Comp Eng 4TN4
Project 1: Image Enhancement

Sara Jamil 1143947

<u>Histogram Equalization</u>

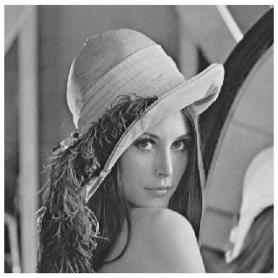
Histogram Equalization (HE) is an image enhancement technique that helps evenly distribute image intensities to enhance contrast. Histogram equalization uses the cumulative distribution function (CDF), as shown below.

$$p_x(i) = p(x=i) = \frac{n_i}{n}, \quad 0 \le i < L$$
$$cdf_x(i) = \sum_{j=0}^{i} p_x(j)$$

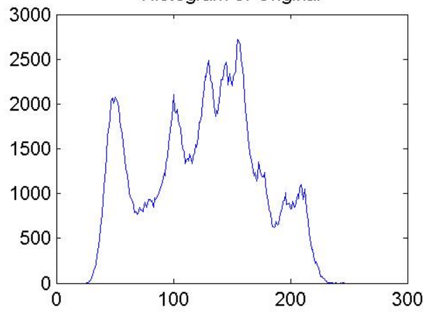
The transform that is used in histogram equalization is a single-valued and monotonically increasing. This forces the mapping of the lowest gray-level to zero and the highest gray-level value to the maximum gray-level, which can be determined by the number of bits used to represent pixel intensity. This results in the gray-levels to be distributed to span across missing gray-level values of gray-levels or less occurring gray-level values. Therefore, the details that may have been previously difficult to detect due to small changes of intensity values, they may become easier to discern as the gray-levels are spread to a larger range. In this way the enhanced image will display a higher contrast.

Test 1:

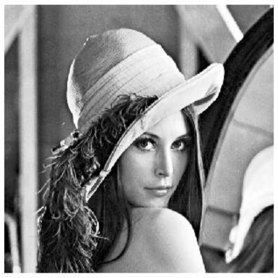
Original Image



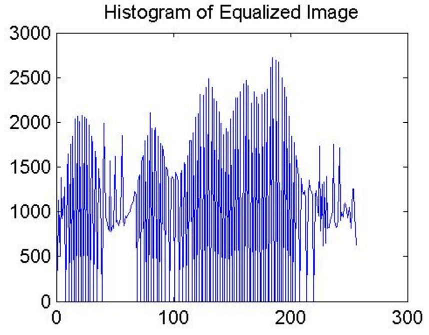
Histogram of Original



Histogram equalization



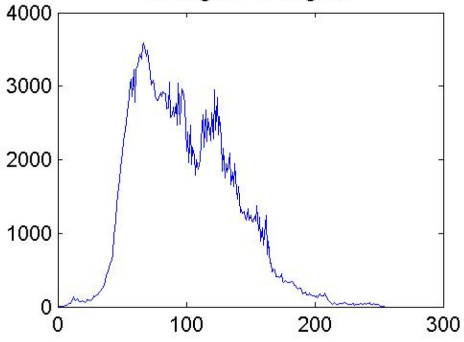
Histogram of Equalized Image

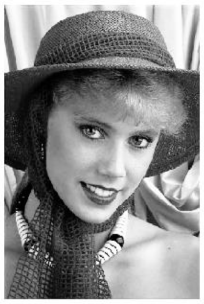


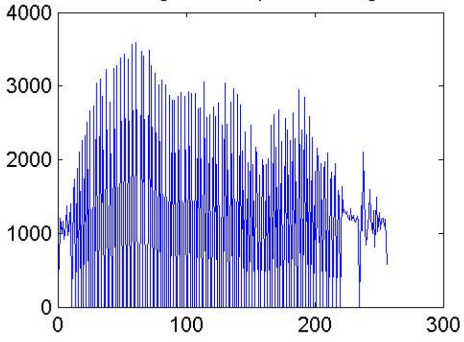
The following test cases show examples of histogram equalization that produce relatively desirable results. The contrast of the images is increased and more details are noticeable.

Test 2:

Original Image

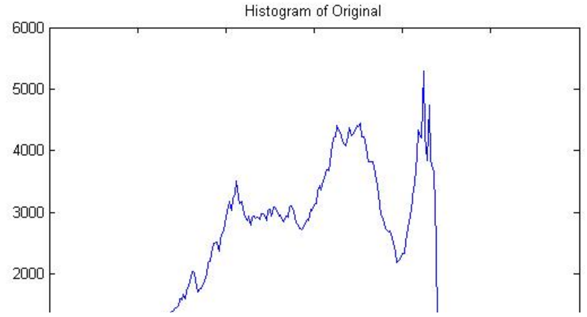

Histogram of Original

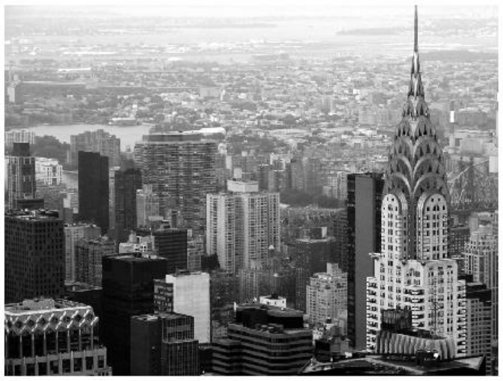

Histogram equalization

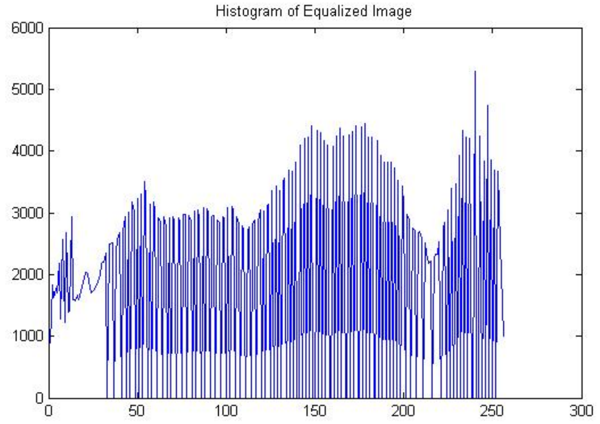

Histogram of Equalized Image

Test 3:

Original Image

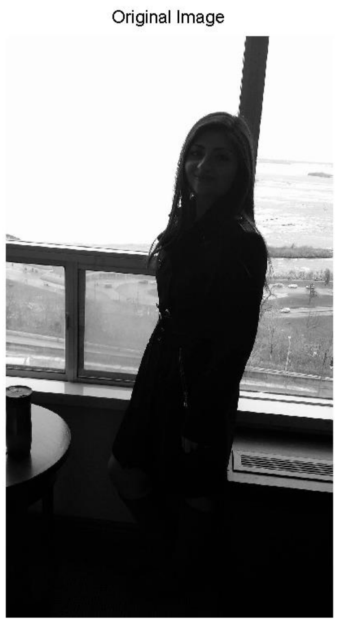

Histogram of Original

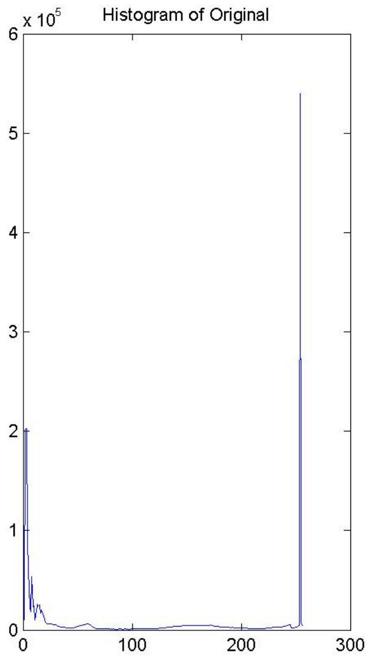

Histogram equalization

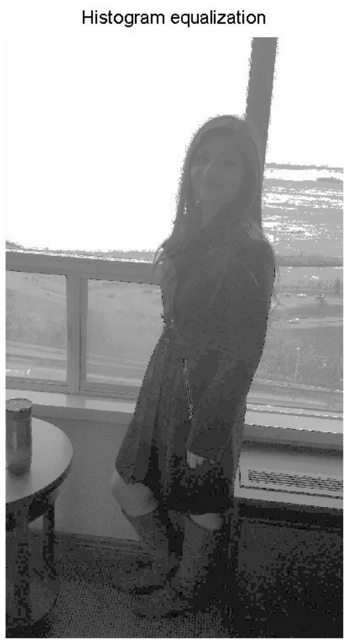

Histogram of Equalized Image

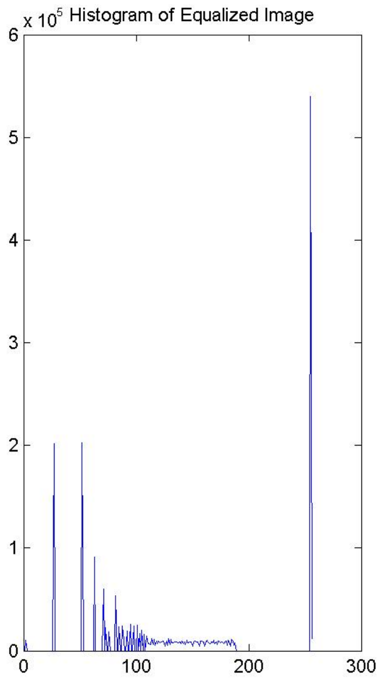
The following test case shows an example of undesirable histogram equalization output. This image already contains high contrast, and when histogram equalization is applied, the levels are spread out to more gray levels, resulting in a distorted image, where neither the foreground nor the background is enhanced.

Test 4:



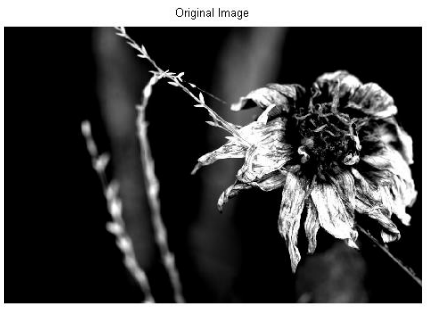


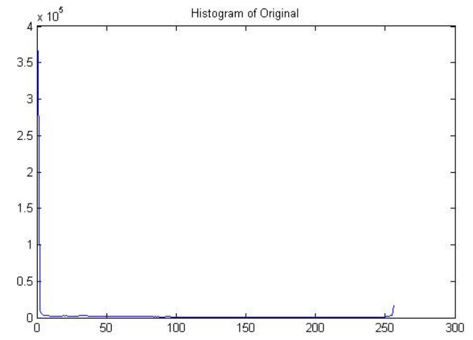


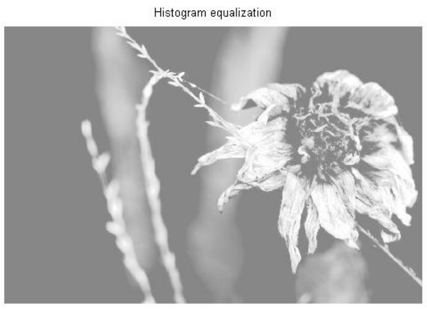


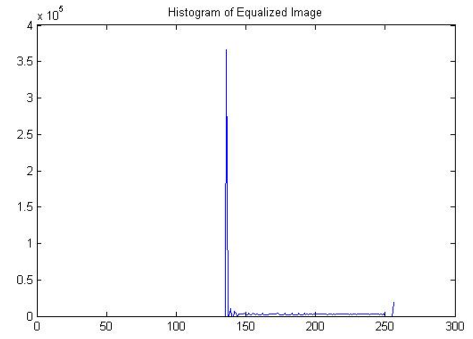
This high contrast image becomes equalized to have a more washed-out appearance. Here, the contrast is lost.

Test 5:

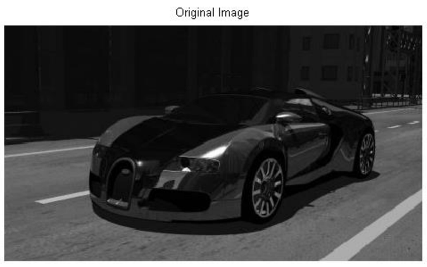


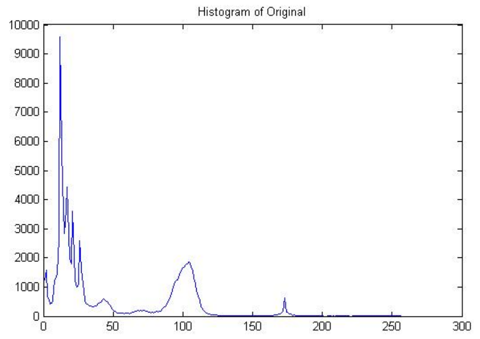


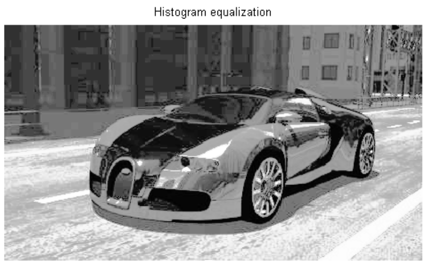


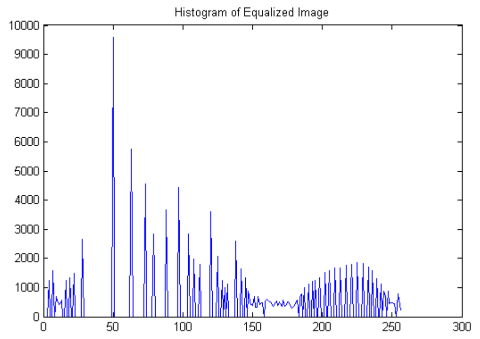


Test 6:









The image of the car was able to make certain ranges more visible; however, the image appears to be pixelated and generally becomes too bright, losing its original tone.

Optimal Contrast Tone Mapping

Optimal Contrast Tone Mapping (OCTM) is a method that has the same goal of contrast stretching as in histogram equalization. However, OCTM also applies constraints to ensure that the tone of the image is not lost in the process. OCTM utilizes linear programming optimisation in which an objective function, such as contrast, is maximized subject to constraints. These constraints may include the limited number of grayscale values that the pixels can take due to the fact that the gray-levels are determined by the number of bits used to represent pixel intensity. Another constraint is limiting the tone distortion so as to not have many input gray-levels being mapped onto the same output gray-level.

There are several conditions that are placed on the output of the OCTM images. Firstly, an integer-to-integer mapping is required, since the same number of gray-levels is to be maintained, and is discrete.

$$T : \{0, 1, \cdots, 255\} \rightarrow \{0, 1, \cdots, 255\}$$

T is the transfer function, and the values represent the intensity values.
Secondly, the condition for monotonicity must also be met as described by the following equations.

$$T(j) \geq T(i) \quad \text{if } j > i$$

$$T(i) = \sum_{0 \leq j \leq i} s_j, \; 0 \leq i < 256$$

$$0 \leq s_j < 256$$

$$\sum_{0 \leq j < 256} s_j < 256$$

Sj is the increment in output intensity versus a unit step up in input level j (i.e. the contrast level at j). The expected contrast level of the image is given by the following equation.

$$C(\mathbf{s}) = \sum_{0 \leq j < N_x} p_j s_j$$

Pj is the probability of the gray-level j. This is the objective function that must be maximized through linear programming methods. The following represent the constraints of the optimization problem.

$$\max_{\mathbf{s}} \sum_{0 \leq j < L} p_j s_j$$

$$\text{subject to (a)} \sum_{0 \leq j < L} s_j < \text{Ł};$$

$$\text{(b) } s_j \geq 0, \; 0 \leq j < L;$$

$$\text{(c)} \sum_{j \leq i < j+d} s_i \geq 1, \; 0 \leq j < L - d.$$

a) Confines the output intensity level to the available dynamic range
b) Ensures the monotonicity condition of the transfer function
c) Specifies the maximum tone distortion allowed by setting d at the upper limit

The contrast-tone optimization problem can therefore be stated by the following statement and the transfer function can be calculated using the equation below.

$$\max_{\mathbf{s}} \sum_{0 \le j < L} p_j s_j$$

$$\text{subject to} \quad \sum_{0 \le j < L} s_j < \text{Ł};$$

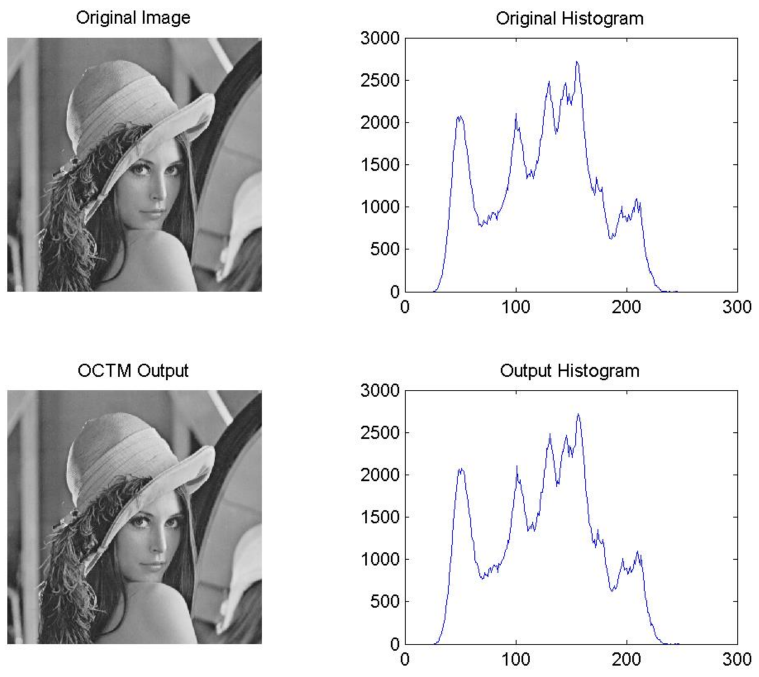
$$s_j \ge 1/d, \; 0 \le j < L.$$

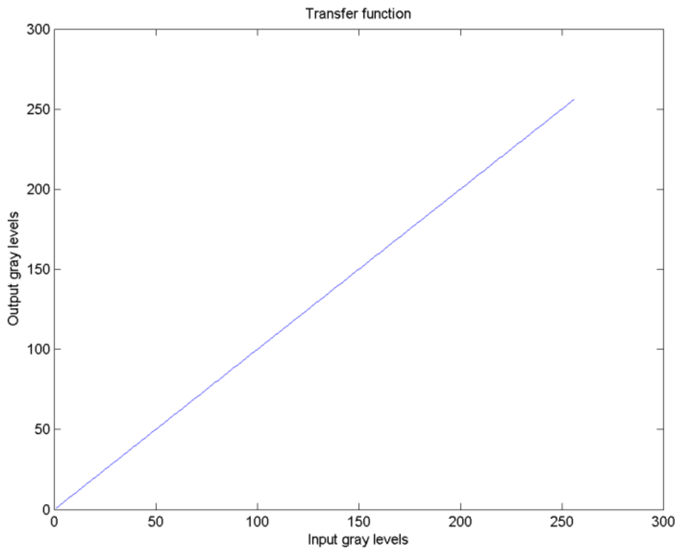
$$T(i) = \left\lfloor \sum_{0 \le j \le i} s_j + 0.5 \right\rfloor, \; 0 \le i < L$$

This problem can be solved using the optimization toolbox linear programming function on Matlab; linprog (f,A,b,lower_bound,upper_bound), where f is the objective function, A and b are the inequality constraints imposed on the objective function, and the upper bound and lower bound of the problem is specified. The function will then solve for Sj.

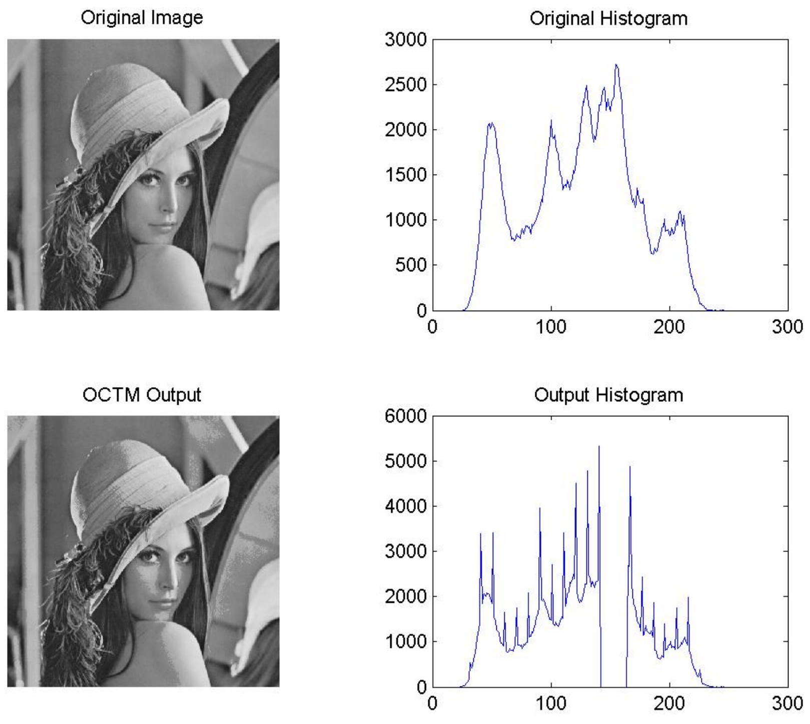
The following 3 test cases apply the constraints stated above to try to produce enhanced images. There are 3 examples of different values of distortion that were tested. When d = 1, the image produced looks the same as the original image. As d increases, so does the contrast of the image, however, some distortion in the tone is noticeable.

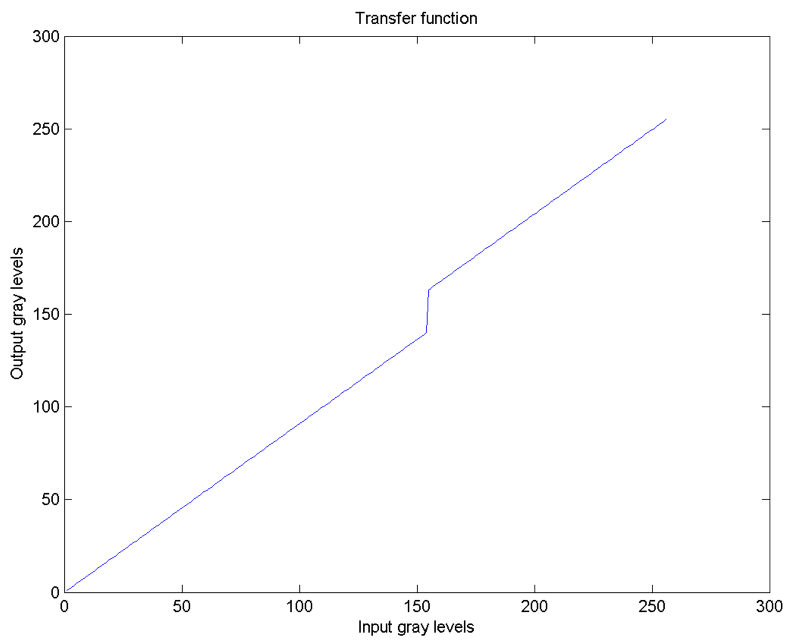
Test 1: d = 1



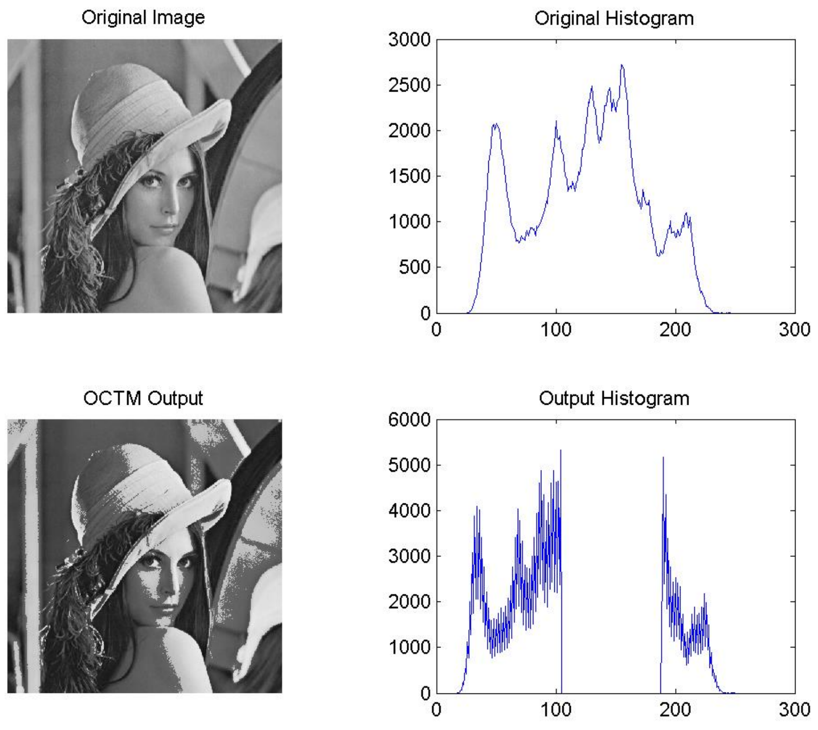

Transfer function
Output gray levels
Input gray levels
300
250
200
150
100
50
0
0
50
100
150
200
250
300

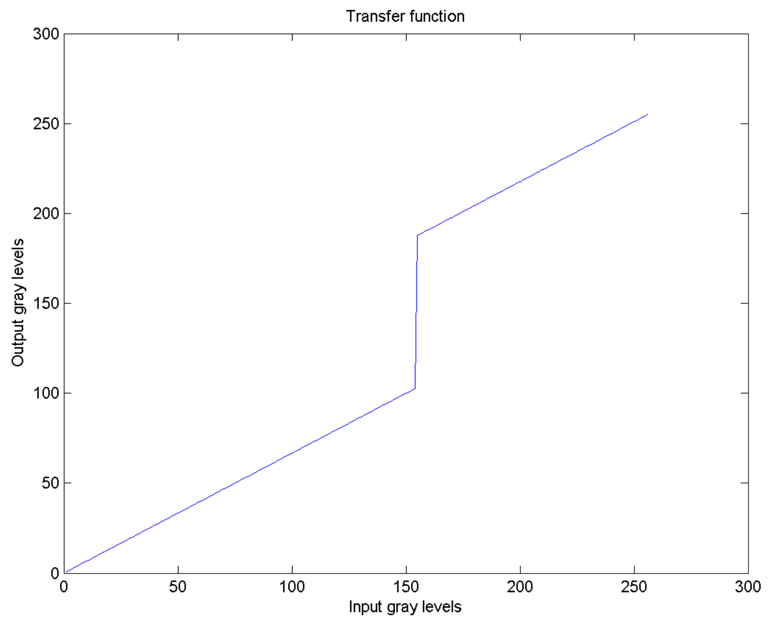
Test 2: d = 1.1


Original Image
Original Histogram
OCTM Output
Output Histogram
3000
2500
2000
1500
1000
500
0
0
100
200
300
6000
5000
4000
3000
2000
1000
0
0
100
200
300


Transfer function
Output gray levels
Input gray levels
0
50
100
150
200
250
300
0
50
100
150
200
250
300

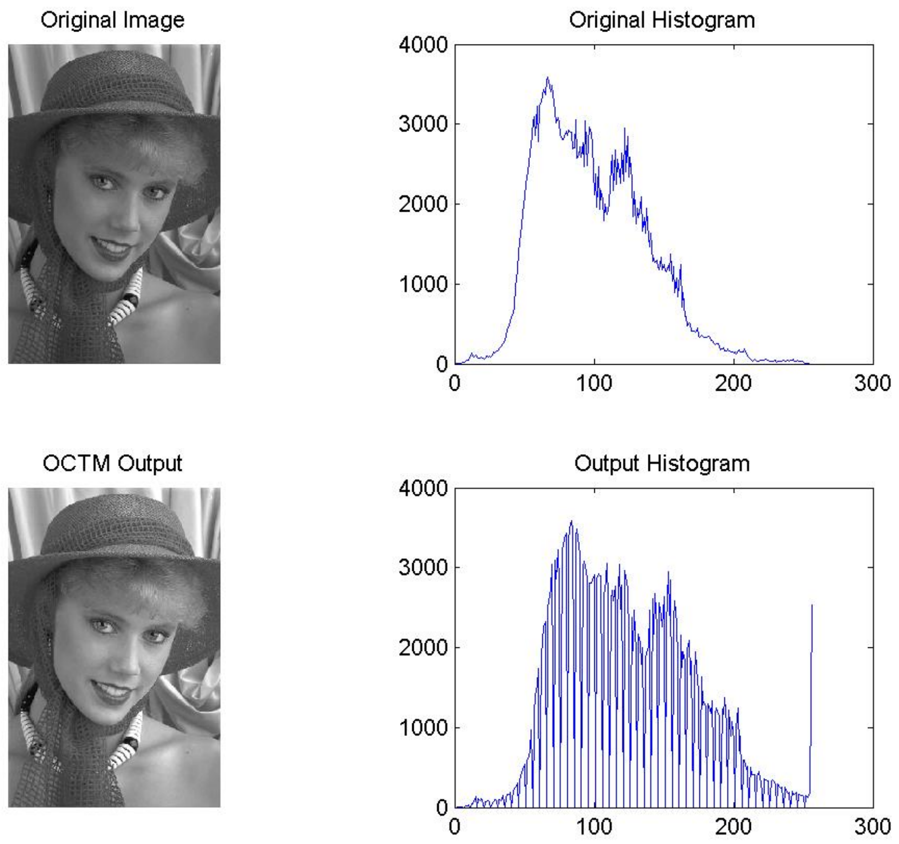
Test 3: d = 1.5

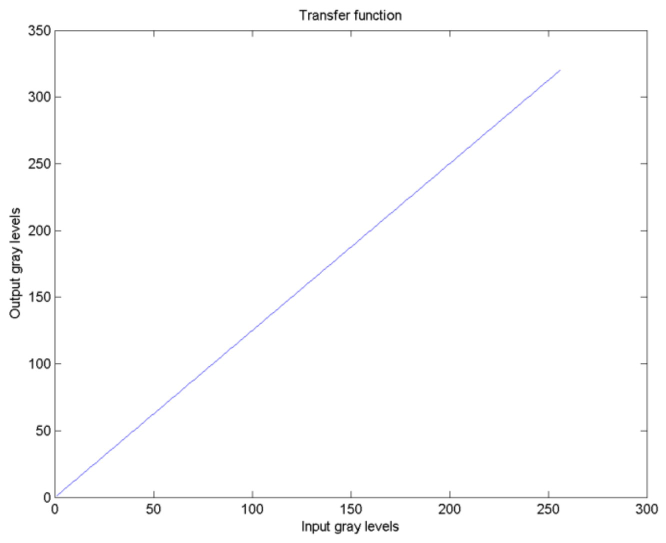

Original Image
Original Histogram
OCTM Output
Output Histogram
3000
2500
2000
1500
1000
500
0
0
100
200
300
6000
5000
4000
3000
2000
1000
0
0
100
200
300



The following 4 test cases show the results of a different image with various values of distortion specified in the lower bound constraint. When the value of d < 1, the transfer function shows that the output level goes beyond the maximum value, therefore there is clipping of the image and it appears to be brighter than intended. When the value of d >1, the transfer function maps correctly but some distortion may be visible at high values of d.

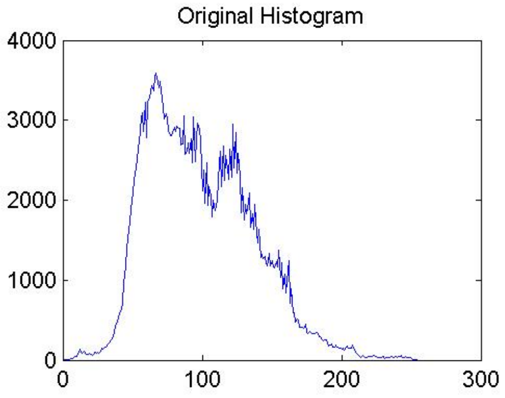
Test 4: d = 0.8

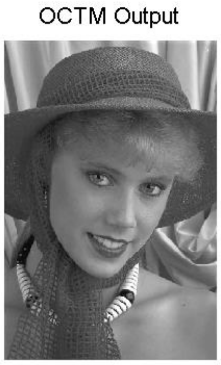

Original Image
Original Histogram
4000
3000
2000
1000
0
0
100
200
300
OCTM Output
Output Histogram
4000
3000
2000
1000
0
0
100
200
300

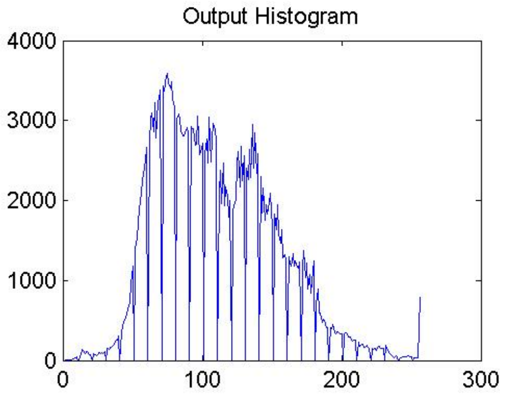

Transfer function
350
300
250
200
150
100
50
0
Output gray levels
0
50
100
150
200
250
300
Input gray levels

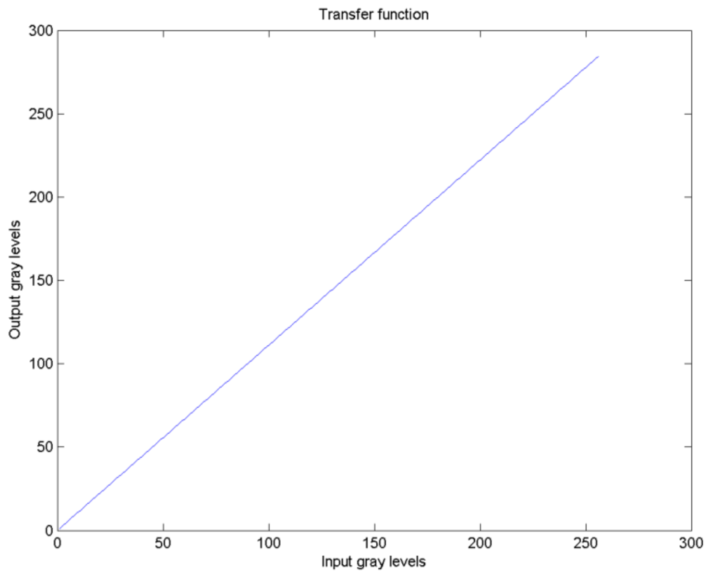
Test5: d = 0.9

Original Image


Original Histogram
4000
3000
2000
1000
0
0
100
200
300

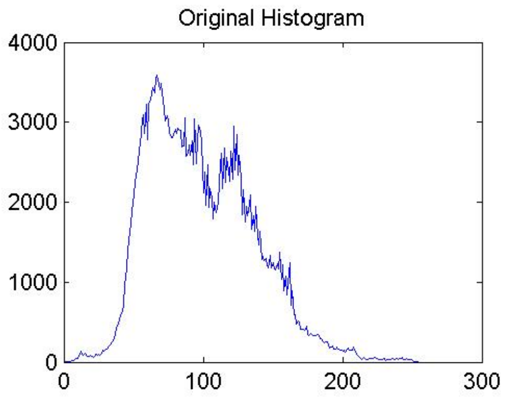

OCTM Output


Output Histogram
4000
3000
2000
1000
0
0
100
200
300

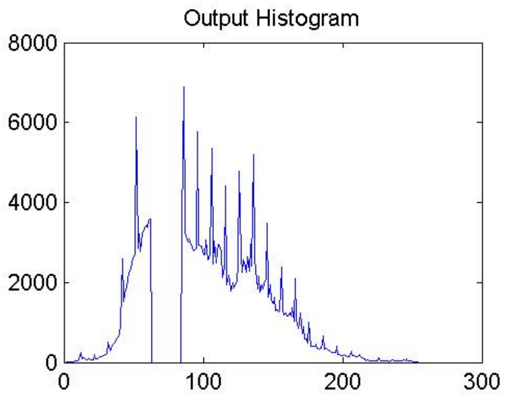

Transfer function
Output gray levels
Input gray levels
300
250
200
150
100
50
0
0
50
100
150
200
250
300

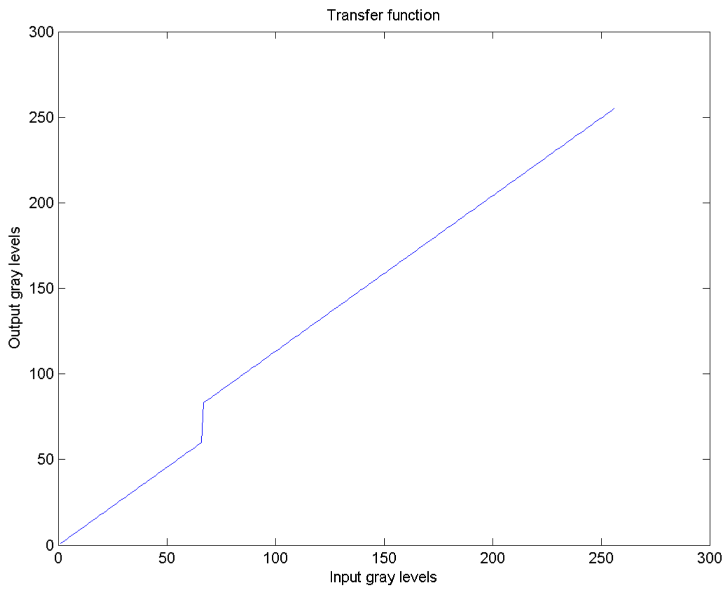
Test 6: d = 1.1

Original Image


Original Histogram
4000
3000
2000
1000
0
0
100
200
300

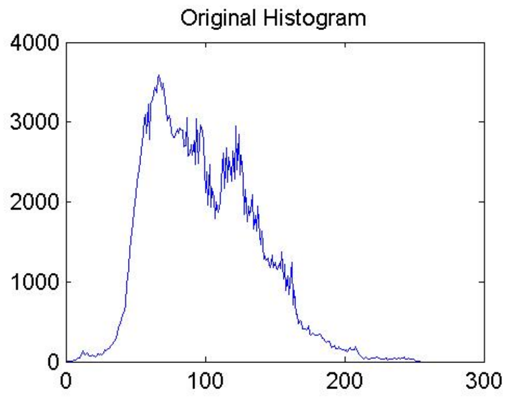
OCTM Output

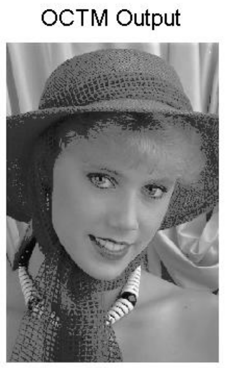

Output Histogram
8000
6000
4000
2000
0
0
100
200
300

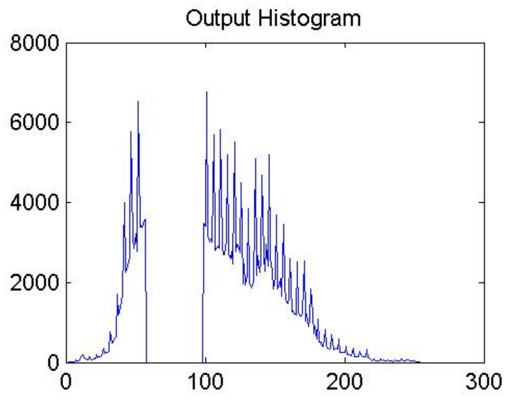

Transfer function
300
250
200
150
100
50
0
0
50
100
150
200
250
300
Input gray levels
Output gray levels

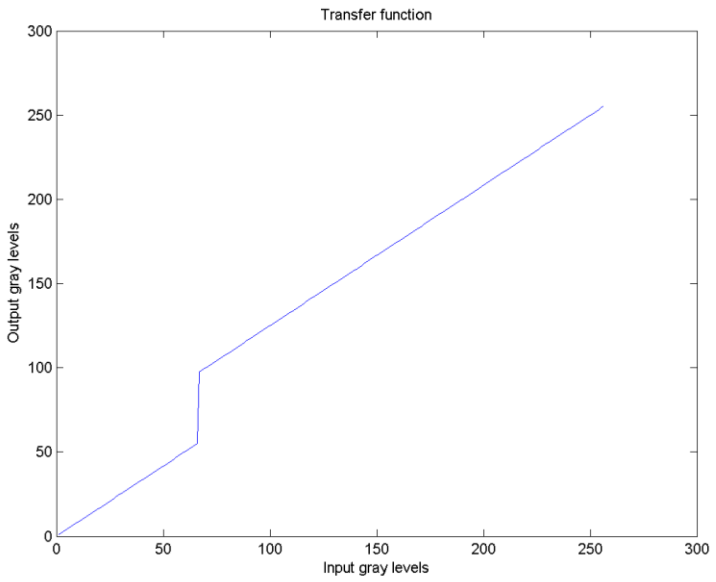
Test 7: d = 1.2

Original Image


Original Histogram
4000
3000
2000
1000
0
0
100
200
300


OCTM Output


Output Histogram
8000
6000
4000
2000
0
0
100
200
300


Transfer function
Output gray levels
Input gray levels
0
50
100
150
200
250
300

Additional constraints – Average intensity

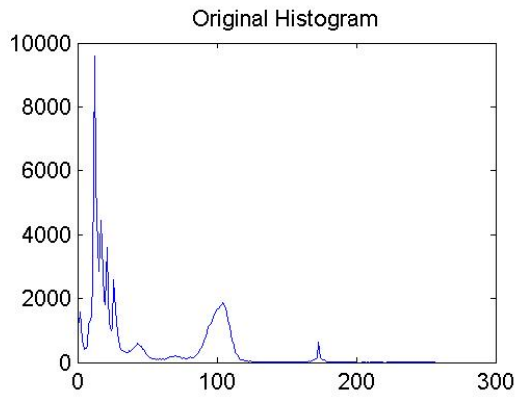
Some additional constraints may be applied to OCTM in order to improve the results of the images. In this case, the average intensity of the image was taken into consideration. The restriction can be added by limiting the intensity of the output image to be within a certain range of the original input image. This is shown by the following equations:

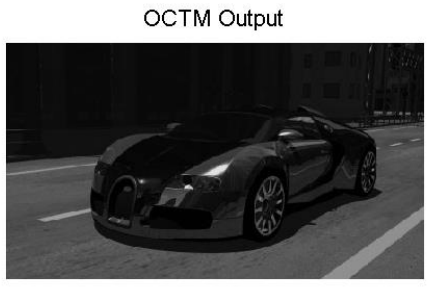
$$\left| \frac{L}{\text{Ł}} \sum_{0 \le i < L} p_i \sum_{0 \le j \le i} s_j - \sum_{0 \le i < L} p_i i \right| \le \Delta_\mu$$

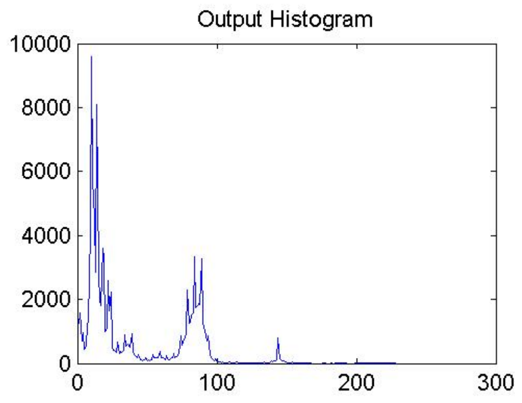
$$\sum_{0 \le i < L} \left| (L-1)^{-1} \sum_{0 \le j \le i} s_j - [i(L-1)^{-1}]^\gamma \right| \le \Delta$$

In the Matlab implementation, the value of the range the intensity can take is represented by the variable *diff*.

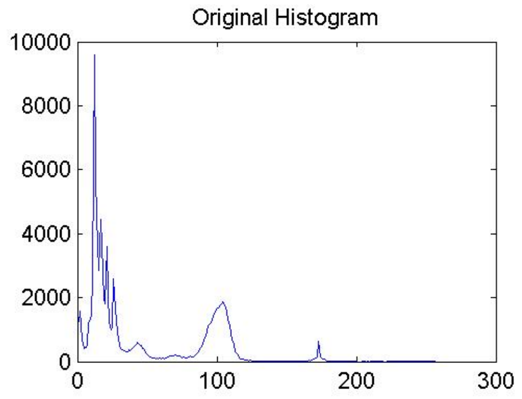
Test 1: d = 3, diff = 0

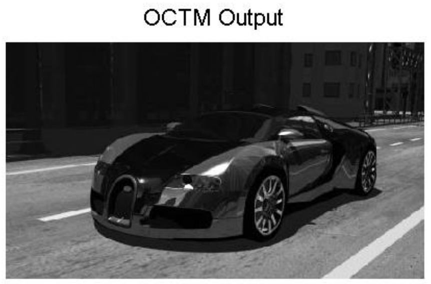


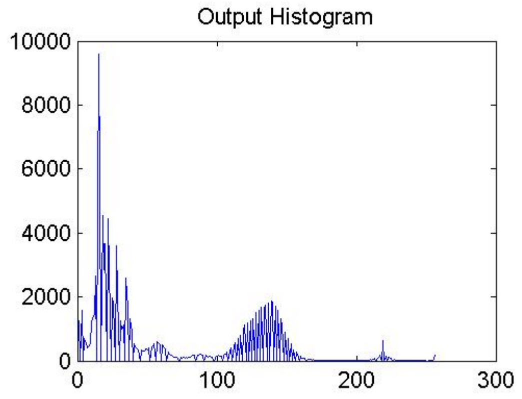




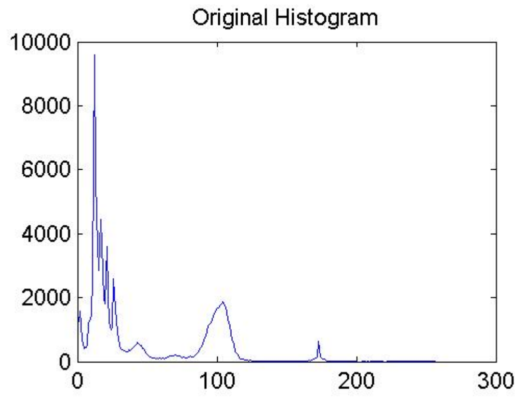
Test 2: d = 3, diff = 1

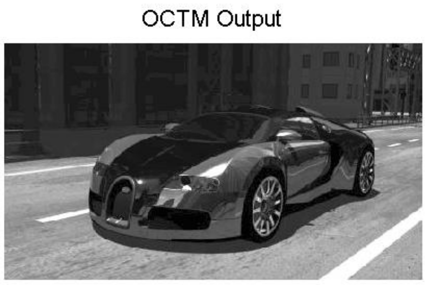


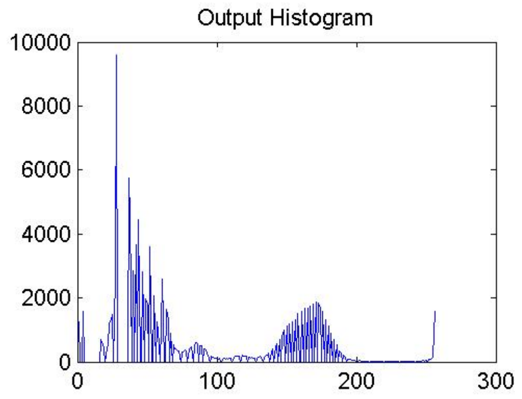




Test 3: d = 3, diff = 2t

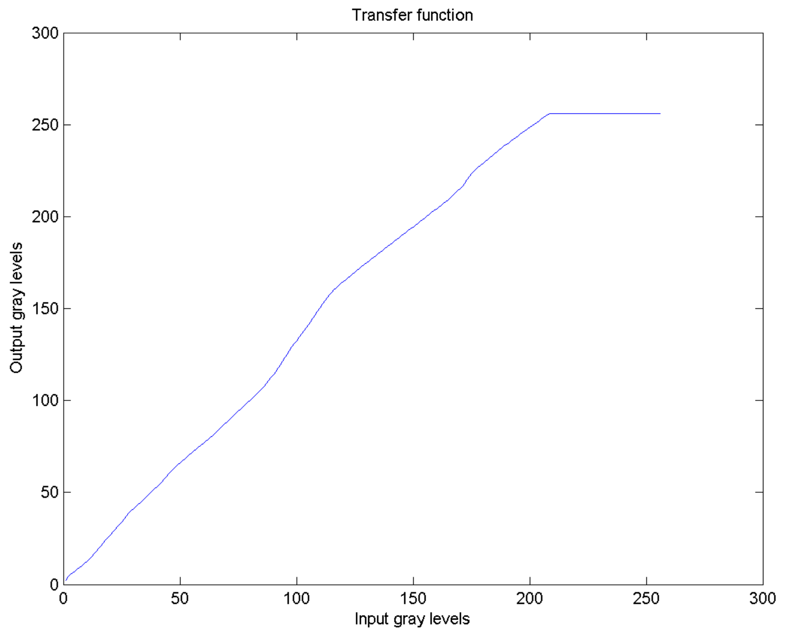






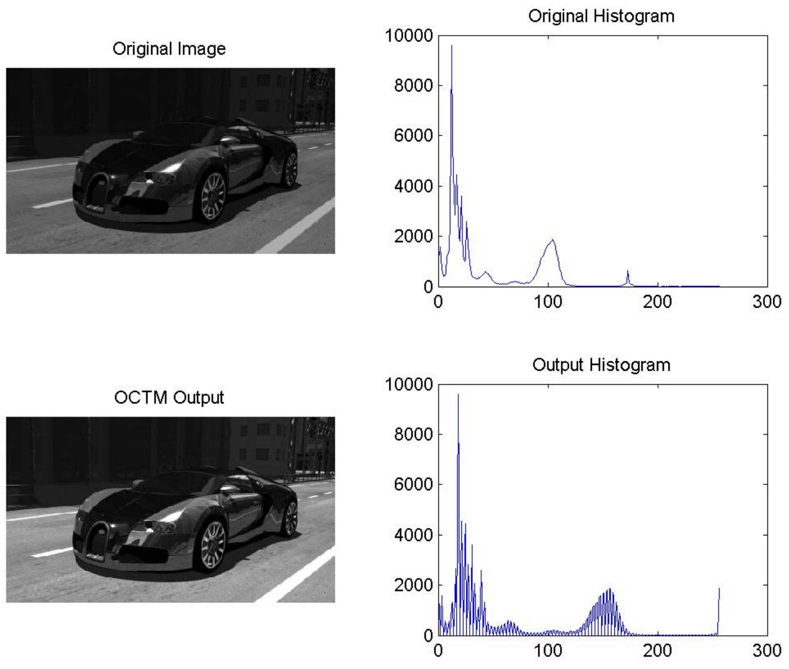
The differences between the images relative to the intensity range variable can be seen by the images above. While at diff = 0, the contrast of the image seemed to decrease, increasing the value seems to increase the contrast. Diff = 1 seems to be the best choice for this example since at diff = 2 the image may become too bright and slightly pixelated in the darker areas.

The transfer function at diff = 1 is shown here:



The value of the distortion that seemed to provide the best images was found to be 3 for this example. A different value of distortion is shown below.

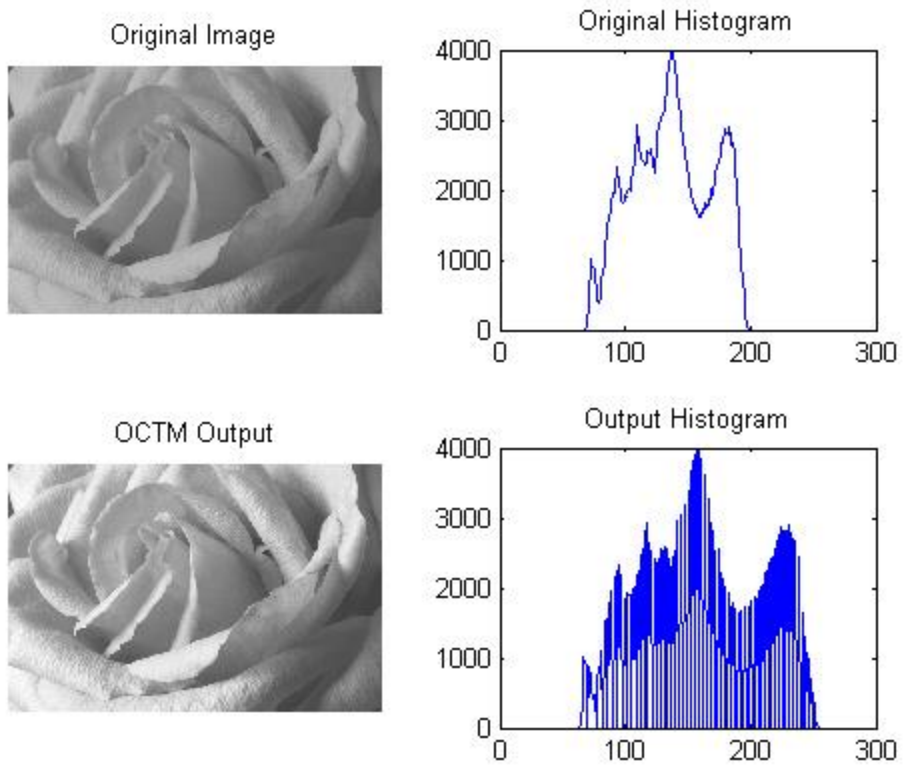
Test 4: d = 2, diff = 1

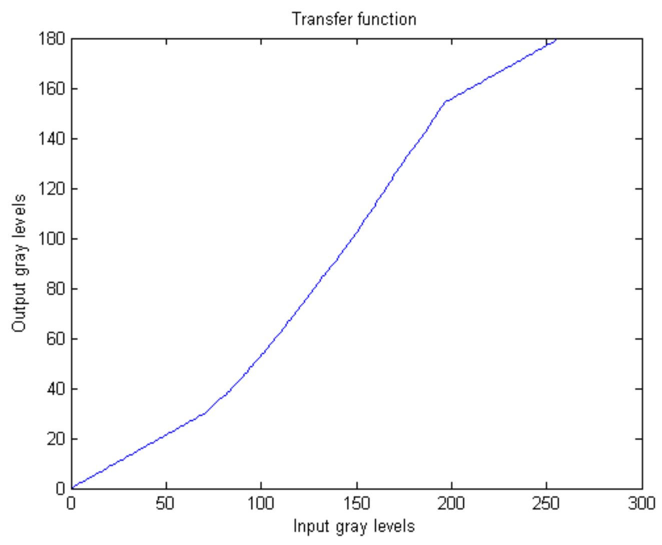


For each example of OCTM, the parameters can be adjusted to provide the best looking image. Therefore the image is highly dependent on these parameters for the constraint on the optimization of the contrast.

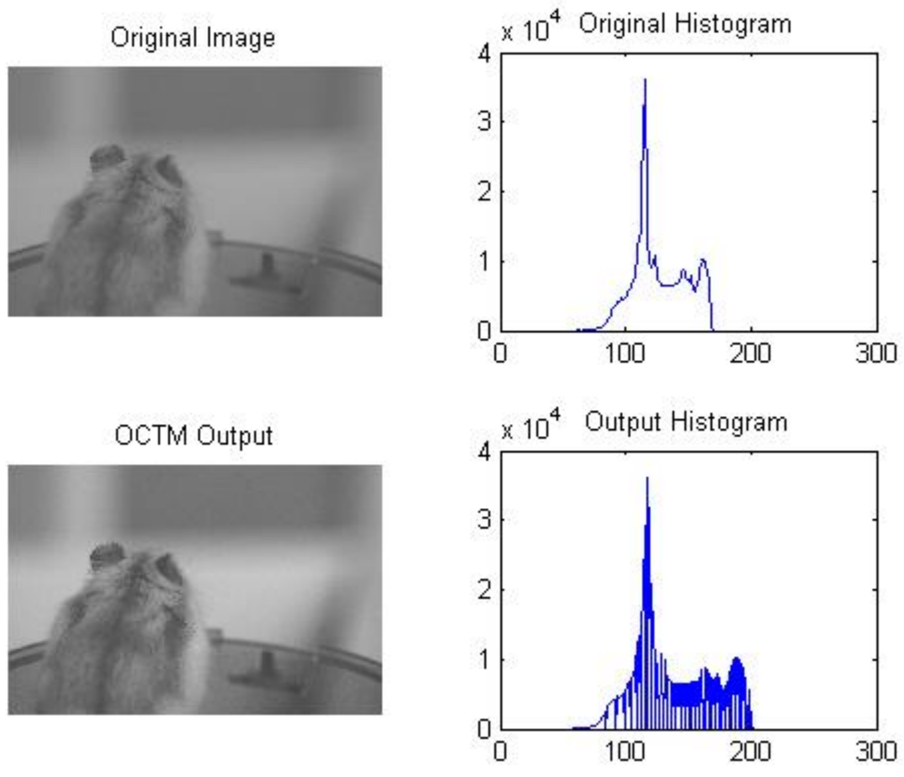
Here are a few more examples where the choice of parameters produced desirable results.

Test 5:

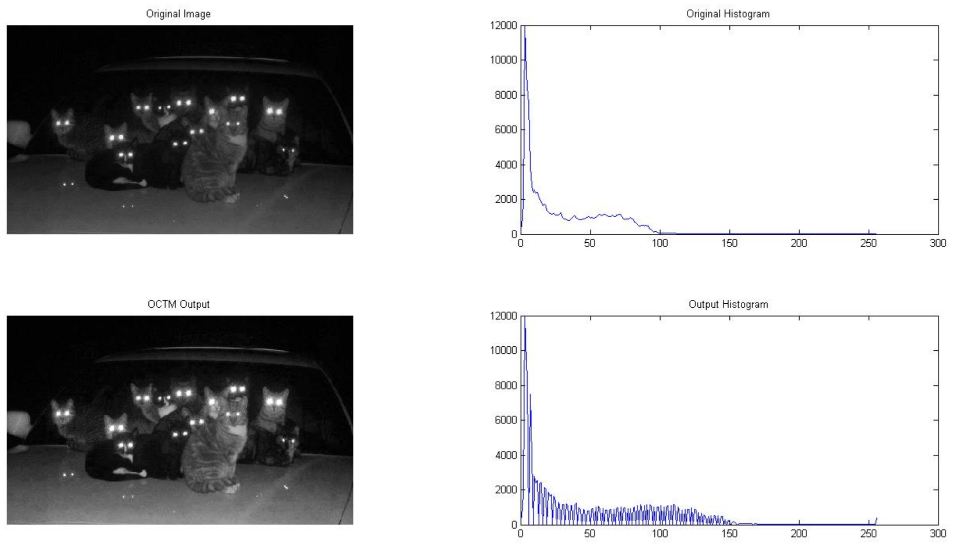




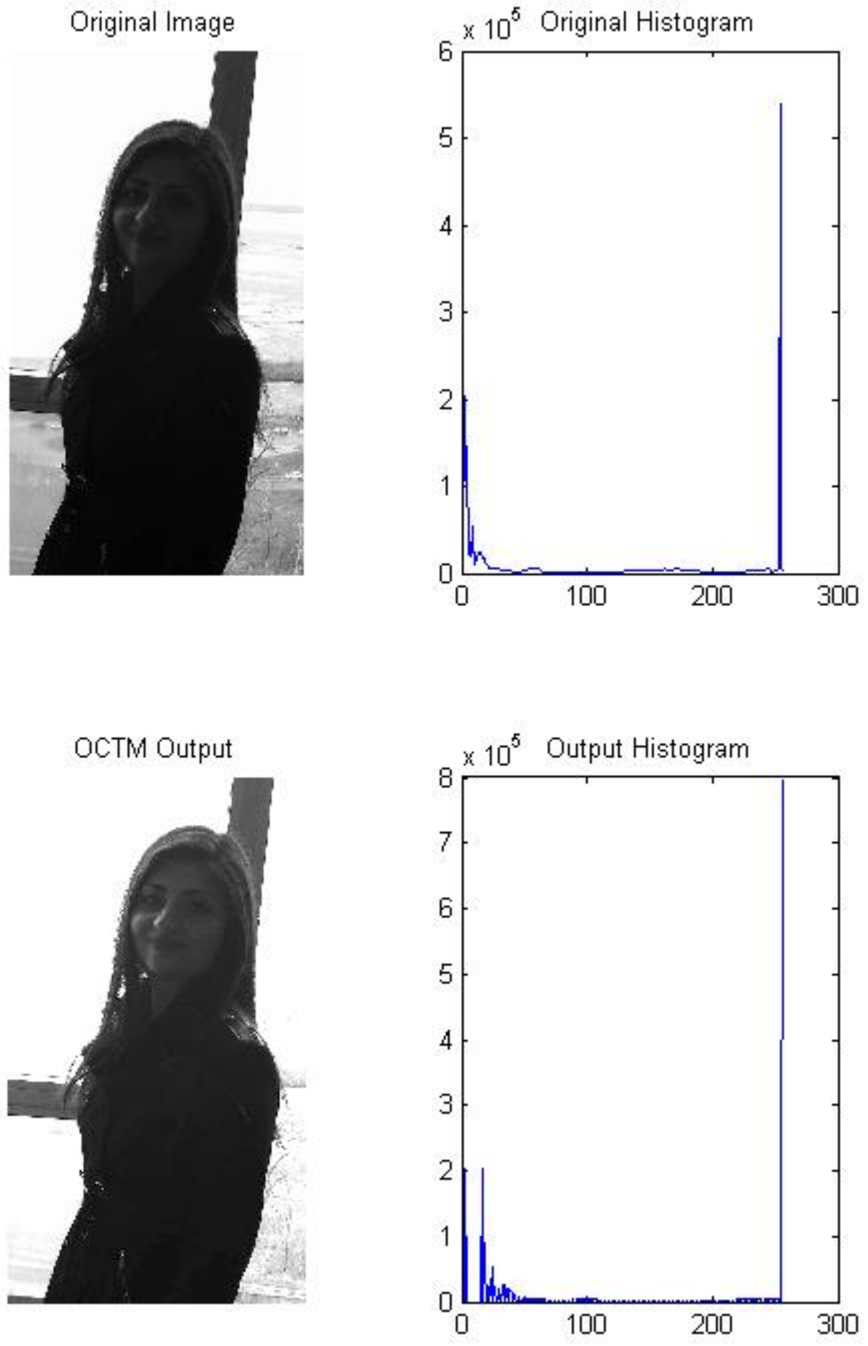
Test 6:


Original Image
x 10^4 Original Histogram
OCTM Output
x 10^4 Output Histogram
0
1
2
3
4
0
100
200
300

Test 7:


Original Image
Original Histogram
OCTM Output
Output Histogram
0
2000
4000
6000
8000
10000
12000
0
50
100
150
200
250
300

Test 8:


Original Image
x 10^5 Original Histogram
0
1
2
3
4
5
6
0
100
200
300
OCTM Output
x 10^5 Output Histogram
0
1
2
3
4
5
6
7
8
0
100
200
300

Matlab Code:

Histogram Equalization

```
clear all; close all; clc;

inputfile = 'low-contrast';
inputext = '.png';
input = [inputfile,inputext];
InputImage = imread(input);

[rows, columns, numberOfColorChannels] = size(InputImage);

% Convert to grayscale image
if numberOfColorChannels == 3
    InputImage = rgb2gray(InputImage);
end

pixelTotal=size(InputImage,1)*size(InputImage,2);

% Calculating number of pixels for each occurence of a gray level
freq=zeros(256,1);

for i=1:size(InputImage,1)
    for j=1:size(InputImage,2)
        value=InputImage(i,j);
        freq(value+1)=freq(value+1)+1;
    end
end

sum=0;
no_bins=255;

% Cumulative distribution probability
cdf=zeros(256,1);
s=zeros(256,1);

for i=1:256
   sum=sum+freq(i);
   cdf(i)=sum;
   s(i)=round((cdf(i)/pixelTotal)*no_bins); %transfer function
end

equalizedImage=uint8(zeros(size(InputImage,1),size(InputImage,2)));

for i=1:size(InputImage,1)
    for j=1:size(InputImage,2)
            equalizedImage(i,j)=s(InputImage(i,j)+1);
            %mapping gray level of original image to equalized image
    end
end

name = ['histeq_',inputfile];
```

```
figure('Position', [100, 425, 1400, 400])
subplot(1,2,1);
imshow(InputImage);
title('Original Image');
subplot(1,2,2);
plot(imhist(InputImage));
title('Histogram of Original');
print([name,'_original'],'-dpng');
figure('Position', [100, 0, 1400, 400])
subplot(1,2,1);
imshow(equalizedImage);
title('Histogram equalization');
subplot(1,2,2);
plot(imhist(equalizedImage));
title('Histogram of Equalized Image');
print([name,'_enhanced'],'-dpng');
```

Optimal Contrast Tone Mapping

```
clc; clear all; close all;

inputfile = 'lcflower';
inputext = '.jpg';
input = [inputfile,inputext];
input_img = imread(input);
[rows, columns, numberOfColorChannels] = size(input_img);

if numberOfColorChannels == 3
    input_img = rgb2gray(input_img);
end

array_hist = imhist(input_img);   % probability from the histogram

% constraint parameters
L = 256;
Lbar = 256;
[x,y] = size(input_img);    %size of the input image is extracted
npixels = x*y;         %the total number of pixels calculated
probability = array_hist/npixels;  %probability of the occurance of a pixel
diff = 1; % for average intensity constraint

% constraints
I=[1:L];

A = [ones(1,256)                        % fundamental constraints
    triu(repmat(probability',L,1))];    % average intensity constraint
%   -triu(repmat(probability',L,1))];

b = [255                                % fundamental constraints
    diff+probability.*I'];              % average intensity constraint
%   diff-probability.*I'];

% linprog arguements
```

```
Aeq = []; % empty matrix
Beq = []; % empty matrix
distortion = 2.5;    % distortion factor
lower_bound = (1/distortion)*ones(1,L);
upper_bound = Lbar*ones(1,L);
% calculation of the step increment
Sj = linprog(-probability,A,b,Aeq,Beq,lower_bound,upper_bound);
output = input_img; % input image is set up as a new variable in order to be
used in the re allocation of pixels.
% for loop, used to re-allocate the pixels
for i=1:x
    for j=1:y
        output(i,j) = (sum(Sj(1:input_img(i,j)))+0.5);
    end
end

name =
['octm_avgint_',inputfile,'_d',num2str(distortion),'_diff',num2str(diff)];
figure(1)
subplot(2,2,1);
imshow(input_img);
title('Original Image');
subplot(2,2,2)
plot(imhist(input_img));
title('Original Histogram');
subplot(2,2,3);
imshow(output);
title('OCTM Output');
subplot(2,2,4);
plot(imhist(output));
title('Output Histogram');
% print(name,'-dpng');

% TRANSFER FUNCTION
% real one using Sj values
X=(1:L)';
Y=zeros(L,1);
for i=1:256
    Y(i,1)=sum(Sj(1:i));
end

figure(2)
plot(X,Y);
title('Transfer function');
xlabel('Input gray levels');
ylabel('Output gray levels');
% print([name,'_TRANS2'],'-dpng');
```