



## DEPT OF ELECTRICAL & ELECTRONICS ENGINEERING

Program: T. Y. B. Tech ECE AI-ML Semester -6

Subject: Image Processing & Pattern Recognition (PC) (ECE3022B)

### EXPERIMENT NO. 3 Image smoothing and sharpening

#### AIM:

- A. To study spatial domain low pass filtering of an image with mean, median filters. Compare the performance of these two filters. Observe the effect of varying mask size on the filtered image.
- B. To study spatial domain high pass filtering of an image with sharpening filter.

#### THEORY:

Many image enhancement techniques are based on spatial operations performed on the local neighborhood of input pixels. Spatial domain methods are procedures that operate directly on these pixels. Spatial domain processes will be denoted by

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

where  $f(x, y)$  is the input image,  $g(x, y)$  is the processed image and  $T$  is an operator on  $f$ , defined over certain neighborhood. The principal approach in defining a neighborhood about a point  $(x, y)$  is to use square or rectangular subimage area centered at  $(x, y)$ . The subimage is called as filter or mask.(FIG. 5A.1)

#### SPATIAL FILTERING

The process consists of moving the filter/mask from point to point in an image. The response of the filter at point  $(x, y)$  is calculated by convolving the mask/filter coefficients with the pixel values under the mask. For a  $3 \times 3$  mask response  $R$  is given by

$$R = ( W1P1 + W2P2 + .....+ W9P9 )$$

Where  $W1, W2, \dots, W9$  are filter coefficients and  $P1, P2, \dots, P9$  are pixel values under the mask area.

#### Blur (Averaging)

During this operation, the image is convolved with a box filter (normalized). In this process, the central element of the image is replaced by the average of all the pixels in the kernel area.

You can perform this operation on an image using the method **blur()** of the **imgproc** class. Following is the syntax of this method –

## **blur(src, dst, ksize, anchor, borderType)**

This method accepts the following parameters –

- **src** – A **Mat** object representing the source (input image) for this operation.
- **dst** – A **Mat** object representing the destination (output image) for this operation.
- **ksize** – A **Size** object representing the size of the kernel.
- **anchor** – A variable of the type integer representing the anchor point.
- **borderType** – A variable of the type integer representing the type of the border to be used to the output.

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

## **MEDIAN FILTER:**

In median filter, the gray level of each pixel is replaced by the median of gray levels in a neighborhood of that pixel. The principal function of median filtering is to force points with distinct intensities to be more like their neighbors. Hence this filter is effective when the noise pattern consists of strong spike-like components.

The Median blur operation is similar to the other averaging methods. Here, the central element of the image is replaced by the median of all the pixels in the kernel area. This operation processes the edges while removing the noise.

You can perform this operation on an image using the **medianBlur()** method of the **imgproc** class. Following is the syntax of this method –

## **medianBlur(src, dst, ksize)**

This method accepts the following parameters –

- **src** – A **Mat** object representing the source (input image) for this operation.
- **dst** – A **Mat** object representing the destination (output image) for this operation.
- **ksize** – A **Size** object representing the size of the kernel.

## **Gaussian Filter**

In the Gaussian Blur operation, the image is convolved with a Gaussian filter instead of the box filter. The Gaussian filter is a low-pass filter that removes the high-frequency components are reduced.

You can perform this operation on an image using the **GaussianBlur()** method of the **imgproc** class. Following is the syntax of this method –

## **GaussianBlur(src, dst, ksize, sigmaX)**

This method accepts the following parameters –

- **src** – A **Mat** object representing the source (input image) for this operation.
- **dst** – A **Mat** object representing the destination (output image) for this operation.
- **ksize** – A **Size** object representing the size of the kernel.
- **sigmaX** – A variable of the type double representing the Gaussian kernel standard deviation in X direction.

## Example

Following steps are performed in the code below:

- Read the test image
- Define the identity kernel, using a  $3 \times 3$  NumPy array
- Use the **filter2D()** function in OpenCV to perform the linear filtering operation
- Display the original and filtered images, using **imshow()**
- Save the filtered image to disk, using **imwrite()**

### **filter2D(src, ddepth, kernel)**

The **filter2D()** function requires three input arguments:

- The first argument **src** is the source image
- The second argument is **ddepth**, which indicates the depth of the resulting image. A value of **-1** indicates that the final image will also have the same depth as the source image
- The final input argument is the **kernel**, which we apply to the source image

## Algorithm

### Part A.

1. Read the grayscale image.
2. Add noise to the image
3. Apply averaging filter on the given image
4. Display original and filtered images.
5. Vary mask size from  $5 \times 5$  to  $15 \times 15$  and display filtered images.
6. Comment on the results.
  
1. Read the grayscale image.
2. Add salt and pepper noise to the image
3. Apply Nonlinear filters like Median, min, and max on the given image
4. Display original and filtered images.
5. Compare the Median filtered image with the averaging filtered image
6. Comment on the results.

### Part B.

1. Read the gray scale image.
2. Apply sharpening filter on the given image
3. Display original and filtered images.
4. Apply sharpening filter on the noisy image.
5. Display original and sharpened images.
6. Comment on the results of step 3 and step 5.

## **OUTPUT:**

Low pass filtered images by mean, median filtering where noise is reduced as compared to the input image.

Low pass filtered images with varying mask size (3 x 3, 5 x 5...etc)

Sharpened Image

## **QUESTIONS:**

1. Write a note on Sharpening Filters.
2. Consider an input image row [4 3 2 1] with intensity values in the range 0 to 15. Determine negative of the image row.
3. Determine the new value of the central pixel of the following image by applying 3x3 size:  
a) Mean filter b) Median filter c) Mode filter

10	11	11
10	255	11
12	12	11

### **Note**

- Write down the Experiment Number, Title, PRN number, and Name, Date of performance in the first cell of the python code using markdown.
- Write down the “Result and Conclusion” in the cell next to the python code using markdown.
- Write down questions and answers in the last cell of the python code notebook using markdown.