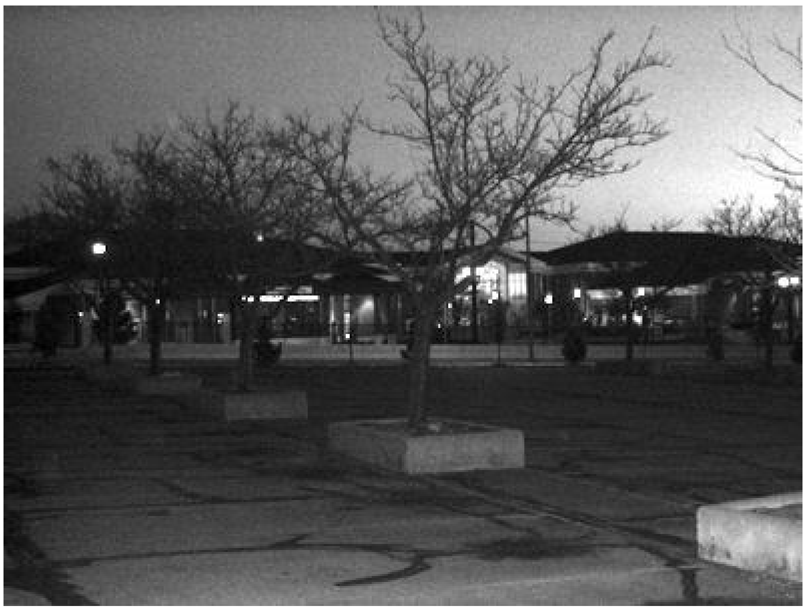
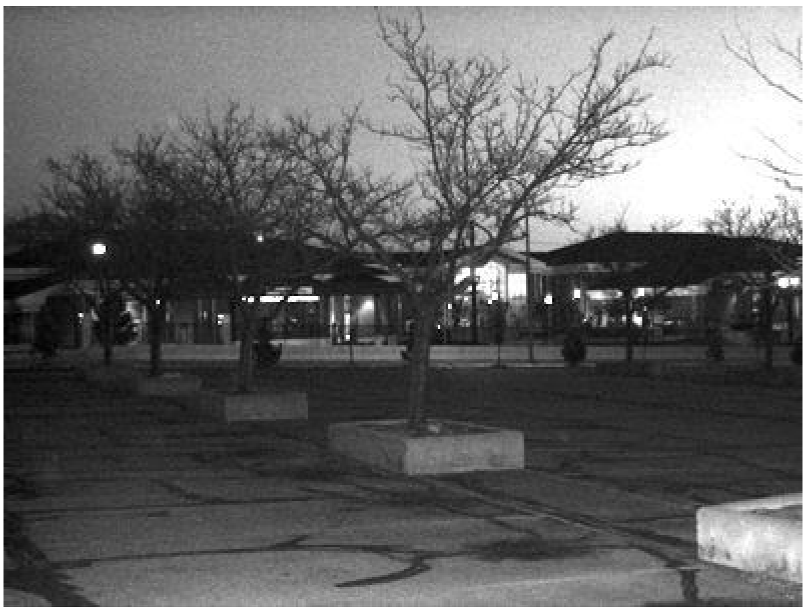
In project 2 we get to demonstrate

2. Test result of image

2.1 Test Results for log transformation



A


B

C

D

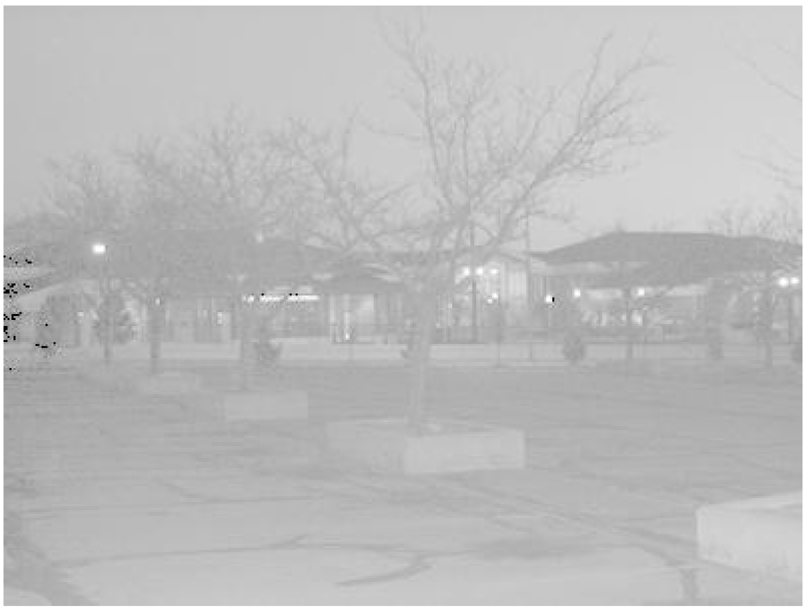
F

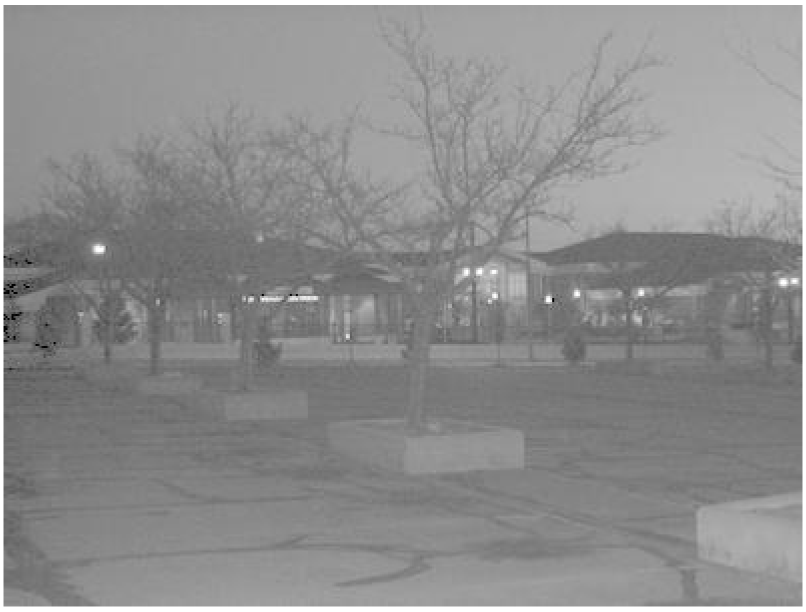
F

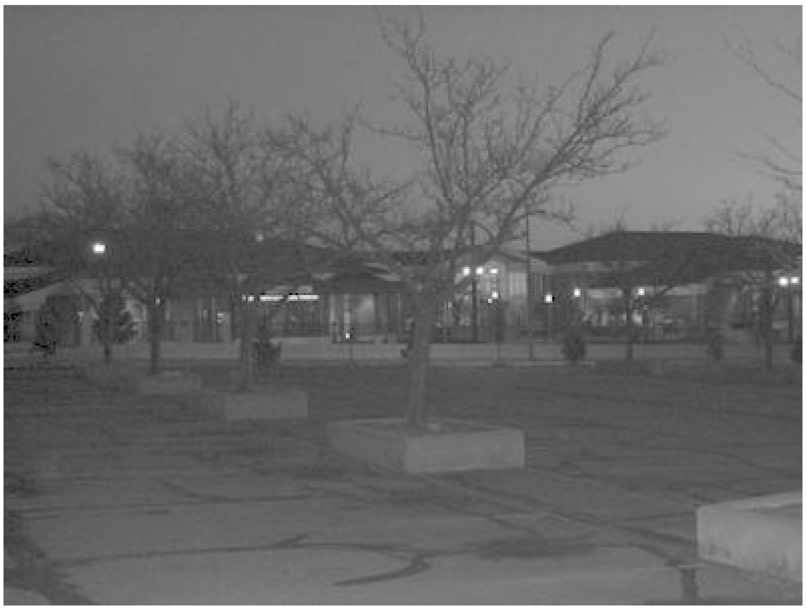
A-F show the effects of log transformation on an image factoring it by 1-9

2.2 Test Results for power transformation

A


B


C


D

E

F

G

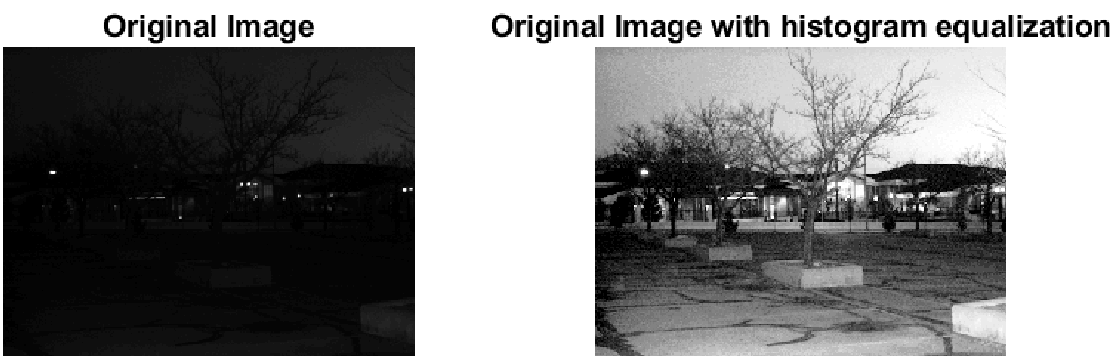
H

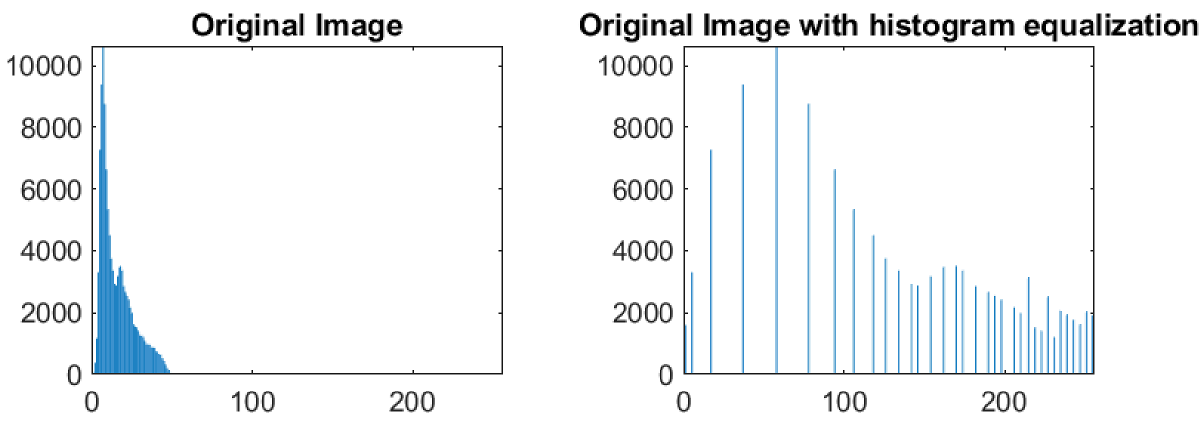
I

J

A-j show the effects of power transformation on an image factoring it by 1-9 With a c value equal to 1

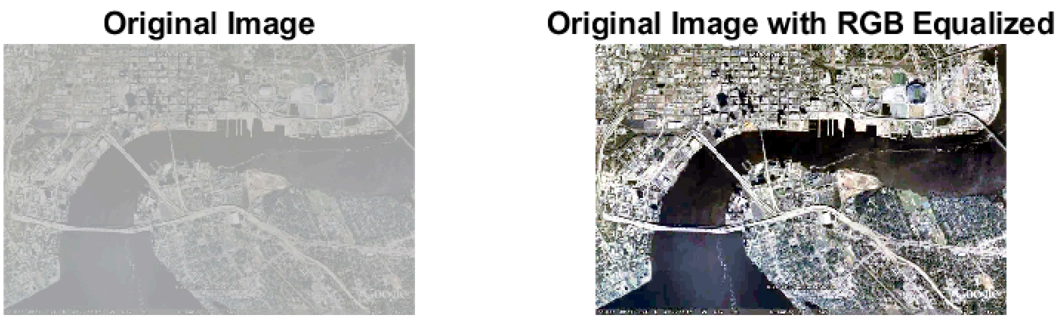
2.3 Test Results for Image histogram equalization

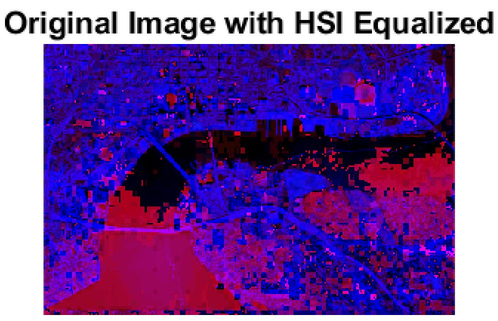




The image above displays both the edited images with it's histogram chart along with the originals

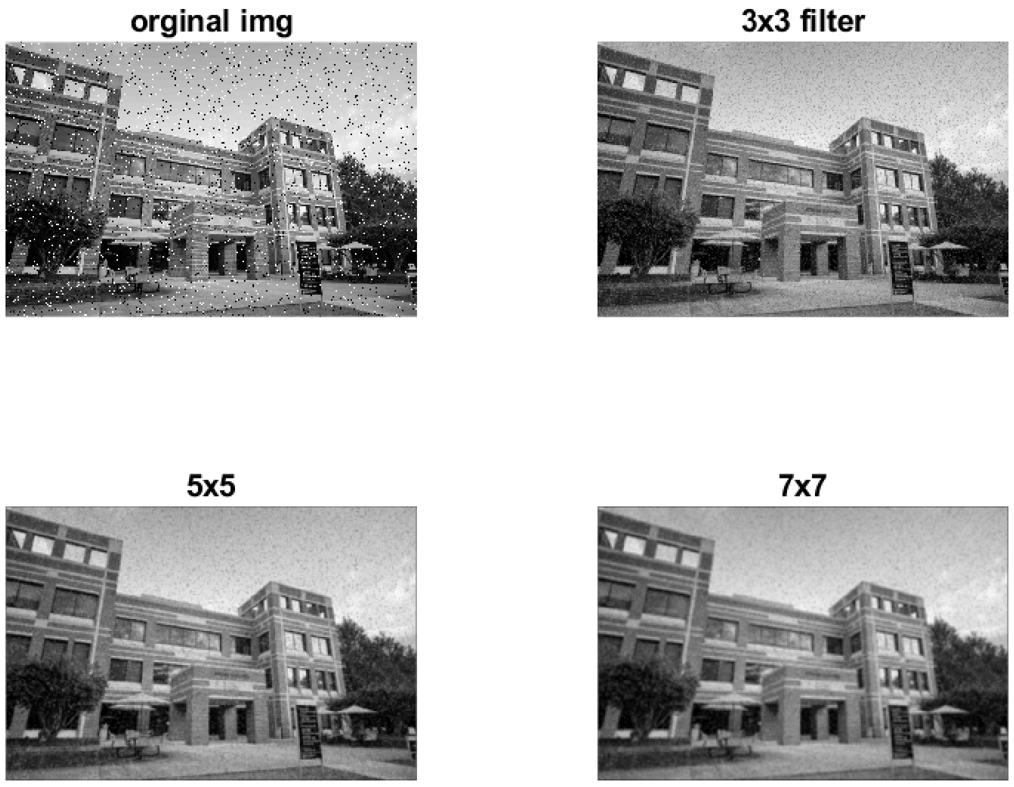
2.4 Test Results for Image histogram equalization to sat_map





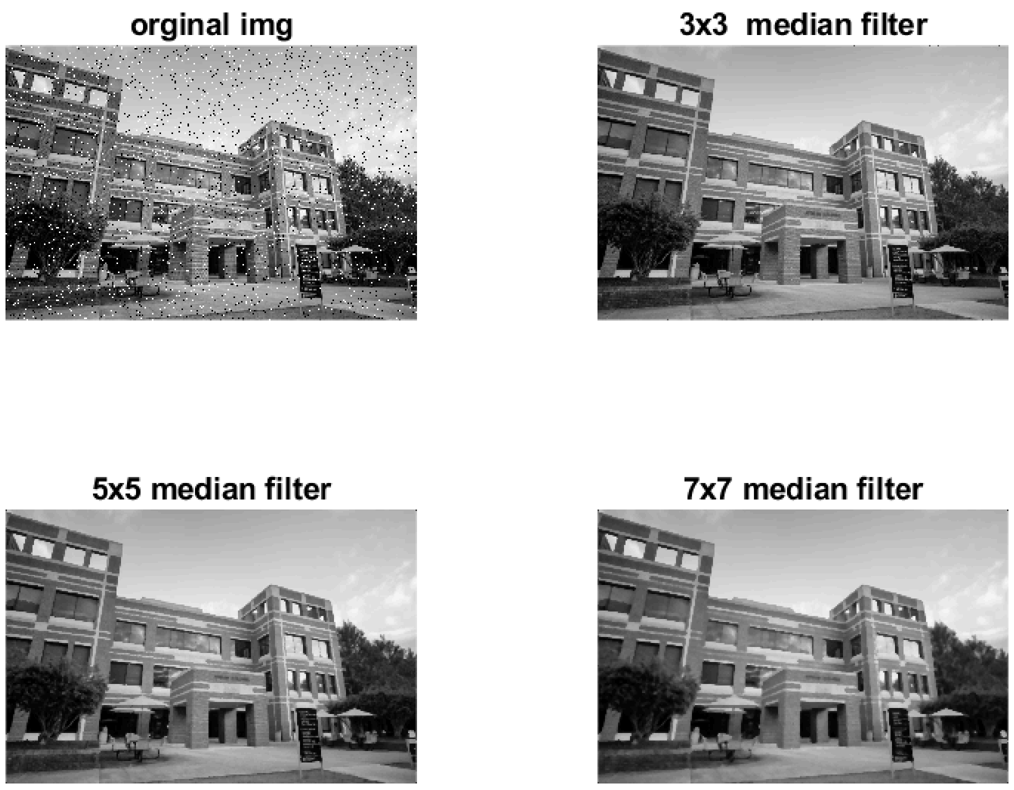
The image above shows the histogram equalization effect on RGB and HSI

2.5 Test Results for average filter



The image above displays the effects of an average filter with a 3x3 5x5 and 7x7 pattern

2.6 Test Results for median filter



The image above shows the effects of the median filter with a 3x3 5x5 and 7x7 filter

4 Discussion

In this project, we had the opportunity to implement new image editing techniques and learn how to code them, as well as understand the reasons behind the results. During my research on the MATLAB official website for necessary functions, I discovered new methods for printing images and generating grayscale histograms. For my next project, I plan to apply some of the techniques I learned while working on this one. Additionally, I will allow more time for future projects, as I spent 3 days on this one. I believe allowing myself extra time will lead to better quality code.Overall my experience in working on this project was much better and more fun now that i'm starting to understand more and more about MATLAB functions and the proper way to implement certain image editing techniques.

5 CODES

5.1 code for log transformation

```
%starts a timer to see run time
tic
%reads in requested image
img_1 = imread('university.png');
%creates an array of factors to be used in log transofrmation
factors = [1,2,3,4,5,6,7,8,9];
%creates a for loop starting at 1 till the the end of the factor array
for i= 1:length(factors)
   %transform the image to double for log transformation
   new_img_1 = im2double(img_1);
   %keeps track of the current element in the array for later use
   factor = factors(i);
   %set c = to the current factor value
   c = factor;
   %calculates the log transformation formula and applies to image
   s = c * log(1+ new_img_1);
   %displayes the image for the user
   imshow(s);
   %pauses the code to allow user input
   pause;
   %creates the file name with its factor number used
   filename = sprintf('log_transformation%d.jpg', factor);
   %will save the images to local file with their name and factor number
   imwrite(s, filename);
end
%end of timer and will print out run time
toc
% avg run time without pause = ~ .467
```

5.2 code for power transformation

```
%starts a timer to see run time
tic
%reads in requested image
img_1 = imread('university.png');
%creates an arry of values to be used for c in the power transformation
factors_for_c = [1,2,3,4,5,6,7,8,9];
%creates an arry of values to be used for y in the power transformation
factors_for_y = [.1,.2,.3,.4,.5,.6,.7,.8,.9];
%creates a for loop starting at 1 till the the end of the factor_y array
for i= 1:length(factors_for_y)
   %transform the image to double for power transformation
   new_img_1 = im2double(img_1);
   %keeps track of the current element in factors_for_c
   factor = factors_for_c(i);
   %keeps track of the current element in factors_for_y
   factor_2 = factors_for_y(i);
 %set c = 1 currently or can be set to factor for diffrent levels to test
   c = 1;%factor;
   %set y = to the valuse of factor_2
   y = factor_2;
   %calculate the power transformation and apply it to the image
   s = c * new_img_1.^y;
   %display the image for the user
   imshow(s);
   %pause the code to allow user input
   pause;
   %creates a file name for the current factor value
   filename = sprintf('power_transformation%d.jpg', factor_2);
   %saves the current image with the files name
   imwrite(s, filename);
end
%end of timer and will printout run time
toc
% avg run time without pause ~= .495
```

5.2 code for histogram equalization

```
%inputs the requested image
img_1 = imread('university.png');
%applies the  histogram equalization to im_1
img_2 = histeq(img_1);
%calculates the histogram of the orginal image up to 255 grayscale
img_edit1 = imhist(img_1,255);
```

```
%calculates the histogram of the new image up to 255 grayscale
img_edit2 = imhist(img_2,255);
% creates a new figure to diplay the images and histograms
figure
%creats a 2x2 tile to place the images and histograms
tiledlayout(2,2)
%display the orginal image in the first tile
nexttile
imshow(img_1)
title('Original Image')
%displayes the edited image with the histogram in the tile next to the
%orginal
nexttile
imshow(img_2)
title('Original Image with histogram equalization')
%display the histogram of the orginal image in the 3rd tile
nexttile
bar(img_edit1);
title('Original Image')
%display the histogram of the edited image in the 4th title
nexttile
bar(img_edit2);
title('Original Image with histogram equalization')
%saves the figure as a png file
saveas(gcf, 'tiledlayout_Question2.png');
```

5.3 code for histogram equalization to sat_map

```
% reads in requested image
img = imread('sat_map.png');
%extracts the red green blue color channels from image
r = img(:,:,1);
g = img(:,:,2);
b = img(:,:,3);
%applies histogram equlization to each color channel
r_HQ = histeq(r);
g_HQ = histeq(g);
b_HQ = histeq(b);
%combines the equlized color channels back into an RGB image
img_RGB = cat(3, r_HQ, g_HQ, b_HQ);
%transform the image into HSI color space
img_HSI = rgb2hsv(img);
%extract each channel of HSI
h = img_HSI(:,:,1);
s = img_HSI(:,:,2);
i = img_HSI(:,:,3);
%applies histogram equlization only to the i channel as requested
I_HQ = histeq(i);
%combines the HSI channels back into an HSI image
img_HSI = cat(3,h,s,I_HQ);
% creates a new figure to diplay 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('Original Image');
%display the RGB equlized image in the second tile
nexttile
imshow(img_RGB);
title('Original Image with RGB Equalized');
%display the HSI equlized channel in the 3rd tile
nexttile
```

```
imshow(img_HSI);
title('Original Image with HSI Equalized');
pause;
%saves the figure as a png file
saveas(gcf, 'tiledlayout_Question3.png');
```

5.4 code for average filter

```
%reads in the requested image
img = imread('noisy_atrium.png');
%defines the sizes of the filters to be used
filter_3 = 3;
filter_5 = 5;
filter_7 = 7;
%creates a 3x3 5x5 7x7 average filters
average_3 = ones(filter_3) / (filter_3^2);
average_5 = ones(filter_5) / (filter_5^2);
average_7 = ones(filter_7) / (filter_7^2);
%applies the 3x3 5x5 7x7 filters to the iamge
filter_3_img = imfilter(img,average_3);
filter_5_img = imfilter(img,average_5);
filter_7_img = imfilter(img,average_7);
% creates a new figure to diplay 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 3x3 filter image  in the second tile
nexttile
imshow(filter_3_img);
title('3x3 filter');
%display the 5x5 filter image  in the third tile
nexttile
imshow(filter_5_img);
title('5x5');
%display the 7x7 filter image  in the 4th tile
nexttile
imshow(filter_7_img);
title('7x7');
%saves the figure as a png file
saveas(gcf, 'tiledlayout_Question4.png');
```

5.5 code for median filters

```
%reads in the requested image
img = imread('noisy_atrium.png');
%defines the sizes of the filters to be used
filter_3 = [3 3];
filter_5 = [5 5];
filter_7 = [7 7];
%applies the 3x3 5x5 7x7 filters to the iamge useing the medfilt2 function
median_3 = medfilt2(img,filter_3);
median_5 = medfilt2(img,filter_5);
median_7 = medfilt2(img,filter_7);
% creates a new figure to diplay the images
figure
%creats a 2x2 tile to place the images
tiledlayout(2,2)
%display the orginal image in the first tile
imshow(img);
title('orginal img');
%display the 3x3 filtered image in the second tile
nexttile
```

```
imshow(median_3);
title('3x3  median filter');
%display the 5x5 filtered image in the 3rd tile
nexttile
imshow(median_5);
title('5x5 median filter');
%display the 7x7 filtered image in the 4th tile
nexttile
imshow(median_7);
title('7x7 median filter');
%saves the figure as a png file
saveas(gcf, 'tiledlayout_Question5.png');
```