FİGES

...../...../20.....

GÖRÜNTÜ İŞLEME

ARA TEKRAR

TEMEL BİLGİLER

08.12.2023

Doğukan Mehmet
KILINÇ

uint 8 → Bu sınıf görselin 0-255 arası değerden oluştuğunu söyler.

double → Değerler 0-1 arasındadır.

Aynı resimse aynı çıktıyı verir. (Ama uint8 daha hızlı işlem yapar.)

FİGES

...../...../20.....

(1) img = imread ("image.jpg") → Resmi okur.

(2) rgb2gray (img) → resmi siyah-beyaz yapar.

(3) size (img) → resmin boyutunu verir.

(4) [a, b, c] = size(img) → img boyutlarını atar.

(5) imhist(img) → Resmin histogram haritasını çizer.

(6) imadjust (img, [0.5 0.9], [0.3 0.4])
↳ 0.5'ten küçükleri → 0.3'le
0.9'dan büyükleri → 0.4'le eşitler.
↳ Aradaki değerleri de belirli bir oranlama yaparak 0-1 arasında değerler atar.
↳ Görüntü yoğunluğu değerlerini ve renk haritasını ayarlar.

(7) im2bw(img) → Resimi binary formata getirir.

Double Görüntüyü uint8'e çevirmek için

(8) im2uint8 (img) kullanırız.

(9) imresize (img, [a b]); → Görüntü boyutlandırılır.
↳ (a x b) boyutunda ölçülendirilir.

(10) imwrite (img, "abj.png") → img nesnesini "abj.png" adıyla kaydeder.
↳ Eğer böyle kaydedersek %75 kalite ile kaydeder.
Eğer %100 kalite ile kaydetmek istersek

(11) imwrite (img, "abj.png", "Quality", 100);
↳ Burada da %100 kalite ile kaydeder.

Eğer "Quality" kullansam bile MATLAB bunu boyut olarak kaydediyor. Hiçbir kayıp yaşanmamak için;

(12) imwrite(img, "kayipsiz.JPG", "Mode", "lossless");
↳ Özellikle tavsiye edilen Bu görülmeyebilir fakat MATLAB'a yüklenince gösterilecektir.

İki resmi toplayabilmek için boyutları aynı olmalıdır. Üst üste yapıştırma gibi bir şey olur.

RESMİN NEGATİFİNİ ALMAK İÇİN)
1) Resmi okuyun
2) Resmi gri tona çevirin
3) Resmi double formata çevirin
4) Resimin matrisini 1'den çıkartın.

(13)
```
img = imread("resim.jpg");
img = im2gray(img))        ——→ uint8 formatında (0-255)
x = im2double(img);        ——→ double formatında (0-1)
img2 = 1-x;
imshow(img2)
```
DİREKT BUÇAKMA

ŞEMA

img2 = 255 - img ; yapabiliriz

HERHANGİ BİR RESMİ 2'lik, 4'lük veya 8'lik RENK SEVİYESİNDE GÖSTERMEK

Herhangi bir resmi "imread" ile okuyup bir değişkene içerisine attığımız zaman 256 farklı renk seviyesi ile hafızada tutar.

(14)
```
a = imread("image.JPG")
a = rgb2gray(a)
```

Beyaz - 255,
Siyah - 0

FİGES

...../...../20.....

Burada dönüşüm yapmak için eşik değeri (threshold) belirler ve ona göre atama gerçekleştiririz. Tüm bu işlemleri yapabilmek için öncelikle resmi double formatına çeviririz.

```
(15) doublea = im2double(a);
     [m,n]= size(a);

(16) for i=1:m;
       for j=1:n;
         if (doublea(i,j) >= 0.5)
           resk2(i,j)=1;
         else
           resk2(i,j)=0
         end
       end
     end
```

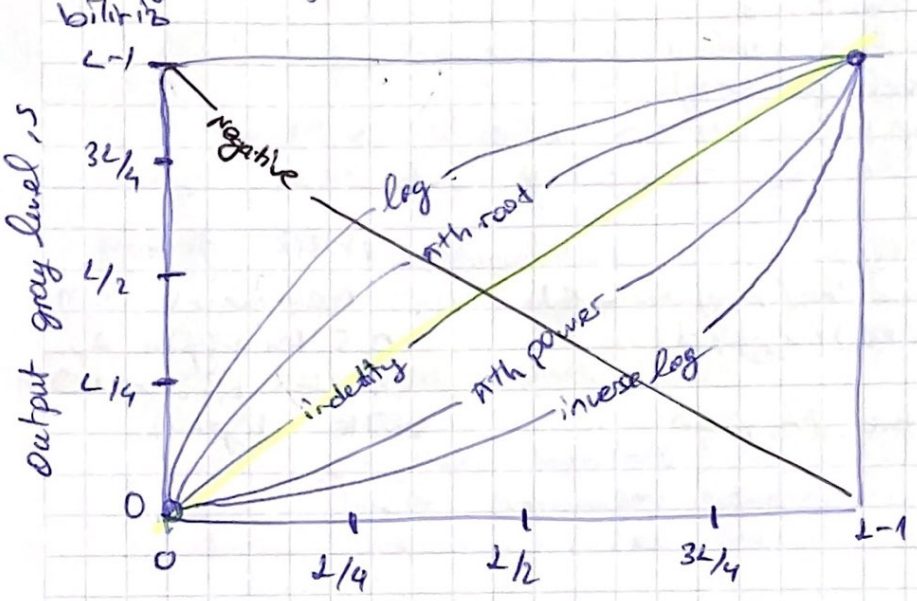
İkilik forma çevirmek için 0,5'ten büyükse → 1, 0,5'ten küçükse → 0'a eşitle diyoruz.

```
for i=1:m;
      for j=1:n;
(17)    if (doublea(i,j) <= 0.25)
          resk4(i,j)=0;
        elseif (0.25< doublea(i,j) <= 0.5)
          resk4 = 0.33 ;
        elseif (0.5< doublea(i,j) <= 0.75)
          resk4(i,j)=0.66 ;
        else
          resk4(i,j)=1;
        end
      end
end
```

Dörtlük forma çevirmek için

LOGARİTMİK TRANSFORM

Koyu değerlerimiz açık renk olsun diye yorumlayabiliriz



Logaritmik transformun genel fonksiyonu

$$s = c * \log(1+r)$$

Kod örneği de aşağıdaki gibidir.

(18)

```
a=imread("image.jpg");
adouble = im2double(a);
s=adouble;
[r,c]=size(adouble)
katsayi = 4;
for i=1:r
    for j=1:c;
        s(i,j) = katsayilar * log(1+adouble(i,j));
    end
end
subplot(1,2,1); imshow(adouble); title("Original");
subplot(1,2,2); imshow(a); title("after transform");
```

→ katsayıyı arttırdıkça görsel daha da beyaz tona dönüşür

FİGES

...../...../20.....

POWER LAW TRANSFORM

Power Law Transform formülü aşağıdaki gibidir.

$$s = c * r^{\gamma}$$

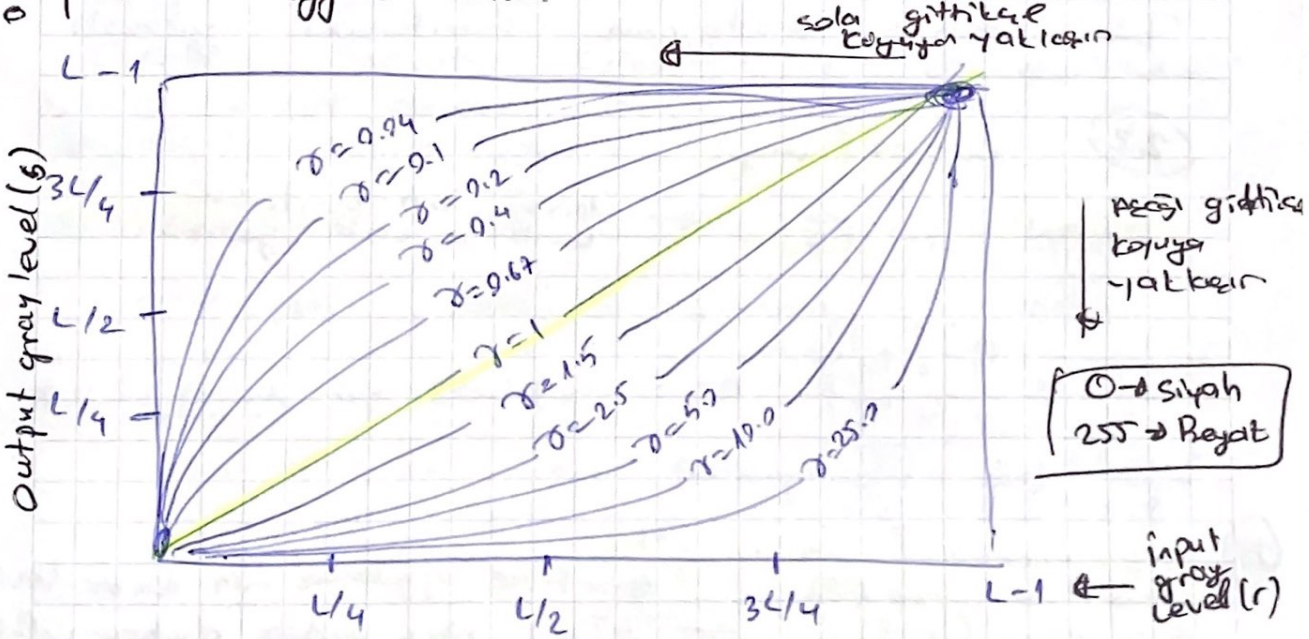
γ: Kaçıncı dereceden uygulamak istediğimizi bunu değiştirerek ayarlarız.

Bunun kodu da aşağıdadır.

```
katsayipower = 1;
    for i=1:r;
        for j=1:c;
            y(i,j) = katsayipower * double(i,j) ^5;
        end
    end
figure, imshow(y);
```

(15)

for döngüsünde yazmanızın sebebi resimdeki tüm pixellere uygulamaktır.



γ azalırsa; Koyu değerler → Açık değer olacak
γ artarsa; Açık değerler → Koyu değer olacak

FİGES

...../...../20.....

HİSTOGRAM GERME (HISTOGRAM EQUALIZATION)

Belirli bir alana sistemin histogramı germek isteriz. En küçük değer sıfır, en büyük değer de 255 olsun

(20)
```
im = [46, 86, 58, 26, 98, 155];
im_strech = (im - min(im(:))) * (255/(max(im(:)) - min(im(:))));
```
↳ Bu kod sonrası vektörün yayılmış hali görülür. Aynı işlem görüntüde de yapılabilir.

Bunun için kısa bir kodumuz vardır. Histogram germesi yapılmış görüntünün çıktısını görmek için

(21)
```
a = histeq(im);
figure;
imshow(a)
```

Bir resmin histogram haritasını görmek için ise)

(22) `imhist(img)`

MEAN FİLTRE ← Resmi biraz daha bulanık hale getirir

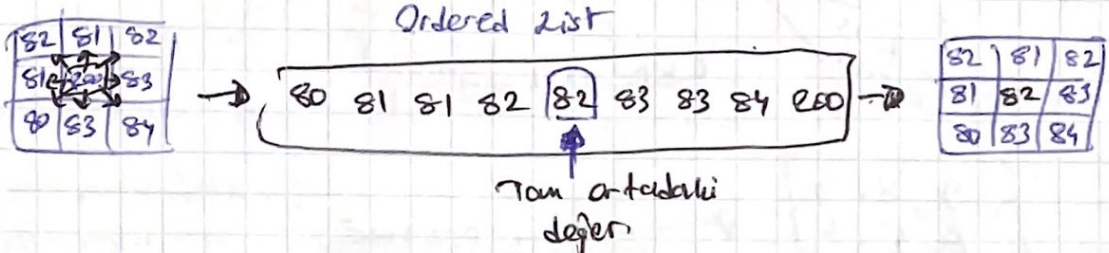
$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$ ← 3x3'lük bir MEAN FİLTER

Bunu kodda uygulayabilmek için)

(23)
```
img = imread("image.jpg");
imgd = im2double(img);   ← filtre uygulanak için double çevirdik.
f = ones(3,3)/9;         ← 3x3'lük birim matris oluşturur ve (÷9)
img1 = filter2(f, imgd); ← filtreyi resime uygular.
imshow(img1)             ← Resmi gösterir
```

MEDİAN FİLTRE

Tüm sayıları sıralarız ve tam ortadaki değeri yazarız.



Bunu kod kısmında şu şekilde yaparız.
→ Gürültü eklenmiş bir resim üzerinde çalışıyoruz.

```
I = imread("image.jpg");
J = imnoise(I, "salt & pepper", 0.02);   % Biz biber gürültüsü ekledik.
K = medfilt2(J)                          % median filtre uygulandı
imshow(K)
```

KORELASYON (ÇARPMAK)

Korelasyon aslında çarpım demektir. Eleman elemana çarpıp toplamak demektir. Matris boyutları aynı değilse kaydırarak çarpma yaparız.

KONVOLÜSYON (TERS ÇEVİRİP ÇARPMAK)

Konvolüsyon, ~~ters~~ çevirip çarpmak demektir.

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \times \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} = a.1 + b.2 + c.3 + d.4 + e.5 + f.6 + g.7 + h.8 + i.9$$

KORELASYON

FİGES

...../...../20.....

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} * \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} = i.1 + h.2 + g.3 + f.4 + e.5 + d.6 + c.7 + b.8 + a.9$$

Filtre ters çevrilir

KONVOLÜSYON

$$\begin{bmatrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{bmatrix}$$ ← YENİ FİLTRE

Bir kod sözdiziminde korelasyon ve konvolüsyon uygulamak için:

```
(25) image = imread("image.jpg");
     image = padarray(image, [1 1]);
     filtre = [-1,-1,-1; -1, 8,-1; -1,-1,-1]  % toplamları 0 olan filter
(26) r1 = conv2(image, filter);  % KONVOLÜSYON
(27) r2 = filter2(filter, image);  % KORELASYON

     figure;
     subplot(1,2,1);
     imshow(uint8(r1));
     subplot(1,2,2);
     imshow(uint8(r2));
```

RGB renkli görsele önce gürültü ekleyip sonrasında gürültüyü kaldırmak istersek;

```
I=imread("image.jpg");
J=imnoise(I,"salt & pepper",0.2);
```

% Kanallara ayrı ayrı filtre uygularız.

(29)
```
r=medfilt2(J(:,:,1),[3 3]);
g=medfilt2(J(:,:,2),[3 3]);
b=medfilt2(J(:,:,3),[3 3]);
```

Her (RGB olan üç boyutlu resme) kanala tek tek filtre uygularız.

% Sonrasında bu kanalları birleştiririz.

(30)
```
K=cat(3,r,g,b);
figure,
subplot(221); imshow(I);
subplot(222); imshow(J);
subplot(223); imshow(K);
```

fspecial METHODU

"filter2" ile filtreyi uygularız.

"fspecial" methodu ile filtreyi sadece oluşturuyuz. (11x11 filtre ortalama al)

- average → f=fspecial("average",11); img=filter2(f,img)
- disk → radius=2; f1=fspecial("disk",radius); img1=filter2(f1,img)
- gaussian → sigma=2; f2=fspecial("gaussian",3,sigma); img2=filter2(f2,img)
- laplacian → ikinci türevini alır ve geçiş, kenar değerlerini ortaya çıkarır
- log → kenar değerlerini ortaya çıkarır.
- motion
- prewitt
- sobel

EDGE DETECTION

Kenar bulmadır. Burada en yaygın olarak dört yöntem vardır. (Prewitt, Canny, Sobel, roberts)
Ortak özelliği ikinci türevi alarak kenar bulmaktır.
Aşağıdaki fonksiyon kullanılarak yapılır.

```
a=imread("image.jpg");
img1=edge(a,"prewitt");
img2=edge(a,"canny");   ← En çok tercih edilen
img3=edge(a,"sobel");
img4=edge(a,"roberts");
figure,imshow(a)
figure,---
---
```

Sobel'i örnek olarak alacak olursak threshold değerini bir manuel olarakta belirleyebiliriz veya kendi belirlediği değeri görebiliriz.

```
a=imread("image.jpg");
[g,t]=edge(a,"sobel");   ← g:görsel, t:threshold'u ifade eder
figure,imshow(a)
figure,imshow(g)
```

```
img1=edge(a,"sobel",0.1);   ← veya manuel olarakta threshold belirleyebiliriz
figure,imshow(img1)
```

GÖRÜNTÜLER ÜZERİNE LAPLACİAN FİLTRE UYGULAMA

En çok kullanılan laplacian filtreler aşağıdakilerdir.

0	-1	0
-1	4	-1
0	-1	0

Laplacian Operatörü (4'lü komşuluk)

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

Laplacian Operatörü (8'li komşuluk)

Aşağıdaki gibi kod kullanınız.

```
I0rJ = rgb2gray (img);
A = double(I0rJ);
filter = [-1,-1,-1;-1,8,-1;-1,-1,-1]
B=padarray (A, [1 1])
Output = zeros (size(A))
for i=1: size(B,1) -2
    for J =1:size (B,2)-2
        Temp = B(i:i+2, J:J+2).* filters
        Output (i,J) = sum (Temp(:));
    end
end
figure,
subplot (1,2,1);
imshow (uint8 (Output))
title("filtre Sonucu")
```

Resimdeki en basit mantıkla türev almak demek, iki tane piksel arasındaki farka bakmak demektir.

Bunlar gradient diyor.

$\Delta x = f(i+1, j) - f(i, j)$

$\Delta y = f(i, j+1) - f(i, j)$

ikinci pikselden ilk pikseli çıkartıyoruz.

Bu değerleri karmalamak lazım

$$M = \sqrt{\Delta x^2 + \Delta y^2}$$

EROSION İŞLEMİ (Aşındırma İşlevi)

Girdi görüntüsü ile yapı elemanının tamamının eşleşmesi gerekir ki "1" yazılsın. Bunun dışında "0" yazılır.
Görseli inceltir.
Bunu MATLAB üzerinde incelersek:

(38)

```
originalBW = imread("image.jpg");   % Girdi binary olmalı.
se = strel("line", 11, 90);
erodeBW = imerode(originalBW, se);
figure
imshow(erodeBW)
```

DILATION İŞLEMİ (Genişletme İşlevi)

Girdi görüntüsü ile yapı elemanının herhangi bir görüntüsü eşleşiyor veya çakışıyorsa "1" yazılır. Eşleşme veya çakışma yoksa sıfır yazılır
Görseli kalınlaştırır.
Bunu MATLAB üzerinde incelersek:

(39)

```
BW = imread("image.jpg");
se = strel("line", 11, 90);
BW2 = imdilate(BW, se);
imshow(BW2), title("Dilated")
```

EROSION ile Sınır Belirleme

Örneğin bir mut görseliniz ele alalım. Sarı renk için 225 değeri uygun gibi. Bunun için aşağıdaki MATLAB kodunu çalıştırınız ve mut sınırlarını belirleyiz.

(4P)

```
A = imread("mut.jpg");
C = im2gray(A);
C(C < 225) = 0;          ← Eşik değeri 225 altını sıfıra eşitle.
S = strel("disk", 4, 0);
D = ~im2bw(C)
F = imerode(D, S);

figure, imshow(A); title("Orijinal Image");
figure, imshow(D); title("Binary Image");
figure, imshow(D-F); title("Boundary extracted Image");
```