المحور الثالث

14. Arithmetic and Logical Operations on Images (Image Algebra)

These operations are applied on pixel-by-pixel basis. So, to add two images together, we add the value at pixel (0 , 0) in image 1 to the value at pixel (0 , 0) in image 2 and store the result in a new image at pixel (0 , 0). Then we move to the next pixel and repeat the process, continuing until all pixels have been visited.

Clearly, this can work properly only if the two images have identical dimensions. If they do not, then combination is still possible, but a meaningful result can be obtained only in ***the area of overlap***. If our images have dimensions of w_1*h_1, and w_2*h_2 and we assume that their origins are aligned, then the new image will have dimensions w*h, where:

$$w = min\ (w_1, w_2)$$

$$h = min\ (h_1, h_2)$$

1. Addition

If In image processing, addition refers to the pixel-wise summation of corresponding pixels in two images. Each pixel in the resulting image is obtained by adding the values of the corresponding pixels in the input images. This operation is also known as pixel-wise addition or image blending.

Explanation of addition in image processing:

- Concept: Imagine you have two images, A and B, where each pixel in both images represents the intensity or color value at that position.
- Operation: To perform addition, you add the values of each pixel in image A to the corresponding pixel in image B. This is done independently for every pixel.
- Mathematically: If A(x, y) represents the pixel value at position (x, y) in image A, and B(x, y) represents the pixel value at the same position in image B, then the resulting pixel value in the new image C(x, y) is calculated as follows: C(x,y)=A(x,y)+B(x,y)
- Purpose: Image addition is often used for various purposes, such as blending two images together. For example, if you have a background image and an overlay image, adding them can create a combined image with elements from both.

- Considerations: When performing addition, it's essential to ensure that the resulting pixel values stay within a valid range (e.g., 0 to 255 for 8-bit images) to prevent overflow and maintain image integrity.

Algorithm 1: image addition

```
read input-image1 into in-array1;
read input-image2 into in- array2;
for i = 1       to no-of-rows do
for j=1 to no-of-columns do begin
out-array (i,j) = in-array1(i,j) + in-array2(i,j);
if ( out-array (i,j) > 255 ) then                    out-array (i,j) = 255;
        end
write out-array to out-image;
```

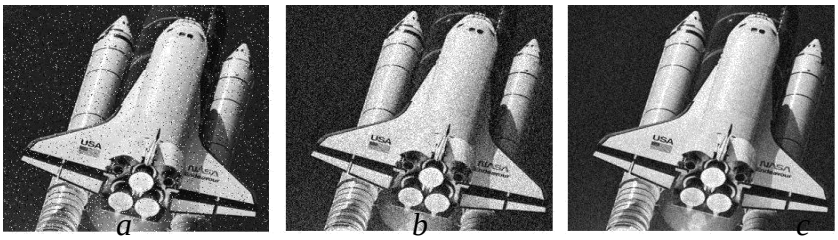


Figure (4) a) noisy image b) average of five observation c) average of ten observation

2. Subtraction

In image processing, subtraction involves performing a pixel-wise subtraction of the values of corresponding pixels in two images. Each pixel in the resulting image is obtained by subtracting the value of the corresponding pixel in one image from the value of the corresponding pixel in another image.

Here's a simple explanation of subtraction in image processing:

- **Concept:** Consider two images, A and B, where each pixel in both images represents the intensity or color value at that position.
- **Operation:** To perform subtraction, you subtract the values of each pixel in image B from the corresponding pixel in image A. This is done independently for every pixel.
- **Mathematically:** If A(x, y) represents the pixel value at position (x, y) in image A, and B(x, y) represents the pixel value at the same position in image B, then the resulting pixel value in the new image C(x, y) is calculated as follows: $C(x,y)=A(x,y)-B(x,y)$

- **Purpose:** Image subtraction is used for various purposes, such as highlighting differences between two images. For example, subtracting a background image from a scene containing an object may emphasize the object's presence.
- **Considerations:** When performing subtraction, it's essential to handle negative values appropriately, and the resulting pixel values may need to be scaled or adjusted based on the desired interpretation of the subtraction operation.

Algorithm2: image subtraction

```
read input-image1 into in-array1;
read input-image2 into in- array2;
for i = 1       to no-of-rows do
    for j=1 to no-of-columns do
        begin
            out-array (i,j) = in-array1(i,j) - in-array2(i,j);
            if ( out-array (i,j) < 0 ) then out-array (i,j) = 0; end
write out-array to out-image;
```

Figure (5) a, b) two frames of video sequence c) their difference

Multiplication and Division

Multiplication and division can be used to adjust brightness of an image. Multiplication of pixel values by a number greater than one will brighten the image, and division by a factor greater than one will darken the image. Brightness adjustment is often used as a ***preprocessing step*** in image enhancement.

One of the principle uses of image multiplication (or division) is to ***correct grey-level shading*** resulting from no uniformities in illumination or in the sensor used to acquire the image.

(a) *(b)* *(c)*

Figure a) original image b) image multiplied by 2 c) image divided by 2

16. Image Histogram

Histograms are widely used in image processing for various purposes, and they provide valuable insights into the distribution of pixel intensities within an image. Here are some key reasons why histograms are important in image processing:

- **Intensity Distribution Analysis:** Histograms illustrate the distribution of pixel intensities in an image. This information is crucial for understanding the overall brightness and contrast characteristics of an image. Peaks and valleys in the histogram can indicate the presence of dominant colors or intensity levels, helping to identify features within the image.
- **Contrast Adjustment:** Histograms are often used to analyze and adjust the contrast of an image. Adjusting the contrast can enhance the visual quality of an image by spreading the pixel intensities across a wider range.

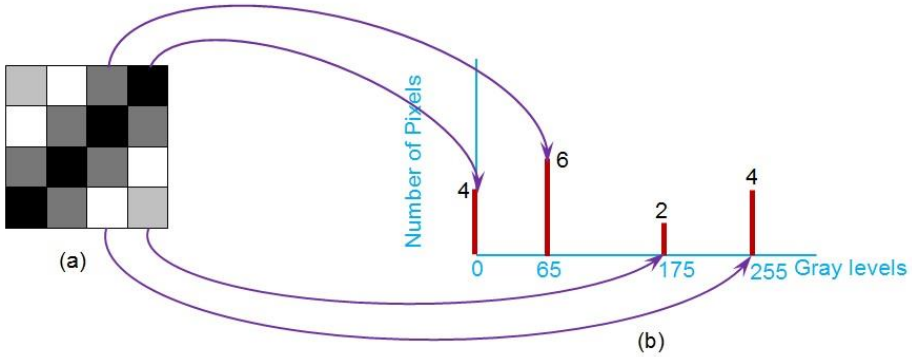
- **Brightness Adjustment:** Histogram analysis allows for the adjustment of image brightness. By shifting the entire histogram to the left or right, it's possible to make the image darker or brighter, respectively.
- **Thresholding**: Thresholding is a technique used to segment an image into regions based on pixel intensity. Histograms help in determining suitable thresholds by identifying peaks and gaps in the intensity distribution.
- **Image Segmentation:** Histograms assist in image segmentation by revealing the presence of different materials or objects in the scene. Peaks in the histogram may correspond to different materials, and segmentation can be performed based on these intensity levels.
- **Noise Detection:** Histograms can help identify and analyze noise in an image. Random noise often appears as small spikes in the histogram, and this information can be used to apply noise reduction techniques.
- **Color Balance and Correction:** In the case of color images, histograms can be created separately for each color channel (e.g., red, green, and blue). Analyzing these histograms helps in correcting color balance issues and achieving a more natural appearance.
- **Quality Control:** Histogram analysis is valuable in quality control processes, such as in medical imaging or industrial applications, where understanding and controlling pixel intensities are critical.
- **Pre-processing for Machine Learning:** Histograms can be used as features for machine learning algorithms. They provide a compact representation of the image's intensity distribution, which can be input into classifiers or regression models.

- **Image Enhancement:** Histogram manipulation techniques, such as stretching or compressing, can be applied to enhance specific features in an image, making it visually more appealing or suitable for specific applications.

A histogram is a graph that shows the frequency of anything.

A histogram is an accurate representation of the distribution of numerical data. It is an estimate of the probability distribution of a continuous variable (quantitative variable). A histogram is a graph showing the number of pixels in an image at each different intensity value found in that image. It's a bar chart of the count of pixels of every tone of gray that occurs in the image.

For an 8-bit grayscale image there are 256 different possible intensities, and so the histogram will graphically display 256 numbers showing the distribution of pixels amongst those grayscale values.



The gray level ***histogram*** is showing, the gray level, for each pixel in the image.

The histogram of an 8-bit image, can be thought of as a table with 256 entries, or " bins",

> The histogram of an image records the frequency distribution of gray levels in the image.

indexed from 0 to 255. in bin 0 we record the number of times a gray level of 0 occurs; in bin 1 we record the number of times a gray level of 1 occurs, and so on, up to bin 255.

An algorithm below shows how we can accumulate in a histogram from an image.

ALGORITHM: for Calculating image Histogram

```
create an array histogram.
For all gray levels ,I,do
Histogram [I] =0
Endfor
For all pixels coordinates, x and y , do
Increment Histogram[ f (x,y) ] by 1
```

- The histogram of a digital image with L total possible intensity levels in the range $[0, G]$ is defined as the discrete function

$$h(r_k) = n_k$$

- Where r_k is the kth intensity level in the interval $[0,G]$ and n_k is the number of pixels in the image whose intensity level is r_k.

Example: Figure shows an image and its histogram.

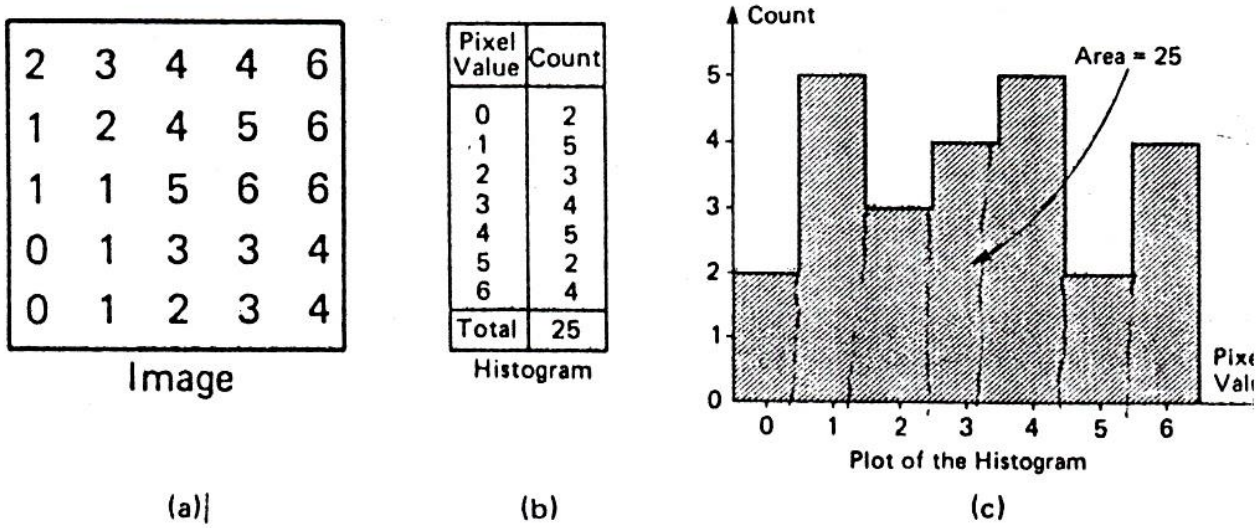


Figure: sub image and its histogram

The shape of the histogram provides us with information about the nature of the image, or sub image if we are considering an object in the image. For example, a ***very narrow*** histogram implies a low contrast, a histogram ***skewed toward the right*** implies a bright image, a histogram ***skewed toward the left*** implies a dark image, and a histogram with ***two major peaks***, implies an object that in contrast with the background.

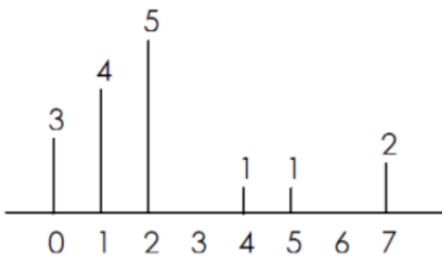
A color histogram counts pixels with a given pixel value in red, green, and blue (RGB). For example, in pseudocode, for images with 8-bit values in each of R, G, B, we can fill a histogram that has 256^3 bins:

```
inthist[256][256][256]; // reset to 0

//image is an appropriate struct

//with byte fields red,green,blue

fori=0..(MAX_Y-1)

for j=0..(MAX_X-1)

{

R = image[i][j].red;

G = image[i][j].green;
```

Example : Plot the Histogram of the following example with 4x4 matrix of a 3-bit image.

0	0	0	4
1	1	1	5
1	2	2	7
2	2	2	7

Answer →



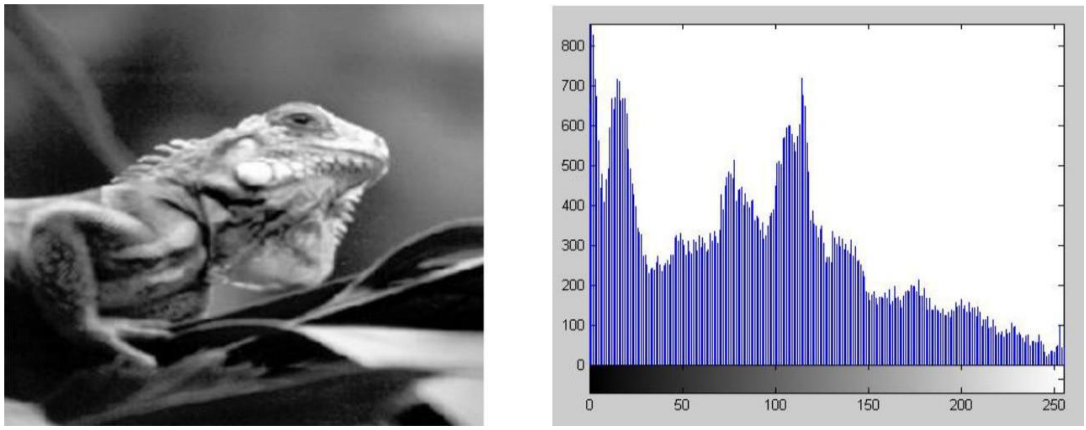
Properties and usage of histogram

One of the ***principle use*** of the histogram is in the selection of ***threshold*** parameter.

The histogram of an image provides a useful indication of the relative importance of different gray levels in an image, indeed, it is sometimes possible to ***determine*** whether ***brightnes***s or ***contras*t *adjustment*** is necessary merely by ***examining*** the ***histogram*** and ***not the image*** itself.

When an image is condensed into a histogram, ***all spatial information is discarded.*** The histogram specifies the number of pixels having each gray level but ***gives no hint*** as to ***where those pixels are located*** within the image. Thus the histogram is ***unique*** for any particular image, but the ***reverse is not true***. Vastly different images could have identical histograms. Such operations as moving objects around within an image typically have no effect on the histogram.

Histograms are used in numerous image processing techniques, such as image enhancement, compression and segmentation.



Note that the horizontal axis of the histogram plot (Figure (b) above) represents gray level values, k, from 0 to 255. The vertical axis represents the values of $h(k)$ i.e. the number of pixels which have the gray level k.

It is customary to **"normalize"** a histogram by dividing each of its values by the total number of pixels in the image, i.e. uses the probability distribution (previously stated) as:

$$p(k) = \frac{h(k)}{M \times N}$$

Thus, (kp) represents the probability of occurrence of gray level k.

As with any probability distribution:

- ***All the values of a normalized histogram*** $p(k)$ ***are less than or equal to 1.***
- ***The sum of all*** $p(k)$ ***values is equal to 1***

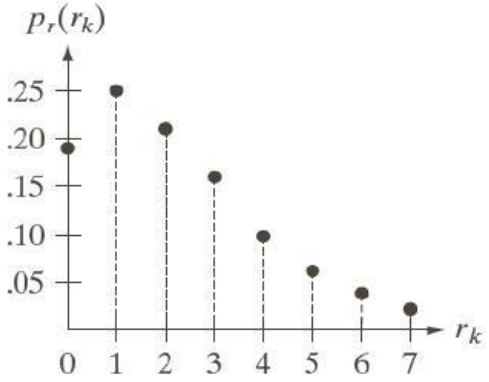
Another way of getting histogram is to plot pixel intensities vs. pixel probabilities. However, probability histogram should be used when comparing the histograms of images with different sizes.

Example: Suppose that a 3-bit image (L = 8) of size 64 × 64 pixels has the gray level (intensity) distribution shown in the table below. ***Perform normalized histogram.***

r_k	n_k
$r_0 = 0$	790
$r_1 = 1$	1023
$r_2 = 2$	850
$r_3 = 3$	656
$r_4 = 4$	329
$r_5 = 5$	245
$r_6 = 6$	122
$r_7 = 7$	81

Solution: $M \times N = 4096$. We compute the normalized histogram:

r_k	n_k	$p(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02



Normalized histogram