به نام خدا

تشخیص پلاک خودرو با تکنیک پردازش تصویر و با کمک شبکه های عصبی

گرد آوری

عباس یاسری، سمیرا ترابی، حمیرا باقری

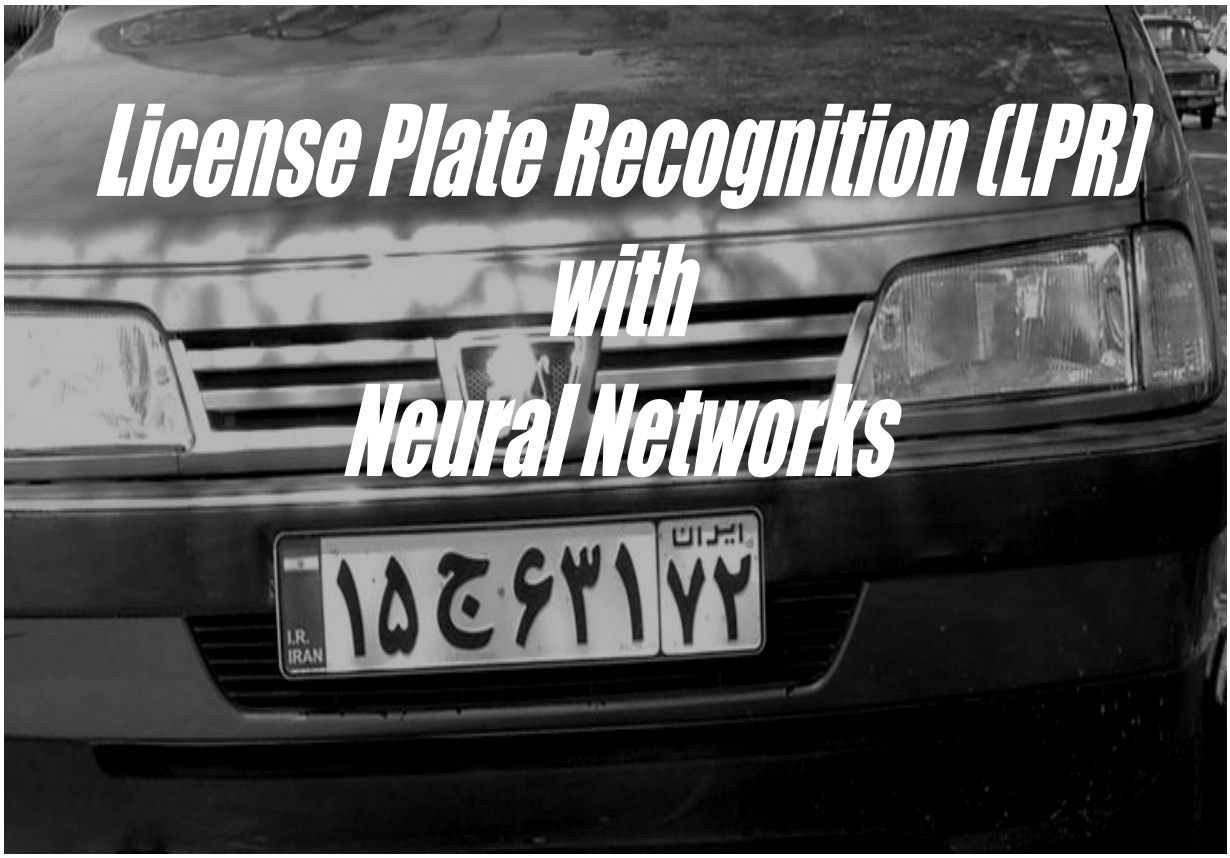
کلمات کلیدی

پردازش تصویر، شبکه های عصبی، تشیخص ارقام، *OCR*

چکیده

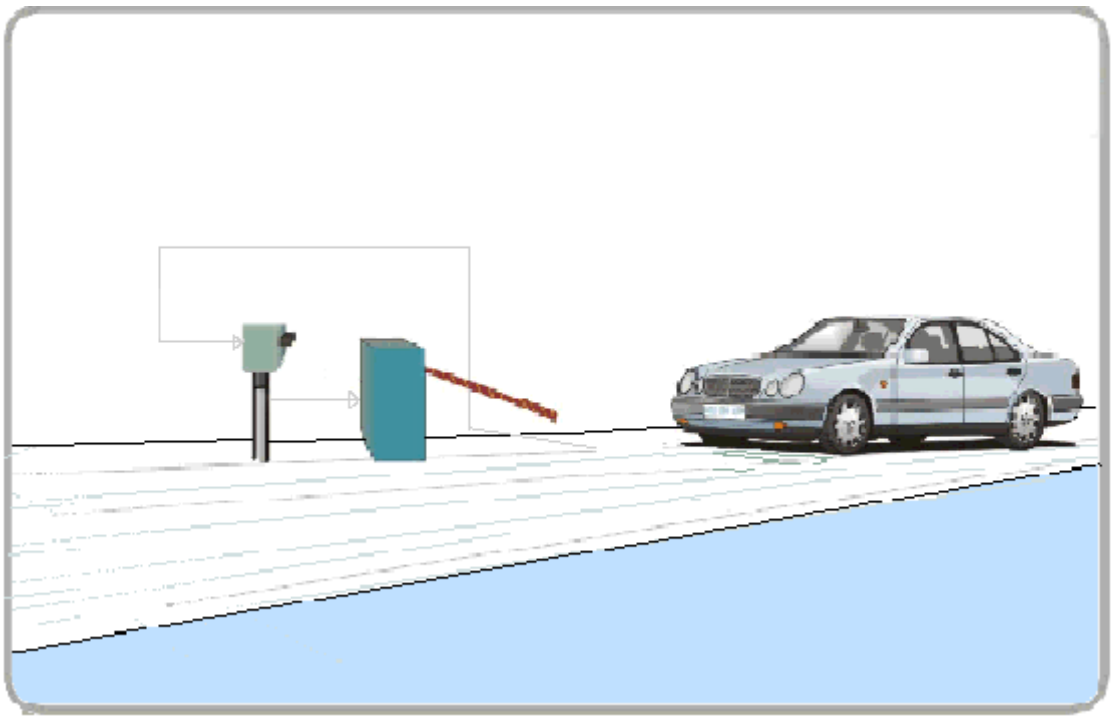
هدف این گزارش، توضیح نحوه ایجاد و پیاده سازی سیستم تشخیص پلاک اتومبیل است.

License Plate Recognition (LPR) with Neural Networks



۱. مقدمه

هدف این گزارش، توضیح نحوه ایجاد و پیاده سازی سیستم تشخیص پلاک اتومبیل میباشد.این گزارش با توضیح انگیزه و هدف اصلی پروژه ،آغاز شده و به ترتیب ،به تشریح چگونگی کارکرد آن خواهد پرداخت.در هر قسمت ،دلیل انتخاب هر روش را گفته و شرایطی را که تحت آنها ،روش مورد نظر شکست خواهد خورد را ،بیان خواهیم نمود ،همچنین توضیح خواهیم داد که چگونه میتوان این روش ها را بهبود بخشید.در آخر نیز گزارش را با توضیح در مورد کارهای آینده پایان خواهیم داد.



نمایی از نحوه کارکرد سیستم

۲. تشریح

۲-۱. هدف پروژه

هدف اصلی این پروژه ایجاد سیستمی جهت تشخیص پلاک اتومبیل ها در مکـانی همچـون درب اصـلی پارکینگ ها می باشد. این سیستم شامل یک کامپیوتر معمولی بوده که با دریافت یک فریم یـا عکـس از اتومبیل شماره پلاک آن را تشخیص می دهد.نرم افزار اصلی بکار گرفتـه میباشـد. ***MATLAB 7***شـده بعلت کمبود وقت ،تمرکز اصلی پروژه بر روی استخراج شماره پلاک از روی عکس ، و تشخیص اعداد آن، قرار گرفت.

۲-۲. محدودیت ها

در اینگونه برنامه ها یا به نوعی ،در پردازش تصویر ،در نظر گرفتن تمام موارد ، روش خـوبی نیسـت.زیـرا بدلیل پیچید گی بیش از حد ،دچار سردرگمی و یا اشتباه شده که سبب می گردد از اصل موضوع عقب بمانیم.به همین دلیل ما محدودیت هایی را به ترتیب ذیل در نظر می گیریم:

الف)پلاک ها بصورت مستطیل شکل و از نوع پلاک های شخصی می باشند.

ب)هیچکدام از عکس ها دچار حالت تار شدگی ،بدلیل سرعت بالا ،نمی باشند.

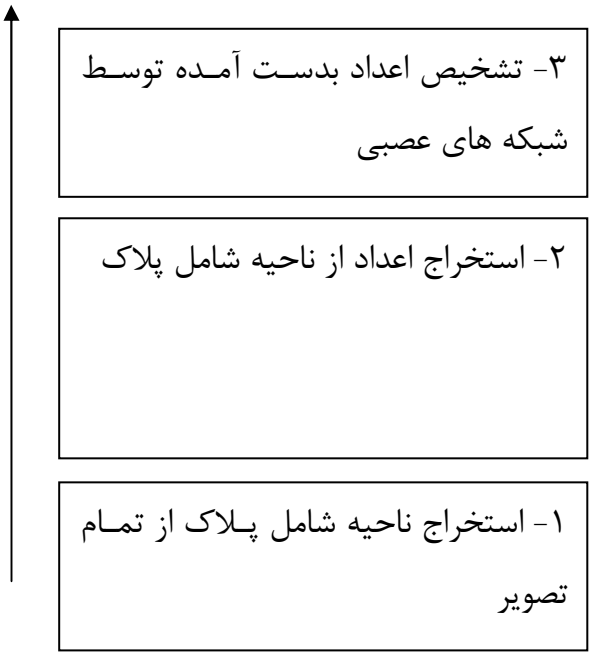
ج)پلاک در انتها و قسمت پایینی اتومبیل قرار گرفته باشد.

د)پلاک بصورت موازی با محور های افقی و عمودی قرار گرفته باشد.

ه)عکس ها از فاصله ۱.۵ متری تا ۲.۵ متری گرفته شوند.

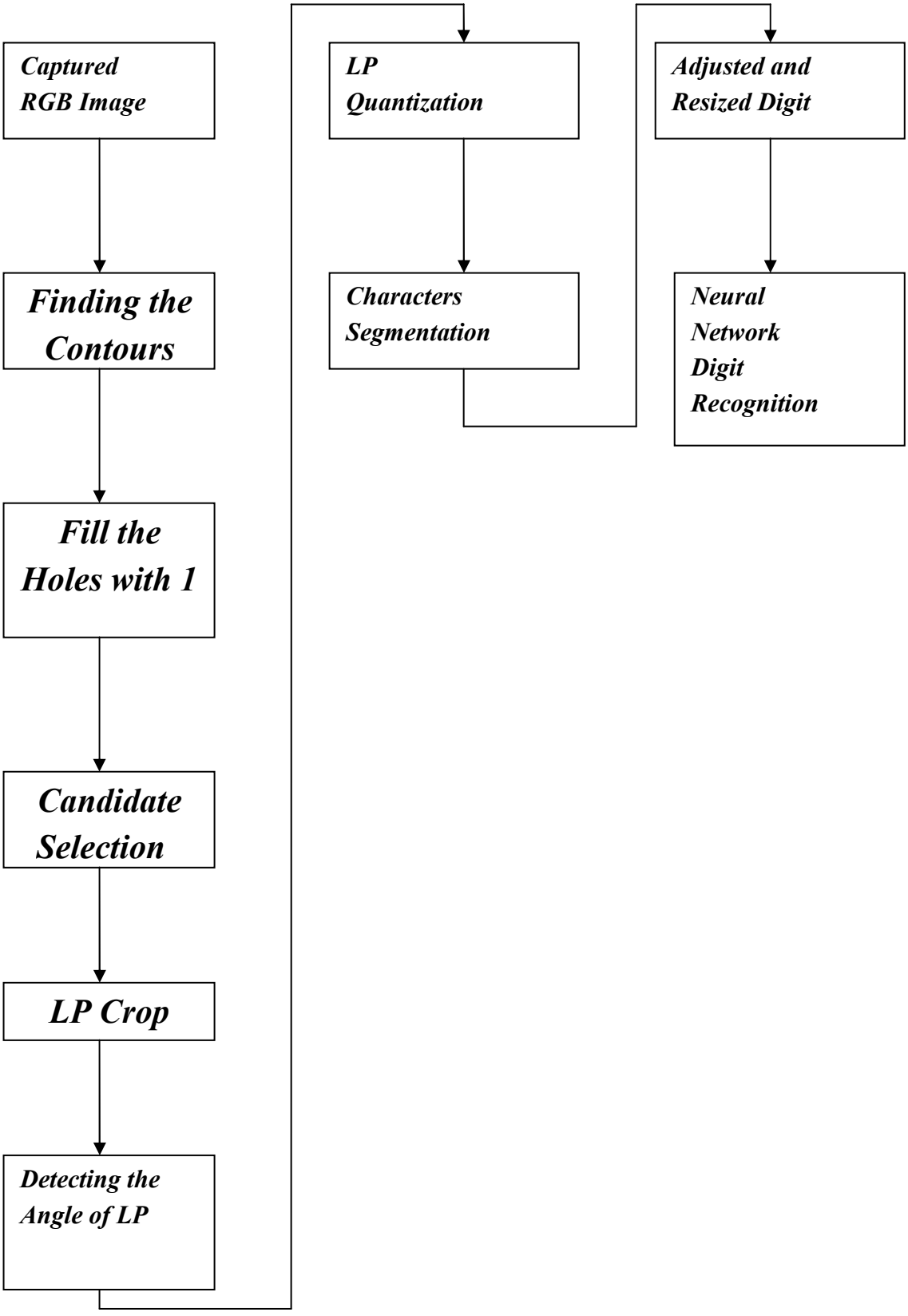
۳. طراحی

سیستم تشخیص پلاک می تواند به بخش های زیر تقسیم گردد:



البته این نوع پیاده سازی را میتوان بصورت حذف کردن اطلاعات از عکس اصلی تا رسیدن به اعداد پلاک ، در نظر گرفت.زیرا عکس اصلی دارای اطلاعات بسیاری ،ولی نامربوط با اعداد پلاک ،می باشد.

دیاگرام زیر باجزئیات بیشتری الگوریتم های بکار رفته را نشان خواهد داد :



۴. نحوه پیاده سازی پروژه

۴-۱. الگوریتم استخراج پلاک

هدف این بخش بیان نحوه استخراج خود پلاک از میان تصویر گرفته شده می باشد.خروجی این قسمت تصویر شامل پلاک می باشد.

ابتدا عکس دریافت شده که دارای ابعاد (۶۰۰*۸۰۰)می باشد را به نوع گری تبدیل میکنیم.سپس جهت دستیابی به محیط اشیاء (پیرامون اشیاء) ،از خروجی حالت قبل مشتق میگیریم.مشتق انتخاب شده از نوع "سبل" می باشد.زیرا برخلاف مشتق "کنی" از پیچیدگی کمتری برخوردار است و سریع میباشد.حال جهت به هم پیوستگی بیشتر این خطوط ،خروجی حالت قبل را به صورت ستاره انبساط میدهیم.هم اکنون که خطوط بیشتر به یکدیگر نزدیک شده اند می توان محیط های بسته را ،پر نمود که البته این کار سبب ایجاد اشیائی توپر می گردد که علت انجام چنین کاری را توضیح خواهیم داد.سپس توسط دستوری این اشیاء به هم پیوسته را شماره گذاری می کنیم.حال از میان این اشیاء شماره گذاری شده بدنبال اشیائی با شرایط ویژه می گردیم.این شرایط ویژه عبارتند از :

۱) داشتن حداقل مساحتی بیشتر از ۶۵۰۰ (در واحد پیکسل)

۲) داشتن حداقل نسبت خانه های سفید به سیاه بیشتر از ۸۲ درصد

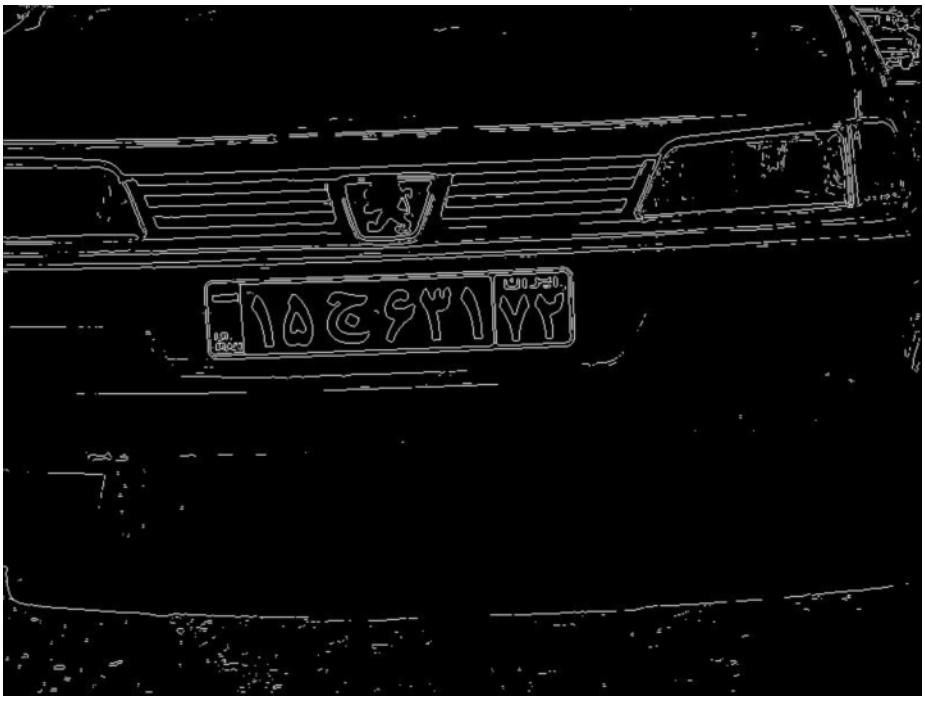
در نهایت بدلیل آنکه ممکن است در بعضی شرایط ،بیش از دو شیئ دارای چنین ویژگی ای باشند ،شیئی با مساحت بیشتر را انتخاب میکنیم.و یا اگر هیچ شیئی با شرایط مورد نظر پیدا نشد ،بزرگترین شیئ موجود در تصویر را انتخاب می کنیم.

حال با داشتن مختصات و ابعاد شیئ مورد نظر آن را از بقیه تصویر جدا میکنیم.معادل همین عکس را از عکس اصلی رنگی نیز ، جدا میکنیم.

سپس با تشخیص زاویه شیئ مورد نظر با محور افقی ، عکس رنگی استخراج شده را به همان اندازه ولی در جهت مخالف دوران میدهیم.

در این وضعیت ما عکسی رنگی ،از ناحیه شامل پلاک خواهیم داشت که میبایست در مرحله بعدی مورد پردازش قرار گیرد.

نمونه ای از عکس دریافت شده از دوربین



نمونه ای از مشتق عکس اصلی

نمونه ای از عکس منبسط شده

نمونه ای از عکسی که حفره هایش پرشده

نمونه ای از ناحیه انتخاب شده

نمونه ای از ناحیه استخراج شده از عکس اصلی

۴-۲. منطق

ابتدا می بایست بیان کرد که جهت استخراج ناحیه شامل پلاک، چندین الگوریتم مد نظر بوده و توضیح خواهیم داد که چرا الگوریتم فوق انتخاب شده است.

الگوریتم اولیه برمبنای رنگ سفید پس زمینه پلاک بود ،که در حقیقت با پیدا کردن ناحیه سفید پلاک ،آنرا از دیگر نواحی موجود در تصویر متمایز می ساخت.ولی بدلیل منحصر بفرد نبودن این ویژگی دچار اشتباه در تشخیص ناحیه مورد نظر می شد ،از جمله دلایل عدم تشخیص صحیح به ترتیب :

۱. تشابه رنگ پس زمینه پلاک با رنگ ماشین
۲. حذف ناحیه سایه دار پلاک
۳. عدم تعیین دقیق رنگ سفید در تصاویر ، بدلیل تفاوت در شدت روشنایی می باشند.

با توجه به معایب ذکر شده، الگوریتم ذیل انتخاب گردید.از آنجایی که هدف اصلی پیدا کردن پلاک میباشد و اینکه ناحیه مذکور ، مستطیلی شکل و دارای نسبت ابعاد معین میباشد ، بدون توجه به رنگ پس زمینه میتوان چنین ناحیه ای را تشخیص داد ، بدین شکل که با گرفتن مشتق از تصویر در حقیقت محیط های بسته موجود در تصویر را تا حدودی مشخص میکنیم و با سفید کردن این نواحی بسته ، اشیاء سفید توپری خواهیم داشت.با توجه به مستطیلی شکل بودن پلاک و با توجه به انتخاب نواحی بصورت مستطیلی شکل ، احتمال بودن پلاک در آن ناحیه ، ماکزیمم میباشد.بنابر این در این نواحی

نسبت پیکسل های سفید به سیاه بیش از ۹۰ در صد خواهد بود.غیر از مواردی که پلاک مورد نظر دارای زاویه بوده ، که در آن صورت این نسبت به حداقل ۸۲ درصد در عمل کاهش خواهد یافت.بنابراین باتوجه به ویژگی های ذکر شده شیئ مورد نظر انتخاب خواهد شد.

۴-۳. نقاط ضعف الگوریتم

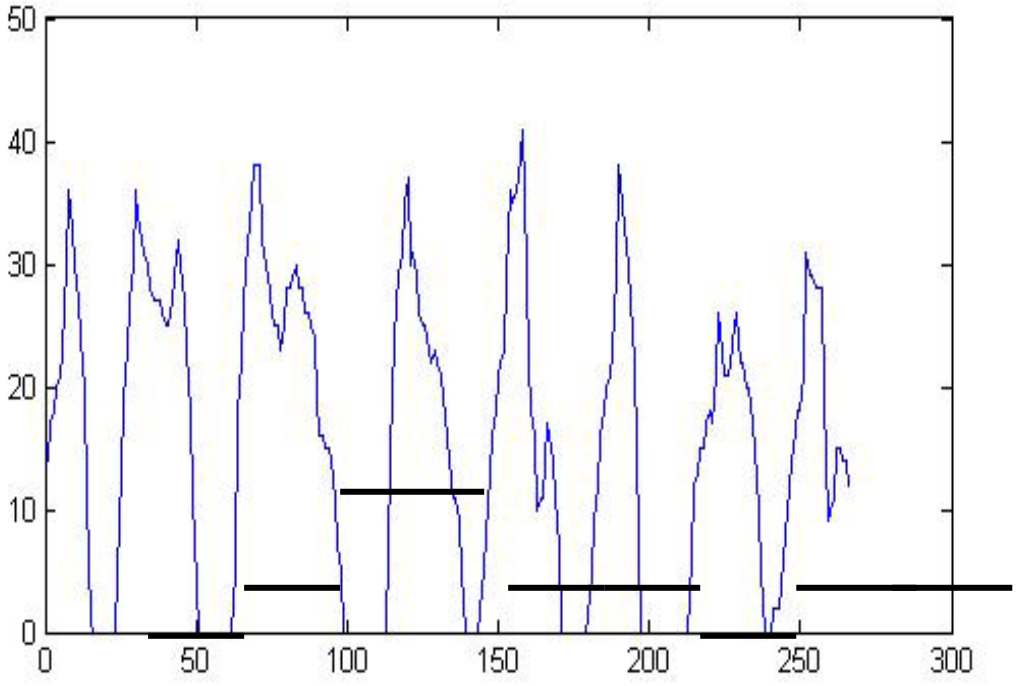
در بعضی موارد الگوریتم فوق هیچ شیئی با شرایط ذکر شده را انتخاب ننموده و در نهایت بزرگترین شیئ موجود در تصویر را از لحاظ مساحت انتخاب نموده که به احتمال ۵۰ در صد حاوی پلاک نمیباشد.

۵. راهبردهایی جهت تکامل الگوریتم برای پیاده سازی در آینده

با اضافه کردن سابروتینی به برنامه تشخیص اشیاء ، احتمال پیدا کردن ناحیه شامل پلاک را افزایش دهیم.

۵-۱. الگوریتم تفکیک اعداد:

جهت تفکیک اعداد از تصویر باینری بدست آمده از مرحله قبل ، در حقیقت با جمع کردن ستونهای ماتریس و رسم آن خواهیم داشت ؛



که برای تشخیص هر عدد ابتدا از پایین سمت چپ شروع کرده و همینطور به طرف راست حرکت میکنیم ، هرگاه به مقداری برابر مقدار شروع شده رسیدیم ، پهنای باند طی شده را محاسبه میکنیم ؛اگر از مقدار از پیش تعیین شده بیشتر باشد مقدار اولیه جهت شروع عملیات را افزایش داده و عملیات را تکرار میکنیم. حال اگر از مقدار از پیش تعیین شده کمتر یا مساوی بود ،این دو نقطه مرز ابتدایی و انتهایی عدد اول از سمت چپ را مشخص میکند .سپس همین روش را برای بقیه تصویر انجام میدهیم تا در نهایت ۸ عدد بدست آوریم.

حال اگر تمام مراحل طی شده ، ولی در نهایت به تعدادی کمتر از ۸ عدد رسیدیم ، آنگاه پهنای باند را کمتر در نظر میگیریم و مراحل قبل را دوباره تکرار میکنیم.

۵-۲. منطق

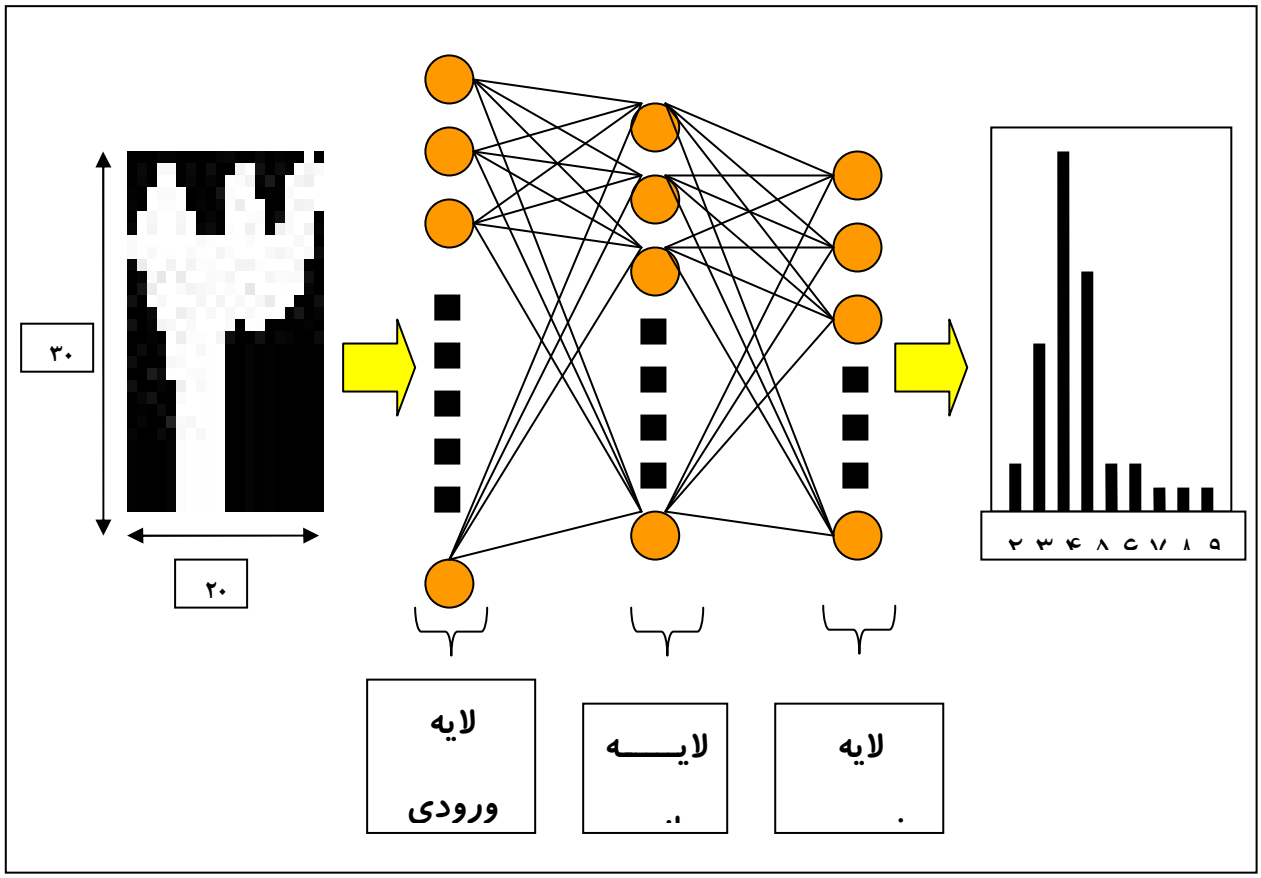
گاهی بدلیل وجود نویز یا تمیز نبودن پلاک ، فاصله میان هر دو عدد بطور واضح مشخص نبوده که با این روش میتوان این نویزها را در نظر نگرفته و اعداد را از هم تفکیک کنیم.

۵-۳. نقاط ضعف

در بعضی موارد ، یک عدد نصف شده و یا دو عدد به عنوان یک عدد نشان داده میشوند.

۶. الگوریتم تشخیص اعداد و شبکه عصبی

ورودی این قسمت ، تصویر تک تک اعداد جدا شده در مرحله قبل میباشد. توسط الگوریتمی این تصاویر به شبکه عصبی داده شده و آنگاه شبکه عصبی آن تصویر را با تصاویری که هنگام آموزش شبکه ایجاد شده اند ، مقایسه نموده و پس از الگوریتم های درونیابی ، تقریب و تصمیم ، بیشترین احتمال اینکه ، به کدام یک از این تصاویر نزدیکتر میباشد را ، به عنوان خروجی ، قرار خواهد داد .



۶-۱. منطق

جهت آموزش این شبکه ، تعداد ۳۴۰ تصویر مورد استفاده قرار گرفت. این تصاویر بصورت تصادفی و بدون نویز بوده ، که توسط الگوریتم دیگری ، از تصاویر پلاک گردآوری شده اند.

۶-۲. نقاط ضعف

بدلیل کافی نبودن بانک اطلاعاتی شبکه ، بعضی مواقع شبکه ، دچار اشتباه شده و مقدار خروجی آن غلط میباشد .

نمونه ای از تصاویری جهت آموزش شبکه برای عدد ۴

۷. راهبردهایی جهت تکامل الگوریتم برای پیاده سازی در آینده

با اضافه کردن و گسترش بانک اطلاعاتی شبکه عصبی و همچنین با در نظر گرفتن نویز ، بصورت عملی ، خواهیم توانست دقت شبکه را تا حد قابل توجهی بهبود دهیم .

۸. برنامه ایجاد و آموزش شبکه عصبی

```
C=[];
for j=20:-1:1
   B=[];
   for i=1:17
   im = imread(strcat(strcat('samples\',num2str(i),'/',num2str(i),'--',strcat(num2str(j),'.jpg'))));
   bin=im2bw(im,.7);
   b=im2col(bin,[30 20],'distinct');
   B=[B b];
   end;
   C=[C,B];
end;


G=eye(17);

net = newff(minmax(C),[17 17],{'logsig' 'logsig'},'traingdx');
net.LW{2,1} = net.LW{2,1}*0.01;
net.b{2} = net.b{2}*0.001;


net.performFcn = 'sse';       % Sum-Squared Error performance function
net.trainParam.goal = 0.0001;     % Sum-squared error goal.
net.trainParam.show = 20;      % Frequency of progress displays (in epochs).
net.trainParam.epochs = 5000;  % Maximum number of epochs to train.
net.trainParam.mc = 0.95;      % Momentum constant.



e1=C(:,1:17);
e2=C(:,18:34);
e3=C(:,35:51);
e4=C(:,52:68);
e5=C(:,69:85);
e6=C(:,86:102);
e7=C(:,103:119);
e8=C(:,120:136);
e9=C(:,137:153);
e10=C(:,154:170);
e11=C(:,171:187);
e12=C(:,188:204);
e13=C(:,205:221);
e14=C(:,222:238);
e15=C(:,239:255);
```

```
e16=C(:,256:272);
e17=C(:,273:289);
e18=C(:,290:306);
e19=C(:,307:323);
e20=C(:,324:340);
% e21=C(:,201:210);
% e22=C(:,211:220);
% e23=C(:,221:230);
% e24=C(:,231:240);
% e25=C(:,241:250);
% e26=C(:,251:260);
% e27=C(:,261:270);
% e28=C(:,271:280);
% e29=C(:,281:290);
% e30=C(:,291:300);



[net,tr] = train(net,e1,G);
[net,tr] = train(net,e2,G);
[net,tr] = train(net,e3,G);
[net,tr] = train(net,e4,G);
[net,tr] = train(net,e5,G);
[net,tr] = train(net,e6,G);
[net,tr] = train(net,e7,G);
[net,tr] = train(net,e8,G);
[net,tr] = train(net,e9,G);
[net,tr] = train(net,e10,G);
[net,tr] = train(net,e11,G);
[net,tr] = train(net,e12,G);
[net,tr] = train(net,e13,G);
[net,tr] = train(net,e14,G);
[net,tr] = train(net,e15,G);
[net,tr] = train(net,e16,G);
[net,tr] = train(net,e17,G);
[net,tr] = train(net,e18,G);
[net,tr] = train(net,e19,G);
[net,tr] = train(net,e20,G);
% [net,tr] = train(net,e21,G);
% [net,tr] = train(net,e22,G);
% [net,tr] = train(net,e23,G);
% [net,tr] = train(net,e24,G);
% [net,tr] = train(net,e25,G);
% [net,tr] = train(net,e26,G);
% [net,tr] = train(net,e27,G);
% [net,tr] = train(net,e28,G);
```

```
% [net,tr] = train(net,e29,G);
% [net,tr] = train(net,e30,G);

A = sim(net,C);
A


C=[];
for j=1:20
   B=[];
   for i=1:17
   im = imread(strcat(strcat('samples\',num2str(i),'/',num2str(i),'--',strcat(num2str(j),'.jpg'))));
   bin=im2bw(im,.7);
   b=im2col(bin,[30 20],'distinct');
   B=[B b];
   end;
   C=[C,B];
end;



e1=C(:,1:17);
e2=C(:,18:34);
e3=C(:,35:51);
e4=C(:,52:68);
e5=C(:,69:85);
e6=C(:,86:102);
e7=C(:,103:119);
e8=C(:,120:136);
e9=C(:,137:153);
e10=C(:,154:170);
e11=C(:,171:187);
e12=C(:,188:204);
e13=C(:,205:221);
e14=C(:,222:238);
e15=C(:,239:255);
e16=C(:,256:272);
e17=C(:,273:289);
e18=C(:,290:306);
e19=C(:,307:323);
e20=C(:,324:340);
% e21=C(:,201:210);
% e22=C(:,211:220);
% e23=C(:,221:230);
% e24=C(:,231:240);
% e25=C(:,241:250);
```

```
% e26=C(:,251:260);
% e27=C(:,261:270);
% e28=C(:,271:280);
% e29=C(:,281:290);
% e30=C(:,291:300);



[net,tr] = train(net,e1,G);
[net,tr] = train(net,e2,G);
[net,tr] = train(net,e3,G);
[net,tr] = train(net,e4,G);
[net,tr] = train(net,e5,G);
[net,tr] = train(net,e6,G);
[net,tr] = train(net,e7,G);
[net,tr] = train(net,e8,G);
[net,tr] = train(net,e9,G);
[net,tr] = train(net,e10,G);
[net,tr] = train(net,e11,G);
[net,tr] = train(net,e12,G);
[net,tr] = train(net,e13,G);
[net,tr] = train(net,e14,G);
[net,tr] = train(net,e15,G);
[net,tr] = train(net,e16,G);
[net,tr] = train(net,e17,G);
[net,tr] = train(net,e18,G);
[net,tr] = train(net,e19,G);
[net,tr] = train(net,e20,G);
% [net,tr] = train(net,e21,G);
% [net,tr] = train(net,e22,G);
% [net,tr] = train(net,e23,G);
% [net,tr] = train(net,e24,G);
% [net,tr] = train(net,e25,G);
% [net,tr] = train(net,e26,G);
% [net,tr] = train(net,e27,G);
% [net,tr] = train(net,e28,G);
% [net,tr] = train(net,e29,G);
% [net,tr] = train(net,e30,G);

A = sim(net,C);
A
```

۹. برنامه اصلی تشخیص پلاک

```
function  go(IM);
% H = fspecial('unsharp');
% I = imfilter(IM,H,'replicate');
G = rgb2gray(IM);
E=edge(G,'sobel');
D=imdilate(E,strel('diamond',1));
%imshow(D);
%pause;
L=imfill(D,'holes');
%imshow(L);
%pause;
stat =
regionprops(bwlabel(L),'Area','Extent','BoundingBox','Image','Orientation','Centroid');




depth = -1;
  for i = 1 : length([stat.Area] )
     if stat(i).BoundingBox(2) >= depth && stat(i).Extent >= 0.82 && stat(i).Area > 6500

        depth = stat(i).BoundingBox(2);
     end;
  end;
  % finding the components which are at the depth "depth":
  r = [];
  for i = 1 : length([stat.Area])
     if stat(i).BoundingBox(2) == depth && stat(i).Extent >= 0.82 && stat(i).Area > 6500
        r = [r stat(i).Area];
     end;
  end;



if(length(r) == 0)
      index = (find([stat.Area] == max([stat.Area])));
else
      % otherwise, taking the candidate with maximum area from the
      % filtered candidates:
      index = (find([stat.Area] == max(r)));
```

```
end;


out=stat(index).Image;


%imshow(out);
%pause;


x1 = floor(stat(index).BoundingBox(1));
x2 = ceil(stat(index).BoundingBox(3));
y1 = ceil(stat(index).BoundingBox(2));
y2 = ceil(stat(index).BoundingBox(4));

% mx = ceil((x2 - x1)/2) + x1;
% my = ceil((y1 - y2)/2) + y1;
%
% dx = mx - stat(index).Centroid(1);
% dy = my - stat(index).Centroid(2);
%
% x2 = x2 - ceil(dx/2);
% x1 = x1 + ceil(dx/2);
% y2 = y2 - ceil(dy/2);
% y1 = y1 + ceil(dy/2);


im2=imcrop(IM(:,:,:),[x1,y1,x2,y2]);
G2=imcrop(G(:,:),[x1,y1,x2,y2]);

%imshow(im2);
%pause;
%imshow(G2);
%pause;




g=rgb2gray(im2);
```

```
g = imadjust(g, stretchlim(g), [0 1]);
g=im2double(g);
thre = opthr(g);
if (thre < 0.44 && thre > 0.36)
   thre1 = 0.5*thre;
else
   if ( thre < 0.36 | thre > 0.5)
      thre1 = thre;
   else
      thre1 = 0.92*thre;
   end;
end;



bw2=im2bw(g,thre1);

%imshow(bw2);
%pause;

angle = 0;
if abs(stat(index).Orientation) >=1
   out=imrotate(out,-stat(index).Orientation);
   bw2=imrotate(bw2,-stat(index).Orientation);
   angle = stat(index).Orientation;
end;

 bw2=imdilate(bw2,strel('line',1,0));

%imshow(bw2);
%pause;




bw22=imfill(bw2,'holes');
%imshow(bw22);
%pause;
bw2 = xor(bw22 , bw2);
%imshow(bw2);
%pause;
```

```
% v=sum(bw2);
%  h=sum(bw2');
% bw2 = cut_bw_img(bw2, 1.1*mean(v), 0);
% bw2 = cut_bw_img(bw2, mean(h), 1);
%
% plot(v);
% %pause;
% plot(h);
% %pause;



% indexmax = (find([stat.Area] == max([stat.Area])));
% maxarea =[stat(index).Area];
%BW=bwareaopen(L,maxarea);

bw3 = bw2;
%bw3=imfill(bw3,'holes');



% %imshow(bw3);
% %pause;




bw3=bwareaopen(bw3,50);

%imshow(bw3);
%pause;

stat = regionprops(bwlabel(bw3),'Area');
index1 = (find([stat.Area] == max([stat.Area])));
maxarea =[stat(index1).Area];
bw4=bwareaopen(bw3,maxarea(1,1));
if (maxarea - mean([stat.Area])) >= 15.7*mean([stat.Area])
   bw3 = bw3 - bw4;
```

```
    q = 1;
else
    bw4=bwareaopen(bw3,2280);
    bw3 = bw3 - bw4;
    q = 0;
    end;



v=sum(bw3);
 h=sum(bw3');
bw3 = cut_bw_img(bw3, 0.8*mean(v), 0);
bw3 = cut_bw_img(bw3, 0.8*mean(h), 1);
%imshow(bw3);
%pause;
% stat = regionprops(bwlabel(bw3),'Area');
%
% index1 = (find([stat.Area] == max([stat.Area])));
%
% maxarea =[stat(index1).Area];
% bw4=bwareaopen(bw3,maxarea);
% if (maxarea - mean([stat.Area])) >= 20*mean([stat.Area])
%     bw3 = bw3 - bw4;
% end;

% bw3=bwareaopen(bw3,30);
% bw3=imcomplement(bw3);
% bw3=bwareaopen(bw3,6000);
% bw3=imcomplement(bw3);
% %imshow(bw3);
% %pause;

seg=character_segmentation(bw3);



load net.mat;
num = [];
for i=1:8
    if i~=3
    ex=bw3(:,seg(i,1):seg(i,2));
    stat =regionprops(bwlabel(ex),'Image','Area','BoundingBox');
```

```
    for j = 1 : length([stat.Area])
      if [stat(j).Area] == max([stat.Area])

        % [hi wi] = size(ex);
         %im = imcrop(ex,[stat(j).BoundingBox(1) stat(j).BoundingBox(2) abs( wi -
stat(j).BoundingBox(1) ) abs( hi - stat(j).BoundingBox(2) )]) ;
         im = stat(j).Image;



      end;
    end;

    else
       im=im2bw(bw3(:,seg(2,2)+13:-13+seg(4,1)),0.5);
       im(1:5,:)=zeros(5,(seg(4,1)-seg(2,2)-25));

    end;
    im = imrotate(im,0.85*angle);
    im=imresize(im,[30 20],'bilinear');
    imshow(im);

    %imwrite(im,strcat('im2\',num2str(rand()),'.jpg'));
    vec = double(im2col(im, [30 20], 'distinct'));

    rslt = my_sim(net, vec);
    [Y,I] = max(rslt);

    digit = char('0'+I);
    %title(digit);
    if I == 10
       digit = 'B';
    end;
    if I == 11
       digit = 'D';
    end;
    if I == 12
       digit = 'EIN';
    end;
    if I == 13
       digit = 'J';
```

```
    end;
    if I == 14
        digit = 'M';
    end;
    if I == 15
        digit = 'SIN';
    end;
    if I == 16
        digit = 'V';
    end;
    if I == 17
        digit = 'SAD';
    end;

    num = [num digit];
%    pause;
%    num = input('prompt','s')
%    som = input('prompt')
%    imwrite(im,strcat('samples\',num,'\',num,'--',num2str(som),'.jpg'));




end;

imshow(bw3);
 title(num);

 return;
```

برنامه تفکیک اعداد

```
function [seg] = character_segmentation(bw);



DIGIT_WIDTH = 46;
```

```
MIN_AREA = 210;



seg = segment(bw, DIGIT_WIDTH, MIN_AREA);
[x y] = size(seg);




if x < 8
   for i = 1 : x
      bw(:,seg(i,2))=0;
   end;
   seg = segment(bw, DIGIT_WIDTH, MIN_AREA);
end;



area = [];
for i = 1 : x
   pic = bw(:, seg(i,1) : seg(i,2), :);
   area(i) = bwarea(pic);
end;

area1 = sort(area);
seg = seg';

for j = 1:(length(area1)-8)
   i = find(area == area1(j));
   len = length(area);
   if i == 1
      area = [area(2:len)];
      seg = [seg(:,2:len)];
   elseif i == len
      area = [area(1:i-1)];
      seg = [seg(:,1:i-1)];
   else
      area = [area(1:i-1) area(i+1:len)];
      seg = [seg(:,1:i-1) seg(:,i+1:len)];
   end;
end;
```

```
seg = seg';

return;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [segmentation] = segment(im, digit_width, min_area);




segmentation = [];


t = sum(im);


seg = clean(find_valleys(t, 2, 1, digit_width), 3);





j = 1;
for i = 1 : (length(seg) - 1)
   band_width = seg(i+1) - seg(i);
   maxi = max(t(1, seg(i):seg(i+1)));
   if(maxi * band_width > min_area)
      segmentation(j, 1) = seg(i);
      segmentation(j, 2) = seg(i+1);
      j = j + 1;
   end;
end;

return;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [s] = find_valleys(t, val, offset, digit_width);
```

```
s = find(t < val);
function again.
if(length(s) < 2)
   s = find_valleys(t, val + 1, offset, digit_width);
   return;
end;

% If no point is found terminating:
if length(s) == 0
   return;
end;



if((t(1,1) >= val) && s(1) ~= 1)
   s = [1 s];
end;
if((t(1, length(t)) >= val) && s(length(s)) ~= length(t))
   s = [s length(t)];
end;



s = add(s, offset - 1);


s = clean(s, 3);



while bad_segm(s, digit_width) == 1
   for i = 1: (length(s) - 1)
      if (s(i + 1) - s(i)) > digit_width



         sub_vec = t(1, s(i) - offset + 1 : s(i+1) - offset + 1);



         s = [s(1 : i) find_valleys(sub_vec, val + 1, s(i), digit_width) s(i+1 : length(s))];
      end;
   end;
```

```
end;

return;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [bool] = bad_segm(s, digit_width);


if length(s) == 0
   bool = 0;
   return;
end;

tmp = s(1);
bool = 0;
for i = 2 : length(s)
   if(s(i) - tmp) > digit_width
      bool = 1;
      return;
   end;
   tmp = s(i);
end;
return;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [t] = clean(s, val);


t = [];
len = length(s);
i = 2;
j = 1;
while i <= len
   while(s(i) - s(i-1) <= val)
      i = i + 1;
      if(i > len)
         return;
      end;
   end;
   if j == 1 || (s(i-1) - t(j-1)) > val
```

```
        t(j) = s(i-1);
        j = j + 1;
    end;
    t(j) = s(i);
    j = j + 1;
    i = i + 1;
end;
return;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [t] = add(s, val);


len = length(s);
t = [];
for i = 1:len
    t(i) = s(i) + val;
end;
return;








return;
```

۱۰. برنامه Demo