A Project Report

on

Automatic Number Plate Recognition for Indian vehicles

By

DHEERAJ KUMAR MAURYA (20195032)

KULDEEP JAISWAL (20195031)

DEVENDRA KUMAR ATOLIYA (20195138)

Submitted to

Dr. V.K. RAO



Department of

Electronics & Communication Engineering

Motilal Nehru National Institute of Technology Allahabad

Prayagraj – INDIA

ACKNOWLEDGEMENTS

First of all we would like to thank the Almighty for giving us the opportunity to complete this work successfully. Our acknowledgement is meant to express our sincere gratitude to all those people who have been associated with this project and have helped us with it and by sharing their experiences and valuable opinions through which we received the required information crucial for our project. We are thankful to our parents for their relentless support. Most importantly we are grateful to our honorable supervisor who took time out to guide us and provide us with all the necessary materials and sufficient knowledge that was the major requirement. Finally, we express our thanks to our honorable teacher Dr. V.K. Rao for giving us the opportunity to learn the subject in a practical.

Table of Content..................................................................................................................

Chapter 1: Introduction.....................................................................................................1

1.1 Requirements…………………………………………………………………….2

Chapter 2: Fundamentals of image processing……………….……………………..…..3

2.1 RGB format………………………………………………………………………3

2.2 Morphological Image Processing………………………………….…………….4

2.2.1 Morphological Dilation and Erosion……………………………………….4

2.3 Convolution………………………………………………………………………5

2.4 Thinning………………………………………………………………………….5

2.5 Character Recognition using template matching…………….…………………..6

Chapter 3: Methodology...................................................................................................7

3.1 Flow chart……………………………………………………………………..…8

3.2 Grayscale Imaging................................................................................................ 9

3.3 Dilate an Image...................................................................................................... 9

3.4 Horizontal and Vertical Edge Processing of an image………………………..…10

3.5 Passing Histograms through a Low Pass Digital filter..........................................11

3.6 Filtering out Unwanted Regions in an Image........................................................12

3.7 Segmentation…………………………………………………………………..…13

3.8 Region of Interest Extraction…………………………………………………….14

3.9 Matlab…………………………………………………………………………….16

Chapter 4: Outputs and Results.........................................................................................18

4.1 Outputs....................................................................................................................18

4.2 Results.....................................................................................................................21

4.3 Conclusion…………………………………………………………………...……22

Chapter 5: References.......................................................................................................23

Chapter 1: Introduction:

Number plates are used for identification of vehicles all over the nations. Vehicles are identifying either manually or automatically. Automatic vehicle identification is an image processing technique of identify vehicles by their number plates. Automatic vehicle identification systems are used for the purpose of effective traffic control and security applications such as access control to restricted areas and tracking of wanted vehicles. Number plate recognition (NPR) is easier method for Vehicle identification. NPR system for Indian license plate is difficult compared to the foreign license plate as there is no standard followed for the aspect ratio of licence plate. The identification task is challenging because of the nature of the light. Experimentation of number plate detection has been conducted from many years, it is still a challenging task .Number plate detection system investigates an input image to identify some local patches containing license plates. Since a plate can exist anywhere in an image with various sizes, it is infeasible to check every pixel of the image to locate it. In parking, number plates are used to calculate duration of the parking. When a vehicle enters an input gate, number plate is automatically recognized and stored in database. In NPR system spectral analysis approach is used were acquiring the image, extract the region of interest, character segmentation using SVM feature extraction techniques. The advantage of this approach is success full recognition of a moving vehicle

[1].It is difficult to detect the boundary of the Number plate from the input car images in outdoors scene due to colour of characters of the number plate and Background of the Number plate the gradients of the original image is adopted to detect candidate number plate regions.

[2]. There are also algorithms which are based on a combination of morphological operation, segmentation and Canny edge detector. License plate location algorithm consist of steps like as Edge Detection, Morphological operation like dilation and erosion, Smoothing, segmentation of characters and recognition of plate characters

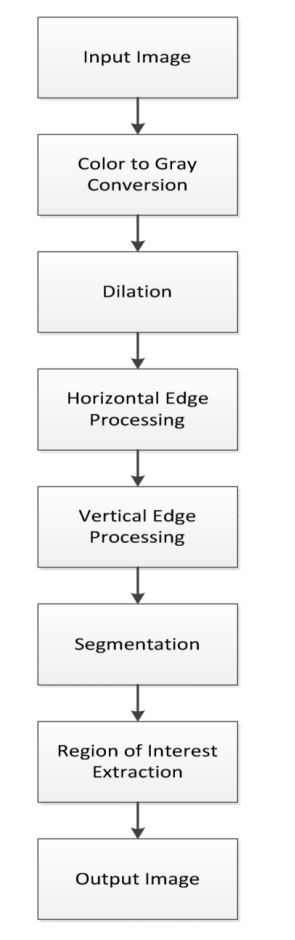


Fig. 1.1 flow chart

Requirements:-

Dataset: Images of Indian vehicles where the number plate is clearly visible



Fig. 1.2 Sample car image.

Programming Software/tool:- Matlab

Chapter 2

FUNDAMENTALS OF IMAGE PROCESSING

An image is used to convey useful information in a visible format. An image is nothing but an arrangement of tiny elements in a two-dimensional plane. These tiny elements are called Pixels. A large number of pixels combine together to form an image, whether small or large. Each pixel represents certain information about the image, like color, light intensity and luminance. A large number of such pixels combine together to form an image. Pixel is the basic element used to describe an image. Mostly, each pixel in an image is represented in either RGB (Red Green Blue) format. In case of an RGB image, all the three components, namely R, G and B combine together to convey information about the color and brightness of a single pixel. Each component consumes certain memory space during image processing.

2.1 RGB Format

In case of an RGB image, each pixel is represented by three different components R, G and B. Each of these components requires at least 8 bits for their storage. In general, a single pixel may require upto 8 \* 3 bits for its storage. An example of a representation of single pixel in RGB format is shown below.

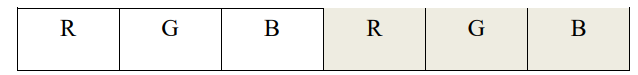


Fig. 2.1 Representation of pixels in RGB format.

The value of R, G and B each ranges from 0-255. A value of (0, 0, 0) represents a black pixel, (255, 0, 0) represents a red pixel and (0, 255, 0) represents a green pixel. So, 8 bits are required to store value for a single component.

2.2 Morphological Image Processing

Binary images may contain numerous imperfections. In particular, the binary regions produced by simple thresholding are distorted by noise and texture. Morphological image processing pursues the goals of removing these imperfections by accounting for the form and structure of the image. These techniques can be extended to greyscale images. Morphological image processing is a collection of non-linear operations related to the shape or morphology of features in an image. Morphological operations rely only on the relative ordering of pixel values, not on their numerical values, and therefore are especially suited to the processing of binary images. Morphological operations can also be applied to greyscale images such that their light transfer functions are unknown and therefore their absolute pixel values are of no or minor interest.

2.2.1 Morphological Dilation and Erosion

The most basic morphological operations are dilation and erosion. Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries. The number of pixels added or removed from the objects in an image depends on the size and shape of the structuring element used to process the image. In the morphological dilation and erosion operations, the state of any given pixel in the output image is determined by applying a rule to the corresponding pixel and its neighbors in the input image. The rule used to process the pixels defines the operation as a dilation or an erosion. This table lists the rules for both dilation and erosion.

2.3 Convolution

Convolution is a simple mathematical operation which is fundamental to many common image processing operators. Convolution provides a way of `multiplying together' two arrays of numbers, generally of different sizes, but of the same dimensionality, to produce a third array of numbers of the same dimensionality. This can be used in image processing to implement operators whose output pixel values are simple linear combinations of certain input pixel values.

In an image processing context, one of the input arrays is normally just a graylevel image. The second array is usually much smaller, and is also two-dimensional (although it may be just a single pixel thick), and is known as the kernel.

2.4 Thinning

Thinning is a morphological operation that is used to remove selected foreground pixels from binary images, somewhat like erosion or opening. It can be used for several applications, but is particularly useful for skeletonization. In this mode it is commonly used to tidy up the output of edge detectors by reducing all lines to single pixel thickness. Thinning is normally only applied to binary images, and produces another binary image as output.

The thinning operation is related to the hit-and-miss transform, and so it is helpful to have an understanding of that operator before reading on.

2.5 Character Recognition using template matching

Template matching or matrix matching, is one of the most common classification methods. Here individual image pixels are used as features. Classification is performed by comparing an input character with a set of templates (or prototypes) from each character class. Each comparison results in a similarity measure between the input characters with a set of templates. If the pixels differ the measure of similarity may be decreased. After all templates have been compared with the observed character image, the character’s identity is assigned the identity of the most similar template.

Chapter 3: Methodology

3.1 Flow chart:

Edge Detection

Noise Reduction

RGB to Grayscale Conversion

Input Car Image

Image pre processing

Extract the region of interest

Media filtering

Plate Localization & Character Segmentation

Number plate text recognition

Pre process region of interest

Number plate de-skewing

Character Recognition

Output Image

The algorithm has three stages, first is pre-processing of a given car image and after that segmentation/ Plate Localization and then Character Recognition and displaying output. Steps of algorithm are as following:-

3.2 Convert a Colored Image into Gray Image :-

The algorithm described here is independent of the type of colors in image and relies mainly on the gray level of an image for processing and extracting the required information. Color components like Red, Green and Blue value are not used throughout this algorithm. So, if the input image is a colored image represented by 3-dimensional array in MATLAB, it is converted to a 2-dimensional gray image before further processing. The sample of original input image and a gray image is shown below:



Fig 3.2.1 Original color image.

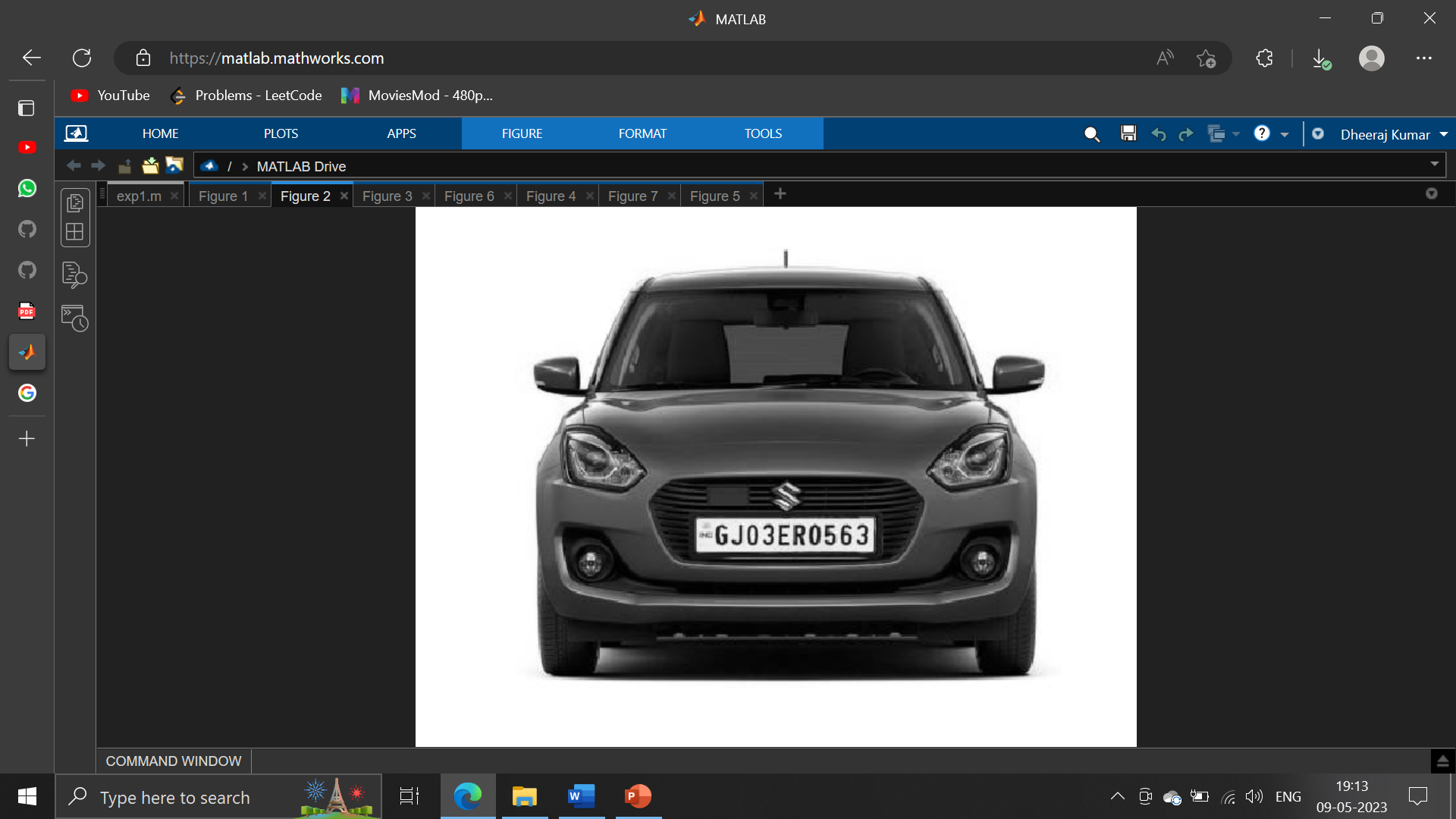


Fig 3.2.2 grey image

3.3 Dilate an Image

Dilation is a process of improvising given image by filling holes in an image, sharpen the edges of objects in an image, and join the broken lines and increase the brightness of an image. Using dilation, the noise with-in an image can also be removed. By making the edges sharper, the difference of gray value between neighboring pixels at the edge of an object can be increased. This enhances the edge detection. In Number Plate Detection, the image of a car plate may not always contain the same brightness and shades. Therefore, the given image has to be converted from RGB to gray form. However, during this conversion, certain important parameters like difference in color, lighter edges of object, etc. may get lost. The process of dilation will help to nullify such losses.

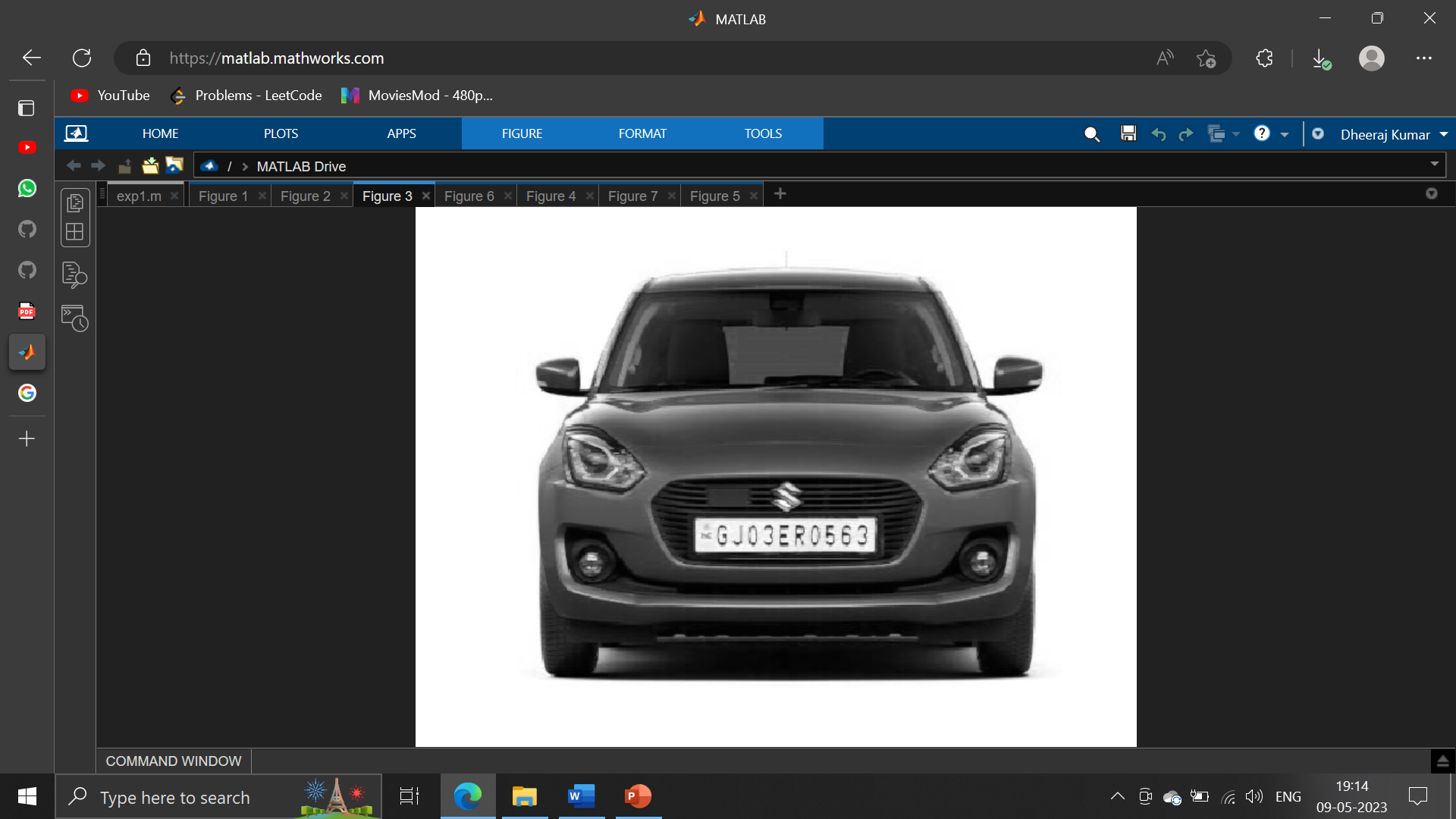


Fig 3.3 dilated image

3.4 Horizontal and Vertical Edge Processing of an Image

Histogram is a graph representing the values of a variable quantity over a given range. In this Number Plate Detection algorithm, the writer has used horizontal and vertical histogram, which represents the column-wise and row-wise histogram respectively. These histograms represent the sum of differences of gray values between neighboring pixels of an image, column-wise and row-wise. In the above step, first the horizontal histogram is calculated. To find a horizontal histogram, the algorithm traverses through each column of an image. In each column, the algorithm starts with the second pixel from the top. The difference between second and first pixel is calculated. If the difference exceeds certain threshold, it is added to total sum of differences. Then, algorithm will move downwards to calculate the difference between the third and second pixels. So on, it moves until the end of a column and calculate the total sum of differences between neighboring pixels. At the end, an array containing the column-wise sum is created. The same process is carried out to find the vertical histogram. In this case, rows are processed instead of columns.

3.5 Passing Histograms through a Low Pass Digital Filter

Referring to the figures shown below, one can see that the histogram values changes drastically between consecutive columns and rows. Therefore, to prevent loss of important information in upcoming steps, it is advisable to smooth out such drastic changes in values of histogram. For the same, the histogram is passed through a low-pass digital filter. While performing this step, each histogram value is averaged out considering the values on it right- hand side and left-hand side. This step is performed on both the horizontal histogram as well as the vertical histogram. Below are the figures showing the histogram before passing through a low-pass digital filter and after passing through a low-pass digital filter.

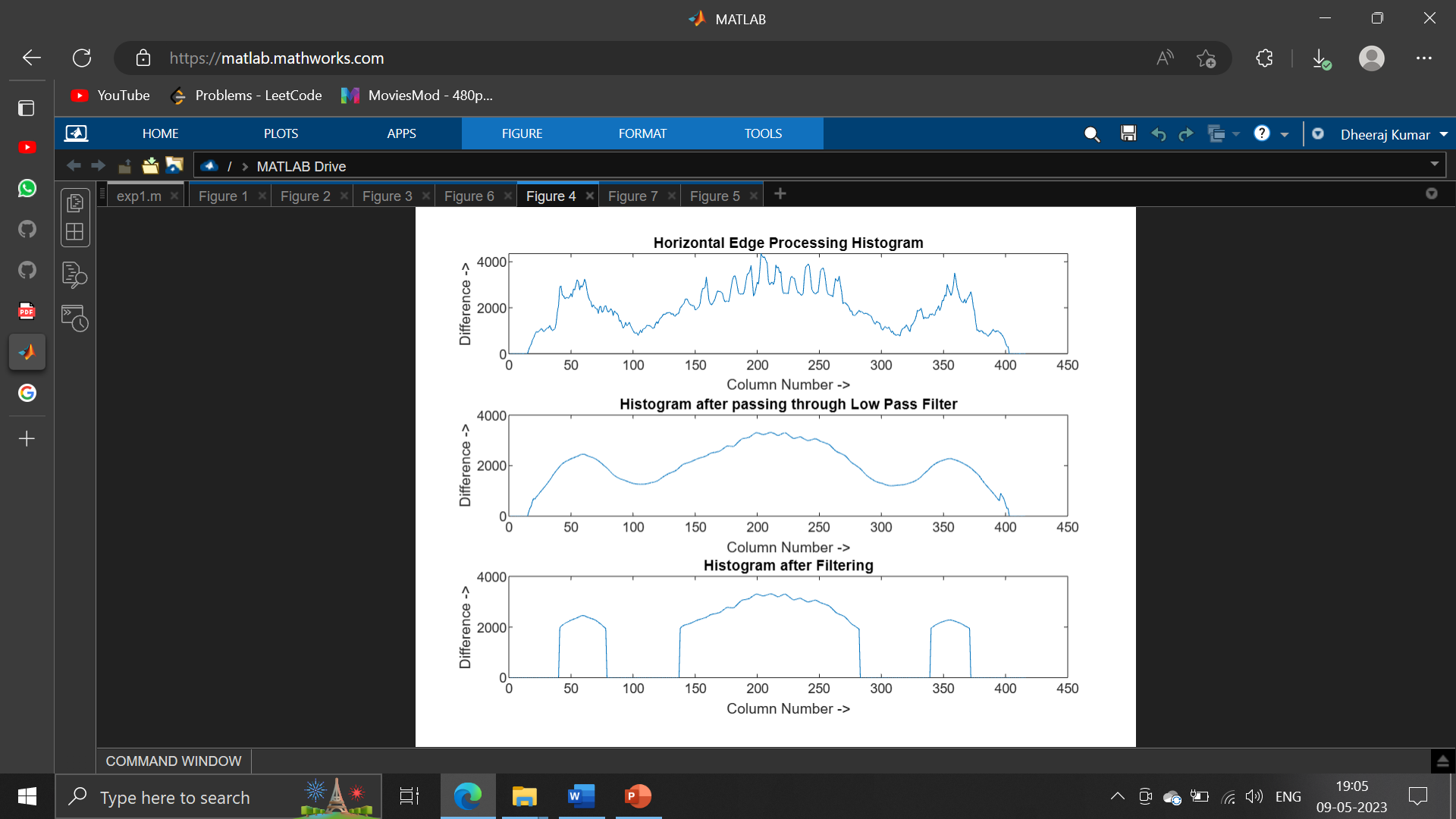


Fig.3.5.1 Horizontal edge processing histogram

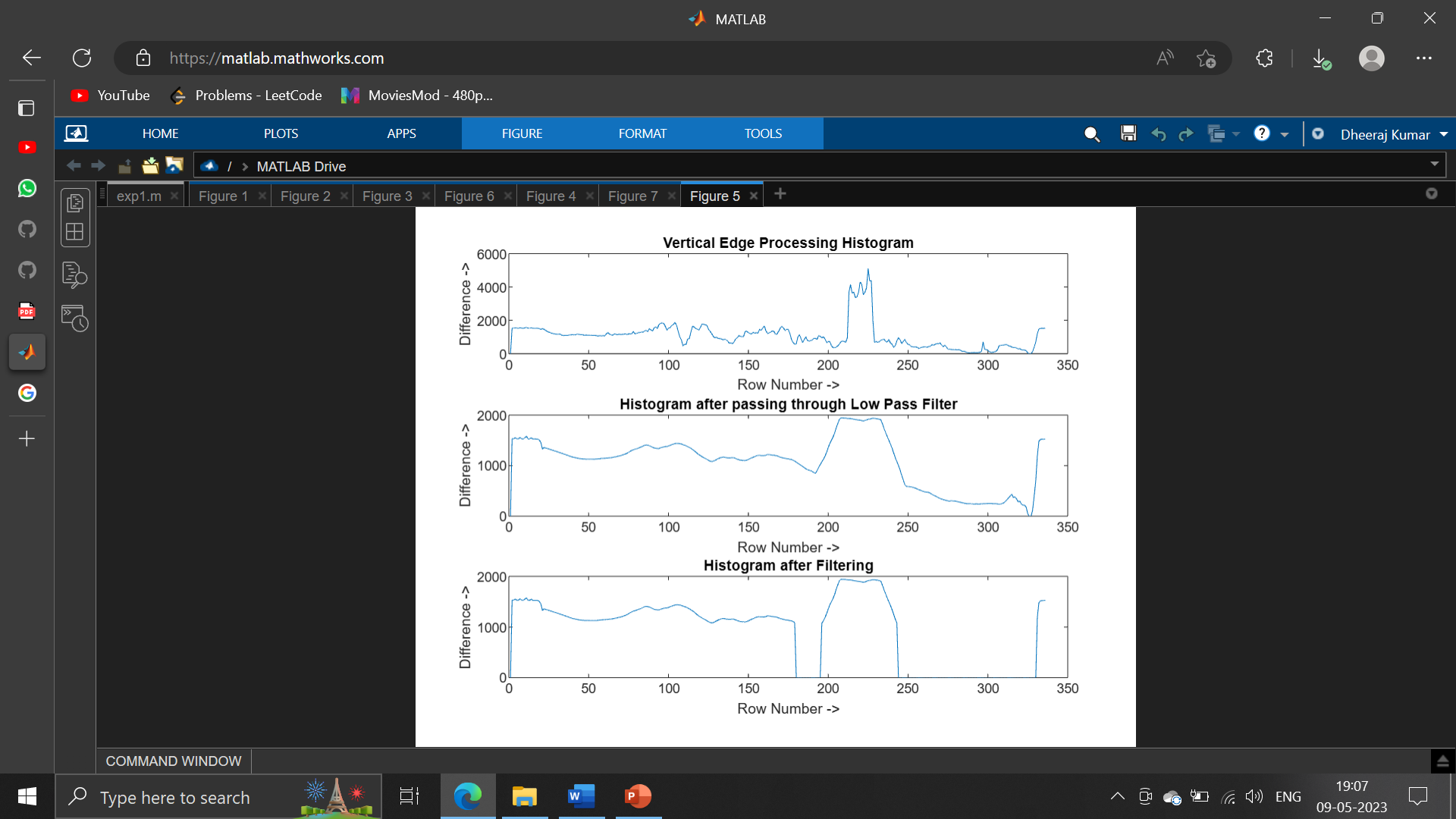


Fig.3.5.2 Vertical edge processing histogram.

3.6 Filtering out Unwanted Regions in an Image

Once the histograms are passed through a low-pass digital filter, a filter is applied to remove unwanted areas from an image. In this case, the unwanted areas are the rows and columns with low histogram values. A low histogram value indicates that the part of image contains very little variations among neighboring pixels. Since a region with a license plate contains a plain background with alphanumeric characters in it, the difference in the neighboring pixels, especially at the edges of characters and number plate, will be very high. This results in a high histogram value for such part of an image. Therefore, a region with probable license plate has a high horizontal and vertical histogram values. Areas with less value are thus not required anymore. Such areas are removed from an image by applying a dynamic threshold. In this algorithm, the dynamic threshold is equal to the average value of a histogram. Both horizontal and vertical histograms are passed through a filter with this dynamic threshold. The output of this process is histogram showing regions having high probability of containing a number plate.

3.7 Segmentation

The next step is to find all the regions in an image that has high probability of containing a license plate. Co-ordinates of all such probable regions are stored in an array. The output image displaying the probable license plate regions is shown below.

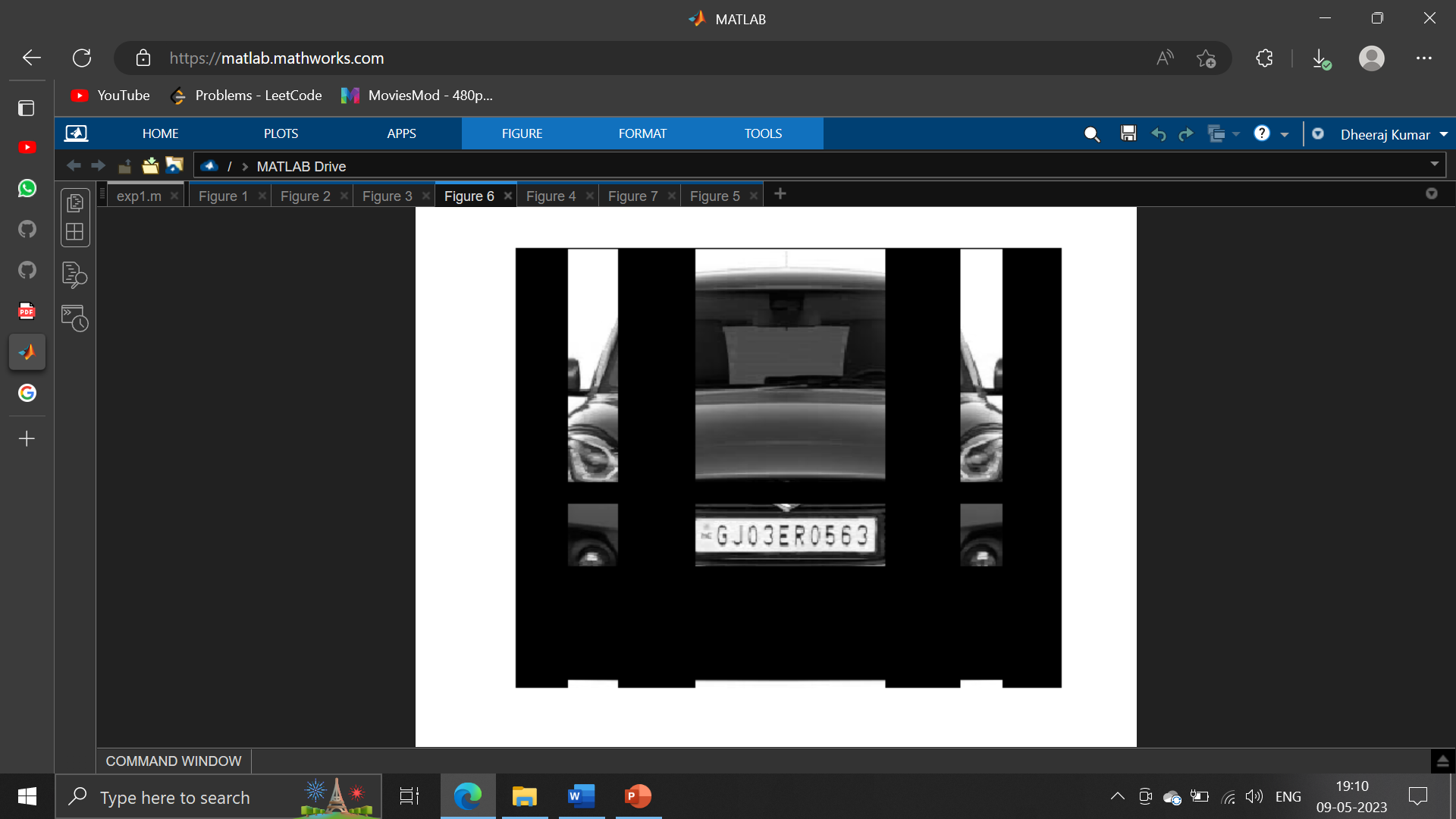


Fig. Output of segmentation.

3.8 Region of Interest Extraction

The output of segmentation process is all the regions that have maximum probability of containing a license plate. Out of these regions, the one with the maximum histogram value is considered as the most probable candidate for number plate. All the regions are processed row-wise and column-wise to find a common region having maximum horizontal and vertical histogram value. This is the region having highest probability of containing a license plate. The image detected license plate is shown below:

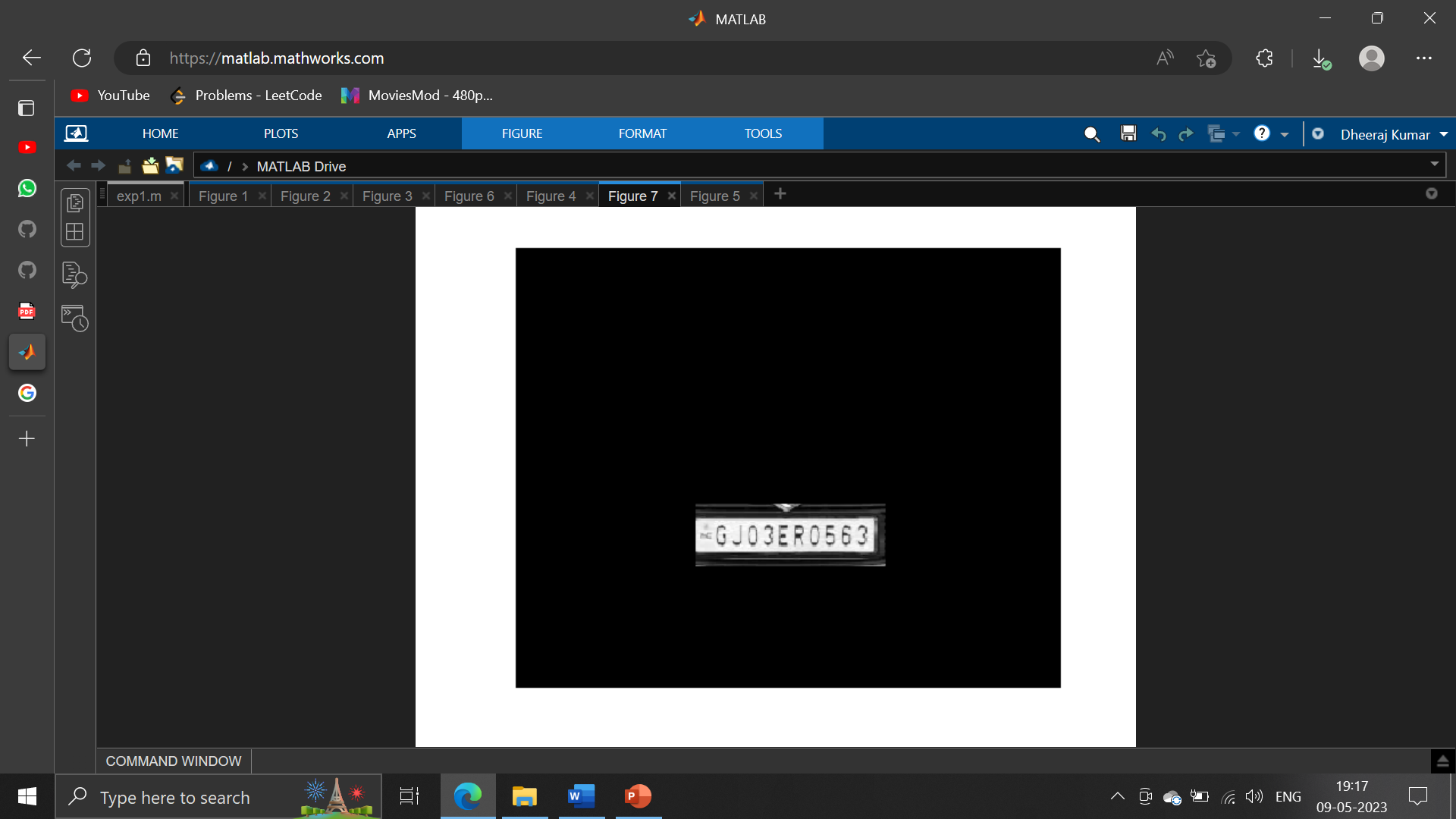


Fig. Detected license plate

This algorithm was verified using several input images having resolution varying from 680 \* 480 to 1600 \* 1200. The images contained vehicles of different colors and varying intensity of light. With all such images, the algorithm correctly recognized the number plate. This algorithm was also tried on images having number plate aligned at certain angle (approximately 8-10 degree) to horizontal axis. Even with such images, the number plates were detected successfully. After successfully implementing and verifying the algorithm in MATLAB, it was coded in C for implementation on actual hardware. The details about the same are covered in next chapter.

3.9 Matlab Code:

clc;

clear all;

close all;

I = imread ('https://rukminim1.flixcart.com/image/416/416/kws5hu80/vehicle-number-plate/x/g/s/50-swift-car-number-plate-frame-all-model-fit-yogiji-12-original-imag9dhhb9jz42xs.jpeg?q=70');

figure(1);

imshow(I);

Igray = rgb2gray(I); %(Convert an RGB Image to Gray Scale image)

[rows cols] = size(Igray);

Idilate = Igray; %% Dilate and Erode Image in order to remove noise

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);

difference = 0;

sum = 0;

total\_sum = 0;

difference = uint32(difference);

%% PROCESS EDGES IN HORIZONTAL DIRECTION

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(4);

% Plot the Histogram for analysis

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(5)

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(6), imshow(I);

%% Find Probable candidates for Number Plate

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(7), imshow(I);

imshow(I);

Chapter 4

4.1 Output:

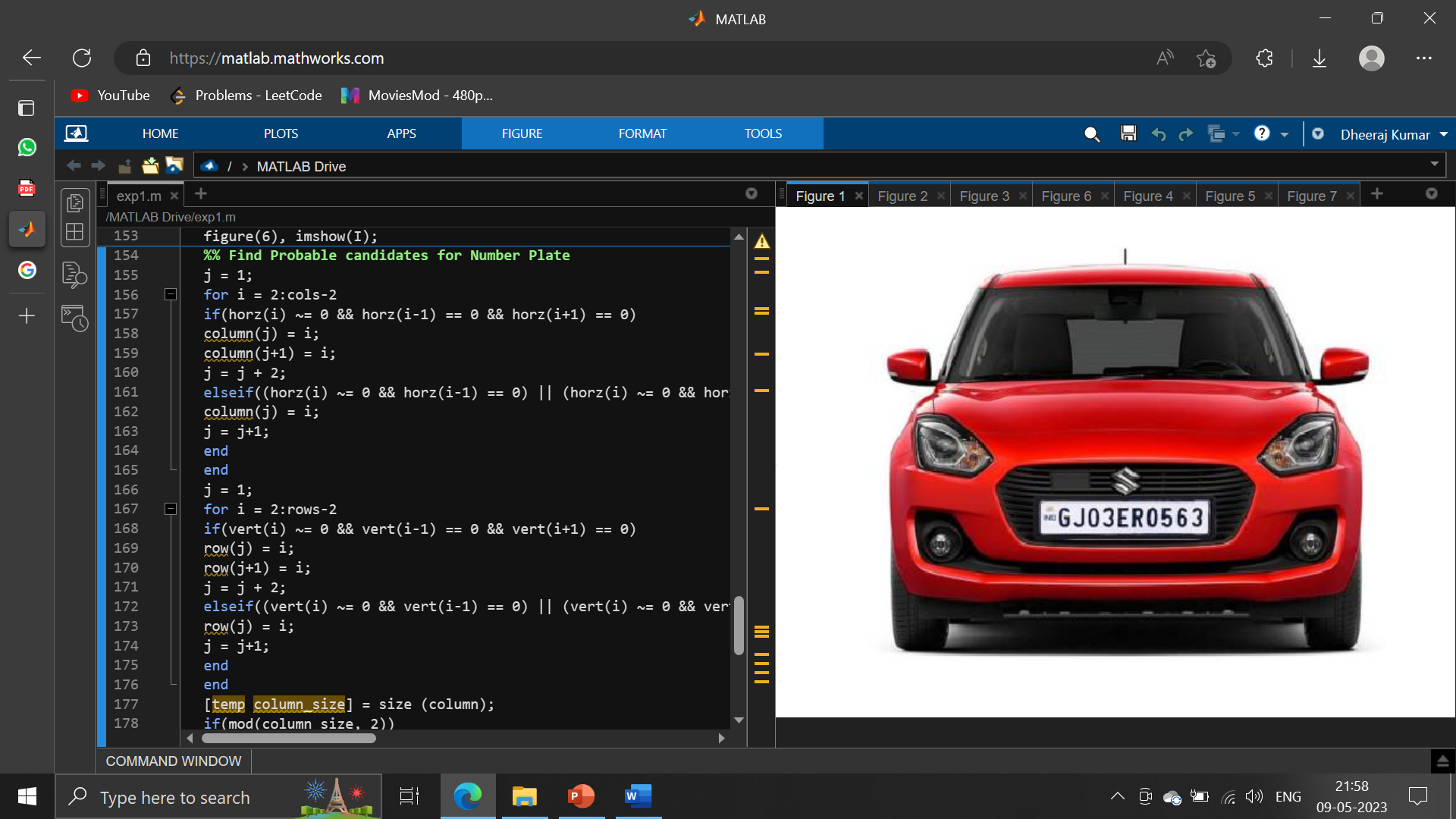


Fig4.1 This is the given image

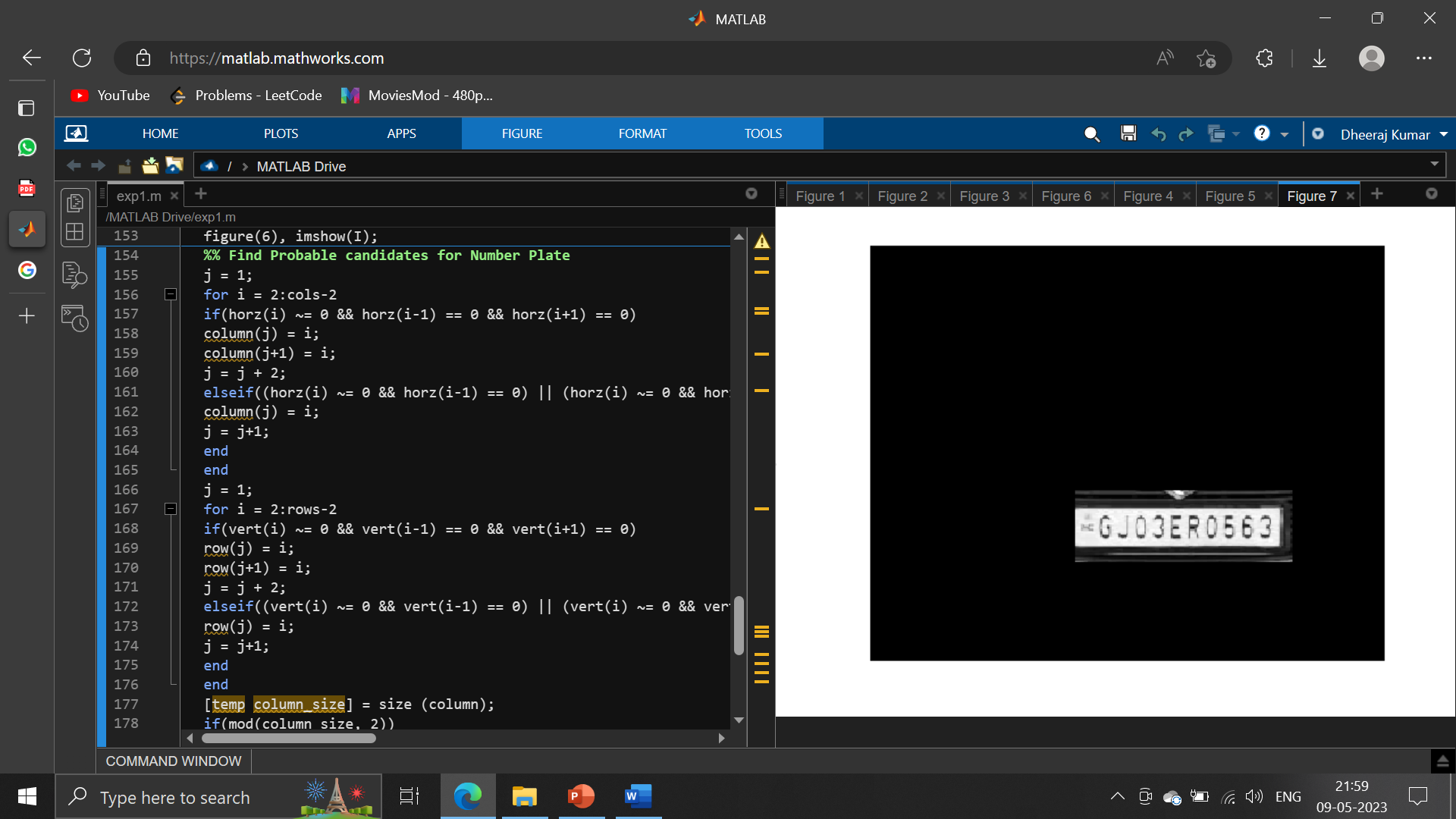


Fig4.2 Output of the code



Fig4.1 This is the given image

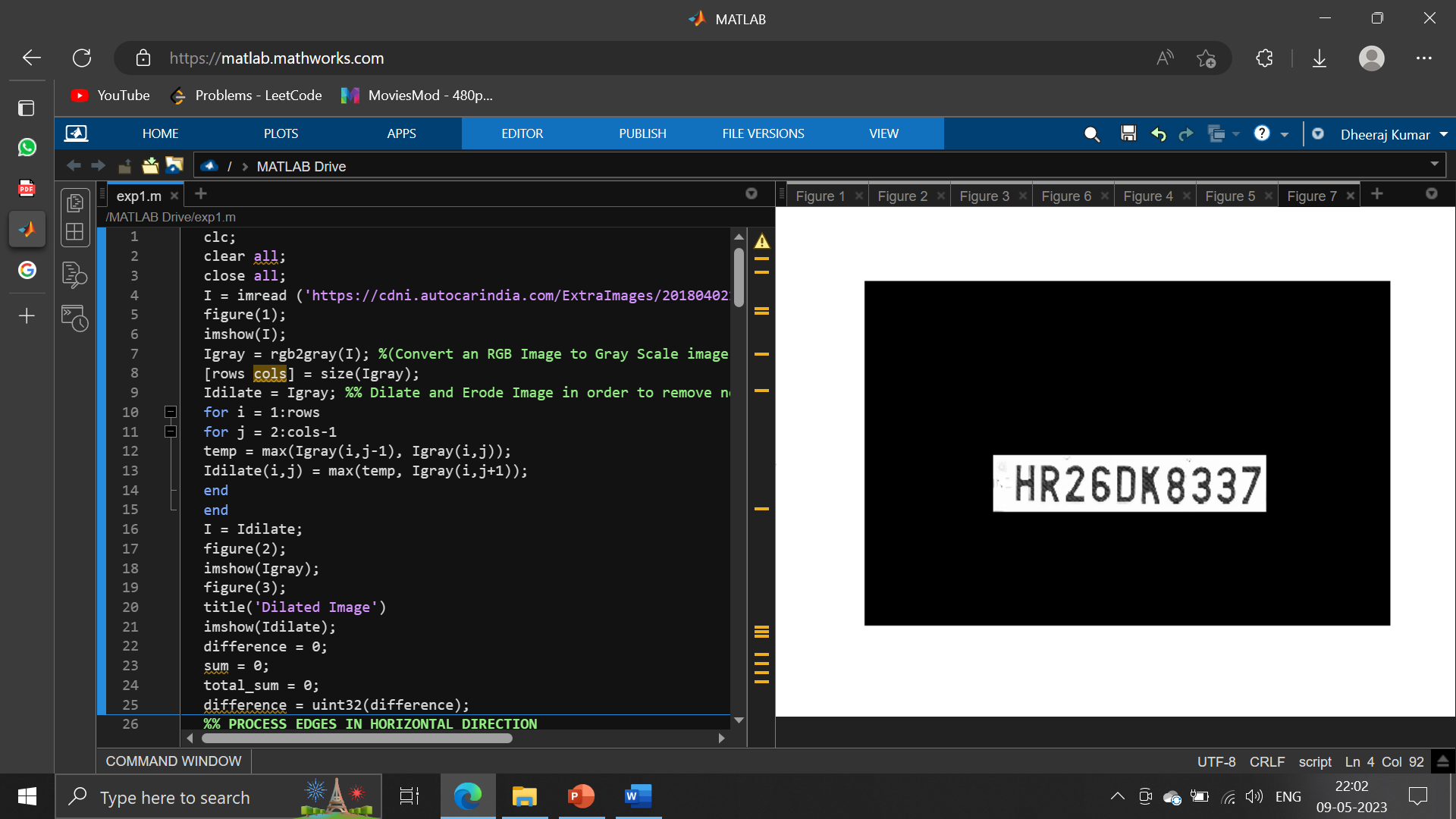


Fig4.4 Output of the code

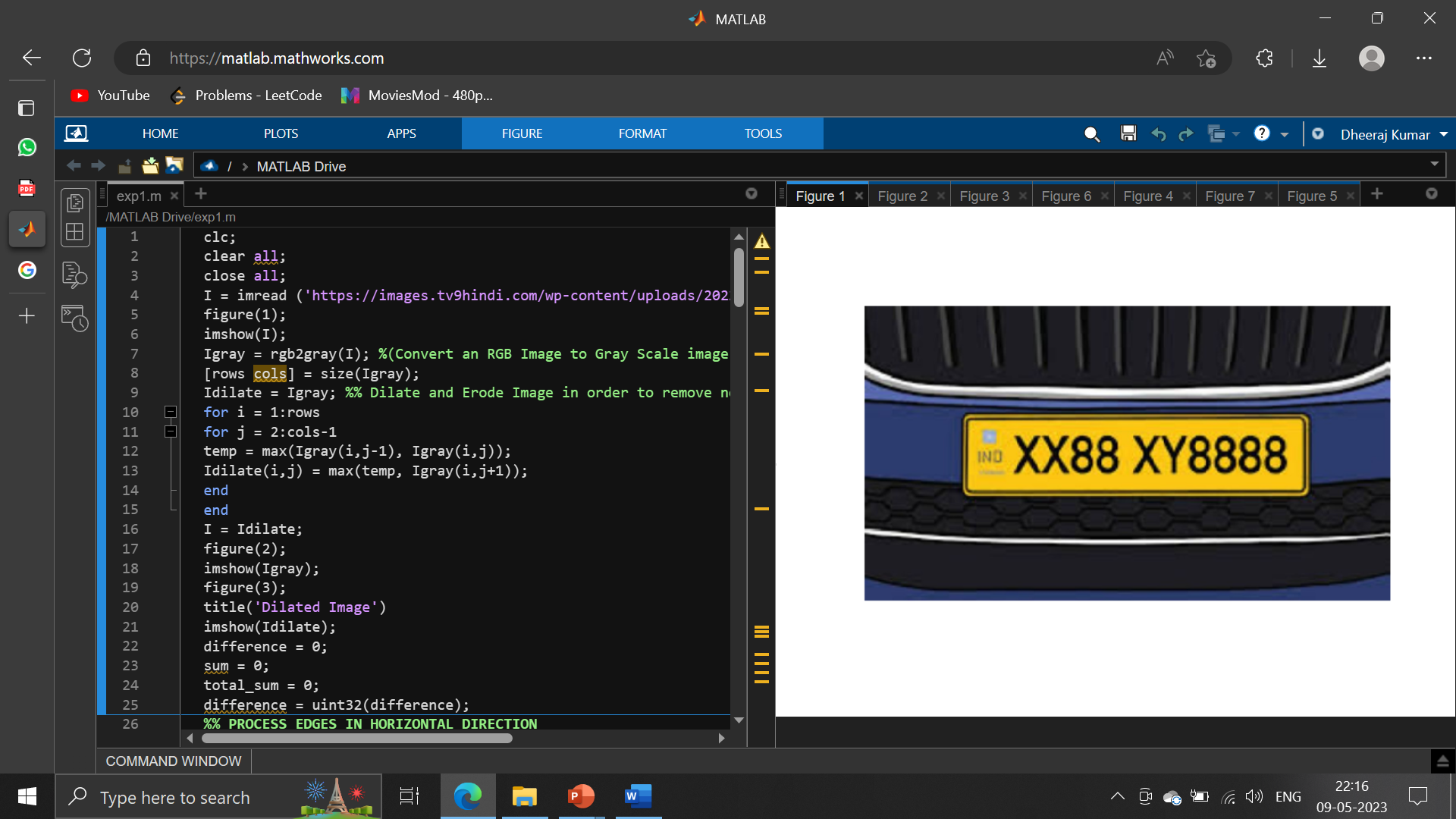


Fig4.5 This is the given image

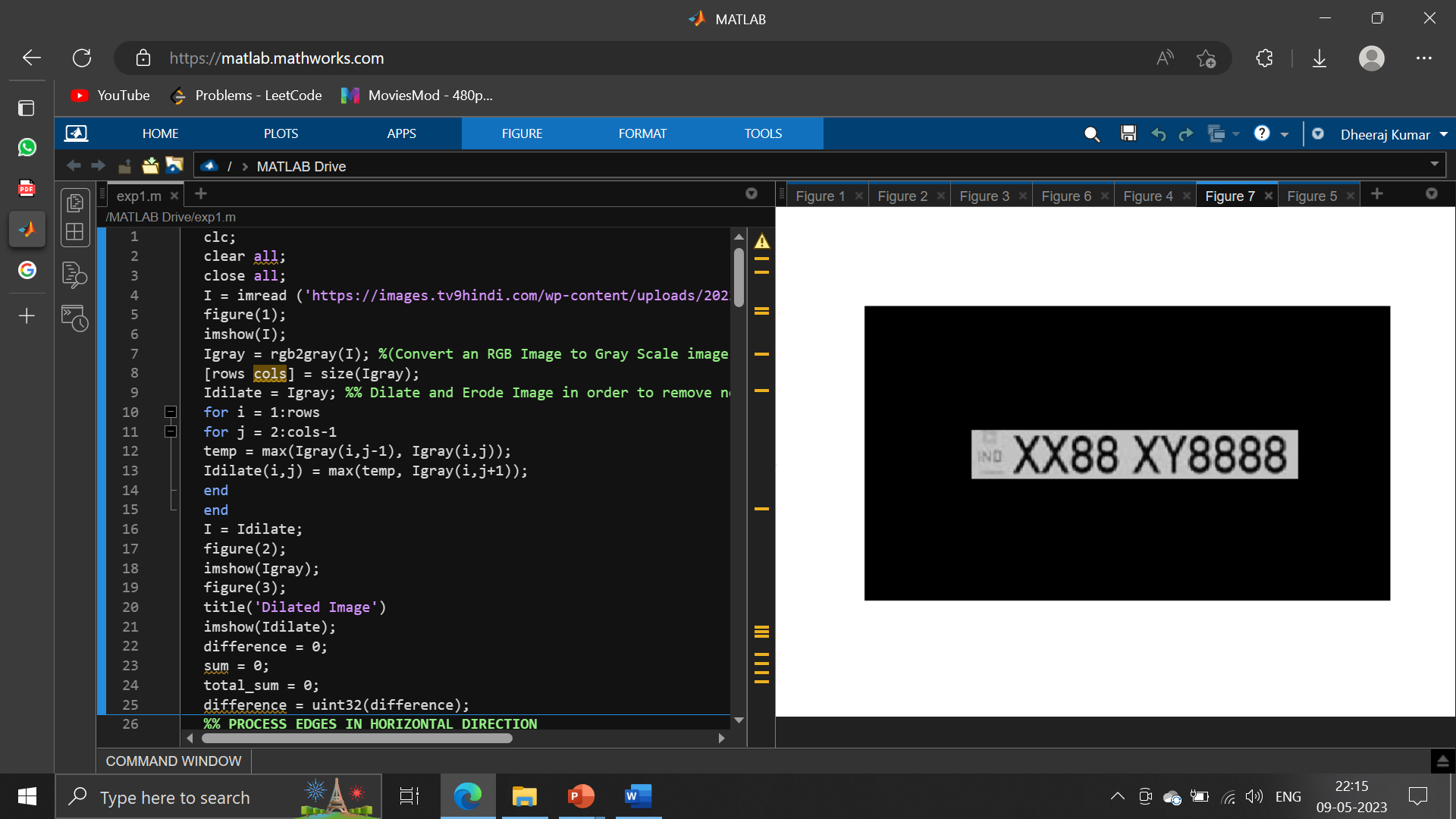


Fig4.6 Output of the code

4.2 Result:-

Successfully Recognition of number plate of cleared image of Car.

4.3 Conclusion:

The purpose of this study has been to create an automatic number plate recognition system. It describes complete intelligent digital image processing system to recognize the vehicle number plate using a sequential multistage approach performed over grayscale images, consists of two parts extraction of vehicle number plate from the image, and recognition of number plate. For extracting the vehicle number plate edge detection algorithm used. The experimental results show that this method is of higher recognition accuracy . This new approach provides a good direction for automatic number plate recognition. The system was designed using MATLAB.

References:

[1] Clemens Arth, Florian Limberger and Horst Bischof, “Real-Time License Plate Recognition on an Embedded DSP-Platform”, Proceedings of IEEE conference on Computer Vision and Pattern Recognition, pp 1-8, June 2007.

[2] Halina Kwasnicka and Bartosz Wawrzyniak, “License plate localization and recognition in camera pictures”, AI-METH 2002, November 13-15, 2002.

[3] Texas Instruments Inc., “TMS320DM6437 DVDP Getting Started Guide”, Texas, July 2007.

[4] Texas Instruments Inc., “TMS320DM6437 Digital Media Processor”, Texas, Page 5, November 2006 – Revised June 2008.

[5] Pramod Kapadia, “Car License Plate Recognition Using Template Matching Algorithm”, Master Project Report, California State University, Sacramento, Fall 2010.