به نام خدا

سیگنال ها و سیستم

پروژه کامپیوتری 2

مهراد لیویان 810101501

مرضیه موسوی 810101526

بخش اول

ابتدا mapset انگلیسی را لود می کنیم:

```
clc;
clear;
close all;

di=dir('Map Set');
st={di.name};
nam=st(3:end);
len=length(nam);

TRAIN=cell(2,len);
for i=1:len
   addres = append("Map Set", "/", cell2mat(nam(i)));
   TRAIN(1,i)={imread(addres)};
   temp=cell2mat(nam(i));
   TRAIN(2,i)={temp(1)};
end

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

در یک بخش از train عکس مپ ست و در بخش دیگر نام عضو مپ ست را ذخیره می کنیم.

حال در p1.1 عکس را از کاربر گرفته و آن را لود می کنیم و سپس resize می کنیم.

```
[file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
s=[path,file];

picture=imread(s);
figure

picture=imresize(picture,[300 500]);
```

مشابه تابع imbinarize تابع mybinaryfun را می نویسیم:

```
function binary_image = mybinaryfun(input_image , threshold)

binary_image=ones(size(input_image));
binary_image(input_image>threshold)=0;
end
```

اگر عدد از threshold بالاتر بود یک و در غیر این صورت 0 بگذار.

تابع mygrayfun را برای برعکس کردن رنگ سیاه و سفید عکس می گذاریم:

```
function output_image =  mygrayfun(input_picture)

if size(input_picture,3)==3
    output_image= 0.2989*input_picture(:,:,1) + 0.5870 * input_picture(:,:,2) +0.1140 * input_picture(:,:,3);
else
    output_image=input_picture;

end
output_image = uint8(output_image);
end
```

برای تابع myremovecom مانند الگوریتم bfs عمل می کنیم:

```
function output_image = myremovecom(input_image, threshold_area)
   % Function to remove small objects from a binary image
   % Input:
   % input_image: Binary image
   % threshold_area: Minimum area of objects to be retained

   [rows, columns] = size(input_image);
   labeled_image = zeros(rows, columns);
   label = 0;

   for i = 1:rows
       for j = 1:columns
           if input_image(i, j) == 1 && labeled_image(i, j) == 0
               label = label + 1;
               stack = [i, j];
               area = 0;

               while ~isempty(stack)
                   current_pixel = stack(1, :);
                   stack(1, :) = [];
                   x = current_pixel(1);
                   y = current_pixel(2);

                   if x >= 1 && x <= rows && y >= 1 && y <= columns && input_image(x, y) == 1 && labeled_image(x, y) == 0
                       labeled_image(x, y) = label;
                       area = area + 1;
                       stack = [stack; x-1, y; x+1, y; x, y-1; x, y+1];
                   end
               end

               if area < threshold_area
                   labeled_image(labeled_image == label) = 0;
                   label = label - 1;
               end
           end
       end
```

```
            end
        end

        output_image = labeled_image > 0;
    end
```

ابتدا طول و عرض نقاطی که سفید هستند را جدا می کنیم و نقاط نزدیک آن ها را پیدا می کنیم و.سپس همسایه های نقاط جدیدی را که پیدا کردیم را پیدا می کنیم تا وقتی که همسایه ی جدیدی یافت نشود.وقتی همسایه جدیدی یافت نشد مساحت بخش را که قبلا حساب کرده ایم اگر از آستانه کمتر بود حذف می کنیم.

تابع mySegmentation به صورت زیر است:

```
function labeled_image = mysegmentation(input_image)
    % Function to label connected components in a binary image
    % Input:
    % input_image: Binary image


    [rows, columns] = size(input_image);
    labeled_image = zeros(rows, columns);
    label = 0;

    for j = 1:columns
        for i = 1:rows
            if input_image(i, j) == 1 && labeled_image(i, j) == 0
                label = label + 1;
                stack = [i, j];

                while ~isempty(stack)
                    current_pixel = stack(1, :);
                    stack(1, :) = [];
                    x = current_pixel(1);
                    y = current_pixel(2);

                    if x >= 1 && x <= rows && y >= 1 && y <= columns && input_image(x, y) == 1 && labeled_image(x, y) == 0
                        labeled_image(x, y) = label;
                        stack = [stack; x-1, y; x+1, y; x, y-1; x, y+1];
                    end
                end
            end
        end
    end
end
```

در این تابع نیز مانند قبل با الگوریتمbfs مانند قسمت قبل نقاط همسایه را که 1 هستند می یابیم و هر وقت نقطه ی جدید پیدا نشد لیبل را یکی اضافه می کنیم.

به ترتیب توابع را اجرا می کنیم.دوبار myremovecom را اجرا می کنیم.یکبار برای حذف نقاط کوچک و بار دوم همه ی نقاط را به جز قاب حذف می کنیم و در یک تصویر حاصل را ذخیره می کنیم.قاب را از تصویر اولیه کم می کنیم تا فقط حروف بماند.

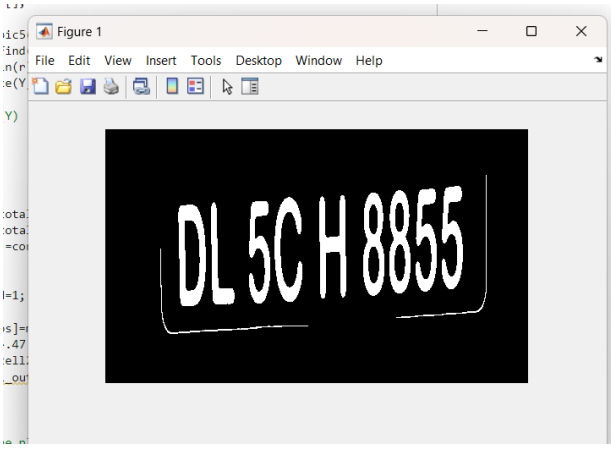
تصاویر را برای چک کردن نشان می دهیم.

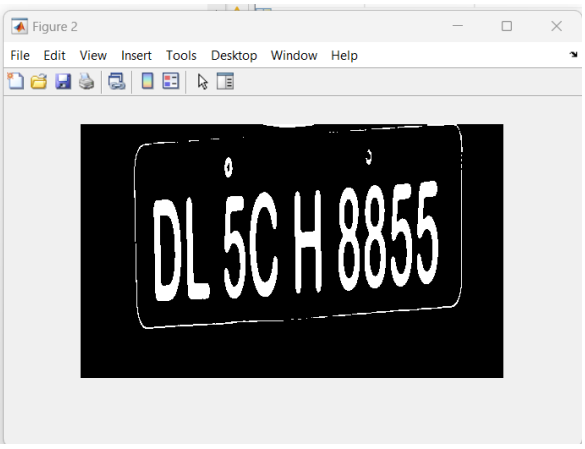
```
pic2=mygrayfun(picture);
pic3=mybinaryfun(pic2,80);
pic4=myremovecom(pic3,300);
totalLetters=size(TRAIN,2);



figure

imshow(pic4);
figure
imshow(pic3);
pic4=imresize(pic4,[500,500]);
pic5=mysegmentation(pic4);


final_output=[];
t=[];
for n=1:max(pic5(:))
    [r,c] = find(pic5==n);
    Y=pic4(min(r):max(r),min(c):max(c));
    Y=imresize(Y,[42,24]);
     %figure
    % imshow(Y)
```

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

% Printing the plate
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
file = fopen('number_Plate.txt', 'wt');
fprintf(file,'%s\n',final_output);
fclose(file);
```

در آخر correlation می گیریم و هر حرفی ماکسیمم correlation را داشت به عنوان جواب برداشته و در خروجی می نویسیم.

کورلیشن را حتما از عددی بیشتر باید در نظر بگیریم که مثلا اجزای اضافی و نویز را با حرف اشتباه نگیرد.





DL5CH8855

بخش دوم:

ابتدا mapset فارسی را لود می کنیم.

```
load TRAININGSET2.mat
%loading picture
[file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
s=[path,file];
picture=imread(s);
picture=imresize(picture,[300 500]);
%figure
pic2=mygrayfun(picture);
pic3=mybinaryfun(pic2,100);
pic4=myremovecom(pic3,30);
totalLetters=size(TRAIN,2);



figure
imshow(pic4);
imshow(pic3);
pic4=imresize(pic4,[500,500]);
pic5=mysegmentation(pic4);


final_output=[];
t=[];
for n=1:max(pic5(:))
    [r,c] = find(pic5==n);
    Y=pic4(min(r):max(r),min(c):max(c));
    Y=imresize(Y,[42,24]);
     %figure
    % imshow(Y)
```

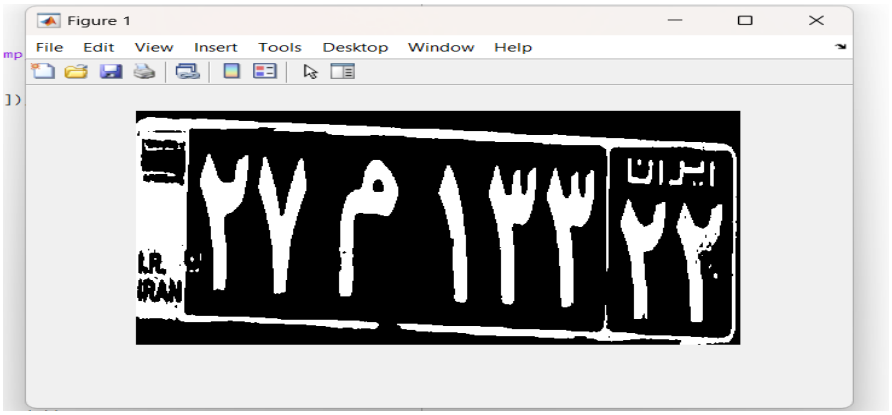
ج

```
   ro=zeros(1,totalLetters);
      for k=1:totalLetters
          ro(k)=corr2(TRAIN{1,k},Y);
      end
      if n==7
          saeed=1;
      end
      [MAXRO,pos]=max(ro);
      if MAXRO>.8
          out=cell2mat(TRAIN(2,pos));
          final_output=[final_output out];
      end
  end

  % Printing the plate
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  file = fopen('number_Plate.txt', 'wt');
  fprintf(file,'%s\n',final_output);
  fclose(file);
```

پروسه ی بخش دوم دقیقا مانند بخش اول است.برای trainingset از ماتریکس ست ترم قبل استفاده شده و خودمان کدی برای لود کردن مپ ست نزدیم.

همه ی پروسه ی تشخیص پلاک مانند گذشته است

```
27M13322
```



بخش سوم

ابتدا تابع mask را معرفی می کنیم:

```
function loc = mask(picture)


imgray = rgb2gray(picture);
threshold = graythresh(imgray);



edgepic = edge(imgray, 'prewitt');
edgepic = imclearborder(edgepic);
imfilled = imfill(edgepic,'holes');
loc = bwareaopen(imfilled,600);
end
```

ابتدا در تابع mask ،edge های تصویر پلاک تشخیص داده می شود و برای بریدن بخش پلاک استفاده شود.

```
function Final = numberplate(picture, loc)
regions = regionprops(loc,'BoundingBox','Area');
pixels = regions.Area;
count = numel(regions);
max = pixels;
boundingBox = regions.BoundingBox;
for i = 1:count
   if max < regions(i).Area
       max = regions(i).Area;
      boundingBox = regions(i).BoundingBox;


    end
end

Final = imcrop(picture, boundingBox);

end
```

سپس با استفاده از تابع numberplate ، تصویر شماره های پلاک را استخراج می کنیم.

```
clc;
clear;
load TRAININGSET2.mat
[file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
s=[path,file];
picture=imread(s);
picture=imresize(picture,[400 600]);
loc = mask(picture);
Final = numberplate(picture, loc);

figure
imshow(Final)

picture=imresize(Final,[300 500]);

pic2=mygrayfun(picture);
pic3=mybinaryfun(pic2,100);
pic4=myremovecom(pic3,30);
totalLetters=size(TRAIN,2);




figure
imshow(pic4);
imshow(pic3);
pic4=imresize(pic4,[500,500]);
pic5=mysegmentation(pic4);
```

```
final_output=[];
t=[];
for n=1:max(pic5(:))
    [r,c] = find(pic5==n);
    Y=pic4(min(r):max(r),min(c):max(c));
    Y=imresize(Y,[42,24]);
     %figure
    % imshow(Y)
```

```
 ro=zeros(1,totalLetters);
    for k=1:totalLetters
        ro(k)=corr2(TRAIN{1,k},Y);
    end
    if n==7
        saeed=1;
    end
    [MAXRO,pos]=max(ro);
    if MAXRO>.8
        out=cell2mat(TRAIN(2,pos));
        final_output=[final_output out];
    end
end
fprintf('final result:%s',final_output)

file = fopen('number_Plate.txt', 'wt');
fprintf(file,'%s\n',final_output);
fclose(file);
```

ابتدا لوکیشن پلاک با استفاده از تابع mask استخراج می کنیم و سپس شماره ها را تشخیص می دهیم.سپس همان پروسه را تکرار می کنیم.تبدیل به تصویر خاکستری می کنیم.باینریش می کنیم.نقاط اضافی را حذف می کنیم.سگمنت بندی می کنیم.کورلیشن می گیریم و حروف را چاپ می کنیم.

خروجی تصویر:

59Y84410

بخش چهارم :

در این بخش ابتدا یک تابع به نام calculate_velocity می نویسیم که به این صورت عمل می کند که نام ویدیو را در ورودی میگیرد سپس ویدیو را می خواند .

سپس بعد از آن دو فریم از آن را میگیریم و آن را با تابع rgb2gray به صورت خاکستری در می آوریم سپس آن ها را به تابع detectSURFFeatures می دهیم تا این تابع نقاط مهم را برای محاسبه سرعت پیدا کند .

1. این فراخوانی تابع تصاویر خاکستری را تحلیل می‌کند تا نقاط جالب را پیدا کرده و برگرداند -این‌ها نقاط کلیدی هستند که در آن‌ها تصویر دارای بافت‌ها یا گوشه‌های متمایز است .
2. **خروجی :**خروجی، points1و points2، اشیایی هستند که اطلاعات مربوط به نقاط کلیدی شناسایی شده را شامل می‌شوند .هر نقطه کلیدی دارای مکان، مقیاس و جهت است .این نقاط در مراحل بعدی برای استخراج و تطبیق ویژگی‌ها بین تصاو یر استفاده می‌شوند.

با تشخیص این ویژگی‌های SURF، ما پایه‌ای را برای مقایسه نقاط کلیدی بین دو فریم ایجاد می‌کنید که برای محاسبه جابجایی و در نتیجه سرعت اجسام در ویدئو ضروری است.

سپس ایندکس های این تصاویر را پیدا می کنیم سپس به روش MSE میانگین مجموع مربعات تفاضل این نقاط مهم در دو فریم را حساب می کنیم و تقسیم بر تفاوت زمان این دو تابع می کنیم مانند فرمول سرعت تا مقدار سرعت بدست بیاید.

اسکریپتی که نوشته شده :

calculate_velocity.m

```
function [velocity,firstFrame,secondFrame] = calculate_velocity(videoFile)
    video = VideoReader(videoFile);

    firstTime = 7.7;
    secondTime = 7.95;

    video.CurrentTime = firstTime;
    firstFrame = readFrame(video);
    video.CurrentTime = secondTime;
    secondFrame = readFrame(video);

    gray1 = rgb2gray(firstFrame);
    gray2 = rgb2gray(secondFrame);

    points1 = detectSURFFeatures(gray1);
    points2 = detectSURFFeatures(gray2);

    [features1, validPoints1] = extractFeatures(gray1, points1);
    [features2, validPoints2] = extractFeatures(gray2, points2);

    indexPairs = matchFeatures(features1, features2);

```

ادامه :

calculate_velocity.m

```
17
18        [features1, validPoints1] = extractFeatures(gray1, points1);
19        [features2, validPoints2] = extractFeatures(gray2, points2);
20
21        indexPairs = matchFeatures(features1, features2);
22
23        matchedPoints1 = validPoints1(indexPairs(:, 1), :);
24        matchedPoints2 = validPoints2(indexPairs(:, 2), :);
25
26        displacements = matchedPoints2.Location - matchedPoints1.Location;
27
28        averageDisplacement = mean(sqrt(sum(displacements.^2, 2)));
29
30
31        timeInterval = secondTime-firstTime;
32
33        velocity = averageDisplacement / timeInterval;
34        disp(['Velocity: ', num2str(velocity), ' pixels per second']);
35    end
36
```

حالا از این تابع در p4.m استفاده می کنیم تا مقدار سرعت را چاپ کند سپس این دو فریم را در دو فایل firstFrame.jpg و secondFrme.jpg می ریزیم تا بتوانیم آن ها را در ادامه کد که مانند بخش p3.m هست استفاده کنیم.

p4.m

```
1
2     clc;
3     clear;
4
5     [v,firstFram,secondFrame] = calculate_velocity("./p4.MOV");
6
7     imwrite(firstFram , "firstFrame.jpg");
8     imwrite(secondFrame,"secondFrame.jpg");
9
10    load TRAININGSET2.mat
11    [file,path]=uigetfile({'*.jpg;*.bmp;*.png;*.tif'},'Choose an image');
12    s=[path,file];
13    picture=imread(s);
14    picture=imresize(picture,[400 600]);
15    figure
16    imshow(picture)
17    loc = mask(picture);
18    Final = numberplate(picture, loc);
19    %figure
20    %imshow(loc)
21    figure
22    imshow(Final)
```

نتیجه آن برای سرعت :

```
Velocity: 701.4877 pixels per second
```

تصویر اول آن :