Name : Sarvagya Singh **LAB5** **SAPID : 60009200030(K1)**

AIM: To perform Image Enhancement(SD) Neighbourhood Processing Techniques: Sharpening Operators

THEORY:

1. **Basic High Pass Filter**

The principal objective of high pass (sharpening) filter is to highlight fine detail in an image or to enhance detail that has been blurred, either in error or as a natural effect of a particular method of image acquisition. Uses of image sharpening vary and include applications ranging from electronic printing and medical imaging to industrial inspection and autonomous target detection in military systems.

The shape of the impulse response needed to have a high pass (sharpening) spatial filter indicates that the filter should have positive coefficients in the outer periphery. For a 3 x 3 mask, choosing a positive value in the centre location with negative coefficients in the rest of the mask meets this condition. Thus when the mask is over an area of constant or slowly varying gray level, the output of the mask is zero or very small. This result is consistent with what is expected from the corresponding frequency domain filter.

-1	-1	-1
-1	8	-1
-1	-1	-1

Fig: A high Pass {Laplacian} filter mask

2. **High Boost Filtering**

A process used for many years in the publishing industry to sharpen images consists of subtracting a blurred version of an image from the image itself.

This process, called unsharp masking, is expressed as:

$$f_s(x,y) = f(x, y) - f'(x,y)$$

where, $f_s(x,y)$ denotes the sharpened image obtained by unsharp masking and $f'(x,y)$ is a blurred version of $f(x,y)$. The origin of unsharp masking is in dark room photography, where it consists of clamping together a blurred negative to corresponding positive film and then developing this combination to produce a sharper image.

A further generalization of unsharp masking is called high-boost filtering. A high-boost filtered image, f_{hb}, is defined at any point (x, y) as

$$f_{hb}(x, y) = A f(x,y) - f'(x,y)$$

where A=>1 and as before $f'(x,y)$ is the blurred image of $f(x,y)$

The final expression for high-boost image is given as

$$f_{hb}(x, y) = (A-1) f(x,y) + f_s(x,y)$$

The above given equation is applicable in general and does not state explicitly how the sharp image is obtained.

The mask for high-boost filtering is given as:

-1	-1	-1
-1	A+8	-1
-1	-1	-1

Fig: A mask for High-Boost filtering

One of the principal applications of high-boost filtering is when the input image is darker than desired. By varying the boost co-efficient, it generally is possible to obtain an overall increase in average gray level of the image, thus helping to brighten the final result.

RESULT:

A high pass filter is the basis for most sharpening methods. The high pass filters only allow those pixels with values above more values. The high pass filters have negative as well as positive values in the pixels. Due to this it creates a net values which results in highly distorted images. The high boost filter can be used to enhance the high frequency components. We can sharpen edges of a image through the amplification and obtain a more clear image. The high boost filter is simply the sharpening operator in image processing. High boost filter is a popular method that allows sharpening in high detail areas but little or no sharpening in flat or smooth areas. A high boost filter is used to retain some of the low-frequency components to and in the interpretation of a image.

Sarvagya Singh

60009200030 -- K1

IPCV - LAB5

In [ ]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from google.colab.patches import cv2_imshow
```

In [ ]:
```
img = cv2.imread('/content/6.jpg', 0)
cv2_imshow(img)
```

In [ ]:
```
img.shape
```

Out[ ]:
```
(534, 800)
```

In [ ]:
```
filter = np.array([[-1, -1, -1], [-1, 9, -1], [-1, -1, -1]])
filter
```

Out[ ]:
```
array([[-1, -1, -1],
       [-1,  9, -1],
       [-1, -1, -1]])
```

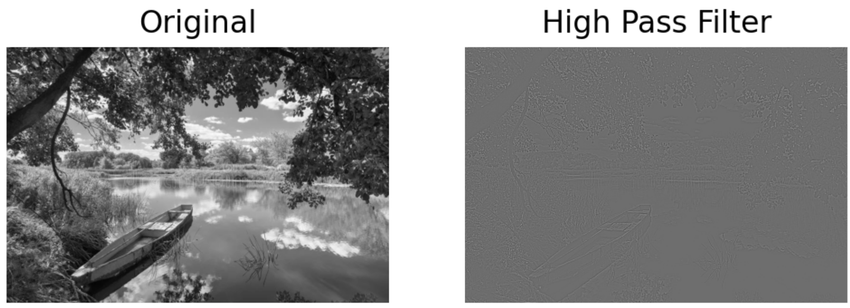
In [ ]:
```
output = []
for r in range(img.shape[0]-2):
  temp = []
  for c in range(img.shape[1]-2):
    lhs = img[r:r+3, c:c+3]
    ans = 0
    for r_f in range(filter.shape[0]):
      for c_f in range(filter.shape[1]):
        ans += lhs[r_f][c_f]*filter[r_f][c_f]
    temp.append(ans)
  output.append(temp)
```

In [ ]:
```
fig=plt.figure(dpi=200)

fig.add_subplot(1,2,1)
plt.imshow(img,cmap='gray')
plt.axis("off")
plt.title("Original")

fig.add_subplot(1,2,2)
plt.imshow(output,cmap='gray')
plt.axis("off")
plt.title("High Pass Filter")
```

Out[ ]:
```
Text(0.5, 1.0, 'High Pass Filter')
```



In [ ]:
```
filter_hb = np.array([[-1, -1, -1], [-1, 8.9, -1], [-1, -1, -1]])
filter_hb
```

Out[ ]:
```
array([[-1. , -1. , -1. ],
       [-1. ,  8.9, -1. ],
       [-1. , -1. , -1. ]])
```

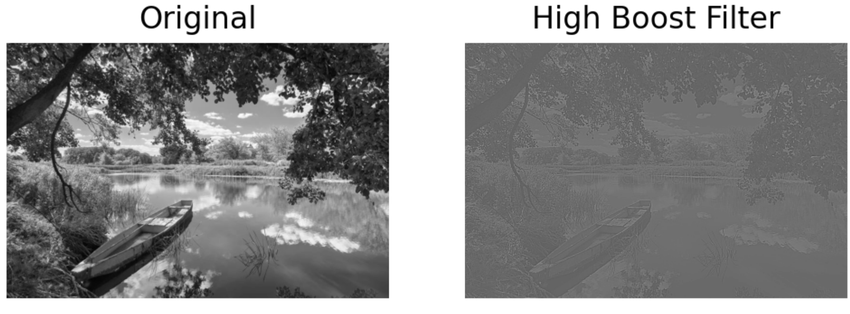
In [ ]:
```
output_hb = []
for r in range(img.shape[0]-2):
  temp = []
  for c in range(img.shape[1]-2):
    lhs = img[r:r+3, c:c+3]
    ans = 0
    for r_f in range(filter_hb.shape[0]):
      for c_f in range(filter_hb.shape[1]):
        ans += lhs[r_f][c_f]*filter_hb[r_f][c_f]
    temp.append(ans)
  output_hb.append(temp)
```

In [ ]:
```
fig=plt.figure(dpi=200)

fig.add_subplot(1,2,1)
plt.imshow(img,cmap='gray')
plt.axis("off")
plt.title("Original")

fig.add_subplot(1,2,2)
plt.imshow(output_hb,cmap='gray')
plt.axis("off")
plt.title("High Boost Filter")
```

Out[ ]:
```
Text(0.5, 1.0, 'High Boost Filter')
```



In [ ]:
```
!jupyter nbconvert --to html ''
```