Table of Contents


```
% amalhot5, Image_Segmentation.m, Arnav Malhotra.
% Worked on by myself
```

Project Statement 1

This reads in the image 'hopkins1.bmp'. MATLAB views the image as a 3-D matrix, with red, green, and blue being each respective dimension. Using the given equation in the project statement, I converted the image to grayscale.

```
clear all

hopkins1 = imread('hopkins1.bmp');
figure;
imshow(hopkins1)
title('Hopkins1.bmp RGB');
R = hopkins1(:,:,1);
G = hopkins1(:,:,2);
B = hopkins1(:,:,3);
gray = 0.2989 * R + 0.5870 * G + 0.1140 * B;
figure;
imshow(gray)
title('Hopkins1.bme Grayscale');
```

```
        Warning: Image is too big to fit on screen; displaying at 67%
        Warning: Image is too big to fit on screen; displaying at 67%
```

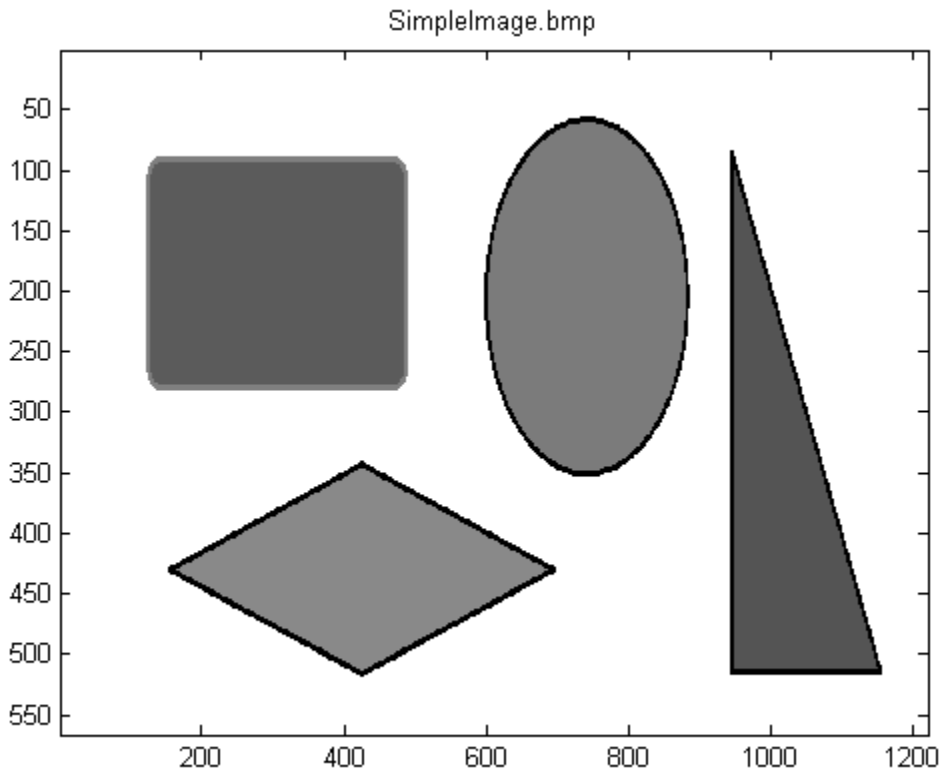
Hopkins1.bmp RGB

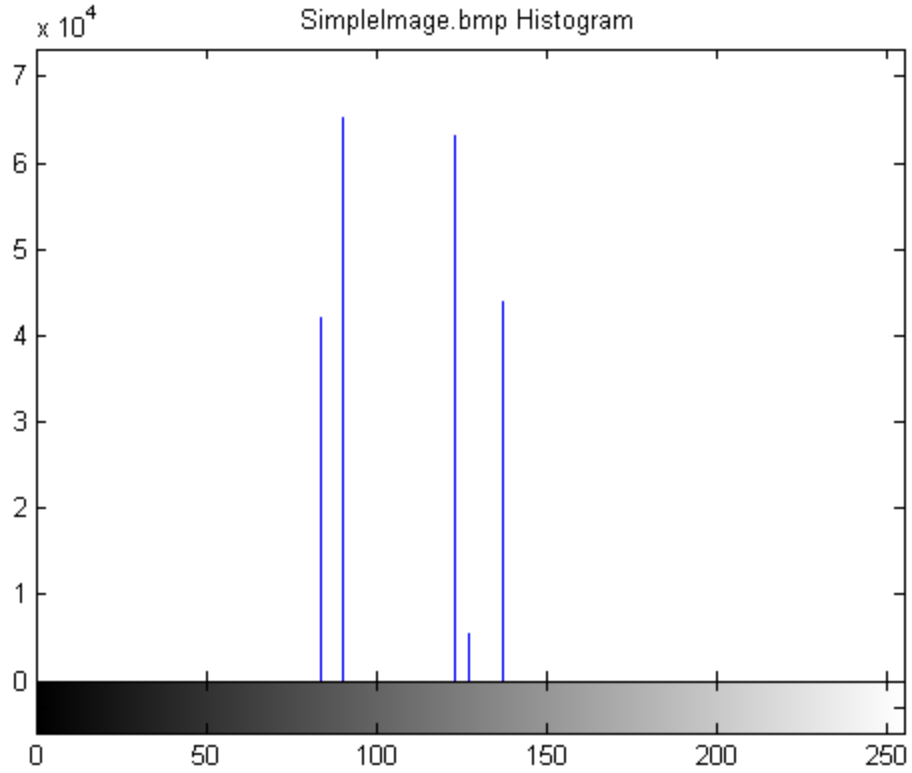
Hopkins1.bme Grayscale

Project Statement 2

Using the built in function 'imhist', I created a histogram for the image. 'imhist' only works with the grayscale, so first I converted the image to grayscale. The number of bins corresponds to the intensity and the y-axis corresponds to the number of pixels at that intensity

```
clear all

SimpleImage = imread('SimpleImage.bmp');
figure;
image(SimpleImage)
title('SimpleImage.bmp');
R = SimpleImage(:,:,1);
G = SimpleImage(:,:,2);
B = SimpleImage(:,:,3);
gray = 0.2989 * R + 0.5870 * G + 0.1140 * B;
figure;
imhist(gray)
title('SimpleImage.bmp Histogram');
```





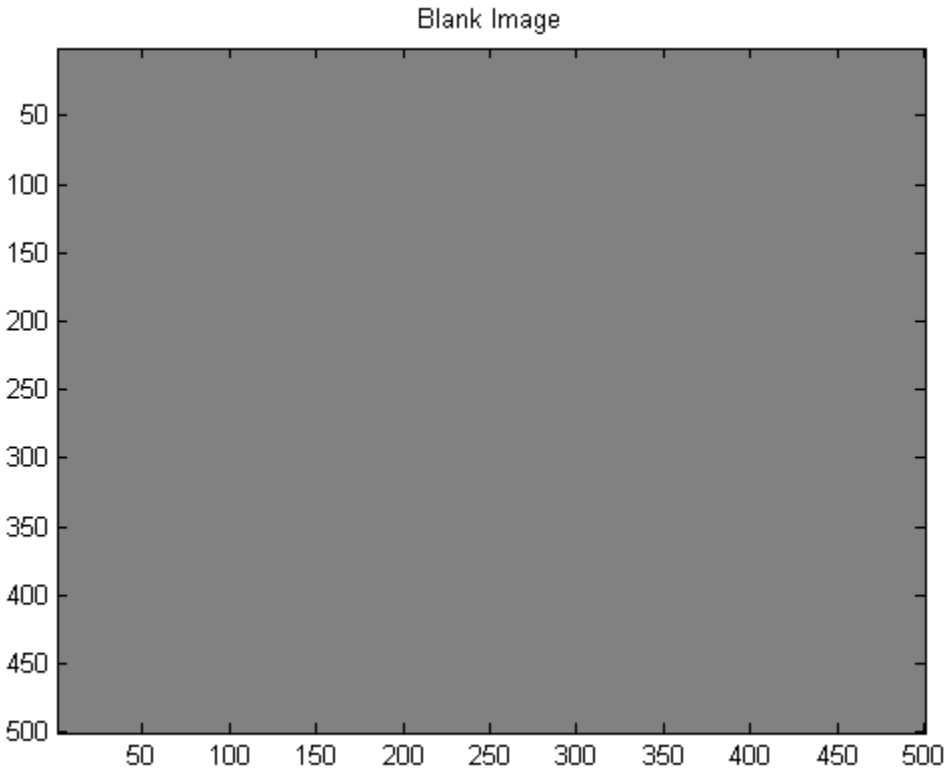
Project statement 3

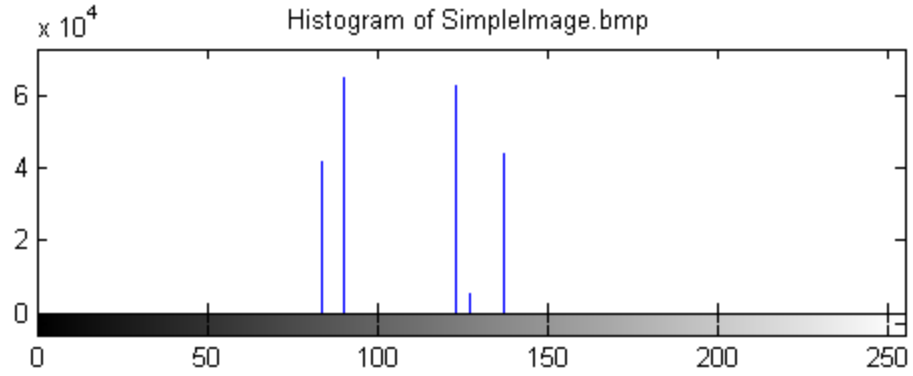
The first part was the same as project statement 1 and 2. Then by looking at the histogram, I decided on the threshold of 150. I then set up a nested for loop to run through all the elements of the grayscale image. Any pixel with an intensity greater than or equal to the threshold is a value of 0 (white) and anything below is a value of 1 (black). Therefore the foreground is in white and the background is in black.

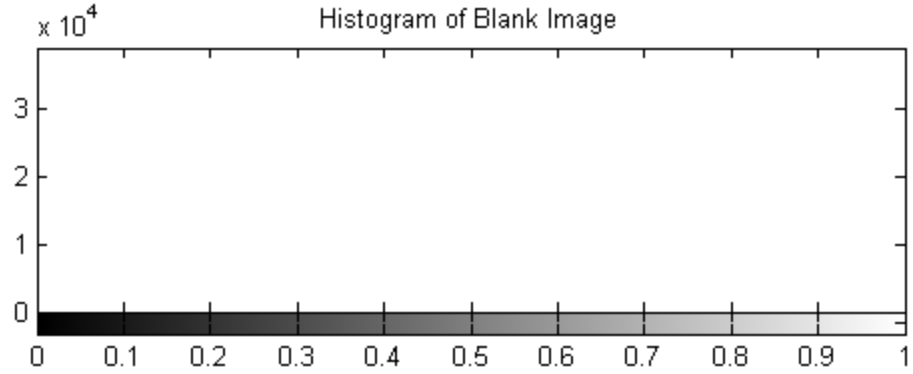
```
clear all

% Read Image, Convert to Grayscale, Display Histogram
% Same as project statement 1 and 2.
SimpleImage = imread('SimpleImage.bmp');
R = SimpleImage(:,:,1);
G = SimpleImage(:,:,2);
B = SimpleImage(:,:,3);
gray = 0.2989 * R + 0.5870 * G + 0.1140 * B;
blankimage = zeros(500);
figure;
colormap('gray');
imagesc(blankimage);
title('Blank Image');
figure;
subplot(2,1,1);
imhist(gray)
title('Histogram of SimpleImage.bmp')
subplot(2,1,2);
```

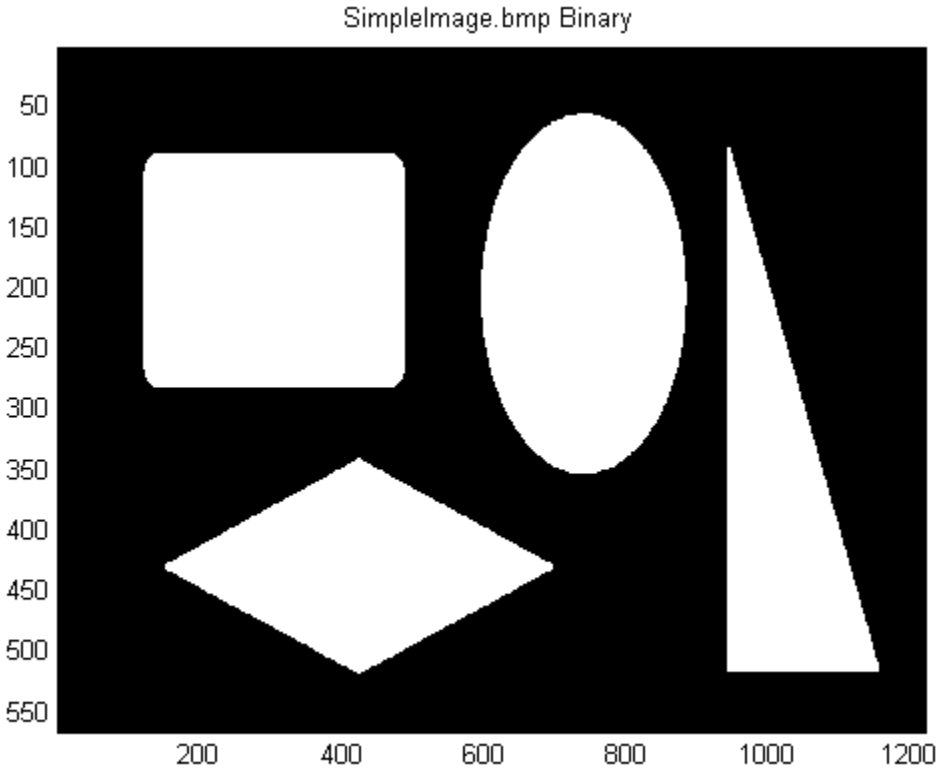
```
imhist(blankimage)
title('Histogram of Blank Image');
threshold = 150
% Segment Image Based on Threshold
s=size(gray);
for i=[1:s(1,1)]
    for j = [1:s(1,2)]
        if gray(i,j)>=threshold
            gray(i,j)=0;
        else
            gray(i,j)=1;
        end
    end
end
figure;
colormap('gray');
imagesc(gray); % Image not displayed unless this is done
title('SimpleImage.bmp Binary');
```

```
threshold =

   150
```




Histogram of SimpleImage.bmp
x 10^4
6
4
2
0
0
50
100
150
200
250


Histogram of Blank Image
x 10^4
3
2
1
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


SimpleImage.bmp Binary
50
100
150
200
250
300
350
400
450
500
550
200
400
600
800
1000
1200

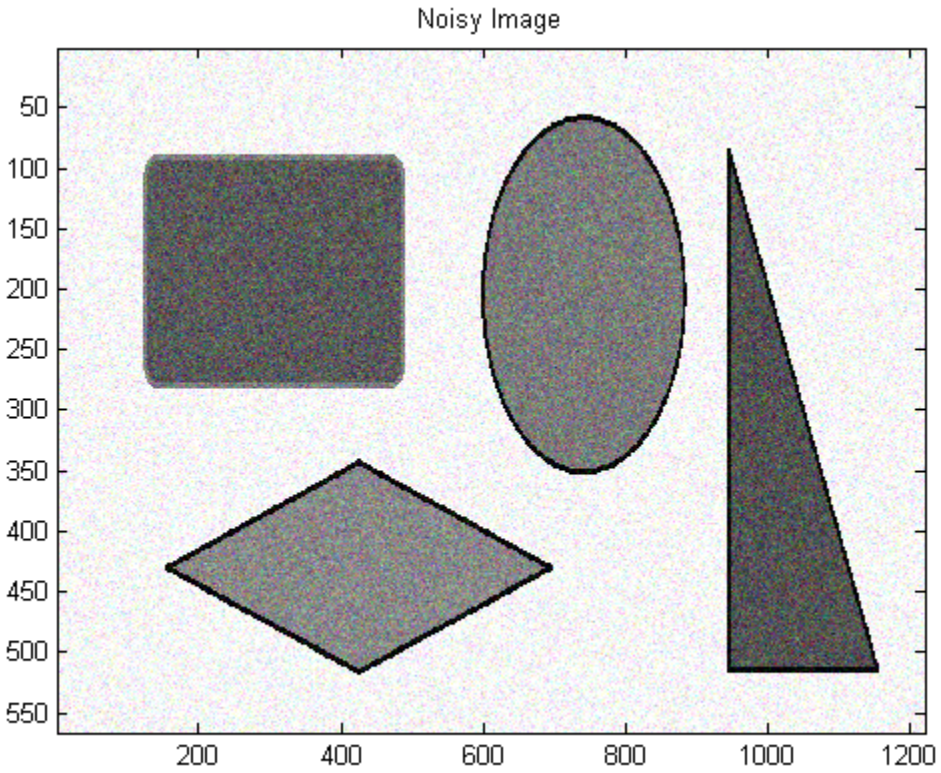
Project Statement 4

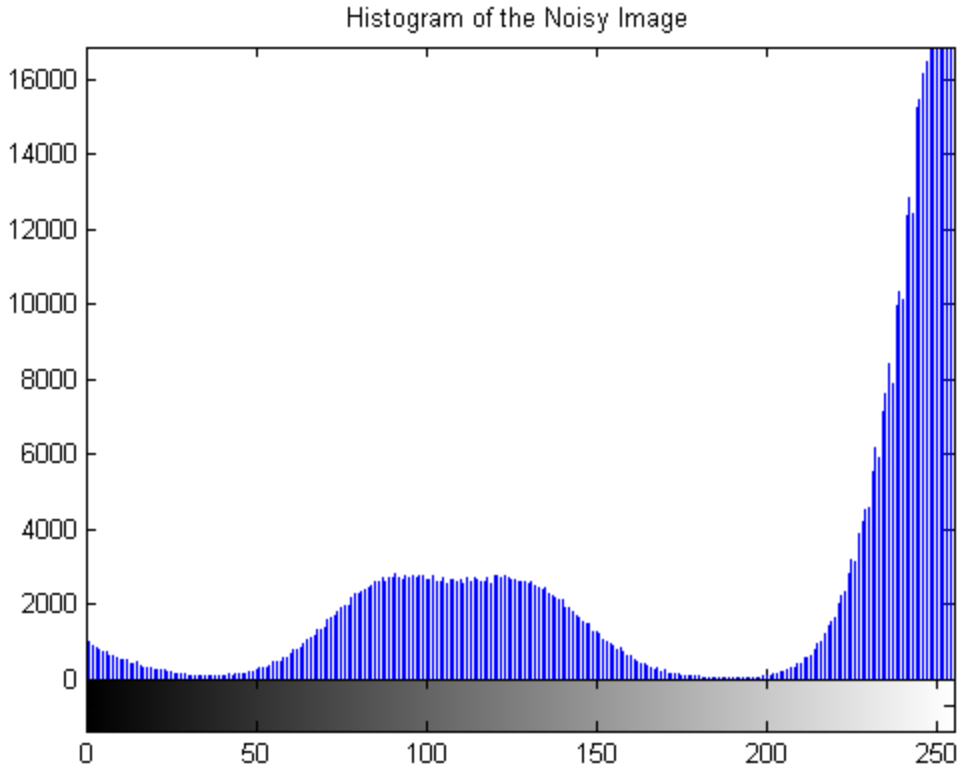
The first part is the same as before. The last part is the same as #3. I set an initial "guess" threshold at 150. Anything above the threshold is the foreground and anything below it is the background. Using the 'fit' function, I fit a Gaussian to the the background. The foreground needed a folded gaussian (Gaussian of Type 2), so that was the one that was fitted. Since the find doesn't work for fit type objects, I had to devise a way to find the intersect. The way I did this was I plotted all the x and y values for the range of the histogram (i.e. x = [1:256]) and then subtracted the two. The minimum value of that was found (since sometimes the intercept would not be an integer value), and the corresponding x value would be the intersect and the new threshold. However this method had some errors. Sometimes it would ocsillate between 2 numbers, therefore I put in a count to break out after a certain number of iterations. The other error was that sometimes the intersect would be 1 and that would not allow it to calculate the y values, therefore I said if the intersect is ever below 100, it would restart the loop.

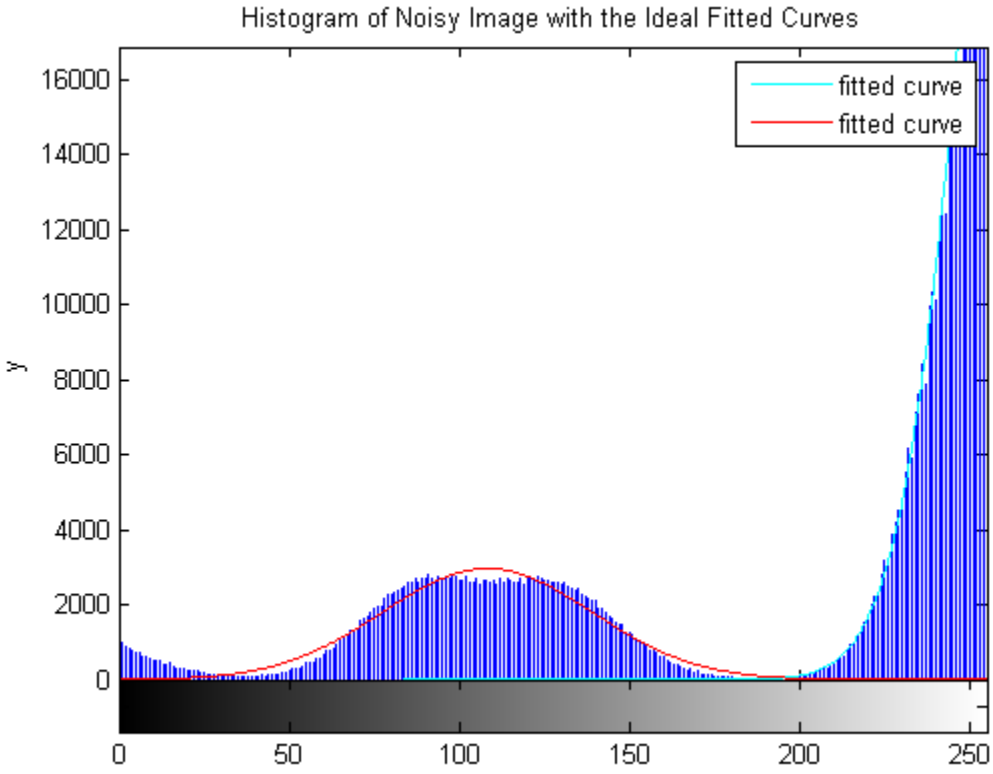
```
clear all
% Read Image, Add Gaussian Noise, Display Histogram
% Same as project statement 1 and 2.
SimpleImage = imread('SimpleImage.bmp');
noise = imnoise(SimpleImage, 'gaussian');
figure;
imagesc(noise)
title('Noisy Image')
R = noise(:,:,1);
G = noise(:,:,2);
B = noise(:,:,3);
gray = 0.2989 * R + 0.5870 * G + 0.1140 * B;
figure;
imhist(gray)
title('Histogram of the Noisy Image');

% Find Ideal Threshold
threshold = 150;
new_threshold = 0;
[pixelCount, intensity] = imhist(gray);
count = 0;
while threshold ~= new_threshold
    background = pixelCount(1:threshold);
    foreground = pixelCount(threshold+1:256);
    g_b = fit(intensity(1:threshold),background,'gauss1');
    g_f = fit(intensity(threshold+1:256),foreground,'gauss2');
    y1 = transpose(feval(g_b,0:255));
    y2 = transpose(feval(g_f,0:255));
    x = transpose(intensity);
    z = abs(y1-y2); % to find the intersection (matlab can't process 'cfit' object
    intersect = find(z == min(z));
    % Since it doesn't intersect at an integer usually, instead of finding
    % the zero, I found the minimum value
    if intersect < 100 % Found error, sometimes it finds the intersect to
    % be 1 so in order to correct that
        threshold = 150; % anytime it is less than 100, we restart
    elseif intersect == threshold
        new_threshold = intersect;
        break
```

```
    else
        threshold = intersect;
    end
    count = count + 1;
    if count == 11 % Another error: sometimes it oscillates between two numbers
        threshold = intersect;
        break
    end
end
threshold
figure;
imhist(gray)
hold on
plot(g_f,'c')
plot(g_b)
hold off
title('Histogram of Noisy Image with the Ideal Fitted Curves')
```

```
threshold =

   197
```



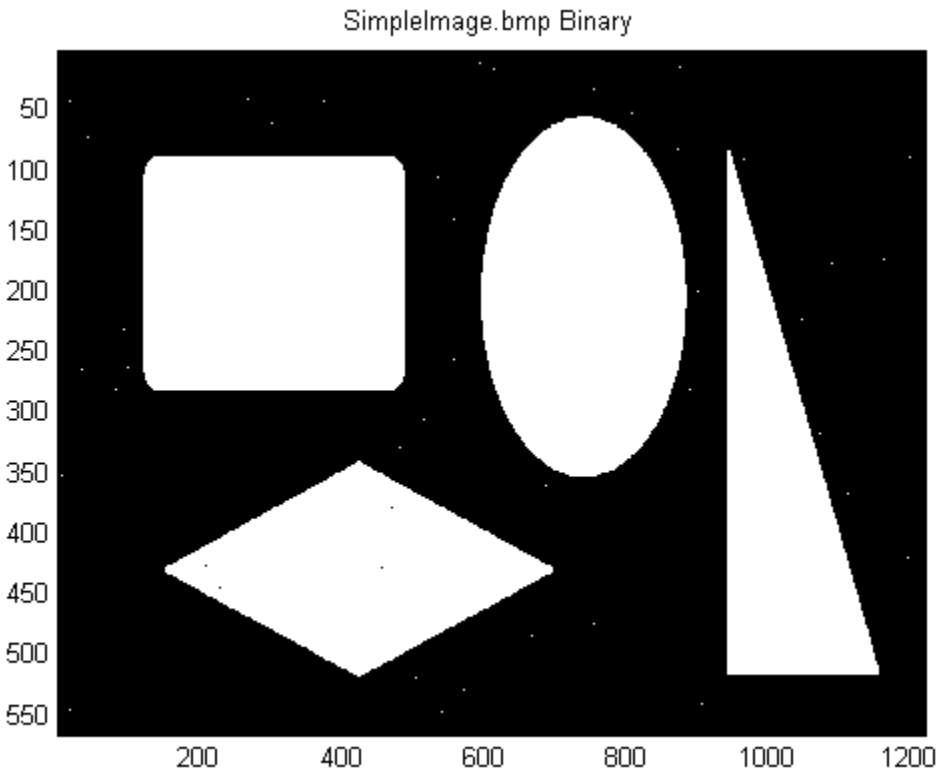

Histogram of the Noisy Image
16000
14000
12000
10000
8000
6000
4000
2000
0
0
50
100
150
200
250


Histogram of Noisy Image with the Ideal Fitted Curves
fitted curve
fitted curve
y
16000
14000
12000
10000
8000
6000
4000
2000
0
0
50
100
150
200
250

Threshold Image Based on Found Threshold

Same as project statement 3

```
s=size(gray);
for i=[1:s(1,1)]
    for j = [1:s(1,2)]
        if gray(i,j)>=threshold
            gray(i,j)=0;
        else
            gray(i,j)=1;
        end
    end
end
figure;
colormap('gray');
imagesc(gray);
title('SimpleImage.bmp Binary');
```



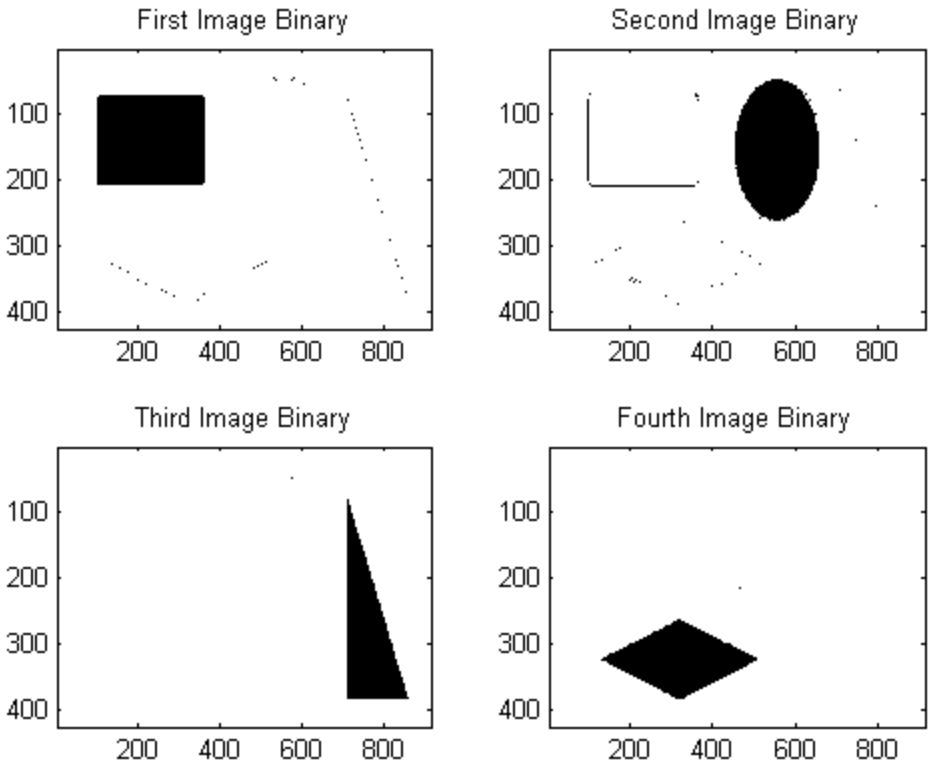
Challenge 1

Using the function 'findpeaks', I found the local maxima in the histogram. I thresholded it such that each image would only display the intensity of that peak and its neighbourhood. With that I found each object and displayed it by itself on an image.

```
clear all
```

```
% Read Image and Convert to Grayscale
SimpleObjects = imread('SimpleObjects.bmp');
R = SimpleObjects(:,:,1);
G = SimpleObjects(:,:,2);
B = SimpleObjects(:,:,3);
gray = 0.2989 * R + 0.5870 * G + 0.1140 * B;

% Find Peaks in Histogram
[pixelCount, intensity] = imhist(gray);
w = ismember(pixelCount,findpeaks(pixelCount,'MinPeakHeight', 1000));
loc = find(w);

% Threshold Image Based on Peaks
s=size(gray);
fig1 = gray;
for i=[1:s(1,1)]
    for j = [1:s(1,2)]
        if fig1(i,j)>=loc(1) & fig1(i,j) < loc(2)
            fig1(i,j)=0;
        else
            fig1(i,j)=1;
        end
    end
end
figure;
subplot(2,2,1)
colormap('gray');
imagesc(fig1);
title('First Image Binary');

s = size(gray);
fig2 = gray;
for i=[1:s(1,1)]
    for j = [1:s(1,2)]
        if fig2(i,j)>=loc(2) & fig2(i,j) < loc(3)
            fig2(i,j)=0;
        else
            fig2(i,j)=1;
        end
    end
end
subplot(2,2,2)
colormap('gray');
imagesc(fig2);
title('Second Image Binary');

s=size(gray);
fig3 = gray;
for i=[1:s(1,1)]
    for j = [1:s(1,2)]
        if fig3(i,j)>=loc(1)-1 & fig3(i,j) <= loc(1)
            fig3(i,j)=0;
        else
```

```
            fig3(i,j)=1;
        end
    end
end
subplot(2,2,3)
colormap('gray');
imagesc(fig3);
title('Third Image Binary');

s=size(gray);
fig4 = gray;
for i=[1:s(1,1)]
    for j = [1:s(1,2)]
        if fig4(i,j)<=loc(4) & fig4(i,j)>=loc(4)-1
            fig4(i,j)=0;
        else
            fig4(i,j)=1;
        end
    end
end
subplot(2,2,4)
colormap('gray');
imagesc(fig4);
title('Fourth Image Binary');
```



Challenge 2

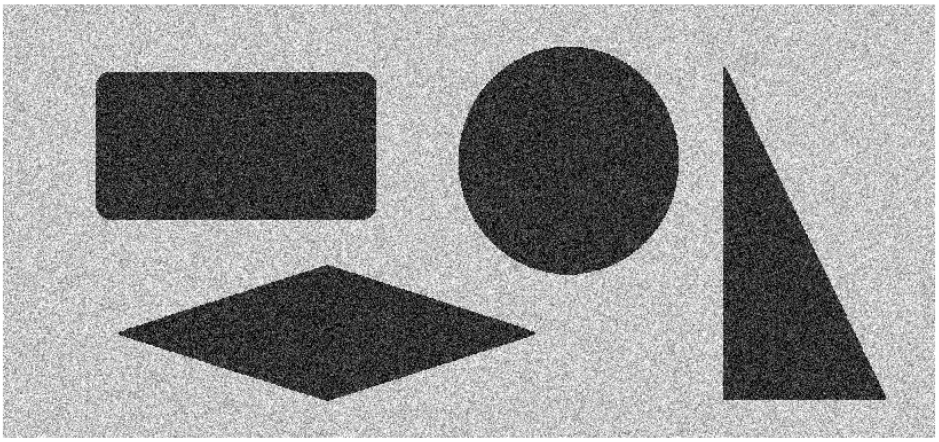
I wasn't sure if there was a hidden image or not, but I just segmented it the same way as in Project Statement 4. The difference was that this time both were fitted with a gaussian curve (no folded gaussian was used) as this gave a better fit.

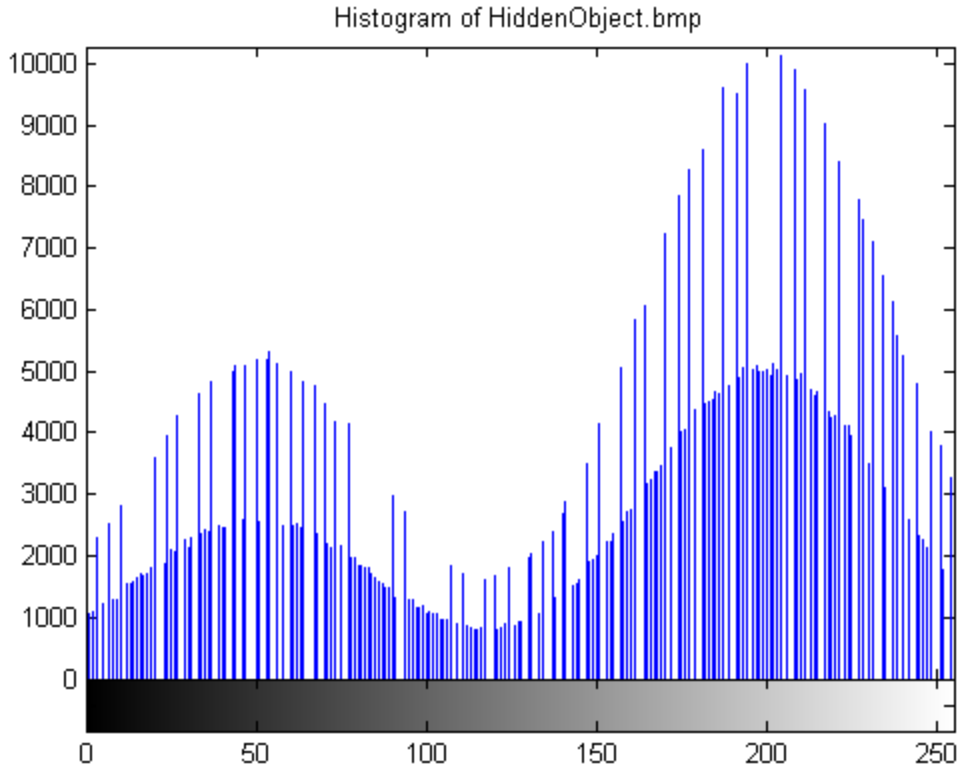
```
clear all
% Read Image, Convert to Grayscale, and Display the Histogram
HiddenObject = imread('HiddenObject.bmp');
figure;
imshow(HiddenObject)
R = HiddenObject(:,:,1);
G = HiddenObject(:,:,2);
B = HiddenObject(:,:,3);
gray = 0.2989 * R + 0.5870 * G + 0.1140 * B;
figure;
imhist(gray)
title('Histogram of HiddenObject.bmp');

% Find Ideal Threshold
threshold = 150;
new_threshold = 0;
[pixelCount, intensity] = imhist(gray);
count = 0;
while threshold ~= new_threshold
    background = pixelCount(1:threshold);
    foreground = pixelCount(threshold+1:256);
    g_b = fit(intensity(1:threshold),background,'gauss1');
    g_f = fit(intensity(threshold+1:256),foreground,'gauss1');
    y1 = transpose(feval(g_b,0:255));
    y2 = transpose(feval(g_f,0:255));
    x = transpose(intensity);
    z = abs(y1-y2);
    intersect = find(z == min(z));
    if intersect < 100
        threshold = 150;
    elseif intersect == threshold
        new_threshold = intersect;
        break
    else
        threshold = intersect;
    end
    count = count + 1;
    if count == 51 % This is because sometimes it oscillates between two numbers
        threshold = intersect;
        break
    end
end
threshold
figure;
imhist(gray)
hold on
plot(g_f,'c')
```

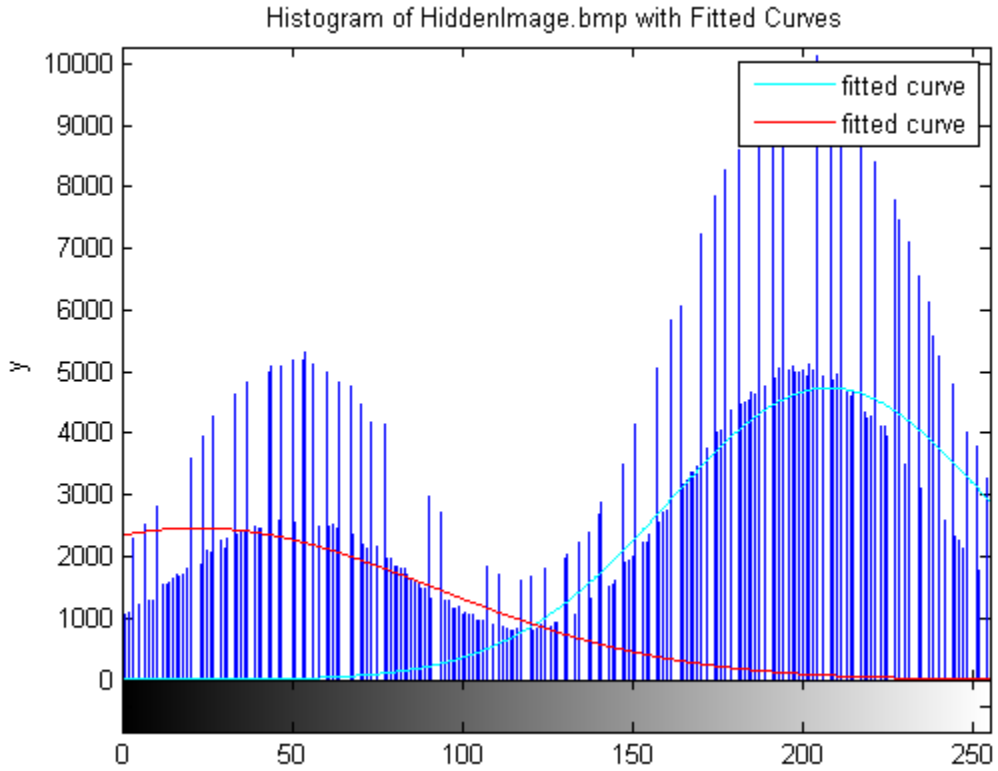
```
plot(g_b)
hold off
title('Histogram of HiddenImage.bmp with Fitted Curves')
% Segment Image
s=size(gray);
for i=[1:s(1,1)]
    for j = [1:s(1,2)]
        if gray(i,j)>=threshold
            gray(i,j)=0;
        else
            gray(i,j)=1;
        end
    end
end
figure;
colormap('gray');
imagesc(gray);
title('SimpleImage.bmp Binary');
```

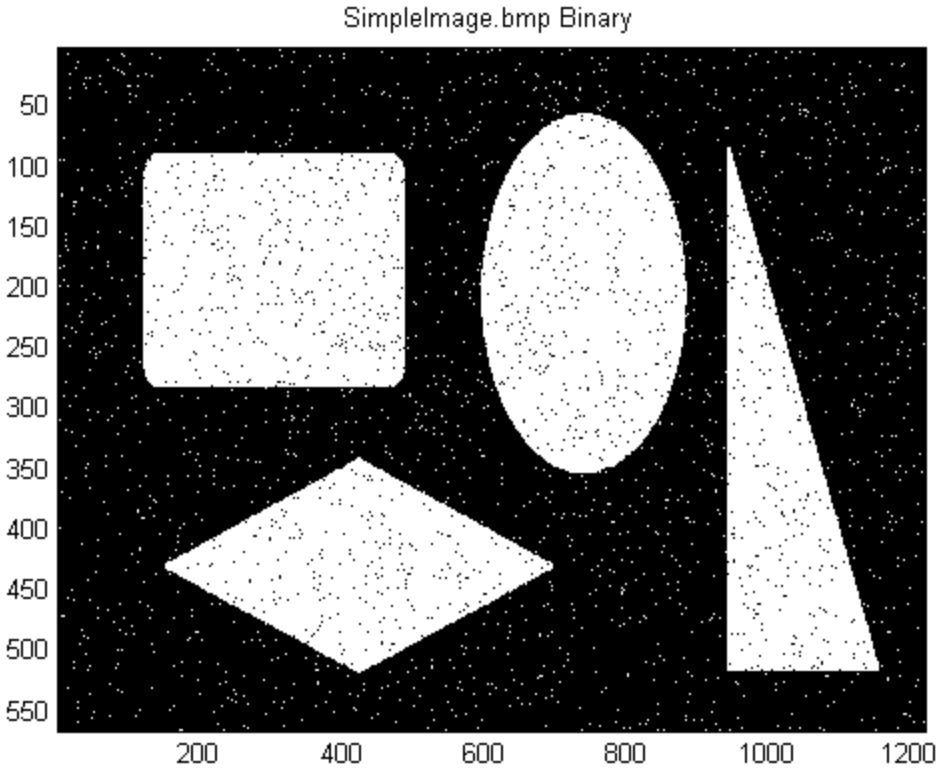
```
Warning: Image is too big to fit on screen; displaying at 67%
```

```
threshold =

   122
```




Histogram of HiddenObject.bmp
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250


Histogram of HiddenImage.bmp with Fitted Curves
fitted curve
fitted curve
y
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
200
250



Published with MATLAB® R2014a