Amrita Vishwa Vidyapeetham
Centre for Excellence in Computational Engineering and Networking
Amrita School of Engineering, Coimbatore

Recognition of Vehicle License Plate

Prepared By:
GROUP-14A
M.D.S. Rama Saran- [CB.EN.U4AIE21034]
Akshayaa BK- [CB.EN.U4AIE21002]
Gajula Sri Vatsanka- [CB.EN.U4AIE21010]
R. Sai Raghavendra- [CB.EN.U4AIE21049]

Supervised by:
Dr. Sachin Kumar S
Asst. Professor

An End Semester Project submitted to the CEN department as a part of course evaluations of **Signal and Image Processing** for B. Tech in **Computer Science Engineering – Artificial Intelligence.**

ACKNOWLEDGMENT

We are deeply thankful to **Centre for Excellence in Computational Engineering and Networking** (CEN) at **Amrita Vishwa Vidyapeetham**, **Coimbatore** for providing us such a wonderful environment to pursue our research. We would like to express our sincere gratitude to **Dr. Sachin Kumar S, Asst. Professor, Department of Centre for Excellence in CEN**, Amrita Vishwa Vidyapeetham. We have completed our research under her guidance. We found the research area, topic, and problem with her suggestions. We would also like to acknowledge our team members for supporting each other and be grateful to our university for providing this opportunity for us. Lastly special thanks to Centre for Excellence in CEN for providing this opportunity to research in this field.

TABLE OF CONTENTS

ABSTRACT

This report introduces a traditional approach for license plate detection in images, employing fundamental computer vision techniques. The method includes the conversion of RGB images to grayscale, edge enhancement through morphological operations, and processing of horizontal and vertical histograms for effective edge detection. Utilizing a moving average window and dynamic thresholding enhances the smoothness and accuracy of the histograms for each edge detection technique. The algorithm identifies potential candidates for the number plate region based on peak values in the histograms, eliminating regions outside the most probable candidate. The simplicity of this approach makes it suitable for scenarios with limited computational resources.

INTRODUCTION

Vehicle identification is a crucial aspect globally, achieved either manually or automatically through image processing techniques. Automatic Vehicle Identification (AVI) systems, a subset of image processing, play a vital role in efficiently managing traffic and enhancing security measures, such as access control and tracking wanted vehicles. Number Plate Recognition (NPR) simplifies vehicle identification, although implementing it for Indian license plates poses challenges due to the absence of a standardized aspect ratio.

The intricacy of the identification task is compounded by varying lighting conditions. Despite years of experimentation with number plate detection, the process remains a formidable challenge. The detection system scrutinizes input images to pinpoint local patches likely to contain license plates. Given that plates can appear anywhere in an image with diverse sizes, exhaustively examining every pixel for plate localization is impractical.

In scenarios like parking management, number plates serve the purpose of calculating parking duration. Upon a vehicle entering a desi gnated gate, the number plate is automatically recognized and stored in a database. This seamless integration of number plate recognition not only enhances security but also streamlines administrative processes related to parking management

Some other scenarios are Law Enforcement and Security, Traffic Surveillance and Management,…etc.

MOTIVATION OF OUR PROJECT

Creating a license plate detection project is exciting because it helps to make our roads safer and more secure. It saves time and resources for law enforcement, making their job easier. It actively contributes to community safety by preventing crimes and assisting investigations. Furthermore, it aligns with the concept of smart cities by promoting efficient transportation systems. The project demonstrates the power of technology by employing AI and computer vision to solve real-world problems. It directly benefits the well-being of our communities by identifying stolen vehicles and those involved in criminal activity. The generated data provides valuable insights for better decision-making, such as better understanding traffic patterns and urban planning. With global applications ranging from law enforcement to toll collection, the project has the potential to have a large-scale positive impact. Working on this project encourages innovation and collaboration by bringing different skills together to address challenges and create a safer, more connected world.

Literature survey

Image Preprocessing

1. Sulehria, Humayun Karim, et al. "Vehicle number plate recognition using mathematical morphology and neural networks." *WSEAS Transactions on Computers* 7.6 (2008): 781-790.

2. Radha, R., and C. P. Sumathi. "A novel approach to extract text from license plate of vehicles." (2012).

Dilation:

- Dilation is a morphological operation that causes objects to grow in size.
- It is performed by replacing each pixel value with the maximum value within a specified neighborhood (defined by the structuring element).

Erosion:

- Erosion is another morphological operation that causes objects to decrease in size.
- It is performed by replacing each pixel value with the minimum value within a specified neighborhood.

Closing:

- Closing is a morphological operation defined as dilation followed by erosion.
- It tends to close small holes or gaps in the bright regions while maintaining the general shape of the objects.

Edge detection:

1. Deshpande, Soojey, et al. "Use of horizontal and vertical edge processing technique to improve number plate detection." *Int. J. Res. Eng. Technol* 4.12 (2015): 286-292.

Histogram Computation:

- Horizontal and vertical histograms are computed to represent variations in gray values among adjacent pixels in columns and rows, respectively.

Low-Pass Digital Filter Application:

- Applying moving average window which is a type of low-pass digital filter is applied to smooth out the histograms, addressing excessive variations and preventing data loss.

- The filter averages values on both sides of each histogram estimate, creating a flatter signal and reducing noise.

Identification of Unnecessary Regions:

- Processed histograms assist in identifying regions with lower histogram values, indicating areas with minimal deviations between adjacent pixels.

Dynamic Thresholding:

- A dynamic threshold, set to the average value of the histogram, is applied to identify and remove less essential regions in the image.
- Areas with histogram values below the threshold are considered less important and are subsequently eliminated.

Focus on Number Plate Regions:

- The outcome is histograms highlighting areas with a high likelihood of containing a number plate, as they exhibit significant variations in gray values.
- Particularly, regions with substantial histogram variations are identified as potential locations where a license plate may be present.

Detection of License plate regions(ROI):

1. Wang, Shen-Zheng, and Hsi-Jian Lee. "Detection and recognition of license plate characters with different appearances." *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems*. Vol. 2. IEEE, 2003.

2. Lekhana, G. C., and R. Srikantaswamy. "Real time license plate recognition system." *International Journal of Advanced Technology & Engineering Research* 2.4 (2012): 5-9.

License Plate Location:

- Identified license plate areas based on significant intensity deviations in the image.

Array Storage of Potential Areas:

- Stored coordinates of all potential license plate areas in an array.

Character Segmentation with Line Scanning:

- Achieved effective character segmentation using the Line Scanning Technique.
- Isolated regions within the image displaying significant intensity deviations.

Extraction of Region of Interest (ROI):

- Extracted the region of interest (ROI) containing areas with the highest likelihood of containing the license plate.

Mutual Analysis with Horizontal and Vertical Histograms:

- Treated the entire area row-wise and column-wise for mutual analysis.
- Identified areas with the highest horizontal and vertical histogram values as potential candidates for the license plate.

Maximum Histogram Estimate Selection:

- Determined the region with the maximum histogram estimate as the most probable candidate for the license plate

METHODS USED FOR IMPLEMENTATION

Our code follows the below methodologies:

- **RGB to Grayscale Conversion:**
  - *Method*: grayscale(I)
  - *Explanation:* Converts the original color image (I) to grayscale. Grayscale images simplify processing and reduce computational complexity.

- **Dilation:**
  - *Method:* Morphological operations (Dilation)
  - *Explanation:* Dilation is applied to the grayscale image. Morphing enhances bright regions in an image, effectively smoothing and removing noise.

- **Erosion:**
  - *Method:* Morphological operations (Erosion)
  - *Explanation:* Erosion is applied to the Dilated image. Erosion diminishes bright regions in an image, effectively sharpening and removing noise.

- **Horizontal Edge Processing:**
  - *Methods:* Horizontal difference calculation, smoothening by applying moving average window and Dynamic thresholding
  - *Explanation:* Horizontal differences between pixel intensities are computed to identify edges. A low-pass filter is applied to smoothen the horizontal histogram.
  - Dynamic thresholding is used to filter out values below an average threshold.

- **Vertical Edge Processing:**
  - *Methods:* Vertical difference calculation, , smoothening by applying moving average window, Dynamic thresholding.
  - *Explanation:* Vertical differences between pixel intensities are computed to identify edges. A low-pass filter is applied to smoothen the vertical histogram.
  - Dynamic thresholding is used to filter out values below an average threshold

- **Smoothening of Histogram by applying Moving average window:**
  - Utilized moving average window, which is a type of low-pass filter to smoothen the histogram, reducing noise and emphasizing prominent features.
  - Implemented a 41-point moving average window to achieve the smoothing effect, enhancing the overall clarity of the horizontal or vertical edge information.

- **Filtering out Histogram Values by applying Dynamic Threshold:**
  - Implemented dynamic thresholding to filter out histogram values below an adaptive average.
  - Thresholder values based on the calculated average, effectively removing low-intensity variations and emphasizing significant edges in the image.

- **Identification of Probable Candidates for Number Plate:**
  - *Methods:* Identifying peaks in histograms

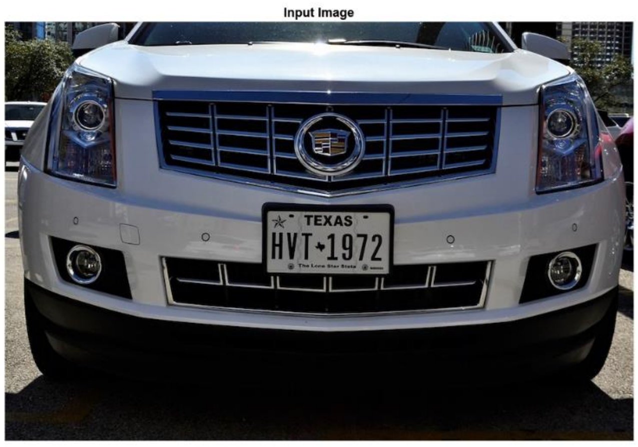
- *Explanation:* Peaks in both horizontal and vertical histograms are identified as probable candidates for the number plate.

- **Region of Interest (ROI) Extraction:**
  - *Methods:* Extracting regions based on identified candidates
  - *Explanation:* Regions that are not identified as probable number plates are removed from the image.

CODE IMPLEMENTATION

Original image as input

```
I = imread("C:\Users\SRI\Desktop\SEM-5\Number-Plate-Detection-master\car.jpg");
figure(1);
imshow(I);
```

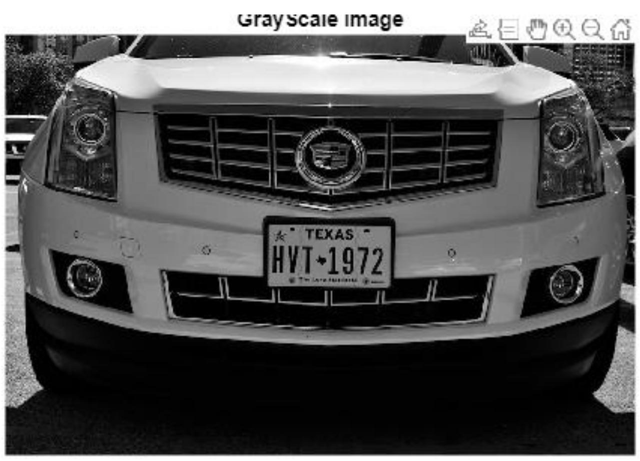


Grayscale conversion

```
% Convert the image to grayscale
Igray = grayscale(I);
[rows, cols] = size(Igray);
figure(2);
imshow(Igray);
title('GrayScale Image')
```

Explanation:

- The code utilizes the rgb2gray function to convert a given RGB image (I) into a grayscale image (Igray).
- Extracts the dimensions (number of rows and columns) of the grayscale image using the size function and assigns them to the variables rows and cols.

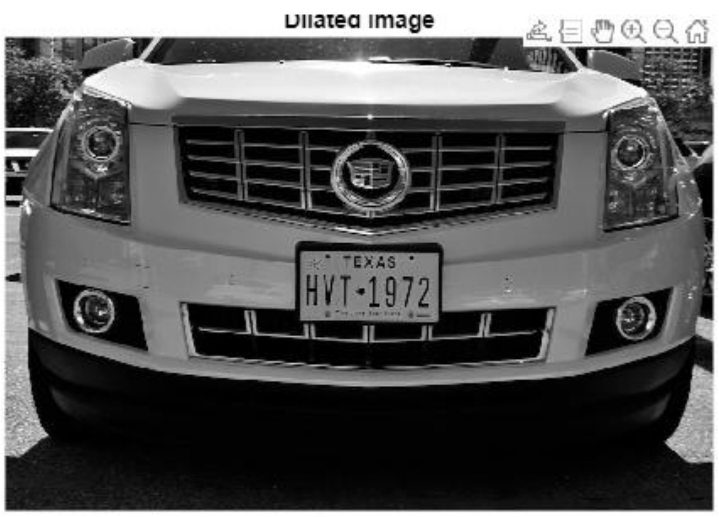


Dilated:

```
%% Dilate Image 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;

%% Display Dilated Image
figure(3);
imshow(Idilate);
title('Dilated Image');
```

Explanation:

- The code performs a dilation operation on the grayscale image (Igray) to reduce noise. It creates a new image (Idilate) with the same dimensions and fills each pixel based on the maximum intensity value within its neighboring pixels.
- Nested loops iterate through each pixel in the image (except the boundary pixels) and calculate the maximum value among the pixel and its neighbors. This process helps to enhance and thicken prominent features while reducing the impact of noise.
- After the dilation operation, the original image (I) is updated with the dilated image (Idilate). This step is crucial for subsequent processing steps to work on the noise-reduced version of the image.

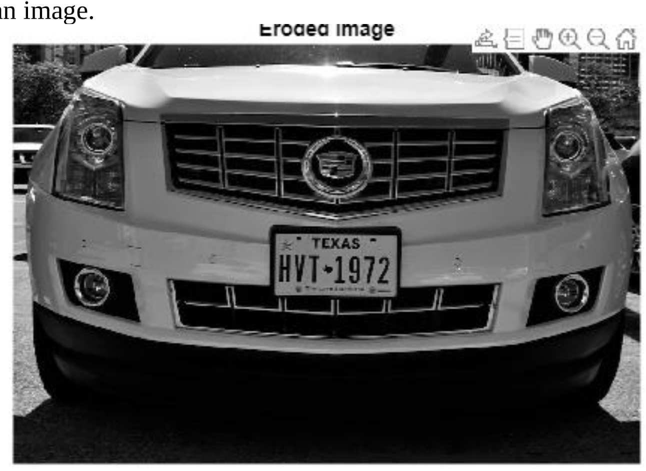


Eroded:

```
%% Erode Image
Ierode = Idilate;
for i = 1:rows
    for j = 2:cols-1
        temp = min(Idilate(i, j-1), Idilate(i, j));
        Ierode(i, j) = min(temp, Idilate(i, j+1));
    end
end
I = Ierode;

% Display Eroded Image
figure(4);
imshow(Ierode);
title('Eroded Image');
```

Explantion:

- The code performs an erosion operation on the previously dilated image (Idilate). Erosion is a morphological operation that reduces the size of bright regions, helping to further remove noise and fine-tune the image.
- Similar to the dilation operation, nested loops iterate through each pixel (excluding boundary pixels) and calculate the minimum value among the pixel and its neighbors. This process aids in smoothing and refining the image by removing small-scale variations.
- After the erosion operation, the original image (I) is updated with the eroded image (Ierode). This sequential application of dilation and erosion is a common technique in morphological operations to enhance or diminish specific features in an image.



Horizontal Edge Processing:

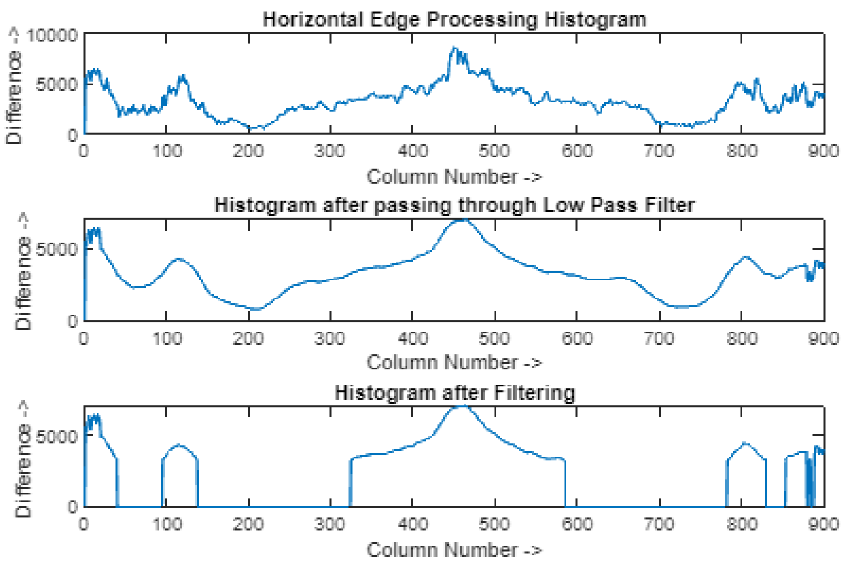
```
%% PROCESS EDGES IN HORIZONTAL DIRECTION
difference = 0;
sum = 0;
total_sum = 0;
difference = uint32(difference);

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 Moving average window
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 applying Moving average window’);
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 ->');
```

Explantion:

- Initializes variables such as difference, sum, and total_sum to manage computations during edge processing. The variable difference is explicitly cast as a 32-bit unsigned integer (uint32). These variables are crucial for storing and updating values during the subsequent horizontal edge detection process.
- Utilizes nested loops to traverse through pixels in the horizontal direction of the image (I). Calculates differences in intensity values between adjacent pixels along each column and accumulates these differences in the sum variable. If the calculated difference exceeds a threshold (20 in this case), it contributes to the sum. The resulting values are stored in the horz1 array, representing the horizontal edge information.

.



Vertical Edge Processing

```
%% 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 Moving average window
disp('Passing Vertical Histogram through moving average window');
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 Moving average window');
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);
```

Explantion:

- Initializes variables such as difference and total_sum for managing computations during vertical edge processing. The variable difference is explicitly cast as a 32-bit unsigned integer (uint32). These variables are essential for storing and updating values during the subsequent vertical edge detection process.
- Uses nested loops to traverse through pixels in the vertical direction of the image (I). Calculates differences in intensity values between adjacent pixels along each row and accumulates these differences in the sum variable. If the calculated difference exceeds a threshold (20 in this case), it contributes to the sum. The resulting values are stored in the vert1 array, representing the vertical edge information.
- Generates a histogram (vert1) based on the accumulated differences, illustrating the distribution of vertical edge strengths. The histogram is then plotted for visual analysis in the first subplot.
- Applies a low-pass filter to smoothen the vertical histogram (vert1). The filter operation involves averaging neighboring values, reducing high-frequency noise and emphasizing prominent features. The resulting smoothed histogram is visualized in the second subplot.
- Filters out values in the vertical histogram based on a dynamic threshold (average). Values below the threshold are set to 0 in the histogram, and the corresponding pixels in the original image (I) are also set to 0. This step aims to emphasize significant vertical edges while suppressing noise. The filtered histogram is shown in the third subplot.

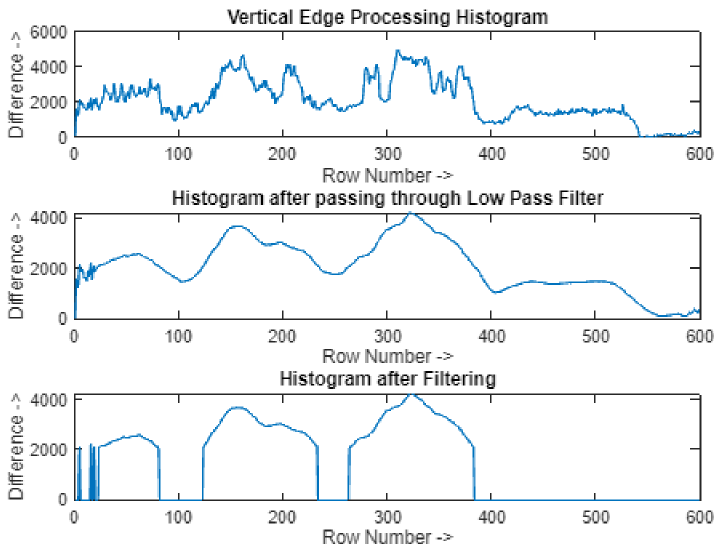
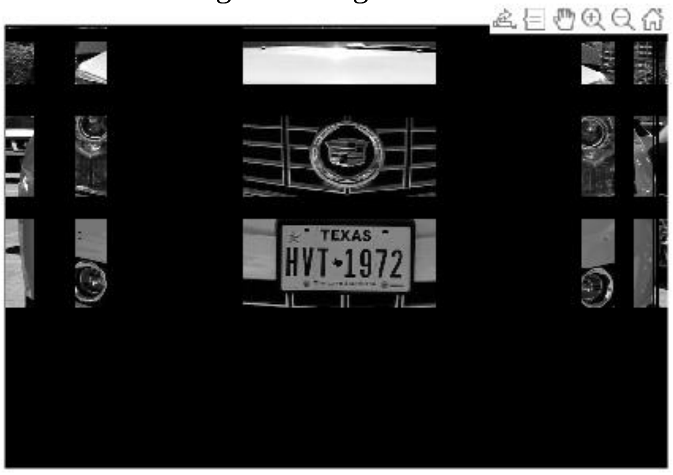


Image after Segmentation:



Finding the Probable Candidates for Number Plate and Region of Interest (ROI) Extraction:

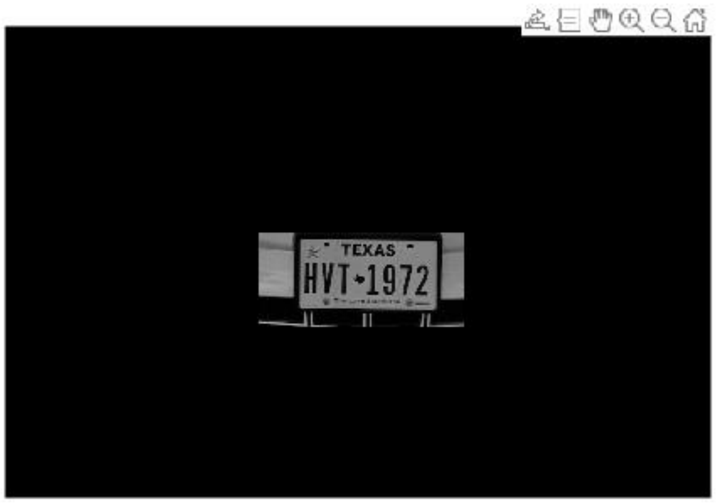
```
%% Find Probable candidates for Number Plate
j = 1;
for i = 2:cols-2
    if (horz(i) ~= 0 && horz(i-1) == 0 && horz(i+1) == 0)
        column(j) = i;
        column(j+1) = i;
        j = j + 2;
    elseif ((horz(i) ~= 0 && horz(i-1) == 0) || (horz(i) ~= 0 && horz(i+1) == 0))
        column(j) = i;
        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);
```

Explantion:

- Iterates through the horizontal histogram (horz) to identify columns where edges are present. Columns with a non-zero value and where the preceding and succeeding columns have zero values are considered potential candidates for containing number plate information. Detected columns are stored in the column array.
- Similar to the column detection process, iterates through the vertical histogram (vert) to identify rows where edges are present. Rows with a non-zero value and where the preceding and succeeding rows have zero values are considered potential candidates for containing number plate information. Detected rows are stored in the row array.
- Checks the size of the column and row arrays. If the size is odd, it adjusts the arrays by adding the last column or row to ensure even sizes. This step is crucial for subsequent processing.
- Examines each probable candidate region defined by columns and rows. If a region is not identified as the most probable region (based on the previously determined maximum horizontal and vertical positions), it is

removed from the original image (I). This process involves setting the pixel values within the non-probable region to 0, effectively eliminating unwanted portions.

- 

INFERENCE

To summarize, developing a license plate detection project is more than just a technological endeavor; it is also a significant contribution to public safety, security, and efficiency. The project directly addresses real-world issues such as crime prevention, traffic safety, and the optimization of law enforcement resources by leveraging the capabilities of AI and computer vision. The impact extends beyond local communities to a global scale, with applications ranging from improved security to smart city urban planning. Participating in this project is about more than just technological innovation; it is about creating a positive and interconnected world in which the benefits of advanced technology are used to improve the well-being and safety of communities all over the world.

CONCLUSION

In conclusion, the MATLAB code presented implements a basic vehicle number plate detection algorithm using image processing techniques and the license plate is detected. It serves as a beacon for improved security, road safety, and law enforcement efficiency. The ease with which it can be used—saving time, preventing crimes, and contributing to community well-being—underscores its importance. As we work to build smarter cities and use technology to solve real-world problems, the project's global impact grows, promising a safer and more connected future.