به نام خدا

سید محمد حسین مظهری 810101520 پروژه 2 درس سیگنال و سیستم

پارسا دقیق 810101419

بخش اول:

1) با استفاده از تابع uigetfile از کاربر تصویر مورد نظرش را درخواست می کنیم. سپس با استفاده از تابع imread عکس انتخاب شده را در قالب یک ماتریس لود می کنیم.

اسکریپت استفاده شده:

```
%1)extracting Original image
[file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
w=[path,file];
picture=imread(w);
figure
subplot(3,2,1)
imshow(picture)
title('Original Picture')
```

2) در این قسمت از تابع imresize استفاده کرده تا ابعاد تصویر را به 300x500 تغییر دهیم.

اسکریپت مورد استفاده:

```
%2)resizing the image for better correlation
picture=imresize(picture,[300 500]);
subplot(3,2,2)
imshow(picture)
title('Resized Picture')
```

3) برای استخراج پلاک و اعداد و حروف ، رنگ تصویر اهمیتی ندارد . پس برای پردازش راحت تر ، با استفاده از فرمول داده شده رنگ را از تصویر حذف کرده و آن را خاکستری می کنیم.

تابع این بخش را خودمان پیاده سازی میکنیم.

اسکریپت تابع mygrayfun :

```
%making the picture gray by the given formula
%this function recieves the colorfull picture and makes it gray
function graypic = mygrayfun (x)
    for i = 1:size(x,1)
        for j = 1:size(x,2)
            graypic(i,j) = (0.299*x(i,j,1)) + (0.578*x(i,j,2)) + (0.114*x(i,j,3)) ;
        end
    end
end
```

استفاده از تابع در کد اصلی:

```
%3)making the image gray
picture=mygrayfun(picture);
subplot(3,2,3)
imshow(picture)
title('Grayscale Picture')
```

4) با بررسی مقدار هر سلول و مقایسه ی آن با مقدار threshold تصویر را دودویی می کنیم.

اسکریپت تابع mybinaryfun :

```
%making the picture binary by using a threshold
%this function recieves a grayscale picture and returns the black and white
%picture
function bipic = mybinaryfun (pic,thresh)
    for i = 1:size(pic,1)
        for j = 1:size(pic,2)
            if pic(i,j) >= thresh
                bipic(i,j) = 1 ;
            else
                bipic(i,j) = 0 ;
            end
        end
    end
end
```

استفاده از تابع در کد اصلی:

```

%4)making the image binary(black & white)
threshold = 110 ;
%the number for threshold is extracted by trying different numbers
picture=mybinaryfun(picture,threshold);
subplot(3,2,4)
imshow(picture)
title('Binary Picture')
```

5) با گرفتن عکس باینری و آستانه تعداد پیکسل های متصل به هم ، تابعی را می نویسیم که بخش هایی که تعداد پیکسل های پیوسته آن از مقدار آستانه کمتر است را حذف کند.

ابتدا تصویر را نقیض می کنیم(تبدیل سیاه به سفید و سفید به سیاه).سپس همانند الگوریتمی که برای قسمت بعد یعنی mysegmentation نوشته ایم تصویر ورودی را بخش بندی می کنیم. اکنون برداری به طول تعداد قسمت ها ایجاد می کنیم و تعداد پیکسل های بر قسمت را پیدا کرده و در بردار مورد نظر ذخیره می کنیم.

سپس با دستور find قسمت هایی که مساحت آن ها از مقدار آستانه بیشتر است را استخراج می کنیم و در برداری دیگر ذخیره می کنیم. با دستور ismember این بردار را با ماتریس segment بندی شده مقایسه می کنیم. نواحی که لیبل آنها در بردار وجود دارند، عدد یک را اتخاذ می کنند و دیگر نواحی عدد صفر. در حقیقت خروجی این تابع یک تصویر دودویی است که شرط مساحت در نواحی آن اعمال شده است

اسکریپت تابع myremovecom :

```
%cleaning the picture by removing the extra parts of plate
%this function recieves a binary picture and a number (n) and returns the
%cleaned picture
function cleanpic = myremovecom (pic,thresh)
    pic = ~pic ;
    B=strel('square',3);
    A=pic;
    p=find(A==1);
    p=p(1);
    segpic=zeros([size(A,1) size(A,2)]);
    N=0;
    while(~isempty(p))
        N=N+1;
        p=p(1);
        X=false([size(A,1) size(A,2)]);
        X(p)=1;
        Y=A&imdilate(X,B);
        while(~isequal(X,Y))
            X=Y;
            Y=A&imdilate(X,B);
        end
        Pos=find(Y==1);
        A(Pos)=0;
        segpic(Pos)=N;
        p=find(A==1);
    end
    componentAreas = zeros(1,N);
    for label = 1:N
        componentAreas(label) = sum(segpic(:) == label);
    end
    validLabels = find(componentAreas >= thresh);
    cleanpic = ismember(segpic, validLabels);
end
```

استفاده از تابع در کد اصلی:

```
%5)cleaning the image by using connected pixels
number_of_pixels = 500 ;
picture=myremovecom(picture,number_of_pixels);
subplot(3,2,5)
imshow(picture)
title('Clean Picture')
```

6) اسکریپت تابع mysegmentation :

```
%segmenting the picture into letters and digits
function [segpic,N] = mysegmentation (pic)
    B=strel('square',3);
    A=pic;
    p=find(A==1);
    p=p(1);
    segpic=zeros([size(A,1) size(A,2)]);
    N=0;
    while(~isempty(p))
        N=N+1;
        p=p(1);
        X=false([size(A,1) size(A,2)]);
        X(p)=1;
        Y=A&imdilate(X,B);
        while(~isequal(X,Y))
            X=Y;
            Y=A&imdilate(X,B);
        end
        Pos=find(Y==1);
        A(Pos)=0;
        segpic(Pos)=N;
        p=find(A==1);
    end
end
```

استفاده از تابع در کد اصلی:

```

%6)divide image into segments(letters and digits)
[picture,Ne]=mysegmentation(picture);
subplot(3,2,6)
imshow(picture)
title('Segmented Picture')
```

در این قسمت قرار است تابعی بنویسیم که تصویر را segment بندی بکند و تعداد segmentها را نیز برگرداند. در ابتدا یک ماتریس با درایه های صفر به ابعاد تصویر ایجاد می کنیم . سپس با استفاده از دستور find آدرس سلول هایی که مقدار آنها برابر با یک است را در بردار p ذخیره می کنیم و مقدار

اولین سلول آن را باز میگردانیم.سپس به تعداد segment ها حلقه می زنیم.(چون تعداد آنها مشخص نیست از while استفاده می کنیم.) سپس یک ماتریس با ابعاد تصویر و مقادیر صفر ایجاد می کنیم. و مقدار سلول p ام آن را یک قرار می دهیم. یعنی اولین خانه ای که تصویر ورودی در آن یک است، در این ماتریس نیز یک است. حال از این سلول شروع کرده و کل ناحیه را می پیماییم.از تابع imdilate استفاده می کنیم.عملکرد این تابع به این صورت است که یک تصویر را بر حسب ناحیه ای که قبلا تعریف کرده ایم گسترش می دهد. ناحیه ی تعریف شده در این بخش مربعی به ضلع 3 پیکسل است.

حال حلقه ای دیگر ایجاد می کنیم. درون این حلقه از نقطه ای که قبلا پیدا کردیم ، شروع کرده و ناحیه را همانطور که توضیح داده شد ، گسترش می دهیم. پس از هر گسترش ناحیه گسترش یافته را با تصویر اصلی and می کنیم تا نقاطی که در تصویر اصلی صفر هستند در ناحیه ی گسترش یافته نیز صفر شوند و تنها نقاط موجود در تصویر اصلی گسترش پیدا کنند. این حلقه تا زمانی ادامه پیدا میکند که تمامی ناحیه تصویر اصلی را در بر گرفته باشیم.

سپس این نقاط را در ماتریسی که در ابتدا به عنوان تصویر segment بندی شده تعریف کرده بودیم ، با N جایگزین می کنیم. بعد این ناحیه را از تصویر اصلی حذف می کنیم و حلقه از ابتدا شروع به اجرا شدن می کند. این حلقه تا زمانی اجرا می شود که ناحیه ای در تصویر اصلی باقی نمانده باشد.هنگامی که تصویر اصلی خالی شد و تمامی نواحی segment بندی شدند، از حلقه خارج می شویم و تابع تصویر segment بندی شده و تعداد segmentها (N) را بر میگرداند.

7) آپلود مپ ست :

```
%loading the english digits and leters dataset
files=dir('Map Set English\');
len=length(files)-2;
TRAIN=cell(2,len);

for i=1:len
   TRAIN{1,i}=imread(['Map Set English','\',files(i+2).name]);
   TRAIN{2,i}=files(i+2).name(1);
end

save('TRAININGSETENG.mat','TRAIN');
```

استفاده از کد اصلی:

```
%7)loading the dataset for letters and digits
load TRAININGSETENG;
all_letters=size(TRAIN,2);

figure
final_result=[];
for n=1:Ne
    [r,c]=find(picture==n);
    Y=picture(min(r):max(r),min(c):max(c));
    imshow(Y)
    Y=imresize(Y,[42,24]);
    imshow(Y)
    pause(0.2)

    ro=zeros(1,all_letters);
    for k=1:all_letters
        ro(k)=corr2(TRAIN{1,k},Y);
    end
    %if (n == 7)
    %       disp(ro)
    %end
    [MAXRO,pos]=max(ro);
    if MAXRO>.47
        out=cell2mat(TRAIN(2,pos));
        final_result=[final_result out];
    end
end
```

برای decision making ، ابتدا باید mapset را در متلب لود کنیم. از دستور dir استفاده می کنیم که با گرفتن اسم فایل شامل mapset ، اطلاعات فایل های درون آن را در struct ذخیره می کند. حال ماتریسی با ابعاد تعداد فایل های mapset در 2 ایجاد می کنیم. در یک ردیف تصویر ها و در ردیف دیگر کاراکتر مربوط به هر تصویر را قرار می دهیم.

هم اکنون کاراکتر های پلاک را با تصویر های ماتریس به دست آمده از mapset مقایسه می کنیم تا کاراکتر ها را تشخیص دهیم.

تصویر از قبل لیبل گذاری شده است. یک حلقه به تعداد لیبل ها می زنیم و با استفاده از دستور find آدرس نقاطی از تصویر که مقداری برابر با لیبل مربوط به حلقه را دارند به دست می آوریم. حال این نقاط را جدا کرده و در ماتریس دیگری دخیره می کنیم. (کوچکترین و بزرگترین ردیف و ستون را با استفاده از دستور های min , max به دست آورده و سلول های بین این چهار مقدار را در ماتریسی دیگر ذخیره می کنیم.)

حال اندازه این تصویر را با دستور imresize تغییر می دهیم تا هم اندازه تصویر موجود در mapset شود . سپس به اندازه ی تعداد کاراکتر های موجود در mapset مقدار correlation ها را محاسبه

کرده و در یک بردار ذخیره می کنیم. پس از پایان کار حلقه ، بیشترین مقدار correlation و موقعیت آن (یعنی کاراکتر مورد نظر) را بر می گردانیم .(توجه شود که مقدار آستانه ای برای correlation در نظر گرفته شده است (0.47)).

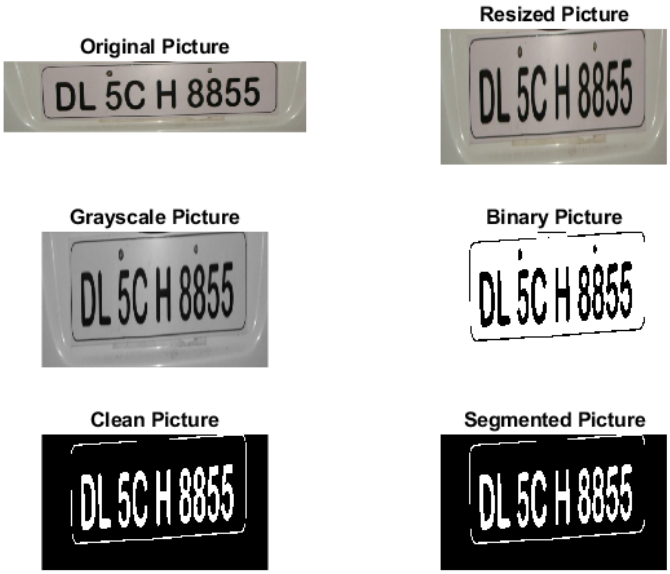
اگر matching به درستی رخ داد ، با استفاده از دستور celltmat و موقعیت بزرگترین عدد بردار، کاراکتر مربوطه را از TRAIN استخراج کرده و به یک ماتریس تبدیل می کنیم. و این ماتریس را به ماتریس نهایی اضافه می کنیم. با تمام شدن حلقه ، ماتریس نهایی نیز کامل می شود .

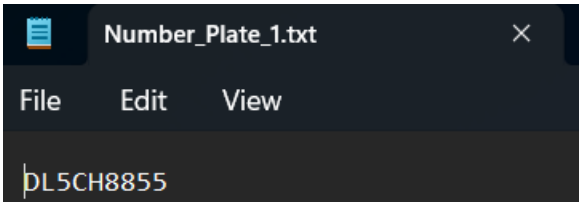
8) اسکریپت مربوط به چاپ پلاک:

```
71
72      %8)writing the plate number into a text file
73      disp(final_result);
74      file = fopen('Number_Plate_1.txt', 'wt');
75      fprintf(file,'%s\n',final_result);
76      fclose(file);
77      winopen('Number_Plate_1.txt')
```

برای چاپ پلاک، از دستور disp استفاده می کنیم. برای ذخیره کردن پلاک در فایل ، ابتدا با دستور fopen یک فایل txt. ایجاد میکنیم. سپس با دستور fprintf کاراکتر های ماتریس نهایی را به شکل استرینگ در فایل می نویسیم. در نهایت هم با دستور fclose به کار خاتمه می دهیم. بدین ترتیب هم پلاک در یک فایل txt. ذخیره می شود .

نتیجه ی کار برای یک پلاک ورودی :


Original Picture
DL 5C H 8855
Resized Picture
DL 5C H 8855
Grayscale Picture
DL 5C H 8855
Binary Picture
DL 5C H 8855
Clean Picture
DL 5C H 8855
Segmented Picture
DL 5C H 8855


Number_Plate_1.txt
File
Edit
View
DL5CH8855

اسکریپت p1.m :

```
clc
close all;
clear;

%1)extracting Original image
[file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
w=[path,file];
picture=imread(w);
figure
subplot(3,2,1)
imshow(picture)
title('Original Picture')

%2)resizing the image for better correlation
picture=imresize(picture,[300 500]);
subplot(3,2,2)
imshow(picture)
title('Resized Picture')

%3)making the image gray
picture=mygrayfun(picture);
subplot(3,2,3)
imshow(picture)
title('Grayscale Picture')

%4)making the image binary(black & white)
threshold = 110 ;
%the number for threshold is extracted by trying different numbers
picture=mybinaryfun(picture,threshold);
subplot(3,2,4)
imshow(picture)
title('Binary Picture')

%5)cleaning the image by using connected pixels
number_of_pixels = 500 ;
picture=myremovecom(picture,number_of_pixels);
subplot(3,2,5)
imshow(picture)
title('Clean Picture')

%6)divide image into segments(letters and digits)
[picture,Ne]=mysegmentation(picture);
subplot(3,2,6)
imshow(picture)
title('Segmented Picture')
```

```
%7)loading the dataset for letters and digits
load TRAININGSETENG;
all_letters=size(TRAIN,2);

figure
final_result=[];
for n=1:Ne
    [r,c]=find(picture==n);
    Y=picture(min(r):max(r),min(c):max(c));
    imshow(Y)
    Y=imresize(Y,[42,24]);
    imshow(Y)
    pause(0.2)

    ro=zeros(1,all_letters);
    for k=1:all_letters
        ro(k)=corr2(TRAIN{1,k},Y);
    end
    [MAXRO,pos]=max(ro);
    if MAXRO>.5
        out=cell2mat(TRAIN(2,pos));
        final_result=[final_result out];
    end
end

%8)writing the plate number into a text file
disp(final_result);
file = fopen('Number_Plate_1.txt', 'wt');
fprintf(file,'%s\n',final_result);
fclose(file);
winopen('Number_Plate_1.txt')
```

بخش دوم:

برای تهیه ی mapset از پلاک ها عکس گرفته و با سیاه سفید کردن عکس و جدا کردن کاراکتر ها به mapset فارسی می رسیم. سپس با استفاده از تابع imbinarize تصاویر را دودویی می کنیم. جهت انجام عملیات correlation گیری نیز اندازه ی تصاویر را به 42x24 تغییر می دهیم .

اسکریپت این بخش :

```
files=dir('Map Set Persian\');
len=length(files)-2;
TRAINPER=cell(2,len);

for i=1:len
    threshold = graythresh(double(imread(['Map Set Persian','\',files(i+2).name])));
    TRAINPER{1,i}=imbinarize(double(imresize(imread(['Map Set Persian','\',files(i+2).name]),[42,24])),threshold);
    TRAINPER{2,i}=files(i+2).name(1);
end

save('TRAININGSETPER.mat','TRAINPER');
```

Main :

گرفتن فایل از کاربر مانند بخش قبل است .

تغییر دادن اندازه تصاویر مانند بخش قبل است .

برای خاکستری کردن تصاویر از تابع rgb2gray استفاده شده است .

برای باینری کردن عکس از graythresh برای پیدا کردن آستانه استفاده می کنیم و سپس با دستور imbinarize تصویر را باینری می کنیم.

در قسمت کاهش نویز از تابع bwareaopen استفاده می کنیم و آستانه تعداد پیکسل ها را 500 در نظر می گیریم .

در قسمت segmentation از تابع bwlabel استفاده می کنیم .

Decision making مانند بخش قبل است .

چاپ و ذخیره پلاک مانند بخش قبل است .

اسکریپت p2.m :

```
clc
close all;
clear;

%1)extracting Original image
[file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
s=[path,file];
picture=imread(s);
figure
subplot(3,2,1)
imshow(picture)
title('Original Image')

%2)resizing the image for better correlation
picture=imresize(picture,[300 500]);
subplot(3,2,2)
imshow(picture)
title('Resized Image')


%3)making the image gray
picture=rgb2gray(picture);
subplot(3,2,3)
imshow(picture)
title('Grayscal Image')
```

```
%4)making the image binary(black & white)
threshold = graythresh(picture);
picture =~imbinarize(picture,threshold);
subplot(3,2,4)
imshow(picture)
title('Binary Image')

%5)cleaning the image by using connected pixels
number_of_pixels = 500 ;
picture = bwareaopen(picture,number_of_pixels);
background=bwareaopen(picture,5000);
picture=picture-background;
subplot(3,2,5)
imshow(picture)
title('Clean Image')

%6)divide image into segments(letters and digits)
[picture,Ne]=bwlabel(picture);
subplot(3,2,6)
imshow(picture)
title('Segmentated Image')

```

```
%7)loading the dataset for letters and digits
load TRAININGSETPER;
totalLetters=size(TRAINPER,2);

figure
final_output=[];
for n=1:Ne
    [r,c]=find(picture==n);
    Y=picture(min(r):max(r),min(c):max(c));
    imshow(Y)
    Y=imresize(Y,[42,24]);
    imshow(Y)
    pause(0.2)

    ro=zeros(1,totalLetters);
    for k=1:totalLetters
        ro(k)=corr2(TRAINPER{1,k},Y);
    end
    [MAXRO,pos]=max(ro);
    if MAXRO>.5
        out=cell2mat(TRAINPER(2,pos));
        final_output=[final_output out];
    end
end

```

```
%8)writing the plate number into a text file
disp(final_output);
file = fopen('Number_Plate_2.txt', 'wt');
fprintf(file,'%s\n',final_output);
fclose(file);
winopen('Number_Plate_2.txt')
```

نتیجه ی کار برای پلاک دلخواه:

Original Image

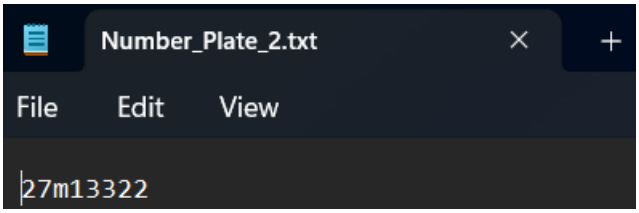
Resized Image

Grayscal Image

Binary Image

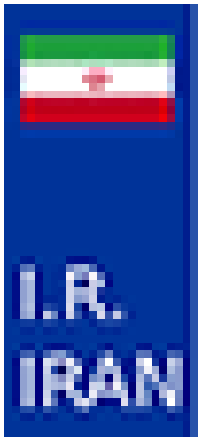
Clean Image

Segmentated Image



بخش سوم:

ایده ی جدا سازی قاب پلاک از تصویر در این بخش ، تشخیص دادن نوار آبی کنار پلاک می باشد. همانطور که میدانیم ، تمام پلاک های مورد بررسی ما ، یک نوار آبی رنگ به شکل زیر دارند:



ایده ی اصلی این پخش هم تشخیص دادن این نوار و بواسطه ی آن تشخیص دادن قاب پلاک است.
برای این کار تابعی به نام mycrop ایجاد می کنیم . در این تابع ابتدا عکس این نوار آبی را آپلود می کنیم. ابعاد تصویر را تنظیم کرده و نسبت تصویر را هم با توجه به تغییر ابعاد محاسبه می کنیم. (تنظیم ابعاد تصویر برای انجام عمل correlation می باشد.) حال شروع به correlation گرفتن می کنیم .
از آنجا که هر دو تصویر rgb هستند ، correlation هر لایه را جداگانه محاسبه می کنیم و در نهایت از آن سه میانگین می گیریک تا correlation نهایی به دست آید. حال با استفاده از حاصل نهایی ، محلی که مقدار آن بیشینه می شود را پیدا میکنیم و جهت اطمینان از اندکی قبل تر از آن به عنوان شروع پلاک یاد می کنیم.
حال با توجه به نسبت تصاویر، یک ماتریس ایجاد می کنیم که حاوی مختصات دو گوشه نوار آبی باشد . می دانیم ارتفاع نوار آبی با ارتفاع قاب پلاک برابر است و فقط طول این دو متفاوت است .طول قاب پلاک تقریبا 14 برابر طول نوار آبی است . بنابر این مقدار سلولی از ماتریس که نشان دهنده x گوشه دیگر قاب است را 14 برابر کرده و با نسبت تصویر نیز تطابق می دهیم .حال که مختصات گوشه های قاب پلاک را داریم، با استفاده از دستور imcrop به راحتی قاب پلاک را از باقی تصویر جدا کره و با فرمت jpg ذخیره میکنیم تا در کد اصلی از آن استفاده بکنیم.

```
function croppic = mycrop(full_picture)
    bluestrip=imread("bluestrip.png");

    pictureresized=imresize(full_picture,[NaN,800]);
    ratio=size(full_picture,1)/size(pictureresized,1);

    corr1=normxcorr2(bluestrip(:,:,1),pictureresized(:,:,1));
    corr2=normxcorr2(bluestrip(:,:,2),pictureresized(:,:,2));
    corr3=normxcorr2(bluestrip(:,:,3),pictureresized(:,:,3));
    corr=(corr1+corr2+corr3)/3;
    [corr_max,corrIdx]=max(abs(corr(:)));
    [peakY,peakX]=ind2sub(size(corr),corrIdx(1));
    corr_offset=[peakX-size(bluestrip,2),peakY-size(bluestrip,1)];
    bbox = [corr_offset(1),corr_offset(2),size(bluestrip,2),size(bluestrip,1)];
    bbox_full=[round((bbox(1)-10)*ratio),round((bbox(2)-10)*ratio),round((bbox(3)+2*10)*ratio),round((bbox(4)+2*10
    bounding_box=bbox_full;
    bounding_box(3)=14*bbox(3)*ratio;

    croppic = imcrop(full_picture, bounding_box);
    imwrite(croppic, 'cropped_pic.jpg');
end
```

اسکریپت تابع mycrop :

```
function croppic = mycrop(full_picture)
    bluestrip=imread("bluestrip.png");

    pictureresized=imresize(full_picture,[NaN,800]);
    ratio=size(full_picture,1)/size(pictureresized,1);

    corr1=normxcorr2(bluestrip(:,:,1),pictureresized(:,:,1));
    corr2=normxcorr2(bluestrip(:,:,2),pictureresized(:,:,2));
    corr3=normxcorr2(bluestrip(:,:,3),pictureresized(:,:,3));
    corr=(corr1+corr2+corr3)/3;
    [corr_max,corrIdx]=max(abs(corr(:)));
    [peakY,peakX]=ind2sub(size(corr),corrIdx(1));
    corr_offset=[peakX-size(bluestrip,2),peakY-size(bluestrip,1)];
    bbox = [corr_offset(1),corr_offset(2),size(bluestrip,2),size(bluestrip,1)];
    bbox_full=[round((bbox(1)-10)*ratio),round((bbox(2)-10)*ratio),round((bbox(3)+2*10)*ratio),round((bbox(4)+2*10)*ratio)];
    bounding_box=bbox_full;
    bounding_box(3)=14*bbox(3)*ratio;

    croppic = imcrop(full_picture, bounding_box);
    imwrite(croppic, 'cropped_pic.jpg');
end
```

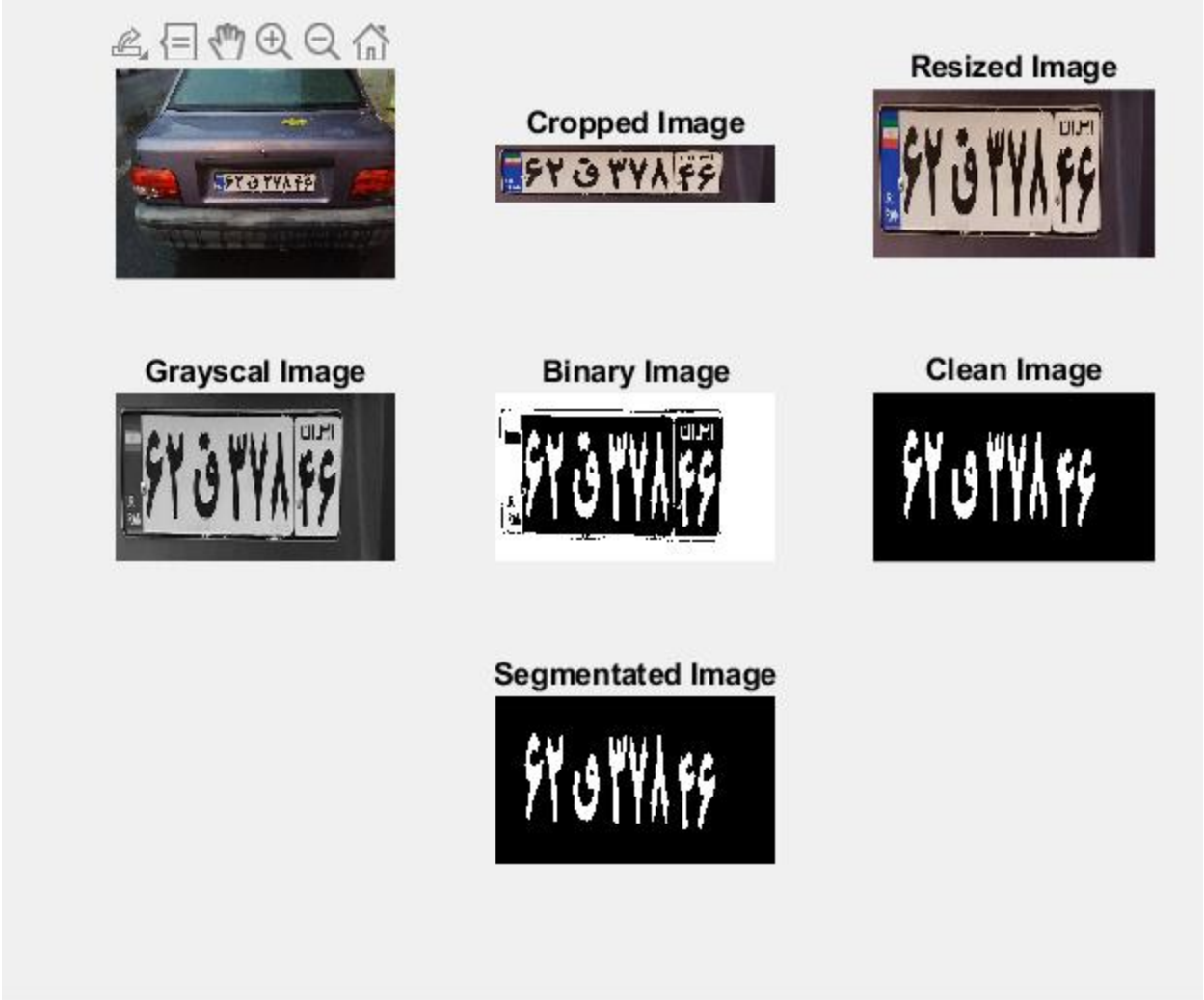
باقی کد مانند بخش است با این تفاوت که پس از آپلود تصویر توسط کاربر ، تابع mycrop صدا زده می شود تا قاب پلاک را جدا و ذخیره کند .

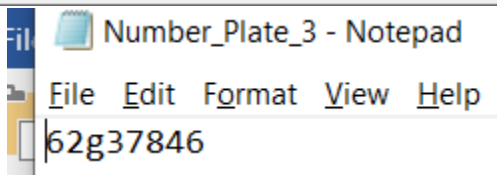
اسکریپت p3.m :

```
clc
close all;
clear;

%1
[file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
s=[path,file];
picture=imread(s);
figure
subplot(3,3,1)
imshow(picture)
title('Original Image')

%2
mycrop(picture);
picture=imread("cropped_pic.jpg");
subplot(3,3,2)
imshow(picture)
title('Cropped Image')

%3
picture=imresize(picture,[300 500]);
subplot(3,3,3)
imshow(picture)
title('Resized Image')


%4
picture=rgb2gray(picture);
subplot(3,3,4)
imshow(picture)
title('Grayscal Image')
%5
threshold = graythresh(picture);
picture =~imbinarize(picture,threshold);
subplot(3,3,5)
imshow(picture)
title('Binary Image')

%6
number_of_pixels=700;
picture = bwareaopen(picture,number_of_pixels);
background=bwareaopen(picture,5500);
picture=picture-background;
subplot(3,3,6)
imshow(picture)
title('Clean Image')

%7
[picture,Ne]=bwlabel(picture);
subplot(3,3,8)
imshow(picture)
title('Segmentated Image')

```

```
%8
load TRAININGSETPER;
totalLetters=size(TRAINPER,2);

figure
final_output=[];
for n=1:Ne
    [r,c]=find(picture==n);
    Y=picture(min(r):max(r),min(c):max(c));
    imshow(Y)
    Y=imresize(Y,[42,24]);
    imshow(Y)
    pause(0.2)

    ro=zeros(1,totalLetters);
    for k=1:totalLetters
        ro(k)=corr2(TRAINPER{1,k},Y);
    end
    [MAXRO,pos]=max(ro);
    if MAXRO>.5
        out=cell2mat(TRAINPER(2,pos));
        final_output=[final_output out];
    end
end

%9
disp(final_output);
file = fopen('Number_Plate_3.txt', 'wt');
fprintf(file,'%s\n',final_output);
fclose(file);
winopen('Number_Plate_3.txt')
```

عملکرد اسکریپت روی یک ورودی:





بخش چهارم:

در دو لحظه t=1s و t=1.5s دو فریم را بررسی می‌کنیم.

فریم‌ها را پردازش می‌کند تا ویژگی‌ها شناسایی شوند، ویژگی‌ها را بین دو فریم مطابقت می‌دهد، جابجایی نقاط مطابقت داده شده را محاسبه می‌کند و سپس سرعت را بر اساس فاصله زمانی بین فریم‌ها محاسبه می‌کند-.نتیج جه در واحد پیکسل بر ثانیه نمایش داده می‌شود.

```
vidObj = VideoReader('car_plate.mp4');
timestamp = 1; % Accessing the frame at 1 seconds
frame_number = round(vidObj.FrameRate * timestamp);
vidObj.CurrentTime = (frame_number - 1) / vidObj.FrameRate;
frame1 = readFrame(vidObj);
imshow(frame1);
imwrite(frame1, 'plate_frame.png');

timestamp = 1.5; % Accessing the frame at 1.5 seconds
frame_number = round(vidObj.FrameRate * timestamp);
vidObj.CurrentTime = (frame_number - 1) / vidObj.FrameRate;
frame2 = readFrame(vidObj);
imshow(frame2);

gray1 = rgb2gray(frame1);
gray2 = rgb2gray(frame2);
points1 = detectSURFFeatures(gray1);
points2 = detectSURFFeatures(gray2);
[features1, validPoints1] = extractFeatures(gray1, points1);
[features2, validPoints2] = extractFeatures(gray2, points2);
indexPairs = matchFeatures(features1, features2);
matchedPoints1 = validPoints1(indexPairs(:, 1), :);
matchedPoints2 = validPoints2(indexPairs(:, 2), :);
displacements = matchedPoints2.Location - matchedPoints1.Location;
averageDisplacement = mean(sqrt(sum(displacements.^2, 2)));
timeInterval = 0.5;
velocity = averageDisplacement / timeInterval;
disp(['Velocity: ', num2str(velocity), ' pixels per second']);
```

که با استفاده از کد قسمت سوم هم پلاک را از فریم اول t=1s تشخیص می دهیم.

با استفاده از محاسبات بالا میبینیم که نتایج زیر بدست می آید:

```
>> p4
Velocity: 183.2587 pixels per second
47j58822
>>
```