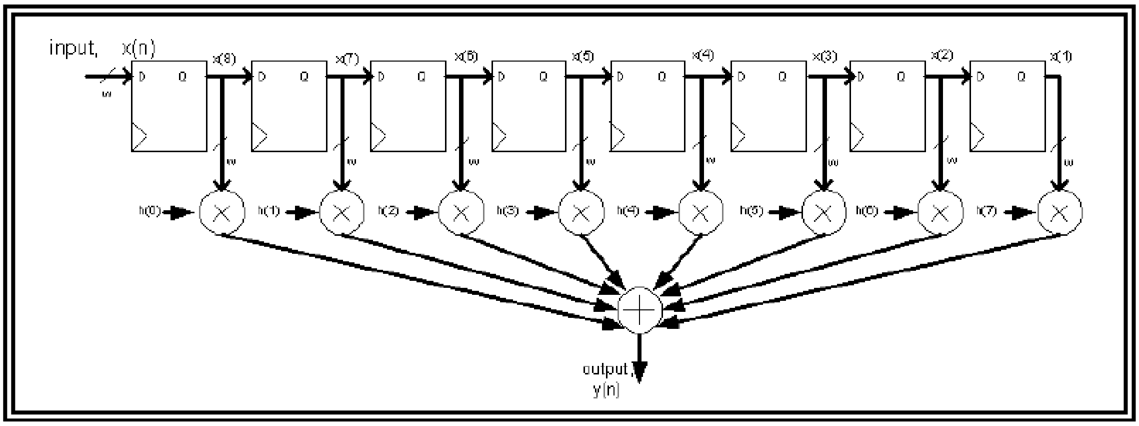
Ministry of Higher Education and Scientific Research
Middle Technical University
Electrical Engineering Technical College

Training package
in

Digital Signal Processing
(Image Processing II)

For
Students of third class
Control and Automation Engineering Techniques



By

Prof. Dr. Raaed F. Hassan

Dep. Control and Automation Eng. Tech.
September/2021

Image Processing II

1/ Overview

1 / A –Target population :-

For students of third class

Department of Medical Instrumentation Eng. Techniques

1 / B –Rationale :-

This unit introduces Image processing techniques

1 / C –Central Idea :-

The major topics discussed in this unit are included in the following outline.

- **Image Level Adjustment and Contrast**
- **Linear Level Adjustment**
- **Adjusting the Level for Display**
- **Image Filtering Enhancement**
- **Lowpass Noise Filtering**
- **Median Filtering**

2/ Performance Objectives :-

After studying the 14th modular unit, the student will be able to know:-

1. Image enhancement and filtering.

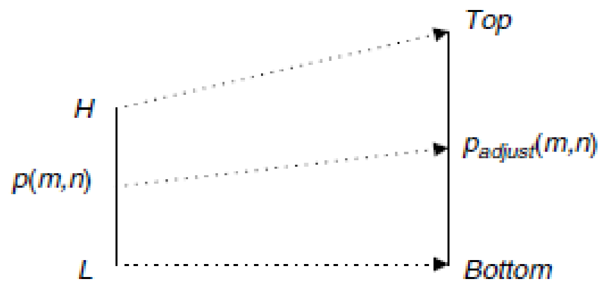
4/ the text :-

Image Level Adjustment and Contrast

Image level adjustment can be used to linearly stretch the pixel level in an image to increase contrast and shift the pixel level to change viewing effects. Image level adjustment is also a requirement for modifying results from image filtering or other operations to an appropriate range for display. We will study this technique in the following subsections.

Linear Level Adjustment

Sometimes, if the pixel range in an image is small, we can adjust the image pixel level to make use of a full pixel range. Hence, contrast of the image is enhanced. Figure below illustrates linear level adjustment.



Linear level adjustment.

The linear level adjustment is given by the following formula:

$$p_{adjust}(m, n) = Bottom + \frac{p(m, n) - L}{H - L} \times (Top - Bottom),$$

where $p(m, n)$ = original image pixel

$p_{adjust}(m, n)$ = desired image pixel
H = maximum pixel level in the original image
L = minimum pixel level in the original image
Top = maximum pixel level in the desired image
Bottom = minimum pixel level in the desired image

Besides adjusting the image level to a full range, we can also apply the method to shift the image pixel levels up or down.

Example

Given the following image (matrix filled with integers) with a grayscale value ranging from 0 to 7, that is, with each pixel encoded in 3 bits,

$$\begin{bmatrix} 3 & 4 & 4 & 5 \\ 5 & 3 & 3 & 3 \\ 4 & 4 & 4 & 5 \\ 3 & 5 & 3 & 4 \end{bmatrix},$$

a. Perform level adjustment to a full range.

b. Shift the level to the range from 3 to 7.

c. Shift the level to the range from 0 to 3.

Solution:

a. From the given image, we set the following for level adjustment to the full range:

$$H = 5, L = 3, Top = 2^3 - 1, Bottom = 0.$$

b. For the shift-up operation, it follows that

$$H = 5, L = 3, Top = 7, Bottom = 3.$$

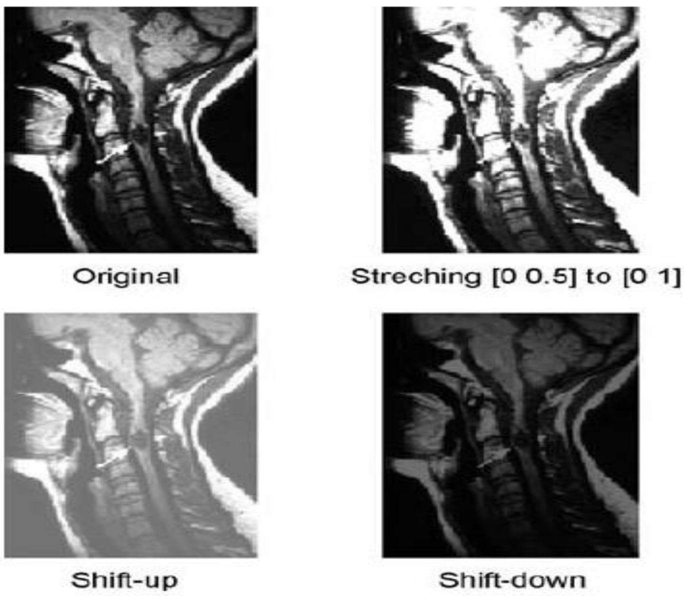
c. For the shift-down operation, we set

$$H = 5, L = 3, Top = 3, Bottom = 0.$$

Pixel $p(m, n)$ Level	Full Range	Range [3–7]	Range [0–3]
3	0	3	0
4	4	5	2
5	7	7	3

According to above Table, we have three images:

$$\begin{bmatrix} 0 & 4 & 4 & 7 \\ 7 & 0 & 0 & 0 \\ 4 & 4 & 4 & 7 \\ 0 & 7 & 0 & 4 \end{bmatrix} \quad \begin{bmatrix} 3 & 5 & 5 & 7 \\ 7 & 3 & 3 & 3 \\ 5 & 5 & 5 & 7 \\ 3 & 7 & 3 & 5 \end{bmatrix} \quad \begin{bmatrix} 0 & 2 & 2 & 3 \\ 3 & 0 & 0 & 0 \\ 2 & 2 & 2 & 3 \\ 0 & 3 & 0 & 2 \end{bmatrix}$$



Adjusting the Level for Display

When two 8-bit images are added together or undergo other mathematical operations, the sum of two pixel values can be as low as 0 and as high as 510. We can apply the linear adjustment to scale the range back to 0–255 for display. The following addition of two 8-bit images:

$$\begin{bmatrix} 30 & 25 & 5 & 170 \\ 70 & 210 & 250 & 30 \\ 225 & 125 & 50 & 70 \\ 28 & 100 & 30 & 50 \end{bmatrix} + \begin{bmatrix} 30 & 255 & 50 & 70 \\ 70 & 3 & 30 & 30 \\ 50 & 200 & 50 & 70 \\ 30 & 70 & 30 & 50 \end{bmatrix} = \begin{bmatrix} 60 & 280 & 55 & 240 \\ 140 & 213 & 280 & 60 \\ 275 & 325 & 100 & 140 \\ 58 & 179 & 60 & 100 \end{bmatrix}$$

Yields a sum that is out of the 8-bit range. To scale the combined image, modify the Equation

$$p_{adjust}(m,\ n) = Bottom + \frac{p(m,\ n) - \mathrm{L}}{\mathrm{H} - \mathrm{L}} \times (\mathrm{Top\ \text{-}\ Bottom}),$$

$$p_{scaled}(m, n) = \frac{p(m, n) - \mathrm{Minimum}}{\mathrm{Maximum} - \mathrm{Minimum}} \times (\mathrm{Maximum\ scale\ level}).$$

Note that in the image to be scaled,

$\mathrm{Maximum} = 325$
$\mathrm{Minimum} = 55$
$\mathrm{Maximum\ scale\ level} = 255,$

We have after scaling:

$$\begin{bmatrix} 5 & 213 & 0 & 175 \\ 80 & 149 & 213 & 5 \\ 208 & 255 & 43 & 80 \\ 3 & 109 & 5 & 43 \end{bmatrix}.$$

Image Filtering Enhancement

As with one-dimensional digital signal processing, we can design a digital image filter such as lowpass, highpass, bandpass, and notch to process the image to obtain the desired effect. In this section, we discuss the most common ones: lowpass filters to remove noise, median filters to remove impulse noise, and edge detection filters to gain the boundaries of objects in the images.

Lowpass Noise Filtering

One of the simplest lowpass filters is the average filter. The noisy image is filtered using the average convolution kernel with a size 3 x 3 block, 4 x 4 block, 8 x 8 block, and so on, in which the elements in the block have the same filter coefficients. The 3 x 3, 4 x 4, and 8 x 8 average kernels are as follows:

3 x 3 average kernel:

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

4 x 4 average kernel:

$$\frac{1}{16}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

8 x 8 average kernel:

$$\frac{1}{64}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

The convolution operates to modify each pixel in the image as follows. By passing the center of a convolution kernel through each pixel in the noisy image, we can sum each product of the kernel element and the corresponding image pixel value and multiply the sum by the scale factor to get the processed pixel. To understand the filter operation with the convolution kernel, let us study the following example.

Example

Perform digital filtering on the noisy image using 2 x 2 convolutional average kernel, and compare the enhanced image with the original one given the following 8-bit grayscale original and corrupted (noisy) images.

$$4 \times 4 \text{ original image: } \begin{bmatrix} 100 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 \end{bmatrix}$$

$$4 \times 4 \text{ corrupted image: } \begin{bmatrix} 99 & 107 & 113 & 96 \\ 92 & 116 & 84 & 107 \\ 103 & 93 & 86 & 108 \\ 87 & 109 & 106 & 107 \end{bmatrix}$$

$$2 \times 2 \text{ average kernel: } \frac{1}{4}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

Solution:

In the following diagram, we pad edges with zeros in the last row and column before processing at the point where the first kernel and the last kernel are shown in dotted-line boxes, respectively.

99	107	113	96	0
92	116	84	107	0
103	93	86	108	0
87	109	106	107	0
0	0	0	0	0

To process the first element, we know that the first kernel covers the image elements as

$$\begin{bmatrix} 99 & 107 \\ 92 & 116 \end{bmatrix}.$$

Summing each product of the kernel element and the corresponding image pixel value, multiplying a scale factor of 1⁄4, and rounding the result, it follows that

$$\frac{1}{4}(99 \times 1 + 107 \times 1 + 92 \times 1 + 116 \times 1) = 103.5$$
$$round(103.5) = 104.$$

In the processing of the second element, the kernel covers

$$\begin{bmatrix} 107 & 113 \\ 116 & 84 \end{bmatrix}.$$

Similarly, we have

$$\frac{1}{4}(107 \times 1 + 113 \times 1 + 116 \times 1 + 84 \times 1) = 105$$
$$round(105) = 105.$$

The process continues for the rest of the image pixels. To process the last element of the first row, 96, since the kernel covers only

$$\begin{bmatrix} 96 & 0 \\ 107 & 0 \end{bmatrix}$$

We assume that the last two elements are zeros. Then:

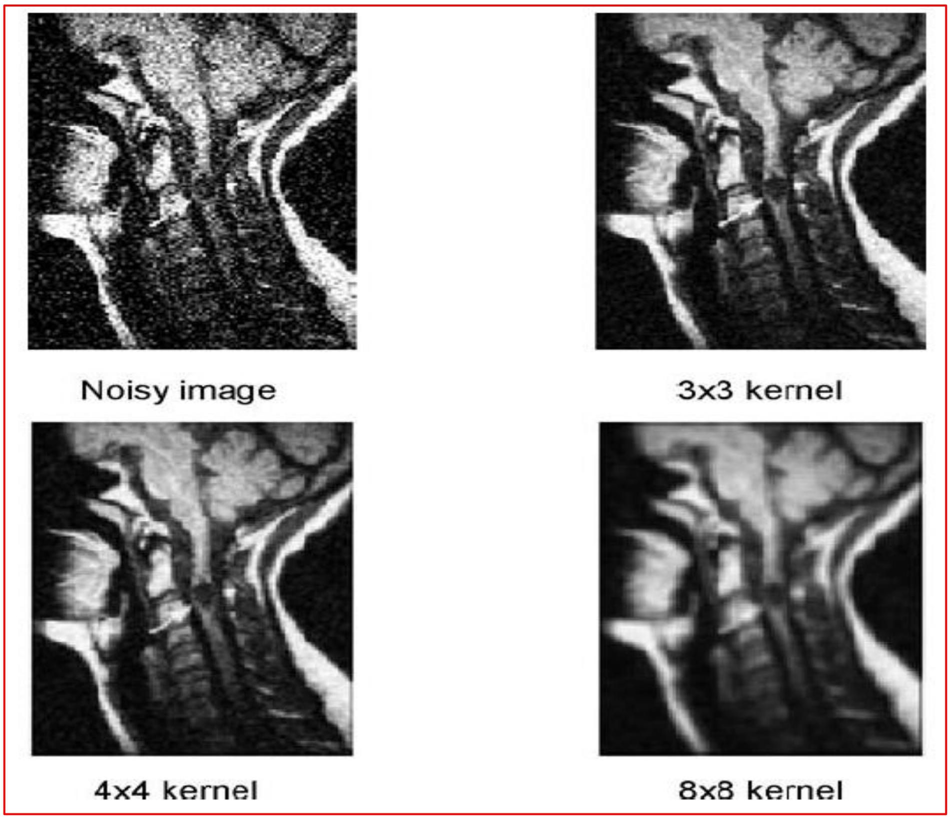
$$\frac{1}{4}(96 \times 1 + 107 \times 1 + 0 \times 1 + 0 \times 1) = 50.75$$
$$round(50.75) = 51.$$

Finally, we yield the following filtered image:

$$\begin{bmatrix} 104 & 105 & 100 & 51 \\ 101 & 95 & 96 & 54 \\ 98 & 98 & 102 & 54 \\ 49 & 54 & 53 & 27 \end{bmatrix}.$$

As we know, due to zero padding for boundaries, the last-row and last-column values are in error. However, for a large image, these errors at boundaries can be neglected without affecting image quality. The first 3 x 3 elements in the processed image have values that are close to those of the original image. Hence, the image is enhanced.

Figure below shows the noisy image and enhanced images using the 3x3, 4x4, and 8x8 average lowpass filter kernels, respectively. The average kernel removes noise. However, it also blurs the image. When using a large-sized kernel, the quality of the processed image becomes unacceptable.



Median Filtering

The median filter is one type of nonlinear filters. It is very effective at removing impulse noise, the ‘‘pepper and salt’’ noise, in the image. The principle of the median filter is to replace the gray level of each pixel by the median of the gray levels in a neighborhood of the pixel, instead of using the average operation.

For median filtering, we specify the kernel size, list the pixel values covered by the kernel, and determine the median level. If the kernel covers an even number of pixels, the average of two median values is used. Before beginning median filtering, zeros must be padded around the row edge and the column edge. Hence, edge distortion is introduced at the image boundary.

Example

Given a 3 x 3 median filter kernel and the following 8-bit grayscale original and corrupted (noisy) images,

$$4 \times 4 \text{ original image}: \begin{bmatrix} 100 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 \end{bmatrix}$$

$$4 \times 4 \text{ corrupted image by impulse noise}: \begin{bmatrix} 100 & 255 & 100 & 100 \\ 100 & 255 & 100 & 100 \\ 255 & 100 & 100 & 0 \\ 100 & 100 & 100 & 100 \end{bmatrix}$$

Perform digital filtering, and compare the filtered image with the original one.

Solution:

Step 1: The 3 x 3 kernel requires zero padding 3/2 = 1 column of zeros at the left and right edges and 3/2 = 1 row of zeros at the upper and bottom edges:

0	0	0	0	0	0
0	100	255	100	100	0
0	100	255	100	100	0
0	255	100	100	0	0
0	100	100	100	100	0
0	0	0	0	0	0

Step 2: To process the first element, we cover the 3 x 3 kernel with the center pointing to the first element to be processed. The sorted data within the kernel are listed in terms of their values as

$$0,\ 0,\ 0,\ 0,\ 0,\ 100,\ 100,\ 255,\ 255.$$

The median value = median(0, 0, 0, 0, 0, 100, 100, 255, 255) = 0. Zero will replace 100.

Step 3: Continue for each element until the last is replaced. Let us see the element at the location (1, 1):

0	0	0	0	0	0
0	100	255	100	100	0
0	100	255	100	100	0
0	255	100	100	0	0
0	100	100	100	100	0
0	0	0	0	0	0

The values covered by the kernel are:

100, 100, 100, 100, 100, 100, 255, 255, 255.

The median value = median(100, 100, 100, 100, 100, 100, 255, 255, 255) = 100. The final processed image is

$$\begin{bmatrix} 0 & 100 & 100 & 0 \\ 100 & 100 & 100 & 100 \\ 0 & 100 & 100 & 0 \\ 100 & 100 & 100 & 100 \end{bmatrix}.$$

Some boundary pixels are distorted due to zero padding effect. However, for a large image, the distortion can be omitted versus the overall quality of the image. The 2 x 2 middle portion matches the original image exactly.

The image in Figure 13.27a is corrupted by ‘‘pepper and salt’’ noise. The median filter with a 3 x 3 kernel is used to filter this impulse noise. The enhanced image in Figure 13.27b has a significant quality improvement.

FIGURE 13.27A Noisy image (corrupted by ''pepper and salt'' noise).

FIGURE 13.27B The enhanced image using the 3 X 3 median filter.

7/References :-

1. Schaum's Outline of Theory and Problems ofDigital Signal processing.
2. Digital signal processing, principles, algorithms, and applications by John G. Proakis and Dimitris G. Manolakis.
3. Signal and systems, Alan Oppenheim.