Image Enhancement and Filtering

Unit 2

Image Enhancement in the Spatial Domain

- Image enhancement approaches fall into two broad categories:
  - ❖**spatial domain methods and**
  - ❖**frequency domain methods.**
- The term spatial domain refers to the image plane itself, and approaches in this category are based on direct manipulation of pixels in an image.
- Frequency domain processing techniques are based on modifying the Fourier transform of an image.
- The term spatial domain refers to the aggregate of pixels composing an image.
- Spatial domain methods are procedures that operate directly on these pixels.
- Spatial domain processes will be denoted by the expression

  g (x, y) = T [f (x, y)]

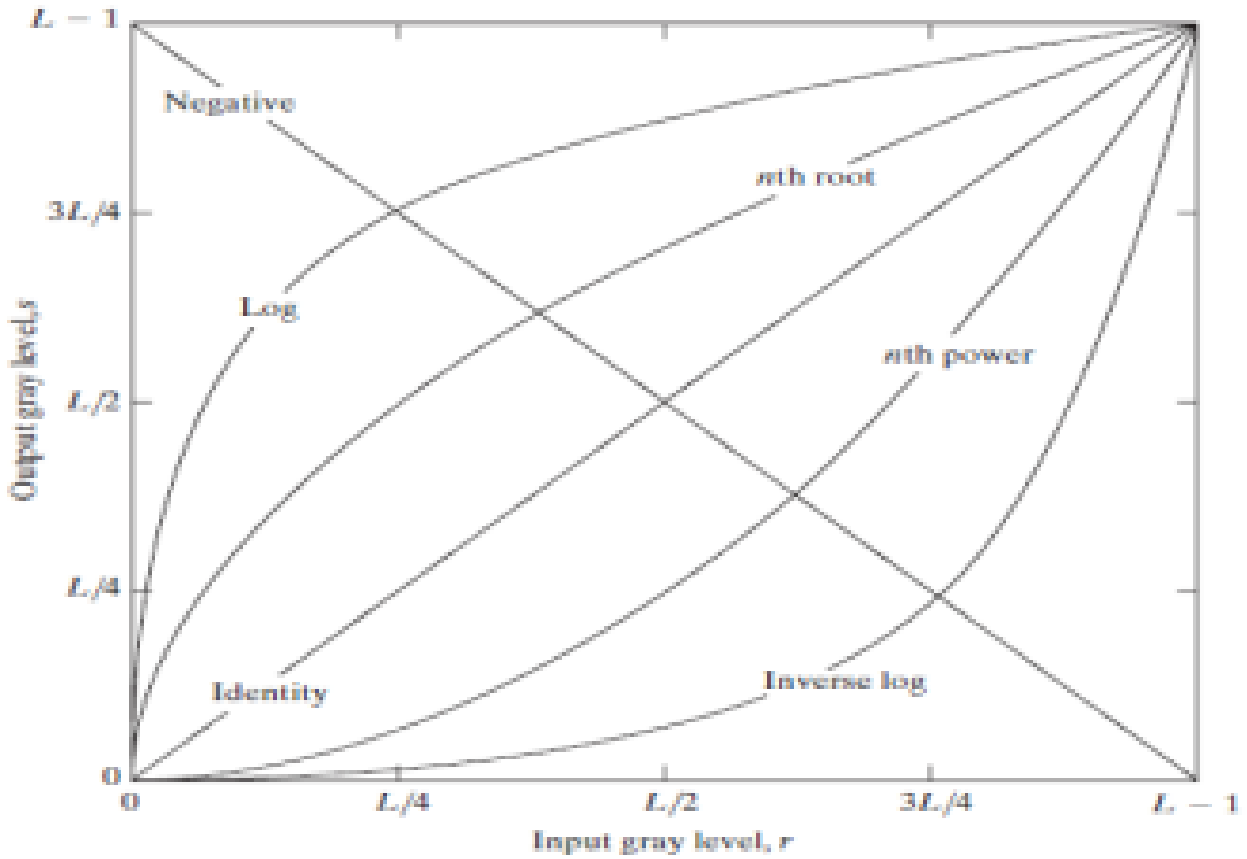
  where f(x, y) is the input image, g(x, y) is the processed image, and T is an operator on f, defined over some neighborhood of (x, y).

  T can operate on a set of input images, such as performing the pixel-by-pixel sum of K images for noise reduction.

Basic Gray Level Transformations

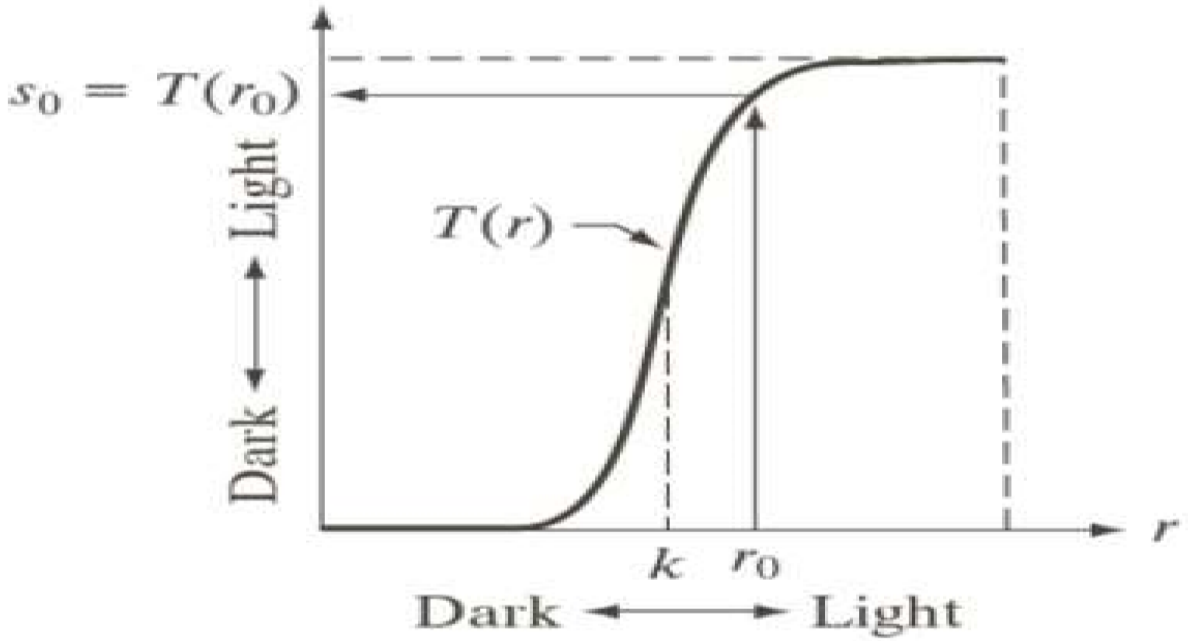
- Point transformation
- Linear (Negative and Identity) transformation
- Log Transformations
- Power-Law Transformations

cont.


L − 1
3L/4
L/2
L/4
0
Output gray levels
Negative
nth root
Log
nth power
Identity
Inverse log
0
L/4
L/2
3L/4
L − 1
Input gray level, r

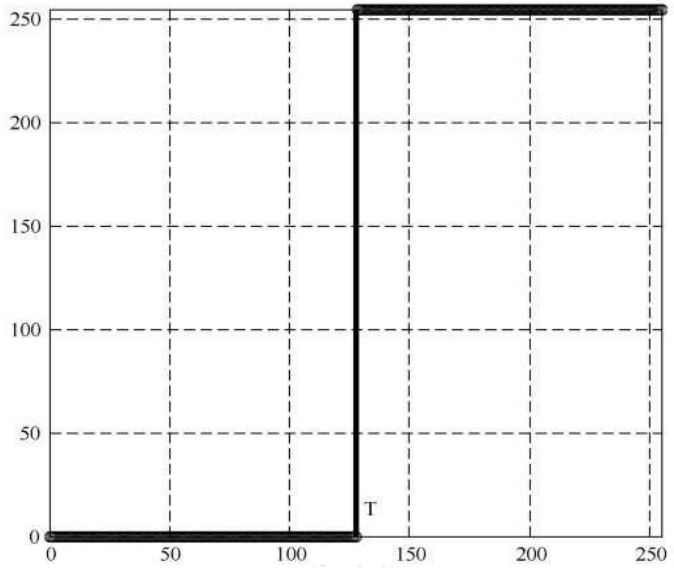
Contrast Stretching

- The goal of the contrast-stretching transformation is to enhance the contrast between different parts of an image, that is, enhances the gray contrast for areas of interest, and suppresses the gray contrast for areas that are not of interest.
- Graph below shows two possible functions:



Thresholding

- Grayscale Threshold Transform converts a grayscale image into a black and white binary image.
- The user specifies a value that acts as a dividing line.
- If the gray value of a pixel is smaller than the dividing, the intensity of the pixel is set to 0, otherwise it is set to 255.
- The value of the dividing line is called the ***threshold***. The grayscale threshold transform is often referred to as thresholding, or binarization



Example1: find the threshold (black and white) image of input image

1	6	4
7	9	4
10	14	5

Solution: The maximum value of image pixel (L) = 16, n = 4 (*n is the number of bits and L is calculate in term of* 2^n)

T = L/2 = 16/2 = 8
s = (L-1), if (r >= T)
s = 0, if (r < T)

0	0	0
0	15	0
15	15	0

Example 2: Clipping with $r_1 = 3$ and $r_2 = 8$

1	6	4
7	9	4
10	14	5

Solution: here, L = 16

S = (L-1), if (3<=r<=8)

S = 0, otherwise

0	15	15
15	0	15
0	0	15

Linear (Negative and Identity) transformation

- The second linear transformation is negative transformation, which is invert of identity transformation. In negative transformation, each value of the input image is subtracted from the (L-1) and mapped onto the output image.
- In this case the following transition has been done.
- $s = (L – 1) – r$
- *Where, L = highest intensity value, r = intensity of each pixel*

Example: find the negative digital image of given input image

1	5	4
3	7	6
5	8	9

Solution: the highest intensity of this image is L = 16

S = (L-1) – r = (16-1) – r = 15-r

14	10	11
12	8	9
10	7	6

Let's examine the image bellow:

Input Image

Output Image

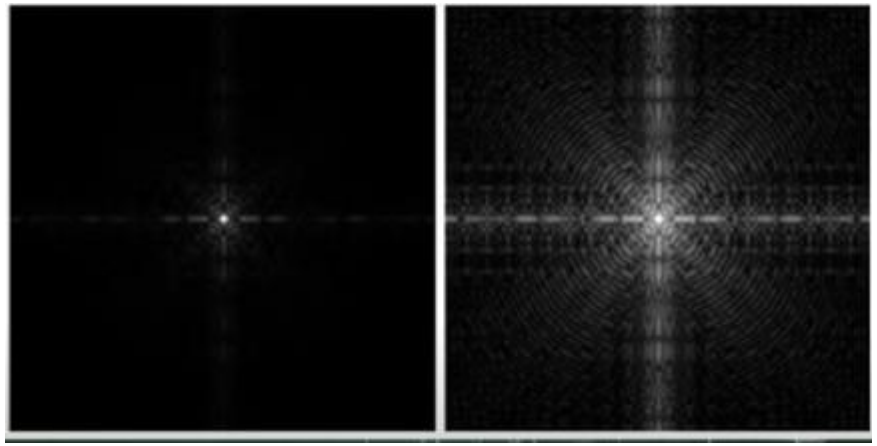
Log Transformations

- The general form of the log transformation

  $s = c \log (1 + r)$

  where c is a constant, and it is assumed that $r \geq 0$.

- The shape of the log curve in Figure shows that this transformation maps a narrow range of low gray-level values in the input image into a wider range of output levels.
- The opposite is true of higher values of input levels.
- We would use a transformation of this type to expand the values of dark pixels in an image while compressing the higher-level values.
- The opposite is true of the inverse log transformation.



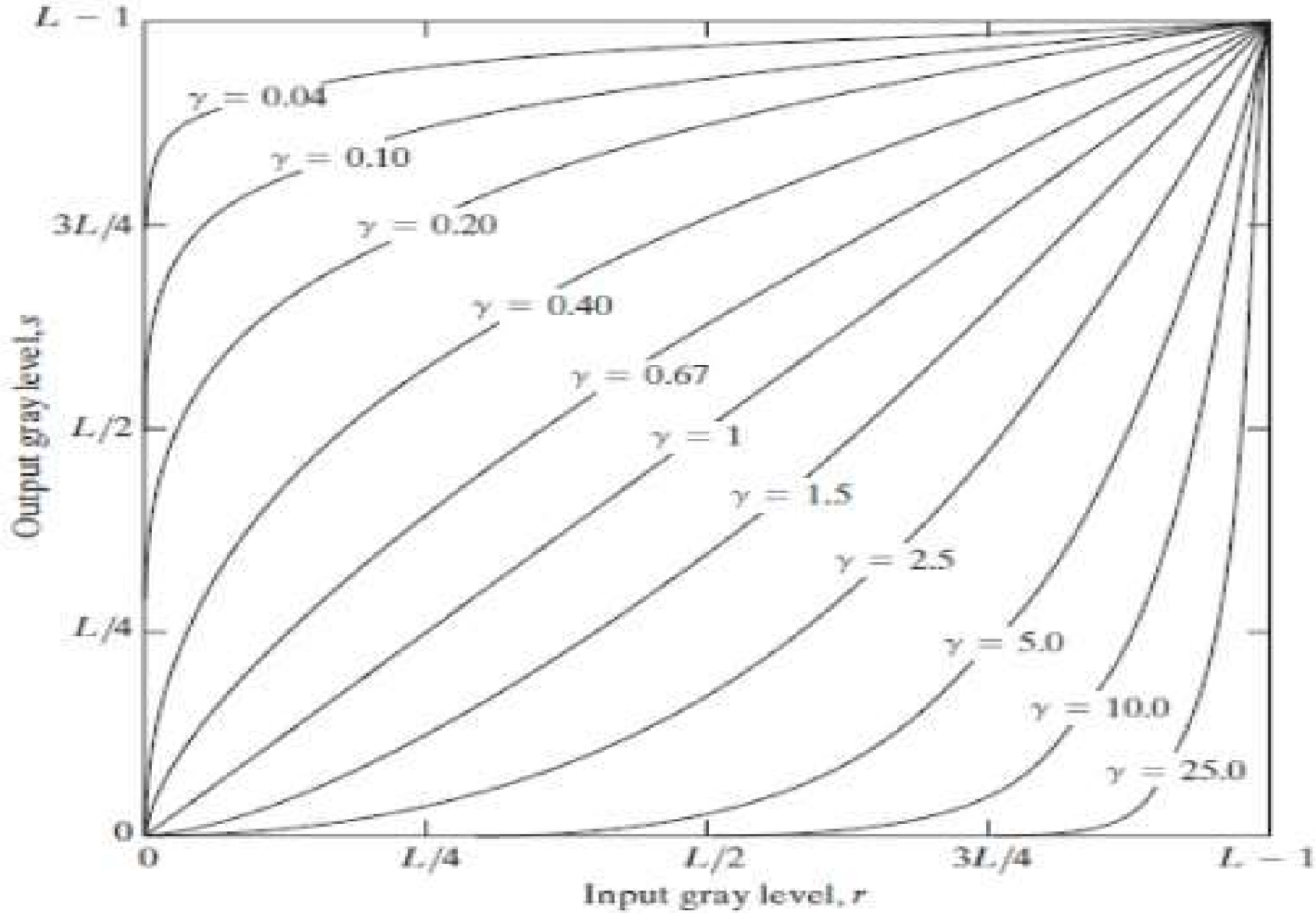
Power-Law Transformations(Gamma Transformation)

- transformation can be done by the expression:

$$S = C * r^{\gamma}$$

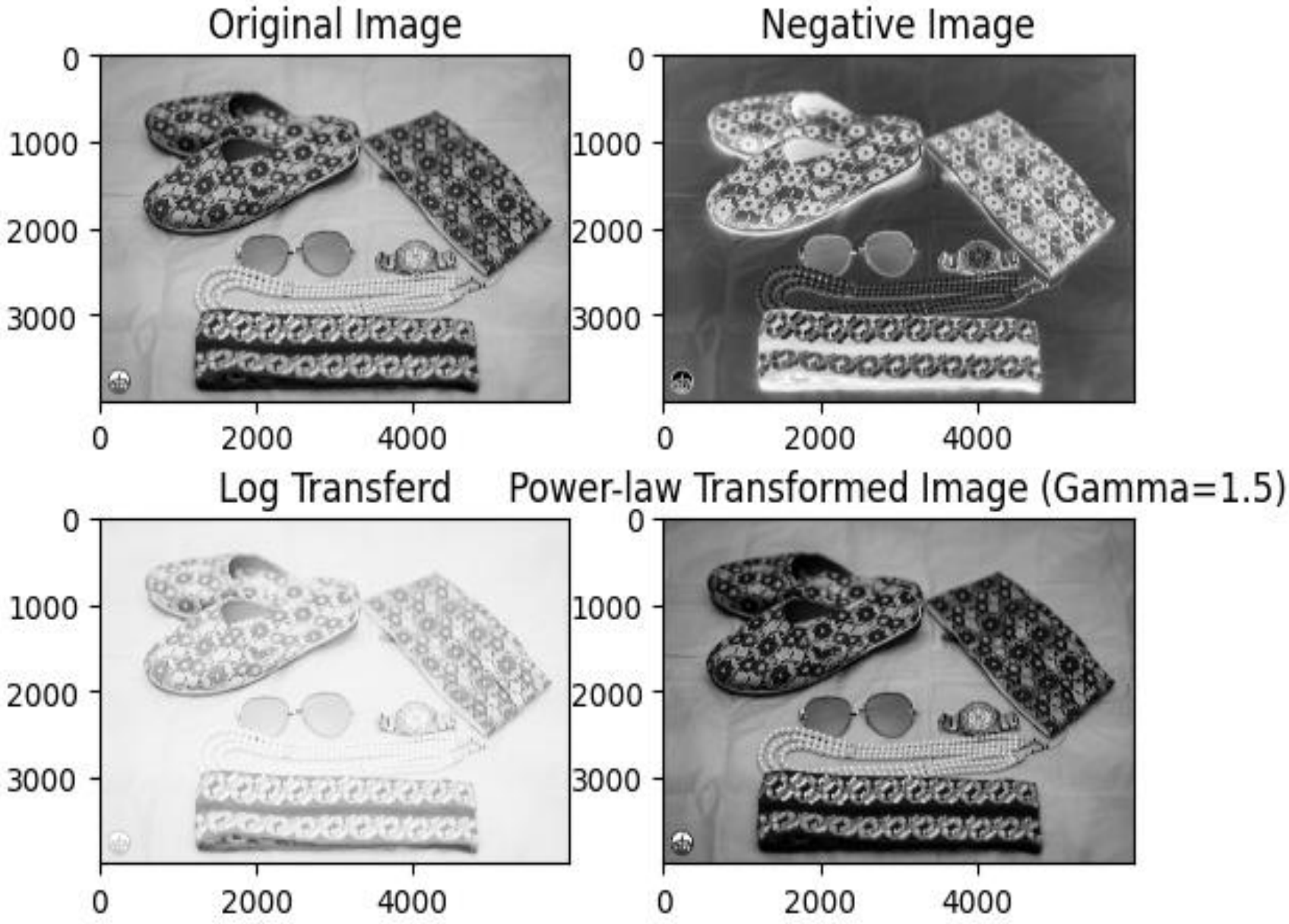
Where, C and γ are positive constant, symbol γ is called gamma, due to which this transformation is also known as *gamma transformation.*

- Variation in the value of γ varies the enhancement of the images. Different display devices (monitors) have their own gamma correction, that's why they display their image at different intensity.
- This type of transformation is used for enhancing images for different type of display devices. The gamma of different display devices is different.


L − 1
3L/4
L/2
L/4
0
Output gray level, s
γ = 0.04
γ = 0.10
γ = 0.20
γ = 0.40
γ = 0.67
γ = 1
γ = 1.5
γ = 2.5
γ = 5.0
γ = 10.0
γ = 25.0
0
L/4
L/2
3L/4
L − 1
Input gray level, r

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
# Load the image in grayscale
image = cv2.imread('image1.jpeg', cv2.IMREAD_GRAYSCALE)
negative_image = 255-image
plt.subplot(2, 2, 1)
plt.imshow(image,cmap='gray')
plt.title('Original Image')

plt.subplot(2, 2, 2)
plt.imshow(negative_image,cmap='gray')
plt.title('Negative Image')

c = 1  # Constant for scaling (adjust for desired effect)
log_image = c * np.log1p(image)


plt.subplot(2, 2, 3)
plt.imshow(log_image,cmap='gray')
plt.title('Log Transferd')
```

```python
# Parameters for the power-law transformation
gamma = 1.5  # Gamma value (adjust for desired effect)
c = 1.0     # Constant for scaling (adjust for desired effect)

# Apply power-law transformation
power_transformed_image = c * np.power(image.astype(np.float32) / 255.0, gamma)

plt.subplot(2, 2, 4)
plt.imshow(power_transformed_image,cmap='gray')
plt.title('Power-law Transformed Image (Gamma={})'.format(gamma))
```


Original Image
Negative Image
Log Transferd
Power-law Transformed Image (Gamma=1.5)
0
1000
2000
3000
0
2000
4000

Histogram

0	2	1	3	4
1	3	4	3	3
0	1	3	1	4
3	1	4	2	0
0	4	2	4	4

Image

Intensity	Number of Pixels
0	4
1	5
2	3
3	6
4	7

Histogram Table

Image Histogram

Histogram Equalization

- The histogram of a digital image, with intensity levels between 0 and (L-1), is a function h(r_k) = n_k , where r_k is the k_{th} intensity level and n_k is the number of pixels in the image having that intensity level.
- We can also normalize the histogram by dividing it by the total number of pixels in the image.

$$p(r_k) = \frac{n_k}{N^2}$$

- This p(r_k) function is the probability of the occurrence of a pixel with the intensity level r_k Clearly,

$$\sum p(r_k) = 1$$

Cont..

- To make the histogram equalization of an image first we have to calculate the **PDF** (Probability Density function) of all the pixels in the image.

$$h[i] = \frac{number\ of\ pixels\ of\ gray\ lavel\ i}{Total\ number\ of\ pixels}$$

- Then, we have to calculate the ***CDF*** (Cumulative distribution function) which is the summation of PDF.

$$H[j] = \sum_{i=0}^{j} h[i], \qquad (j = 0,1,2,3 \ldots .255)$$

- Now, the ***New CDF*** can be calculate, that is the histogram equalization of an image.

New CDF = CDF*(L-1)

- Where, L-1 is the upper value of gray level of given image.

Example: perform histogram equalization on the following 3-bit digital image. The gray level distribution of the image is given below:

Gray Level (r_k)	0	1	2	3	4	5	6	7
No. of pixel p_k	8	10	10	2	12	16	4	2

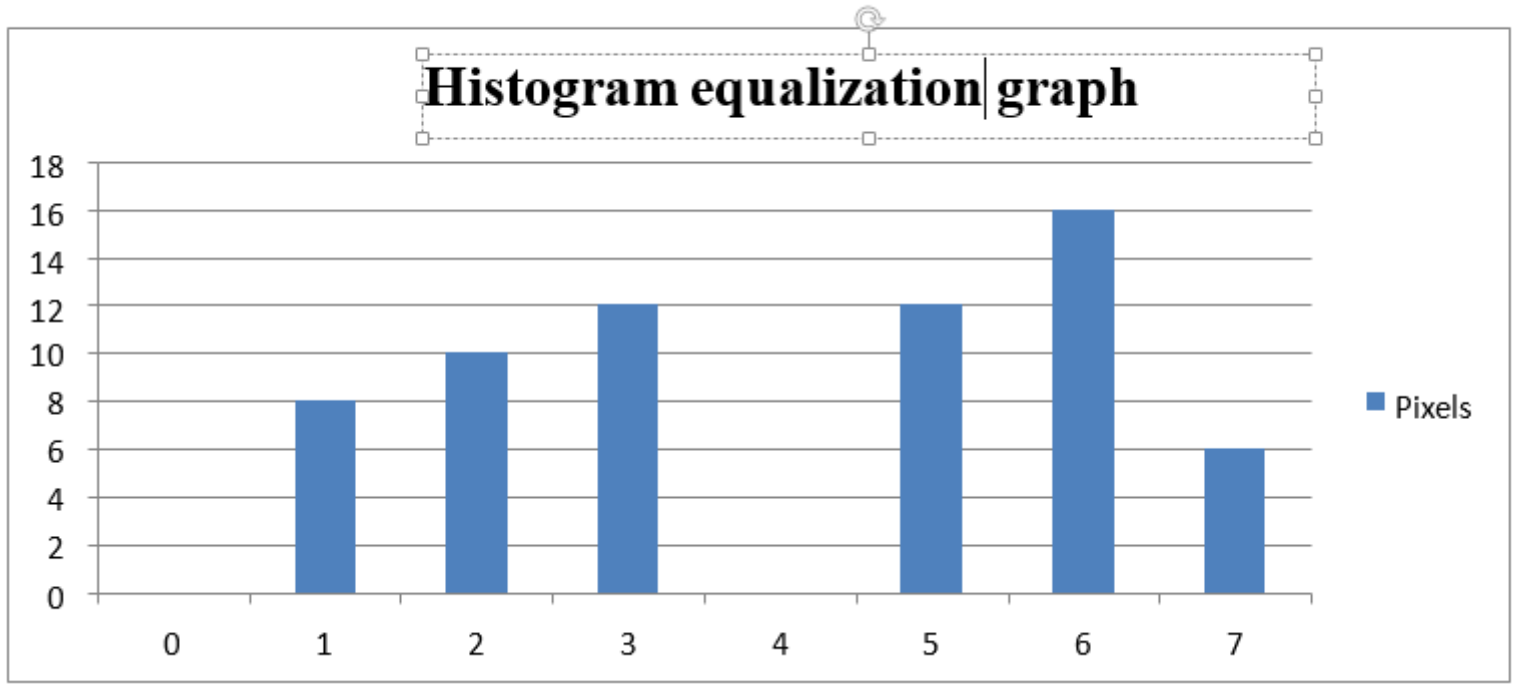
Total no of pixel (**N**) = 8+10+10+2+12+16+4+2 = 64

Upper value of gray level of 3-bit digital image (L) = 2^3 = 8 = **(L-1) = 7**

Now calculate the **PMF, CDF**, and **New CDF**

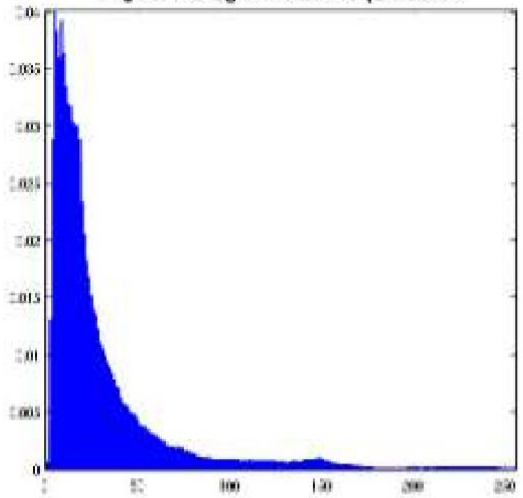
r_k	p_k	**PMF (p_k/N)**	**CDF**	**New CDF (CDF*(L-1))**	**Round off Value**	**No. of pixel**
0	8	0.125	0.125	0.875	1	8
1	10	0.15625	0.28125	1.96875	2	10
2	10	0.15625	0.4375	3.0625	3	10
3	2	0.03125	0.46875	3.28125	3	2
4	12	0.1875	0.65625	4.59375	5	12
5	16	0.25	0.90625	6.34375	6	16
6	4	0.0625	0.96875	6.78125	7	4
7	2	0.03125	1	7	7	2

Now we can draw a histogram graph for new CDF with their corresponding pixels.

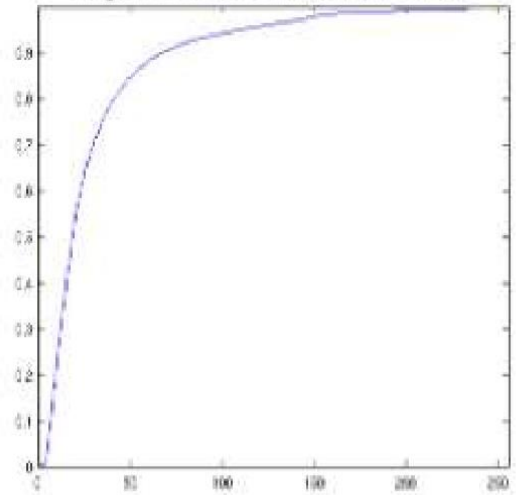


Input Image

Original Histogram before equalization

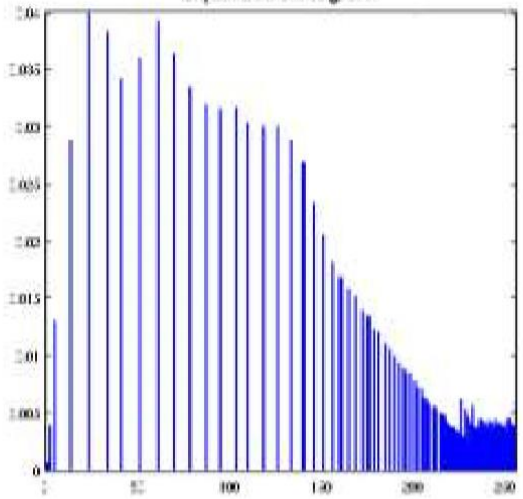


Original Cumulative Distribution Function

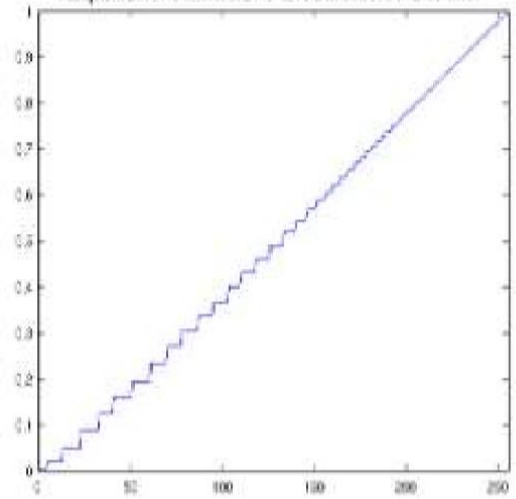


Output Image

Equalized histogram

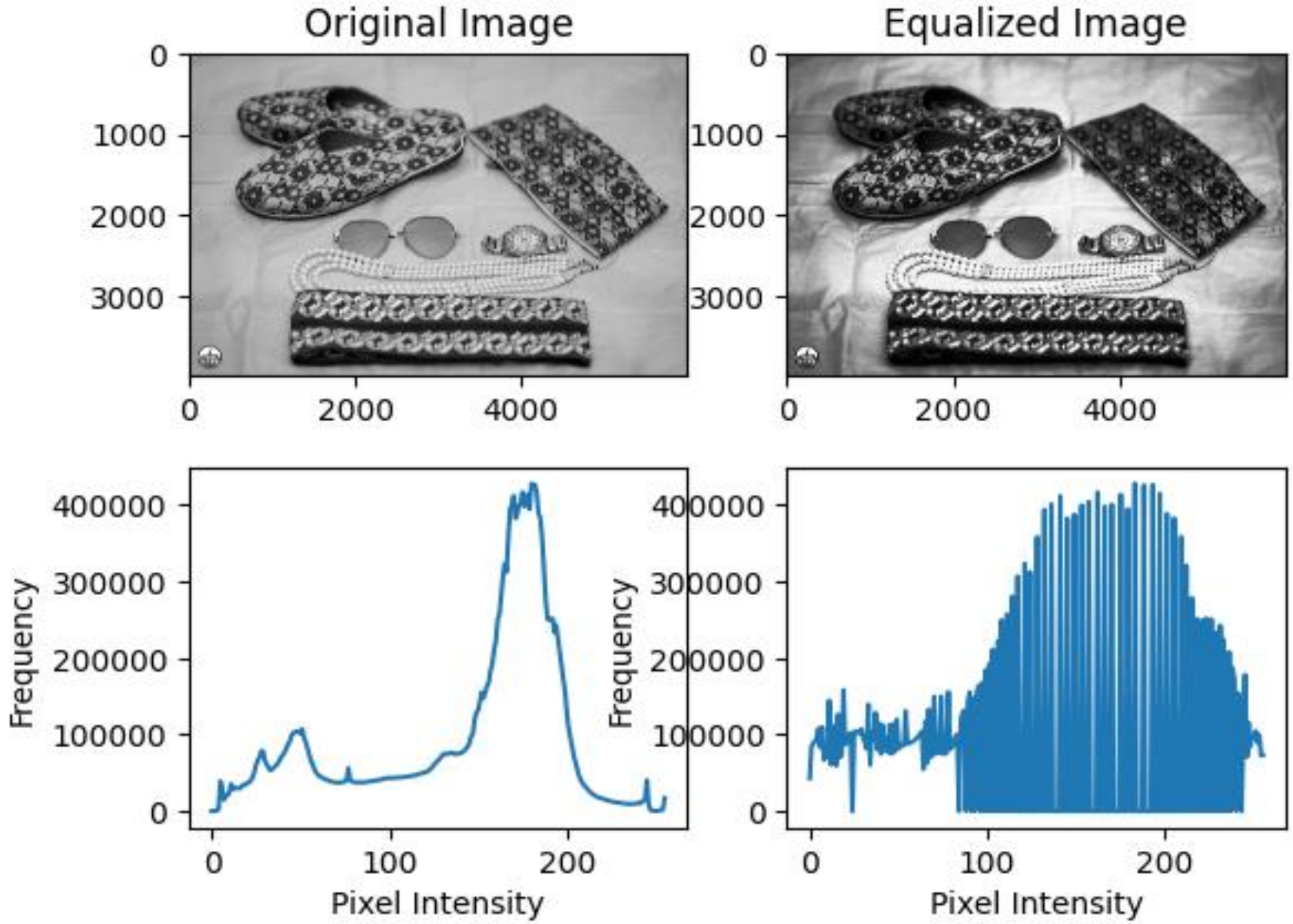


Equalized Cumulative Distribution Function



```
import cv2
import numpy as np
import matplotlib.pyplot as plt
# Load the image in grayscale
image = cv2.imread('image1.jpeg', cv2.IMREAD_GRAYSCALE)
#Apply histogram equalization
equalized_image = cv2.equalizeHist(image)
plt.subplot(2, 2, 1)
plt.imshow(image, cmap='gray')
plt.title('Original Image')
plt.subplot(2, 2, 2)
plt.imshow(equalized_image, cmap='gray')
plt.title('Equalized Image')
```

```
histogram = cv2.calcHist([image], [0], None, [256], [0, 256])
# Plot histogram
plt.subplot(2, 2, 3)
plt.plot(histogram)
plt.xlabel('Pixel Intensity')
plt.ylabel('Frequency')
histogram1 = cv2.calcHist([equalized_image], [0], None, [256], [0, 256])
# Plot histogram
plt.subplot(2, 2, 4)
plt.plot(histogram1)
plt.xlabel('Pixel Intensity')
plt.ylabel('Frequency')
plt.show()
```


Original Image
Equalized Image
0
1000
2000
3000
0
2000
4000
Frequency
400000
300000
200000
100000
0
100
200
Pixel Intensity

Histogram Matching

- Assume we have two images and each has its specific histogram. So we want to answer this question before going further, *is it possible to modify one image based on the contrast of another one*?
    - And the answer is YES.
- In fact, this is the definition of the histogram matching. In other words, given images A, and B, it is possible to modify the contrast level of A according to B.
- Histogram matching is useful when we want to unify the contrast level of a group of images. In fact, Histogram equalization is also can be taken as histogram matching, since we modify the histogram of an input image to be similar to the normal distribution.
- In order to match the histogram of images A and B, we need to first equalize the histogram of both images. Then, we need to map each pixel of A to B using the equalized histograms. Then we modify each pixel of A based on B.

Cont..

- Histogram matching, also known as histogram specification or histogram equalization matching, is a method used to adjust the total distribution of an image to match a specified histogram.
- This technique is particularly useful for transferring the total characteristics of one image (the reference image) to another image (the target image).
  - **Compute Histograms**: Calculate the histograms of both the target and reference images. Histograms represent the distribution of pixel intensities in an image. Each bin in the histogram corresponds to a range of pixel intensities, and the value of each bin represents the frequency of pixels with intensities falling within that range.
  - **Compute Cumulative Distribution Functions (CDFs)**: Compute the cumulative distribution functions (CDFs) of the histograms. The CDF at a given intensity level represents the cumulative probability of observing a pixel with intensity less than or equal to that level. CDFs are obtained by cumulatively summing up the histogram values.
  - **Normalize CDFs**: Normalize the CDFs so that they span the full intensity range of [0, 255]. This normalization step ensures that the CDF values are evenly distributed across the entire intensity range, which makes them suitable for use as mapping functions.
  - **Interpolation**: Perform interpolation to create a mapping function that maps pixel intensities from the target image to the corresponding intensities in the reference image. This mapping function is typically generated by interpolating between the CDFs of the target and reference images. The goal is to find a transformation that aligns the tonal distributions of the two images.
  - **Apply Mapping**: Apply the mapping function to the pixel intensities of the target image. This step involves replacing each pixel intensity in the target image with its corresponding intensity in the reference image, based on the mapping function generated in the previous step.
  - **Generate Matched Image**: The resulting image is the matched image, which has been transformed to match the tonal distribution of the reference image. The matched image retains the spatial details of the original target image but with tonal characteristics similar to those of the reference image.

0	2	1	3	4
1	3	4	3	3
0	1	3	1	4
3	1	4	2	0
0	4	2	4	4

input: Image A

Pixel Value	Histogram	Equalized Histogram
0	4	4
1	5	9
2	3	12
3	6	18
4	7	25

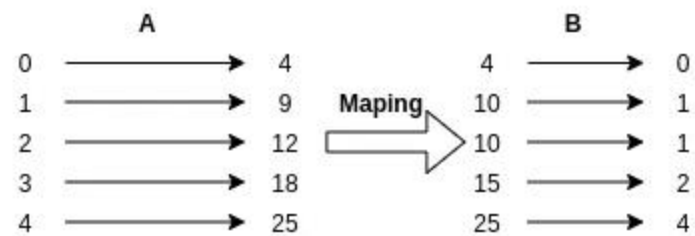
Equalized Histogram of A

2	1	2	1	0
3	3	2	4	4
1	3	2	4	4
0	0	3	2	1
1	3	1	4	0

Target: Image B

Pixel Value	Histogram	Equalized Histogram
0	4	4
1	6	10
2	5	15
3	5	20
4	5	25

Equalized Histogram of B

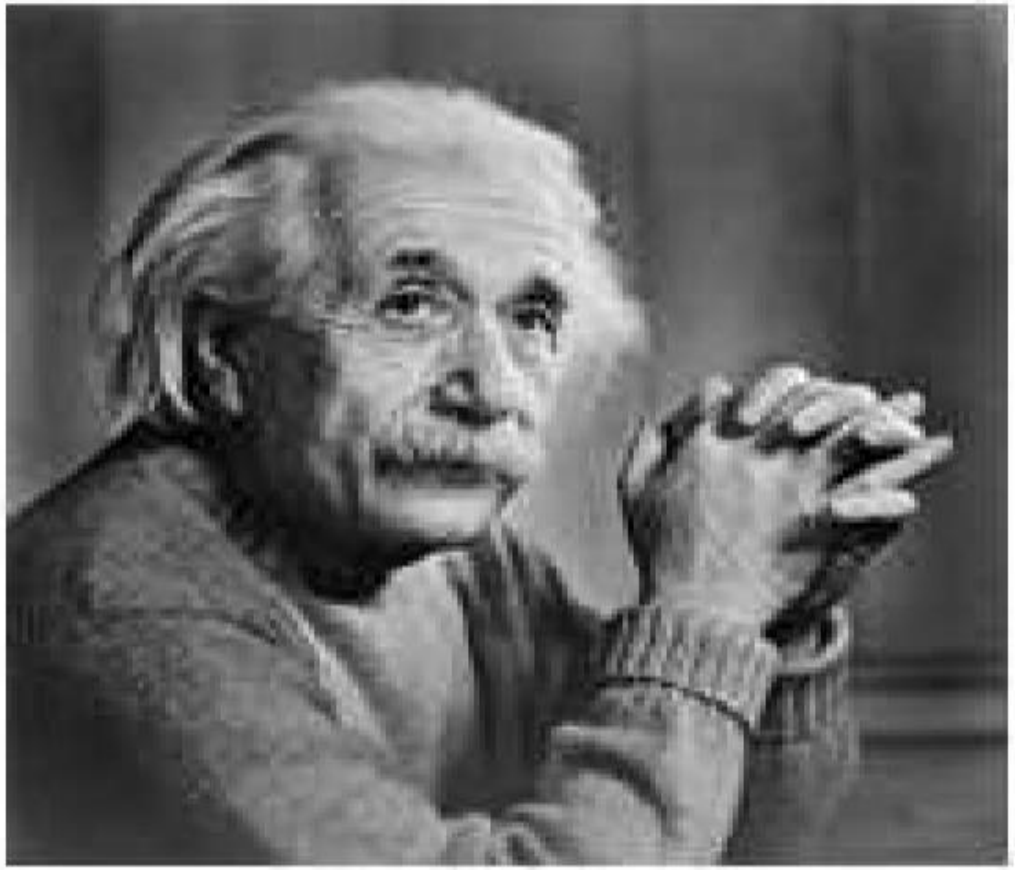


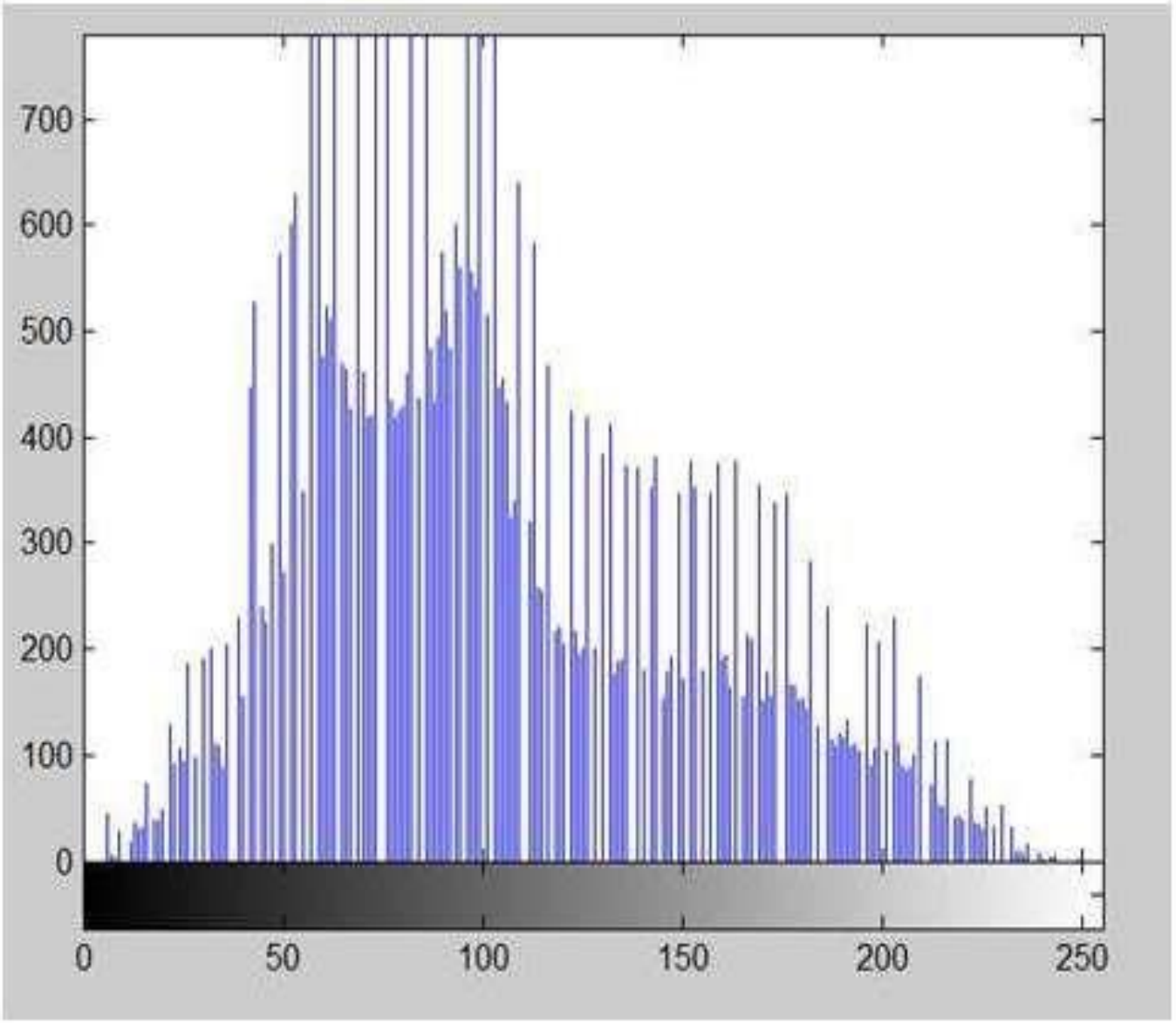
0	1	1	2	4
1	2	4	2	2
0	1	2	1	4
2	1	4	1	0
0	4	1	4	4

Modified: Image A

Histogram Processing

- In digital image processing, the histogram is used for graphical representation of a digital image. A graph is a plot by the number of pixels for each intensity value.
- Nowadays, image histogram is present in digital cameras. Photographers use them to see the distribution of intensity captured.
- In a graph, the horizontal axis of the graph is used to represent intensity variations whereas the vertical axis is used to represent the number of pixels in that particular intensity which is also called the frequency of intensity.





The x-axis of the histogram shows the range of pixel intensity. Since it is an 8 bpp image, which means it has 2^8 (256) levels of gray or shades of gray in it. That's why the range of x axis starts from 0 and end at 255 with a gap of 50.

The y-axis indicates the count of pixel in particular intensity. As you can see from the graph, that most of the bars that have high frequency lies in the first half portion which is the darker portion. That means that the image we have got is darker. And this can be proved from the image too.

Local Enhancement

- Local enhancement, also known as local contrast enhancement or local histogram equalization, is a technique used to improve the contrast and visibility of details in an image by enhancing the contrast in localized regions. Unlike global enhancement techniques, which operate on the entire image, local enhancement methods apply contrast adjustments selectively to small neighborhoods or regions within the image.

steps involved in local enhancement:

- **Divide Image into Local Regions**: The first step is to divide the image into small, overlapping regions or windows. These regions can be square or rectangular and typically have a fixed size. The choice of window size depends on the characteristics of the image and the level of detail you want to enhance.
- **Apply Enhancement to Each Region**: For each local region, apply a contrast enhancement technique to improve the contrast and visibility of details within that region. Histogram equalization, adaptive histogram equalization (AHE), and various non-linear enhancement methods are commonly used for local contrast enhancement.

Cont..

- **Merge Enhanced Regions**: After enhancing each local region, merge the enhanced regions back together to reconstruct the final enhanced image. This step may involve blending the enhanced regions or simply replacing the original pixel values with the enhanced values from each region.
- **Adjust Parameters**: Fine-tune the parameters of the local enhancement technique, such as the size of the local regions, the strength of the enhancement, and the degree of overlap between neighboring regions. These parameters can affect the quality of the enhancement and the computational efficiency of the method.
- **Evaluate Results**: Finally, evaluate the results of the local enhancement technique to assess its effectiveness in improving the contrast and visibility of details in the image. Visual inspection and quantitative metrics can be used to measure the degree of enhancement achieved by the technique.

Examples of local enhancement methods include local histogram equalization

- Contrast Limited Adaptive Histogram Equalization – CLAHE
- Multi-scale retinex,
- wavelet-based methods
- various filtering techniques (e.g., bilateral filtering, guided filtering).

The choice of method depends on the specific requirements of the application and the characteristics of the input images.