Comparison of Vehicle License Recognition Algorithms with MATLAB

Name: Siqiao Ruan

School Number: 1727123

Email: ruansiqiao@stu.pkuschool.edu.cn

Discipline of the thesis: Computer Vision, Artificial Intelligence

Supervisor: Zhao Chun

Word Count: 6582

Abstract

This essay describes, analyzes and compares different computer vision algorithms applied in vehicle license recognition, discussing their advantages, disadvantages and applicability. In this essay, a full license-plate recognition algorithm is coded, using MATLAB as its platform. The program is divided into three main phases: image preprocessing and vehicle plate locating, license processing and character segregation, and Optical Character Recognition (OCR). The purpose of this essay is to achieve a perfect-percentage vehicle license recognition algorithm and find the most suitable algorithm in each step to apply in the field of vehicle and traffic management.

Key Words: Vehicle plate recognition, MATLAB, Computer vision, OCR

Contents

1. Introduction

A Vehicle license recognition algorithm is an algorithm aim to digitize the numbers in the picture of license plate that was taken by surveillance camera. To achieve this goal, the algorithm needs to locate the car plate from the background, cutting every character out, and digitalize the image number or characters. The bases of this program is computer vision.

Computer vision is a discipline that studies how to reconstruct, interpret and understand a 3D scene from its 2D images in terms of the properties of the structures present in the scene.[1] The reason that we are able to digitalize a car plate is that every image that we see in our daily life can be converted into a matrix. The composition of every pixel form the picture we see. The most popular form is RGB(Red, Green and Blue), using a three channel matrix with the value of 0-255 to express a picture. By manipulating the matrix, we can make the algorithm function as our bionic eyes let the algorithm recognize and distinguish objects in images. This is how computer vision works and why it is fascinating. Using the multiple method of computer vision, we can achieve the recognition of car license. The reason that this research use MATLAB as its platform is that MATLAB have many build-in function about matrixes which would make my coding process simplified. Also, MATLAB is easy to present the matrix into picture which give the tester a great experience and convenience to adjust and present.

In conclusion, this essay is aim to analyze different computer vision algorithms using in vehicle license recognition algorithm and search for the most suitable and accurate one in each step. With the demonstration of experiment with actual license plate, one can clearly see

[1] "WS16a: Human Interaction in Computer Vision." 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops) (2011): n. page 3.

the difference between these algorithm and their features. This essay will also describe the building process of this algorithm.

2. The Development of Vehicle License Recognition

2.1 Current Situation of Vehicle License Recognition

The technology of vehicle license has been used all over the world. The most successful and precise application is in Chinese ETC system. It is applied in the electronic automatic fee collection system on the highway. The application of vehicle license recognition algorithm reduce the demand of man power, implementing the unmanned charging system.

2.2 Usage and Prospect of Vehicle-Plate Recognition Algorithm

The vehicle plate recognition algorithm is applied in many field related traffic and urban management. The most common application is shown below:

(1) Over-speed detection and recording

The vehicle plate recognition algorithm can be linked to the surveillance camera all over the city, partnered with the speed monitoring system. Once the speed monitor detect anomaly in speed, the vehicle plate recognition algorithm is able to digitalize the car plate and record in the database. With this technology, there is no need to hire many person to watch these surveillance camera.

(2) Enforcement Department

The enforcement department can use this technology to find “wanted” vehicle. Linking with the cameras all over the nation, it is much easy to locate and track down the criminal using this technology.

(3) Intelligent Community

The vehicle plate recognition algorithm can greatly contribute to the future intelligent community. The gate of an intelligent community can automatically record the in and out situation without the guard, and track down the movement of these vehicle to ensure the safety of the community.

(4) Parking Management

Using vehicle plate algorithm, parking can be more easily managed. We can let every car only park in their exclusive spot without monitoring, recognizing their car plate to confirm identity, so that other car can not steal their parking spots.

3. Preprocessing

3.1. Resizing the Original Vehicle License

Uniformization and initialization is a critical process in every image recognition algorithm. Initializing the program can help you clear all the variables and command and close all the residue windows, making the whole program ready for a new run. Uniformization, in this case, is to uniform the size of the photos, so that some fixed parameters would be more fit to the image, thus have a better outcome. Another reason is that some image is too big for MATLAB, so the program choose to resize all the photos in to 900×1200 pixel. The codes are as follows:

```
%Initializing
clc;clear all     %Clear workspace(variables) and command windows
close all         %Close all Windows

%Resizing
I = imread('车牌照片\1-8.jpg');         %Load Image
figure(1), imshow(I); title('Original Picture');          %Show Picture
I = imresize(I,[900,1200]);     %Resize the image
```

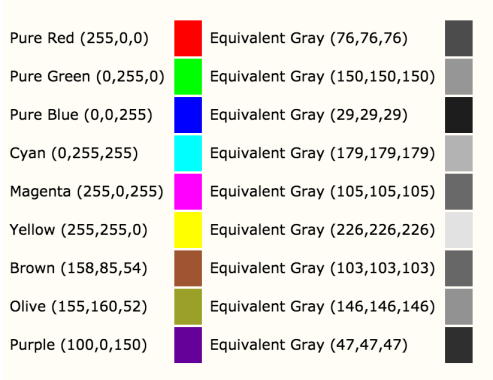

```
[m,n] = size(I);          %Record the size of the image
n = n/3;          % Change the 3 channel n value into one channel
```

3.2. Graying Algorithm Comparison

Graying is the process that transform the three channel matrix picture into a one channel matrix(transfer the three value in one pixel into one), representing the intensity of the color from white to black. There are lots of graying methods which have different features.

(1)Gray = 0.299×R + 0.587×G + 0.114×B

This formula and this ratio is formulate by mathematician, biologist and cognitive scientist, believing that this ratio is best for human bring to receive when expressing the colorful world in gray. Since pure green is lighter than pure red and pure blue, it has a higher weight. Pure blue is the darkest of the three, so it receives the least weight.[2]



[2]How to Convert RGB to Grayscale." http://www.had2know.com/technology/rgb-to-gray-scale-converter.html"

(2) R, G, B single Channel

This method is pretty self-explanatory. Just simply extract the value of one channel, and use it as the grayscale.

(3) RGB mean value

Gray = (R+G+B)/3

(4) LAB Color Space

LAB color space is a color presenting mode send out by CIE(Commission Internationale de L'Eclairage) in 1976. L stands for luminosity. A stands for red to green opponent colors with green at negative A values and red at positive A values. B stands for yellow/blue opponent with blue at negative b* values and yellow at positive b* values.[3] They form a three dimensional color system as below.

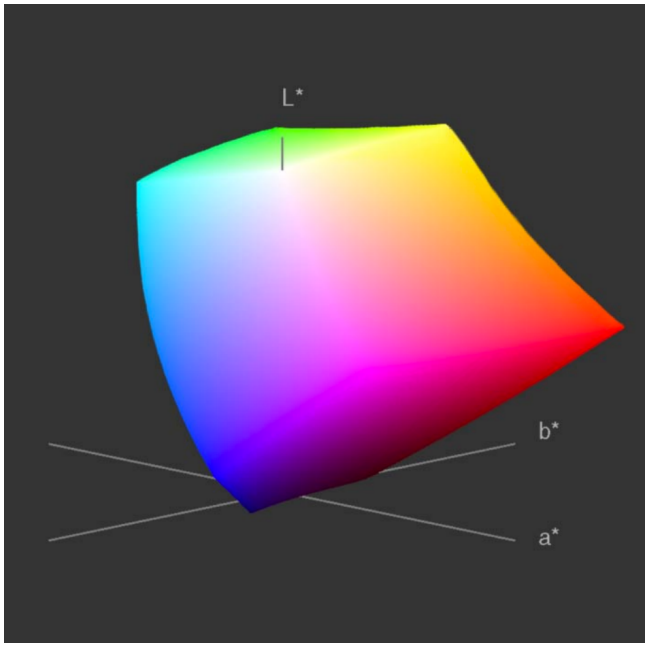


Figure 1, LAB[4]

[3] "Lab Color Space." https://en.wikipedia.org/wiki/Lab_color_space

[4] X. Gernot Hoffmann CIELab Color Space Contents (n.d.)

To see how well these method apply to the car plate recognition. A little test was established, the result is shown as below.

Original Image

Gray=0.299*R+0.587*G+0.114*B

RGB mean value

R Channel

G Channel

B Channel

LAB Color Space

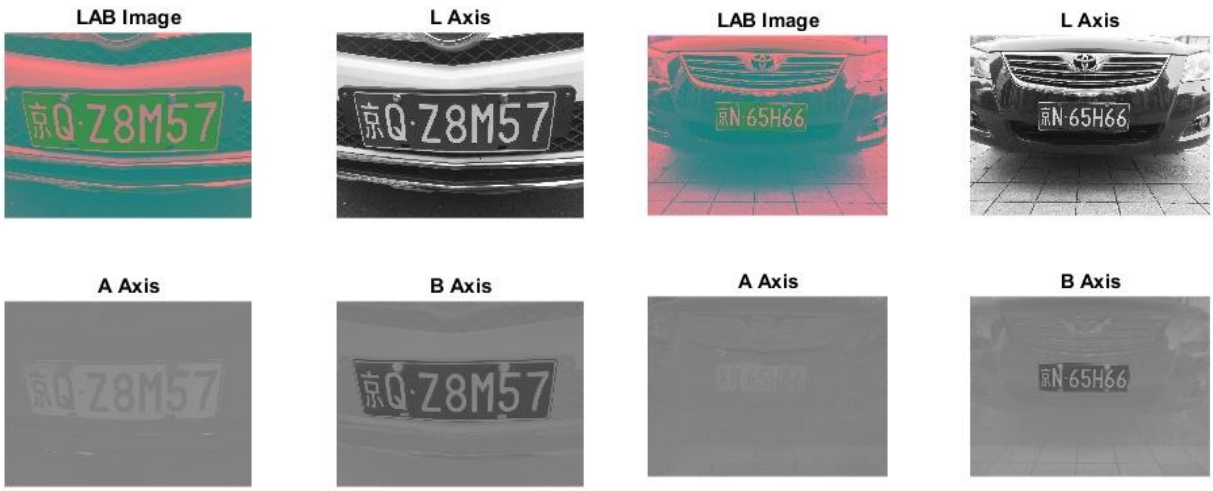
L Axis

A Axis

B Axis

Clearly, the purpose of the graying in vehicle recognition is to highlight the car plate portion in the image. Since, the car plate in China is blue(C75% M68% Y67% K89%). We would want to particularly high light it. From the test, we can see that in for RGB channel, R channel display the car plate most concentrated. However, that is not the best solution. In LAB color spacing mode, the B channel not only highlight the car plate, but also weaken the background, since blue is the positive value in this axis, and the background usually contain just a small portion of blue. Therefore, B axis in LAB channel would be use in the vehicle license recognition algorithm. The code and result is shown below:

```
%Graying
cform = makecform('srgb2lab');
labS = applycform(I,cform);              %Transfer RGB into LAB
figure(2),subplot(2,2,1)
imshow(labS);title(‘LAB Image');

L = labS(:,:,1);          %Extract L channel
A = labS(:,:,2);          %Extract A channel
B = labS(:,:,3);          %Extract B channel
subplot(2,2,2)
imshow(L);title(‘L Axis')
subplot(2,2,3)
imshow(A);title(‘A Axis')
subplot(2,2,4)
imshow(B);title(‘B Axis')
```



4. License Plate Locating

4.1. Binaryzation and Threshold

Binaryzation is the process that transfer a grayscale image into a image that is only consists of black and white(0 & 1). Threshold is the value between 0-255 that determine which color is black and which color is white. For example, if the threshold is 156, than the pixel from 0-155 will be black(0), and 156-255 will be white. The choosing of the threshold is the key of binaryzation. There are many ways to choose a threshold, including Otsu, mode, Bernsen, Niblack, Sauvola. Otsu is an algorithm that based on the interclass variance of the matrix. Mode method is based on its mode. Bernsen, Niblack and Sauvola are all locally threshold choosing which distinguish the image into several part, and choose the different threshold in every part to maximize the accuracy and contrast ratio. However, actually in the license plate locating process, the threshold choosing do not need to be extremely precise. The resulting threshold of these algorithm is basically the same, and these small difference would not influence the outcome. In this program, the binaryzation process would be combine with the edging algorithm.

4.2. Edging Algorithm Comparison

Edging algorithm is the process to determine the edges in the picture. Academically, edges are set of the pixels that has great value changes. Because of the edges in car plate is most concentrated, we can use this feature to locate the car plate. Also there are multiple algorithms that can detecting edges, including Sobel, Roberts, Prewitt, Lapalacian, and Canny etc. Some analyze of these algorithms are shown below.

(1)Sobel

Sobel uses the form of filtering operator to detect edges. The model it use is shown as below. Sobel has a better effect, dealing with the image that have more noise. But, the edges' locating is not that precise.

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & 1 \end{bmatrix}$$

Figure 1, Sobel[5]

(2)Roberts

Roberts uses the form of gradient operator to detect edges. The model it use is shown as below. Roberts is good with the image with low noise and gradient ramp, but the edges it form is usually thick.

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Figure 2, Roberts[5]

(3)Prewitt

Prewitt uses the form of weighted average operator to detect edges. The model it use is shown as below. Prewitt can effective detect the edge of image with more noise and gradient ramp, but it is easily form discontinuity point.

$$\begin{bmatrix} 1 & -1 & 1 \\ 1 & -1 & 0 \\ 1 & -1 & -1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & -0 \\ -1 & -1 & -1 \end{bmatrix}$$

Figure 3, Prewitt[5]

[5] "Comparison between edging algorithms in Matlab" http://www.kongzhi.net/cases/caseview.php?id=2368

(4)Laplacian

Laplacian uses the form of second derivative operator to detect edges. The model it use is shown as below. Laplacian is weak with noises.

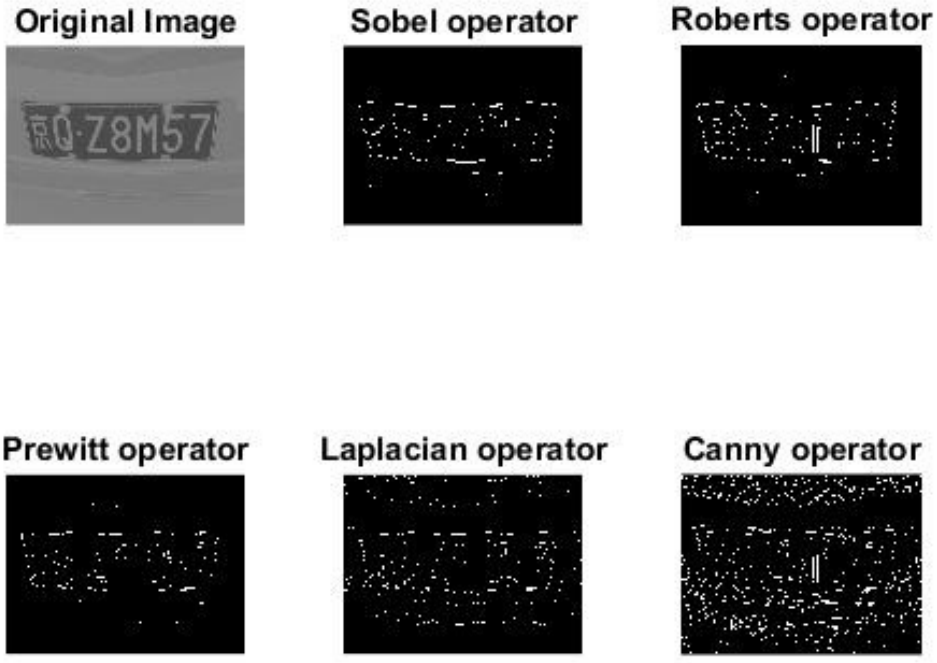
$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Figure 4, Laplacian[5]

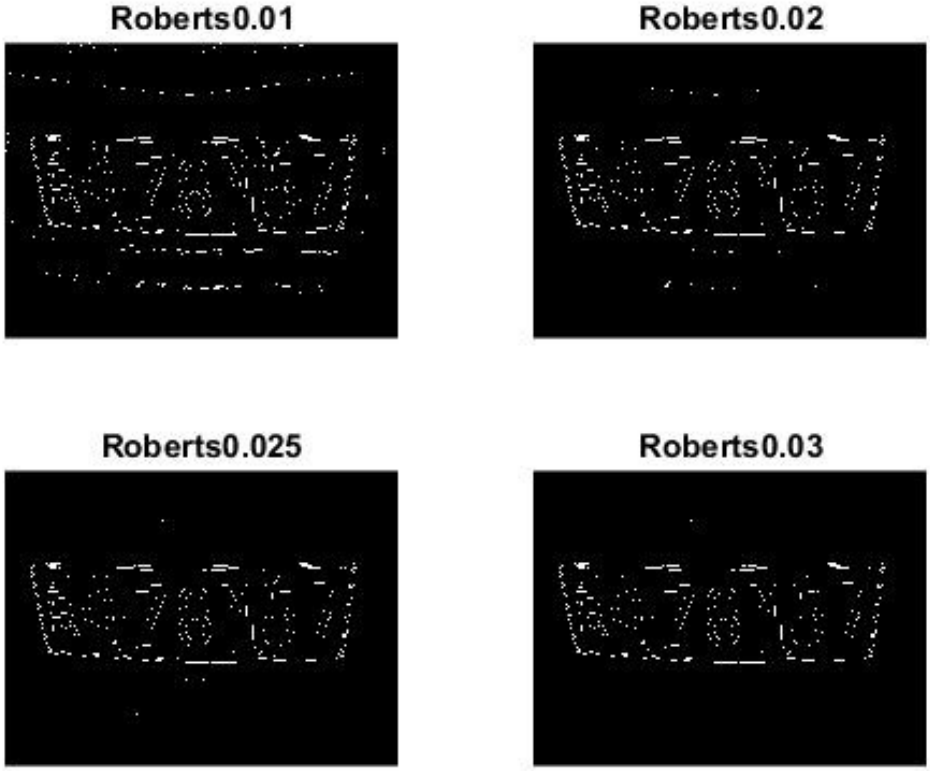
(5)Canny

Canny is kind of mixture of the method above. It has a strong capability to detect weak edges, but it is too sensitive that some who is not edge would be considered as edges.

To see how well these method apply to the car plate recognition. A little test was established, the result is shown as below.

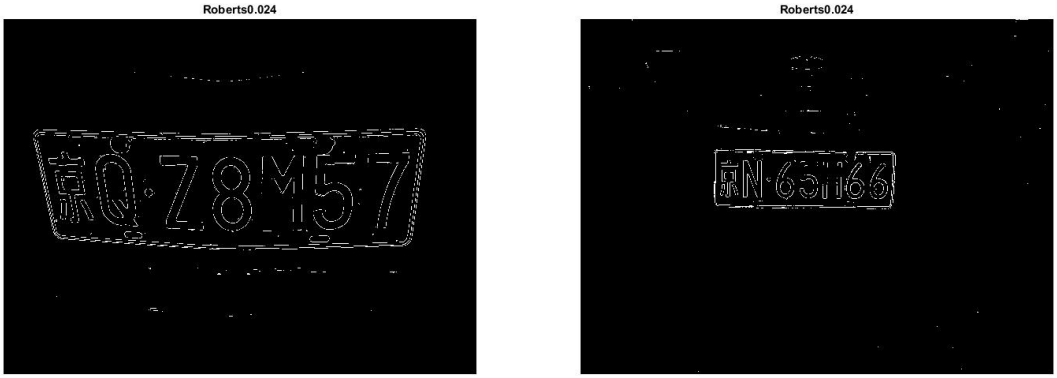


From the test, we can seen that Laplacian and Canny create too many edges with the background which is bad for car plate locating. In the other three operator, Roberts display more edge detail within the car plate which would be better serve for erode and dilate algorithm in car plate locating. The parameter of Roberts is also being tested and compare.



0.01 parameter display many noise around. 0.025 seems to be the best choice, since it not only almost eliminate all the noise, but also preserve the as much details within. The code and result is shown below:

```
%Edging
IR3 = edge(B,'roberts',0.024);
figure(3),imshow(IR3);title(‘Roberts0.024’);
```



4.3. Morphology Processing

Morphology process is an image disposal method based on mathematical morphology set theory. The morphology basically consists of four algorithm — erosion, dilation, open compound operation and close compound operation.

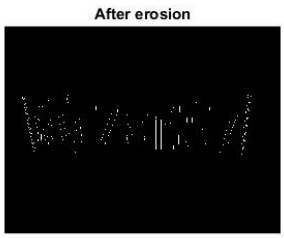
Erosion of Image I by structuring element s is given by I⊕b. The structuring element s is positioned with its origin at (x,y) and the new pixel value is determined using the rule:

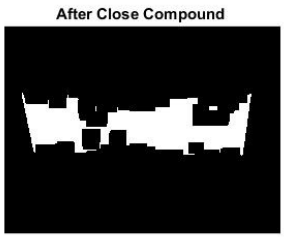
$$G(x,y) = \begin{cases} 1 \text{ if s fits f} \\ \\ 0 \text{ otherwise} \end{cases}$$

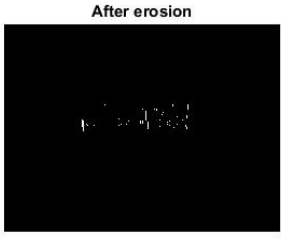
Dilation uses the reverse rule above. Open compound operation is the algorithm that first use erosion than use the same f for dilation. Close compound operation is the algorithm that first use dilation than use the same f for erosion. f is the modal of the morphology process.

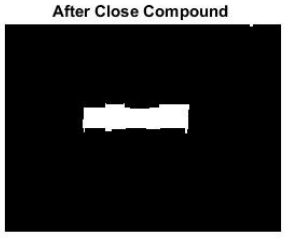
For vehicle plate recognition, I choose to first erode the image with the model of [1;1;1] to eliminate most of the noise. Then I use the close compound operation with two model — 50×50 rectangle and 20 pixel line. Using the rectangle can let discrete parts of car plate united, and further eliminate the other left noises. The reason that I also use the 20 pixel line model is to let the car plate connected into one domain, avoiding separation. After this process, I eliminate the object that is smaller than 3200 pixel. In the test, I found out some of the larger noise is unable to be deleted, because it is bigger than 3200 pixel. Therefore, I add a connecting domain algorithm to found out the biggest connected domain in the image and delete the rest. The codes and results are shown below:

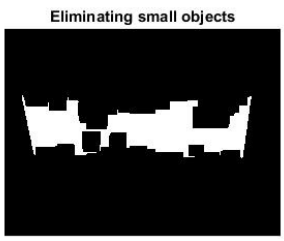
```
%Morphology
se = [1;1;1];   %set the model
I3 = imerode(IR3,se);   %erosion
figure(4),subplot(2,2,1)
imshow(I3);title('After erosion');

se = strel('rectangle',[50,50]);
I4 = imdilate(I3,se);
se = strel('line',20,0);
I4 = imdilate(I4,se);
se = strel('rectangle',[50,50]);
I4 = imerode(I4,se);
se = strel('line',20,0);
I4 = imerode(I4,se);    %Close Compound
figure(4),subplot(2,2,2)
imshow(I4);title('After Close Compound ');

I5 = bwareaopen(I4,3200);   %Eliminating small objects
figure(4),subplot(2,2,3)
imshow(I5);title('Eliminating small objects');

stats = regionprops(I5,'Area'); % Forming Connecting domain matrix
cellstats = struct2cell(stats); %Transform in to cell mode
matstats = cell2mat(cellstats);
maxregion = max(matstats); %Finding maximum
I6 = bwareaopen(I5,maxregion);
figure(4),subplot(2,2,4)
imshow(I6);title('Eliminating region other than the biggest domain')
```

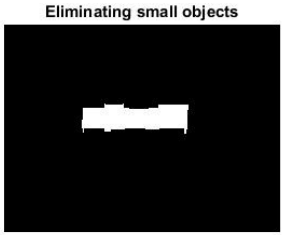


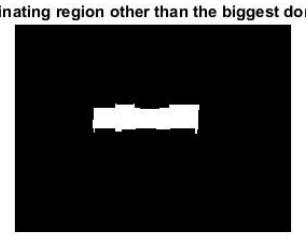












5. Vehicle License Segregation

Vehicle License Segregation is the process to cut out the car plate image. Since we have already locate the car plate, what we are going to do is to find the up, down, left and right coordinate and use these to cut out the car plate. There are several ways to achieve that, the easiest way is to traverse the matrix from four side until the met a “1” value which means it reach the white pixel and record the coordinate of them. But from my experiment I found out that this process requires millions time of calculation that MATLAB would break down. Therefore, I use another way which will severely decrease the times of calculation and dramatically increase the processing time.

First, I build a matrix to record the amount of white pixel in each line and a matrix recording the amount of white pixel in each line. Using the max function, I can easily find the maximum value of line and column. After locating the maximum value line, I use a while function to traverse up and down the pixel amount matrix until it reaches zero, then I get the upper and lower boundary. Using the same way, I can get the left and right boundary. This method dramatically decrease the amount of calculation, since it only traverse a matrix that is [x,1] and [1,y], but not [x,y]. The codes and results are as below:

```
%Vehicle License Segregation

[x,y] = size(I6);
I6d = double(I6);

BlueX = zeros(x,1);
 for i=1:x
    for j=1:y
       if(I6d(i,j,1)==1)
          BlueX(i,1) = BlueX(i,1)+1;%Record the amount of pixel in every line
       end
     end
 end
```

```
[temp MaxX] = max(BlueX);%Max amount of pixel line
up = MaxX;
while ((BlueX(up,1)>=1) && (up>1))
      up = up-1;%upper boundary
end
bottom=MaxX;
while ((BlueX(bottom,1)>=1)&&(bottom<x))
      bottom = bottom+1;%lower boundary
end
IX = I(up:bottom,:,:);
figure(5),subplot(1,2,1)
imshow(IX);title('Y-axis Cutting');

BlueY=zeros(1,y);
for j=1:y
   for i=up:bottom
      if(I6d(i,j,1)==1)
         BlueY(1,j) = BlueY(1,j)+1;  %Record the amount of pixel in every column
      end
   end
end

left = 1;
while(BlueY(1,left)<1 && left<y)
   left = left + 1;
end
right = y;
while(BlueY(1,right)<1 && right>left)
   right = right - 1;
end
IY = I(:,left:right,:);
figure(5),subplot(1,2,2)
imshow(IY);title('X-axis Cutting');

chepai = I(up:bottom,left:right,:);
figure(6),imshow(chepai);title('Vehicle plate')
```

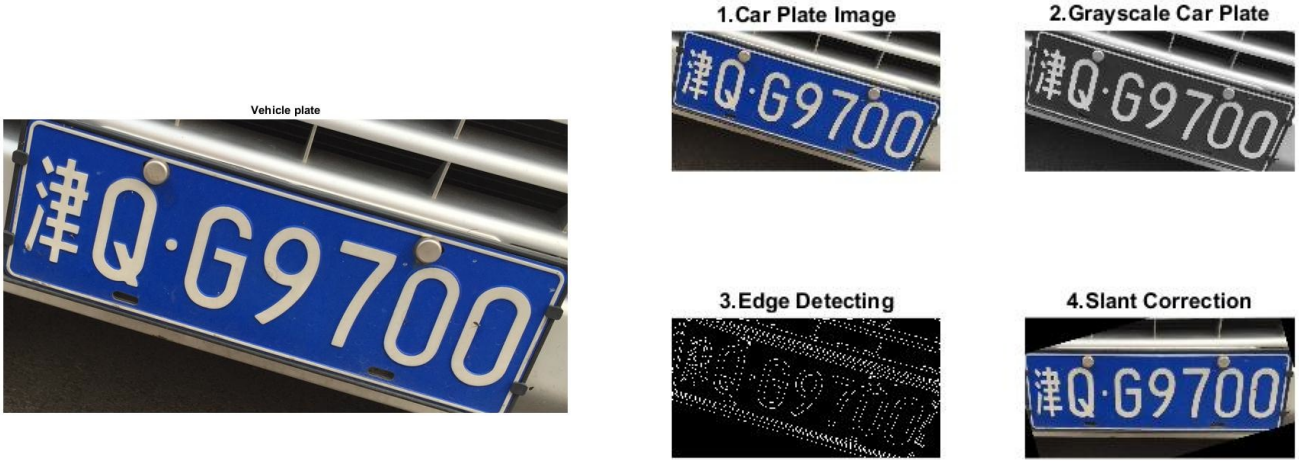
6. License Plate Preprocessing

6.1. Angle Adjustment

For angle adjustment, I use the most popular radon transform. The radon function computes the line integrals from multiple sources along parallel paths, or beams, in a certain direction. The beams are spaced 1 pixel unit apart. To represent an image, the radon function takes multiple, parallel-beam projections of the image from different angles by rotating the source around the center of the image. The following figure shows a single projection at a specified rotation angle.[6] The codes and results are as below.

```
%Angle Adjustment
imwrite(chepai,'chepai.jpg');
a=imread('chepai.jpg');
figure(7),subplot(2,2,1),imshow(a),title('1.Car Plate Image');
gray=rgb2gray(a);
figure(7),subplot(2,2,2),imshow(gray);title('2.Grayscale Car Plate');
bw=edge(gray,'canny');
figure(7),subplot(2,2,3),imshow(bw);title('3.Edge Detecting');
theta=1:180;
[R,xp]=radon(bw,theta);
[I0,J]=find(R>=max(max(R)));%Record the angle
JD=90-J;
b=imrotate(a,JD,'bilinear','crop');
figure(7),subplot(2,2,4),imshow(b);title('4.Slant Correction');
chepaigray = rgb2gray(b);
imwrite(chepaigray,'chepaigray.jpg');
```

[6] http://cn.mathworks.com/help/images/radon-transform.html



6.2. Binaryzation and Threshold Choosing

The rule of car plate binaryzation is different with the initial image processing. In car plate binaryzation, there are basically only two color — blue and white. By this characteristic, some traditional method like otsu would not be suitable, since the color of the picture is monotonous, the threshold would be too average that some noise would be expose and the character would not be highlighted. Therefore, I choose to custom the threshold by the ratio of maximum and minimum pixel value. I choose to use the percentage low, since the characters are pure white, and others would all be considered noise. A test was established to contrast these methods.

From this test, we can clearly see that the custom threshold is better in this case to eliminate frame and other noise. Therefore, I choose this method to binaryze. The code are as below:

```
% Car Plate Processing
g_max=double(max(max(chepaigray)));
g_min=double(min(min(chepaigray)));
T=round(g_max-((g_max-g_min)*36/100)); % T is threshold
[m,n]=size(chepaigray);
BW=(double(chepaigray)>=T);
figure(8),subplot(2,3,1),imshow(BW);title('1.Binaryzation Image')
```

6.3. Filtering and Noise Elimination

Filtering is a nonlinear smoothing technique for noise elimination. There are basically two kinds of filtering that popular in image processing — median filtering and average filtering. The algorithm uses a windows which can be 3×3, 5×5, 7×7, etc to traverse the image, the middle of every windows will be replace by the median or the average value of the windows. This method would significantly eliminant pixels that are not fit with its surrounding, since their intensity difference is too large. In Vehicle License Recognition, I choose to use median filter by 3×3 model, since the difference between noise and the background of car plate is big, which makes median filter more fit. After that, I use the connecting domain algorithm to select the biggest domain and eliminate domains that are smaller than 28% of its area, considering them as big noise. The codes and results are shown below:

```
filtBW = medfilt2(BW,[3,3]);
subplot(2,2,2)
figure(8),subplot(2,3,2),imshow(filtBW);title('2.Median Filtering Image');

[label,num] = bwlabel(filtBW,8);%Eliminating the biggest percentage connected domain
[lm,ln] = size(label);
new_label = zeros(num,1);
for i = 1:lm
   for j = 1:ln
      for k = 1:num
         if label(i,j) == k
            new_label(k) = new_label(k) + 1;
         end
      end
   end
end
```

```
new_img = filtBW;

for i = 1:num
   if new_label(i)<= (max(new_label) * 0.28)
     new_img(label==i) = 0;
   end
end

figure(8),subplot(2,3,3),imshow(new_img);title('3.Percentagewise Delete Noise')
```

6.4. Frame Elimination

Although filtering already fix most of the noise, some big frame would still not be eliminated. I choose to use two step to fix this problem. First, since the frame are all thin line, I choose to use a one way linear open compound operator to first erode them to nothing than dilate the erode part of the characters. Then, I use jump threshold to further eliminate the frames, considering background if the range of the change of black and white is smaller than the threshold. After that using the same method of car plate segregation to narrow the car plate with only characters left. The codes are shown as below:

```
[Nlabel,Nnum] = bwlabel(new_img,8);
if Nnum > 7                %If connected domain less than 7, than operate open compound
   se=[0,0,0,0,0,0,0,1,1,1,1,1,1,1,1];
   new_img = imopen(new_img,se);
   se=[1;1;1;1;1;1;1;1;0;0;0;0;0;0;0];
   new_img = imopen(new_img,se);
```

```
end

figure(8),subplot(2,3,4),imshow(new_img);title('4.First Elimination of Frame')

[label,num] = bwlabel(new_img,8);   %Connecting Domain Elimination
[lm,ln] = size(label);
new_label = zeros(num,1);
for i = 1:lm
   for j = 1:ln
      for k = 1:num
         if label(i,j) == k
            new_label(k) = new_label(k) + 1;
         end
      end
   end
end

for i = 1:num
   if new_label(i)<= (max(new_label) * 0.3)
     new_img(label==i) = 0;
   end
end


d = new_img;
[m,n]=size(d);

y1=10;  % Jump Threshold
for i=1:round(m/4)
   count=0;jump=0;temp=0;
   for j=1:n
      if d(i,j)==1
         temp=1;
      else
         temp=0;
      end
      if temp==jump
         count=count;
      else
         count=count+1;
      end
      jump=temp;
   end
   if count<y1
      d(i,:)=0;
   end
```

```
end

for i=3*round(m/4):m
   count=0;jump=0;temp=0;
   for j=1:n
      if d(i,j)==1
         temp=1;
      else
         temp=0;
      end
      if temp==jump
         count=count;
      else
         count=count+1;
      end
      jump=temp;
   end
   if count<y1
      d(i,:)=0;
   end
end

y2=15;  % Jump Threshold
for i=1:round(n/35)
   count=0;jump=0;temp=0;
   for j=1:m
      if d(j,i)==1
         temp=1;
      else
         temp=0;
      end
      if temp==jump
         count=count;
      else
         count=count+1;
      end
      jump=temp;
   end
   if count<y2
      d(:,i)=0;
   end
end

for i=34*round(n/35):n
   count=0;jump=0;temp=0;
   for j=1:m
```

```
        if d(j,i)==1
            temp=1;
        else
            temp=0;
        end
        if temp==jump
            count=count;
        else
            count=count+1;
        end
        jump=temp;
    end
    if count<y2
        d(:,i)=0;
    end
end

y2=round(n/2); % y2: Threshold
for i=1:round(m/5)
    temp=sum(d(i,:));
    y2=round(2*n/3);
    if temp>y2
         d(i,:)=0;
    end
end

for i=round(4*m/5):m
    temp=sum(d(i,:));
    y2=round(2*n/3);
    if temp>y2
       d(i,:)=0;
    end
end

ii=1;
for i=1:round(m/2)
    if sum(sum(d([i:i+0],:)))==0
        ii=i;
    end
end
d([1:ii],:)=0;
% 下边 1/2
ii=m;
for i=m:-1:round(m/2)
if sum(sum(d([i-0:i],:)))==0
      ii=i;
```

```
end
end
d([ii:m],:)=0;
figure(8),subplot(2,3,5),imshow(d),title('5.Second Frame Elimination');
new_img = d;

[nm,nn] = size(new_img);
top = 1;
bottom = nm;
left = 1;
right = nn;
while sum(new_img(top,:)) == 0 && top <= m
   top = top+1;
end
while sum(new_img(bottom,:)) == 0 && bottom > 1
   bottom = bottom-1;
end
while sum(new_img(:,left)) == 0 && left < n
   left = left+1;
end
while sum(new_img(:,right)) == 0 && right >= 1
   right = right-1;
end
dd = right-left;
hh = bottom-top;
bwcarplate = imcrop(new_img,[left top dd hh]);
figure(8),subplot(2,3,6),imshow(bwcarplate);title('6.Final Car plate');
```

7. Character Segregation

Character Segregation is a relatively easy process. You form a matrix that contain the sum of every column. The value that are not zero means it contain part of the character, and if it is zero, it means that it is the interspace between the characters. Using this feature we can easily

cut apart every characters. This program only deal with english character and numbers, so chinese character will be left out. The codes and results are shown below:

```
function [word,result]=getword(d)
word=[];flag=0;y1=8;y2=0.5;
 while flag==0
    [m,n]=size(d);
    width=0;
    while sum(d(:,width+1))~=0&&width<=n-2
       width=width+1;
    end
    temp=qiege(imcrop(d,[1 1 width m]));
    [m1,n1]=size(temp);
    if width<y1&&n1/m1>y2
       d(:,[1:width])=0;
       if sum(sum(d))~=0
          d=qiege(d);%C
       else word=[];flag=1;
       end
    else
       word=qiege(imcrop(d,[1 1 width m]));
       d(:,[1:width])=0;
       if sum(sum(d))~=0
          d=qiege(d);flag=1;
       else d=[];
       end
    end
 end
 result=d;
```

```
function e=qiege(d)
[m,n]=size(d);
top=1;bottom=m;left=1;right=n; %int
while sum(d(top,:))==0&&top<=m
   top=top+1;
end
while sum(d(bottom,:))==0&&bottom>1
   bottom=bottom-1;
end
while sum(d(:,left))==0&&left<n
   left=left+1;
end
while sum(d(:,right))==0&&right>=1
   right=right-1;
end
dd=right-left;
```

```
hh=bottom-top;
e=imcrop(d,[left top dd hh]);
```

```
%Character Segregation
[Clabel,Cnum] = bwlabel(bwcarplate,8);

%Eliminate Chinese Character
if Cnum == 7
   [zz,bwcarplate] = getword(bwcarplate);
   [aa,bwcarplate] = getword(bwcarplate);
   [bb,bwcarplate] = getword(bwcarplate);
   [cc,bwcarplate] = getword(bwcarplate);
   [dd,bwcarplate] = getword(bwcarplate);
   [ee,bwcarplate] = getword(bwcarplate);
   [ff,bwcarplate] = getword(bwcarplate);
   figure(9),subplot(1,7,2),imshow(aa);
   figure(9),subplot(1,7,3),imshow(bb);
   figure(9),subplot(1,7,4),imshow(cc);
   figure(9),subplot(1,7,5),imshow(dd);
   figure(9),subplot(1,7,6),imshow(ee);
   figure(9),subplot(1,7,7),imshow(ff);
end
if Cnum == 6
   [aa,bwcarplate] = getword(bwcarplate);
   [bb,bwcarplate] = getword(bwcarplate);
   [cc,bwcarplate] = getword(bwcarplate);
   [dd,bwcarplate] = getword(bwcarplate);
   [ee,bwcarplate] = getword(bwcarplate);
   [ff,bwcarplate] = getword(bwcarplate);
   figure(9),subplot(1,7,1),imshow(aa);
   figure(9),subplot(1,7,2),imshow(bb);
   figure(9),subplot(1,7,3),imshow(cc);
   figure(9),subplot(1,7,4),imshow(dd);
   figure(9),subplot(1,7,5),imshow(ee);
   figure(9),subplot(1,7,6),imshow(ff);
end
```

8. OCR(Optical Character Recognition)

8.1. Pixel Comparison

Pixel comparison is one of the algorithms that use in OCR. The concept of OCR is pretty self-explanatory. It is basically the algorithm that aim to recognize and digitize the character on a image. For pixel comparison, first you need to prepare a train set which include some samples of characters as figure 1.

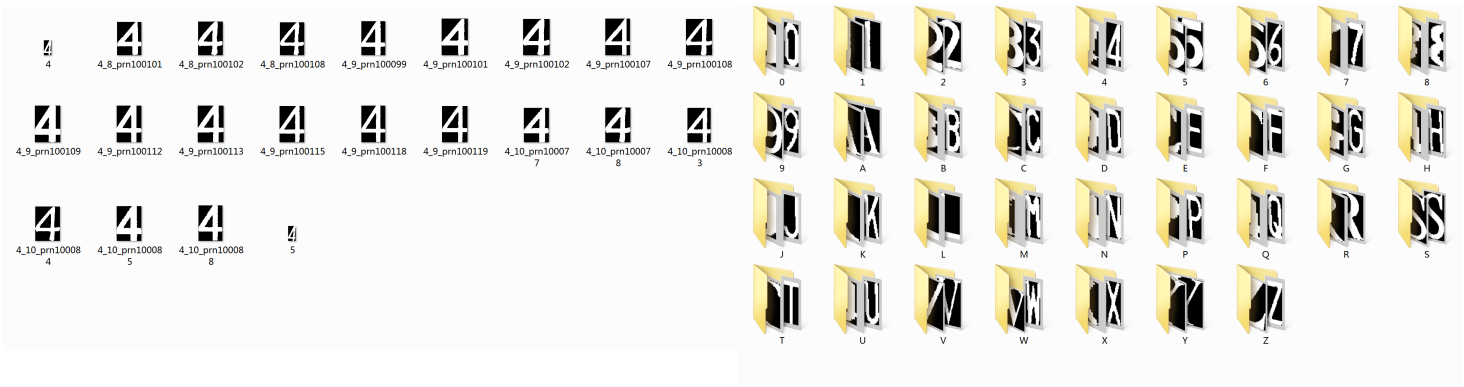


Figure 1

The basic principle of this algorithm is to compare the character that we eliminate from the car plate to the ones in train set for every pixel and see which folder of train set have the highest match up rate with the character. First, we extract and uniform these pictures in folders in MATLAB and turn them into 34 matrixes(each folder) with sub matrixes represent each train image in the folders. Then, we traverse all the pixel in the car plate character and compare to all the image in the train set, then record the similarity rate within each folder. The folder that have the highest match-up rate would be considered as the result. However, this algorithm have a really obvious disadvantage which is its low efficiency. The calculation in the algorithm for the computer is too much which will make the whole process very slow. Also, it requires many image in the train set, making the whole program bigger. Nonetheless, it is still a considerable method, because with the increasing amount of train image, the

recognition rate will also increase. The codes are shown as below. The characters that eliminate from the car plate are save into a folder called Testset.

```
trainpath = 'Trainset\';
folder = dir(trainpath);
trainfea = cell(34,1);
for i = 3:length(folder)
    subdir = [trainpath '\' folder(i).name];
    subfolder = dir([subdir '\*.bmp']);
    norm_sizex = 40;
    norm_sizey = 20;
    features = zeros(length(subfolder),norm_sizex*norm_sizey);
    for j = 1:length(subfolder)
        filepath = [subdir '\' subfolder(j).name];
        img = imread(filepath);
        norm_img = imresize(img,[norm_sizex,norm_sizey]);
        fea = double(norm_img(:));
        features(j,:) = fea;
    end
    trainfea{i-2,1} = features;
end

testpath = 'Testset\';
res = zeros(1,6);
folder = dir(testpath);
correct = 0;
test_num = 0;
for i=3:length(folder)
    subdir = [testpath '\' folder(i).name];
    subfolder = dir([subdir '\*.bmp']);
    norm_sizex = 40;
    norm_sizey = 20;
    for j = 1:length(subfolder)
        test_num = test_num + 1;
        filepath = [subdir '\' subfolder(j).name];
        img = imread(filepath);
        norm_img = imresize(img,[norm_sizex,norm_sizey]);
        fea = double(norm_img(:));

        result = Recognize(fea,trainfea);
        res(i) = result;
        if result == i-2
            correct = correct + 1;
        end
```

```
    end
end
accuracy = correct/test_num;
final = zeros(1,8);
for i = 1:8
  if res(i) == 1
     final(i) = '0';
  end
  if res(i) == 2
     final(i) = '1';
  end
  if res(i) == 3
     final(i) = '2';
  end
  if res(i) == 4
     final(i) = '3';
  end
  if res(i) == 5
     final(i) = '4';
  end
  if res(i) == 6
     final(i) = '5';
  end
  if res(i) == 7
     final(i) = '6';
  end
  if res(i) == 8
     final(i) = '7';
  end
  if res(i) == 9
     final(i) = '8';
  end
  if res(i) == 10
     final(i) = '9';
  end
  if res(i) == 11
     final(i) = 'A';
  end
  if res(i) == 12
     final(i) = 'B';
  end
  if res(i) == 13
     final(i) = 'C';
  end
  if res(i) == 14
     final(i) = 'D';
```

```
end
if res(i) == 15
    final(i) = 'E';
end
if res(i) == 16
    final(i) = 'F';
end
if res(i) == 17
    final(i) = 'G';
end
if res(i) == 18
    final(i) = 'H';
end
if res(i) == 19
    final(i) = 'J';
end
if res(i) == 20
    final(i) = 'K';
end
if res(i) == 21
    final(i) = 'L';
end
if res(i) == 22
    final(i) = 'M';
end
if res(i) == 23
    final(i) = 'N';
end
if res(i) == 24
    final(i) = 'P';
end
if res(i) == 25
    final(i) = 'Q';
end
if res(i) == 26
    final(i) = 'R';
end
if res(i) == 27
    final(i) = 'S';
end
if res(i) == 28
    final(i) = 'T';
end
if res(i) == 29
    final(i) = 'U';
end
```

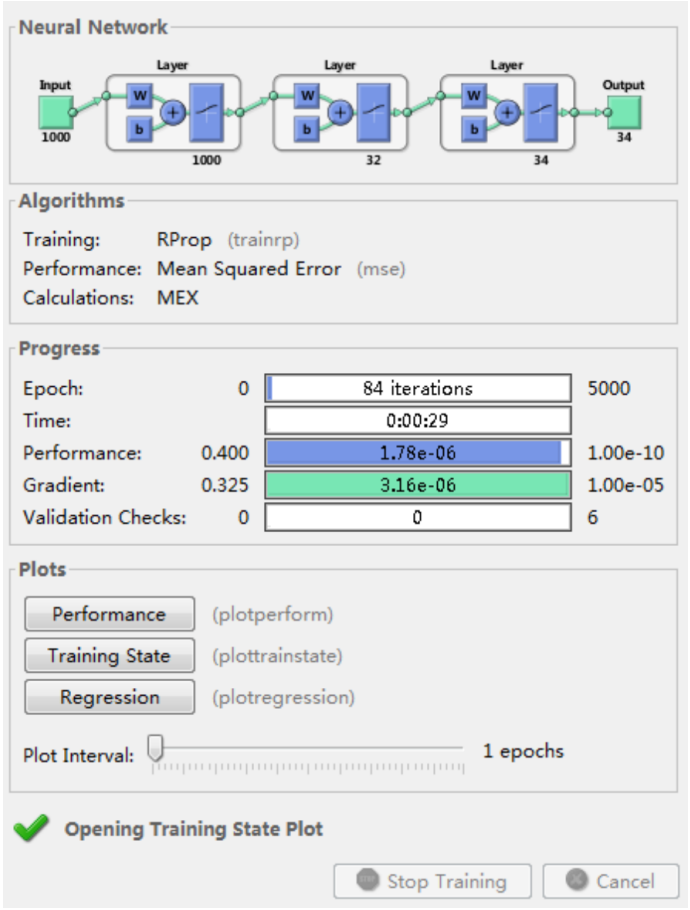
```
  if res(i) == 30
     final(i) = 'V';
  end
  if res(i) == 31
     final(i) = 'W';
  end
  if res(i) == 32
     final(i) = 'X';
  end
  if res(i) == 33
     final(i) = 'Y';
  end
  if res(i) == 34
     final(i) = 'Z';
  end
end
final = char(final);
final
```

8.2. Neural Network Algorithm

Neural networks is a connected networks, composed of a wide neural unit. It mimic the way our brain work, but it is just a metaphor. It basically divided into two stages — learning phase and working phase.

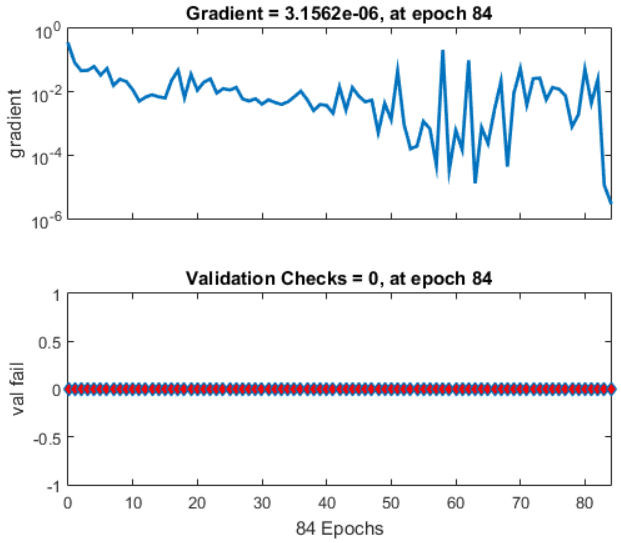
For the learning phase we use the algorithm with called BP(Back Propagation). BP is considered as a supervised learning algorithm. Its learning principle is based on steepest descent method. It can adjust the weight and threshold, making the sum of the squared errors minimized. BP neural network's model of topological structures include input layer, hidden layer and output layer. In MATLAB, using the newff function, we can easily train an BP network. First, we build a train set with images of different character as 8.1 shown. Then, we uniform each image and turn it into a line matrix, and combine them into a matrix P, recording the features of each character. After that, the matrix is input into a newff function and we adjust the parameter of three layers. Running the program , we can get a .net file which record the training features result. The code are shown below:

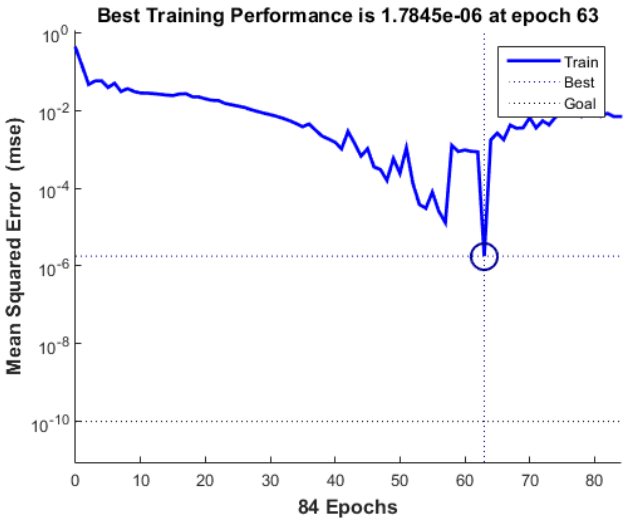
```
I0=pretreatment(imread('Trainset\0.bmp'));
I1=pretreatment(imread('Trainset\1.bmp'));
I2=pretreatment(imread('Trainset\2.bmp'));
I3=pretreatment(imread('Trainset\3.bmp'));
I4=pretreatment(imread('Trainset\4.bmp'));
I5=pretreatment(imread('Trainset\5.bmp'));
I6=pretreatment(imread('Trainset\6.bmp'));
I7=pretreatment(imread('Trainset\7.bmp'));
I8=pretreatment(imread('Trainset\8.bmp'));
I9=pretreatment(imread('Trainset\9.bmp'));
I10=pretreatment(imread('Trainset\A.bmp'));
I11=pretreatment(imread('Trainset\B.bmp'));
I12=pretreatment(imread('Trainset\C.bmp'));
I13=pretreatment(imread('Trainset\D.bmp'));
I14=pretreatment(imread('Trainset\E.bmp'));
I15=pretreatment(imread('Trainset\F.bmp'));
I16=pretreatment(imread('Trainset\G.bmp'));
I17=pretreatment(imread('Trainset\H.bmp'));
I18=pretreatment(imread('Trainset\J.bmp'));
I19=pretreatment(imread('Trainset\K.bmp'));
I20=pretreatment(imread('Trainset\L.bmp'));
I21=pretreatment(imread('Trainset\M.bmp'));
I22=pretreatment(imread('Trainset\N.bmp'));
I23=pretreatment(imread('Trainset\P.bmp'));
I24=pretreatment(imread('Trainset\Q.bmp'));
I25=pretreatment(imread('Trainset\R.bmp'));
I26=pretreatment(imread('Trainset\S.bmp'));
I27=pretreatment(imread('Trainset\T.bmp'));
I28=pretreatment(imread('Trainset\U.bmp'));
I29=pretreatment(imread('Trainset\V.bmp'));
I30=pretreatment(imread('Trainset\W.bmp'));
I31=pretreatment(imread('Trainset\X.bmp'));
I32=pretreatment(imread('Trainset\Y.bmp'));
I33=pretreatment(imread('Trainset\Z.bmp'));
```




```
P=[I0',I1',I2',I3',I4',I5',I6',I7',I8',I9',I10',I11',I12',I13',I14',I15',I16',I17',I18',I19',I20',I21',I22',I
23',I24',I25',I26',I27',I28',I29',I30',I31',I32',I33'];

T=eye(34,34);

net=newff(minmax(P),[1000,32,34],{'logsig','logsig','logsig'},'trainrp');
net.inputWeights{1,1}.initFcn ='randnr';
net.layerWeights{2,1}.initFcn ='randnr';
net.trainparam.epochs=5000;
net.trainparam.show=50;
%net.trainparam.lr=0.003;
```

```
net.trainparam.goal=0.0000000001;
net=init(net);

[net,tr]=train(net,P,T);
save('ym','net');
```





We than load the .net file to our main program to recognize the characters, which we needs to first turn the character's image into line matrixes to match the training. The codes are shown below:

```
function  inpt  = pretreatment(I)
%YUCHULI Summary of this function goes here
%   Detailed explanation goes here
if ndims(I)==3
  I1 = rgb2gray(I);
else
  I1=I;
end
I1=imresize(I1,[50 20]);%½«Í¼Æ¬Í³Ò»»®Îª50*20´óÐ¡
I1=im2bw(I1,0.9);
[m,n]=size(I1);
inpt=zeros(1,m*n);
%%%%%½«Í¼Ïñ°´ÁÐ×ª»»³ÉÒ»¸öÐÐÏòÁ¿
for j=1:n
   for i=1:m
      inpt(1,m*(j-1)+i)=I1(i,j);
   end
end
```

```
PIN1=pretreatment(aa);
PIN2=pretreatment(bb);
PIN3=pretreatment(cc);
PIN4=pretreatment(dd);
PIN5=pretreatment(ee);
PIN6=pretreatment(ff);
P0=[PIN1',PIN2',PIN3',PIN4',PIN5',PIN6'];
for i=1:6
  T0= sim(net ,P0(:,i));
  T1 = compet (T0) ;
  d =find(T1 == 1) - 1
 if (d==10)
    str='A';
 elseif (d==11)
     str='B';
 elseif (d==12)
     str='C';
 elseif (d==13)
     str='D';
 elseif (d==14)
     str='E';
 elseif (d==15)
     str='F';
 elseif (d==16)
     str='G';
 elseif (d==17)
     str='H';
 elseif (d==18)
     str='J';
      elseif (d==19)
     str='K';
      elseif (d==20)
     str='L';
      elseif (d==21)
     str='M';
      elseif (d==22)
     str='N';
      elseif (d==23)
     str='P';
      elseif (d==24)
     str='Q';
      elseif (d==25)
     str='R';
          elseif (d==26)
```

```
    str='S';
        elseif (d==27)
    str='T';
        elseif (d==28)
    str='U';
        elseif (d==29)
    str='V';
        elseif (d==30)
    str='W';
        elseif (d==31)
    str='X';
        elseif (d==32)
    str='Y';
     elseif (d==33)
    str='Z';

 else
   str=num2str(d);
 end
 switch i
    case 1
        str1=str;
    case 2
        str2=str;
    case 3
        str3=str;
    case 4
        str4=str;
    case 5
        str5=str;
    otherwise
        str6=str;
  end
end

s=strcat(str1,str2,str3,str4,str5,str6);
figure(10);
imshow('chepai.jpg'),title(s);
```

9. Conclusion

The whole program is being tested with 100 pictures that I randomly taken on the street with different size, angle and background. The car plate detection precision rate is 97%. The character segregation precision rate is 94.8%, and the recognition rate is 91.2%. Overall, a precise car plate recognition program was made.

10. Bibliography

1. Chao Tian(田, 超). lmage Recognition and Processing of Vehicle Number Plate auto Detection in Transportation System (汽车车牌自动检测的图像识别和处理）. Thesis. Xi`an Jiaotong University(西安科技大学), 2008. Print.

2. Zhou Kewei(周, 科伟). Car Plate Recognition based on Neuron Network in Matlab(环境下基于神经网络的车牌识别). Thesis. Xi`an Jiaotong University(西安电子科技大学), 2009. Print.

3. K Fukanaga, L. D.Hostetler, "The estimation of the gradient of a density function with applications in pattern recognition," IEEE Trans Information Theory, 1975, 21

4. Zhongli Xu(徐, 中立). Intelligent Information Processing of Digital image(数字图像的智能信息处理). Beijing: National Defend Industry Press(国防工业出版社), 2001. Print.

5. S.Haykin,Neural networks-A comprehensive foudation, 2nd ed,Prentice Hall,Englewood cliffs,NJ,1999.

6. Liu Jing(刘, 静). The Comparison of Several Kinds of License Plate Character Recognition algorithm(几种车牌字符识别算法的比较). Thesis. 2008. Computer & Telecommunication(电脑与电信), 2008. Print.

7. Luo Jianjun(罗建军)，MATLAB Tutorial(MATLAB教程). Beijing(北京)：Electronic Industry Press(电子工业出版社)，2005

11. Appendix

11.1 Reflection

This project gave me a sense of how image recognition work and its great potential in the future artificial intelligence industry. I learned lots of image disposal algorithm in the process. The image recognition algorithm are amazing! We, humans, evolved millions of years to form the physical characteristics we now possess, owed to the best engineering in the world, nature. However, the invention of AI technology has allowed us to simulate ourselves, the product of the nature, through numbers. Image recognition algorithms mimic our eyes, allowing us bring more convenience in our life. My interests with image recognition and artificial intelligence will lead me to a further research on these fields.