

به نام خدا

پروژه 2 درس سیگنال و سیستم

810101520

سید محمد حسین مظهري

810101419

پارسا دقيق

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.
اسکرپت استفاده شده:

```
5 %1)extracting Original image
6 [file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'}, 'Choose an image');
7 w=[path,file];
8 picture=imread(w);
9 figure
10 subplot(3,2,1)
11 imshow(picture)
12 title('Original Picture')
```

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

```
14 %2)resizing the image for better correlation
15 picture=imresize(picture,[300 500]);
16 subplot(3,2,2)
17 imshow(picture)
18 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.

اسکرپت تابع `mygrayfun` :

```

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

```

استفاده از تابع در کد اصلی:

```

20 %3)making the image gray
21 picture=mygrayfun(picture);
22 subplot(3,2,3)
23 imshow(picture)
24 title('Grayscale Picture')

```

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

اسکرپت تابع mybinaryfun :

```

1 %making the picture binary by using a threshold
2 %this function recieves a grayscale picture and returns the black and white
3 %picture
4 function bipic = mybinaryfun (pic,thresh)
5     for i = 1:size(pic,1)
6         for j = 1:size(pic,2)
7             if pic(i,j) >= thresh
8                 bipic(i,j) = 1 ;
9             else
10                bipic(i,j) = 0 ;
11            end
12        end
13    end
14 end

```

استفاده از تابع در کد اصلی:

```

25
26 %4)making the image binary(black & white)
27 threshold = 110 ;
28 %the number for threshold is extracted by trying different numbers
29 picture=mybinaryfun(picture,threshold);
30 subplot(3,2,4)
31 imshow(picture)
32 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 هر 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

اسکرپت تابع myremovecom :

```
1 %cleaning the picture by removing the extra parts of plate
2 %this function recieves a binary picture and a number (n) and returns the
3 %cleaned picture
4 function cleanpic = myremovecom (pic,thresh)
5     pic = ~pic ;
6     B=strel('square',3);
7     A=pic;
8     p=find(A==1);
9     p=p(1);
10    segpic=zeros([size(A,1) size(A,2)]);
11    N=0;
12    while(~isempty(p))
13        N=N+1;
14        p=p(1);
15        X=false([size(A,1) size(A,2)]);
16        X(p)=1;
17        Y=A&imdilate(X,B);
18        while(~isequal(X,Y))
19            X=Y;
20            Y=A&imdilate(X,B);
21        end
22        Pos=find(Y==1);
23        A(Pos)=0;
24        segpic(Pos)=N;
25        p=find(A==1);
26    end
27    componentAreas = zeros(1,N);
28    for label = 1:N
29        componentAreas(label) = sum(segpic(:) == label);
30    end
31    validLabels = find(componentAreas >= thresh);
32    cleanpic = ismember(segpic, validLabels);
33 end
```

استفاده از تابع در کد اصلی:

```
34 %5)cleaning the image by using connected pixels
35 number_of_pixels = 500 ;
36 picture=myremovecom(picture,number_of_pixels);
37 subplot(3,2,5)
38 imshow(picture)
39 title('Clean Picture')
```

(6) اسکریپت تابع mysegmentation :

```
1 %segmenting the picture into letters and digits
2 function [segpic,N] = mysegmentation (pic)
3     B=strel('square',3);
4     A=pic;
5     p=find(A==1);
6     p=p(1);
7     segpic=zeros([size(A,1) size(A,2)]);
8     N=0;
9     while(~isempty(p))
10         N=N+1;
11         p=p(1);
12         X=false([size(A,1) size(A,2)]);
13         X(p)=1;
14         Y=A&imdilate(X,B);
15         while(~isequal(X,Y))
16             X=Y;
17             Y=A&imdilate(X,B);
18         end
19         Pos=find(Y==1);
20         A(Pos)=0;
21         segpic(Pos)=N;
22         p=find(A==1);
23     end
24 end
```

استفاده از تابع در کد اصلی:

```
40
41 %6)divide image into segments(letters and digits)
42 [picture,Ne]=mysegmentation(picture);
43 subplot(3,2,6)
44 imshow(picture)
45 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 قبلاً 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 قبلاً 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 segment ها (N) را بر میگردانند.

(7) آپلود مپ ست :

```

1 %loading the english digits and letters dataset
2 files=dir('Map Set English\');
3 len=length(files)-2;
4 TRAIN=cell(2,len);
5
6 for i=1:len
7     TRAIN{1,i}=imread(['Map Set English','\ ',files(i+2).name]);
8     TRAIN{2,i}=files(i+2).name(1);
9 end
10
11 save('TRAININGSETENG.mat','TRAIN');|

```

استفاده از کد اصلی:

```

47 %7)loading the dataset for letters and digits
48 load TRAININGSETENG;
49 all_letters=size(TRAIN,2);
50
51 figure
52 final_result=[];
53 for n=1:Ne
54     [r,c]=find(picture==n);
55     Y=picture(min(r):max(r),min(c):max(c));
56     imshow(Y)
57     Y=imresize(Y,[42,24]);
58     imshow(Y)
59     pause(0.2)
60
61     ro=zeros(1,all_letters);
62     for k=1:all_letters
63         ro(k)=corr2(TRAIN{1,k},Y);
64     end
65     %if (n == 7)
66     %     disp(ro)
67     %end
68     [MAXRO,pos]=max(ro);
69     if MAXRO>.47
70         out=cell2mat(TRAIN(2,pos));
71         final_result=[final_result out];
72     end
73 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 ذخیره 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)).

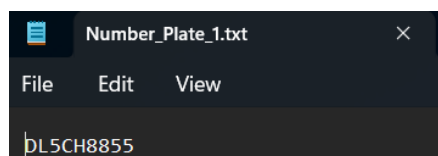
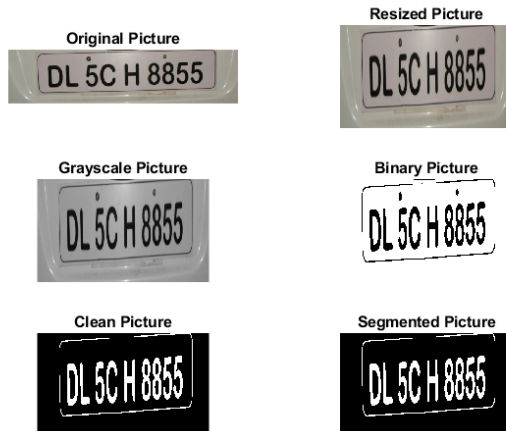
اگر **matching** به درستی رخ داد ، با استفاده از دستور **cellmat** و موقعیت بزرگترین عدد بردار، کاراکتر مربوطه را از **TRAIN** استخراج کرده و به یک ماتریس تبدیل می کنیم. و این ماتریس را به ماتریس نهایی اضافه می کنیم. با تمام شدن حلقه ، ماتریس نهایی نیز کامل می شود .

(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** کاراکترهای 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 :



اسکرپت p1.m :


```

1  clc
2  close all;
3  clear;
4
5  %1)extracting Original image
6  [file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'}, 'Choose an image');
7  w=[path,file];
8  picture=imread(w);
9  figure
10 subplot(3,2,1)
11 imshow(picture)
12 title('Original Picture')
13
14 %2)resizing the image for better correlation
15 picture=imresize(picture,[300 500]);
16 subplot(3,2,2)
17 imshow(picture)
18 title('Resized Picture')
19
20 %3)making the image gray
21 picture=mygrayfun(picture);
22 subplot(3,2,3)
23 imshow(picture)
24 title('Grayscale Picture')
25
26 %4)making the image binary(black & white)
27 threshold = 110 ;
28 %the number for threshold is extracted by trying different numbers
29 picture=mybinaryfun(picture,threshold);
30 subplot(3,2,4)
31 imshow(picture)
32 title('Binary Picture')
33

```

```

34 %5)cleaning the image by using connected pixels
35 number_of_pixels = 500 ;
36 picture=myremovecom(picture,number_of_pixels);
37 subplot(3,2,5)
38 imshow(picture)
39 title('Clean Picture')
40
41 %6)divide image into segments(letters and digits)
42 [picture,Ne]=mysegmentation(picture);
43 subplot(3,2,6)
44 imshow(picture)
45 title('Segmented Picture')
46

```

```

47 %7)loading the dataset for letters and digits
48 load TRAININGSETENG;
49 all_letters=size(TRAIN,2);
50
51 figure
52 final_result=[];
53 for n=1:Ne
54     [r,c]=find(picture==n);
55     Y=picture(min(r):max(r),min(c):max(c));
56     imshow(Y)
57     Y=imresize(Y,[42,24]);
58     imshow(Y)
59     pause(0.2)
60
61     ro=zeros(1,all_letters);
62     for k=1:all_letters
63         ro(k)=corr2(TRAIN{1,k},Y);
64     end
65     [MAXRO,pos]=max(ro);
66     if MAXRO>.5
67         out=cell2mat(TRAIN(2,pos));
68         final_result=[final_result out];
69     end
70 end
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')

```

بخش دوم:

برای تهیه ی mapset از پلاک ها عکس گرفته و با سیاه سفید کردن عکس و جدا کردن کاراکترها 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 .

اسکرپت این بخش :

```

1 files=dir('Map Set Persian\');
2 len=length(files)-2;
3 TRAINPER=cell(2,len);
4
5 for i=1:len
6     threshold = graythresh(double(imread(['Map Set Persian','\',files(i+2).name])));
7     TRAINPER{1,i}=imbinarize(double(imresize(imread(['Map Set Persian','\',files(i+2).name]),[42,24])),threshold);
8     TRAINPER{2,i}=files(i+2).name(1);
9 end
10
11 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 .

اسکرپٹ p2.m :

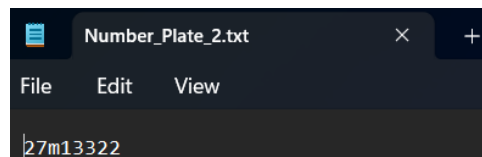
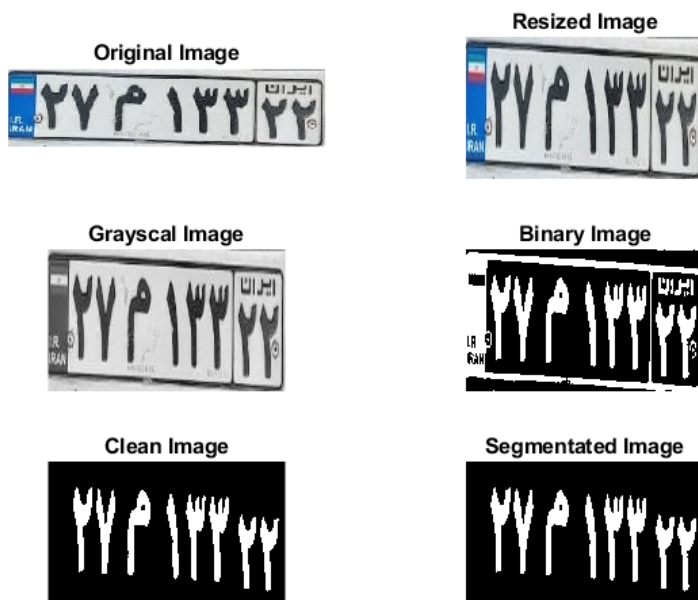
```
1  clc
2  close all;
3  clear;
4
5  %1)extracting Original image
6  [file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'}, 'Choose an image');
7  s=[path,file];
8  picture=imread(s);
9  figure
10 subplot(3,2,1)
11 imshow(picture)
12 title('Original Image')
13
14 %2)resizing the image for better correlation
15 picture=imresize(picture,[300 500]);
16 subplot(3,2,2)
17 imshow(picture)
18 title('Resized Image')
19
20
21 %3)making the image gray
22 picture=rgb2gray(picture);
23 subplot(3,2,3)
24 imshow(picture)
25 title('Grayscale Image')
26
27 %4)making the image binary(black & white)
28 threshold = graythresh(picture);
29 picture = ~imbinarize(picture,threshold);
30 subplot(3,2,4)
31 imshow(picture)
32 title('Binary Image')
33
34 %5)cleaning the image by using connected pixels
35 number_of_pixels = 500 ;
36 picture = bwareaopen(picture,number_of_pixels);
37 background=bwareaopen(picture,5000);
38 picture=picture-background;
39 subplot(3,2,5)
40 imshow(picture)
41 title('Clean Image')
42
43 %6)divide image into segments(letters and digits)
44 [picture,Ne]=bwlabel(picture);
45 subplot(3,2,6)
46 imshow(picture)
47 title('Segmentated Image')
48
```

```

49 %7)loading the dataset for letters and digits
50 load TRAININGSETPER;
51 totalLetters=size(TRAINPER,2);
52
53 figure
54 final_output=[];
55 for n=1:Ne
56     [r,c]=find(picture==n);
57     Y=picture(min(r):max(r),min(c):max(c));
58     imshow(Y)
59     Y=imresize(Y,[42,24]);
60     imshow(Y)
61     pause(0.2)
62
63     ro=zeros(1,totalLetters);
64     for k=1:totalLetters
65         ro(k)=corr2(TRAINPER{1,k},Y);
66     end
67     [MAXRO,pos]=max(ro);
68     if MAXRO>.5
69         out=cell2mat(TRAINPER{2,pos});
70         final_output=[final_output out];
71     end
72 end
73
74 %8)writing the plate number into a text file|
75 disp(final_output);
76 file = fopen('Number_Plate_2.txt', 'wt');
77 fprintf(file,'%s\n',final_output);
78 fclose(file);
79 winopen('Number_Plate_2.txt')

```

نتیجه ی کار برای پلاک دلخواه:



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:

Formatted: Font color: Dark Blue



ایده ی اصلی این یخش 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 گیریم 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 تقریباً 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.

Formatted: Font color: Dark Blue

Formatted: Font color: Dark Blue

```
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
```

Formatted: Font color: Dark Blue

اسکرپت تابع mycrop :

Formatted: Font color: Dark Blue

```
1 function croppic = mycrop(full_picture)
2     bluestrip=imread("bluestrip.png");
3
4     pictureresized=imresize(full_picture,[NaN,800]);
5     ratio=size(full_picture,1)/size(pictureresized,1);
6
7     corr1=normxcorr2(bluestrip(:,1),pictureresized(:,1));
8     corr2=normxcorr2(bluestrip(:,2),pictureresized(:,2));
9     corr3=normxcorr2(bluestrip(:,3),pictureresized(:,3));
10    corr=(corr1+corr2+corr3)/3;
11    [corr_max,corrIdx]=max(abs(corr(:)));
12    [peakY,peakX]=ind2sub(size(corr),corrIdx(1));
13    corr_offset=[peakX-size(bluestrip,2),peakY-size(bluestrip,1)];
14    bbox = [corr_offset(1),corr_offset(2),size(bluestrip,2),size(bluestrip,1)];
15    bbox_full=[round((bbox(1)-10)*ratio),round((bbox(2)-10)*ratio),round((bbox(3)+2*10)*ratio),round((bbox(4)+2*10)*ratio)];
16    bounding_box=bbox_full;
17    bounding_box(3)=14*bbox(3)*ratio;
18
19    croppic = imcrop(full_picture, bounding_box);
20    imwrite(croppic, 'cropped_pic.jpg');
21 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 .

Formatted: Font color: Dark Blue

اسکرپت p3.m :


```

1      clc
2      close all;
3      clear;
4
5      %1
6      [file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
7      s=[path,file];
8      picture=imread(s);
9      figure
10     subplot(3,3,1)
11     imshow(picture)
12     title('Original Image')
13
14     %2
15     mycrop(picture);
16     picture=imread("cropped_pic.jpg");
17     subplot(3,3,2)
18     imshow(picture)
19     title('Cropped Image')
20
21     %3
22     picture=imresize(picture,[300 500]);
23     subplot(3,3,3)
24     imshow(picture)
25     title('Resized Image')
26
27
28     %4
29     picture=rgb2gray(picture);
30     subplot(3,3,4)
31     imshow(picture)
32     title('Grayscale Image')
33
34     %5
35     threshold = graythresh(picture);
36     picture =~imbinarize(picture,threshold);
37     subplot(3,3,5)
38     imshow(picture)
39     title('Binary Image')
40
41     %6
42     number_of_pixels=700;
43     picture = bwareaopen(picture,number_of_pixels);
44     background=bwareaopen(picture,5500);
45     picture=picture-background;
46     subplot(3,3,6)
47     imshow(picture)
48     title('Clean Image')
49
50     %7
51     [picture,Ne]=bwlabel(picture);
52     subplot(3,3,8)
53     imshow(picture)
54     title('Segmentated Image')
55

```

```

56 %8
57 load TRAININGSETPER;
58 totalLetters=size(TRAINPER,2);
59
60 figure
61 final_output=[];
62 for n=1:Ne
63     [r,c]=find(picture==n);
64     Y=picture(min(r):max(r),min(c):max(c));
65     imshow(Y)
66     Y=imresize(Y,[42,24]);
67     imshow(Y)
68     pause(0.2)
69
70     ro=zeros(1,totalLetters);
71     for k=1:totalLetters
72         ro(k)=corr2(TRAINPER{1,k},Y);
73     end
74     [MAXRO,pos]=max(ro);
75     if MAXRO>.5
76         out=cell2mat(TRAINPER{2,pos});
77         final_output=[final_output out];
78     end
79 end
80
81 %9
82 disp(final_output);
83 file = fopen('Number_Plate_3.txt', 'wt');
84 fprintf(file,'%s\n',final_output);
85 fclose(file);
86 winopen('Number_Plate_3.txt')

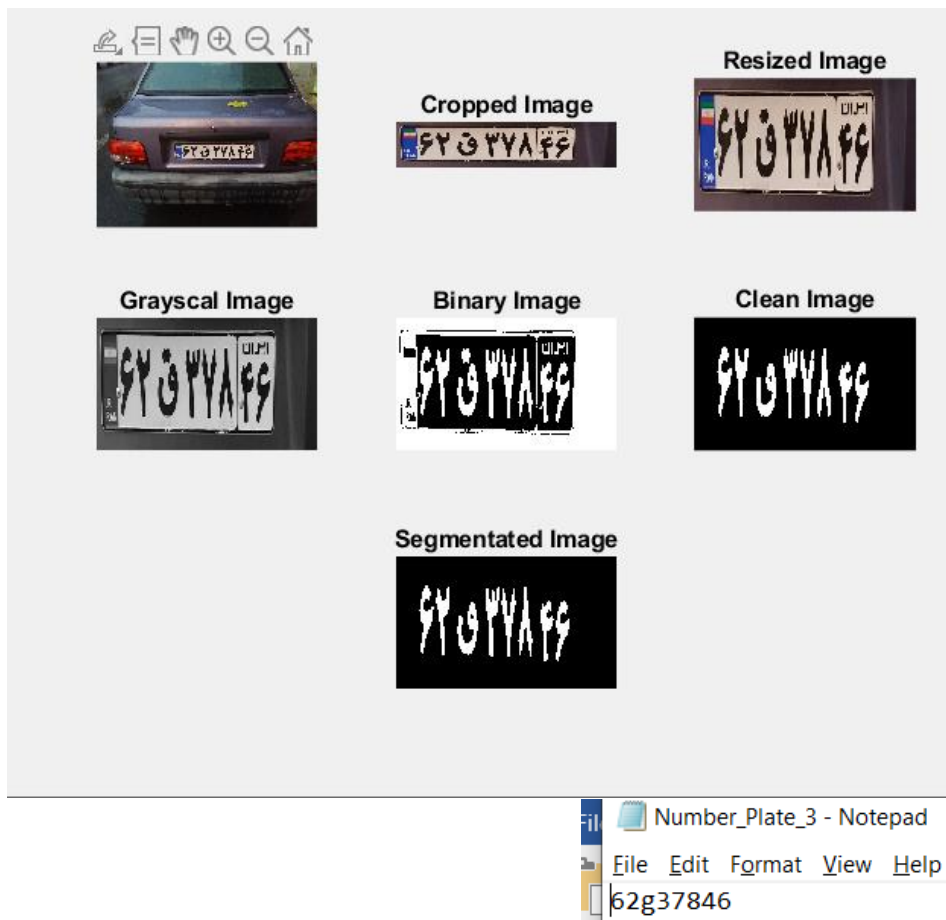
```

Formatted: Font color: Dark Blue

عملکرد اسکریپت روی یک ورودی:

Formatted: Font color: Dark Blue, Complex Script Font: Calibri

Formatted: Right



بخش چهارم:

در دو لحظه $t=1s$ و $t=1.5s$ دو فریم را بررسی می کنیم.

فریم ها را پردازش می کند تا ویژگی ها شناسایی شوند، ویژگی ها را بین دو فریم تطابقت می دهد، جابجایی نقاط تطابقت داده شده را محاسبه می کند و سپس سرعت را بر اساس فاصله زمانی بین فریم ها محاسبه می کند. نتیجه چه در واحد پیکسل بر ثانیه نمایش داده می شود.

Formatted

```

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
>>

```

Formatted: Font color: Dark Blue