Image Processing
Home work 03
Image Enhancement in Frequency Domain

Aqeel Labash
Lecturer: Gholamreza Anbarjafari

2 April 2016

Filtering Methods in Frequency Domain

1 Low Pass filters

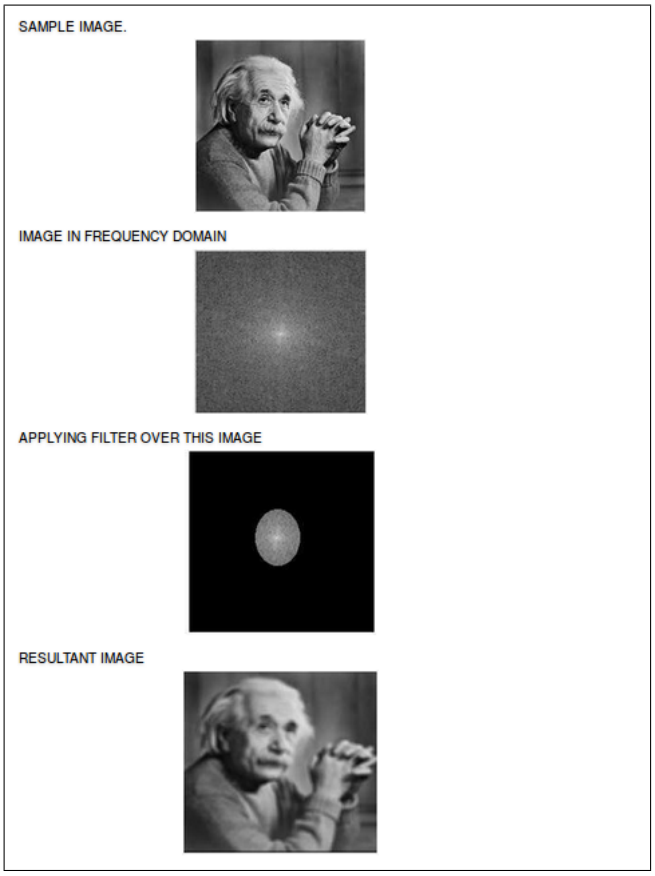
They are also called blurring filters.[1]

1.1 Ideal

In idea low pass filter we put zero for all the frequencies above the cutoff frequencies.So instead of rectangular pulse in 1D it's cylindrical in 2D.[3]
So the multiplication matrix is defined as following :

$$H(u,v) = \begin{cases} 1 \text{ if } D(u,v) \leq D_0 \\ 0 \text{ if } D(u,v) > D_0 \end{cases}$$

To understand it better here is an example showing the steps:



In the previous figure we can see the steps and how the frequency domain looks like after applying the filter.We saw also how the image become after we do the Fourier inverse.

1.2 Butterworth

Butter worth :"designed to have as flat a frequency response as possible in the passband"[**?**]. In butter worth we generate the multiplication matrix with the following formula :

$$H(u,v) = \frac{1}{1 + [\frac{D(u,v)}{D_0}]^{2n}}$$

In this way will have smooth change not 1 or 0 value. So The more far we are from D_0 the more close the value to zero.

1.3 Gaussian

The multiplication matrix is definded by the following formula :

$$H(u,v) = e^{\frac{-D^2(u,v)}{2D_0^2}}$$

Gaussian filter in general (low pass or high pass) solve the Ideal filter ringing problem.That's because if we noticed the formula we can see that's it's gradually change not working as switch on/off.

2 High Pass filters

The are also called derivative filters.[1]

2.1 Ideal

It's exactly the opposite of Ideal low pass filter where it allow the filters with high frequencies to pass and prevent other frequencies.Next figure shows an example of how the multiplication matrix looks like :

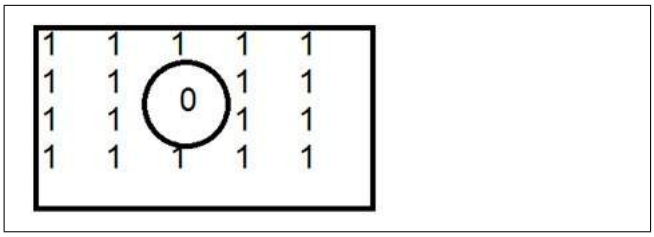


Figure 1: Shows example of how H(u,v) matrix would look like

The formula for $H(u, v)$ is defined:

$$H(u,v) = \begin{cases} 0 \text{ if } D(u,v) \leq D_0 \\ 1 \text{ if } D(u,v) > D_0 \end{cases}$$

2.2 Butterworth

The multiplication matrix is given by the following formula:

$$H(u,v) = \frac{1}{1 + [\frac{D_0}{D(u,v)}]^{2n}}$$

If we look more to the formula we can see it's the same as Butterworth for low pass frequency after reversing $D_0, D(u, v)$ places which will allow us to reverse the behavior of the filter to pass the high frequencies.

2.3 Gaussian

As I wrote earlier Gaussian filter in general (low pass or high pass) solve the Ideal filter ringing problem. $H(u, v)$ is given by the following formula :

$$H(u,v) = 1 - e^{\frac{-D^2(u,v)}{2D_0^2}}$$

From the previous formula we can notice that we are taking the complement value and we can write as :

$$H(u,v) = 1 - GLPF$$

Where GLPF = Gaussian low pass filter.

3 Implementation Part

Figure 2: Left : orignal , Right: Gaussian noised picture

The code used to implement Gaussian noise the image is here :

```
def NoiseImage(img):
my_nois = np.reshape(np.random.randint(5,20,img.shape[0]*img.shape[1]) ,(img.shape[0],img.shape
    [1]))
return Fiximage(img+my_nois)
```

Figure 3: Left : Ideal low pass filter , Right: ideal high pass filter

```
def distance(u,v):
return np.sqrt(pow(float(u),2)+pow(float(v),2))

def ILPF(img,d0):
imginFD = np.fft.fft2(img)
for i in range(imginFD.shape[0]):
for j in range(imginFD.shape[1]):
imginFD[i,j]*=distance(i,j)<=d0

return  np.array(np.fft.ifft2(imginFD),dtype=int)

def IHPF(img,d0):
imginFD = np.fft.fft2(img)
for i in range(imginFD.shape[0]):
for j in range(imginFD.shape[1]):
imginFD[i,j]*=distance(i,j)>=d0

return  np.array(np.fft.ifft2(imginFD),dtype=int)
```

Figure 4: Left : Gaussian low pass filter , Right: Gaussian high pass filter

The code used for the previous images :

```
def GHPF(img,d0):
imginFD = np.fft.fft2(img)
imginFD= np.array(imginFD,dtype=complex)
for i in range(imginFD.shape[0]):
for j in range(imginFD.shape[1]):
imginFD[i,j]*=1-exp(-pow(distance(i,j),2)/2*d0)

return   np.array(np.fft.ifft2(imginFD),dtype=int)

cv2.imwrite('mypic_GHPF.jpg',GHPF(noisedimage,50))
cv2.imwrite('mypic_GLPF.jpg',cv2.GaussianBlur(noisedimage,(5,5),20))
```

References

[1] high pass vs low pass filters

[2] image filters

[3] Butterworth filter