COMPUTER VISION

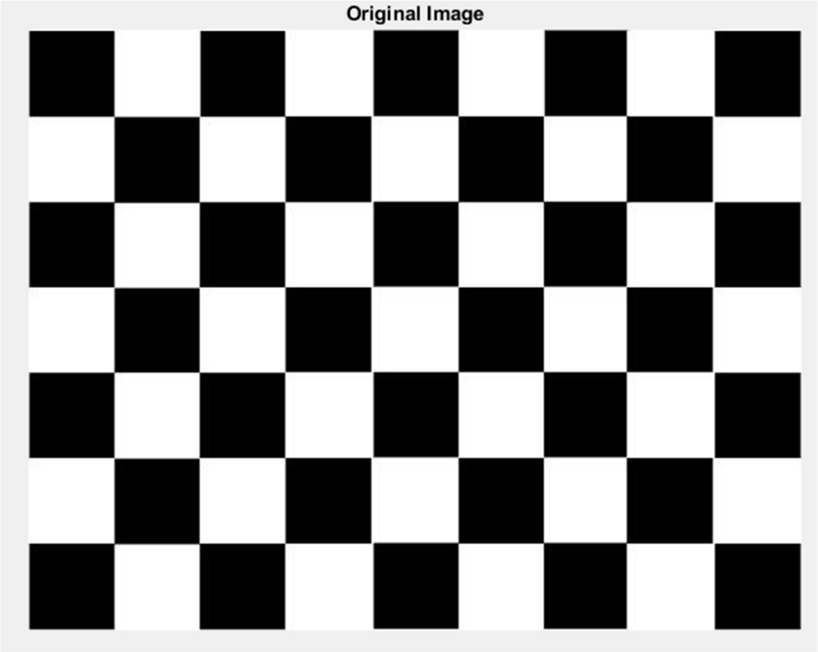
PURVANG LAPSIWALA

UIN = 662689378

Homework 5

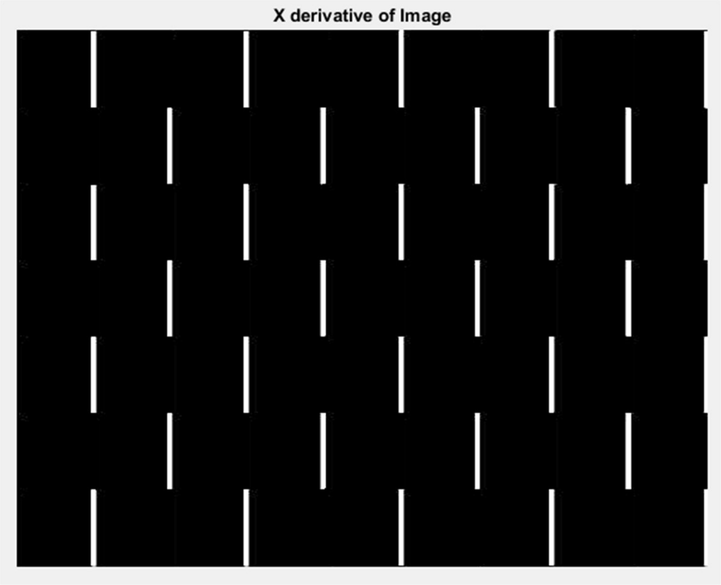
Answer 1 =

1.) Display of original image

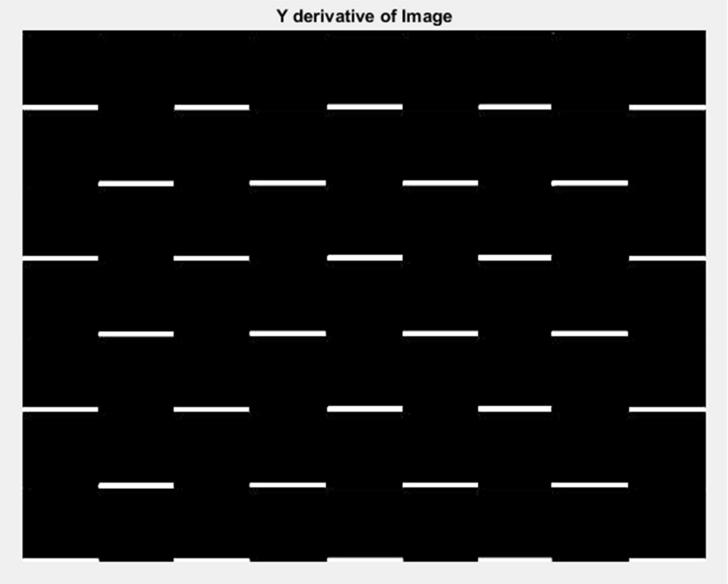


2.)

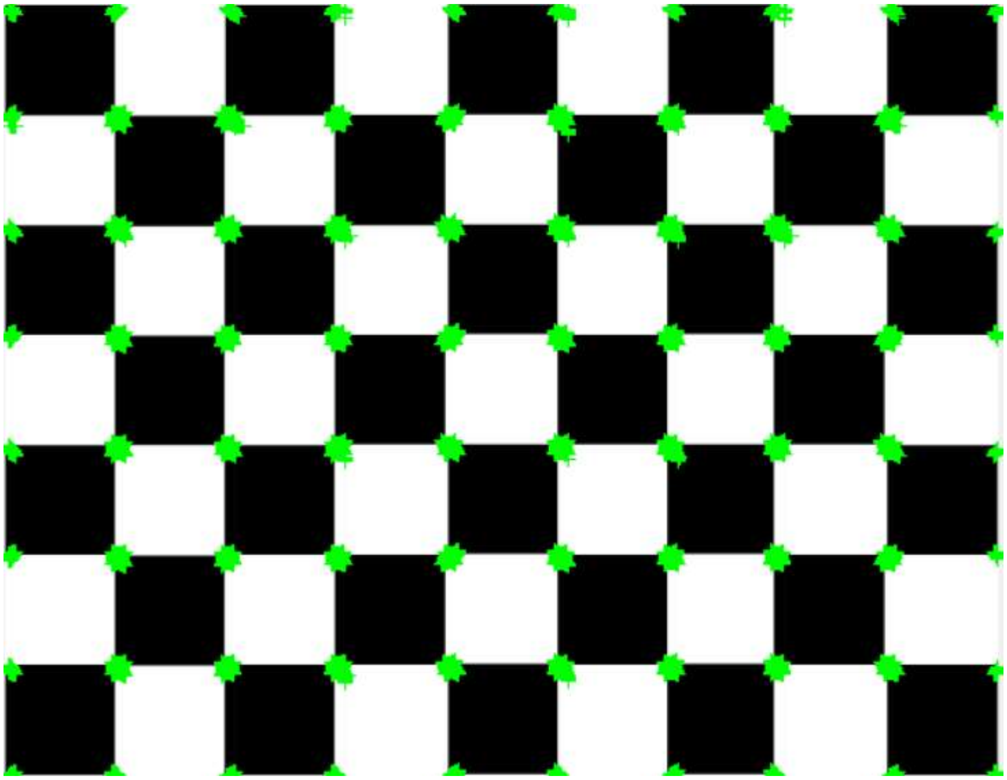
(a) X--derivative of the image



(b) Y- derivative of the image

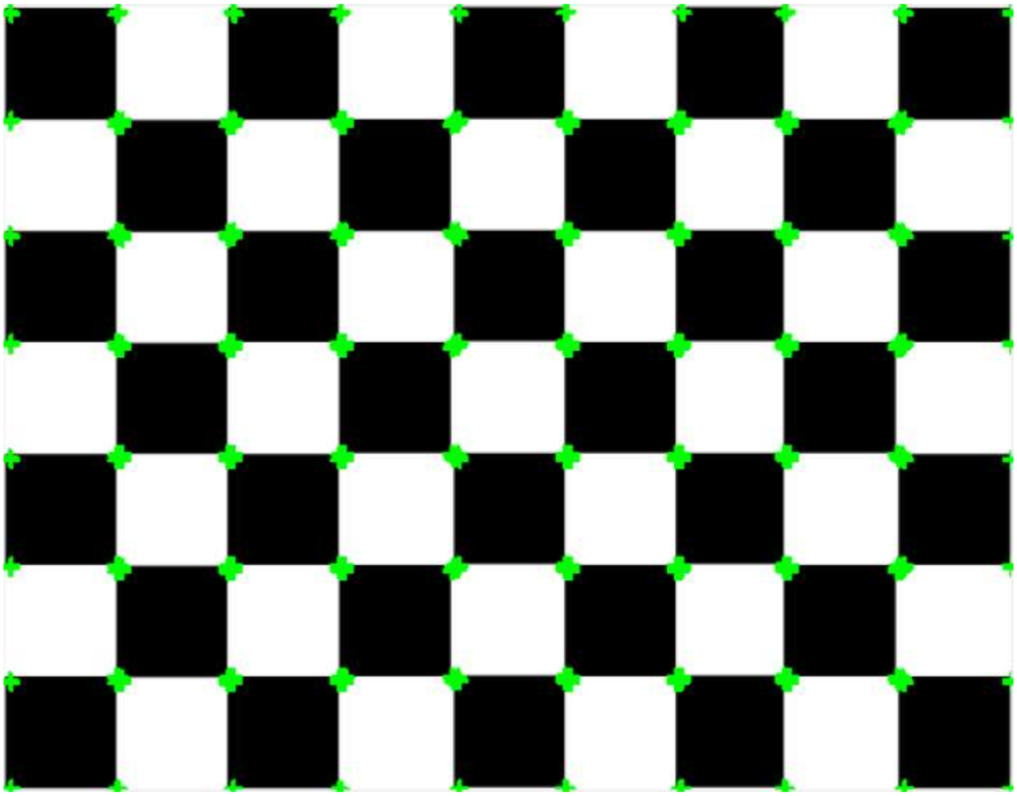


(c) Image with selected features based on value of local maximums Value of threshold used is 85000.



3.)

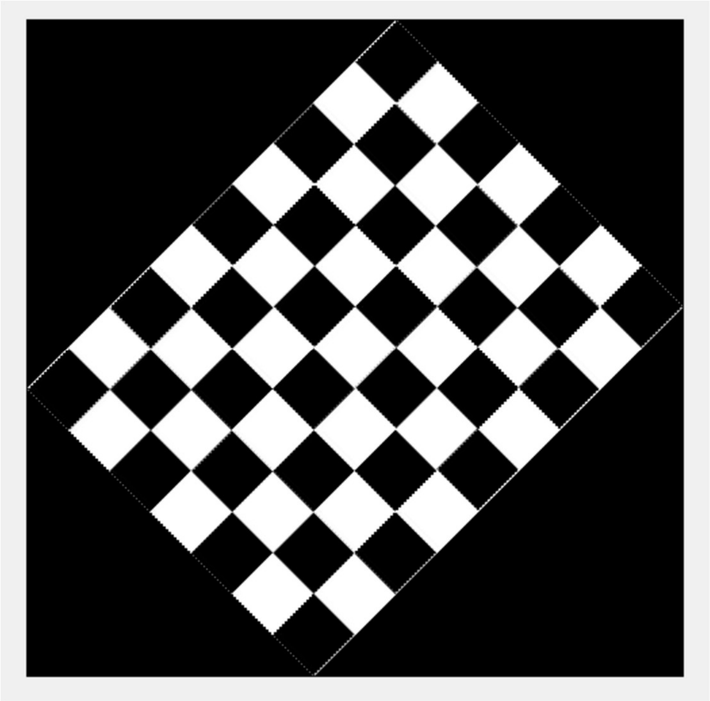
(b) Display of image using adaptive non-maximal suppression



(c)

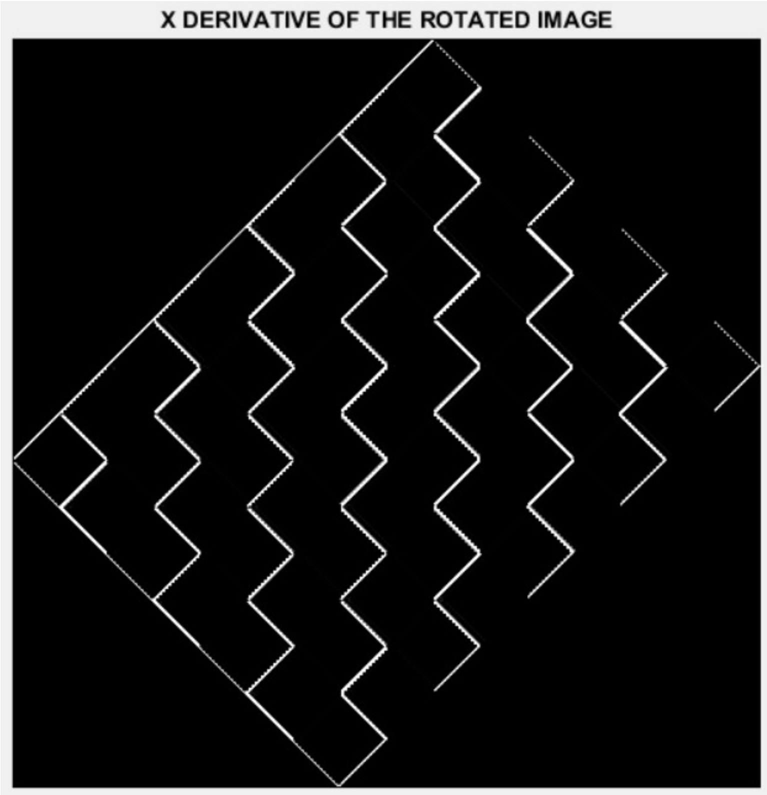
By analyzing both results, adaptive non-maximal suppression method works with the uneven distribution of image in a better way than the Harris Detector. As Harris detector method uses threshold value to find feature, there are high chances of detecting high intensity values in corners or due to darkness, which are not features. This can be eliminated with adaptive method, as it only selects features with certain value, and removes which are not actual features within a selected radius.

4.) Rotated image t0 45 degree.

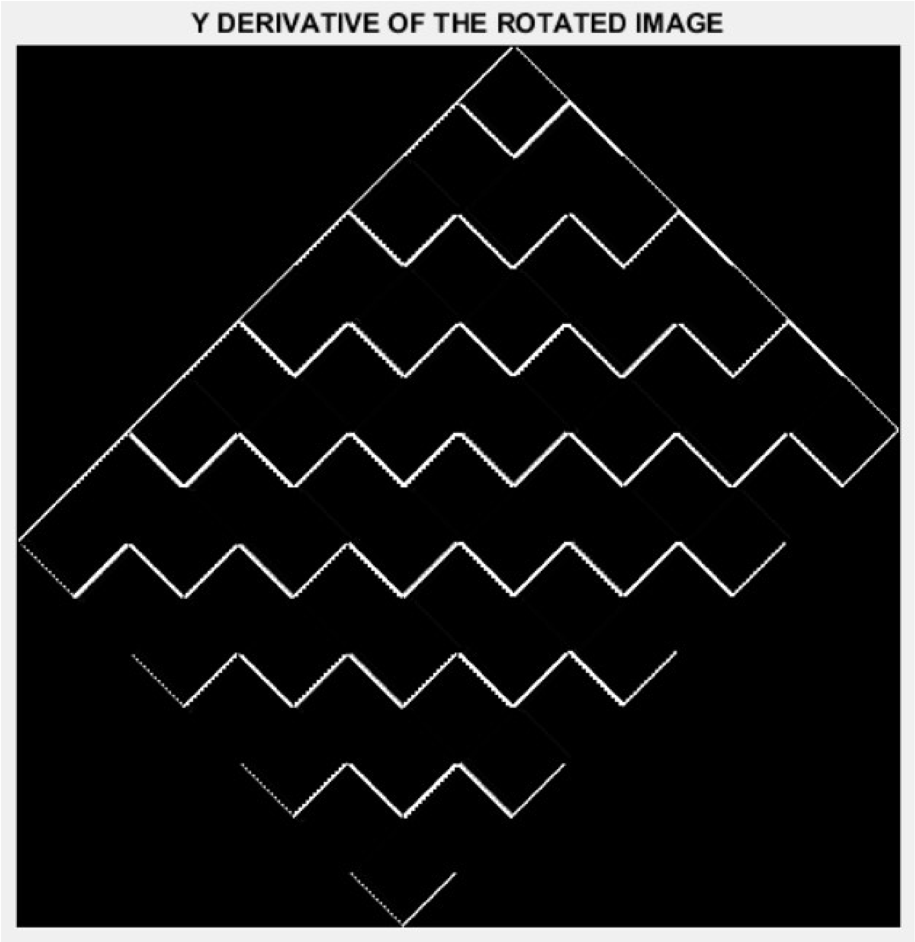


5.)

(a) X- derivative of the image

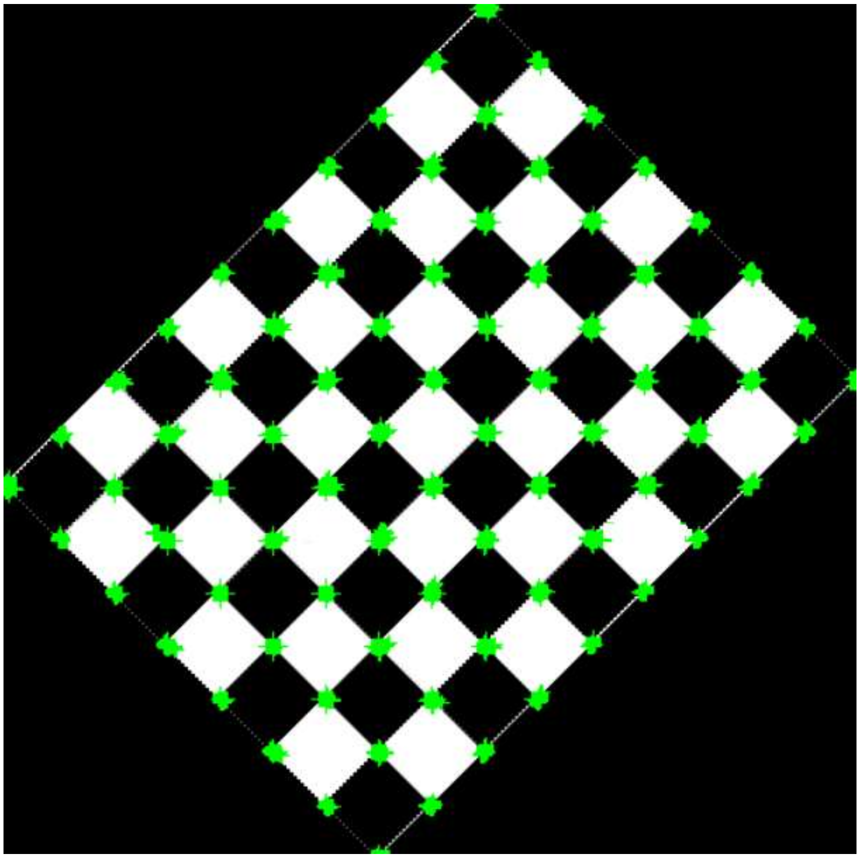


(b) Y-derivative of the rotated image

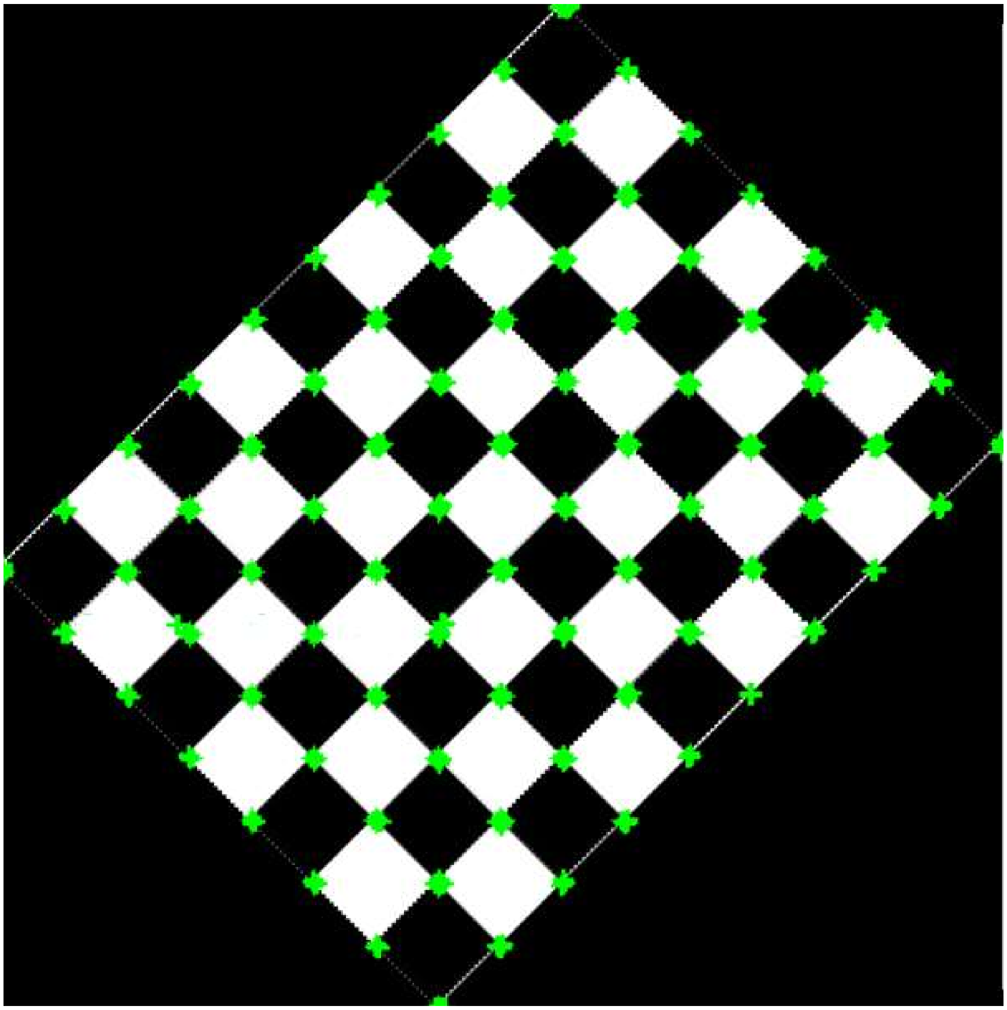


(c) **Marked locations of the selected features on the rotated image**

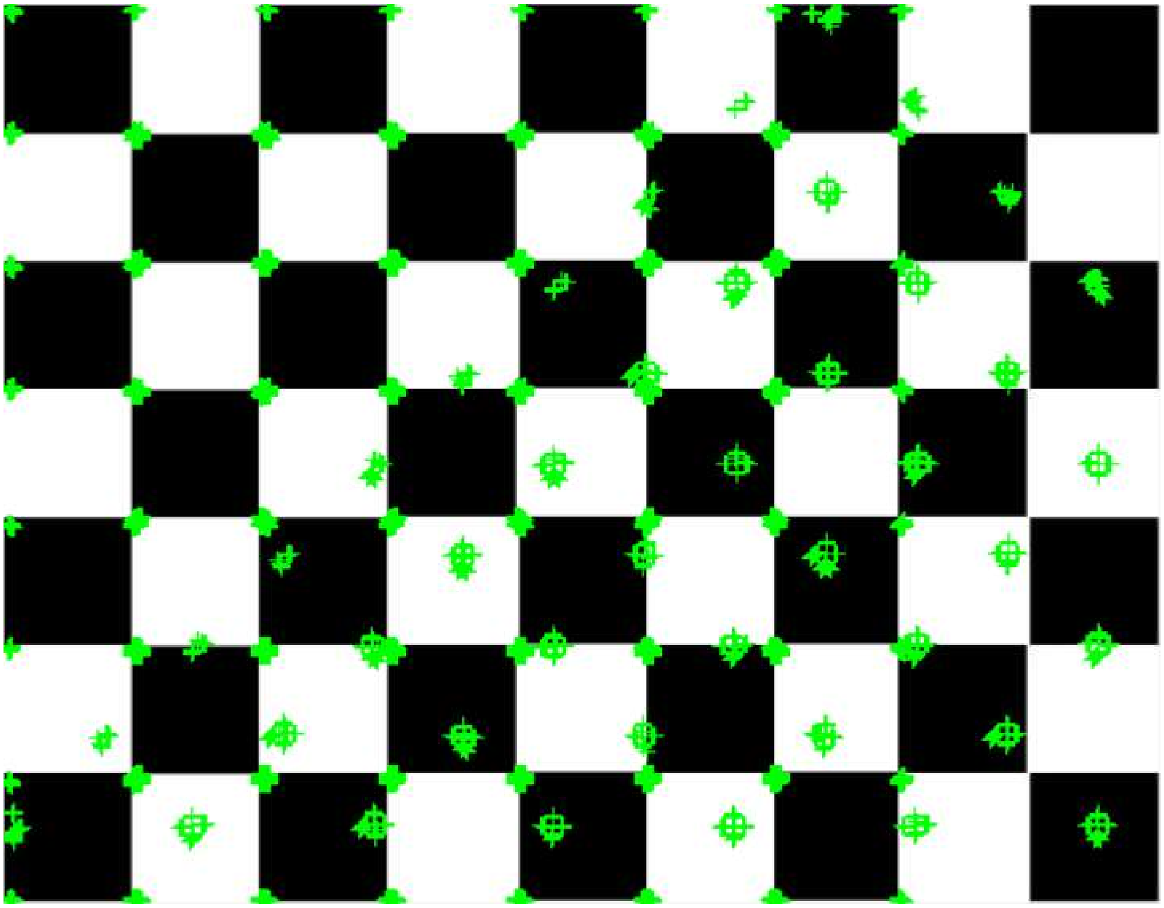
Selected Threshold value is 85000



(d) Display of image using adaptive non-maximal suppression to select features

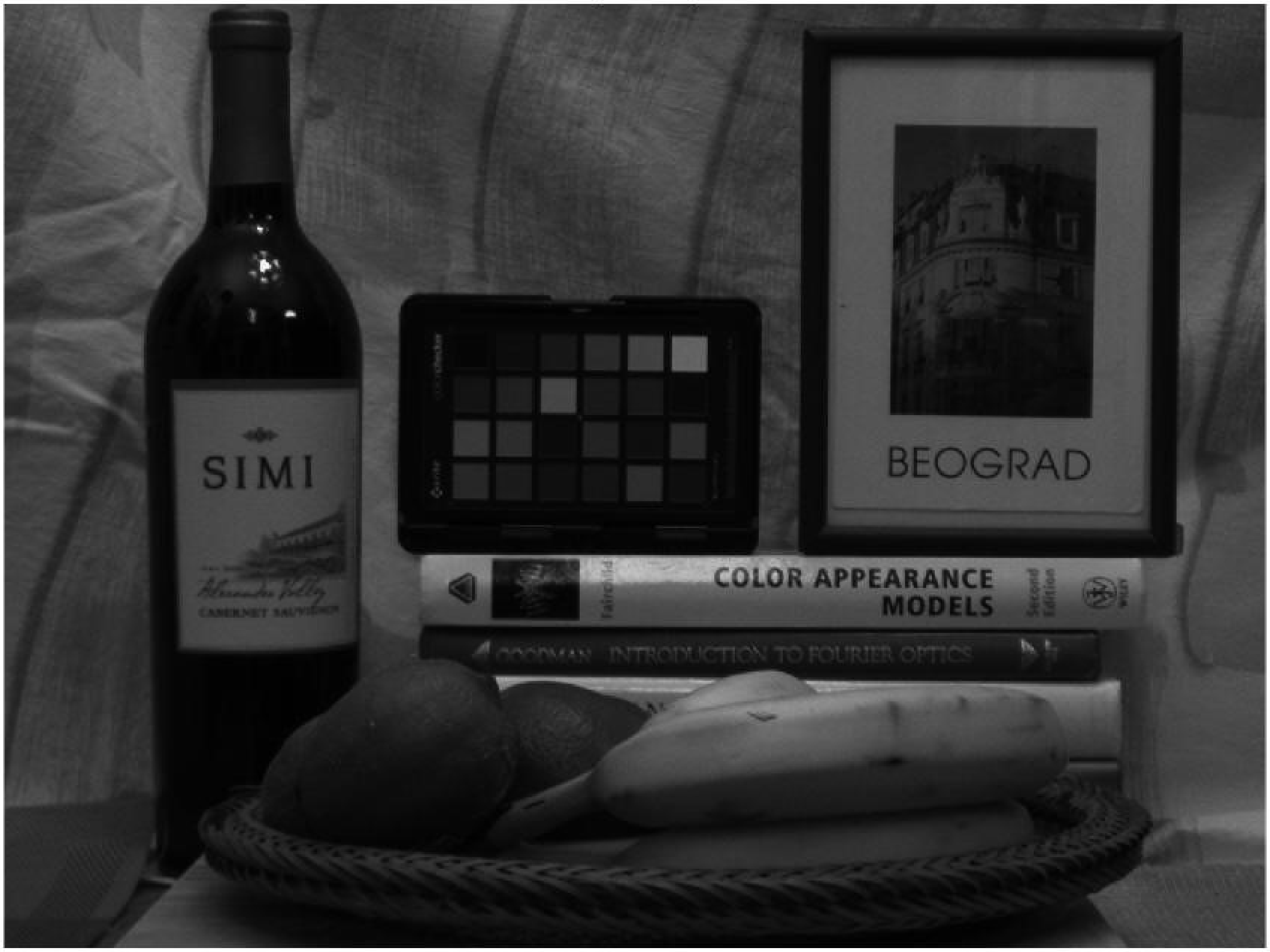


6.) Final image with matched features marked as green dots
A total of 5344 features were matched.



Answer 2 =

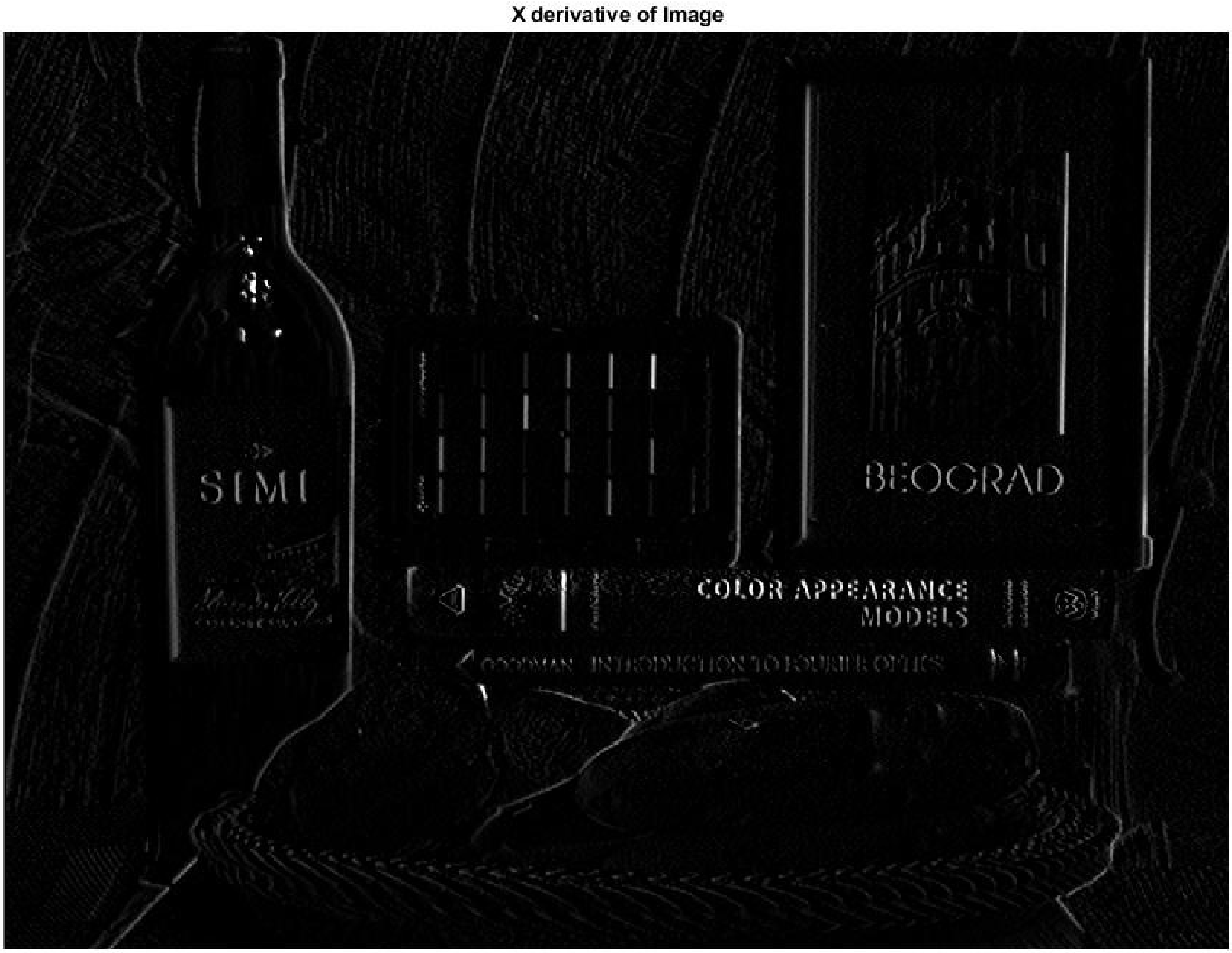
1.) **Display of original image**



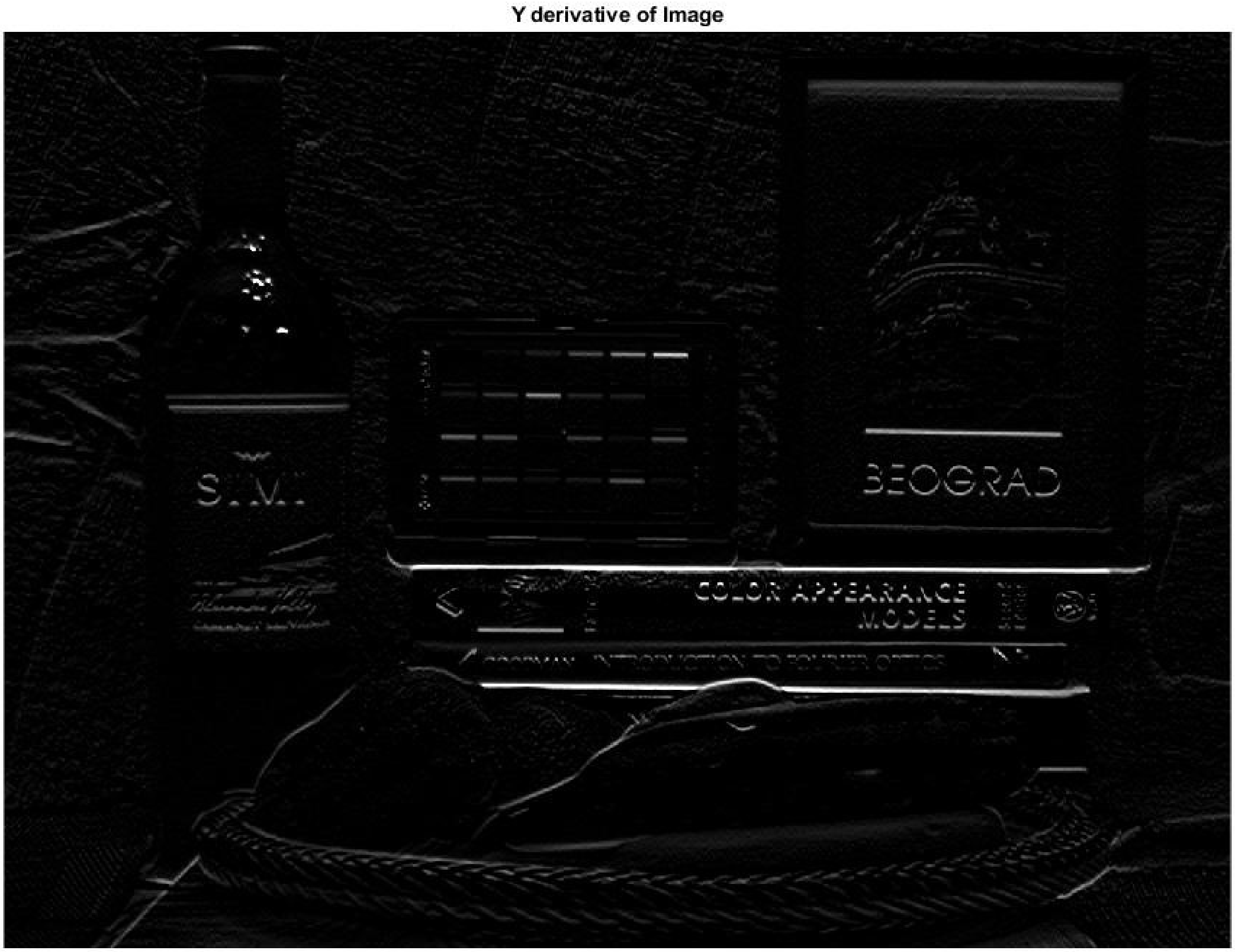
Original Image

2.)

(a) X--derivative of the image

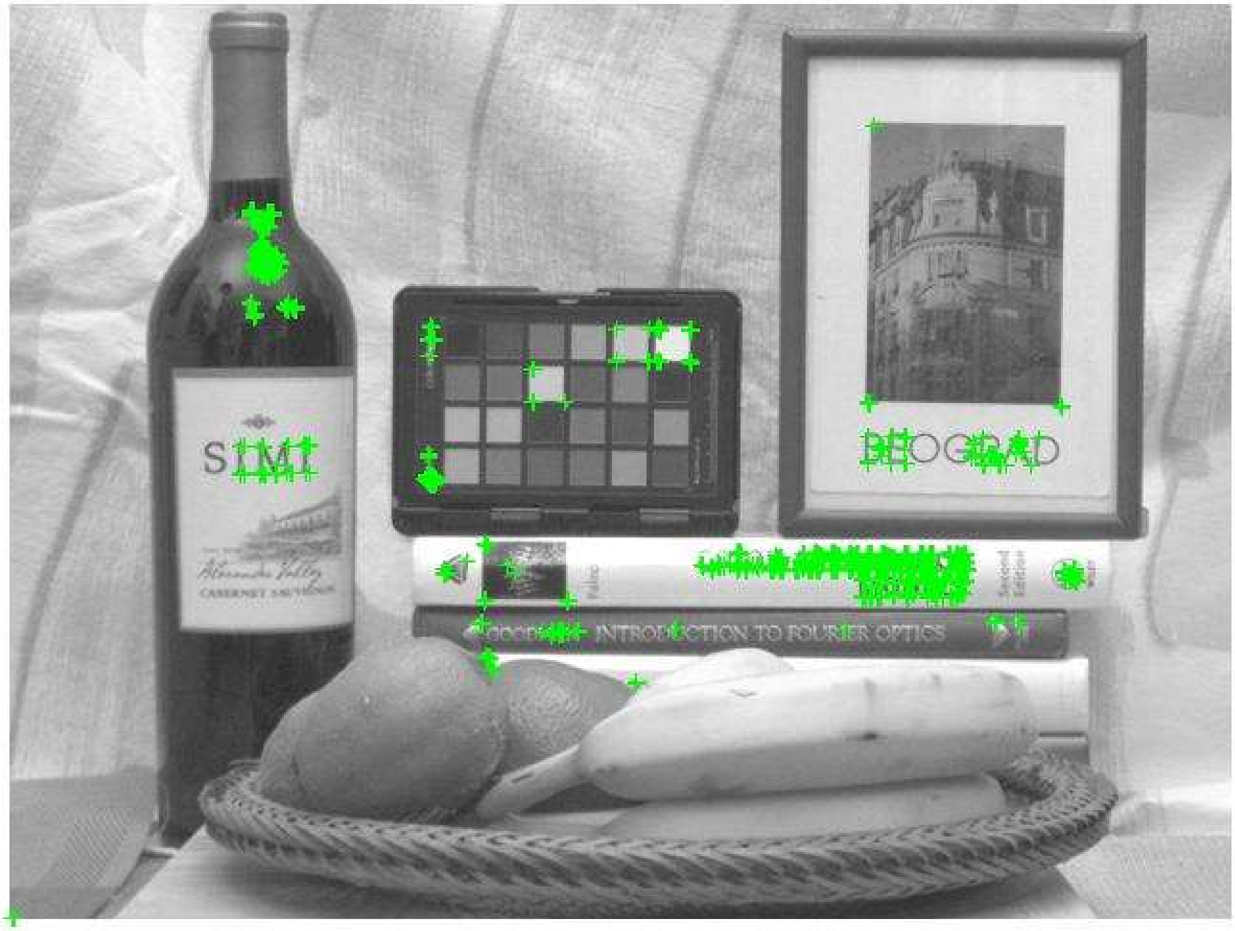


(b) Y- derivative of the image



Y derivative of Image

(c) Image with selected features based on value of local maximums
Value of threshold used is 3000000.



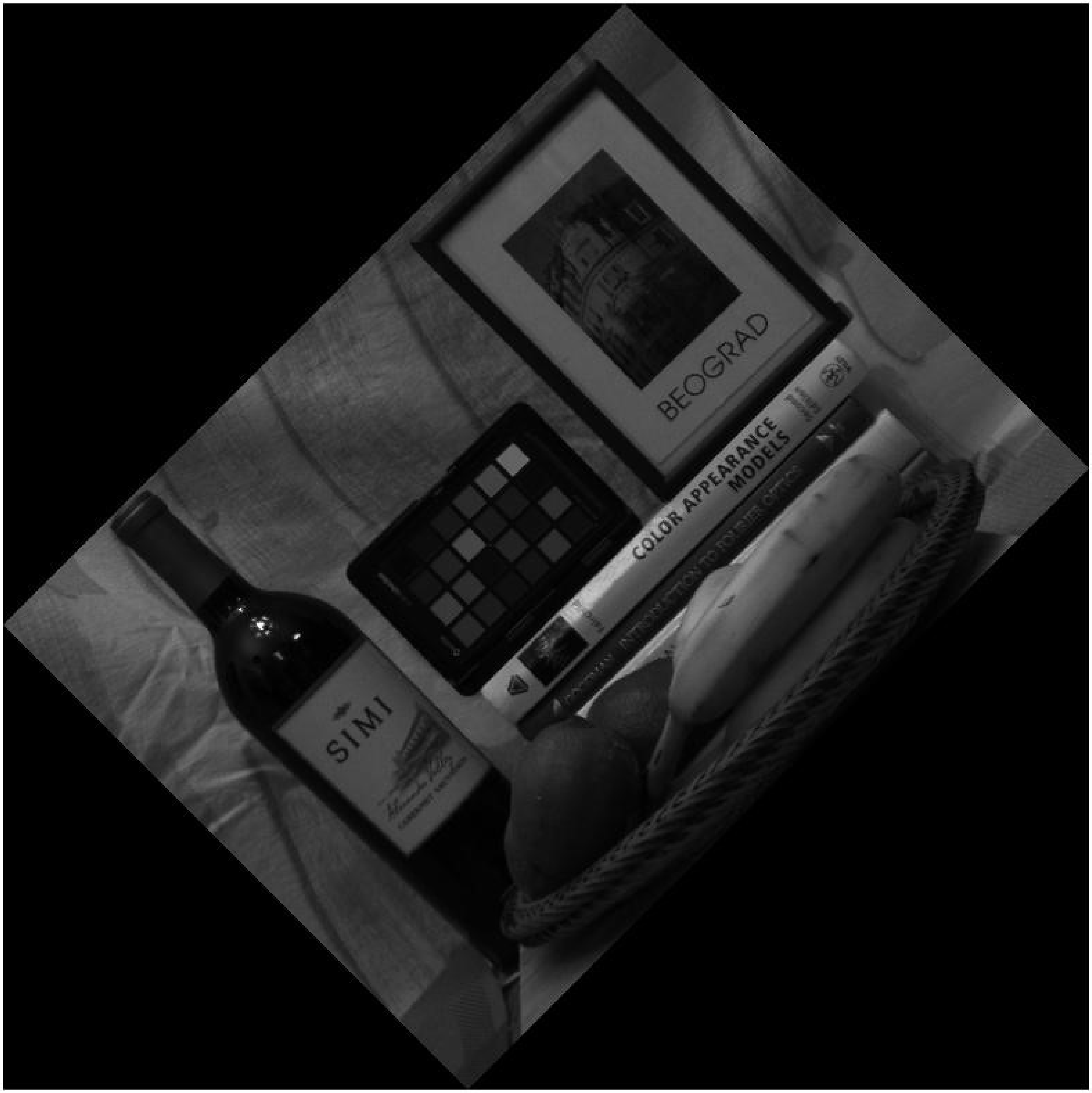
3.)

(b) Display of image using adaptive non-maximal suppression

(c)

By analyzing both results, adaptive non-maximal suppression method works with the uneven distribution of image in a better way than the Harris Detector. As Harris detector method uses threshold value to find feature, there are high chances of detecting high intensity values in corners which are not features. This can be eliminated with adaptive method, as it matches features within a selected radius. As you can see from results that Harris detector detects so many things as features with same threshold value but adaptive method selects only features which have genuine maximum value within certain range.

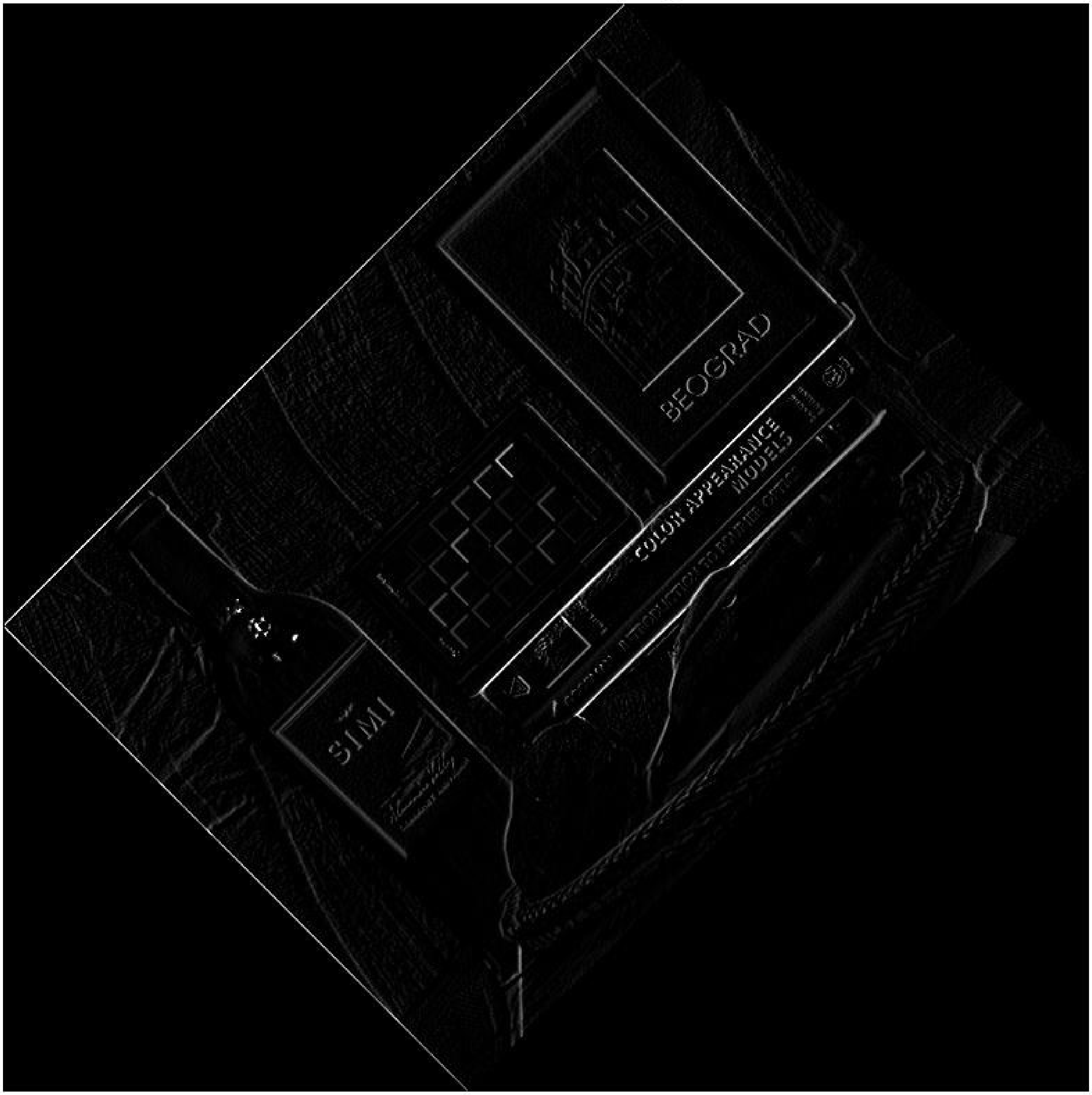
4.) Rotated image to 45 degree.



5.)

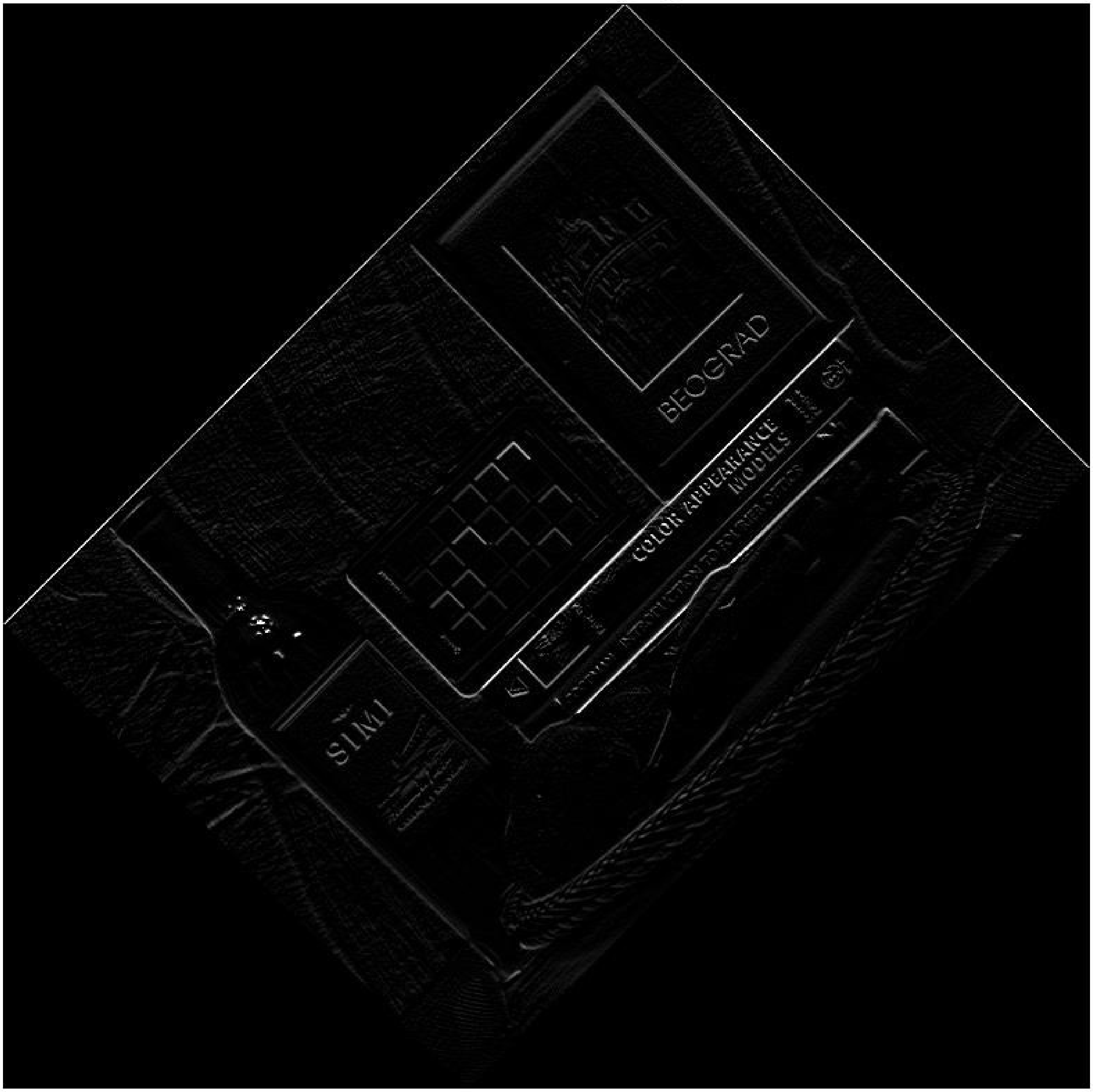
(a) X- derivative of the image

X derivative of Rotated Image

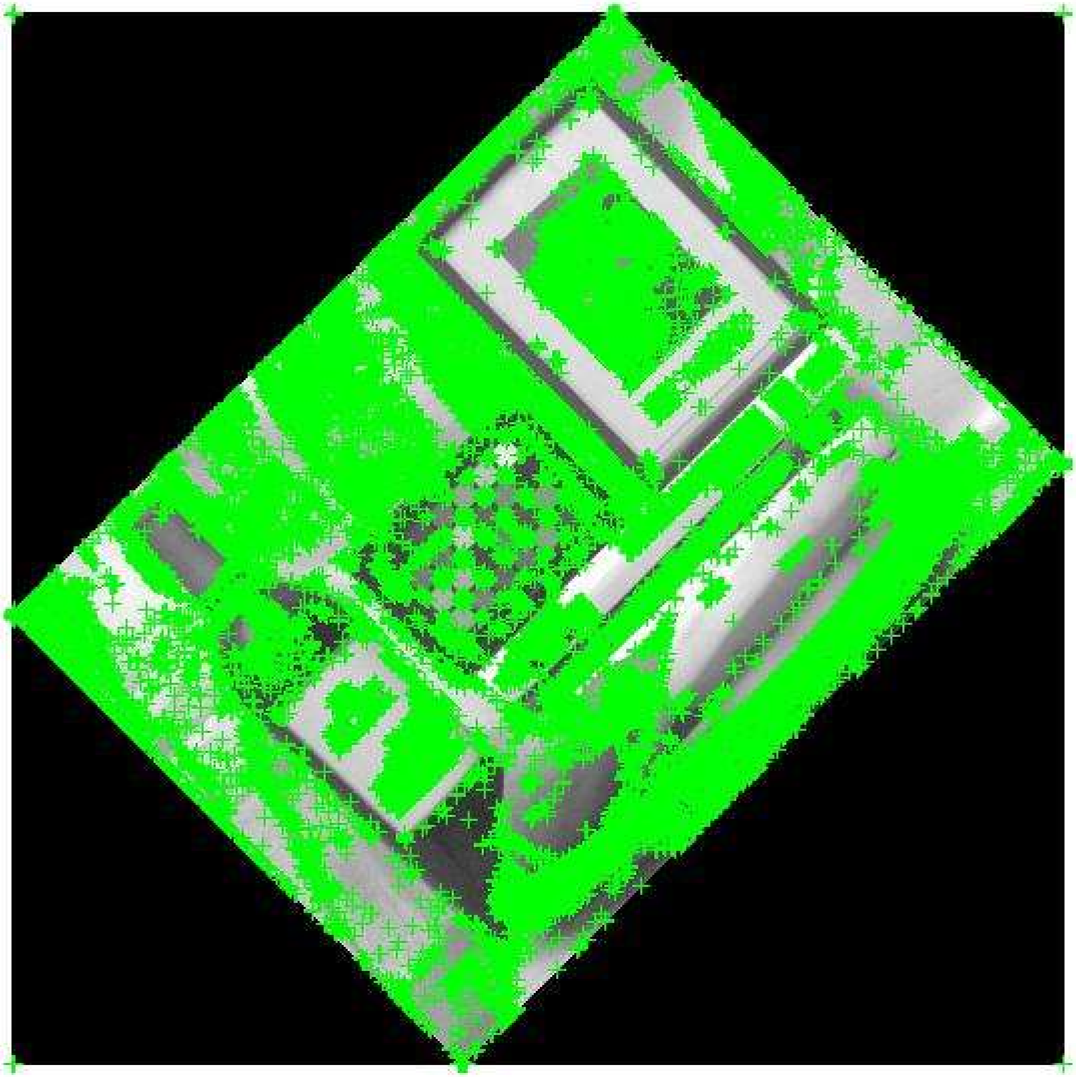


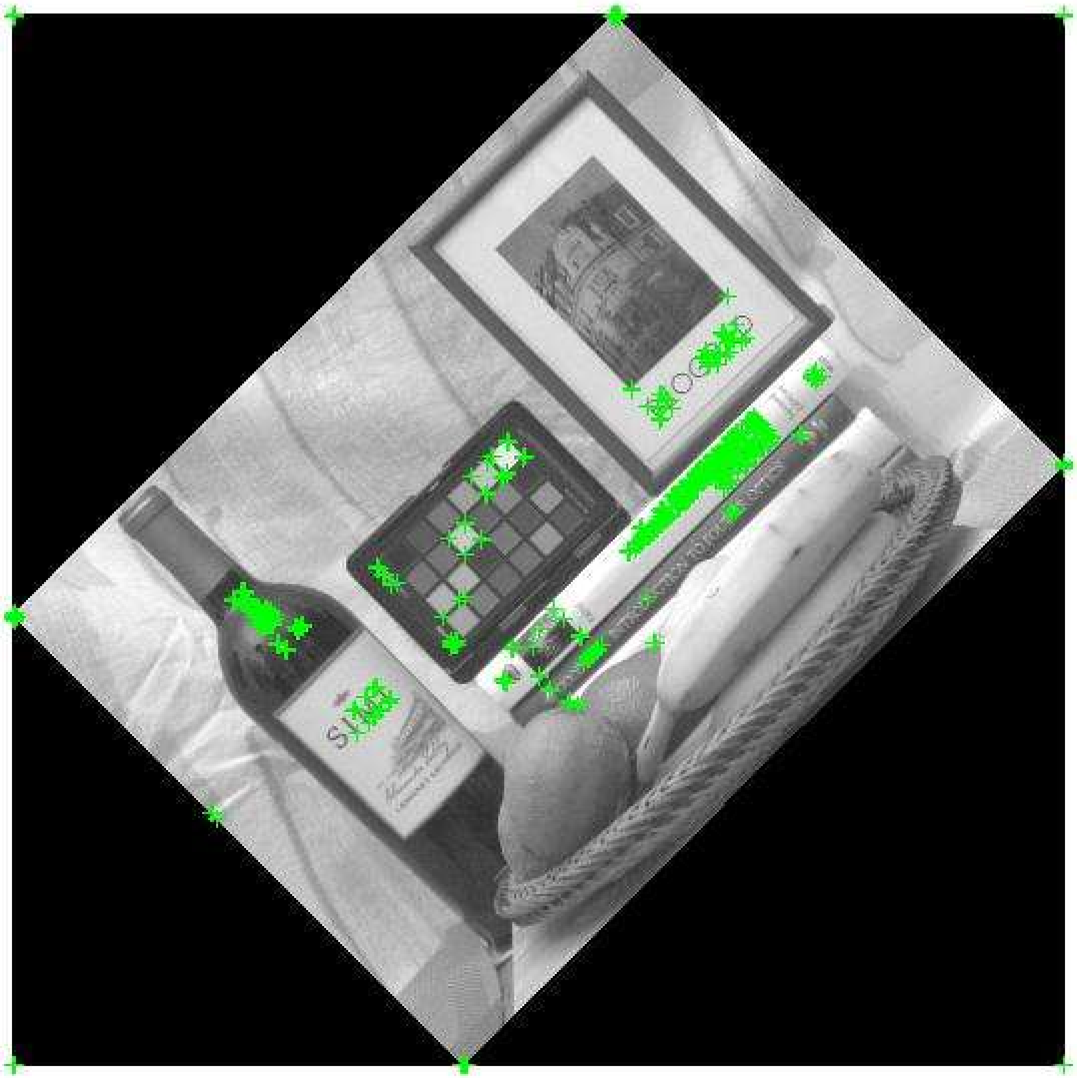
(b) Y-derivative of the rotated image

Y derivative of Rotated Image



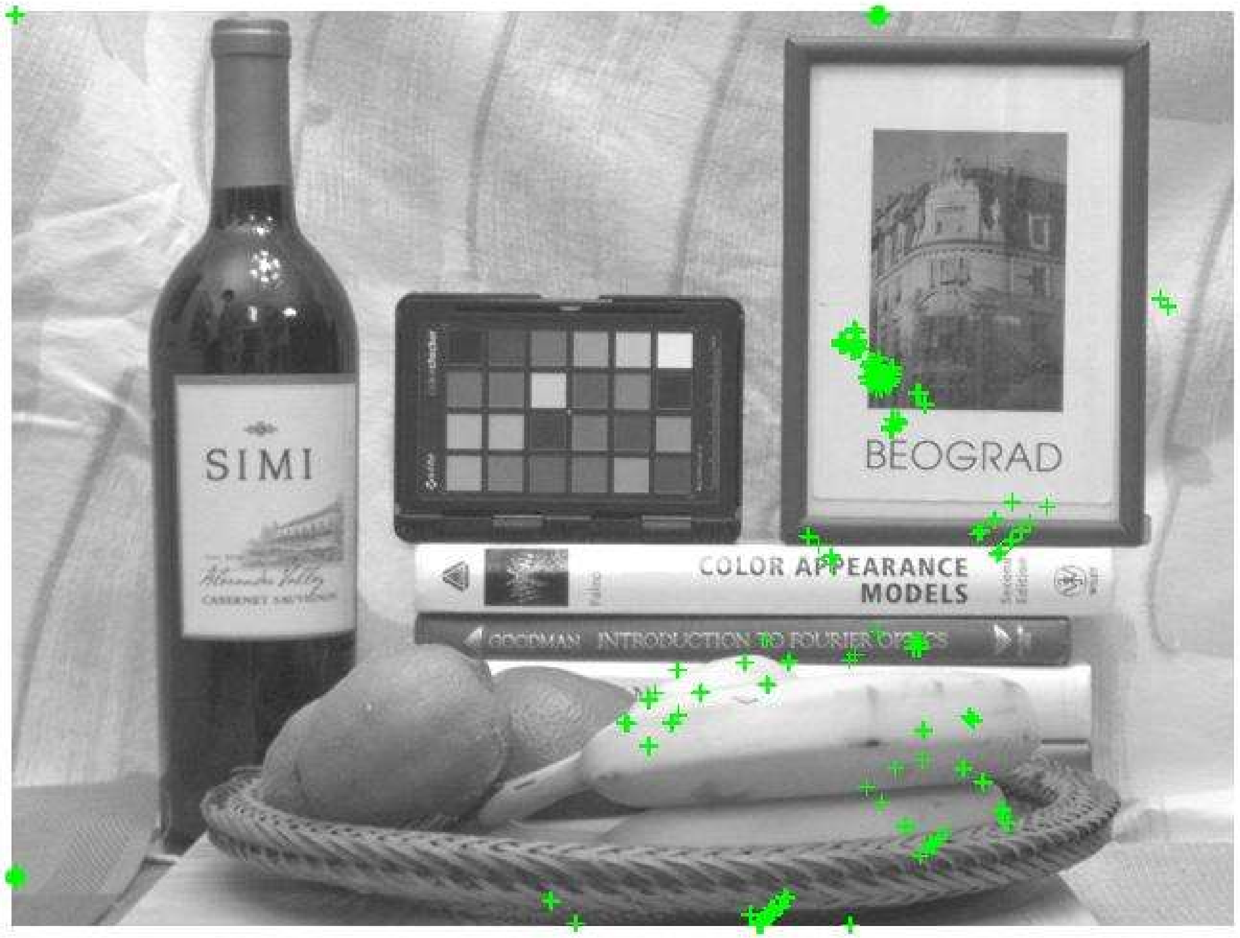
(c) **Marked locations of the selected features on the rotated image**
Selected Threshold value is 3000000





(d) Display of image using adaptive non-maximal suppression to select features

6.) Final image with matched features marked as green dots
A total of 536 features were matched.



It is seen that computation time was too much for detecting features in 2nd part as it is gray scale image.

Code:

```
clc
close all
clear all
tic
original_image = imread('checkerboard.jpg');
x_derivative = [-2 -1 0 1 2];
y_derivative = x_derivative';
original_image = original_image(:,:,1);
AxDerivative1 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AyDerivative1 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AxDerivative3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AyDerivative3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
A_y_der_3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
base = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
kernel =
[0.0183,0.0821,0.1353,0.0821,0.0183;0.0821,0.3679,0.6065,0.3679,0.0821;0.1353,0.6065,1,0.6065,0.1
353;0.0821,0.3679,0.6065,0.3670,0.0821;0.0183,0.0821,0.1353,0.0821,0.0183];
figure;
imshow(original_image)
title('Original Image');
```

```
AxDerivative = padarray (original_image, [1 1], 255);
AxDerivative = padarray (AxDerivative, [1 1], 255);
AyDerivative = padarray (original_image, [1 1 ], 255);
AyDerivative = padarray (AyDerivative, [1 1 ], 255);
for i = 3 : size(AxDerivative,1) - 2
    for j = 3 : size(AxDerivative,2) - 2
        temp = double(AxDerivative(i, j-2:j+2));
        AxDerivative1(i,j) = temp * x_derivative';
    end
end
finalAxDerivative = AxDerivative1(3:size(AxDerivative,1)-2,3:size(AxDerivative,2)-2);
figure
imshow(uint8(finalAxDerivative));
title('X derivative of Image');
for i = 3 : size(AyDerivative,1) - 2
    for j = 3 : size(AyDerivative,2) - 2
        temp = double(AyDerivative(i-2: i+2, j));
        AyDerivative1(i,j) = temp' * y_derivative;
    end
end
finalAyDerivative= AyDerivative1(3:size(AyDerivative,1)-2, 3:size(AyDerivative,2)-2);
figure
imshow(uint8(finalAyDerivative));
title('Y derivative of Image');
A_x_der_2 = AxDerivative1 .^2;
A_y_der_2 = AyDerivative1 .^2;
A_x_y_der = AxDerivative1 .* AyDerivative1 ;
for i = 3 :size(AxDerivative1, 1) - 2
    for j = 3 :size(AxDerivative1, 2) - 2
        temp = A_x_der_2(i-2: i+2, j-2: j+2);
        AxDerivative3(i, j) =  sum(sum(kernel .* temp));
        temp = A_y_der_2(i-2: i+2, j-2: j+2);
        A_y_der_3(i, j) = sum(sum(kernel .* temp));
        temp = A_x_y_der(i-2: i+2, j-2: j+2);
        AyDerivative3(i, j) = sum(sum(kernel .* temp));
    end
end
for i = 3:size(AxDerivative1, 1) - 2
    for j = 3: size(AxDerivative1, 2) - 2
        A_pixel = double([AxDerivative3(i,j) AyDerivative3(i,j) ;  AyDerivative3(i,j)
A_y_der_3(i, j)]);
        base(i, j) = det(A_pixel) - 0.06 * ((trace(A_pixel))^2);
    end
end
threshold_value = 85000;
r = 1;
for i = 3:size(AxDerivative1, 1) - 2
    for j =  3:size(AxDerivative1, 2) - 2
        if( base (i, j)>threshold_value)
            base1(i,j) = base(i , j);
             x_1(r) = i;
             y_1(r) = j;
             r = r+1;
        else
            base1(i , j) = 0;
        end
    end
end
figure
imshowpair(original_image, uint8(base1(3:size(base,1)-2, 3:size(base,2)-2)), 'montage')
title('Original Image and Thresholded Image');
figure
imshow(original_image);
hold on
plot(y_1(:,:), x_1(:,:), '+g')
```

```
q=1;
for i=1:size(base1,1)
    for j=1:size(base1,2)
        if(base1(i,j)~=0)
        base1Array(q)=base1(i,j);
        x_2(q)=i;
        y_2(q)=j;
        q=q+1;
        end
    end
end

[R,a]=sort(base1Array,'descend');
        for i=1:size(a,2)
            x_3(i)=x_2(a(i));
            y_3(i)=y_2(a(i));
        end

k=15;
R_new=0;
for i=2:size(R,2)
    minimum_distance=100;
    for j=i-1:-1:1
        if(((R(j)-R(i))>(k*R(i)/100)))
            distance=sqrt((x_3(i)-x_3(j))^2+((y_3(i)-y_3(j))^2));
            if (distance<minimum_distance)
                minimum_distance=distance;
                if(R_new==0)
                    R_new(1)=max(max(R));
                    radius(1)=450*579;
                    x_4(1)=x_3(1);
                    y_4(1)=y_3(1);
                end
                R_new(i)=R(j);
                radius(i)=minimum_distance;
                x_4(i)=x_3(i);
                y_4(i)=y_3(i);
            end
        end
    end
end
R_new = R;
top_n=2000;
for i=top_n:size(R_new,2)
    R_new(i)=0;
end

final_matrix=zeros(450,579);

for i=1:size(x_3,2)
    final_matrix(x_3(i),y_3(i))=R_new(i);
end
r=1;
for i = 1:size(final_matrix,1)
    for j =1:size(final_matrix,2)
        if(final_matrix(i,j) ~= 0)
            x_5(r,1) = i;
            y_5(r,1) = j;
            r = r+1;
        end
    end
end
```

```
figure
imshow(original_image)
hold on
plot(y_5(:,:), x_5(:,:),'+g')

original_image = imread('checkerboard.jpg');
original_image=imrotate(original_image,45);
x_derivative = [-2 -1 0 1 2];
y_derivative = x_derivative';
original_image = original_image(:,:,1);
AxDerivative1 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AyDerivative1 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AxDerivative3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AyDerivative3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
A_y_der_3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
base = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
kernel =
[0.0183,0.0821,0.1353,0.0821,0.0183;0.0821,0.3679,0.6065,0.3679,0.0821;0.1353,0.6065,1,0.6065,0.1
353;0.0821,0.3679,0.6065,0.3670,0.0821;0.0183,0.0821,0.1353,0.0821,0.0183];
figure
imshow(original_image)
AxDerivative = padarray (original_image, [1 1], 255);
AxDerivative = padarray (AxDerivative, [1 1], 255);
AyDerivative = padarray (original_image, [1 1 ], 255);
AyDerivative = padarray (AyDerivative, [1 1 ], 255);
for i = 3 : size(AxDerivative,1) - 2
    for j = 3 : size(AxDerivative,2) - 2
        temp = double(AxDerivative(i, j-2:j+2));
        AxDerivative1(i,j) = temp * x_derivative';
    end
end
finalAxDerivative = AxDerivative1(3:size(AxDerivative,1)-2,3:size(AxDerivative,2)-2);
figure
imshow(uint8(finalAxDerivative));
title('X derivative of Rotated Image');
for i = 3 : size(AyDerivative,1) - 2
    for j = 3 : size(AyDerivative,2) - 2
        temp = double(AyDerivative(i-2: i+2, j));
        AyDerivative1(i,j) = temp' * y_derivative;
    end
end
finalAyDerivative= AyDerivative1(3:size(AyDerivative,1)-2, 3:size(AyDerivative,2)-2);
figure
imshow(uint8(finalAyDerivative));
title('Y derivative of Rotated Image');
A_x_der_2 = AxDerivative1 .^2;
A_y_der_2 = AyDerivative1 .^2;
A_x_y_der = AxDerivative1 .* AyDerivative1 ;
for i = 3 :size(AxDerivative1, 1) - 2
    for j = 3 :size(AxDerivative1, 2) - 2
        temp = A_x_der_2(i-2: i+2, j-2: j+2);
        AxDerivative3(i, j) =  sum(sum(kernel .* temp));
        temp = A_y_der_2(i-2: i+2, j-2: j+2);
        A_y_der_3(i, j) = sum(sum(kernel .* temp));
        temp = A_x_y_der(i-2: i+2, j-2: j+2);
        AyDerivative3(i, j) = sum(sum(kernel .* temp));
    end
end
for i = 3:size(AxDerivative1, 1) - 2
    for j = 3: size(AxDerivative1, 2) - 2
        A_pixel = double([AxDerivative3(i,j) AyDerivative3(i,j) ;  AyDerivative3(i,j)
A_y_der_3(i, j)]);
        base(i, j) = det(A_pixel) - 0.06 * ((trace(A_pixel))^2);
    end
```

```
end
threshold_value = 900000;
r = 1;
for i = 3:size(AxDerivative1, 1) - 2
    for j =  3:size(AxDerivative1, 2) - 2
        if( base (i, j)>threshold_value)
            base1(i,j) = base(i , j);
             x_1(r) = i;
             y_1(r) = j;
             r = r+1;
        else
            base1(i , j) = 0;
        end
    end
end
figure
imshowpair(original_image, uint8(base1(3:size(base,1)-2, 3:size(base,2)-2)), 'montage')
title('Rotated Image and Thresholded Image');
figure
imshow(original_image)
hold on
plot(y_1(:,:), x_1(:,:),'+g')

q=1;
for i=1:size(base1,1)
    for j=1:size(base1,2)
        if(base1(i,j)~=0)
        base1Array(q)=base1(i,j);
        x_2(q)=i;
        y_2(q)=j;
        q=q+1;
        end
    end
end

[R,a]=sort(base1Array,'descend');
        for i=1:size(a,2)
            x_3(i)=x_2(a(i));
            y_3(i)=y_2(a(i));
        end

k=15;
R_new=0;
for i=2:size(R,2)
    minimum_distance=100;
    for j=i-1:-1:1
        if(((R(j)-R(i))>(k*R(i)/100)))
            distance=sqrt((x_3(i)-x_3(j))^2+((y_3(i)-y_3(j))^2));
            if (distance<minimum_distance)
                minimum_distance=distance;
                if(R_new==0)
                    R_new(1)=max(max(R));
                    radius(1)=450*579;
                    x_4(1)=x_3(1);
                    y_4(1)=y_3(1);
                end
                R_new(i)=R(j);
                radius(i)=minimum_distance;
                x_4(i)=x_3(i);
                y_4(i)=y_3(i);
            end
        end
    end
end
```

```
R_new = R;
top_n=2000;
for i=top_n:size(R_new,2)
    R_new(i)=0;
end

final_mat_rot=zeros(729,729);

for i=1:size(x_3,2)
    final_mat_rot(x_3(i),y_3(i))=R_new(i);
end

r=1;
for i = 1:size(final_mat_rot,1)
    for j =1:size(final_mat_rot,2)
        if(final_mat_rot(i,j) ~= 0)
            x_5(r,1) = i;
            y_5(r,1) = j;
            r = r+1;
        end
    end
end



figure
imshow(original_image)
hold on
plot(y_5(:,:), x_5(:,:),'+g')



c=1;
for i=1:size(final_matrix,1)
    for j=1:size(final_matrix,2)
        if(final_matrix(i,j)~=0)
            array_without_rot(c)=final_matrix(i,j);
            x_2(c)=i;
            y_2(c)=j;
            c=c+1;
end
end
end
[R,a]=sort(array_without_rot,'descend');
        for i=1:size(a,2)
            x_3(i)=x_2(a(i));
            y_3(i)=y_2(a(i));
        end
c=1;
for i=1:size(final_mat_rot,1)
    for j=1:size(final_mat_rot,2)
        if(final_mat_rot(i,j)~=0)
            array_with_rot(c)=final_mat_rot(i,j);
            x_cord_rot(c)=i;
            y_cord_rot(c)=j;
            c=c+1;
end
end
end
[R_rot,a_rot]=sort(array_with_rot,'descend');
        for i=1:size(a_rot,2)
            x_cord_rot_new(i)=x_cord_rot(a_rot(i));
            y_cord_rot_new(i)=y_cord_rot(a_rot(i));
end
count=0;
for i=1:size(R,2)
```

```
    for j=1:size(R,2)
        if(sqrt((x_3(i)-x_cord_rot_new(j))^2+(y_3(i)-y_cord_rot_new(j))^2)<3)
            count=count+1;
end
end
end
toc
tt=toc-tic;
original_image = imread('checkerboard.jpg');
figure;imshow(original_image);
hold on;
plot(x_3, y_3, '+g');

%% Part 2



clc
close all
clear all
tic
original_image = imread('image.bmp');
x_derivative = [-2 -1 0 1 2];
y_derivative = x_derivative';
original_image = original_image(:,:,1);
AxDerivative1 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AyDerivative1 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AxDerivative3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AyDerivative3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
A_y_der_3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
base = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
kernel =
[0.0183,0.0821,0.1353,0.0821,0.0183;0.0821,0.3679,0.6065,0.3679,0.0821;0.1353,0.6065,1,0.6065,0.1
353;0.0821,0.3679,0.6065,0.3670,0.0821;0.0183,0.0821,0.1353,0.0821,0.0183];
figure;
imshow(original_image)
title('Original Image');
AxDerivative = padarray (original_image, [1 1], 255);
AxDerivative = padarray (AxDerivative, [1 1], 255);
AyDerivative = padarray (original_image, [1 1 ], 255);
AyDerivative = padarray (AyDerivative, [1 1 ], 255);
for i = 3 : size(AxDerivative,1) - 2
    for j = 3 : size(AxDerivative,2) - 2
        temp = double(AxDerivative(i, j-2:j+2));
        AxDerivative1(i,j) = temp * x_derivative';
    end
end
finalAxDerivative = AxDerivative1(3:size(AxDerivative,1)-2,3:size(AxDerivative,2)-2);
figure
imshow(uint8(finalAxDerivative));
title('X derivative of Image');
for i = 3 : size(AyDerivative,1) - 2
    for j = 3 : size(AyDerivative,2) - 2
        temp = double(AyDerivative(i-2: i+2, j));
        AyDerivative1(i,j) = temp' * y_derivative;
    end
end
finalAyDerivative= AyDerivative1(3:size(AyDerivative,1)-2, 3:size(AyDerivative,2)-2);
figure
imshow(uint8(finalAyDerivative));
title('Y derivative of Image');
A_x_der_2 = AxDerivative1 .^2;
A_y_der_2 = AyDerivative1 .^2;
A_x_y_der = AxDerivative1 .* AyDerivative1 ;
for i = 3 :size(AxDerivative1, 1) - 2
```

```
    for j = 3 :size(AxDerivative1, 2) - 2
        temp = A_x_der_2(i-2: i+2, j-2: j+2);
        AxDerivative3(i, j) =  sum(sum(kernel .* temp));
        temp = A_y_der_2(i-2: i+2, j-2: j+2);
        A_y_der_3(i, j) = sum(sum(kernel .* temp));
        temp = A_x_y_der(i-2: i+2, j-2: j+2);
        AyDerivative3(i, j) = sum(sum(kernel .* temp));
    end
end
for i = 3:size(AxDerivative1, 1) - 2
    for j = 3: size(AxDerivative1, 2) - 2
        A_pixel = double([AxDerivative3(i,j) AyDerivative3(i,j) ;  AyDerivative3(i,j)
A_y_der_3(i, j)]);
        base(i, j) = det(A_pixel) - 0.06 * ((trace(A_pixel))^2);
    end
end
threshold_value = 3000000;
r = 1;
for i = 3:size(AxDerivative1, 1) - 2
    for j =  3:size(AxDerivative1, 2) - 2
        if( base (i, j)>threshold_value)
            base1(i,j) = base(i , j);
             x_1(r) = i;
             y_1(r) = j;
             r = r+1;
        else
            base1(i , j) = 0;
        end
    end
end
figure
imshowpair(original_image, uint8(base1(3:size(base,1)-2, 3:size(base,2)-2)), 'montage')
title('Original Image and Thresholded Image');
figure
imshow(original_image);
hold on
plot(y_1(:,:), x_1(:,:), '+g')

q=1;
for i=1:size(base1,1)
    for j=1:size(base1,2)
        if(base1(i,j)~=0)
        base1Array(q)=base1(i,j);
        x_2(q)=i;
        y_2(q)=j;
        q=q+1;
        end
    end
end

[R,a]=sort(base1Array,'descend');
        for i=1:size(a,2)
            x_3(i)=x_2(a(i));
            y_3(i)=y_2(a(i));
        end

k=15;
R_new=0;
for i=2:size(R,2)
    minimum_distance=100;
    for j=i-1:-1:1
        if(((R(j)-R(i))>(k*R(i)/100)))
            distance=sqrt((x_3(i)-x_3(j))^2+((y_3(i)-y_3(j))^2));
            if (distance<minimum_distance)
```

```
                minimum_distance=distance;
                if(R_new==0)
                    R_new(1)=max(max(R));
                    radius(1)=450*579;
                    x_4(1)=x_3(1);
                    y_4(1)=y_3(1);
                end
                R_new(i)=R(j);
                radius(i)=minimum_distance;
                x_4(i)=x_3(i);
                y_4(i)=y_3(i);
            end
        end
    end
end
R_new = R;
top_n=2000;
for i=top_n:size(R_new,2)
    R_new(i)=0;
end

final_matrix=zeros(450,579);

for i=1:size(x_3,2)
    final_matrix(x_3(i),y_3(i))=R_new(i);
end
r=1;
for i = 1:size(final_matrix,1)
    for j =1:size(final_matrix,2)
        if(final_matrix(i,j) ~= 0)
            x_5(r,1) = i;
            y_5(r,1) = j;
            r = r+1;
        end
    end
end



figure
imshow(original_image)
hold on
plot(y_5(:,:), x_5(:,:),'+g')

original_image = imread('image.bmp');
original_image=imrotate(original_image,45);
x_derivative = [-2 -1 0 1 2];
y_derivative = x_derivative';
original_image = original_image(:,:,1);
AxDerivative1 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AyDerivative1 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AxDerivative3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
AyDerivative3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
A_y_der_3 = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
base = zeros(size(original_image, 1)+ 4, size(original_image, 2)+ 4);
kernel =
[0.0183,0.0821,0.1353,0.0821,0.0183;0.0821,0.3679,0.6065,0.3679,0.0821;0.1353,0.6065,1,0.6065,0.1
353;0.0821,0.3679,0.6065,0.3670,0.0821;0.0183,0.0821,0.1353,0.0821,0.0183];
figure
imshow(original_image)
AxDerivative = padarray (original_image, [1 1], 255);
AxDerivative = padarray (AxDerivative, [1 1], 255);
AyDerivative = padarray (original_image, [1 1 ], 255);
AyDerivative = padarray (AyDerivative, [1 1 ], 255);
for i = 3 : size(AxDerivative,1) - 2
```

```
    for j = 3 : size(AxDerivative,2) - 2
        temp = double(AxDerivative(i, j-2:j+2));
        AxDerivative1(i,j) = temp * x_derivative';
    end
end
finalAxDerivative = AxDerivative1(3:size(AxDerivative,1)-2,3:size(AxDerivative,2)-2);
figure
imshow(uint8(finalAxDerivative));
title('X derivative of Rotated Image');
for i = 3 : size(AyDerivative,1) - 2
    for j = 3 : size(AyDerivative,2) - 2
        temp = double(AyDerivative(i-2: i+2, j));
        AyDerivative1(i,j) = temp' * y_derivative;
    end
end
finalAyDerivative= AyDerivative1(3:size(AyDerivative,1)-2, 3:size(AyDerivative,2)-2);
figure
imshow(uint8(finalAyDerivative));
title('Y derivative of Rotated Image');
A_x_der_2 = AxDerivative1 .^2;
A_y_der_2 = AyDerivative1 .^2;
A_x_y_der = AxDerivative1 .* AyDerivative1 ;
for i = 3 :size(AxDerivative1, 1) - 2
    for j = 3 :size(AxDerivative1, 2) - 2
        temp = A_x_der_2(i-2: i+2, j-2: j+2);
        AxDerivative3(i, j) =  sum(sum(kernel .* temp));
        temp = A_y_der_2(i-2: i+2, j-2: j+2);
        A_y_der_3(i, j) = sum(sum(kernel .* temp));
        temp = A_x_y_der(i-2: i+2, j-2: j+2);
        AyDerivative3(i, j) = sum(sum(kernel .* temp));
    end
end
for i = 3:size(AxDerivative1, 1) - 2
    for j = 3: size(AxDerivative1, 2) - 2
        A_pixel = double([AxDerivative3(i,j) AyDerivative3(i,j) ;  AyDerivative3(i,j)
A_y_der_3(i, j)]);
        base(i, j) = det(A_pixel) - 0.06 * ((trace(A_pixel))^2);
    end
end
threshold_value = 3000000;
r = 1;
for i = 3:size(AxDerivative1, 1) - 2
    for j =  3:size(AxDerivative1, 2) - 2
        if( base (i, j)>threshold_value)
            base1(i,j) = base(i , j);
             x_1(r) = i;
             y_1(r) = j;
             r = r+1;
        else
            base1(i , j) = 0;
        end
    end
end
figure
imshowpair(original_image, uint8(base1(3:size(base,1)-2, 3:size(base,2)-2)), 'montage')
title('Rotated Image and Thresholded Image');
figure
imshow(original_image)
hold on
plot(y_1(:,:), x_1(:,:),'+g')

q=1;
for i=1:size(base1,1)
    for j=1:size(base1,2)
        if(base1(i,j)~=0)
        base1Array(q)=base1(i,j);
        x_2(q)=i;
        y_2(q)=j;
        q=q+1;
        end
    end
end

[R,a]=sort(base1Array,'descend');
        for i=1:size(a,2)
            x_3(i)=x_2(a(i));
            y_3(i)=y_2(a(i));
        end

k=15;
R_new=0;
for i=2:size(R,2)
    minimum_distance=100;
    for j=i-1:-1:1
        if(((R(j)-R(i))>(k*R(i)/100)))
            distance=sqrt((x_3(i)-x_3(j))^2+((y_3(i)-y_3(j))^2));
            if (distance<minimum_distance)
                minimum_distance=distance;
                if(R_new==0)
                    R_new(1)=max(max(R));
                    radius(1)=450*579;
                    x_4(1)=x_3(1);
                    y_4(1)=y_3(1);
                end
                R_new(i)=R(j);
                radius(i)=minimum_distance;
                x_4(i)=x_3(i);
                y_4(i)=y_3(i);
            end
        end
    end
end
R_new = R;
top_n=2000;
for i=top_n:size(R_new,2)
    R_new(i)=0;
end

final_mat_rot=zeros(729,729);

for i=1:size(x_3,2)
    final_mat_rot(x_3(i),y_3(i))=R_new(i);
end

r=1;
for i = 1:size(final_mat_rot,1)
    for j =1:size(final_mat_rot,2)
        if(final_mat_rot(i,j) ~= 0)
            x_55(r,1) = i;
            y_55(r,1) = j;
            r = r+1;
        end
    end
end



figure
imshow(original_image)
hold on
```

```
plot(y_55(:,:), x_55(:,:),'+g')


c=1;
for i=1:size(final_matrix,1)
    for j=1:size(final_matrix,2)
        if(final_matrix(i,j)~=0)
            array_without_rot(c)=final_matrix(i,j);
            x_2(c)=i;
            y_2(c)=j;
            c=c+1;
end
end
end
[R,a]=sort(array_without_rot,'descend');
        for i=1:size(a,2)
            x_3(i)=x_2(a(i));
            y_3(i)=y_2(a(i));
        end
c=1;
for i=1:size(final_mat_rot,1)
    for j=1:size(final_mat_rot,2)
        if(final_mat_rot(i,j)~=0)
            array_with_rot(c)=final_mat_rot(i,j);
            x_cord_rot(c)=i;
            y_cord_rot(c)=j;
            c=c+1;
end
end
end
[R_rot,a_rot]=sort(array_with_rot,'descend');
        for i=1:size(a_rot,2)
            x_cord_rot_new(i)=x_cord_rot(a_rot(i));
            y_cord_rot_new(i)=y_cord_rot(a_rot(i));
end
count=0;
for i=1:size(R,2)
    for j=1:size(R,2)
        if(sqrt((x_3(i)-x_cord_rot_new(j))^2+(y_3(i)-y_cord_rot_new(j))^2)<3)
            count=count+1;
end
end
end
toc
tt=toc-tic;
original_image = imread('image.bmp');
figure;imshow(original_image);
plot(x_5, y_5, '+g');
hold on;
plot(x_cord_rot_new, y_cord_rot_new, '+g');
```