ABSTRACT: -

In this Project, we are going to describe the usage and numerous advantages of license plate recognition in daily life. It has a great scope, as it can be used in tollbooth for automated recognition of license plate using a simple photograph of that vehicle.

This automated recognition can be implemented in car parking, if any vehicle is violating traffic rules then photograph of that vehicle will be taken and by using that photograph we can use our license plate

Recognition method can get the vehicle license plate information and then the concern authority can search in their database and find out the defaulters.

There are considered an approach to identify vehicle through recognizing of it license plate using image fusion, neural networks and threshold techniques as well as some experimental results to recognize the license plate successfully

INTRODUCTION: -

We intend to create an automated system by using MATLAB which is going to detect and recognize the license plate of the car. Moreover, we want to contribute in the Smart City initiative

Basically we are taking an image from environment and processing is a very important part and it will be done by using certain algorithm in MATLAB and we will show the output as a string or a group of characters.

There are so many examples like traffic monitoring, tracking stolen cars, managing parking toll, red-light violation enforcement, border and customs checkpoints. It uses optical character recognition on images to read the license plate of that vehicle.

Existing traffic control system

- Present system is completely a static case
- Vehicles must wait at the intersection for a predefined time until microcontroller switches green light for that lane.
- Exists no process of preemption.
- No green light service for priority based vehicles.
- No alarm/call for emergency – No V2V Communication

From the past decades, management of traffic has been one of the biggest issues of modernization.

Researchers have followed a long way to overcome the traffic crises. Right from the very beginning of,

"Manual Traffic Control" in which man power was required to control the traffic. Depending on countries and states the traffic polices are allotted to different areas to control traffic. These men carry sign board, sign light and whistle to control the traffic. They are instructed to wear specific uniforms in order to be easily identified by the drivers. After this came the traditional "Vehicle Actuated Control System" in which, lights are loaded with constant numerical value in the form of timers. The lights are automatically getting ON and OFF depending on timer value changes. The main disadvantage is that the algorithm for this control system does not change the green signal even if the traffic has already passed until the counter is complete, while not taking into account the number of vehicles waiting at red. Hence the density of the traffic does not matter. Next in generation is the "Automatic Traffic Light", which is the modified version of vehicle actuated control system with addition to timers and electrical sensors. In this technique, electronic and electrical sensors are added to detect vehicles and produce signals that the time is being wasted by a green light on an empty road. The established traffic control management systems are inadequate for handling huge amount of traffic load as they are incapable of meeting the growing number of vehicles on road. Drawbacks to these particular controlling methods: -

1. Only skilled operators can make suitable judgment and decisions because sometimes the situation is very complicated and many factors are needed to be considered.
2. The operator is under very high work load as he has to continuously take decisions and review the traffic conditions at small intervals of time.

WORKING PRINCIPLE:

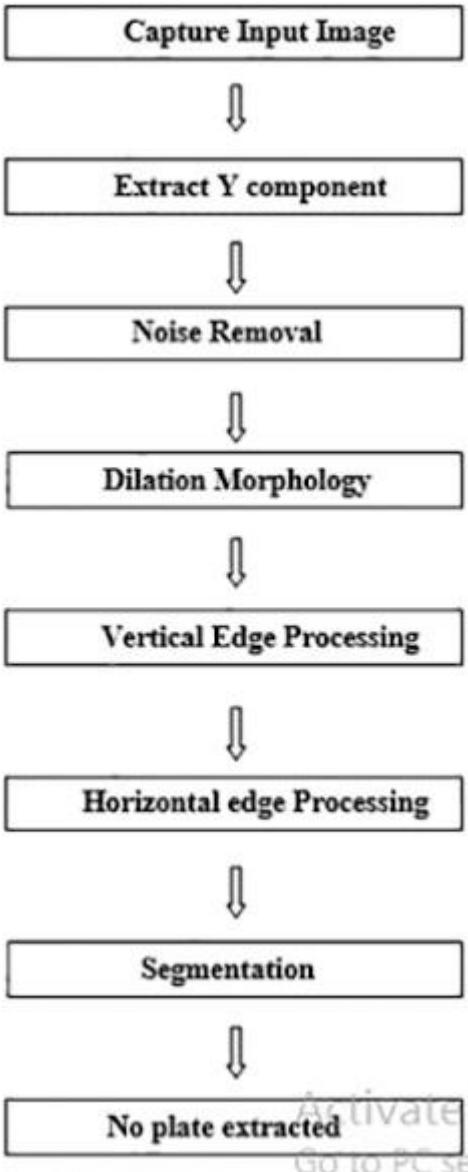
- ❖ Firstly, we need an image of a vehicle in order to perform the following tasks.
- ❖ Edge enhancement followed by morphological operators and finally extracting the plate region. ❖

Finally, by the help of optical character recognition we can get the numbers or string in text file.

REQUIREMENTS:

- ❖ MATLAB, WINDOWS 10 OS

Flow Chart:



CODE PROCESSING:-

Original Image


```
[FileName,PathName] = uigetfile('*.jpg;*.png;*.gif;*.tif','Select a image');
I=imread(strcat(PathName,FileName));
figure(1);
```

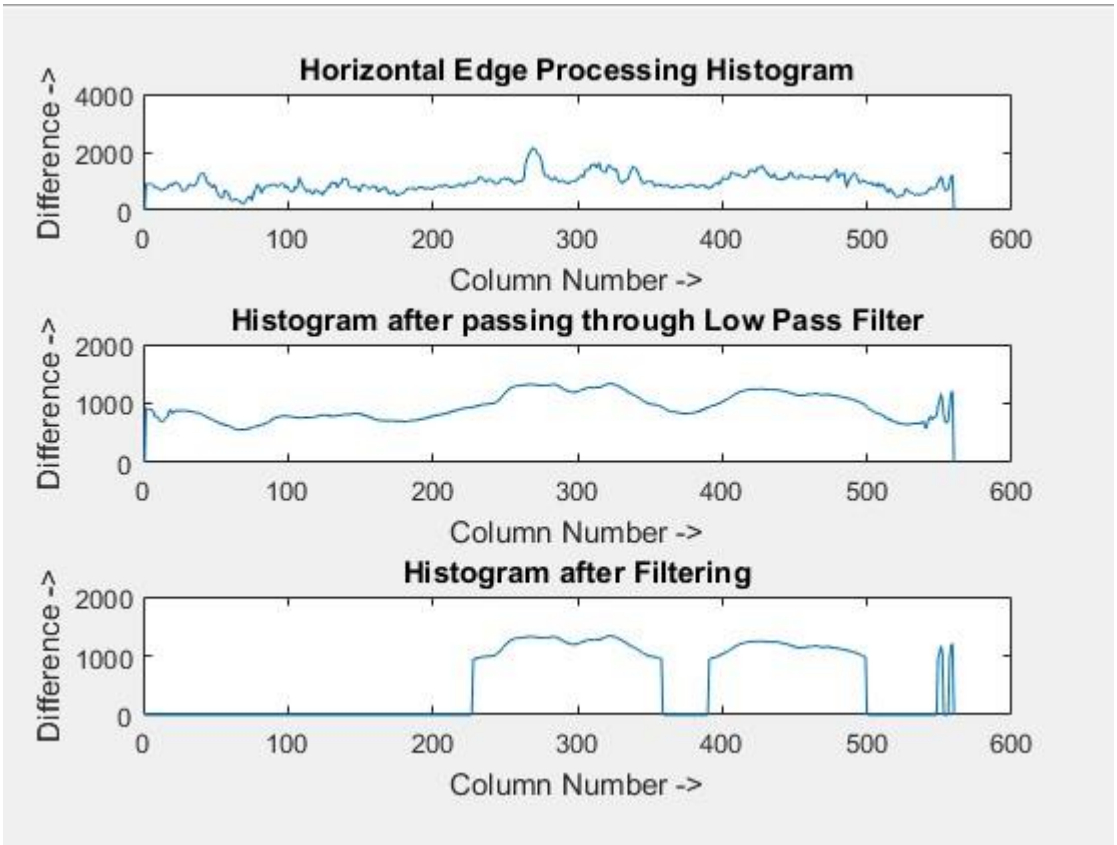

```
imshow(I);
% Extract Y component (Convert an Image to Gray)
Igray = rgb2gray(I);
[rows cols] = size(Igray);
%% Dilate and Erode Image in order to remove noise
Idilate = Igray; for i = 1:rows for j
= 2:cols-1 temp = max(Igray(i,j-1),
```

```
Igray(i,j)); Idilate(i,j) = max(temp,
Igray(i,j+1)); end end I = Idilate;
figure(2);
```


```
imshow(Igray); figure(3);
```

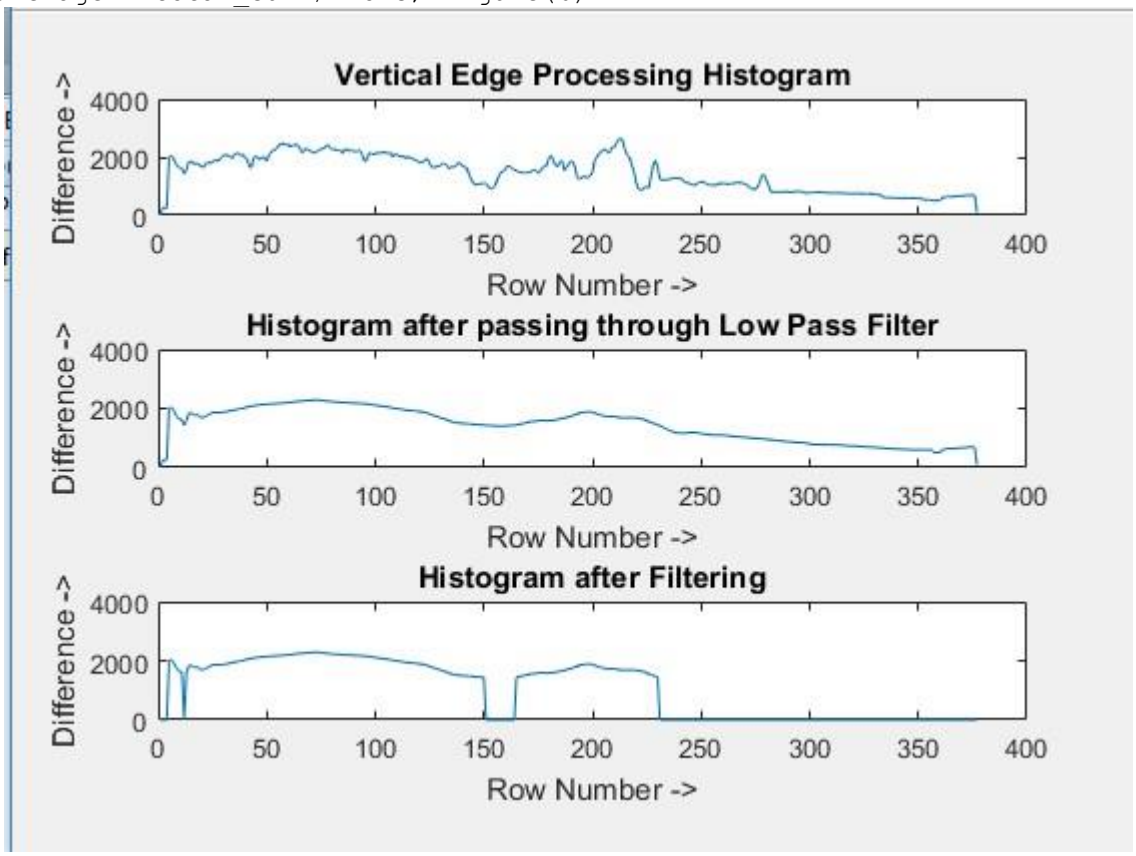

```
title('Dilated Image')
imshow(Idilate); figure(4);
```


```
imshow(I);
difference = 0;
sum = 0; total_sum
= 0;
difference = uint32(difference);
%% PROCESS EDGES IN HORIZONTAL DIRECTION
disp('Processing Edges Horizontally...');
max_horz = 0; maximum = 0; for i =
2:cols sum = 0; for j = 2:rows if(I(j,
i) > I(j-1, i)) difference = uint32(I(j,
i) - I(j-1, i)); else
difference = uint32(I(j-1, i) - I(j, i));
end
if(difference > 20)
sum = sum + difference;
end end horz1(i) =
sum; % Find Peak Value
if(sum > maximum)
max_horz = i; maximum
= sum; end
total_sum = total_sum + sum; end
average = total_sum / cols; figure(5);
```



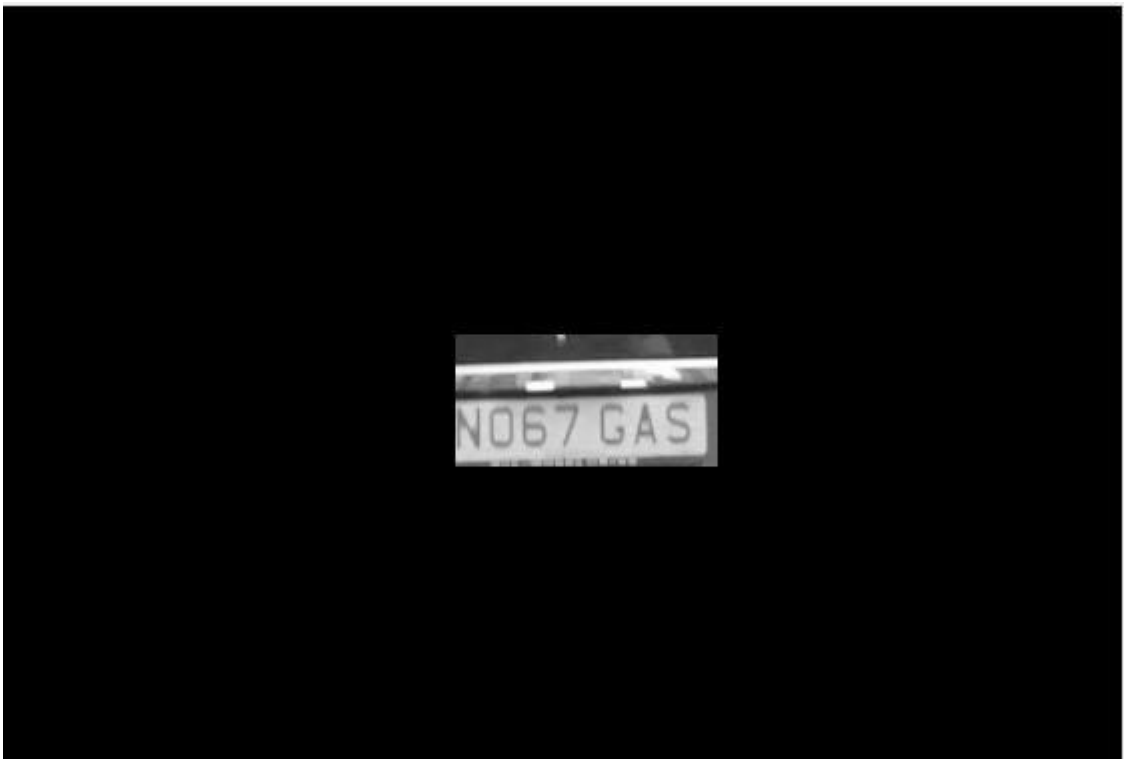

```
subplot(3,1,1); plot
(horz1);
title('Horizontal Edge Processing Histogram');
xlabel('Column Number ->'); ylabel('Difference
->');
%% Smoothen the Horizontal Histogram by applying Low Pass Filter sum
= 0;
horz = horz1; for i
= 21:(cols-21) sum =
0; for j = (i-
20):(i+20) sum = sum
+ horz1(j); end
horz(i) = sum / 41;
end subplot(3,1,2);
plot (horz);
title('Histogram after passing through Low Pass Filter');
xlabel('Column Number ->'); ylabel('Difference ->');
%% Filter out Horizontal Histogram Values by applying Dynamic Threshold
disp('Filter out Horizontal Histogram...'); for i = 1:cols if(horz(i)
< average) horz(i) = 0; for j = 1:rows I(j, i) = 0; end end end
subplot(3,1,3); plot (horz);
title('Histogram after Filtering');
xlabel('Column Number ->'); ylabel('Difference
->');
%% PROCESS EDGES IN VERTICAL DIRECTION
difference = 0; total_sum = 0;
difference = uint32(difference);
disp('Processing Edges Vertically...');
maximum = 0; max_vert = 0; for i =
2:rows sum = 0; for j = 2:cols %cols
if(I(i, j) > I(i, j-1)) difference =
uint32(I(i, j) - I(i, j-1)); end
if(I(i, j) <= I(i, j-1)) difference =
uint32(I(i, j-1) - I(i, j)); end
```

```
if(difference > 20)
sum = sum + difference;
end end
vert1(i) = sum;
%% Find Peak in Vertical Histogram
if(sum > maximum) max_vert = i;
maximum = sum; end
total_sum = total_sum + sum; end
average = total_sum / rows; figure(6)
```




```
subplot(3,1,1); plot
(vert1);
title('Vertical Edge Processing Histogram');
xlabel('Row Number ->'); ylabel('Difference
->');
%% Smoothen the Vertical Histogram by applying Low Pass Filter
disp('Passing Vertical Histogram through Low Pass Filter...');
sum = 0; vert = vert1; for i = 21:(rows-21) sum = 0; for j =
(i-20):(i+20) sum = sum + vert1(j); end
vert(i) = sum / 41;
end subplot(3,1,2);
plot (vert);
title('Histogram after passing through Low Pass Filter');
xlabel('Row Number ->'); ylabel('Difference ->');
%% Filter out Vertical Histogram Values by applying Dynamic Threshold
disp('Filter out Vertical Histogram...'); for i = 1:rows if(vert(i)
< average) vert(i) = 0; for j = 1:cols I(i, j) = 0; end end end
subplot(3,1,3); plot (vert);
title('Histogram after Filtering');
xlabel('Row Number ->');
ylabel('Difference ->'); figure(7),
imshow(I);
```


```
j = 1; for i = 2:cols-2 if(horz(i) ~= 0 && horz(i-1) == 0 && horz(i+1) == 0) column(j) =
i; column(j+1) = i; j = j + 2;
elseif((horz(i) ~= 0 && horz(i-1) == 0) || (horz(i) ~= 0 && horz(i+1) == 0))
column(j) = i; j = j+1; end end j = 1; for i = 2:rows-2 if(vert(i) ~= 0 &&
vert(i-1) == 0 && vert(i+1) == 0) row(j) = i; row(j+1) = i; j = j + 2;
elseif((vert(i) ~= 0 && vert(i-1) == 0) || (vert(i) ~= 0 && vert(i+1) == 0))
row(j) = i; j = j+1; end end
[temp column_size] = size (column);
if(mod(column_size, 2))
column(column_size+1) = cols;
end
[temp row_size] = size (row);
if(mod(row_size, 2))
row(row_size+1) = rows; end
%% Region of Interest Extraction
%Check each probable candidate
for i = 1:2:row_size for j =
1:2:column_size
% If it is not the most probable region remove it from image if(~((max_horz >= column(j)
&& max_horz <= column(j+1)) && (max_vert >=row(i) && max_vert <= row(i+1))))
%This loop is only for displaying proper output to User
for m = row(i):row(i+1) for n = column(j):column(j+1)
I(m, n) = 0;
end end end
end end
figure(8), imshow(I); imshow(I);
```



Processing

Comparing proposed technique to an existing technique

We take here a method by Xiaofeng Zhang, Fengchang Xu and Yan Su* as proposed in their paper "Research on Licence Plate Recognition based on MATLAB”

The methodology they used was to first convert the coloured image into grey scale image using histogram equalization, remove the noises through self-adaptive median filter and then to stretch that image to enhance the overall contrast. Then they sharpened using Log-Prewitt operator the image and edges were enhanced using Prewitt edge detection method.

The above procedure is quick but doesn’t give quality results. It might work for some static images but for for real time implementation it is a very bad idea.

Sample processing

Sample Image

License Plate Image of gray

The adaptive median filter of vehicle images

The vehicle images gray stretch

Vehicle images gradient sharpening

Conclusion

As we can observe in above sample that this processing algorithm will give a image that will be full of noise, this type of character recognition can be done by a human eye but not with a normal computer processing system.
In real time, the vehicles will be moving and the images will be blurry and then this type of a processing can make the output even more worse.

Hence the method proposed in this paper is a very reliable method, is quick and give results with high accuracy.