LICENSE PLATE RECOGNITION AND DETECTION BY MACHINE LEARNING APPROACH

A Project report submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

M. BALAJI (317126512036) **K.SOWMYASREE(317126512030)**

B.NAGA BHARATH CHANDRA (317126512003) **O.SRILEKHA(317126512045)**

Under the guidance of

Mr.BIBEKANANDA JENA(Ph.D)

Assistant Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES
(UGC AUTONOMOUS)
(*Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC with 'A' Grade*)
Sangivalasa, bheemili mandal, visakhapatnam dist.(A.P)2020-2021

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES
(UGC AUTONOMOUS)
(*Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC with 'A' Grade*)
Sangivalasa, Bheemili mandal, Visakhapatnam dist. (A.P)

PRAGNANAM BRAHMA

ANITS

CERTIFICATE

This is to certify that the project report entitled **"LICENSE PLATE RECOGNITION AND DETECTION BY MACHINE LEARNING APPROACH"** submitted by **M.BALAJI (317126512036), K. SOWMYA SREE (317126512030), B. NAGA BHARATH CHANDRA (317126512003), O. SRILEKHA (317126512045)** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Electronics & Communication Engineering** of Andhra University, Visakhapatnam is a record of bonafide work carried out under my guidance and supervision.

Project Guide	**Head of the Department**
[illegible]	[illegible]
Mr. B. JENA	**Dr. V. Rajyalakshmi**
Assistant Professor	Professor &HOD
Department of E.C.E	Department of E.C.E
ANITS	ANITS
Assistant Professor	Head of the Department
Department of E.C.E.	Department of E C E
Anil Neerukonda	Anil Neerukonda Institute of Technology & Sciences
Institute of Technology & Sciences	Sangivalasa - 531 162
S[illegible]ivalasa, Visakhapatnam-531 162	

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our project guide **Mr. B. JENA(Ph.D),** Assistant professor, Department of Electronics and Communication Engineering, ANITS, for his/her guidance with unsurpassed knowledge and immense encouragement. We are grateful to **Dr. V. Rajyalakshmi**, Head of the Department, Electronics and Communication Engineering, for providing us with the required facilities for the completion of the project work.

We are very much thankful to the **Principal and Management, ANITS, Sangivalasa,** for their encouragement and cooperation to carry out this work.

We express our thanks to all **teaching faculty** of Department of ECE, whose suggestions during reviews helped us in accomplishment of our project. We would like to thank **all non-teaching staff** of the Department of ECE, ANITS for providing great assistance in accomplishment of our project.

We would like to thank our parents, friends, and classmates for their encouragement throughout our project period. At last but not the least, we thank everyone for supporting us directly or indirectly in completing this project successfully.

PROJECT STUDENTS
M.BALAJI (317126512036),
K.SOWMYASREE(317126512030),
B.NAGABHARATH CHANDRA(317126512003),
O.SRILEKHA(317126512045).

ABSTRACT

This paper provides a robust and efficient method to detect and recognize the license plate from a given image using machine learning approach. LPD plays an important role in license plate detection in terms of both detection accuracy and efficiency and thus its influence in intelligent transport systems in smart cities is very great. This method has three steps: pre-processing, candidate extraction and candidate verification.

In pre-processing, we convert the high-resolution input image into gray-scaled image after down scaling the image. In candidate extraction, we apply edge detection to the gray-scaled image by using an extended Sobel operator method and then a binary image is generated using adaptive thresholding method. Then, the license plate region is detected using a novel line density filter approach. A cascaded license plate classifier based on linear SVMs using color saliency features is introduced to identify the true license plate from among the candidate regions. For this we train the system with different images with a few characteristics to differentiate the true region from the unnecessary regions. Finally, the characters from the true license plate region are detected and displayed.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

- LP - license plate
- NLP - non licensed plate
- SVM - support vector machine
- AT - adaptive thresholding
- LDF - line density filter
- VPLR: Vehicle plate license recognition
- RFIO: Radio frequency identification
- SVM: support vector machine
- BPNN: back propagation neutral network
- VTD: vertical traverse density
- HTD: horizontal traverse density
- LPR: License plate recognition
- LPD: license plate detection

CHAPTER 1
INTRODUCTION

Vehicle plate licenses recognition (VPLR) plays a significant role in the field of intelligent transportation system. It has been widely used in traffic management, vehicle monitoring, suspect vehicle tracking, and many other fields. For example, in some cities in China, a new VPLR technology which enables drivers to pay parking fee using electronic wallet in a short time without leaving cars has received widely favourable reception. The parking fee from the drivers can be automatically collected by OCR (Optical Character Recognition) systems which can recognize license plates. However, errors sometimes occur when a vehicle is not identified or when a vehicle is wrongly identified as another vehicle. Therefore, RFID (Radio Frequency Identification) device and Bluetooth equipment can be combined to yield a better recognition performance. Many new technologies like that are emerging and rising due to people's changing demands; it is necessary to improve license plate location and recognition algorithms to increase vehicle management efficiency. In general, license plate recognition process consists of three steps: license plate localization, characters segmentation, and characters classification and recognition. Since the license plate localization is the first and essential step of the recognition process, the result has a direct impact on the accuracy of character segmentation and character recognition. However, the license plate can be easily affected by external factors such as lightning conditions, weather, and backgrounds besides, most VPLR systems do not fully consider the complexity of background and illumination conditions in the practical application, so locating and detecting the license plate from original images accurately and efficiently are still vital steps and the main difficulties for successful license plate recognition.

At present, the representative license plate classification and recognition task is performed by machine learning methods, such as a Support Vector Machine (SVM) classifier, and the neural network methods including Backpropagation Neural Network (BPNN). In particularly, multilayer neural networks and backpropagation training is adequate for vehicle plate licenses reading and recognition . In addition, since the high dimensional original images contain redundant information, it is better to extract useful image characteristics rather than using each pixel value of the images as feature vectors . Therefore, the feature extraction method plays a significant role in these license plate recognition algorithms, which need to extract different sets of features as the input layer of the network.

In the past several years, different methods have been exploited to extract features to describe various characters by many researchers.

They present three algorithms to extract features but each of them has its limitations in recognizing some of the characters. In , it is found that the Position of Peaks algorithm can recognize English numbers successfully. However, it cannot distinguish between Arabic numbers 0, 1, and 6. In , a Pixel Density method is presented, which relies on processing pixels along vertical and horizontal lines taken across the character to achieve recognition. In , the proposed algorithm develops the line containing a feature or a set of combined features after the process of quantization, which can identify a specific character in the different training datasets. In , a hybrid license plate extraction method based on the edge statistics and morphology can detect the region of license plate quickly and accurately. In , a vertical traverse density (VTD) vector and horizontal traverse density (HTD) vector are proposed to describe each character object. This method is convenient to achieve recognition and it is less time-consuming; however, due to the similar structures in different characters, the proposed algorithm has difficulty distinguishing between T and L, Z and E, and some other groups. In summary, a robust character recognition algorithm based on an efficient feature extraction method further needs to be proposed to improve license plate recognition.

In order to overcome the problem that the features extracted from the characters cannot describe the detail clearly, which leads to error recognition, and that the number of features extracted is so large that its implementation costs much time, this paper proposes a robust license plate detection and character recognition algorithm based on a novel combined feature extraction model and BPNN. The combined feature extraction method uses VTD, HTD features, and edge distance features as the learning and training samples of the VPLR classifier. Compared with the traditional license plate detection and character recognition method, it has the following features and advantages:

1) A robust license plate detection and character recognition algorithm based on a novel combined feature extraction model and BPNN has been proposed in order to improve recognition efficient of VPLR algorithm.
2) The new combined feature extraction model consists of three sets of features: two sets of traverse density features (VTD, HTD) and a set of edge distance features, which contains more information for the neural network training.
3) The proposed algorithm can effectively determine and recognize variable license plates and has a good compatibility to regional difference under both weaker illumination and complex backgrounds.

In the further topics the license detection and character segmentation algorithm using pre-processing, accurately license plate positioning, and histogram of transformation numbers projection is proposed.

1.1 PROJECT OBJECTIVE:

The main objective of the project is to provide a robust and efficient method to recognize and detect the license plate of a vehicle. As it is a tedious process , most of the times , the task of detecting illegally parked vehicles has been left largely to the human operators of surveillance systems. The detection of Indian vehicles by their number plates is the most interesting and challenging research topic from past few years.

1.2 RELATED WORK

As the key step in a license plate recognition (LPR) system, LPD methods have made great strides in recent years. The following briefly reviews several LPD methods. Based on the observation that vehicle license plates are characterized by abundant edge information, many methods exploit edge information for license plate detection. In , the magnitude of the vertical gradients is used to detect candidate license plate regions. Shapiro et al. applied Robert's edge operator to emphasize vertical edges and used the projection of vertical edges to detect license plates. Zheng proposed a license plate extraction method that searches for a license plate in a convolution output image using a rectangular shift window. Although this method is sensitive to window size, only a single license plate can be detected in any given image. Jia et al. proposed a region-based method for LPD that uses the mean-shift approach to segment a color vehicle image and uses edge density information for license plate verification. Anagnostopoulos et al. proposed an adaptive image segmentation technique to accelerate license plate detection. In , a block-based edge density prediction method was used to find candidate license plate regions, and a voting method based on multiple features was used for license plate verification. Although the detection step of this method is fast, its location accuracy primarily depends on the block size. Lalimi et a modified the region-based method of and used morphological filtering to extract candidate regions. Ghaili et al. proposed a vertical edge detection algorithm to speed up LPD methods. However, the improved computational efficiency is achieved at the cost of reduced edge information. In , edge clustering was exploited for license plate localization. Wang et al. used gradient information and a trained cascade detection model for license plate detection. The connection of character regions is another import of Chinese license plates. These MSER-based methods can achieve high localization accuracy in relatively simple scenes. However, they have difficulty detecting character regions in more complex ones, e.g., scenes in which some areas of the license plate are contaminated. The morphology technique , an important tool that is widely used in image processing tasks such as salient region detection and object segmentation , has also been successfully applied for license plate detection by many authors. The morphology technique is typically used to detect the structural information of license plates. Hsieh et al. used the differences between a 7×1 open operator and a 7×1 closed operator to locate license plates. In, a morphology gradient method for extracting license plate candidates was introduced that achieves an impressive average extraction ratio of 96.6%. However, the morphology technique is time consuming and is not suitable for license plate detection against complex backgrounds. A number of previous approaches have extensively exploited color features for LPD, based on the observation that a license plate usually exhibits a regular color appearance of both its background and its characters. In , a neural network was applied to extract color features from the hue, saturation and lightness channels separately. Kim et al. proposed combining color and texture features for the detection of license plates in images. In , Tian presented a license plate localization method based on a fixed color pair for the characters and background regions of a license plate. In , an edge-based and color-aided algorithm for license plate detection was proposed.

1.3 FRAME WORK OF LICENSE PLATE DETECTION APPROACH

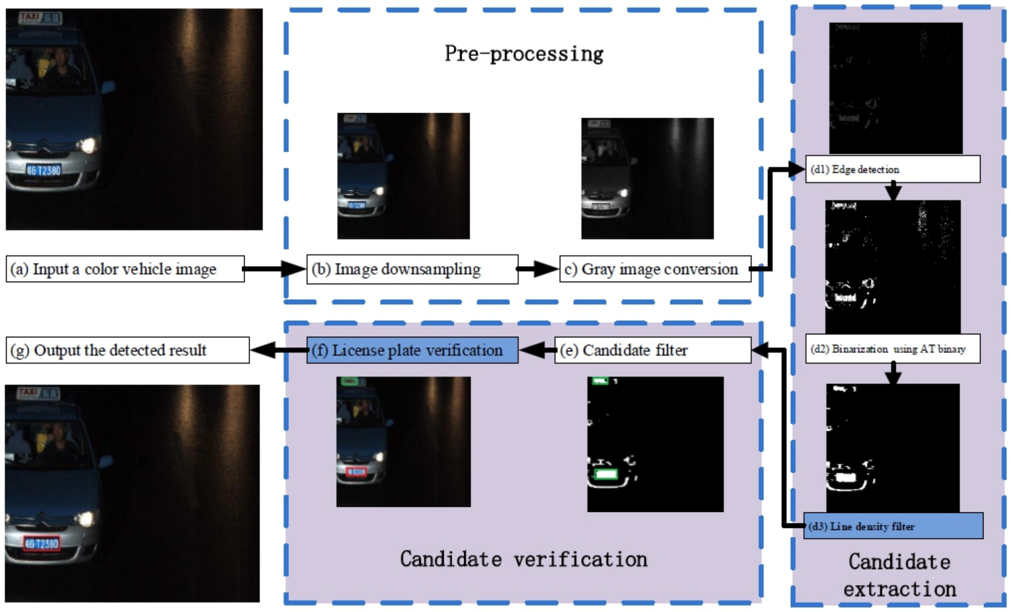


Figure.1.1. Framework of license plate detection approach

1.4 SCOPE OF THE PROJECT

The scope of this project is to develop real time number plate recognition system which can be implemented in vehicle tracking, traffic monitoring, automatic payment of tolls on highways or bridges, surveillance systems, tolls collection points, and parking management systems.We proposed a robust and efficient method for license plate recognition and detection which consists of three steps mainly, image pre-processing , candidate extraction and license plate verification.A simple yet effective image downscaling method is proposed for use in the image pre-processing step, this method is able to substantially decrease the run-time complexity of license plate localization without sacrificing detection accuracy compared with that achieved using the original image.

CHAPTER 2

PRE-PROCESSING

2.1 INTRODUCTION:

The original color images for license plate detection and recognition are generally captured at a high resolution (e.g.,1082×728), which ensures that the small license plates and the even smaller characters on them can be processed and recognized using computer vision algorithms. However, this high resolution also imposes a high computational cost for detecting the license plate in an image. To address this issue, one suggestion might be to downscale the input image for license plate detection. Unfortunately, the downscaling operation may result in a loss of information and lead to a decrease in license plate detection performance; for this reason, most previously developed methods do not perform image downscaling as part of the license plate detection task. Thus, a fundamental problem to be addressed is "how to balance detection accuracy and run-time efficiency for license plate detection".

In many cases, the step image pre-processing has been ignored, but it is one of the most important steps in the field of image processing. Here in this study the input images have been preprocessed by using (i) Image re-sizing (ii) RGB to Gray scale conversion in order to get better accuracy rate in the segmentation stage. For re-sizing the image the aspect ratio of it has been preserved, by calculating the number of columns and by specifying the number of rows. In this case, the number of rows has been specified to 400. After that, the images have been converted to gray scale. The reason is the color of RGB image is not stable as a result the currently available solutions fail to provide a higher accuracy rate for the images that contain natural scene. As one of the main purposes of this study is to detect the license plate from images that contain natural scene, therefore, the gray scale conversion process has a lot of significance. Furthermore, the median filtering method of noise removal has also been considered as one of the main parts of the pre-processing stage. Since the images are often get corrupted by noise. The reason is digital images are subjected to a wide variety of distortions during image acquisition.

In this , we propose a novel image downscaling method for license plate detection that can substantially reduce the image size without incurring an obvious decrease in performance compared with that achieved when using the original image. This method is based on the following observations. First, the width of a license plate is obviously greater than its height. Second, the characters on a license plate are printed in the horizontal direction. Thus, we define different scale factors for the vertical and horizontal directions to downscale the original image, i.e.,

$$\mathrm{w_s} = \frac{w_i}{d_w} \tag{2.1}$$

$$h_s = \frac{h_i}{d_h} \tag{2.2}$$

where w_i and h_i denote the width and height, respectively, of the original image, whereas w_s and h_s represent the corresponding downscaled dimensions, and d_w and d_h (s.t. $d_h < d_w$) are the downscaling factors for width and height, respectively. Note that a larger scale factor d_w should be assigned to downscale the original image in the horizontal direction for the following two reasons. First, we can compress more image data in the horizontal direction because the width of a license plate is obviously greater than its height. Second, a larger scale factor in the horizontal direction makes the characters on the license plate more compact, which allows the subsequently applied candidate region extraction method to group all characters into a single region. In our experiments, d_w and d_h were set to 3 and 2, respectively. Using these well-defined scale factors, we exploit bi-linear interpolation for image downscaling, in which each output pixel value is computed as a weighted average of the nearest pixels in a 2×2 neighborhood.

Image acquisition refers to the capturing of the number plate by using a camera. However, the image is not just only the number plate when capturing but some background and some unnecessary items. Therefore turning the original colourful image into grayscale as colour is not necessary in this section. Grayscale conversion is the process of searching the average colour by multiplying the number of RGB of the particular pixel with a specified equation which is*0.114 *R+0.587*G+0.299*B.* Different colour components are having their own number to be multiplied to achieve the best result when turning into grayscale. Turning the input image into grayscale format will also help the image binarization later.

Therefore recovering the original image from a noisy data is essential. Moving on to the next stages of pre-processing the input image has been inverted so that the accuracy rate of the SCW technique improves. As stated in, the SCW method does not guarantee to detect the ROI in the case of vehicles that have a dark background and white foreground or characters. The resulted image after the preprocessing stage has been shown. In the transmission process, image is affected by many factors, which leads to some differences with the original image. The image pre-processing is that the useless information such as noise is removed from the acquired image information and the useful information is strengthened by enhancing the boundary effect and increasing the boundary brightness. Thus itis helpful to process the image further

<u>2.2 IMAGE DOWNSAMPLING</u>

Rescaling or resampling is the technique used to create a new version of an image with a different size. Increasing the size of the image is called up sampling, and reducing the size of an image is called down sampling. It turns out that these operations are not lossless. Conventionally, digital color images are represented by setting specific values of the color space coordinates for each pixel. Color spaces with decoupled luminance and chrominance coordinates (YUV type) allow the number of bits required for acceptable color description of an image to be reduced. In computer graphics and digital imaging, image scaling refers to the resizing of a digital image. In video technology, the magnification of digital material is known as upscaling or resolution enhancement.

When scaling a vector graphic image, the graphic primitives that make up the image can be scaled using geometric transformations, with no loss of image quality. When scaling a raster graphics image, a new image with a higher or lower number of pixels must be generated. In the case of decreasing the pixel number (scaling down) this usually results in a visible quality loss. From the standpoint of digital signal processing, the scaling of raster graphics is a two-dimensional example of sample-rate conversion, the conversion of a discrete signal from a sampling rate (in this case the local sampling rate) to another.

Image scaling can be interpreted as a form of image resampling or image reconstruction from the view of the nyguist sampling theorem. According to the theorem, down sampling to a smaller image from a higher-resolution original can only be carried out after applying a suitable 2D anti-aliasing filter to prevent aliasing artifacts. The image is reduced to the information that can be carried by the smaller image.

The idea behind this approach is to set individual value of luminance component to each pixel, while assigning the same color (chrominance components) to certain groups of pixels (sometimes called macro pixels) in accordance with some specific rules.

This process is called down sampling and there are different sampling formats depending on the underlying scheme. When data is removed the image also degrades to some extent, although not nearly as much as when you up sample. By removing this extra data (down sampling) this results in a much smaller file size.

2.2.1 EFFECTS OF IMAGE DOWN SAMPLING:

Many images you see on the Internet today have undergone compression for various reasons. Image compression can benefit users by having pictures load faster and webpages use up less space on a Web host. Image compression does not reduce the physical size of an image but instead compresses the data that makes up the image into a smaller size.

SIZE REDUCTION:

File size reduction remains the single most significant benefit of image compression. Depending on what file type you're working with, you can continue to compress the image until it's at your desired size. This means the image takes up less space on the hard drive and retains the same physical size, unless you edit the image's physical size in an image editor. This file size reduction works wonderfully for the Internet, allowing webmasters to create image-rich sites without using much bandwidth or storage space.

SLOW DEVICES:

Some electronic devices, such as computers or cameras, may load large, uncompressed images slowly. CD drives, for example, can only read data at a specific rate and can't display large images in real time. Also, for some webhosts that transfer data slowly, compressed images remain necessary for a fully functional website. Other forms of storage mediums, such as hard drives, will also have difficulty loading uncompressed files quickly. Image compression allows for the faster loading of data on slower devices.

DEGRADATION:

When you compress an image, sometimes you will get image degradation, meaning the quality of the image has declined. If saving a GIF or PNG file, the data remains even though the quality of the image has declined. If you need to show a high-resolution image to someone, large or small, you will find image compression as a disadvantage.

DATA LOSS:

With some common file types, such as JPEG, when an image shrinks in size the compression program will discard some of the photo's data permanently. To compress these images, you need to ensure you had an uncompressed backup before starting. Otherwise, you will lose the high quality of the original uncompressed image permanently.

APPLICATIONS:

For image compression applications the above discussed Lossy Compression and Lossless compression methods are used. Various techniques and algorithms are used in compressing images by different image compression applications. Among these two methods, selection of a method to compress images depends on the quality of the output that is expected by the users. When a very highquality output is expected without any loss of data by users from image compression applications then lossless compression technique can be used by the application. Lossless compression techniques are used when a high degree of accuracy is needed. In some cases where the quality is not that much important and can be compromised, lossy compression technique can be used. When this compression technique is used there will be very little bit of loss in quality and that loss cannot be visible to the human eye at most of times. It can be used in such applications where a little compromise on quality of image is tolerable. The areas where image compression is used are in television broadcasting, Remote sensing via satellite, for military communication systems through radars, Tele conferencing systems, communications systems built through computers, Facsimile transmission medical images in computer tomography magnetic resonance imaging, capturing and transmitting satellite images, geological surveys, weather reporting applications.

2.3 GRAY SCALE IMAGE CONVERSION:

In digital photography, computer-generated imagery, and colourimetry, a greyscale or image is one in which the value of each pixel is a single sample representing only an amount of light; that is, it carries only intensity information. Grayscale images, a kind of black-and-white or gray monochrome, are composed exclusively of shades of gray. The contrast ranges from black at the weakest intensity to white at the strongest.Greyscale images are distinct from one-bit bi-tonal black-and-white images, which, in the context of computer imaging, are images with only two colours: black and white (also called bi-level or binary images).

Greyscale images have many shades of gray in between.Greyscale images can be the result of measuring the intensity of light at each pixel according to a particular weighted combination of frequencies (or wavelengths), and in such cases they are monochromatic proper when only a single frequency (in practice, a narrow band of frequencies) is captured. The frequencies can in principle be from anywhere in the electromagnetic spectrum (e.g. infrared, visible light, ultraviolet, etc.). A colourimetric (or more specifically photometric) greyscale image is an image that has a defined greyscale colour space, which maps the stored numeric sample values to the achromatic channel of a standard colour space, which itself is based on measured properties of human vision.If the original colour image has no defined colour space, or if the greyscale image is not intended to have the same human-perceived achromatic intensity as the colour image, then there is no unique mapping from such a colour image to a greyscale image.The intensity of a pixel is expressed within a given range between a minimum and a maximum, inclusive.

This range is represented in an abstract way as a range from 0 (or 0%) (total absence, black) and 1 (or 100%) (total presence, white), with any fractional values in between. This notation is used in academic papers, but this does not define what "black" or "white" is in terms of colourimetry. Sometimes the scale is reversed, as in printing where the numeric intensity denotes how much ink is employed in half toning, with 0% representing the paper white (no ink) and 100% being a solid black (full ink).

In computing, although the greyscale can be computed through rational numbers, image pixels are usually quantized to store them as unsigned integers, to reduce the required storage and computation. Some early greyscale monitors can only display up to sixteen different shades, which would be stored in binary form using 4 bits. But today greyscale images (such as photographs) intended for visual display (both on screen and printed) are commonly stored with 8 bits per sampled pixel. This pixel depth allows 256 different intensities (i.e., shades of grey) to be recorded, and also simplifies computation as each pixel sample can be accessed individually as one full byte. However, if these intensities were spaced equally in proportion to the amount of physical light they represent at that pixel (called a linear encoding or scale), the differences between adjacent dark shades could be quite noticeable as banding artifacts, while many of the lighter shades would be "wasted" by encoding a lot of perceptually-indistinguishable increments. Therefore, the shades are instead typically spread out evenly on a gamma-compressed nonlinear scale, which better approximates uniform perceptual increments for both dark and light shades, usually making these 256 shades enough (just barely) to avoid noticeable increments.

Technical uses (e.g. in medical imaging or remote sensing applications) often require more levels, to make full use of the sensor accuracy (typically 10 or 12 bits per sample) and to reduce rounding errors in computations. Sixteen bits per sample (65,536 levels) is often a convenient choice for such uses, as computers manage 16-bit words efficiently. The TIFF and PNG (among other) image file formats support 16-bit greyscale natively, although browsers and many imaging programs tend to ignore the low order 8 bits of each pixel. Internally for computation and working storage, image processing software typically uses integer or floating-point numbers of size 16 or 32 bits.The first step is the capturing of an image using electronic devices such as optical (digital/video) camera; webcam etc can be used to capture the acquired images. For this project, vehicle images will be taken with a Panasonic FX/Nikon digital camera. In this project pre-captured image will taken. The images will be stored as colour JPEG format on the camera. Next, we might proceed in using the Matlab function to convert the vehicle JPEG image into gray scale format Input of this system is the image captured by a camera placed at a distance of 1-2 meters away from the vehicle Grayscale conversion is the process of searching the average colour by multiplying the number of RGB of the particular pixel. Different colour components are having their own number to be multiplied to achieve the best result when turning into grayscale. Turning the input image into grayscale format will also help the image binarization later. Binarization is the process of applying a threshold value to change the grey value having the range from 0 (most black) to 255 (most white) to the binary value which contain only 0 or 1 . Taking a threshold value 80 as example, if the pixel grey value is below 80, then the pixel will be set with 0, if the pixel value is greater than 80, then the value 1will be set to that pixel. By doing this, it is easier to differentiate the foreground and also background of an image for the purpose of recognition later.

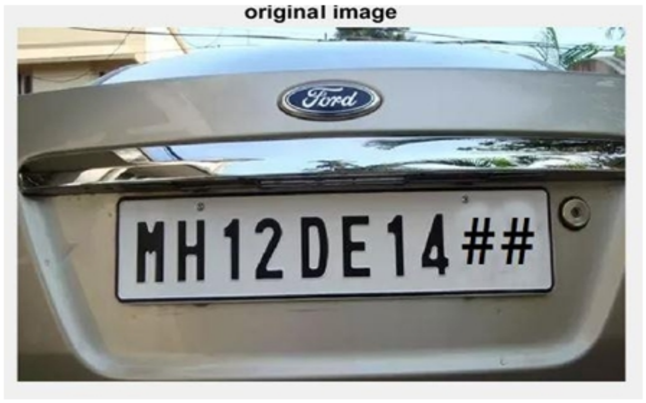


Figure.2.1 original image

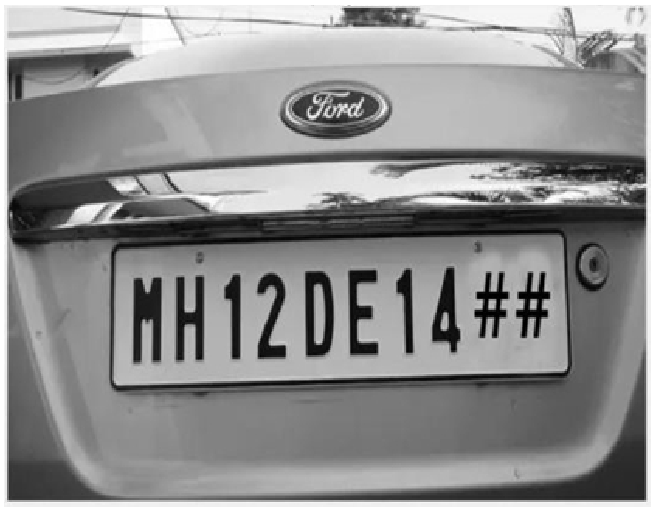


Figure.2.2 Gray scaled image

A grayscale (or gray level) image is simply one in which the only colors are shades of gray. The reason for differentiating such images from any other sort of color image is that less information needs to be provided for each pixel. In fact a `gray' color is one in which the red, green and blue components all have equal intensity in RGB space, and so it is only necessary to specify a single intensity value for each pixel, as opposed to the three intensities needed to specify each pixel in a full color image.Often, the grayscale intensity is stored as an 8-bit integer giving 256 possible different shades of gray from black to white. If the levels are evenly spaced then the difference between successive gray levels is significantly better than the gray level resolving power of the human eye.Grayscale images are very common, in part because much of today's display and image capture hardware can only support 8-bit images. In addition, grayscale images are entirely sufficient for many tasks and so there is no need to use more complicated and harder-to-process colour images. It is a range of monochromatic shades from black to white. Therefore, a grayscale image contains only shades of gray and no colour. While digital images can be saved as grayscale (or black and white) images, even colour images contain grayscale information.To convert an color image into a grayscale image. There are two methods to convert it. Both has their own merits and demerits. The methods are:

- Average method
- Weighted method or luminosity method

Average method is the most simple one. You just have to take the average of three colours. Since its an RGB image, so it means that you have add r with g with b and then divide it by 3 to get your desired grayscale image.

$$grayscale\ image = (R + G + B)/3$$

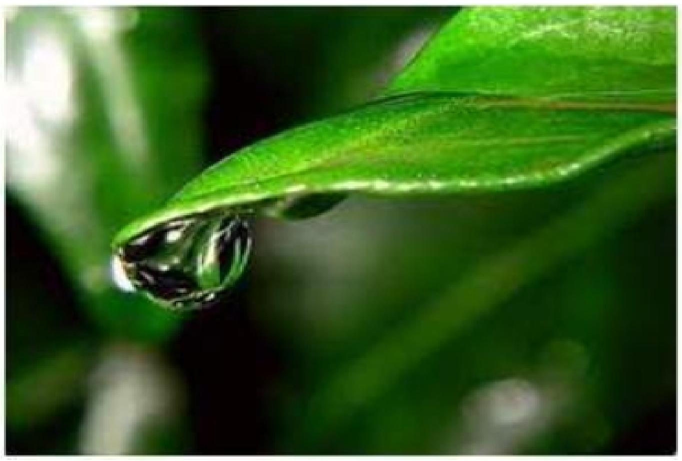


Figure 2.3: original input image for gray scaling

If you have an color image like the image shown above and you want to convert it into grayscale using average method. The following result would appear.But the results were not as expected. We wanted to convert the image into a grayscale, but this turned out to be a rather black image.This problem arise due to the fact, that we take average of the three colours. Since the three different colours have three different wavelength and have their own contribution in the formation of image

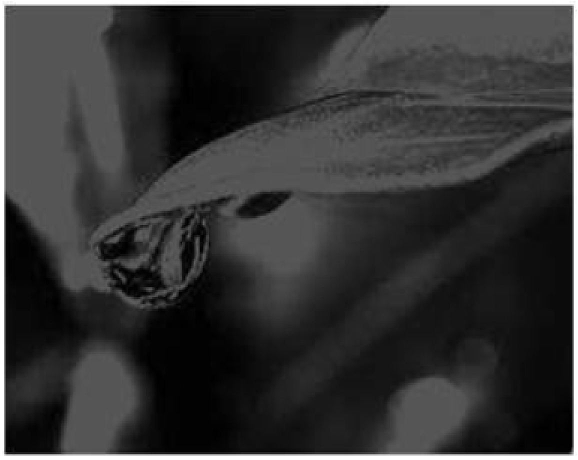


Figure.2.4 Gray scaled image by average method

So we have to take average according to their contribution, not done it averagely using average method. Right now what we are doing is this,33% of Red, 33% of Green, 33% of Blue

We are taking 33% of each, that means, each of the portion has same contribution in the image. But in reality that's not the case. The solution to this has been given by luminosity method.You have seen the problem that occur in the average method. Weighted method has a solution to that problem. Since red colour has more wavelength of all the three colours, and green is the colour that has not only less wavelength then red color but also green is the colour that gives more soothing effect to the eyes.It means that we have to decrease the contribution of red colour, and increase the contribution of the green colour, and put blue colour contribution in between these two.So, the new equation that form is:

$$New\ grayscale\ image = ((0.3 * R) + (0.59 * G) + (0.11 * B))$$

According to this equation, Red has contribute 30%, Green has contributed 59% which is greater in all three colors and Blue has contributed 11%.Applying this equation to the image, we get the image as in figure 2.5.

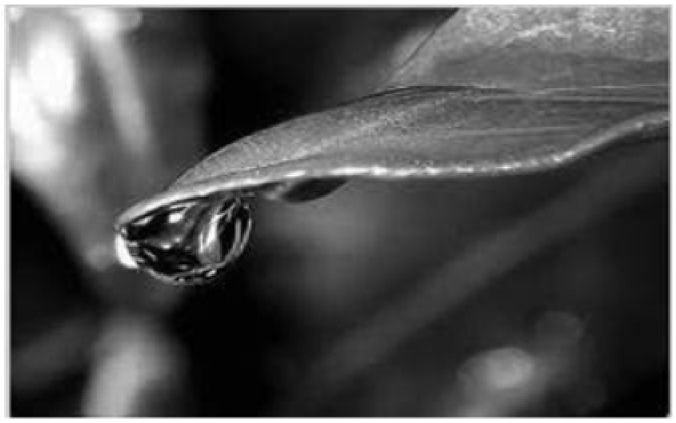


Figure 2.5 : Grayscale Image by weighted method

CHAPTER 3
EDGE DETECTION USING EXTENDED SOBEL OPERATOR

3.1 INTRODUTION OF EDGE DETECTION:

Edge detection is a technique is used to find the boundaries of an object within the image.it works by detecting discontinuous in brightness. Edge detection is used for image segmentation and data extraction in areas such as image processing, computer vision, and machine vision. Edge detection includes a variety of mathematical methods that aim at identifying points in a digital image at which the image brightness changes sharply or, more formally, has discontinuities. The points at which image brightness changes sharply are typically organized into a set of curved line segments termed ‘edges’. The same problem of finding discontinuities in one-dimensional signals is known as step detection and the problem of finding signal discontinuities over time is known as change detection. The purpose of detecting sharp changes in image brightness is to capture important events and changes in properties of the world. It can be shown that under rather general assumptions for an image formation model, discontinuities in image brightness are likely to correspond to:

- discontinuities in depth,
- discontinuities in surface orientation,
- changes in material properties and
- variations in scene illumination.

In the ideal case, the result of applying an edge detector to an image may lead to a set of connected curves that indicate the boundaries of objects, the boundaries of surface markings as well as curves that correspond to discontinuities in surface orientation.

Thus, applying an edge detection algorithm to an image may significantly reduce the amount of data to be processed and may therefore filter out information that may be regarded as less relevant, while preserving the important structural properties of an image.

If the edge detection step is successful, the subsequent task of interpreting the information contents in the original image may therefore be substantially simplified. However, it is not always possible to obtain such ideal edges from real life images of moderate complexity.

Edges extracted from non-trivial images are often hampered by fragmentation, meaning that the edge curves are not connected, missing edge segments as well as false edges not corresponding to interesting phenomena in the image – thus complicating the subsequent task of interpreting the image data.

Edge detection is one of the fundamental steps in image processing, image analysis, image pattern recognition, and computer vision techniques.The edges extracted from a two-dimensional image of a three-dimensional scene can be classified as either viewpoint dependent or viewpoint independent. A viewpoint independent edge typically reflects inherent properties of the three-dimensional objects, such as surface markings and surface shape. A viewpoint dependent edge may change as the viewpoint changes, and typically reflects the geometry of the scene, such as objects occluding one another.

A typical edge might for instance be the border between a block of red colour and a block of yellow. In contrast a line (as can be extracted by a ridge detector) can be a small number of pixels of a different colour on an otherwise unchanging background. For a line, there may therefore usually be one edge on each side of the line.

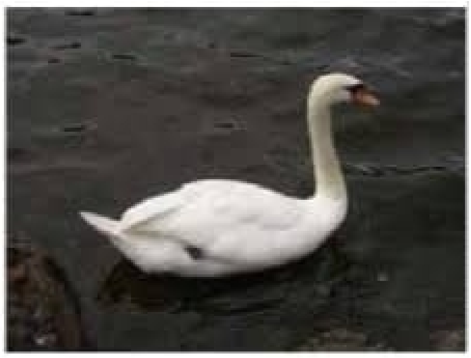


Figure 3.1: input image for edge detection

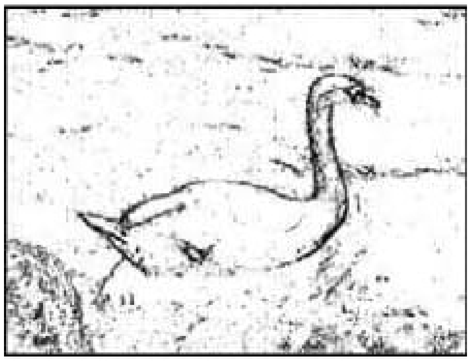


Figure 3.2: edge detected image

3.2 TYPES OF EDGE DETECTION:

3.2.1 CANNY EDGE DETECTION:

The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. It was developed by John F. Canny in 1986. Canny also produced a *computational theory of edge detection* explaining why the technique works.

Canny edge detection is a technique to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed. It has been widely applied in various computer vision systems. Canny has found that the requirements for the application of edge detection on diverse vision systems are relatively similar. Thus, an edge detection solution to address these requirements can be implemented in a wide range of situations. The general criteria for edge detection include:

1. Detection of edge with low error rate, which means that the detection should accurately catch as many edges shown in the image as possible
2. The edge point detected from the operator should accurately localize on the Center of the edge.
3. A given edge in the image should only be marked once, and where possible, image noise should not create false edges.

To find edges by separating noise from the image before find edges of image the Canny is a very important method. Canny method is a better method without disturbing the features of the edges in the image afterwards it applying the tendency to find the edges and the serious value for threshold.The algorithmic steps are as follows:

- Convolve image f(r, c) with a Gaussian function to get smooth image

  *f^(r, c). f^(r, c)=f(r,c)*G(r,c,6)*

- Apply first difference gradient operator to compute edge strength then edge magnitude and direction are obtain as before.
- Apply non-maximal or critical suppression to the gradient magnitude.
- Apply threshold to the non-maximal suppression image.

Unlike Roberts and Sobel, the Canny operation is not very susceptible to noise. If the Canny detector worked well it would be superior.

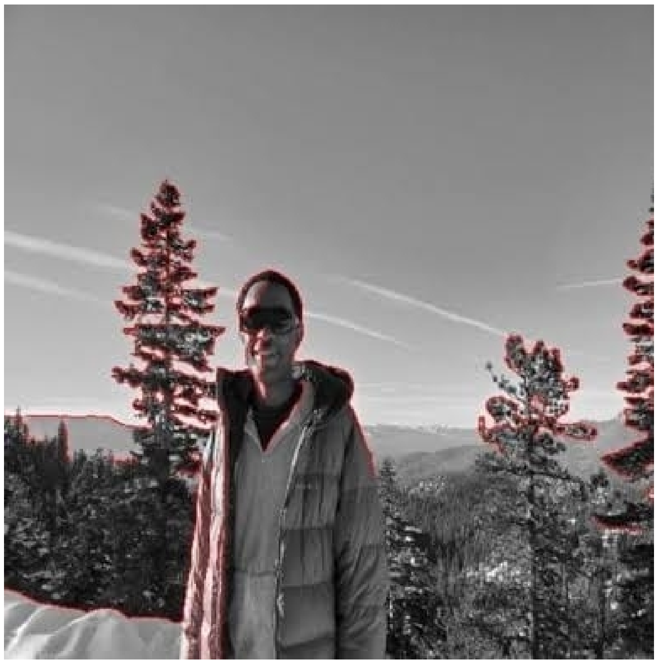


Figure 3.3: input image for canny detection

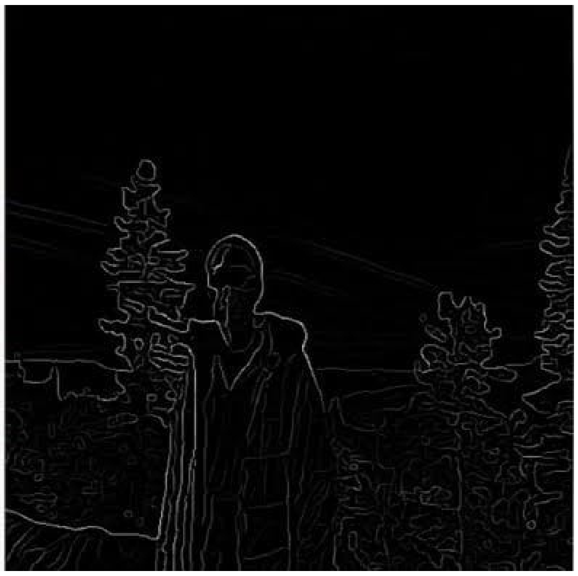
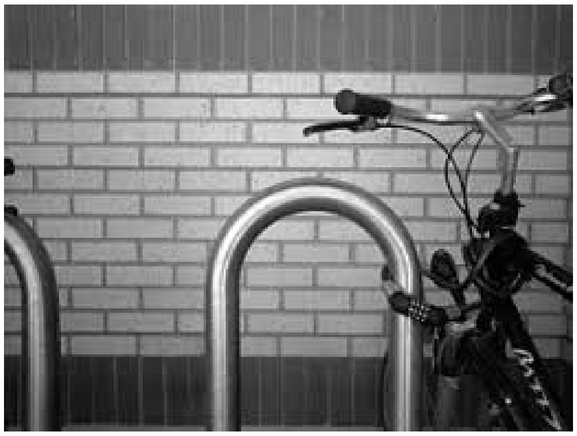


Figure 3.4: output image produced by canny edge detection

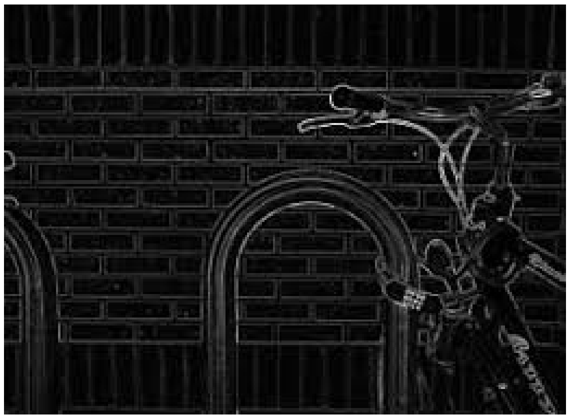
3.2.2 PREWITT EDGE DETECTION:

The Prewitt operator is used in image processing, particularly within edge detection algorithms. Technically, it is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Prewitt operator is either the corresponding gradient vector or the norm of this vector. The Prewitt operator is based on convolving the image with a small, separable, and integer valued filter in horizontal and vertical directions and is therefore relatively inexpensive in terms of computations like Sobel and Kayyali operators. On the other hand, the gradient approximation which it produces is relatively crude, in particular for high frequency variations in the image. The Prewitt operator was developed by "Judith M. S. Prewitt".

The Prewitt edge detection is proposed by Prewitt in 1970 (Rafael Conzalez). To estimate the magnitude and orientation of an edge Prewitt is a correct way. Even though different gradient edge detection wants a quite time-consuming calculation to estimate the direction from the magnitudes in the x and y-directions, the compass edge detection obtains the direction directly from the kernel with the highest response. It is limited to 8 possible directions; however, knowledge shows that most direct direction estimates are not much more perfect. This gradient based edge detector is estimated in the 3x3 neighbourhood for eight directions. All the eight convolution masks are calculated. One complication mask is then selected, namely with the purpose of the largest module.



(a)



(b)

Figure 3.5: Prewitt operator(a): Gray scale image.

(b): Prewitt output image.

Prewitt detection is slightly simpler to implement computationally than the Sobel detection, but it tends to produce somewhat noisier results.

3.2.3 ROBERT EDGE DETECTION:

The Roberts edge detection is introduced by Lawrence Roberts (1965). It performs a simple, quick to compute, 2-D spatial gradient measurement on an image. This method emphasizes regions of high spatial frequency which often correspond to edges. The input to the operator is a grayscale image the same as to the output is the most common usage for this technique. Pixel values in every point in the output represent the estimated complete magnitude of the spatial gradient of the input image at that point.

3.2.4 SOBEL OPERATOR:

The Sobel operator [8] , sometimes called the Sobel–Feldman operator or Sobel filter, is used in image processing and computer vision, particularly within edge detection algorithms where it creates an image emphasising edges. It is named after Irwin Sobel and Gary Feldman, colleagues at the Stanford Artificial Intelligence Laboratory (SAIL). Sobel and Feldman presented the idea of an "Isotropic 3x3 Image Gradient Operator" at a talk at SAIL in 1968.

It works by calculating the gradient of image intensity at each pixel within the image. It finds the direction of the largest increase from light to dark and the rate of change in that direction. The result shows how abruptly or smoothly the image changes at each pixel, and therefore how likely it is that that pixel represents an edge. It also shows how that edge is likely to be oriented. The result of applying the filter to a pixel in a region of constant intensity is a zero vector. The result of applying it to a pixel on an edge is a vector that points across the edge from darker to brighter values.

Technically, it is a discrete differentiation operator, computing an approximation of the gradient of the image intensity function. At each point in the image, the result of the Sobel–Feldman operator is either the corresponding gradient vector or the norm of this vector. The Sobel–Feldman operator is based on convolving the image with a small, separable, and integer-valued filter in the horizontal and vertical directions and is therefore relatively inexpensive in terms of computations. On the other hand, the gradient approximation that it produces is relatively crude, in particular for high-frequency variations in the image.

3.2.5 EXTENDED SOBEL OPERATOR:

The Sobel method of edge detection [8] for image segmentation finds edges using the Sobel approximation to the derivative. It precedes the edges at those points where the gradient is highest. The Sobel technique performs a 2-D spatial gradient quantity on an image and so highlights regions of high spatial frequency that correspond to edges. In general, it is used to find the estimated absolute gradient magnitude at each point in 'n' input grayscale image. In conjecture at least the operator consists of a pair of 3x3 complication kernels as given away in under table. One kernel is simply the other rotated by 90°. This is very alike to the Roberts cross operator.

The operator uses two 3×3 kernels which are convolved with the original image to calculate approximations of the derivatives – one for horizontal changes, and one for vertical. If we define A as the source image, and $G_x$and G_y are two images which at each point contain the horizontal and vertical derivative approximations respectively,

The computation is as follows:

$$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * A \qquad (4.1)$$

$$G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A \qquad (4.2)$$

The operator uses two 3×3 kernels which are convolved with the original image to calculate approximations of the derivatives – one for horizontal changes, and one for vertical. If we define A as the source image, and G_x and G_y are two images which at each point contain the horizontal and vertical derivative approximations respectively, the computations are as follows:

$$G_x = \begin{bmatrix} +1 \\ +2 \\ +1 \end{bmatrix} * ([+1 \quad 0 \quad -1] * A) \qquad (4.3)$$

$$G_y = \begin{bmatrix} +1 \\ 0 \\ -1 \end{bmatrix} * ([+1 \quad +2 \quad +1] * A) \qquad (4.4)$$

Since the intensity function of a digital image is only known at discrete points, derivatives of this function cannot be defined unless we assume that there is an underlying differentiable intensity function that has been sampled at the image points. With some additional assumptions, the derivative of the continuous intensity function can be computed as a function on the sampled intensity function, i.e. the digital image. It turns out that the derivatives at any particular point are functions of the intensity values at virtually all image points. However, approximations of these derivative functions can be defined at lesser or larger degrees of accuracy.

The Sobel-Feldman operator represents a rather inaccurate approximation of the image gradient, but is still of sufficient quality to be of practical use in many applications. More precisely, it uses intensity values only in a 3×3 region around each image point to approximate the corresponding image gradient, and it uses only integer values for the coefficients which weight the image intensities to produce the gradient approximation.For edge detection, we propose a simple extension of the Sobel operator for edge density enhancement. As illustrated in Fig.3.6. $p_0$denotes the current pixel, and p_1, p_2, p_3, p_4 are the nearest neighbour pixels of p0. The edge intensity e0 for pixel p0 is defined as follow:

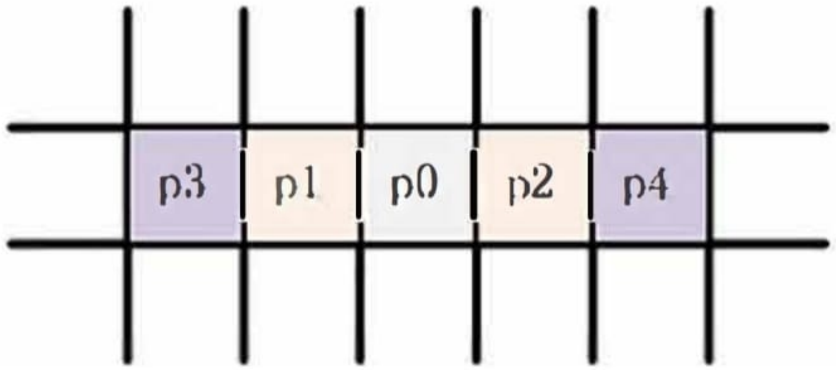


Figure 3.6: current pixel p_0 with neighbouring pixels

Edge intensity e_0 is calculated by,

$$e_0 = \begin{cases} T \; if \; d_0 \geq \; T \\ d_0 \; if \; d_0 < T \end{cases} \quad (4.5)$$

Where, T is the threshold value and d_0 is calculated by,

$$d_0 = |p_1 + p_2 - 2 * p_0| + |p_3 + p_4 - 2 * p_0 \quad (4.6)$$

Compared with the typical Sobel operator, which uses a pair of 1×3 (or 3×3) convolution masks to estimate the gradient in the vertical and horizontal directions, we use a single

1×5 mask to estimate the gradient in the vertical direction only. Experimentally, we have found that a 1×5 convolution mask is more robust for edge detection than a 1×3 mask.

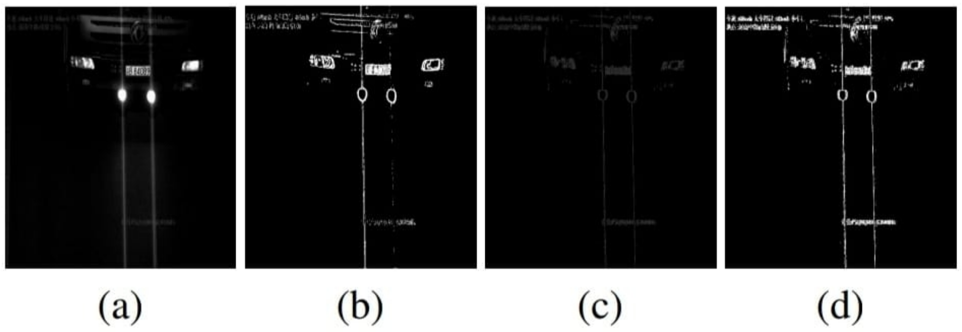
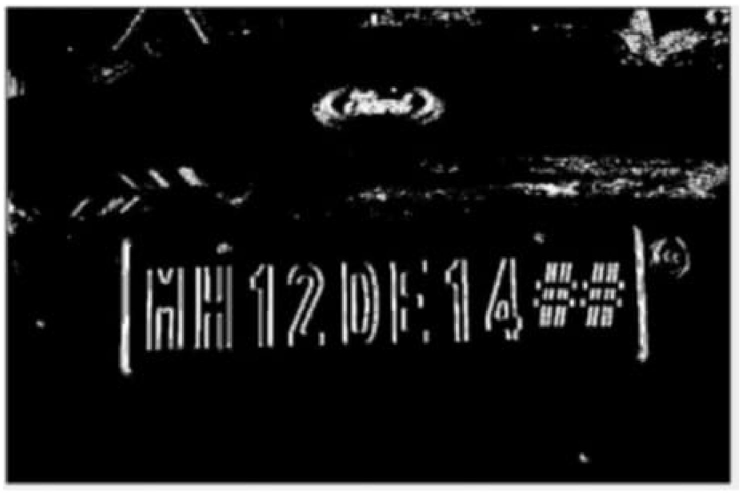


Figure 3.7: Edge detection results generated by the Sobel operator and by our proposed method

(a) grayscale input image,(b) edge detection result obtained using the Sobel operator, (c) edge detection result obtained using the proposed extension of the Sobel operator, (d)the result of applying adaptive thresholding to (c).

An example of the edge detection results generated by the Sobel operator and by our proposed method is presented for visual comparison in Fig. 3.7.

(a)



(b)

Figure 3.8: extended Sobel operator.

(a) original image. (b) extended Sobel operator output image.

CHAPTER 4
BINARIZATION USING ADAPTIVE THRESHOLDING

4.1 INTRODUCTION:

In the computer memory generally all the documents are stored in the form of gray level which has a maximum 256 different gray values from 0 to 255. Each gray value generates the different color of the gray scale palette. If some information is required from the document image, then this is required to process the action number of times. To reduce this time requirement for extracting the part of the image, binary image is more useful. Binarization is the method of converting any gray – scale image (multi tone image) into black – white image (two tone image). To perform binarization process, first find the threshold value of gray scale and check whether a pixel having a particular gray value or not. If the gray value of the pixels is greater than the threshold, then those pixels are converted into the white . Similarly if the gray value of the pixels is lesser than the threshold, then those pixels are converted into the black .The simplest approach to binarization is thresholding. In thresholding an optimal threshold value is chosen and the pixels are classified as foreground or background by comparison with this threshold value. Generally, binarization is categorized as Global and Local. Global binarization is the technique in which a single threshold value is applied to binarize the entire image. It is fast process but fails for the documents with complex background.

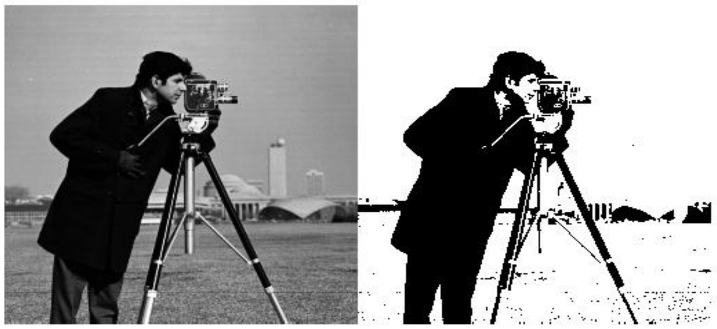


Figure.4.1:Binarization of a gray-scaled image using thresholding method

In Local binarization technique instead of single threshold for the whole image, a different value of threshold is chosen for every pixel. The threshold is chosen depending upon the neighbourhood pixels. Image thresholding segments a digital image based on a certain characteristic of the pixels (for example, intensity value). The goal is to create a binary representation of the image, classifying each pixel into one of two categories, such as “dark” or “light”. This is a common task in many image processing applications, and some computer graphics applications. For example, it is often one of the first steps in marker-based augmented reality systems ,and it has been used in high dynamic range photography.The most basic thresholding method is to choose a fixed threshold value and compare each pixel to that value.However, fixed thresholding often fails if the illumination varies spatially in the image or over time in a video stream.In order to account for variations in illumination, the common solution is adaptive thresholding. The main difference here is that a different threshold value is computed for each pixel in the image. This technique provides more robustness to changes in illumination. . We present a very simple and clear technique using integral images. Our method is easy to implement for real-time performance on a live video stream.

4.2 THRESHOLDING

With the growth of image processing applications, image segmentation has become an important part of image processing. The simplest method to segment an image is thresholding. Using the thresholding method, segmentation of an image is done by fixing all pixels whose intensity values are more than the threshold to a foreground value. The remaining pixels are set to a background value. Such technique can be used to obtain binary images from grayscale images. The conventional thresholding techniques use a global threshold for all pixels, whereas adaptive thresholding changes the threshold value dynamically over the image. We have performed a comparative study on adaptive thresholding techniques to choose the accurate method for binarizing an image based on the contrast, texture, resolution etc. of an image.

Suppose that an image, f(x,y), is composed of light objects on a dark background, and the following figure is the histogram of the image.

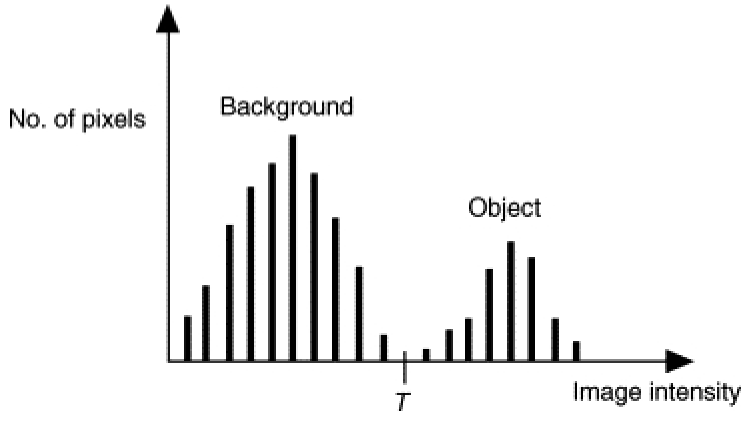


Figure.4.2:histogram of an image with dark background and light object

Then, the objects can be extracted by comparing pixel values with a threshold T. One way to extract the objects from the background is to select a threshold T that separates object from background. Any point (x,y) for which $f(x,y) > T$ is called an object point, otherwise the point is called a background point.

$$g(x,y) = \begin{cases} 1 \; if \; f(x,y) \geq T \\ 0 \; if \; f(x,y) < T \end{cases} \tag{4.1}$$

Where, $'T'$ is a constant applicable over an entire image, then the above process is called as Global thresholding.When the value of T changes over an image then that process is referred as Variable thresholding.Sometimes it is also termed as local or regional thresholding.Where, the value of T at any point (x,y) in an image depends
on properties of a neighborhood of (x,y). If T depends on the spatial coordinates (x,y) themselves, then variable thresholding is often referred to as dynamic or adaptive thresholding.

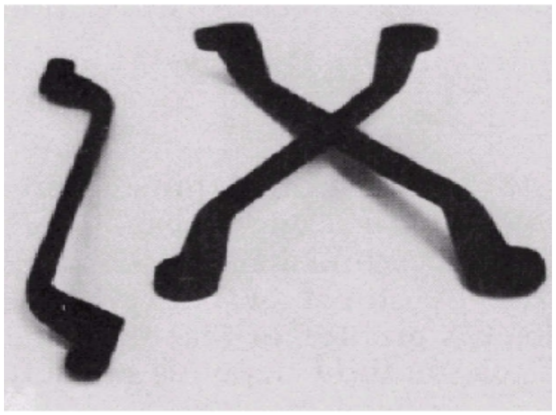


Figure.4.3 :input image for basic thresholding

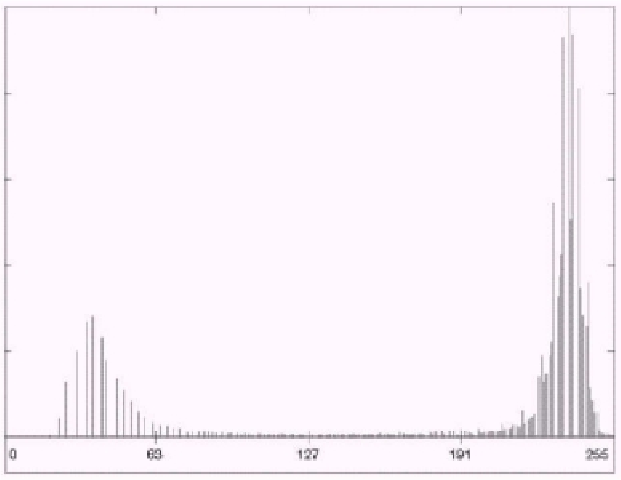


Figure.4.4 :histogram of the input image

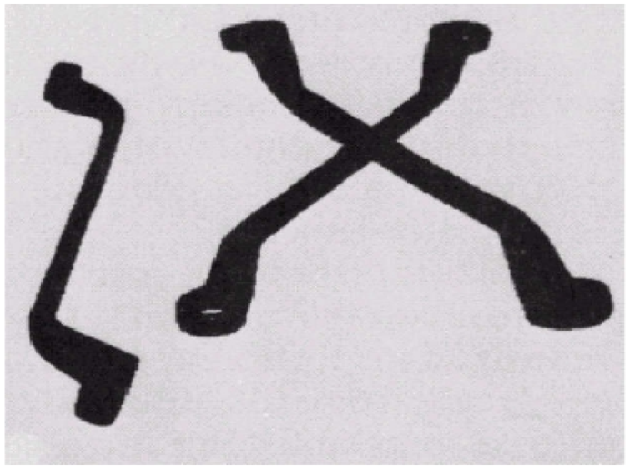
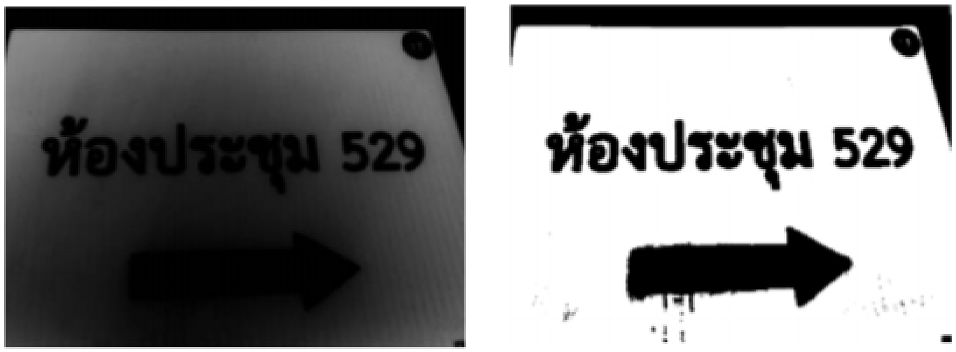


Figure.4.5 :output image after applying basic thresholding

4.3 ADAPTIVE THRESHOLDING

In image processing, Thresholding is applied to obtain binary images from grayscale images. Adaptive thresholding [12] is found to be better as compared to conventional thresholding technique. In an image, some parts remain under more shadow and often illuminations also affect the image. In conventional thresholding method, a global or standard threshold value is taken as mean value. In an image, if darker part or pixels contain a value larger than the threshold value, then that part of the image appears in the foreground. Similarly if the value is less than the threshold value, then that pixel or part appears in the background.



(a) (b)

Fig.4.6:(a) original gray-scaled image (b)resultant image after binarization using adaptive thresholding

Binary image is the resultant image of adaptive thresholding as it depicts the differences between different threshold values. The white region describes values less than threshold and black region describes values greater than threshold value.Previously, lot of work has been done on several techniques for computing the adaptive thresholding value based on contrast, texture, resolution etc. of the image. The techniques are mainly based on some parameters of the respective image to compute the threshold value. The main problem of such techniques is two class classification paradigms. The two classes are foreground and background. Following the selection of the threshold value, the image turns into a binary image. This issue demanded the threshold value being image dependent.subdivide original image into small areas. In basic adaptive thresholding it utilizes a different threshold to segment each sub-images. since the threshold used for each pixel since the threshold used for each pixel depends on the location of the pixel in terms of the sub-images, this type of thresholding is adaptive.

4.4 THE TECHNIQUE

Our adaptive thresholding technique is a simple extension of Wellner's method [Wellner 1993]. The main idea in Wellner's algorithm is that each pixel is compared to an average of the surrounding pixels. Specifically, an approximate moving average of the last s pixels seen is calculated while traversing the image. If the value of the
current pixel is 't' percent lower than the average then it is set to black, otherwise it is set to white. This method works because comparing a pixel to the average of nearby pixels will preserve hard contrast lines and ignore soft gradient changes. The advantage of this method is that only a single pass through the image is required. Wellner uses 1/8th of the image width for the value of s and 15 for the value of t.

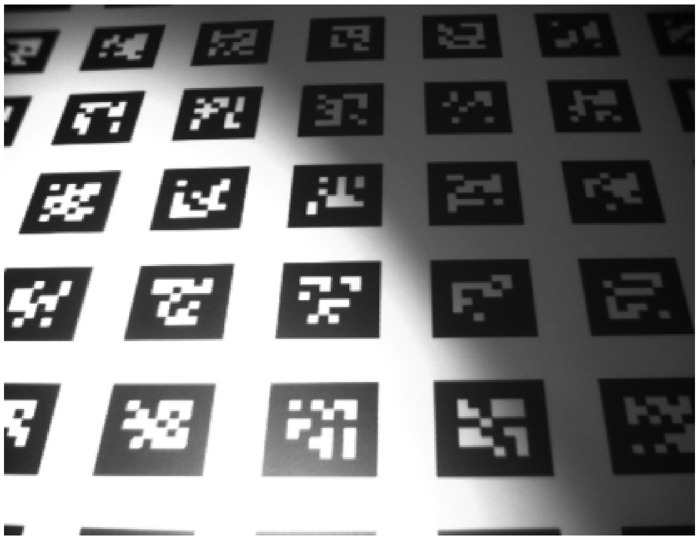


Figure.4.7: input image

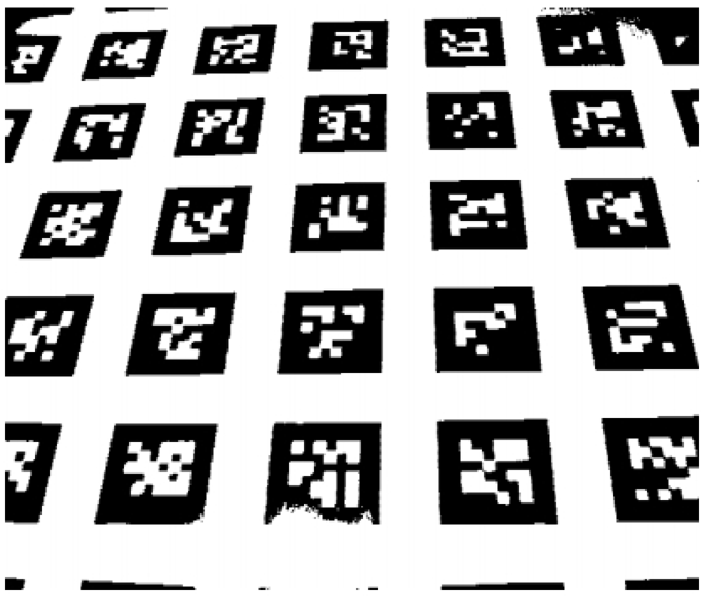


Figure.4.8:output thresholded image after applying wellner's previous method

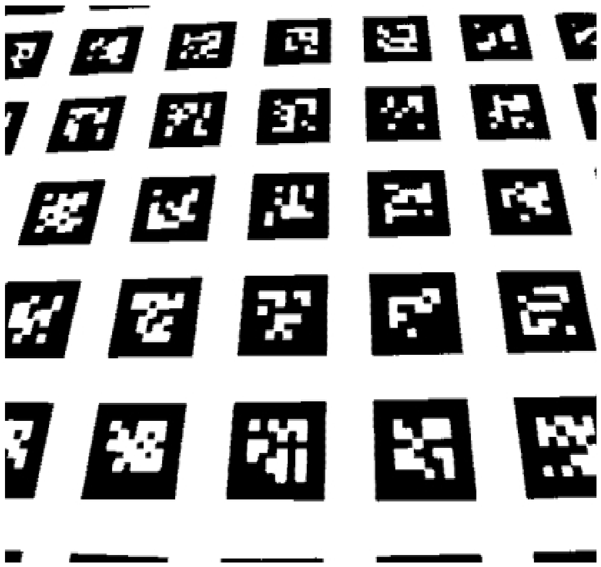


Figure.4.9 :output image after applying the improved version of wellner's method

However, a problem with this method is that it is dependent on the scanning order of the pixels. In addition, the moving average is not a good representation of the surrounding pixels at each step because the neighbourhood samples are not evenly distributed in all directions. By using the integral image (and sacrificing one additional iteration through the image), we present a solution that does not suffer from these problems. Our technique is clean, straightforward, easy to code, and produces the same output independently of how the image is processed. Instead of computing a running average of the last s pixels seen, we compute the average of an s x s window of pixels centered around each pixel. This is a better average for comparison since it considers neighbouring pixels on all sides. The average computation is accomplished in linear time by using the integral image. We calculate the integral image in the first pass through the input image. In a second pass, we compute the s x s average using the integral image for each pixel in constant time and then perform the comparison.If the value of the current pixel is t percent less than this average then it is set to black, otherwise it is set to white. The following pseudocode demonstrates our technique for input image ‘in’, output binary image ‘out’, image width ‘w’ and image height ‘h’.

<u>PSEUDOCODE:</u>

```
Procedure AdaptiveThreshold(in,out,w,h)
for i = 0 to w do
    sum← 0
    For j = 0 to h do
        sum ← sum+in[i, j]
        if i = 0 then
            intImg[i, j] ← sum
        else
            intImg[i, j] ← intImg[i-1, j] +sum
        End if
    End for
End for
for i = 0 to w do
    for j = 0 to h do
        x1 ← i- s/2
        x2 ← i+s/2
        y1 ← j - s/2
        y2 ← j +s/2
        count ← (x2 - x1)×(y2 - y1)
        sum ← intImg[x2,y2] intImg[x2,y1-1] intImg[x1-1,y2] +intImg[x1-1,y1-1]
        if (in[i, j]×count) ≤ (sum×(100- t)/100) then
            out[i, j] ← 0
        else
            out[i, j] ← 255
        end if
    end for
end for
```

4.4.1 INTEGRAL IMAGE

An integral image (also known as a summed-area table) is a tool that can be used whenever we have a function from pixels to real numbers $f(x,y)$ (for instance, pixel intensity), and we wish to compute the sum of this function over a rectangular region of the image. Examples of where integral images have been applied include texture mapping [Crow 1984], face detection in images [Viola and Jones 2004], and stereo correspondence [Veksler 2003]. Without an integral image, the sum can be computed in linear time per rectangle by calculating the value of the function for each pixel individually. However, if we need to compute the sum over multiple overlapping rectangular windows, we can use an integral image and achieve a constant number of operations per rectangle with only a linear amount of pre-processing. To compute the integral image, we store at each location, $I(x,y)$, the sum of all $f(x,y)$ terms to the left and above the pixel (x,y). This is accomplished in linear time using the following equation for each pixel (taking into account the border cases),

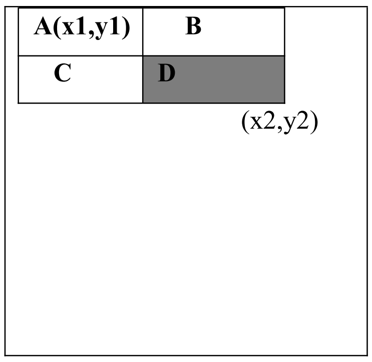
$$I(x,y) = f(x,y) + I(x-1,y) + I(x,y-1) - I(x-1,y-1) \tag{4.2}$$

3	6	9	0
0	1	2	4
5	2	3	6
2	1	4	16

(a)

3	9	18	18
3	10	21	25
8	17	31	41
10	20	38	64

(b)



(c)

Fig.4.10:integral image (a): input image values.(b): The computed integral image.c): tells us how to calculate the sum over a rectangle D using the integral image.

Figure 4.10 (a and b) illustrates the computation of an integral image. Once we have the integral image, the sum of the function for any rectangle with upper left corner (x1,y1), and lower right corner (x2,y2) can be computed in constant time using the following equation,

$$\sum_{x=x_1}^{x_2}\sum_{y=y_1}^{y_2} f(x,y) = I(x_2,y_2) - I(x_2,y_1-1) - I(x_1-1,y_2) + I(x_1-1,y_1-1) \quad (4.3)$$

Figure 4.10 (c) illustrates that computing the sum of f(x,y) over the rectangle D using Equation 2 is equivalent to computing the sums over the rectangles,

$$(A+B+C+D)-(A+B)-(A+C)+A \quad (4.4)$$

4.5 BINARIZATION OF THE IMAGE

Image binarization is the process of separation of pixel values into two groups, black as background and white as foreground. To further enhance the estimated edges and generate a binary edge image E_b, we perform adaptive thresholding (AT) on the previously generated grayscale edge image E_g. Given E_g, the AT method generates an integral image E_i by summing all pixel values from the upper left corner for each pixel in E_g. Then, a binary edge image E_b is generated by thresholding each pixel *p(x, y)*, in the integral image E_i using a threshold that is adaptively computed from a local window in E_i. Specifically, E_b(x, y) is computed as follows:

$$E_b(x,y) = \begin{cases} 255 \; if \; E_i(x,y) \geq \beta\mu(x,y) \\ 0 \; if \; E_i(x,y) < \beta\mu(x,y) \end{cases} \tag{4.5}$$

Where β is the coefficient used to control the threshold and $\mu(x,y)$ is the average of all pixel values within the $h_w * h_w$ window surrounding pixel *p(x, y)*, i.e.,

$$\mu(x,y) = \frac{1}{h_w * h_w} * \sum_{\substack{-h_w/2 \leq k \leq h_w/2 \\ -h_w/2 < j < h_w/2}} E_i(x+k, y+j) \tag{4.6}$$

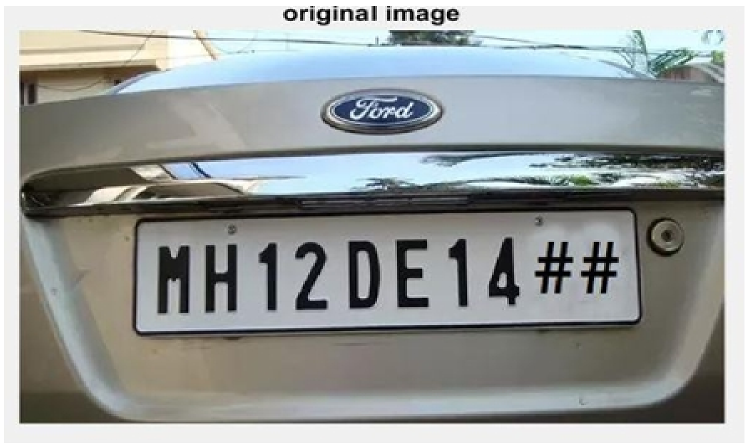


Figure.4.11: sample input image for the processing

The above method is applied for the input images we have taken. The input image after undergoing pre-processing method which involves down sampling and gray scaling of the image.Then the image is further processed by using and extended sobel operator. Now the obtained output of the sobel operator is taken as the input for the binarization of image. The final output we get after applying the above method is shown in the below figure,

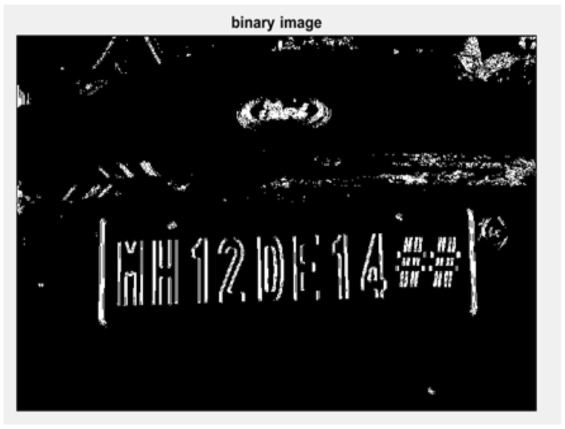


Figure.4.12 :binarized image by using adaptive thresholding

CHAPTER 5

LINE DENSITY FILTER

5.1 INTRODUCTION

It is known that a license plate is a pattern composed of several characters, which have high contrast changes to their background. Several morphological operations are generally used to find the high contrast area as important features to detect license plates. But as these require more time and also high computational costs. So we have proposed an efficient line density filter approach which solves the above mentioned problems and also provide better results. Before introducing the proposed method, some morphological operations should he introduced first.

5.2 MORPHOLOGICAL METHOD

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.Morphological techniques probe an image with a small shape or template called a structuring element. The structuring element is positioned at all possible locations in the image and it is compared with the corresponding neighbourhood of pixels. Some operations test whether the element "fits" within the neighbourhood, while others test whether it "hits" or intersects the neighbourhood:

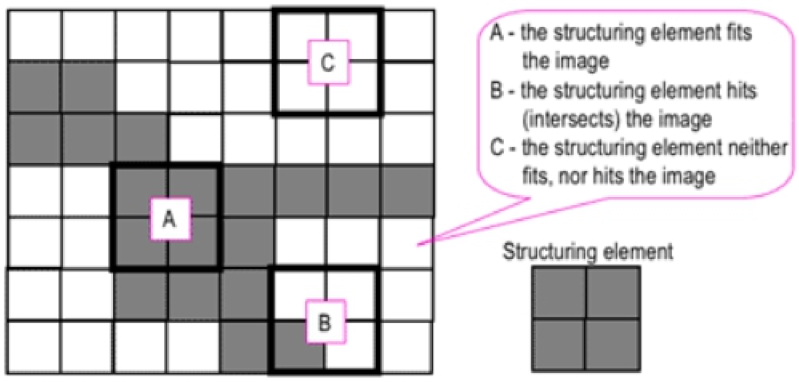
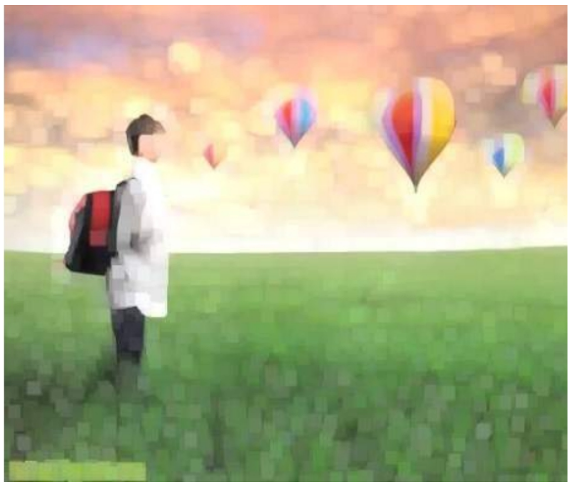

C
A
B
A - the structuring element fits the image
B - the structuring element hits (intersects) the image
C - the structuring element neither fits, nor hits the image
Structuring element

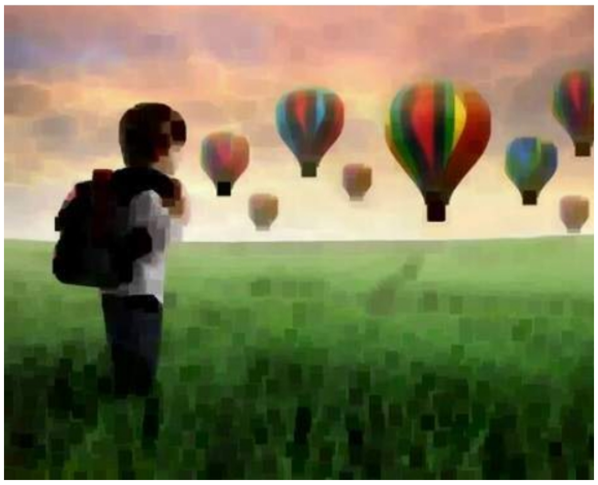
Figure 5.1:Probing of an image with a structuring element

The basic morphological method include methods like dilation, erosion and some smoothening methods. Let us briefly discuss about the above mentioned methods. 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.

(a)



(b)



(c)

Figure 5.2: images after performing basic morphological operations
(a)Input image (b)Dilated image (c) Eroded image

For dilation , the value of the output pixel is the maximum value of all pixels in the neighborhood. In a binary image, a pixel is set to 1 if any of the neighboring pixels have the value 1.Morphological dilation makes objects more visible and fills in small holes in objects.

For erosion, the value of the output pixel is the minimum value of all pixels in the neighborhood. In a binary image, a pixel is set to 0 if any of the neighboring pixels have the value 0.Morphological erosion removes islands and small objects so that only substantive objects remain.

All possible vertical edges can he extracted with a thresholding operation. It is known the vertical edges in a license plate are close and adjacent to each other. These adjacent edges can he connected together through a closing operation and then form a connected segment. Therefore, before thresholding. a closing operation is applied first to let all adjacent vertical edges form a connected region. Then a labeling process is executed to extract the license-plate-analogue segments. Then. a set of potential license plates can be obtained from a cluttered environment.

5.3 PROPOSED LDF APPROACH

Given the edge image generated via AT [12] , we attempt to highlight the license plate area and remove noise in the binary image. The morphology technique is typically used as a mathematical tool to address such problems in image processing. However, the conventional morphology filter is very time-consuming because its template usually contains several pixels, which increases the computational cost. Thus, this filter may not be appropriate for applications that require real-time processing.Therefore, we propose an efficient line density filter (LDF) to highlight the character region.Our proposed LDF approach is motivated by the following observations:

1) License plates generally exhibit a relatively high edge density.
2) The characters on a license plate are printed in a horizontal orientation, and the height of each character is nearly identical.
3) If an image contains multiple license plates, some spatial distance will exist between each of them.

The underlying concept of the LDF [5] is to, by observing the above mentioned cues, connect regions of high edge density and remove sparse regions in each row and column in the binary edge image E_b. An illustration of the LDF processing of two rows in an image is shown in Figure.5.3, where non-edge pixels are shown in black and edge pixels are shown in white. Suppose that l_1 and l_2 represent two continuous black line segments, with lengths of l_{b1} and l_{b2} pixels, respectively. The target line segment l_3, which contains both edge pixels and non-edge pixels, is the one for which we wish to either connect or remove the white edge pixels. We make this decision based on the line density d_l , which is defined as the proportion of edge pixels in the line segment l_3 i.e.,

$$d_l = \frac{w_l}{(w_l+b_l)} \qquad (5.1)$$

where w_l and b_l denote the numbers of white and black pixels, respectively, in the line segment l_3.

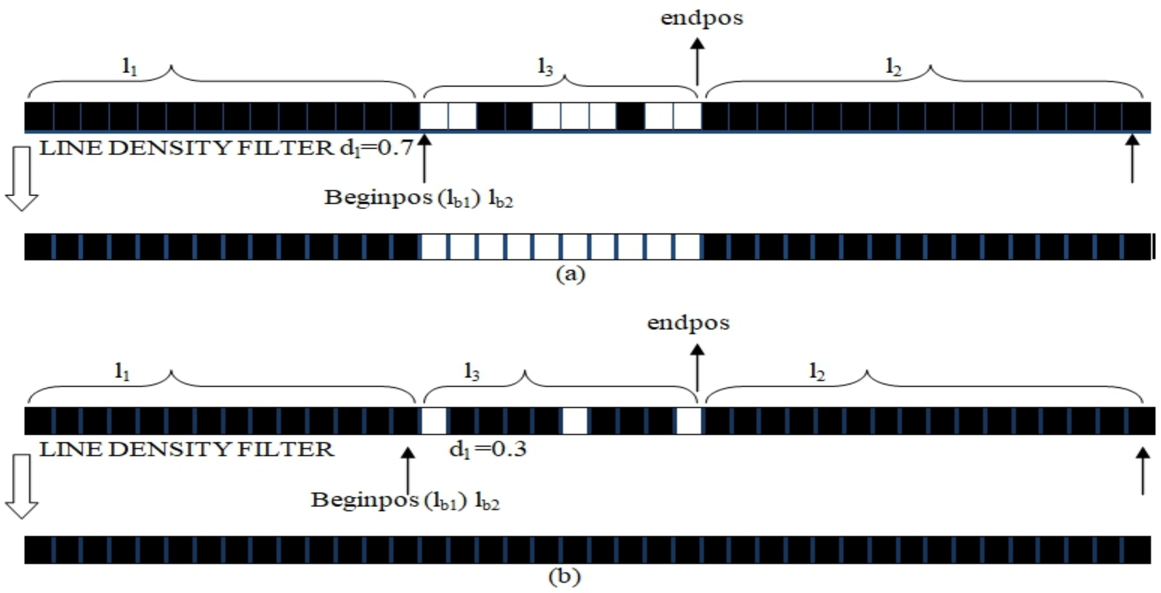

endpos
l_1
l_3
l_2
LINE DENSITY FILTER d_l=0.7
Beginpos (l_{b1}) l_{b2}
(a)
endpos
l_1
l_3
l_2
LINE DENSITY FILTER
d_l=0.3
Beginpos (l_{b1}) l_{b2}
(b)

Figure.5.3 :.An illustration of the LDF processing of two rows in an image, where and represent the lengths of line segments l_1 and l_2, respectively.The high-density line segment l_3 in (a) is connected (in white) by the LDF, whereas the lower density one in (b) is removed (in black).

Ideally, we suppose that non-plate areas exist on both sides of the license plate. In other words, there should be relatively long black line segments (e.g., l_1 and l_2) on both sides of a discontinuous line segment (e.g., l_3) to be processed. To find such a relationship between the line segments l_1, l_2 and l_3, we propose the following scheme for calculating the lengths of the line segments. A black pixel is counted as 1, whereas a white pixel is counted as -1. If continuous black pixels exist, we group them together. Similarly, if continuous white pixels exist, we calculate the cumulative length of each white line segment, which is represented by a negative number. Several examples of the length calculation for horizontal lines are shown in Figure. 5.3. The length calculation for vertical lines is the same but in the vertical direction

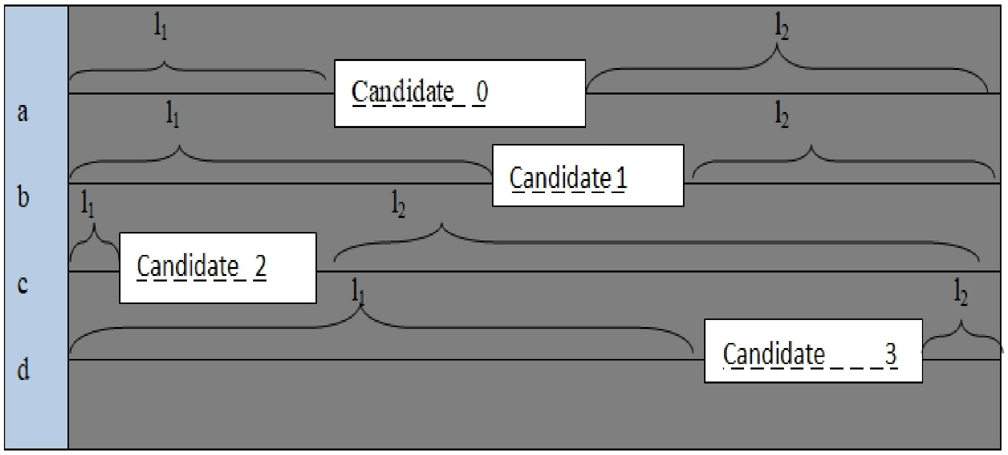


Figure 5.4:Candidate license plate locations in four lines

The lengths of line segments l_1 and l_2 are l_{b1} and l_{b2}, respectively. Candidate 0 and candidate 1 are located in the middle of the image, such that $l_{b1} > T_{min}$ and $l_{b2} > T_{min}$. Candidate 2 is close to the leftmost side of the image, such that $l_{b1} < T_{min}$ and $l_{b2} > T_{min}$. Candidate 3 located near the rightmost boundary of the image, such that $l_{b1} > T_{min}$ and $l_{b2} < T_{min}$.

Using the generated horizontal line-length map for each pixel of the binary image, we apply the proposed LDF to extract the license plate candidates. The pseudocode for our horizontal line density filter (HLDF) is presented in flowchart . This pseudocode contains four input parameters: the binary image E_b; the minimum length threshold for a black line, T_{min}, for l_1 or l_2; the gap-length threshold, T_{gap}, for l3 (between l_1 and l_2); and the line density threshold T_{dl} for l_3. Note that as a special case, the candidate may be located too close to the left or right image boundary, that is, the leftmost line segment l_1 or the rightmost line segment l_2 may have a length of less than T_{min}; in this case, the algorithm should reapply the same processing to l_3. In Fig. 5.4, the top two lines illustrate condition C_1 as denoted in flowchart in figure 5.5, whereas the bottom two lines represent conditions C_2 and C_3, which require reprocessing.

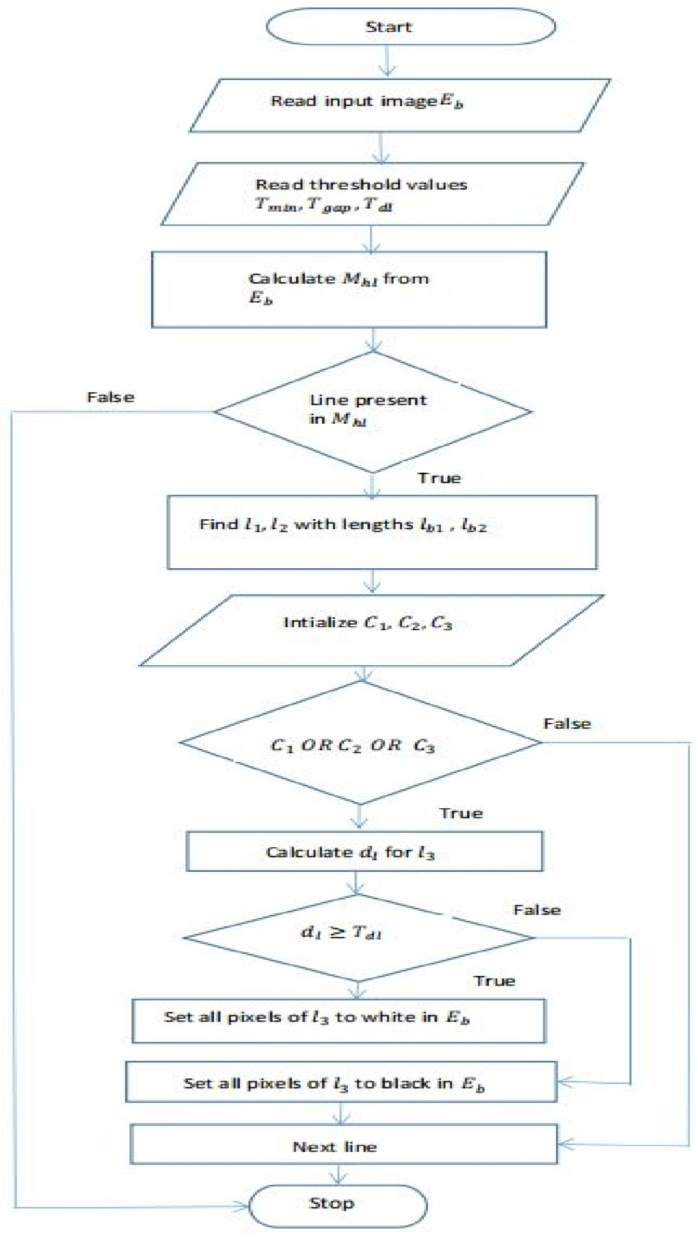

Start
Read input imageE_b
Read threshold values T_{min}, T_{gap}, T_{dl}
Calculate M_{hl} from E_b
Line present in M_{hl}
False
True
Find l_1, l_2 with lengths l_{b1} , l_{b2}
Intialize C_1, C_2, C_3
C_1 OR C_2 OR C_3
False
True
Calculate d_l for l_3
$d_l \geq T_{dl}$
False
True
Set all pixels of l_3 to white in E_b
Set all pixels of l_3 to black in E_b
Next line
Stop

Figure.5.5:Horizontal line density filter flowchart

For the vertical direction, we can apply the same code to the transposed binary image E_b. To remove the minor noise from the image we used a small code based on the flowchart in figure 5.6.

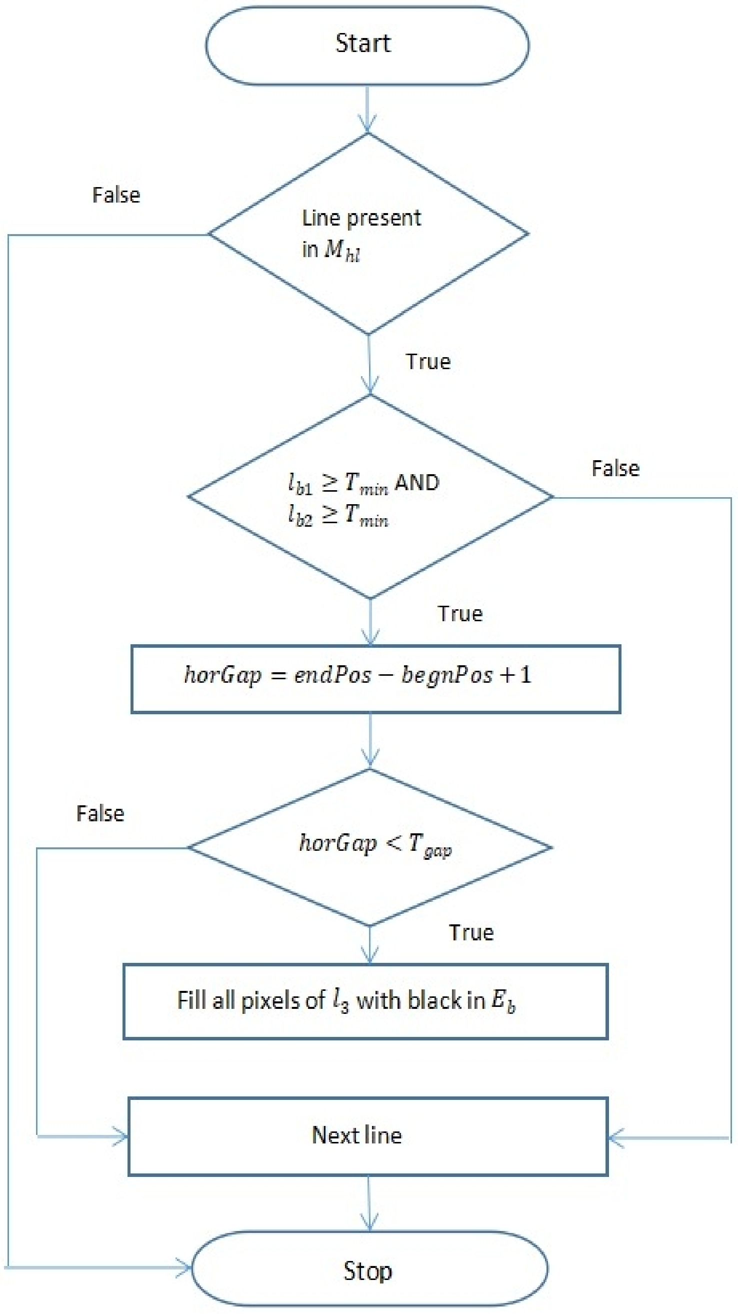
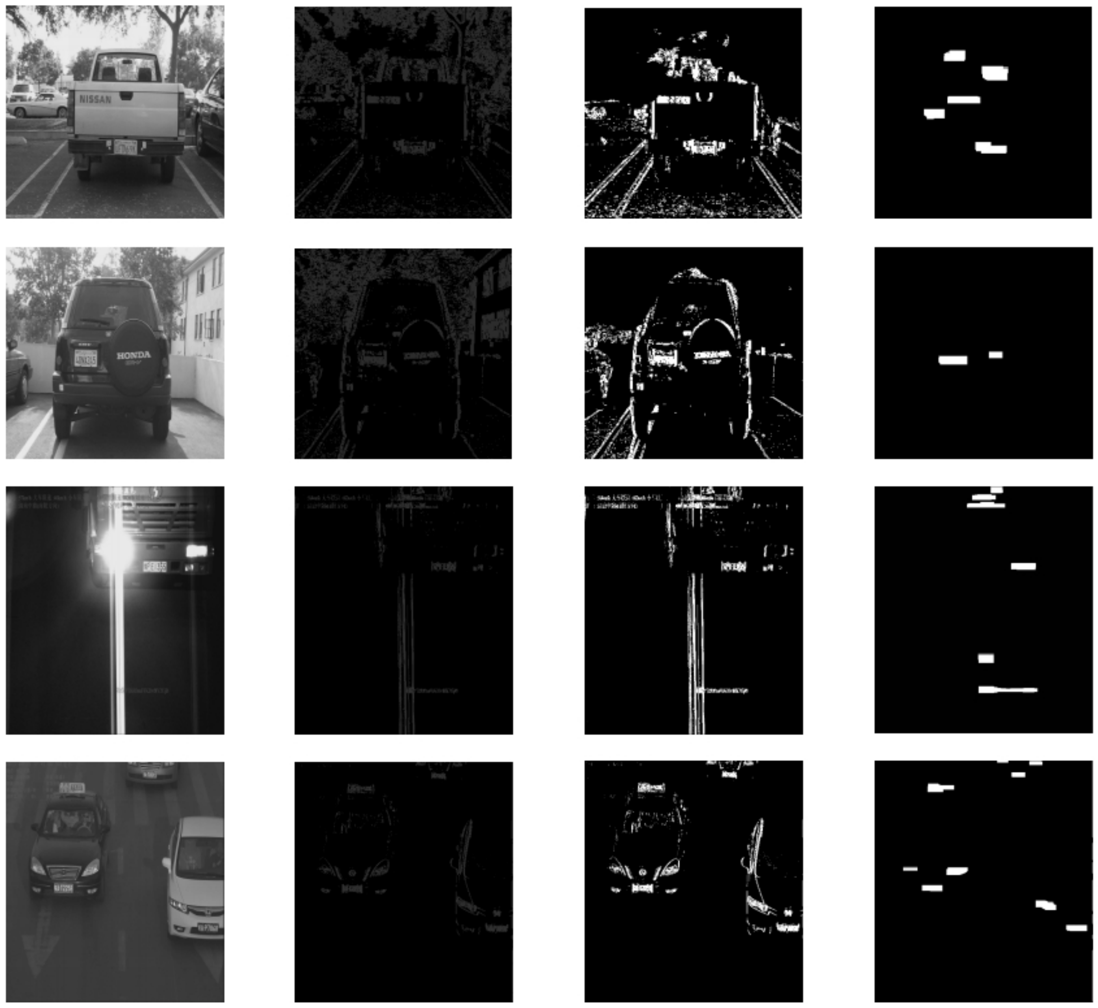

Start
Line present in M_{hl}
False
True
$l_{b1} \geq T_{min}$ AND $l_{b2} \geq T_{min}$
False
True
$horGap = endPos - begnPos + 1$
$horGap < T_{gap}$
False
True
Fill all pixels of l_3 with black in E_b
Next line
Stop

Figure 5.6: flowchart of the code to remove minor noise in image

Figure 5.7 shows several candidate license plate regions processed using the LDF method to demonstrate the robustness of our LDF algorithm under different illumination conditions.



(a) (b) (c) (d)

Figure 5.7:Vehicle images and their license plate candidate results generated by the proposed LDF approach

(a) original grayscale source images, (b) results of edge density detection and enhancement, (c) binary edge images obtained via adaptive thresholding (AT) (d) candidate results generated by our line density filter (LDF).

CHAPTER 6

CANDIDATE EXTRACTION USING SUPPORT VECTOR MACHINE

6.1 INTRODUCTION OF SVM :

SVM [21] means support vector machine. It is a machine learning model. Machine Learning is considered as a sub field of Artificial Intelligence and it is concerned with the development of techniques and methods which enable the computer to learn. In simple terms development of algorithms which enable the machine to learn and perform tasks and activities. Machine learning overlaps with statistics in many ways. Over the period of time many techniques and methodologies were developed for machine learning tasks .Support Vector Machine (SVM) [23] was first heard in 1992, introduced by Boser, Guyon, and Vapnik in COLT-92. Support vector machines (SVMs) are a set of related supervised learning methods used for classification and regression . They belong to a family of generalized linear classifiers. In another terms, Support Vector Machine (SVM) is a classification and regression prediction tool that uses machine learning theory to maximize predictive accuracy while automatically avoiding over-fit to the data. Support Vector machines can be defined as systems which use hypothesis space of a linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory. Support vector machine was initially popular with the NIPS community and now is an active part of the machine learning research around the world. SVM becomes famous when, using pixel maps as input; it gives accuracy comparable to sophisticated neural networks with elaborated features in a handwriting recognition task . It is also being used for many applications, such as hand writing analysis, face analysis and so forth, especially for pattern classification and regression based applications. The foundations of Support Vector Machines (SVM) have been developed by Vapnik and gained popularity due to many promising features such as better empirical performance. The formulation uses the Structural Risk Minimization (SRM) principle, which has been shown to be superior to traditional Empirical Risk Minimization (ERM) principle, used by conventional neural networks. SRM minimizes an upper bound on the expected risk, where as ERM minimizes the error on the training data. It is this difference which equips SVM with a greater ability to generalize, which is the goal in statistical learning. SVMs were developed to solve the classification problem, but recently they have been extended to solve regression problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well (look at the below snapshot).

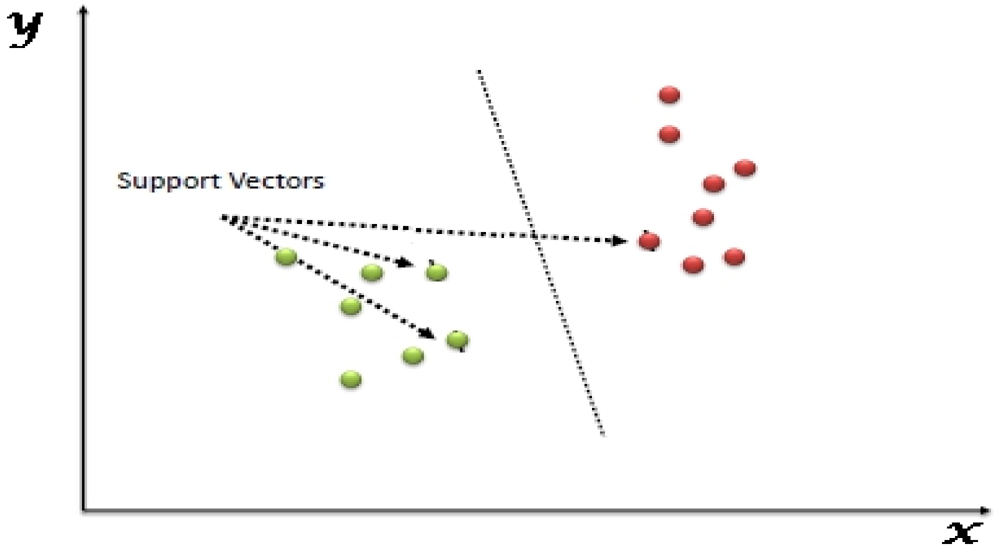


Figure 6.1 hyper planes of SVM

Support Vectors are simply the co-ordinates of individual observation. The SVM classifier is a frontier which best segregates the two classes (hyper-plane/ line).

6.2 WORKING OF SVM:

6.2.1 Identification of right hyper-plane (Scenario-1):

Here, we have three hyper-planes (A, B and C). Now, identify the right hyper-plane to classify star and circle.You need to remember a thumb rule to identify the right hyper-plane: “Select the hyper-plane which segregates the two classes better”. In this scenario, hyper-plane “B” has excellently performed this job.

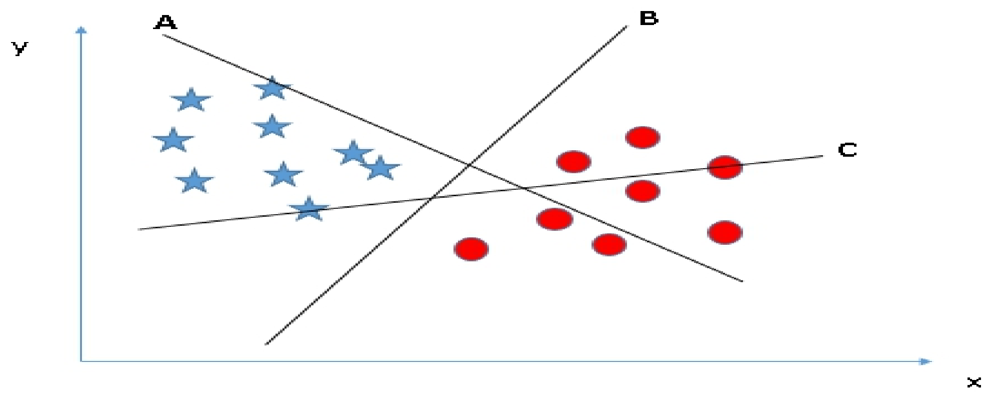


Figure 6.2 hyper line identification in scenario-1

6.2.2 Identification of right hyper-plane (Scenario-2):

Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane ?.

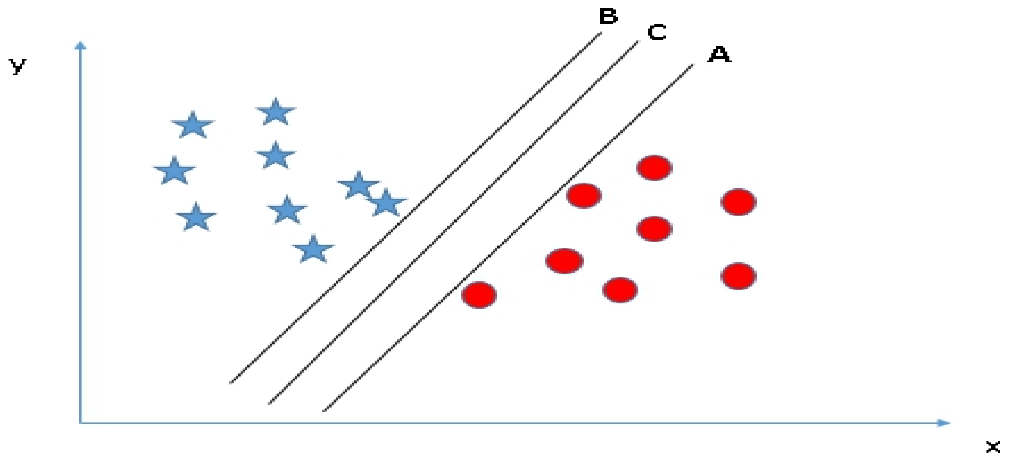


Figure 6.3 hyper line identification in scenario-2

Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**. Let's look at the below snapshot.

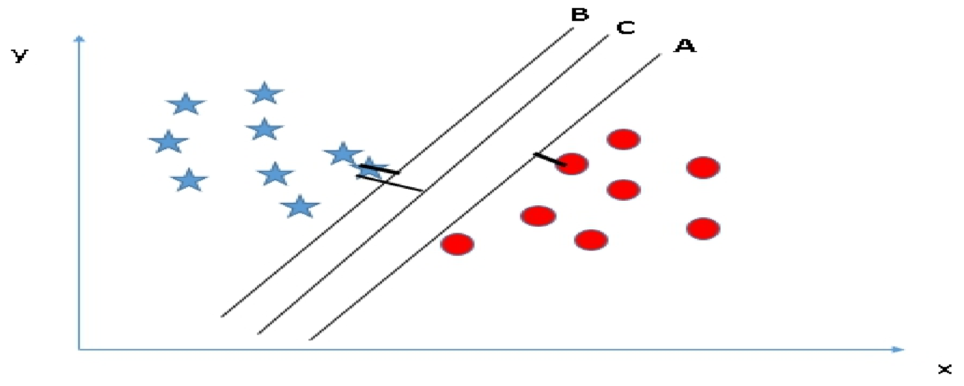


Figure 6.4 margin calculation in hyper plane

<u>6.2.3 Identification of right hyper-plane (Scenario-3):</u>

Some of you may have selected the hyper-plane **B** as it has higher margin compared to **A.** But, here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is **A.**

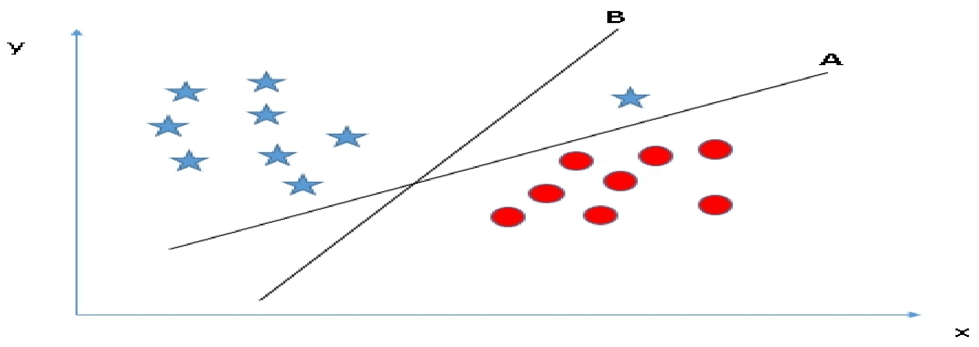


Figure 6.5 hyper plane identification in scenario -3

6.2.4 Identification of right hyper-plane (Scenario-4):

As I have already mentioned, one star at other end is like an outlier for star class. The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers.

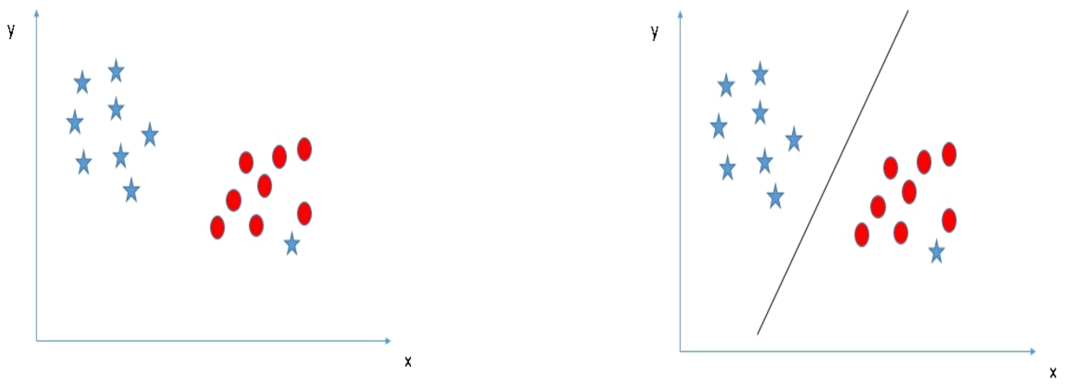


Figure 6.6 hyper plane identification in scenario -4

6.3 SVM FOR CLASSIFICATION:

SVM is a useful technique for data classification. Even though it's considered that Neural Networks are easier to use than this, however, sometimes unsatisfactory results are obtained. A classification task usually involves with training and testing data which consist of some data instances . Each instance in the training set contains one target values and several attributes. The goal of SVM is to produce a model which predicts target value of data instances in the testing set which are given only the attributes . Classification in SVM is an example of Supervised Learning. Known labels help indicate whether the system is performing in a right way or not. This information points to a desired response, validating the accuracy of the system, or be used to help the system learn to act correctly. A step in SVM classification involves identification as which are intimately connected to the known classes. This is called feature selection or feature extraction. Feature selection and SVM classification together have a use even when prediction of unknown samples is not necessary. They can be used to identify key sets which are involved in whatever processes distinguish the classes .

6.4 SVM FOR REGRESSION:

SVMs can also be applied to regression problems by the introduction of an alternative loss function . The loss function must be modified to include a distance measure. The regression can be linear and non linear. Linear models mainly consist of the following loss functions, e-intensive loss functions, quadratic and Huber loss function. Similarly to classification problems, a non-linear model is usually required to adequately model data. In the same manner as the non-linear SVC approach, a non-linear mapping can be used to map the data into a high dimensional feature space where linear regression is performed. The kernel approach is again employed to address the curse of dimensionality. In the regression method there are considerations based on prior knowledge of the problem and the distribution of the noise. In the absence of such information Huber's robust loss function, has been shown to be a good alternative.

6.5 APPLICATIONS OF SVM:

SVM has been found to be successful when used for pattern classification problems. Applying the Support Vector approach to a particular practical problem involves resolving a number of questions based on the problem definition and the design involved with it. One of the major challenges is that of choosing an appropriate kernel for the given application . There are standard choices such as a Gaussian or polynomial kernel that are the default options, but if these prove ineffective or if the inputs are discrete structures more elaborate kernels will be needed. By implicitly defining a feature space, the kernel provides the description language used by the machine for viewing the data. Once the choice of kernel and optimization criterion has been made the key components of the system are in place . Let's look at some examples. The task of text categorization is the classification of natural text documents into a fixed number of predefined categories based on their content. Since a document can be assigned to more than one category this is not a multi-class classification problem, but can be viewed as a series of binary classification problems, one for each category. One of the standard representations of text for the purposes of information retrieval provides an ideal feature mapping for constructing a Mercer kernel. Indeed, the kernels somehow incorporate a similarity measure between instances, and it is reasonable to assume that experts working in the specific application domain have already identified valid similarity measures, particularly in areas such as information retrieval and generative models . Traditional classification approaches perform poorly when working directly because of the high dimensionality of the data, but Support Vector Machines can avoid the pitfalls of very high dimensional representations.A very similar approach to the techniques described for text categorization can also be used for the task of image classification, and as in that case linear hard margin machines are frequently able to generalize well. The first real-world task on which Support Vector Machines were tested was the problem of hand-written character recognition. Furthermore, multi-class SVMs have been tested on these data. It is interesting not only to compare SVMs with other classifiers, but also to compare different SVMs among themselves. They turn out to have approximately the same performance, and furthermore to share most of their support vectors, independently of the chosen kernel. The fact that SVM can perform as well as these systems without including any detailed prior knowledge is certainly remarkable.

<u>6.6 STRENGTH AND WEAKNESS OF SVM</u>:

The major strengths of SVM are the training is relatively easy. No local optimal, unlike in neural networks. It scales relatively well to high dimensional data and the trade-off between classifier complexity and error can be controlled explicitly. The weakness includes the need for a good kernel function.

<u>6.7 TRAINING OF SVM MODEL:</u>

Let us take an example to understand SVM properly. Suppose we see a strange cat that has some features of dog ,so If you want to model that can accurately identify whether it is dog or cat ,such model can be created by SVM. first we train our model with lot of images of cats and dogs so that we can extract different features of cat and dog and then we test it with that strange creature ,it will classify it as a cat. This can be done by training our strange creature with features of cat and dog that are extracted from different images.

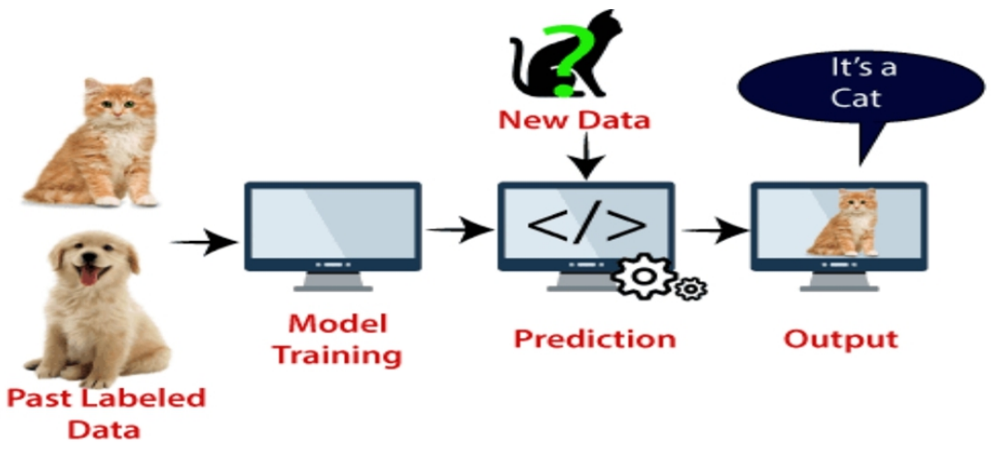


Figure 6.7 training of SVM model

<u>6.7.1 TRAINING OF SVM MODEL FOR DETECTION OF LP AND NLP REGIONS:</u>

similar to above case here we have to train a SVM model for detection of licensed and non licensed plate regions. For training of svm we have taken 50 images . those 50 images undergoes all preprocessing techniques, binarization and line density filtering.

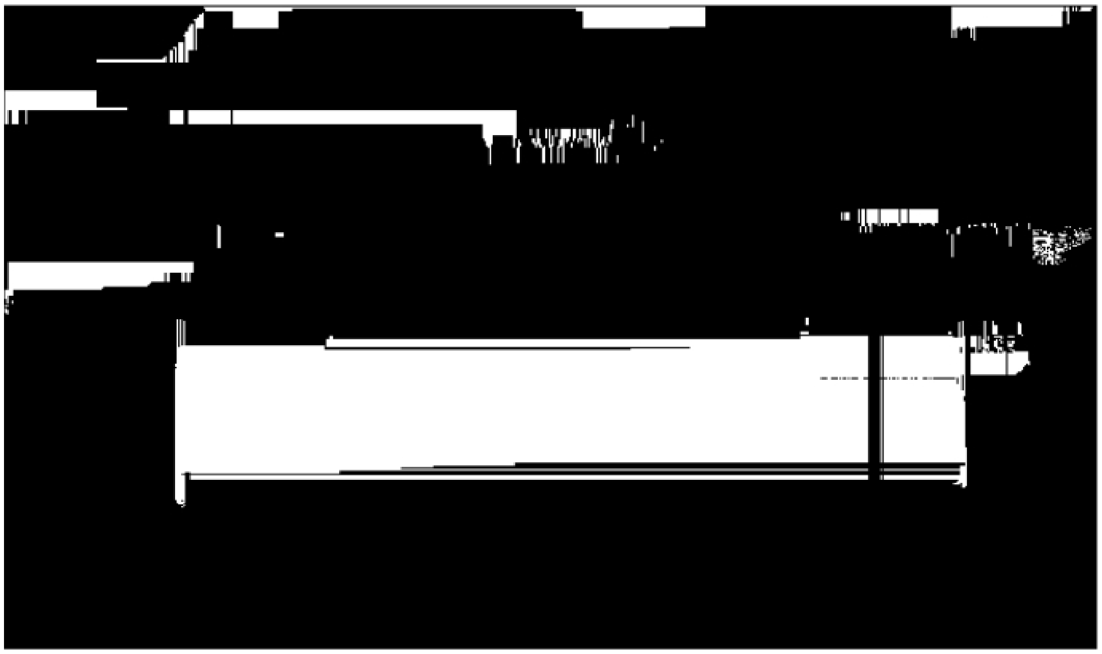


Figure 6.8 output of line density filter

If we observe the output that is obtained from line density filter there are so many white regions are visible. Among all those white regions only one region is true license plate region and that is the only part that we have to extract. Remaining all other white regions are non licensed plate regions. Suppose we are taking 50 images (figure 4.9) for training of svm so there is only 50 true licensed plate regions. For each output obtained from the line density filer there are some random number of non licensed plate regions. So for 50 images we got 50 licensed plate regions and 200 non licensed plate regions in our case. There are some standard characteristics for true license plate . those characteristics are area, perimeter, mean value of RGB (red, green, blue) intensity. For each image we have to calculate these five characteristics . there are 50 licensed and 200 non licensed regions .

we have to calculate these five characteristics for all 250 images. So our SVM training matrix become 5x250. so here we are training our svm model with all 250 images and compare our unknown plate with these characteristics to identify it is licensed or not. Among this 5x240 matrix we have to compare each plate characteristics with standard license plate characteristics and we should identify where the standard license plate characteristics are present. We have noticed characteristics of some licensed and non-licensed regions which are listed in Table-4.1. if we observe the figure 4.10 those are the different vehicle plates that are used for the training of svm model. Here we extracted characteristics of licensed and non licensed plates . Each extracted region is correlated with available template and the template provide high correlated value is recognize as the character present in the license plate.

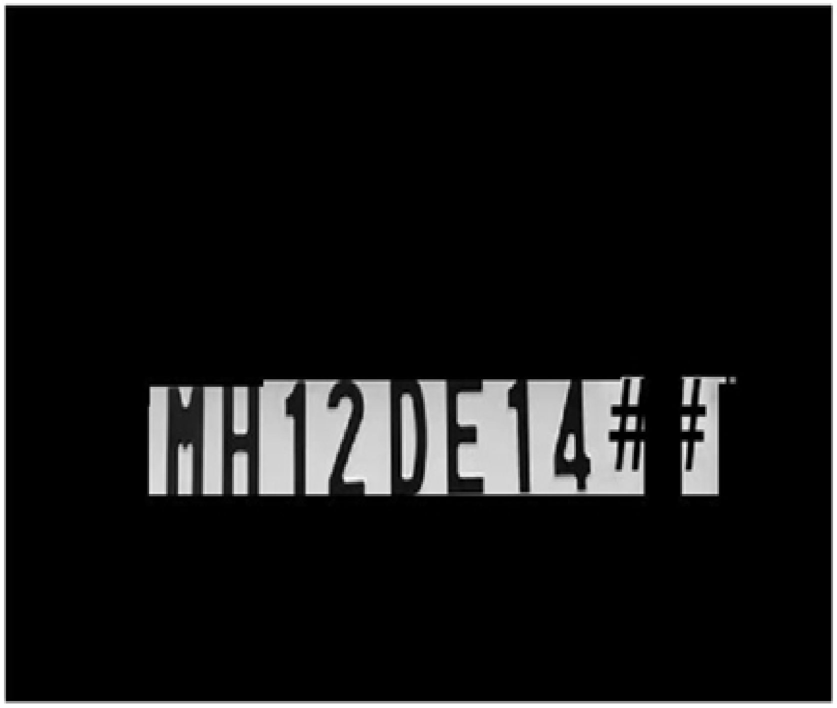


Figure 6.9 detection of license plate region

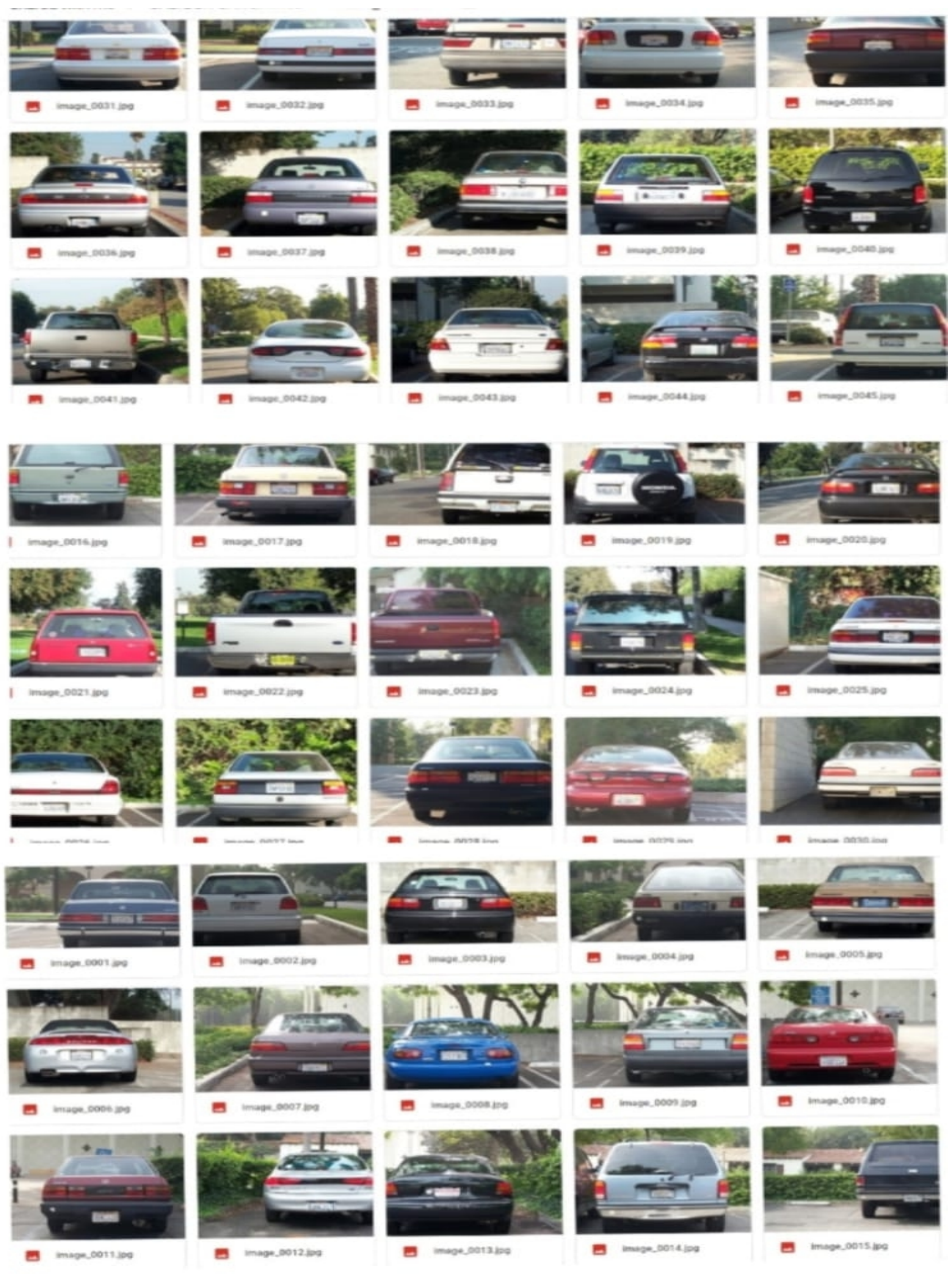


Figure 6.10 different images taken from data set

Table 6.1 characteristics of licensed and non licensed plate regions

region	Area	Perimeter	Red component	Green Component	Blue Component
LP	62996	4.870e3	127.233	129.160	130.81
NLP	2	0	25.5	30.5	31
NLP	45	64.92	124.529	139.980	162.20
NLP	24	5.88	168.5	174.2	196.4
NLP	120	105.05	109.920	124.197	146.02
NLP	38	23.52	73.1786	86.3571	108.2
NLP	2	4.962	167.5	171.5	170.4
NLP	2	3.92	41.625	44.75	39.25
NLP	1	0	56.75	65.75	62.25

LP - (LICENSE PLATE) NLP - (NON-LICENSED PLATE)

6.7.2 RECOGNITION OF CHARACTER IN THE PLATE:

Using the region function property of matlab we have extracted different connected regions in the number plate. In the above figure we have noticed true licensed plate region. Now our second task is to recognize the characters present in that license plate region. For that purpose we have created a data base (figure 6.11) containing 'a' to 'z' alphabets and 0 to 9 numbers.So the height and width of each extracted region is compared with height and width of the licensed plate region. If the character is present in the license plate then it will be displayed. characters present in that license plate region. For that purpose we have created a data base containing 'a' to 'z' alphabets and 0 to 9 numbers.So the height and width of each extracted region is compared with height and width of the licensed plate region. If the character is present in the license plate then it will be displayed. Using the region property function of matlab we have extracted different connected regions in the number plate . the height and width of the each extracted region is compared with height and width of licensed plate region. Each extracted region is correlated with the available template and template provide high correlated value is recognize as the character present in the license plate.

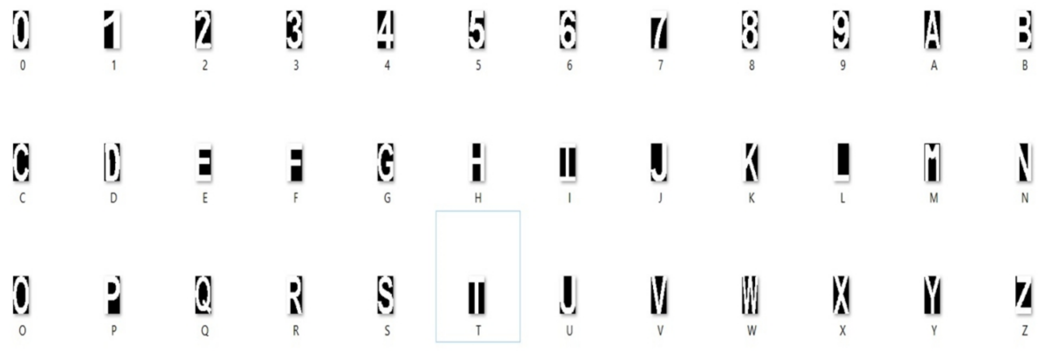


Figure 6.11 template formation

the characters that extracted from the license plate is compared with the characters and numbers that are present in the template . if the number or character is present in the template correlated output become high and it is displayed . similarly all the characters are compared with the characters in template and all the characters will displayed.

MH12DE14

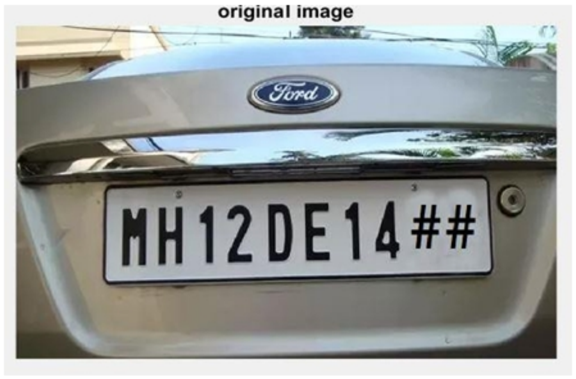
Figure 6.12 character extraction from number plate

The original input image is is converted into gray scale image. Then the edges are detected using extended sobel operator which are further defined by binarization using adaptive thresholding . the correct license plate region defined using Line Density Filter(LDF) and out of all the detected regions the correct region is detected using SVM's algorithm.then, by using character templates we recognized the characters in the license and print them.

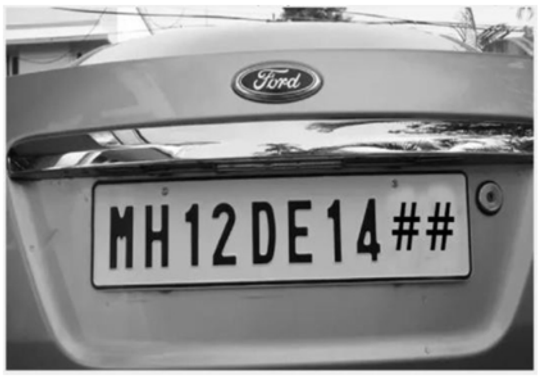
CHAPTER 7
RESULTS

7.1 RESULTS

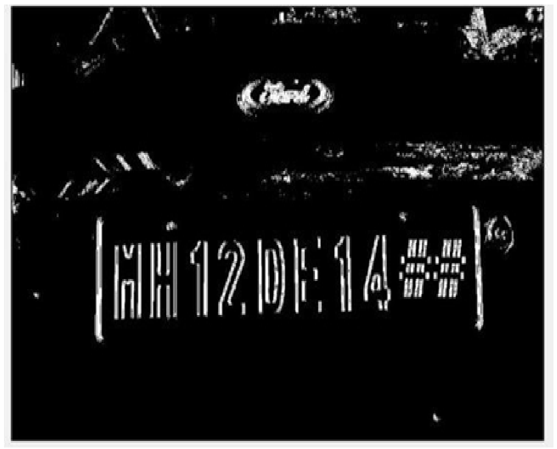
The original input image is first downscaled and then it is converted into grayscale image as shown in fig.7.1(a) and 7.1(b). Then the edges are detected using extended sobel operator which are further defined by binarization using adaptive thresholding as in fig.7.1.(d). the correct license plate region defined using Line Density Filter(LDF) and out of all the detected regions the correct region is detected using SVM's algorithm as in fig.7.1(f).then, by using character templates we recognized the characters in the license and print them. The detected output is shown in fig.7.1(g)



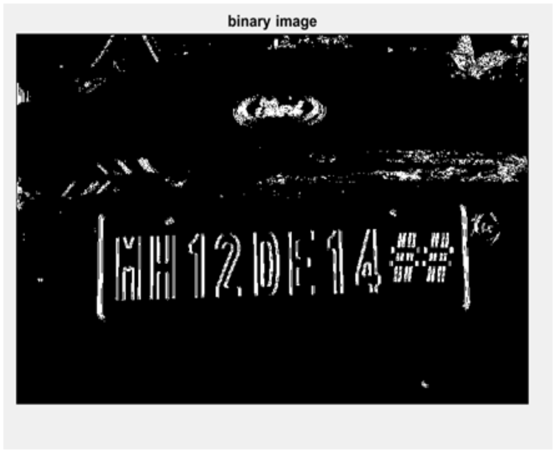
(a)



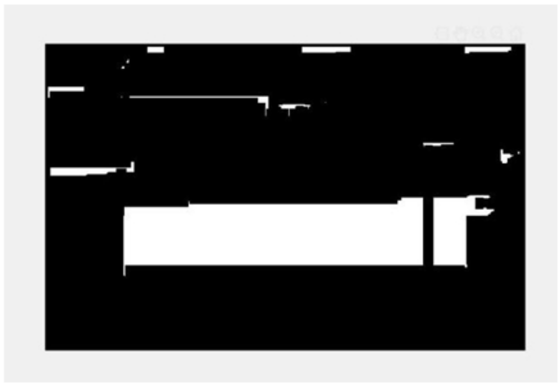
(b)



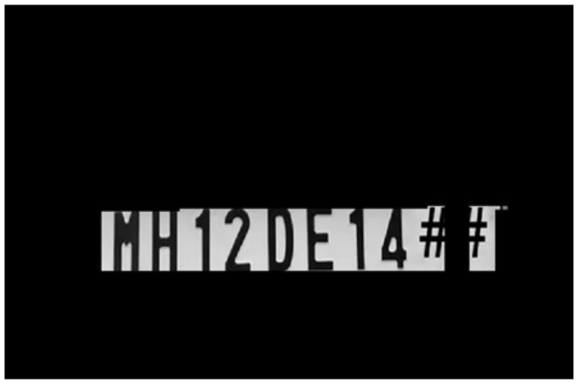
(c)



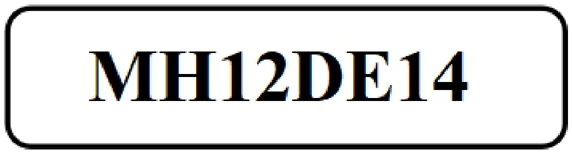
(d)



(e)



(f)



MH12DE14

(g)

Figure 7.1:final output results

CONCLUSION

SUMMARY

Thus, in this paper we propose a novel and efficient method to detect the license plate. In the license plate detection approach we have proposed it mainly consists of three parts: pre-processing, candidate extraction and candidate verification. In this we have proposed methods which are less time consuming as well as efficient so they can be used in real time applications. An extension of sobel operator is used to provide better edge density enhancement and a binary image is generated from the edge detected image using adaptive thresholding method instead of regular simple thresholding as it has a global threshold value for all the pixels unlike adaptive thresholding we have mentioned in our paper. To detect the license plate region from the binary image we use line density filter. In order to identify the correct license plate region from the region we have extracted from LDF method we have used SVM algorithm and trained a set of images to the system to detect the exact region. Finally, after detecting the region of our interest we used character templates to recognize the characters on license plate.

FUTURE SCOPE

As demonstrated in the experiments, the proposed approach and the other state-of-the-art methods are all still subject to certain limitations when addressing difficult scenes, e.g., reflective glare on license plates. In the future, it would be interesting to exploit the MSER or Hough transform approach to address such difficult cases.

REFERENCES

[1] Yule Yuan, Member, IEEE, Wenbin Zou, Yong Zhao, Xinan Wang, Xuefeng Hu, and Nikos Komodakis "A Robust and Efficient Approach to License Plate Detection" IEEE Transactions On Image Processing

[2] Abhinav Deo Guided By Prof. Umesh Chandra Pati "Recognision Of License Plate And Detection Of Optical Nerve Pattern Using Hough Transform" dept. of ECE, National Institute of Technology, Rourkela.

[3] "Adaptive Thresholding Using the Integral Image" guided by Derek Bradley Carleton University, Canada derek@derekbradley.ca; Gerhard Roth National Research Council of Canada Gerhard.Roth@nrc-cnrc.gc.ca

[4] Faizal Patel, Jaimini Solanki, VivekRajguru, Ankit Saxena "Recognition of Vehicle Number Plate Using Image Processing Technique" Medicaps Institute of Technology and Management, Indore, MP, India.

[5] D.Tran,L.H.D.Tran, P.V.Tran, V.H.Nguyen, "Building an Automatic Vehicle LicensePlate Recognition System", In Proceedings of the Pattern recognition conference, vol. 1, pp. 747-750, Hongkong, 2005.

[6] S.Kranthi, K.Pranathi, A.Srisaila, "Automatic Number Plate Recognition", International journal on advancement of technology, vol. 2, pp. 408-422, 2011.

[7] D. Shan, M. Ibrahim, M. Shehata, W. Badawy, "Automatic license plate Recognition (ALPR): A State-of-the-Art Review," IEEE Trans. Circuits Syst. Video Technol., vol.23, no. 2, pp.311-324, (2013)

[8] M. A. Lalimi, S. Ghofrani, D. McLernon, "A vehicle license plate detection method using region and edge based methods," Computers and Electrical Engineering, vol.39, pp.834-845, (2013).

[9] R. Wang, N. Sang, R. Huang, Y. Wang, "License plate detection using gradient information and cascade detecors", Optik, vol.125, pp.186-190, (2014).

[10] J. W. Hsieh, S. H. Yu, Y. S. Chen, "Morphology-based license plate detection from complexscenes," in Proc. Int. Conf. Pattern Recognit.(ICPR),Quebec City, Canada, pp.176-179, (2002).

[11] A.M. Al.Ghaili, S. Mashohor, A. R. Ramli, A. Ismail, "Vertical-EdgeBased Car-License-Plate Detection Method," IEEE trans. veh. technol., vol.62, no.1, (2013).

[12] D. Bradley, G. Roth, "Adaptive thresholding using the integral image," J. Graph. Tools, vol.12, no.2, pp.13-21, (2007).

[13] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology," IEEE Trans. Pattern Anal. Mach. Intell., vol.4, pp.532-550, (1987).

[14] W. Zou, Z. Liu, K. Kpalma, J. Ronsin, Y. Zhao and N. Komodakis, "Unsupervised Joint Salient Region Detection and Object Segmentation," IEEE Trans. Image Process., vol. 24, no. 11, pp. 3858-3873, (2015).

[15] J. W. Hsieh, S. H. Yu, Y. S. Chen, "Morphology-based license plate detection from complex scenes," in Proc. Int. Conf. Pattern Recognit. (ICPR),Quebec City, Canada, pp.176-179, (2002).

[16] E. R. Lee, P. K. Kim, H. J. Kim, "Automatic recognition of a car license plate using color image processing," in Proc. IEEE Int. Conf. Image Process. (ICIP), pp.301-305, (1994).

[17]]V. Abolghasemi, A. Ahmadyfard, "An edge-based color-aided method for license plate detection," Image and Vis. Comput., vol.27, pp.1134-1142.

[18] W.J. Jia, H.F. Zhang, X.J. He, "Region-based license plate detection," J. Network Comput. Appl., vol.30, pp.1324-1333, (2007).

[19] G.S. Hsu, J.-C. Chen, and Yu-Zu Chung, "Application-Oriented License Plate Recognition," IEEE Trans. Veh. Technol., vol.62,no.2,pp.552-561, (2013).

[20] A. H. Ashtari, M. J. Nordin, and M. Fathy, "An Iranian License Plate Recognition System Based on Color Features," IEEE Trans. Intell. Transp. Syst., vol.15, no.4, pp.1690-1704, (2014).

[21] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features", *Proc. 10th European Conf. Machine Learning (ECML)*, 1998.

[22] K. Sung, *Learning and Example Selection for Object and Pattern Detection*, 1995.

[23] E. Osuna, R. Freund and F. Girosi, "An Improved Training Algorithm for Support Vector Machines", *Proc. IEEE Workshop on Neural Networks and Signal Processing*, 1997.

[24] A. Blumer, A. Ehrenfeucht, D. Haussler, And M.Warmuth, Classifying learnable geometric concepts with the vapnik-chervonenkis dimension, in Proceedings of the eighteenth annual ACM symposium on Theory of computing, STOC '86, New York, NY, USA, 1986, ACM, pp. 273–282.

[25] V. N. Vapnik Anda.Y. Chervonenkis, On the uniform convergence of relative frequencies of events to their probabilities, Doklady Akademii Nauk USSR, 181 (1968)