GÖRÜNTÜ İŞLEME YARDIMCI NOTLARI -GÜZ 2021-2022

GÖRÜNTÜ FİLTRELEME -2

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.

Uzaysal Domain'de görüntü işlemleri

Herhangi bir fonksiyonda olduğu gibi, çeşitli operatörleri bir görüntüye uygulayabiliriz

$$g(x,y) = f(x,y) + 20 \qquad g(x,y) = f(-x,y)$$

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

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

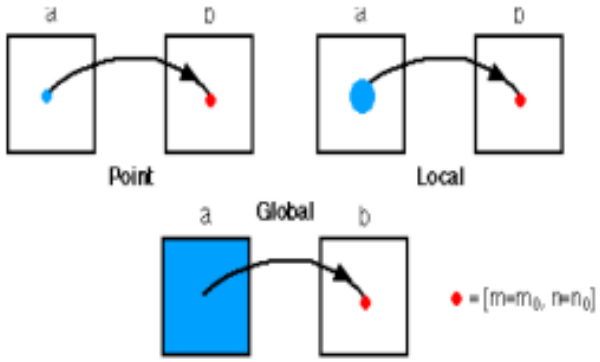
T ise f'de belirli bir (x,y) komşuluk ilişkisi bölgesinde işlem yapan bir operatördür. Örneğin T operatörü; K görüntülerinde gürültü azaltmak için, bir görüntü seti işlemi olarak ta çalışabilir.

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(parlaklık değerinden) hesaplanır.** Bu sebeple bu işlemlere "piksel değeri haritalama" yeya "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 İlişkili) işlemlerde merkez pikselin değeri komşu piksellerin değeri ile belirlenir. Filtreleme işlemlerinde çok kullanılır. Konvolüsyon tekniği en çok kullanılan tekniktir.

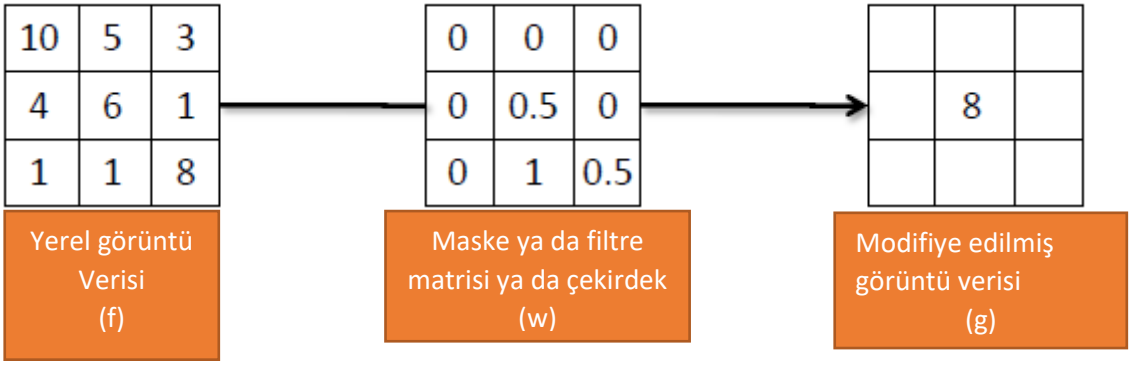
Global İşlemlerde ise Domain dönüşümü (uzaysaldan frekans domenine veya tersi) yapılarak imge üzerinde işlem yapılır.



FİLTRELEME TEKNİKLERİ

LOKAL (BÖLGESEL) İŞLEMLER

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.

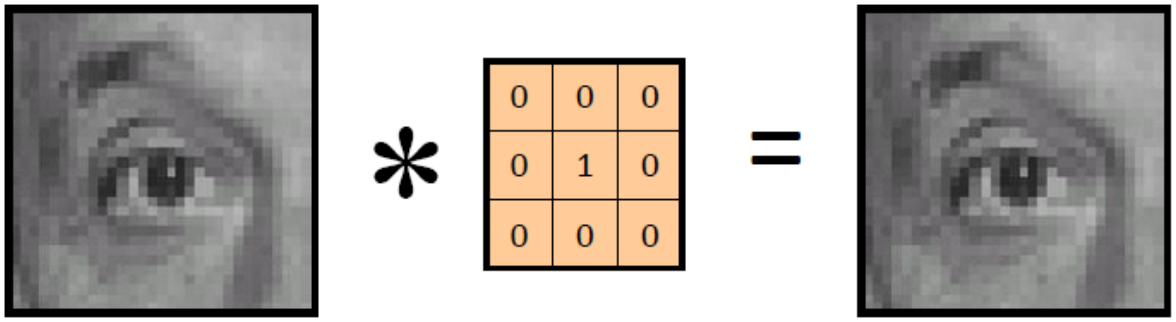
Matlab standart doğrusal uzaysal filtreleri

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

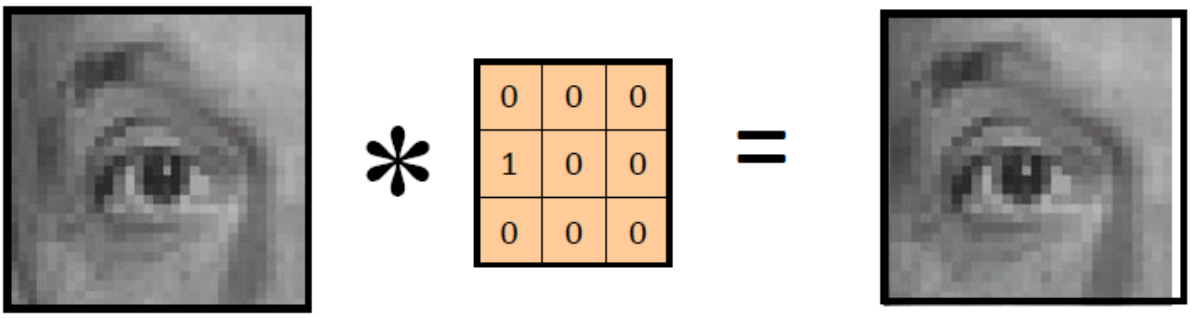
w = f s p e c i al (' t ype ' , parameters)

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

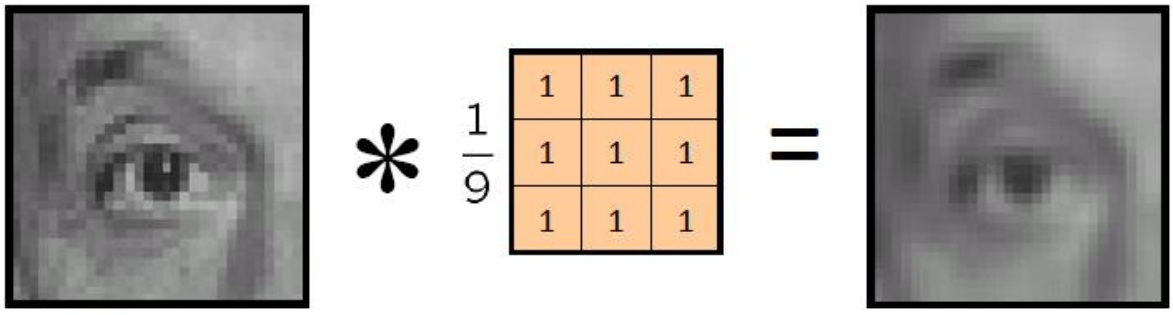
Bir kısım Doğrusal Filtreleme İşlemleri:



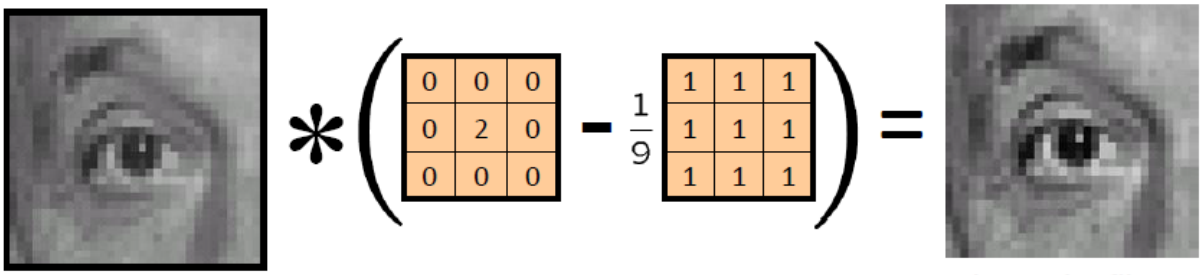
Original — Identical image



Original — Shifted left By 1 pixel



Original — Blur (with a mean filter)



Original — **Sharpening filter** (accentuates edges)

Kaynak: D. Lowe

Filtreleme Örnek İşlemleri

%average filtre kullanımı; >> x=imread('cameraman.tif'); >> filtre=fspecial('average', [5 5]); >> g=imfilter(x,filtre); >> figure >> imshow(g)	% disk filtresi kullanımı >> b1=fspecial('disk',3); >> g1=imfilter(x,b1); >> figure >> imshow(g1)

%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(x,b4);
	figure
	imshow(g4)
	
	g4_1=g4>100;
	figure,imshow(g4_1);
	g4_2=bwareaopen(g7_1,10);
	figure,imshow(g4_2)

Kenar Bulma:

%Prewitt yöntemi kenar bulma 1
prewitt=fspecial('prewitt');
g5=imfilter(t, prewitt);
figure
imshow(g5)
	%Prewitt yöntemi kenar bulma 2
	prewittF=edge(t,'prewitt');
	figure
	imshow(prewittF)

%Sobel yöntemi kenar bulma 1
sobel=fspecial('sobel');
g6=imfilter(t,sobel);
figure
imshow(g6)
	%Sobel yöntemi kenar bulma 2
	sobelFiltresi=edge(t,'sobel');
	figure
	imshow(sobelFiltresi)

%Log yöntemi kenar bulma logFiltresi=edge(t,'log');	%canny yöntemi cannyFiltresi=edge(t,'canny'); figure imshow(cannyFiltresi)
%roberts yöntemi robertsFiltresi=edge(t,'roberts'); figure imshow(robertsFiltresi)	

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)
       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')
%%
```

Filtreler Kullanılarak İmge İyileştirme

Aşağıdaki işlemlerin 4 aşamasını takip ediniz.

```
1.)%İmgenin bir bölümü orjinalinden
%kırpılıp alındı.
clear;close all;clc;
x=imread('cameraman.tif');
y=imcrop(x,[91,31,51,51]);
z=imresize(y,5);
figure(1);imshow(z)
```

Figure1

```
2.)%Yeni imge 5x5 lik ortalama
filtre %ile blurlaştırıldı
avg5=ones(5)/25;
zDouble=double(z);
z_avg5=conv2(zDouble,avg5,'same');
figure(2);imshow(uint8(z_avg5));
title('Figure2')
```

Figure2

```
3.)%İlk imgeden blurlaştırılan
%imge çıkartıldı.
fark=(zDouble-z_avg5);
figure(3);imshow(uint8(fark));
title('Figure3')
```

Figure3

```
4.)%Oluşan farkın alfa katı ile
ilk imge toplanarak
%daha keskin bir imge elde
edildi.Yani ilk imge
%keskinleştirildi.
alfa=2.2;
farkAlfa=alfa.*fark;
zYeni=z+uint8(farkAlfa);
figure(4);imshow(zYeni);
title('Figure4');
```

Figure4

Gauss filtreleri, imgedeki yüksek frekanslı bileşenleri baskılarken, alçak frekanslı bileşenleri geçiriri. Yani bir çeşit alçak geçiren filtre görevi görür.

LİNEER OLMAYAN FİLTRELEME

Doğrusal olmayan uzaysal filtreleme, komşuluk işlemlerine de dayanır ve bir m x 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 işlemde 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" 'ncı elemanını yanıt olarak belirler.

Type of Filtering Operation	MATLAB code	Neighborhood
Median filter	`B = ordfilt2(A,5,ones(3,3))`	1 1 1 1 1 1 1 1 1
Minimum filter	`B = ordfilt2(A,1,ones(3,3))`	1 1 1 1 1 1 1 1 1
Maximum filter	`B = ordfilt2(A,9,ones(3,3))`	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 filtrelerdi. Maskeyi oluşturan boyuttaki resim piksel değerlerinin küçükten büyüğe sıralanıp ortadaki değeri merkez piksele atama işlemiydi. Örneğin;

50	65	52
63	255	58
61	60	57

$\longrightarrow$ 50 52 57 58 **60** 61 63 65 255 $\longrightarrow$ 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)
```