# PROJE KODLARI

|  |  |
| --- | --- |
| kartTanima.m | |
| clc, closeall, clearall |  |
| gR=imread('resim3.png'); | **Dışarıdan görüntü alma** |
| figure, imshow(gR), title('INPUT RESMİ'); | **Görüntüyü ekranda gösterme** |
| gR=imresize(gR,[342 NaN]);  figure, imshow(gR); | **Görüntüyü yeniden boyutlandırıldı ve ekranda gösterildi.** |
| g = imcrop(gR, [0 140 828 140]); | **Kart numaralarının olduğu kesim kesildi** |
|  |  |
| figure, imshow(g), title(**'KIRPILMIS HALI'**); | **Kırpılmış halin ekranda gösterilmesi** |
| f=g; |  |
|  |  |
| if size(f,3)==3  imagen=rgb2gray(f);  end | **Griye Dönüştürme** |
|  |  |
| %imagen = imadjust(imagen,[0 1],[1 0]); | **Karakterleri beyaz yapmak için** |
|  |  |
| imagen=medfilt2(imagen,[3 3]); | **Medyan filtreleme için gürültü temizleme** |
|  |  |
| figure, imshow(imagen), title(**'FİLTRELEMEDEN SONRA'**); | **Filtrelemeden sonraki görüntü** |
|  |  |
| [~, threshold] = edge(imagen, 'sobel');  fudgeFactor = .70;  KB = edge(imagen,'sobel', threshold \* fudgeFactor);  figure, imshow(KB), title(**'KENARLARI BULMAK İÇİN'**); | **Sobel Algoritması** |

|  |  |
| --- | --- |
| **MORFOLOJİK İSLEMLER** | |
| df=strel('disk',3); | **Morfolojik işlemler için disk filtreleme yarıçapı=3** |
| ra=imdilate(KB,df); | **Gri resmi strel işlemine göre açma** |
| ras=imerode(KB,df); | **Stele göre aşındırma işlemi** |
| gdiff=imsubtract(ra,ras); | **Kenarları iyileştirmek için gri tonları ayırma işlemi** |
| KBks = imclearborder(gdiff, 26); |  |
| figure, imshow(KBks), title(**'KENARLARI TEMİZLENMİS HALİ'**); |  |
| imagen = imadjust(KBks,[0 1],[1 0]); | **Negatifi** |
| figure, imshow(imagen); |  |
|  |  |
| threshold = graythresh(imagen); | **Tüm değeri baz alarak Matlab otomatik olarak threshold değeri atıyor** |
|  |  |
| imagen =~im2bw(imagen,threshold); | **İnput resmimizeuyguluyoruz.Fakat not equal operatörünü kullanıyoruz çünkü bizim templateslerimiz siyah üzerine beyaz karakterler** |
|  |  |
| imagen = bwareaopen(imagen,30); | **%30-connected neighborhood olmayanları resimden eliyoruz.Siyah sıfır yapıyoruz.** |
|  |  |
| word=[ ]; | **Text dosyasına yazağımız karakterleri word stringinde depolayacağız.** |
|  |  |
| re=imagen; | **Kolay yazmak için kısaltık.** |
| fid = fopen('text.txt', 'wt'); |  |
| loadtemplates | **Belirlenen templatesleri yükleme** |
| globaltemplates | **Templatesleri global yapma** |
| num\_letras=size(templates,2); | **36 karakterimiz var (Rakam,Harf)** |

|  |  |
| --- | --- |
| while 1 |  |
| [fl re]=lines(re); | **Lines fonksiyonu** |
| imgn=fl; |  |
| [L Ne] = bwlabel(imgn); | **Burada resmin içindeki componentleri (farklı karakterleri) tanımlıyoruz.’Ne’**  **Tane karakter olduğunu gösteriyor.** |
| for n=1:Ne |  |
| [r,c] = find(L==n); | **Mesela n=2 için bulduğu karakter rc matrixini oluştuyor.Rc matrisi onun yerini belirliyor(Location).** |
| n1=imgn(min(r):max(r),min(c):max(c)); | **N1 karakter için sol üstten sağ alta kadar sınırlıyoruz,o karakteri oluşturan yerleri belirtiyor.** |
|  |  |
| img\_r=imresize(n1,[42 24]); | **Templateslerimiz 42\*42 px oldukları için bulunan karakteri yeniden boyutlandırıyoruz** |
|  |  |
| letter=sayilari\_oku(img\_r,num\_letras); | **Sayilari\_oku() fonksiyonu çağrılıyor** |
| word=[wordletter]; | **Word matrixi içerisine bulunan karakter yazılıyor.Her seferinde bulunan karakter silinmemesi için bu formatta tanımlıyoruz.** |
| end |  |
| fprintf(fid,'%s\n',lower(word)); |  |
| word=[ ]; | **İkinci satır için matrisi boşaltıyoruz** |
| ifisempty(re) | **Re boşsa döngüden çık** |
| break |  |
| end |  |
| end |  |
| fclose(fid); | **Text dosyasını kapatıyoruz** |
| winopen('text.txt'); |  |

|  |  |
| --- | --- |
| **lines.m** | |
|  |  |
| function [fl re]=lines(im\_text) |  |
|  |  |
| im\_text=clip(im\_text); |  |
|  |  |
| num\_files=size(im\_text,1); |  |
|  |  |
| for s=1:num\_files |  |
| ifsum(im\_text(s,:))==0 |  |
| nm=im\_text(1:s-1, :); | İlk satır |
| rm=im\_text(s:end, :); | Geriye kalan satırlar |
| fl = clip(nm); |  |
| re=clip(rm); |  |
| break |  |
| else |  |
| fl=im\_text; | Tek satırdan oluşuyor |
| re=[ ]; | Geriye kalan satır yok |
| end |  |
| end |  |
|  |  |
| functionimg\_out=clip(img\_in) | Clip fonksiyonunu tanıtma |
| [f c]=find(img\_in); |
| img\_out=img\_in(min(f):max(f),min(c):max(c)); |
|  |

|  |  |
| --- | --- |
| sayilari\_oku.m | |
|  |  |
|  |  |
| functionletter=sayilari\_oku(imagn,num\_letras) |  |
| globaltemplates |  |
| comp=[ ]; | Compare etmek için boş bir matrix tanımlıyoruz. |
|  |  |
| for n=1:num\_letras | 1’den 36’ya kadar |
| sem=corr2(templates{1,n},imagn); | Herbir karakterlerle olan benzerliği bulunuyor |
|  |  |
| comp=[comp sem]; | Bulunan benzerlikleri comp matrixine tek satır oluşturacak şekilde yerleştiriyoruz. |
| end |  |
| vd=find(comp==max(comp)); | Comp matrixinddeki en yüksek oranda olanını,yani en çok benzeyenini alıyoruz. |
|  |  |
| ifvd==1 |  |
| letter='A'; |  |
| elseifvd==2 |  |
| letter='B'; |  |
| elseifvd==3 |  |
| letter='C'; |  |
| elseifvd==4 |  |
| letter='D'; |  |
| ..... |  |
| elseifvd==27 |  |
| letter='1'; |  |
| elseifvd==28 |  |
| letter='2'; |  |
| elseifvd==29 |  |
| letter='3'; |  |
| ..... |  |
| elseifvd==34 |  |
| letter='8'; |  |
| elseifvd==35 |  |
| letter='9'; |  |
| else |  |
| letter='0'; |  |
| end |  |