



GÖRÜNTÜ İŞLEME DERS-8 YARDIMCI NOTLARI -2018

Gri Seviye Dönüşümleri

Herhangi bir görüntü işleme operasyonu, görüntüdeki pikselin gri seviye değerlerini dönüştürme işlemidir. Ancak, görüntü işleme operasyonları; dönüşümü gerçekleştirmek için, ihtiyaç duyacağı bilgilere göre 3 sınıfa ayrılabilir. Bunlar en zordan en basite göre;

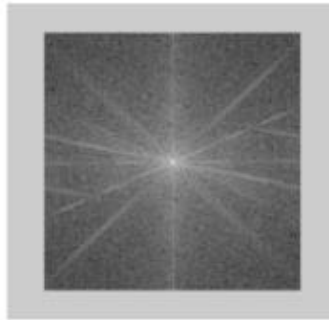
1- Transformlar (Dönüşümler): değişik domainlere dönüşüm yapılarak görüntü işleme işlemidir. (Bu derste uzaysal domain(Spatial domain) ve frekans domaininde (frequency domain) işlemler yapılacaktır.) Çok etkili ve verimli algoritmalar bu şekilde çalıştırılır. Bir dönüşümü kullanarak, tüm görüntünün tek bir büyük blok gibi işlenmiş olduğunu düşünebilirsiniz.

2- Komşuluk ilişkili (Neighbourhood processing-Bölgesel) işlemler: Belirli bir pikselin gri düzeyini değiştirmek için bilmemiz gereken tek şey verilen piksel etrafında küçük bir bölgedeki (komşuluk ilişkisinin olduğu yerde) gri düzeylerinin değeridir.

3- Noktasal İşlemler: Bir pikselin yeni gri seviye değerini, bağımsız olarak, etrafındaki piksel bilgilerine ihtiyaç olmadan elde etme işlemidir. Noktasal işlemler en basit işlemler olmasına rağmen birçok görüntü işleme operasyonlarında kullanılırlar. Özellikle bir görüntünün; ana işlemlerden geçirilmesine hazırlamak üzere kullanılırlar.

Uzaysal domain (Spatial Domain): Günlük hayatta kullandığımız sayısal resimlerin oluşturulduğu domaindir. Bu domaindeki resimlerin pikselleri doğrudan doğruya işlenebilir.

Frekans domain(Frequency domain): Görüntünün birçok farklı frekanslı bileşenden oluştuğu kabul edilir. Uzaysal domaindeki görüntü fourier v.b dönüşümü ile frekans domenine çevrilir. Burada işlenip ters dönüşüm yapılır.



Uzaysal Domain'de görüntü işlemleri

Herhangi bir fonksiyonda olduğu gibi, çeşitli operatörleri bir görüntüye uygulayabiliriz



$$g(x,y) = f(x,y) + 20$$



$$g(x,y) = f(-x,y)$$

Uzaysal domain teknikleri, bir görüntünün pikselleri üzerinde doğrudan işlem yapar. Bu domendeki işlemler aşağıdaki denklemlerle ifade edilir. Burada $f(x, y)$ giriş görüntüsüdür. $g(x, y)$ ise çıkış (işlenmiş) görüntüsüdür.

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

T ise f 'de belirli bir (x,y) komşuluk ilişkisi bölgesinde işlem yapan bir operatördür. Örneğin T operatörü; K görüntülerinde gürültü azaltmak için, bir görüntü seti işlemi olarak çalışabilir.

T ile belirtilen operasyonlar, Noktasal, Lokal(yerel) ve Global olarak yapılabilir. Noktasal Operasyon: Sadece 1x1 lik bölgede yapılan işlemlerdir. Nokta operasyonlarında, bir resimdeki **her pikselin gri seviyesi yalnızca onun orijinal gri seviyesinden(tonundan) hesaplanır.** Bu sebeple bu işlemlere "piksel değeri haritalama" veya "gri ton değişikliği" (modification) gibi isimler verilir.

Nokta operasyonları genellikle "resim onarımı" (manipulation) için kullanılır. Mesela, bir resmin kontrastının /yükseltilmesi gibi. Nokta operasyonları sıfır hafıza operasyonlarıdır.

Bölgesel (lokal-Komşuluk ilişkili) işlemlerde merkez pikselin değeri komşu piksellerin değeri ile belirlenir. Filtreleme işlemlerinde çok kullanılır.

10	5	3
4	5	1
1	1	7

Yerel görüntü
Verisi

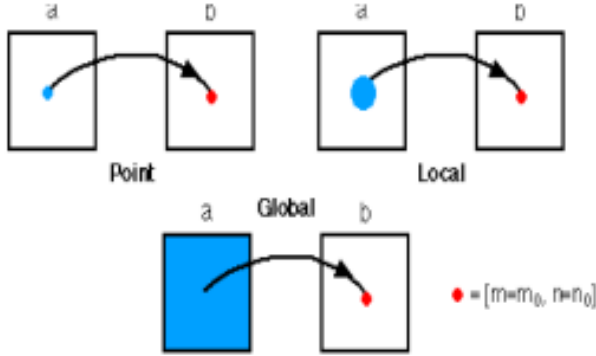
Bazı fonksiyonlar



	7	

Modifiye edilmiş
görüntü verisi

Global işlemlerde ise Domain dönüşümü (uzaysaldan frekans domenine veya tersi) yapılarak imge üzerinde işlem yapılır.



Bir piksel (x,y) 'in komşuluk bölgesi veya komşuluk ilişkisi için; merkezi (x,y) olan kare, dikdörtgen tanımlama kullanılır.

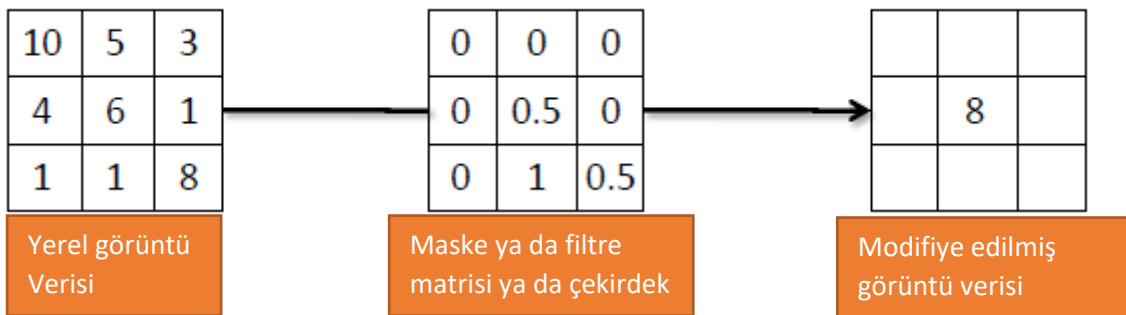
Bu bölgenin merkezi,pikselden piksele hareket ettirilerek (Her yöne), etrafındaki farklı komşuları içine alır. T operatörü herbir lokasyona (x,y) uygulanarak lokasyonda işlenmiş $g(x,y)$ çıkışı elde edilir.

En küçük lokasyon (komşuluk ilişkisi-bölgesel) resim içindeki $l \times l$ 'lik bölgedir. Bu bölge içinde yapılan işlemlerde (çalışılan pikselde) diğer piksellerin hiçbir etkisi olamaz. Yani o tek piksele yapılan işlemde komşu piksellerin rolü olamaz. Bu tür işlemlere Noktasal işlemler denir.

Burada T yapılan işlemi belirtir. Yani her piksele; komşuluk ilişkisine göre gezilerek T'nin belirttiği işlem yapılır.

FİLTRELEME TEKNİKLERİ

Doğrusal Filtreleme:



Matlab ortamında görüntü filtreleme işlemleri

`g = imfilter (f , w , filtering_mode , boundary_options , size_options)`



Options	Description
Filtering Mode	
'corr'	Filtering is done using correlation (see Figs. 3.14 and 3.15). This is the default.
'conv'	Filtering is done using convolution (see Figs. 3.14 and 3.15).
Boundary Options	
P	The boundaries of the input image are extended by padding with a value, P (written without quotes). This is the default, with value 0.
'replicate'	The size of the image is extended by replicating the values in its outer border.
'symmetric'	The size of the image is extended by mirror-reflecting it across its border.
'circular'	The size of the image is extended by treating the image as one period a 2-D periodic function.
Size Options	
'full'	The output is of the same size as the extended (padded) image (see Figs. 3.14 and 3.15).
'same'	The output is of the same size as the input. This is achieved by limiting the excursions of the center of the filter mask to points contained in the original image (see Figs. 3.14 and 3.15). This is the default.

Matlab standart doğrusal uzaysal filtreleri

Maske ya da Filtre matrisi olarak da isimlendirilen matrisleri oluşturmak için aşağıdaki komut kullanılır. Ancak özelleştirilmiş pek çok filtrenin kendisine özel fonksiyonları da mevcuttur.

`w = fspecial('type', parameters)`



Type	Syntax and Parameters
'average'	<code>fspecial('average', [r c])</code> . A rectangular averaging filter of size $r \times c$. The default is 3×3 . A single number instead of <code>[r c]</code> specifies a square filter.
'disk'	<code>fspecial('disk', r)</code> . A circular averaging filter (within a square of size $2r + 1$) with radius r . The default radius is 5.
'gaussian'	<code>fspecial('gaussian', [r c], sig)</code> . A Gaussian lowpass filter of size $r \times c$ and standard deviation <code>sig</code> (positive). The defaults are 3×3 and 0.5. A single number instead of <code>[r c]</code> specifies a square filter.
'laplacian'	<code>fspecial('laplacian', alpha)</code> . A 3×3 Laplacian filter whose shape is specified by <code>alpha</code> , a number in the range <code>[0, 1]</code> . The default value for <code>alpha</code> is 0.2.
'log'	<code>fspecial('log', [r c], sig)</code> . Laplacian of a Gaussian (LoG) filter of size $r \times c$ and standard deviation <code>sig</code> (positive). The defaults are 5×5 and 0.5. A single number instead of <code>[r c]</code> specifies a square filter.
'motion'	<code>fspecial('motion', len, theta)</code> . Outputs a filter that, when convolved with an image, approximates linear motion (of a camera with respect to the image) of <code>len</code> pixels. The direction of motion is <code>theta</code> , measured in degrees, counterclockwise from the horizontal. The defaults are 9 and 0, which represents a motion of 9 pixels in the horizontal direction.
'prewitt'	<code>fspecial('prewitt')</code> . Outputs a 3×3 Prewitt filter, <code>wv</code> , that approximates a vertical gradient. A filter mask for the horizontal gradient is obtained by transposing the result: <code>wh = wv'</code> .
'sobel'	<code>fspecial('sobel')</code> . Outputs a 3×3 Sobel filter, <code>sv</code> , that approximates a vertical gradient. A filter for the horizontal gradient is obtained by transposing the result: <code>sh = sv'</code> .
'unsharp'	<code>fspecial('unsharp', alpha)</code> . Outputs a 3×3 unsharp filter; <code>alpha</code> controls the shape; it must be in the range <code>[0, 1]</code> ; the default is 0.2.

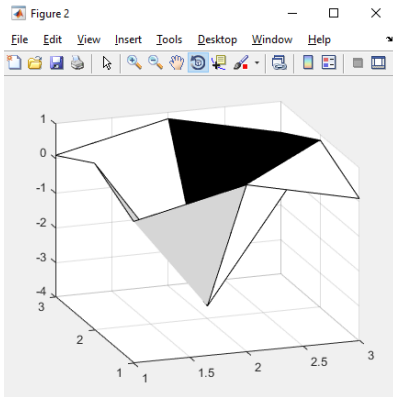
Filtreleme Örnekleri

<pre>%average filtre kullanımı; >> t=imread('cameraman.tif'); >> tg= imnoise(t,'salt & pepper', 1e-3); % gürültü ekleme >> b=fspecial('average', [5 5]); >> g=imfilter(tg,b,'replicate'); >> figure >> imshow(g)</pre>	<pre>% disk filtresi kullanımı >> b1=fspecial('disk',7); >> g1=imfilter(tg,b1,'replicate'); >> figure >> imshow(g1) %disk filtre mesh ve contour graf. >> b1=fspecial('disk',5); >> g1=imfilter(tg,b1,'replicate'); >> figure >> imshow(g1)</pre>
<pre>%laplacian filtre. Kenar bulmada etkili. b2=fspecial('laplacian',0.9); %alpha g2=imfilter(tg,b2); figure</pre>	<pre>%gaussian filtre. %b3=fspecial('gaussian',boyut,stdsapma); b3=fspecial('gaussian',[5 5], 2); g3=imfilter(tg,b3);</pre>

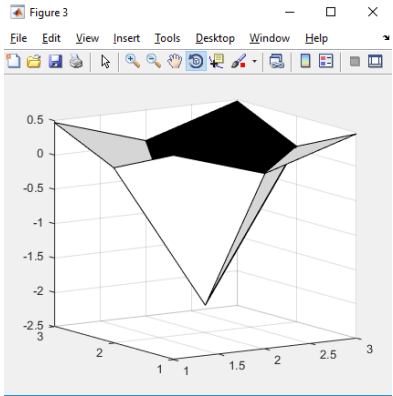


imshow(g2)

Alpha=0.1



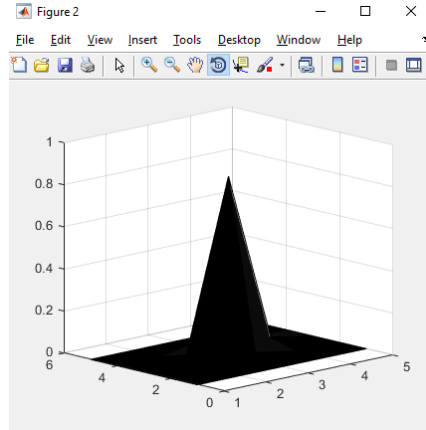
Alpha=0.9



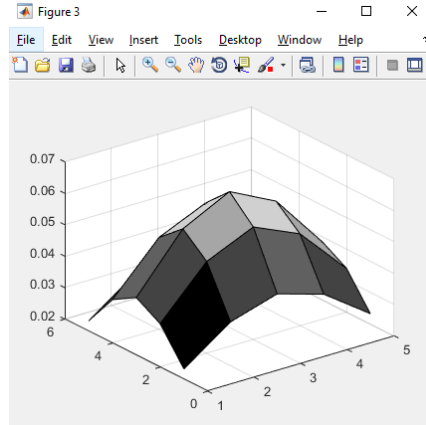
figure

imshow(g3)

stdsapma=0.4



Stdsapma=2



%unsharp filtresi

```
orj = imread('moon.tif');  
F = fspecial('unsharp');  
unsharpF = imfilter(orj, F);  
figure;  
subplot(1,2,1);  
imshow(orj);  
title('Resmin Orjinal Hali');  
subplot(1,2,2);  
imshow(unsharpF);  
title('Unsharp Filtresi');
```

%log filtresi

```
b4=fspecial('log')  
g4=imfilter(tg,b4);  
figure  
imshow(g4)  
  
g4_1=g4>100;  
figure,imshow(g4_1);  
g4_2=bwareaopen(g7_1,10);  
figure,imshow(g4_2)
```



Kenar Bulma:

<pre>%Prewitt yöntemi kenar bulma 1 prewitt=fspecial('prewitt'); g5=imfilter(t, prewitt); figure imshow(g5)</pre>	<pre>%Prewitt yöntemi kenar bulma 2 prewittF=edge(t,'prewitt'); figure imshow(prewittF)</pre>
<pre>%Sobel yöntemi kenar bulma 1 sobel=fspecial('sobel'); g6=imfilter(t,sobel); figure imshow(g6)</pre>	<pre>%Sobel yöntemi kenar bulma 2 sobelFiltresi=edge(t,'sobel'); figure imshow(sobelFiltresi)</pre>
<pre>%Log yöntemi kenar bulma logFiltresi=edge(t,'log');</pre>	<pre>%canny yöntemi cannyFiltresi=edge(t,'canny'); figure imshow(cannyFiltresi)</pre>
<pre>%roberts yöntemi robertsFiltresi=edge(t,'roberts'); figure imshow(robertsFiltresi)</pre>	

Subplot ile çizim:

```
%%
t=imread('cameraman.tif');
subplot(3,3,1)
imshow(t)
title('Orjinal Görüntü')
canny=edge(t,'canny');
subplot(3,3,2)
imshow(canny)
title('Canny Görüntü')
roberts=edge(t,'Roberts');
subplot(3,3,3)
imshow(roberts)
title('Roberts Görüntü')
prwt=edge(t,'prewitt');
subplot(3,3,4)
imshow(prwt)
title('Prewitt Görüntü')
sobel=edge(t,'sobel');
subplot(3,3,5)
imshow(sobel)
title('Sobel Görüntü')
gaus=fspecial('gaussian');
g1=imfilter(t,gaus,'replicate');
subplot(3,3,6)
```



AKÜ TEKNOLOJİ FAKÜLTESİ MEKATRONİK MÜHENDİSLİĞİ BÖLÜMÜ



```
imshow(g1)
title('Gaussian Görüntü')

laplace=fspecial('laplacian');
g2=imfilter(t,laplace,'replicate');
subplot(3,3,7)
imshow(g2)
title('Laplacian Görüntü')

log=fspecial('log');
g3=imfilter(t,log,'replicate');
subplot(3,3,8)
imshow(g3)
title('Log Görüntü')

%%
```

LİNEER OLMAYAN FİLTRELEME

Doğrusal olmayan uzaysal filtreleme, komşuluk işlemlerine de dayanır ve bir $m \times n$ filtrenin merkez noktasını bir görüntü boyunca kaydırmanın mekaniği, önceki bölümde anlatılanla aynıdır. Bununla birlikte, lineer uzamsal filtreleme, (lineer bir işlem olan) çarpımların toplamının hesaplanmasına dayansa da, adından da anlaşılacağı üzere doğrusal olmayan uzamsal filtreleme, piksel komşulukları içindeki filtre tarafından kapsanan pikselleri içeren doğrusal olmayan işlemlere dayanır. Örneğin, maksimum filtresinde her orta noktadaki filtrelenmiş değer kendi komşuluğundaki maksimum piksel değerine eşittir. Bu da doğrusal olmayan bir işlemdir. Bir diğer temel fark ise, maske kavramının doğrusal olmayan işlemlerde yaygın olmadığıdır.

“ordfilt2” fonksiyonu, sıralama-istatistik filtreleri (sıra filtreleri olarak da bilinir) oluşturur. Bunlar, işlem yapılan noktadaki piksellerin sıralanmasına (sıralamaya) ve daha sonra çevredeki merkez piksel değerinin sıralama sonucu tarafından belirlenen değer ile değiştirilmesine dayanan doğrusal olmayan uzaysal filtrelerdir.

```
g = ordfilt2(f, order, domain)
```

Burada ordfilt2 filtresi f görüntü matrisi içinde “domain” kısmında belirtilen büyüklükte filtre matrisinin “order” ‘ncü elemanını yanıt olarak belirler.



Type of Filtering Operation	MATLAB code	Neighborhood									
Median filter	<code>B = ordfilt2(A,5,ones(3,3))</code>	<table border="1"><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	1	1	1	1	1	1	1	1	1
1	1	1									
1	1	1									
1	1	1									
Minimum filter	<code>B = ordfilt2(A,1,ones(3,3))</code>	<table border="1"><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	1	1	1	1	1	1	1	1	1
1	1	1									
1	1	1									
1	1	1									
Maximum filter	<code>B = ordfilt2(A,9,ones(3,3))</code>	<table border="1"><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	1	1	1	1	1	1	1	1	1
1	1	1									
1	1	1									
1	1	1									

```
t=imread('cameraman.tif');  
tg=imnoise(t,'salt & pepper');  
figure,imshow(tg)  
m=ordfilt2(t,5,ones(3));  
figure,imshow(m)  
mx=ordfilt2(tg,9,ones(3));  
figure,imshow(mx)  
mn=ordfilt2(tg,1,ones(3));  
figure,imshow(mn)
```

Median Filtreleme

Median filtreleme, tuz-biber gürültüsünü yok etmek için çok uygundur. Medyan filtreler nonlineer uzaysal filtrelerdir. Maskeyi oluşturan boyuttaki resim piksel değerlerinin küçükten büyüğe sıralanıp ortadaki değeri merkez piksele atama işlemidir. Örneğin;

50	65	52
63	255	58
61	60	57

 → 50 52 57 58 **60** 61 63 65 255 → 60

```
>>t=imread('cameraman.tif');  
>>c=imnoise(t,'salt & pepper',0.1);  
>>imshow(c)  
>>d=medfilt2(c);  
>>imshow(d)
```



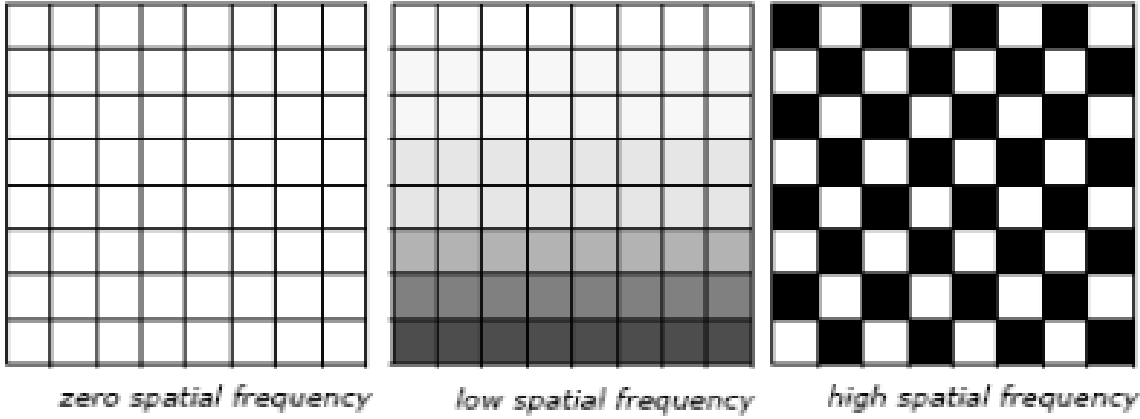
%kendi komutuyla medyan filtreleme

```
md=medfilt2(tg);  
figure,imshow(md)
```

FREKANS UZAYI

- İmge uzayında yapılabilecek işlemlerin yanında, frekans düzlemindeki bilgi de imge işlemede sıkça kullanılmaktadır.
- Daha önce imge filtreleme için konvülüsyondan bahsedilmişti. İmge uzayında yapılan bu işlem her bir piksel için tekrarlanmakla birlikte, çekirdek elemanına bağlı olarak hesapsal yükü oldukça fazla olabilmektedir.
- Frekans uzayına geçildiğinde konvülüsyon işlemi çarpma işlemine dönüşeceği için, bu uzayda yapılacak filtreleme işlemlerinde frekans uzayına geçiş ve geri dönüş işlemleri için hesapsal yükten bahsedilebilir.
- Ayrıca frekans uzayında imgedeki piksellerin dağılımına ilişkin bilgileri gözlemlemek de mümkündür.
- Frekans uzayına geçiş için genellikle Fourier dönüşümü kullanılmaktadır.

Uzaysal Frekanslar:



Fourier Dönüşümü

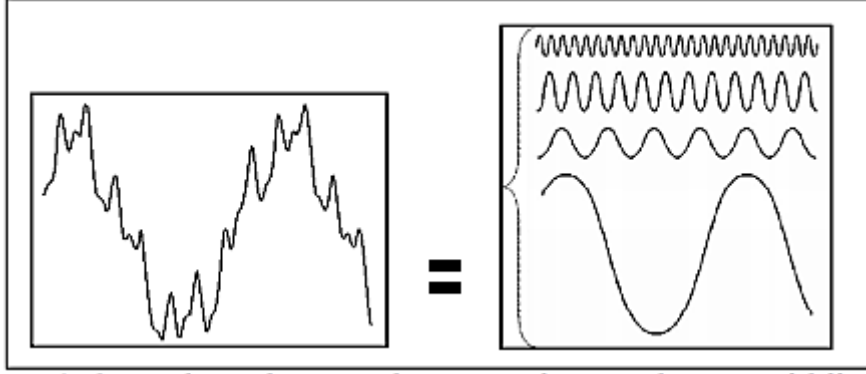
Bir uzaysal alan görüntüsünün geometrik özelliklerine erişmek istiyorsak Fourier Dönüşümü kullanılır. Fourier domenindeki görüntü sinüzoidal bileşenlerine ayrıştırıldığı için, görüntünün belirli frekanslarını incelemek veya işlemek, böylece uzamsal alandaki geometrik yapıyı etkilemek kolaydır.

Bu dönüşüm, görüntü işleminin çok önemli konularından biridir. Uzaysal domainde başarılması zor işlemleri, frekans domain'inde başaracak yapıda olan bu dönüşüm, **“görüntüyü oluşturan frekans**



bileşenlerini birbirinden ayırt edebildiği için değişik derecelerden alçak ve yüksek geçiren filtreleme işlemleri” kolaylıkla başarılabilir.

- Önce tek boyutlu sonra iki boyutlu Fourier dönüşümünü kısaca hatırlayalım;



Periyodik ve sonlu değer alabilen her fonksiyon, değişik frekanslarda titreşen sinüs veya cosinüslü bileşenlerin toplamından oluşur.

$$f(x) = f(x + T) \quad (-\infty < x < \infty)$$

$$f(x) = \frac{1}{2}a_0 + \sum_{k=1}^{\infty} a_k \cos(\omega kx) + b_k \sin(\omega kx) \quad (\omega = \frac{2\pi}{T})$$

$$a_k = \frac{2}{T} \int_0^T f(x) \cos(\omega kx) dx$$

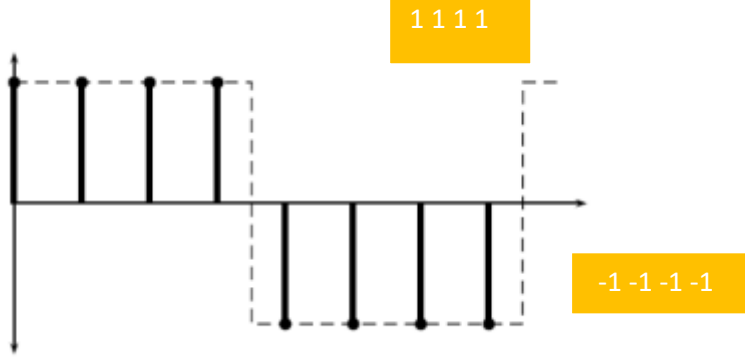
$$b_k = \frac{2}{T} \int_0^T f(x) \sin(\omega kx) dx$$

Fourier transformu bir görüntüye uygulandıktan sonra yüksek frekanslı geçiş yapılan piksel değerleri yüksek, düşük frekanslı geçiş yapılan frekansların değerleri düşük çıkar. Biz filtreleme işlemlerinde “yüksek frekanslı geçişleri filtrele” dersek alçak geçiren filtre oluşturmuş oluruz.



Tek boyutlu ayırık Fourier dönüşümü (DFT)

Herhangi bir fonksiyon ayrıklaştırıldığında sonlu sayıda elemanlı bir dizi şeklinde ifade edilir. Örneğin bir kare dalganın ayırık zamanlı dizi şeklinde ifadesi; 1 1 1 1 ; -1 -1 -1 -1 şeklinde olabilir.



$$\mathbf{f} = [f_0, f_1, f_2, \dots, f_{N-1}]$$

Eleman sayısı N olan bir ayırık f fonksiyonun, ayırık Fourier katsayıları dizisi aşağıda tanımlansın.

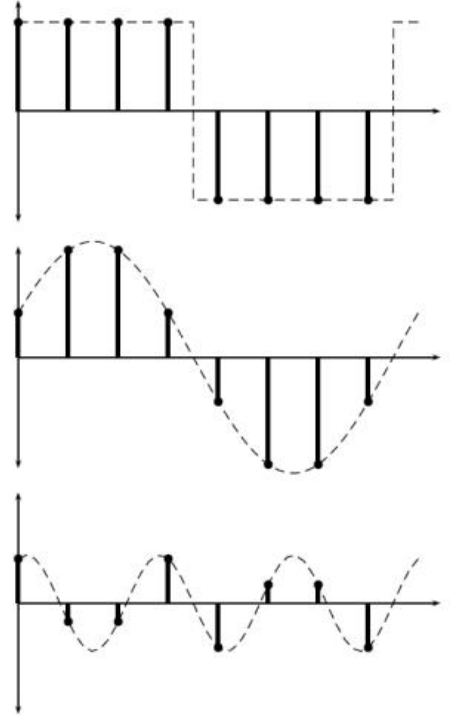
$$\mathbf{F} = [F_0, F_1, F_2, \dots, F_{N-1}]$$

Burada; u.ayırık fourier bileşninin katsayısı aşağıdaki bağıntıyla hesaplanır (DFT).

$$F_u = \frac{1}{N} \sum_{x=0}^{N-1} f(x) \cdot \exp \left[-2\pi i \frac{xu}{N} \right]$$

Aynı şekilde, fourier katsayılarından dizi elemanını elde etmek için ters fourier dönüşümü yapılır (IDFT).

$$f(x) = \sum_{u=0}^{N-1} F(u) \cdot \exp \left[2\pi i \frac{xu}{N} \right]$$



$$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) e^{-j2\pi ux/M}$$
$$u = 0, 1, 2, \dots, M - 1$$



Burada;

F(u) yu bulmak için:

- u=0 için x'in bütün değerlerinde yukarıdaki toplamı hesapla,
- u=1 için x'in bütün değerlerinde yukarıdaki toplamı hesapla.
-
-
-
- u=M için x'in bütün değerlerinde yukarıdaki toplamı hesapla.

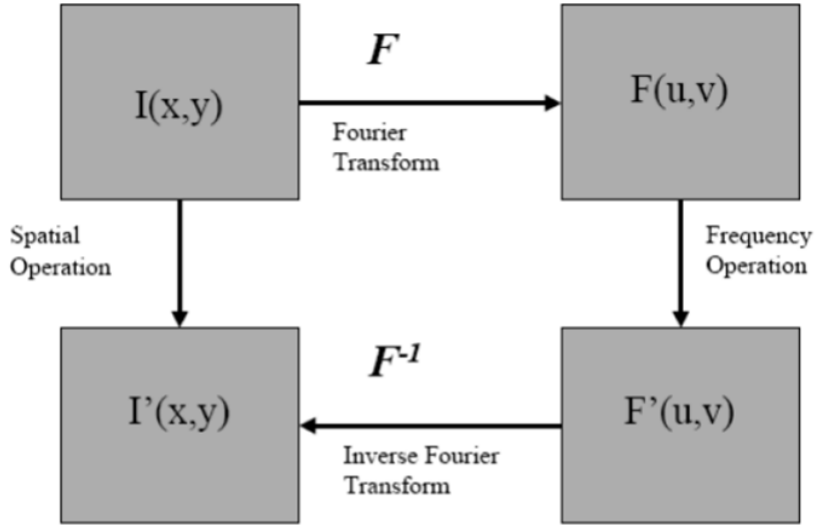
- Euler teoremine göre: $e^{j\theta} = \cos\theta + j\sin\theta$
 $e^{-j\theta} = \cos\theta - j\sin\theta$

$$F(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(x) [\cos(2\pi ux / M) - j\sin(2\pi ux / M)]$$
$$u = 0, 1, 2, \dots, M-1$$

- Özetle f(x) farklı frekanslardaki Sin ve Cos bileşenleri ile çarpılıyor.

$$\left. \begin{array}{l} F(0) \\ F(1) \\ \cdot \\ \cdot \\ F(M) \end{array} \right\} \begin{array}{l} \text{Dönüşümün} \\ \text{frekans} \\ \text{bileşenleri} \end{array}$$
$$F(u) = |F(u)| e^{-j\varphi(u)}$$
$$|F(u)| = [R^2(u) + I^2(u)]^{1/2} \quad \text{Genlik}$$
$$\varphi(u) = \tan^{-1} \left[\frac{I(u)}{R(u)} \right] \quad \text{Faz açısı}$$

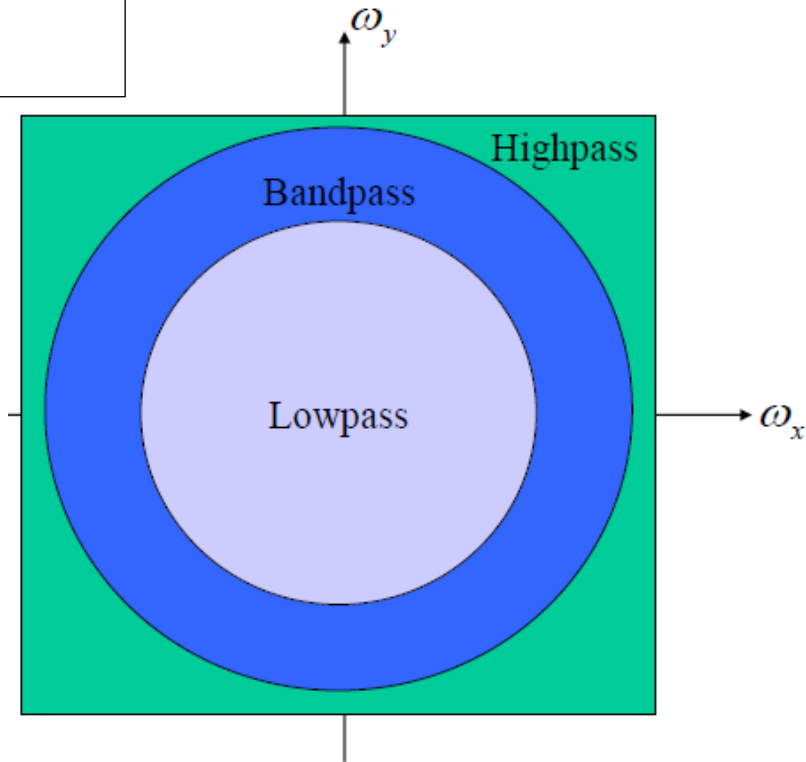
Frekans Uzayı – İmge İşleme Aşamaları



$$F'(u,v) = F(u,v)H(u,v)$$

$H(u,v) = \text{Filtre}$

Filtre Tipleri

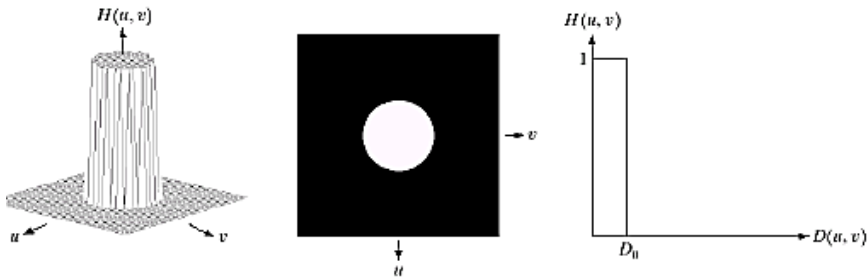




İdeal alçak geçiren filtre

Ideal Low-pass Filters: Cutoff Frequency

$$H(\omega_x, \omega_y) = \begin{cases} 1 & \sqrt{\omega_x^2 + \omega_y^2} \leq c \\ 0 & \text{else} \end{cases}$$



Butterworth Filters:

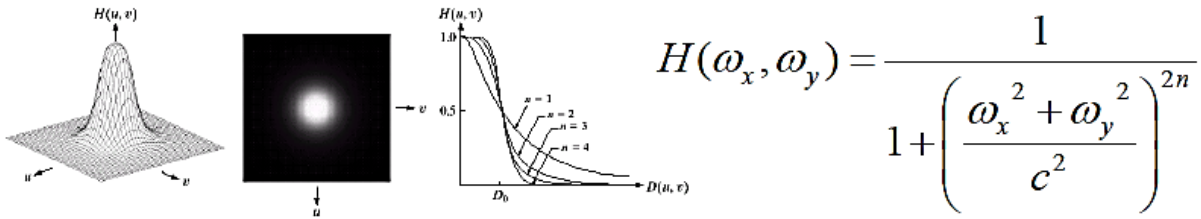


FIGURE 4.14 (a) Perspective plot of a Butterworth lowpass filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections of orders 1 through 4.

Gaussian Filters:

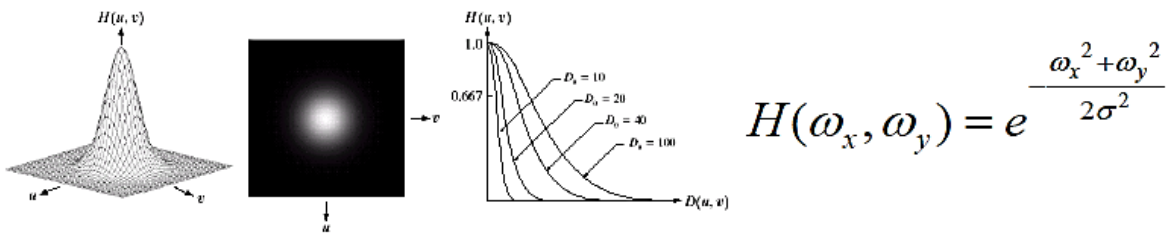


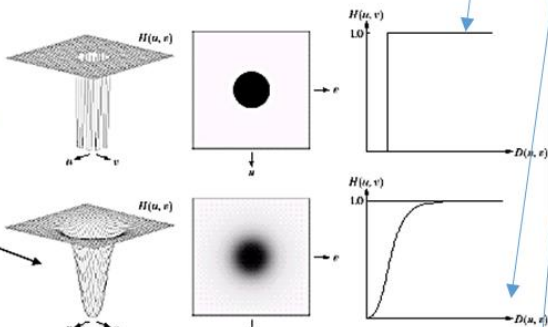
FIGURE 4.17 (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of D_0 .

Yüksek geçiren filtreler

Transfer Fonksiyonları

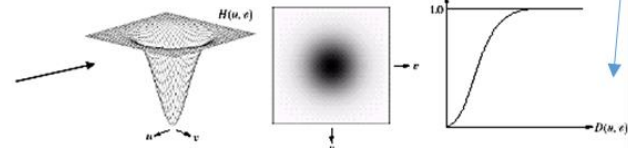
Butterworth Filters:

$$H(\omega_x, \omega_y) = \frac{1}{1 + \left(\frac{c^2}{\omega_x^2 + \omega_y^2} \right)^{2n}}$$



Gaussian Filters:

$$H(\omega_x, \omega_y) = 1 - e^{-\frac{\omega_x^2 + \omega_y^2}{2\sigma^2}}$$



$$h(x, y) = \delta(x, y) - \sqrt{2\pi}\sigma e^{-2\pi^2\sigma^2(x^2+y^2)}$$



FAST FOURIER DÖNÜŞÜMÜ

•DFT, ayrık fonksiyonların fourier dönüşümünü yapmak için çok önemli bir yapı olmakla beraber, dizi eleman sayısının çok fazla olduğu dönüşümler için uzun zaman harcamaktadır. Özellikle resim işleme gibi data sayısı fazla olan işlemler için yavaştır.

•Bunun yerine FFT(Hızlı fourier dönüşümü) ile bileşen katsayıları bulunur. Burada dikkat edilecek konunun dizinin 2'nin kuvvetleri sayısında elemandan oluşturulmasıdır.

Çoğu uygulamada Fourier görüntüsü, görüntünün merkezinde DC (Direct Current) değeri (yani görüntü ortalaması, yani frekansın sıfır olduğu bölgeleri temsil eden frekans değerleri) F(0,0) görüntülenecek şekilde kaydırılır. Merkezden uzak bir görüntü noktası, daha yüksek olan karşılık gelen frekanstır.

DC katsayısı ve kaydırma (shifting): DFT'deki DC katsayısı F(0,0) değeridir. Aşağıdaki denklemde v=u=0 konulduğunda bulunur. Buna göre DC katsayısının değeri orijinal görüntü matrisindeki tüm eleman değerlerinin toplamı olarak ortaya çıkar. Bu değer yeni matrisin en üst sol köşesindedir.

2-D DFT

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M+vy/N)}$$

for $u = 0, 1, 2, \dots, M - 1$ and $v = 0, 1, 2, \dots, N - 1$.

Kaydırma: Görüntüleme amacı için; DC katsayısının matrisin orta nokta elemanı olarak oluşması uygundur. Bunun için kaydırma yapılabilir. Kaydırma işlemi, dönüşümden önce f(x,y) matrisinin tüm elemanlarının (-1) ^(X+Y) ile çarpılacağı anlamındadır.

Dönüşümün gösterilmesi

• F(x,y) görüntüsünden elde edilen F(u,v) fourier katsayılar matrisinin elemanları kompleks sayılardır. Kompleks sayılar DOĞRUDAN görüntülenemez.



- Onların magnitüdüleri (genlikleri) $|F(u,v)|$ alınır. Bunlar bir takım double sınıfı büyük sayılardır. Bu büyük sahada uğraşmak için;
- 1-) $|F(u,v)|$ 'deki en büyük değer m bulunur (Bu DC değerdir) ve $|F(u,v)|/m$ işlemi yapılır. Böylece imshow ile görüntülenebilir.
- 2-) $|F(u,v)|$ 'yi görmek için mat2gray fonksiyonunu doğrudan kullanabiliriz.

mat2gray fonksiyonu: double sınıfı bir dizinin yine double sınıfı fakat eleman değerlerinin 0, 1 arasına yerleştirilmesi istendiğinde kullanılır.

$$G=\text{mat2gray}(A,[A_{\min},A_{\max}])$$

```
>> x=[1 2;3 4];
```

```
>> y=mat2gray(x)
```

```
y =
```

```
0      0.3333
```

```
0.6667  1.0000
```

- Genelde DC değerler çok büyük değerler olacağından; Bunun oluşturacağı görüntüde bu değer baskın çıkar. Bunun önüne geçmek için $|F(u,v)|$ 'nın logaritmasının alınması daha uygundur.

```
log(1+ |F(u,v)|)
```

- Fourier dönüşümün genliğinin görüntülenmesi, transformasyonun spektrumu diye isimlendirilir.



MATLAB'da Fourier Dönüşümü

fft :Tek boyutlu DFT yapar(çıkışı vektördür)

ifft : DFT vektörünün tersini alır. (inverse fft)

fft2: 2 boyutlu DFT yapar. Çıkışı Matristir.

ifft2: DFT matrisinin tersini alır.

fftshift: Bir dönüşümü kaydırır.

$Y = \text{fftshift}(X)$ sıfır frekans bileşenini dizinin merkezine kaydırarak X 'in Fourier dönüşümünü yeniden düzenler.

Kaynak:

Yrd. Doç. Dr. İlhan AYDIN Ders Notları