CS 216 Homework 1

Zachary DeStefano, 15247592

Due Date: April 9

Problem 4

Problem 4a

The variable x will be assigned to an array that is the numbers 1 through 5 arranged in a random order.

Problem 4b

The variable a will be assigned to an array that is the numbers 1 through 10 arranged in chronological order. The variable b will end up being assigned an array that is the numbers $1, 4, 7, 10$ because it is the first number and then every third number after that.

Problem 4c

The variable f will be assigned to an array that is the numbers 1501 to 2000 in chronological order. The variable g will be assigned to the indices of f where the entries are greater than 1850 which is 351 to 500. The variable h will then be assigned an array that is the numbers 1851 to 2000.

Problem 4d

The variables x will be assigned to a vector of 10 entries where each entry has value 22. The variable y will be assigned the sum of the entries which will be 220.

Problem 4e

The variable a will be assigned an array that is the numbers 1 to 1000 in chronological order. The variable b will be assigned a vector that is the numbers 1 to 1000 in reverse order.

Problem 5

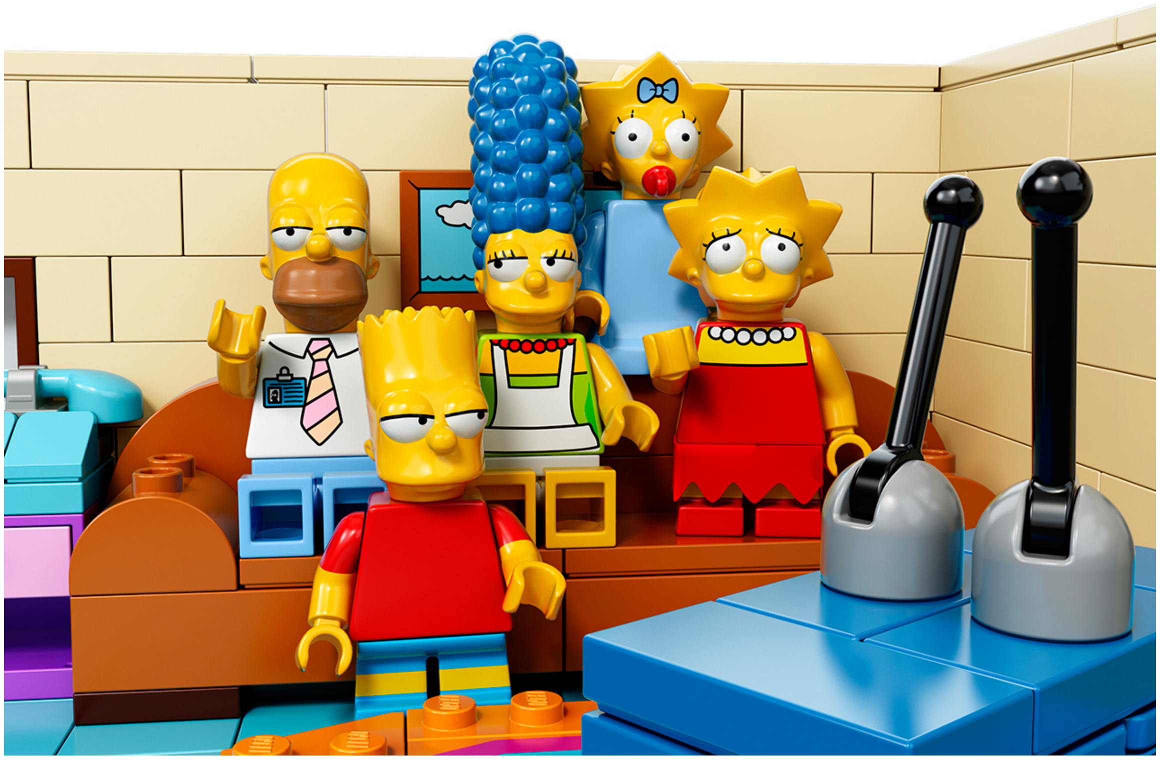

Figure 1: Original Image from the Internet

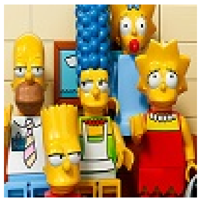

Figure 2: Modified 100 by 100 image

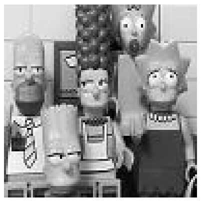

Figure 3: Modified 100 by 100 image in grayscale

Here is the initialization code:

```
A = rgb2gray(imread('modifiedImageProb5.jpg'));
A = im2double(A);
imwrite(A,'blackWhiteImage.jpg','JPEG');
```

Problem 5a

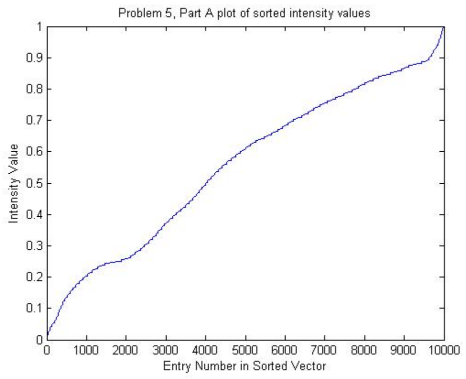

Figure 4: Plot of Intensity Values

Here is the code used to get the plot:

```
singleVectorA = A(:);
sortedValuesA = sort(singleVectorA);
figure
plot(sortedValuesA);
xlabel('Entry Number in Sorted Vector');
ylabel('Intensity Value');
title('Problem 5, Part A plot of sorted intensity values');
```

Problem 5b

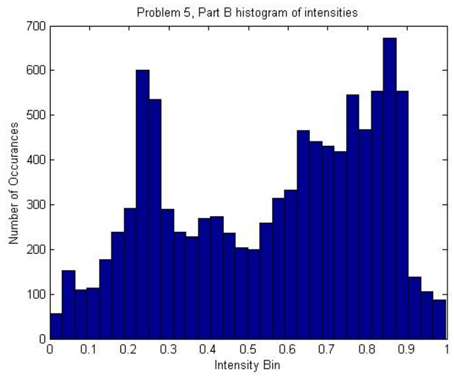

Figure 5: Histogram of Intensities with 32 bins

Here is the code used to get the plot:

```
figure
hist(singleVectorA,32);
xlabel('Intensity Bin');
ylabel('Number of Occurances');
title('Problem 5, Part B histogram of intensities');
```

Problem 5c

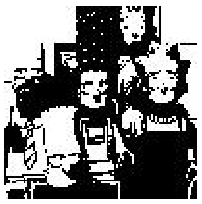

Figure 6: Binary Image using 0.65 as the threshold

Here is the code used to get the image:

```
threshold = 0.65;
binaryA = zeros(size(A));
```

```
binaryA(A > threshold) = 1;
imwrite(binaryA,'partCimage.jpg','JPEG');
```

Problem 5d

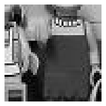

Figure 7: Bottom Right Quadrant of A

Here is the code used to get the image:

```
sizeA = size(A);
sizeAhoriz = sizeA(1);
sizeAvert = sizeA(2);
bottomRightA = A(sizeAhoriz/2:sizeAhoriz,sizeAvert/2:sizeAvert);
imwrite(bottomRightA,'partDimage.jpg','JPEG');
```

Problem 5e

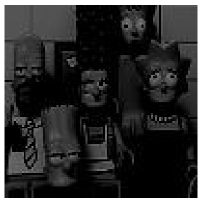

Figure 8: New image with mean subtracted

Here is the code used to get the image:

```
meanBrightness = mean(singleVectorA);
meanSubtractedA = A-meanBrightness;
meanSubtractedA( meanSubtractedA < 0) = 0;
imwrite(meanSubtractedA,'partEimage.jpg','JPEG');
```

Problem 5f

```
function [ result ] = diceRoll_partF()
%DICEROLL_PARTF Summary of this function goes here
%   Detailed explanation goes here

result = floor(rand(1)*6) + 1;

end
```

Problem 5g

```
y = [1:6];
z = reshape(y,3,2);
```

Problem 5h

Result is that $x = 0$ and $[r, c] = [95, 99]$. Code to accomplish it is the following

```
x = min(singleVectorA);
[r,c] = find(A==x,1);
```

Problem 5i

The result is 5 unique values. Here is the code that was used:

```
v = [1 8 8 2 1 3 9 8];
sizeUniqueV = size(unique(v));
numUnique = sizeUniqueV(2);
```

Problem 5 Compiled Code

Here is the entire script used for problem 5, except for part F which was a separate function. This just puts all the code pieces from the previous sections together:

```
A = rgb2gray(imread('modifiedImageProb5.jpg'));
A = im2double(A);
imwrite(A,'blackWhiteImage.jpg','JPEG');

%Part A
singleVectorA = A(:);
sortedValuesA = sort(singleVectorA);
figure
plot(sortedValuesA);
xlabel('Entry Number in Sorted Vector');
ylabel('Intensity Value');
title('Problem 5, Part A plot of sorted intensity values');

%Part B
figure
hist(singleVectorA,32);
xlabel('Intensity Bin');
ylabel('Number of Occurances');
title('Problem 5, Part B histogram of intensities');

%Part C
threshold = 0.65;
binaryA = zeros(size(A));
binaryA(A > threshold) = 1;
imwrite(binaryA,'partCimage.jpg','JPEG');

%Part D
sizeA = size(A);
sizeAhoriz = sizeA(1);
sizeAvert = sizeA(2);
bottomRightA = A(sizeAhoriz/2:sizeAhoriz,sizeAvert/2:sizeAvert);
imwrite(bottomRightA,'partDimage.jpg','JPEG');

%Part E
meanBrightness = mean(singleVectorA);
```

```
meanSubtractedA = A-meanBrightness;
meanSubtractedA( meanSubtractedA < 0) = 0;
imwrite(meanSubtractedA,'partEimage.jpg','JPEG');

%diceRoll_partF.m is part F

%Part G
y = [1:6];
z = reshape(y,3,2);

%Part H
x = min(singleVectorA);
[r,c] = find(A==x,1);

%Part I
v = [1 8 8 2 1 3 9 8];
sizeUniqueV = size(unique(v));
numUnique = sizeUniqueV(2);
```

Problem 6

Helper method for getting image data

The following function is used by the rest of the functions to get image data:

```
function [ colorImages, grayscaleImages ] = getImageSetData( dirname, fileformat )
%GETIMAGESETDATA This gets the multi-dimensional array of color image data
%                and grayscale image data. All images must be same size
%   Input
%       dirname: name of directory to search
%       fileformat: file format string for images to put into set
%   Output
%       colorImages: color data array
%       grayscaleImages: grayscale data array

dirString = strcat(dirname,fileformat);
setList = dir(dirString);
numImages = length(setList);
```

```
imnameStart = strcat(dirname,'/');

%get info on first image
firstColorImage = imread([imnameStart setList(1).name]);
firstGrayscaleImage = rgb2gray(firstColorImage);
colorImages = zeros([size(firstColorImage) numImages]);
grayscaleImages = zeros([size(firstGrayscaleImage) numImages]);

for i=1:numImages
   imname = [imnameStart setList(i).name];
   currentColorImage = imread(imname);
   currentGrayscaleImage = rgb2gray(currentColorImage);
   colorImages(:,:,:,i) = im2double(currentColorImage);
   grayscaleImages(:,:,i) = im2double(currentGrayscaleImage);
end

end
```

Part A and D, Average Grayscale Image

The following function writes and displays the average grayscale image:

```
function [] = writeAverageGrayscaleImage( dirname, fileformat, outputImageName )
%WRITEAVERAGEGRAYSCALEIMAGE This writes and displays the average grayscale image
%   Input
%       dirname: name of directory to search
%       fileformat: file format string for images to put into set
%       outputImageName: name to use when writing output file


[~, grayscaleImages] = getImageSetData(dirname,fileformat);
averageGrayscaleImage = mean(grayscaleImages,3);
imwrite(averageGrayscaleImage,outputImageName,'JPEG');

figure
imagesc(averageGrayscaleImage);
axis image;

end
```

Part B and D, Average Color Image

The following function computes, writes, and displays the average color image:

```
function [] = writeAverageColorImage( dirname, fileformat, outputImageName )
%WRITEAVERAGEGRAYSCALEIMAGE This writes and displays the average color image
%
%   Input
%       dirname: name of directory to search
%       fileformat: file format string for images to put into set
%       outputImageName: name to use when writing output file


[colorImages, ~] = getImageSetData(dirname,fileformat);
averageColorImage = mean(colorImages,4);
imwrite(averageColorImage,outputImageName,'JPEG');

figure
image(averageColorImage);
axis image;

end
```

Part C and D, Standard Deviation Image

The following function computes and writes the standard deviation of the grayscale image data.

```
function [] = writeStndDevImage( dirname, fileformat, outputImageName )
%WRITESTNDDEVIMAGE This writes and displays the standard deviation image
%
%   Input
%       dirname: name of directory to search
%       fileformat: file format string for images to put into set
%       outputImageName: name to use when writing output file
```

```
[~, grayscaleImages] = getImageSetData(dirname,fileformat);

%compute standard deviation
stndDevMatrix = std(grayscaleImages,1,3);

figure
imagesc(stndDevMatrix);
axis image;

minStndDev = min(stndDevMatrix(:));
maxStndDev = max(stndDevMatrix(:));
scaledStndDevMatrix = (stndDevMatrix-minStndDev)./(maxStndDev-minStndDev);
imwrite(scaledStndDevMatrix,outputImageName,'JPEG');

end
```

Problem 6 script

This is the script used to writes all the images for problem 6

```
setname = 'set2';
dirname = strcat('imageSet/',setname);
fileformat = '/*.jpg';

colorImageName = strcat(setname,'color.jpg');
grayscaleImageName = strcat(setname,'grayscale.jpg');
stndDevImageName = strcat(setname,'grayscaleStndDev.jpg');

writeAverageGrayscaleImage(dirname,fileformat,grayscaleImageName);
writeAverageColorImage(dirname,fileformat,colorImageName);
writeStndDevImage(dirname,fileformat,stndDevImageName);
```

set1 results

Here are the images that resulting from running the script where the setname variable was set to $set1$.

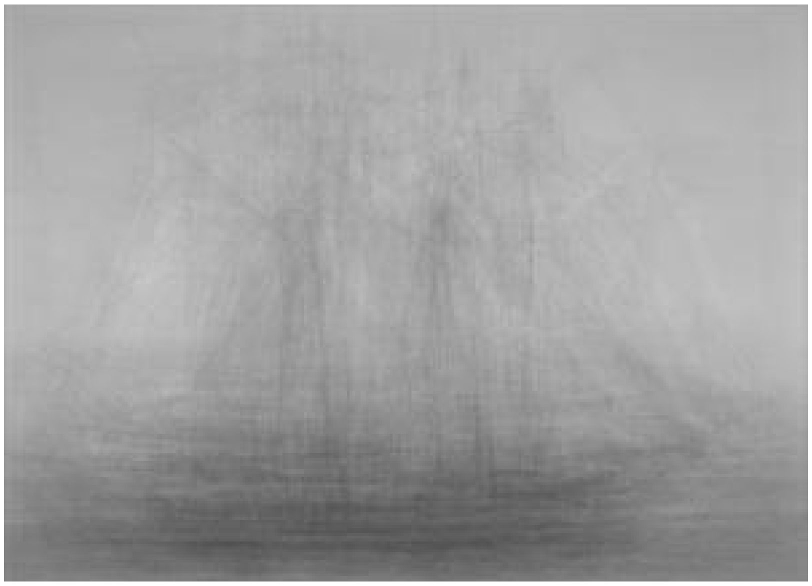

Figure 9: Part A, average grayscale image

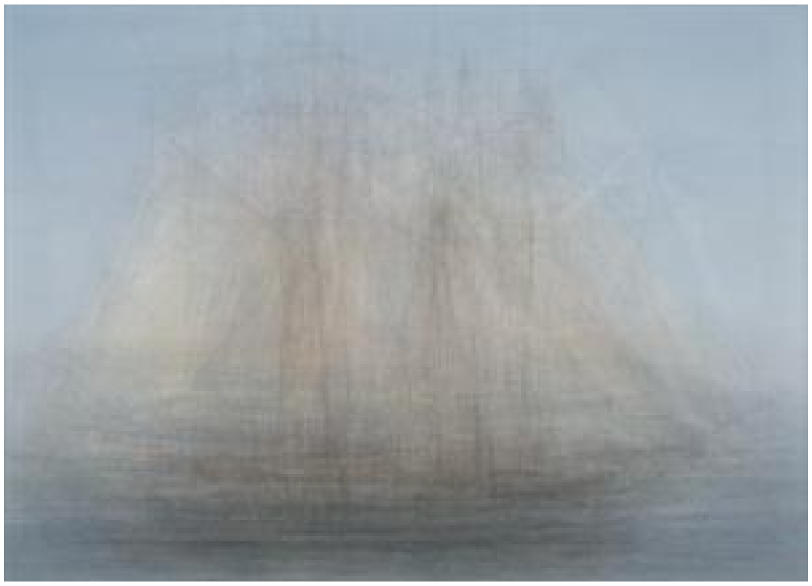

Figure 10: Part B, average color image

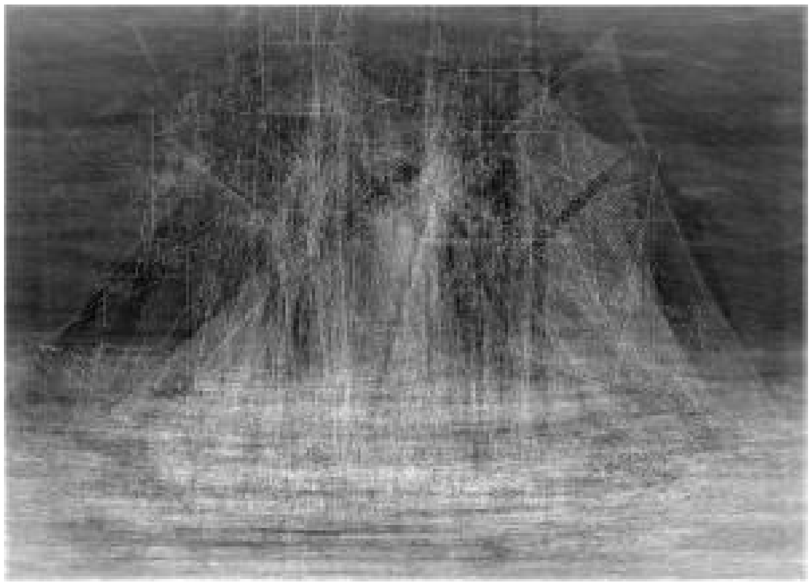

Figure 11: Part C, standard deviation image

set2 results

Here are the images that resulting from running the script where the setname variable was set to $set2$.

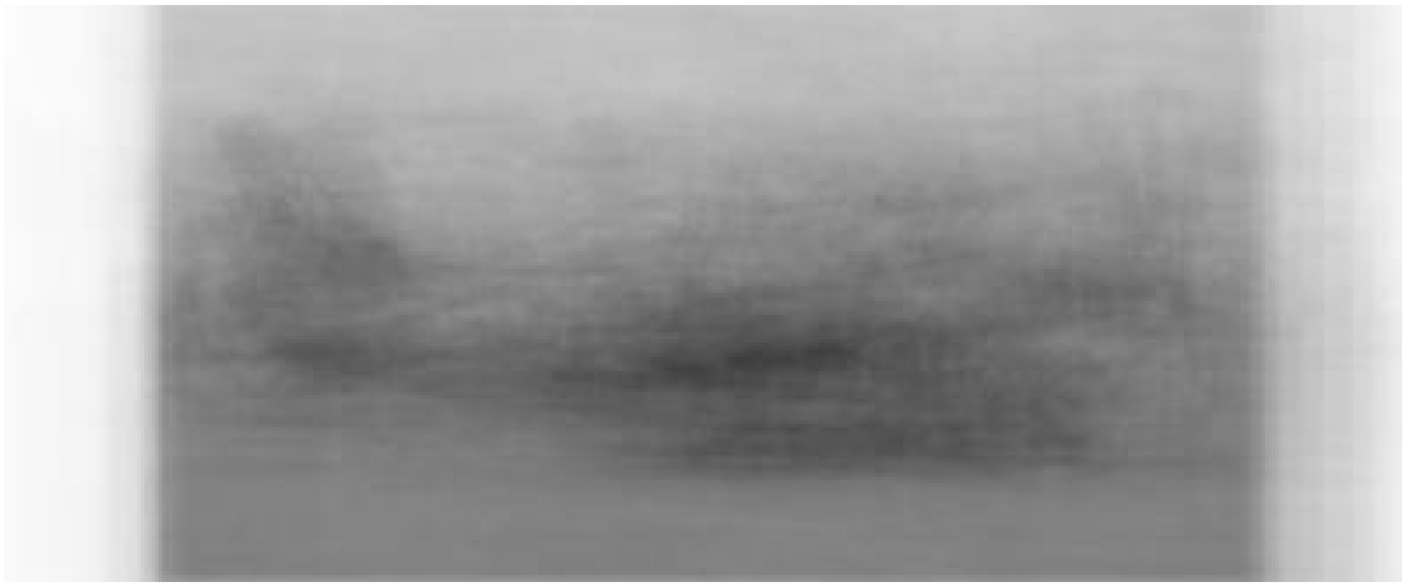

Figure 12: Part A, average grayscale image

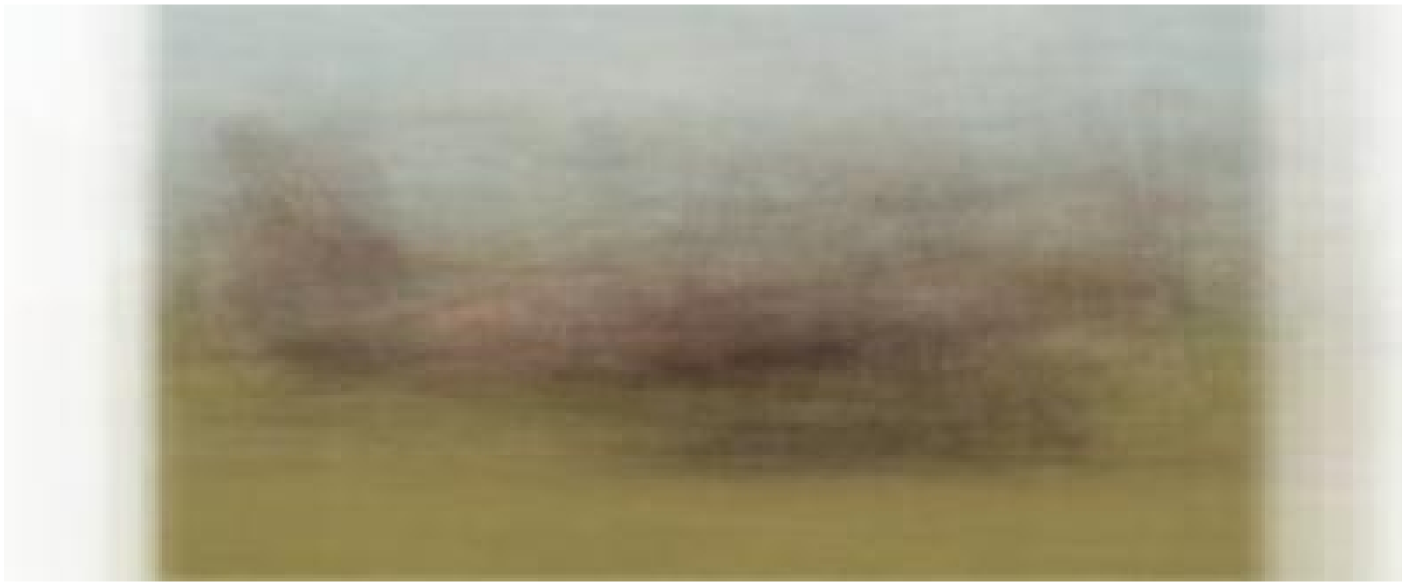

Figure 13: Part B, average color image

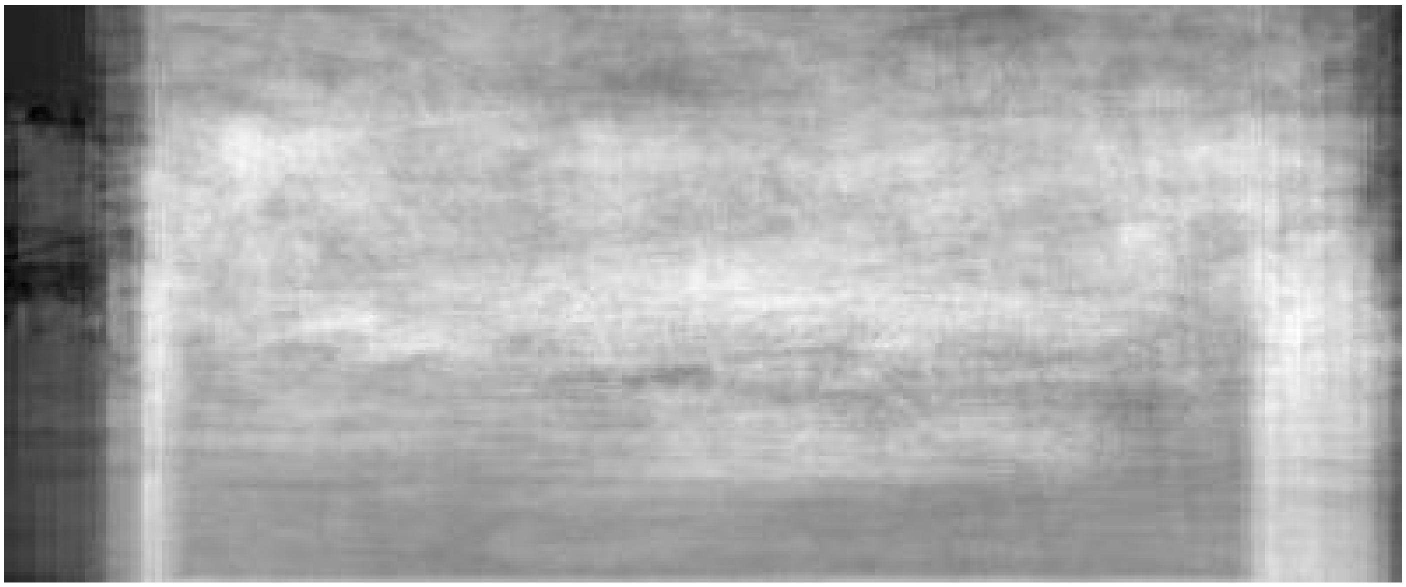

Figure 14: Part C, standard deviation image

Explanation of Results

The image set $set1$ is images of sailing ships and they all look relatively similar. Therefore, the average color image and the average grayscale image looks like a sailing ship. You can definitely notice the shape of a sailing ship in the average image. In the color image it is easier to notice this due to the fact that there is more detail. The standard deviation image is darker in the background and lighter where the ship is meant to be. This could suggest that the background is often sky so it does not vary as much thus more of the varation is in the shape of the ship.

The image set $set2$ is images of planes but there is considerable variation in background as well as type of plane. Therefore, the average color image looks vaguely like a plane but the grayscale image does not look nearly as much like a plane. The standard deviation image does not look much like a plane due to the fact that the foreground and background varies considerably between images.