Azmi Tewfiq. Hussain Al-Rawi,
Faculty of C. S.& I.T. University of Anbar

Spatial Filtering

Spatial filtering may be considered as an extension of point processing, where we apply a function to a neighborhood of each pixel. The idea is to move a "mask": a rectangle or other shape over the given image. we create a new image whose pixels have grey values calculated from the grey values under the mask, as shown in figure (1). The combination of mask and function is called filter

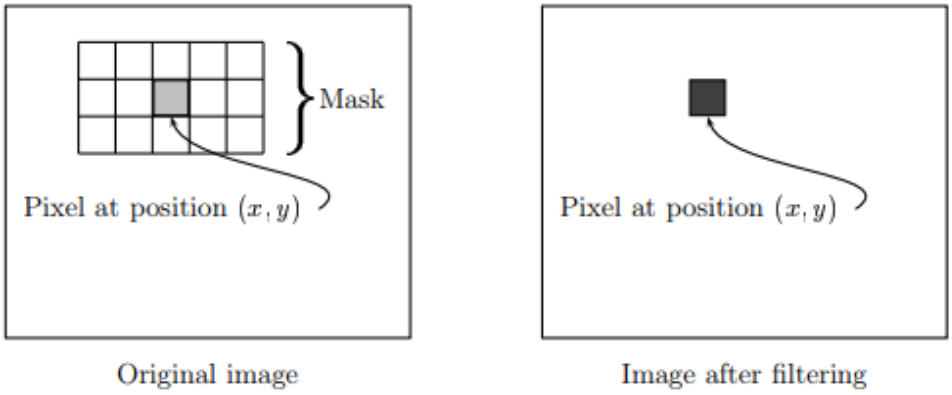


Figure (1): Using a spatial mask on an image.

If the function by which the new grey value is calculated is a linear function of all the grey values in the mask, then the filter is called a **linear filter**. Suppose that the mask values are given by:

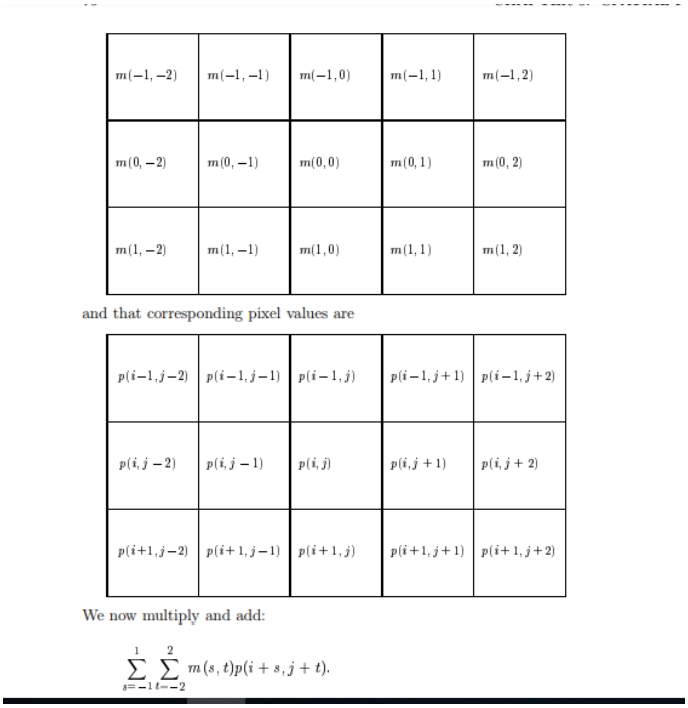


Figure (2): Diagram of spatial filtering.

A diagram illustrating the process for performing spatial filtering is given in figure 2. We see that spatial filtering requires three steps:

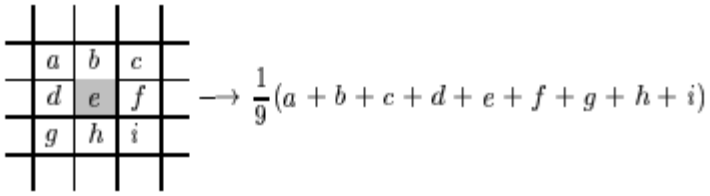
1. position the mask over the current pixel.

2. form all products of filter elements with the corresponding elements of the neighborhood.

3. add up all the products.

This must be repeated for every pixel in the image.

We may describe this operation as follow:



Where the letters a,b,c…..,I are the values of gray levels for the pixels in the mask. The average is the grey value of the corresponding pixel in the new image.

Example 1: To apply this to an image, consider the 5x5 “image” with 3x3 mask.

170	240	10	80	150
230	50	70	140	160
40	60	130	200	220
100	120	190	210	30
110	180	250	20	90

The result is:

```
111.1111  108.8889  128.8889
110.0000  130.0000  150.0000
131.1111  151.1111  148.8889
```

It is convenient to describe a linear filter simply in terms of the coefficients of all the grey values of pixels within the mask:

$$\frac{1}{9}a+\frac{1}{9}b+\frac{1}{9}c+\frac{1}{9}d+\frac{1}{9}e+\frac{1}{9}f+\frac{1}{9}g+\frac{1}{9}h+\frac{1}{9}i$$

and so we can describe this filter by the matrix

$$\left[\begin{array}{ccc}\frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9}\end{array}\right]=\frac{1}{9}\left[\begin{array}{ccc}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{array}\right]$$

An example: The filter

$$\left[\begin{array}{ccc}1 & -2 & 1\\ -2 & 4 & -2\\ 1 & -2 & 1\end{array}\right]$$

would operate on grey values as

a	b	c
d	e	f
g	h	i

$\longrightarrow a-2b+c-2d+4e-2d+g-2h+i$

Edge of the image:

There are a number of different approaches to dealing with edge of image.

1- **Ignore the edges**:
we only apply all pixels except for the edges, and results in an output image which is smaller than the original. In this method lose a significant amount of information.

2- **"Pad" with zeros**:
We assume that all necessary values outside the image are zero.
Figure (3) explain the edge of image.

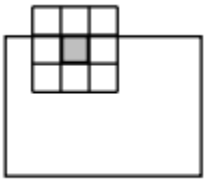


Figure (3): A mask at the edge of an image.

1-Applies the mask only to "**inside" pixels**. The image in example 1. We can see that the result will always be smaller than the original. It used ''**ignore the edge** ''by using the filter:

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

We get:

```
111.1111   108.8889   128.8889
110.0000   130.0000   150.0000
131.1111   151.1111   148.8889
```

2- It produces a matrix of **equal size (same)** to the original image matrix. It uses **zero padding:**

```
     0     0     0     0     0     0     0
     0   170   240    10    80   150     0
     0   230    50    70   140   160     0
     0    40    60   130   200   220     0
     0   100   120   190   210    30     0
     0   110   180   250    20    90     0
     0     0     0     0     0     0     0
```

```
76.6667   85.5556   65.5556   67.7778   58.8889
87.7778  111.1111  108.8889  128.8889  105.5556
66.6667  110.0000  130.0000  150.0000  106.6667
67.7778  131.1111  151.1111  148.8889   85.5556
56.6667  105.5556  107.7778   87.7778   38.8889
```

3-it used larger **than the origin(full)** by **padding with zero**, and applying the filter at all places on and around the image where the mask intersects the image matrix.

```
18.8889   45.5556   46.6667   36.6667   26.6667   25.5556   16.6667
44.4444   76.6667   85.5556   65.5556   67.7778   58.8889   34.4444
48.8889   87.7778  111.1111  108.8889  128.8889  105.5556   58.8889
41.1111   66.6667  110.0000  130.0000  150.0000  106.6667   45.5556
27.7778   67.7778  131.1111  151.1111  148.8889   85.5556   37.7778
23.3333   56.6667  105.5556  107.7778   87.7778   38.8889   13.3333
12.2222   32.2222   60.0000   50.0000   40.0000   12.2222   10.0000
```

Frequencies; low and high pass filters:

One important aspect of an image which enables us to do this is the notion of frequencies. Fundamentally, the **frequencies** of an image are the amount by which grey values change with distance. **High frequency components** are characterized by large changes in grey values over small distances; example of high frequency components are edges and noise. **Low frequency components**, on the other hand, are parts of the image characterized by little change in the grey values. These may include backgrounds, skin textures.

We then say that a filter is a

high pass filter if it "passes over" the high frequency components, and reduces or eliminates low frequency components.

low pass filter if it "passes over" the low frequency components, and reduces or eliminates high

frequency component.

The 3x3 averaging filter is low pass filter, as it tends to blur edges.

The filter is a high pass filter as:

$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

We note that the sum of the coefficients (that is, the sum of all e elements in the matrix), in the high pass filter is zero. This means that in a low frequency part of an image, where the grey values are similar.

Example: consider a 4x4 block of similar values pixels, and apply the above high pass filter to the center four.

150	152	148	149
147	152	151	150
152	148	149	151
151	149	150	148

$\longrightarrow$

11	6
−13	−5

The resulting values are close to zero.

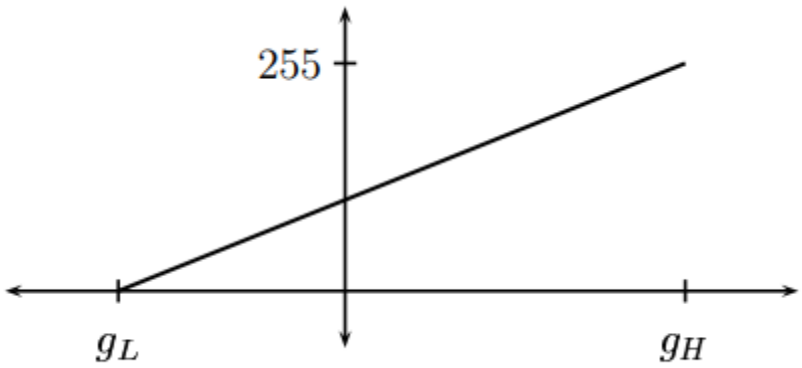
High pass filters are of particular value in edge detection and edge enhancement.

Values outside the range 0–255 :

$$y = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } 0 \leq x \leq 255 \\ 255 & \text{if } x > 255 \end{cases}$$

This will produce an image with all pixel values in the required range, but is not suitable if there are many grey values outside the 0–255 range.

Scaling transformation. Suppose the lowest grey value produced by the filter if g_L and the highest value is g_H. We can transform all values in the range (g_L-g_H) the range 0–255 by the linear transformation illustrated below.



Since the gradient of the line is 255/((g_H-g_L) we can write the equation:

$$y = 255\frac{x - g_L}{g_H - g_L}$$

and applying this transformation to all grey levels x produced by the filter will result (after any necessary rounding) in an image which can be displayed.

Gaussian filters:

Gaussian filters are a class of low-pass filters, all based on the Gaussian probability distribution function

$$f(x) = e^{-\frac{x^2}{2\sigma^2}}$$

Where σ is the standard deviation: a large value of σ produces to a flatter curve, and a small value leads to a "pointier" curve filter.

Large value of σ Small value of σ

A two dimensional Gaussian function is given

$$f(x, y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Gaussian filters have a blurring effect which looks very similar to that produced by neighborhood averaging. Although the results of Gaussian blurring and averaging look similar, the Gaussian filter has some elegant mathematical properties which make it particularly suitable for blurring. We see that to obtain a spread out blurring effect, we need a large standard deviation. In fact, if we let the standard deviation grow large without

bound, we obtain the averaging filters as limiting values as shown in Figure (4).

Figure (4) : Effects of different Gaussian filters on an image.

Non-linear filters Linear :

Linear filters, as we have seen in the previous sections, are easy to describe, and can be applied very

quickly and efficiently by Mat lab. non-linear filters can also be used, if less efficiently. There are two types:

1-A corresponding filter which takes the maximum value can be just as easily applied

2-A corresponding filter which takes the minimum value can be just as easily applied

(a) Using a maximum filter

(b) Using a minimum filter

Exercises

1. The array below represents a small greyscale image. Compute the images that result when the image is convolved with each of the masks (a) to (h) shown. At the edge of the image use a restricted mask. (In other words, pad the image with zeroes.)

20	20	20	10	10	10	10	10	10
20	20	20	20	20	20	20	20	10
20	20	20	10	10	10	10	20	10
20	20	10	10	10	10	10	20	10
20	10	10	10	10	10	10	20	10
10	10	10	10	20	10	10	20	10
10	10	10	10	10	10	10	10	10

20	10	20	20	10	10	10	20	20
20	10	10	20	10	10	20	10	20

(a)
-1	-1	0
-1	0	1
0	1	1

(b)
0	-1	-1
1	0	-1
1	1	0

(c)
-1	-1	-1
2	2	2
-1	-1	-1

(d)
-1	2	-1
-1	2	-1
-1	2	-1

(e)
-1	-1	-1
-1	8	-1
-1	-1	-1

(f)
1	1	1
1	1	1
1	1	1

(g)
-1	0	1
-1	0	1
-1	0	1

(h)
0	-1	0
-1	4	-1
0	-1	0