

A Hybridized Approach to Efficient and Robust License Plate Detection and Recognition

A DISSERTATION

*Submitted in partial fulfillment of
the requirements for the award of the degree*

of

Master of Technology

in

ELECTRONICS AND COMMUNICATION ENGINEERING

with Specialization in Communication Systems

By

Lt Col Amandeep Singh Puri

(Enrollment No. 17531004)



DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY ROORKEE

ROORKEE - 247667 (INDIA)

JUNE, 2019

CANDIDATE'S DECLARATION

I declare that the work presented in this dissertation with title “**A Hybridized Approach to Efficient and Robust License Plate Detection and Recognition**” towards the fulfillment of the requirement for the award of the degree of **Master of Technology** submitted in the **Department of Electronics and Communication Engineering, Indian Institute of Technology Roorkee**, India. It is an authentic record of my own work carried out under the supervision of **Dr. D Ghosh**, Professor, Department of Electronics and Communication Engineering, IIT Roorkee. The content of this dissertation has not been submitted by me for the award of any other degree of this or any other institute.

DATE :

SIGNATURE:

PLACE:

(LT COL AMANDEEP SINGH PURI)

CERTIFICATE

This is to certify that the statement made by the candidate is correct to the best of my knowledge and belief.

DATE :

SIGNATURE:

(DR. D GHOSH)

PROFESSOR,
DEPT. OF ECE,
IIT ROORKEE-247667

ACKNOWLEDGEMENT

On completion of my dissertation I would like to extend my sincere gratitude to both my guides Dr. Debashis Ghosh, Professor, Department of Electronics and Communication Engineering, IIT Roorkee and Dr. M. J. Nigam, Professor and Head of Department, Department of Electronics and Communication Engineering, Jaypee University of Information Technology and former Professor at Department of Electronics and Communication Engineering, IIT Roorkee, for providing their unstinting support and profound knowledge, which in turn has empowered me to carry out my research on the subject in a comprehensive manner. The out of box thinking and rich experience possessed by both my guides in the field of image processing has been a source of inspiration for me which has in turn assisted me to approach my study and other issues linked to it in a logical, clear and flexible manner. I will always be indebted to them for the same.

I would also like to deeply thank Dr. Soumik Bhattacharya, Assistant Professor, Department of Electronics and Communication Engineering, IIT Roorkee, for providing me with his in-depth and exemplary guidance for making my research work an extremely satisfying and enriching experience. He selflessly and tirelessly, took out extra time from his busy schedule, and gave me a systematic and step by step approach to complete my research work, for which I would always be thankful to him.

I am also thankful to my esteemed institution for providing me with the opportunity and the required infrastructure to pursue this research. Finally, I would like to thank my family, friends and colleagues for their constant encouragement and motivation at all times.

Date :

Place : Roorkee

(Lt Col Amandeep Singh Puri)

Abstract

With the exponential increase in the number of private and public transportation systems, automatic detection and recognition of car license plates has translated into a pivotal technology that has enabled effective management of modern cumbersome terrestrial traffic. The technology has several diverse applications such as traffic monitoring, law enforcement, real time intelligence gathering, border management, intelligent transportation systems etc. At the same time there are many typical and challenging problems linked to it such as the variety of plate formats, varying geometry, non-uniformity in illumination, diverse weather conditions, complexity of backgrounds etc. The research brought out in this dissertation puts forward a novel method to counter yet another challenge for implementation of license plate recognition for real time traffic typically in an expressway environment, encompassing critical issues such as computation complexity and computation time, without compromising accuracy. Focus has been given on the enhancement of those sub-modules of License Plate Detection systems which are primary contributors towards processing time and computational complexity. A novel idea of a hybridized approach to counter the issues of cross-platform integration and flexibility has been put forward, wherein the accuracy of algorithm proposed will not be dependent on the resolution of the input image.

To further effectively leverage the results of successful detection of the license plate, a novel technique of character segmentation for accurate and efficient extraction of license plate number from the detected license plate has also been proposed. The proposed character segmentation technique is designed keeping in focus the issues related to adaptability to variable plate backgrounds and illumination without compromising the factors of processing time and computation complexity.

Contents

Acknowledgement	ii
Abstract	iii
List of Figures	v
List of Tables	vi
List of Algorithms	vii
Abbreviations	viii
Symbols and Notations	ix
1 Introduction	1
1.1 Background and Basic Concept	1
1.2 Motivation	3
1.3 Objectives Achieved	3
1.4 Thesis Organization	4
2 Literature Survey	5
2.1 Work Related to LPD	5
2.2 Work Related to CS	6
2.3 Real Time State of Art LPD Models	8
2.3.1 LPD Using Vertical Edge Detection Algorithm (VEDA)	8
2.3.2 LPD Using Extended Sobel and Line Density Filter (LDF)	9
2.4 State of Art CS Algorithms	11
2.4.1 CS Using MSER Algorithm	11
2.4.2 CS Using Watershed Algorithm	13
3 Proposed Hybridized LPR Model	16
3.1 Brief Overview	16
3.1.1 Detection	16
3.1.2 Character Region Segmentation	18

3.1.3	Recognition Using OCR	20
3.2	Hybrid LPD Module	20
3.2.1	Decision Module	20
3.2.2	Image Downsampling	20
3.2.3	Edge Detection	23
3.2.4	Adaptive Thresholding	26
3.2.5	Redundant Details Reduction Filter	27
3.2.6	Region Extraction Filter	28
3.2.7	Candidate Extraction Filter	31
3.2.8	Verification Using SLPC	32
3.3	Adaptive CS Module	34
3.3.1	Image Re-sizing and Otsu Thresholding	35
3.3.2	Adaptive Module	37
3.3.3	Unwanted Details Removal Using Border Thresholding	40
3.3.4	Removal of Objects of Low Pixel Value	41
4	Results, Comparative Analysis and Experimental Observations	43
4.1	Qualitative Results for Hybrid LPD Module	43
4.2	Qualitative Results for Adaptive CS Module	46
4.3	Quantitative Results of Hybrid LPD Module	47
4.3.1	Computation Time Results and Analysis	47
4.3.2	Results and Analysis of Detection Accuracy	50
4.4	Quantitative Results of Adaptive CS Module	51
4.4.1	Results and Analysis of Segmentation Accuracy	51
4.4.2	Results and Analysis of Processing Time	54
4.5	Analysis of Failure Cases: Hybrid LPD Module	55
4.6	Analysis of Failure Cases: Adaptive CS Module	56
5	Conclusions and Future Work	59
5.1	Conclusions	59
5.2	Future Work	60

List of Figures

1.1	A Typical LPR System.	2
2.1	Flowchart of VEDA based LPD method.	9
2.2	Flowchart of method using Extended Sobel and LDF.	10
2.3	Implemented model for CS using MSER	12
2.4	Illustration of Watershed Algorithm	14
3.1	Proposed Hybridized LPR Model	17
3.2	Flowchart of Hybrid LPD Module	21
3.3	Downsampling and Grayscale Conversion. (a) Original high resolution image. (b) Downsampled image. (c) Grayscale converted image.	22
3.4	VEDA mask	23
3.5	VEDA output. (a) Original low resolution image. (b) Grayscale conversion. (c) Edges generated by VEDA.	24
3.6	Extended Sobel Mask	25
3.7	Result of edge detection using Extended Sobel operator.(a) Downscaled grayscale image. (b) Edge detection result.	25
3.8	Output of AT.(a) Result of edge detection. (b) Binarized image.	26
3.9	Cases of pixel conversion using RDRF.	27
3.10	Visualization of functioning of REF. (a) Original image. (b) Result of edge detection. (c) Output of AT. (d) Output of REF.	31
3.11	Final result of detection. (a) Original image. (b) Result of edge detection. (c) Output of AT. (d) Output of REF. (e) Detected license plate region.	32
3.12	Flowchart of Adaptive CS Module.	34
3.13	Visualization of binarized output using Otsu thresholding. (a) Detected license plate image. (b) Binarized image. (c) Bi-modal histogram of the detected license plate image indicating potential for Otsu thresholding.	37
3.14	Flowchart of Adaptive Module	39
3.15	Visualization of binarized output using Otsu thresholding. (a) Detected license plate image. (b) Otsu thresholded image. (c) Correct binarized image derived from adaptive module.	40

3.16	Output of Border thresholding. (a) Detected license plate image. (b) Correctly binarized image using adaptive module. (c) Border thresholded image.	41
3.17	Removal of Objects of low pixel value. (a) Detected license plate image. (b) Correctly binarized image using adaptive module. (c) Border thresholded image. (d) Low pixel value objects removal.	41
4.1	Qualitative Results (Part 1). (a) Input images. (b) Results of Edge Detection. (c) Results of AT combined with RDRF. (d) Results of REF. (e) Detected License plates.	44
4.2	Qualitative Results (Part 2). (a) Input images. (b) Results of Edge Detection. (c) Results of AT combined with RDRF. (d) Results of REF. (e) Detected License plates.	45
4.3	Visual Results of Adaptive CS Module. (a) Detected license plate images. (b) Binarization using Otsu thresholding and adaptive module. (c) Result of Border thresholding. (d) Final segmented characters after removal of low pixel objects.	46
4.4	Visualization of stage wise computation time analysis for Hybrid LPD Module.	48
4.5	Visualization of comparative computation time analysis for common processes.	49
4.6	Graphical depiction of Detection Accuracy.	50
4.7	Segmentation performance of MSER.	52
4.8	Segmentation performance of Adaptive CS module.	52
4.9	Recognition performance of Tesseract.	53
4.10	Comparative analysis of segmentation and recognition performance.	54
4.11	Some typical failure cases of Hybrid LPD module. (a) Original input image. (b) Edge detection. (c) AT with RDRF. (d) REF output. (e) Detected region using Candidate filter.	56
4.12	Some typical failure cases of Adaptive CS module. (a) Detected license plate image. (b) Otsu thresholding using Adaptive module. (c) Border thresholding. (d) Final segmented image after removal of low pixel value objects.	57

List of Tables

4.1	Stage Wise Breakdown of Average Computation Time (in secs)	47
4.2	Comparative Analysis for Overall Average Computation Time (in secs) .	48
4.3	Comparative Analysis for Overall Average Computation Time for Common Processes (in secs)	49
4.4	Comparative Analysis Depicting Detection Rate in Percentages	50
4.5	Comparative Analysis of Accuracy in Terms of Percentage Success Rates.	53
4.6	Comparative Analysis of Average Processing Times (in secs).	54




List of Algorithms

1	Watershed Algorithm	15
2	Line Density Filter	30
3	Adaptive Module	38



Abbreviations



LPR	License Plate Recognition
LPD	License Plate Detection
CS	Character Segmentation
CR	Character Recognition
OCR	Optical Character Recognition
RDRF	Redundant Details Reduction Filter
MSER	Maximally Stable Extremal Regions
WA	Watershed Algorithm
VEDA	Vertical Edge Detection Algorithm
AT	Adaptive Thresholding
ULEA	Unwanted Line Elimination Algorithm
PRS	Plate Region Selection
PD	Plate Detection
REF	Region Extraction Filter
SLPC	Surged License Plate Classifier
SVM	Support Vector Machine
RGB	Red Green Blue
HSV	Hue Saturation Value
CCA	Connected Component Analysis
CCL	Connected Component Labeling
CRE	Candidate Region Extraction
CEF	Candidate Extraction Filter

Symbols and Notations

Q_{i^*}	Obtained MSER.
Δ	Step length of gray level threshold.
$\mathbf{J}(T)$	Pixel set in the recognized catchment basins (used in Watershed Algorithm).
$\mathbf{R}(T)$	Pixel set used to define the watershed.
\mathfrak{R}	Calculated resolution for decision module.
d_w	Downsampling scaling factor in the horizontal direction.
d_h	Downsampling scaling factor in the vertical direction.
Γ	Limiting value for edge intensity calculation using extended Sobel operator.
β	Coefficient for controlling value of threshold in AT.
$\omega(x, y)$	Average of all pixels within the local window used in executing AT.
h_w	Dimension value of the local window used for AT.
ξ	Extraction constant for CRE process
d_{line}	Value of line density calculated for application of LDF.
T_{min}	Threshold for minimum length (used in LDF).
T_{gap}	Threshold depicting gap length (used in LDF).
$T_{d_{line}}$	Threshold of line density of l_3 (used in LDF).
$\delta_{saliency}$	Colour saliency figure for SLPC.
ϕ	Quantization processing in case of SLPC.
μ_j	Mean values for individual R,G,B used for calculating feature vector in SLPC.
σ^2	Total variance of the input image used for optimal threshold calculation.
$\sigma_w^2(t)$	Total within class variance used for optimal threshold calculation.
Ω_{bin}	Binarized image generated by Otsu thresholding
Ψ_{bin}	Compliment of binarized image generated by Otsu thresholding
Φ_{bin}	Correctly chosen binarized image through adaptive module

Chapter 1

Introduction

License plate recognition(LPR) is an open problem under the domain of image processing utilized for categorizing and authenticating transportation systems by extracting their number plates. LPR systems can commonly be known by other terms like automatic number-plate recognition, automatic vehicle identification or car license plate recognition.

1.1 Background and Basic Concept

The topic became an attractive area of research because of its diverse day to day applications, such as parking payments, tariff payments at toll plazas, data analytics in traffic management, and crime patrol. Off lately it is being widely used for real time intelligence gathering by surveillance grid approach, speed control of cars on expressways, border control and intelligent transportation systems. To leverage the full potential of LPR in these fields, accuracy, run time efficiency and computation time of the algorithms executing it take a forefront.

A typical License plate recognition system has been shown in fig 1.1. The system is generally divided into three main modules: License Plate Detection (LPD), Character Segmentation (CS) and Character Recognition(CR). A brief explanation of the three modules is given as under:

- **License Plate Detection(LP D):** Once the vehicle image acquisition has been successfully carried out by the camera device, the LPD module carries out the pre-processing and ultimately the correct detection of the license plate region from the original image. Edge detection forms the core part of this module.
- **Character Segmentation(CS):** After the successful detection of the license plate region, the job of the CS module is to identify those regions in the license plate which contain the characters of the license plate. The correct recognition of the character regions is absolutely essential for the correct extraction of the license plate number in the subsequent stage.

- **Character Recognition(CR):** The identified character regions from the CS stage are then further processed by the CR module for the correct number plate extraction in a compatible text. Typically an optical character recognition(OCR) technique is used for this purpose.

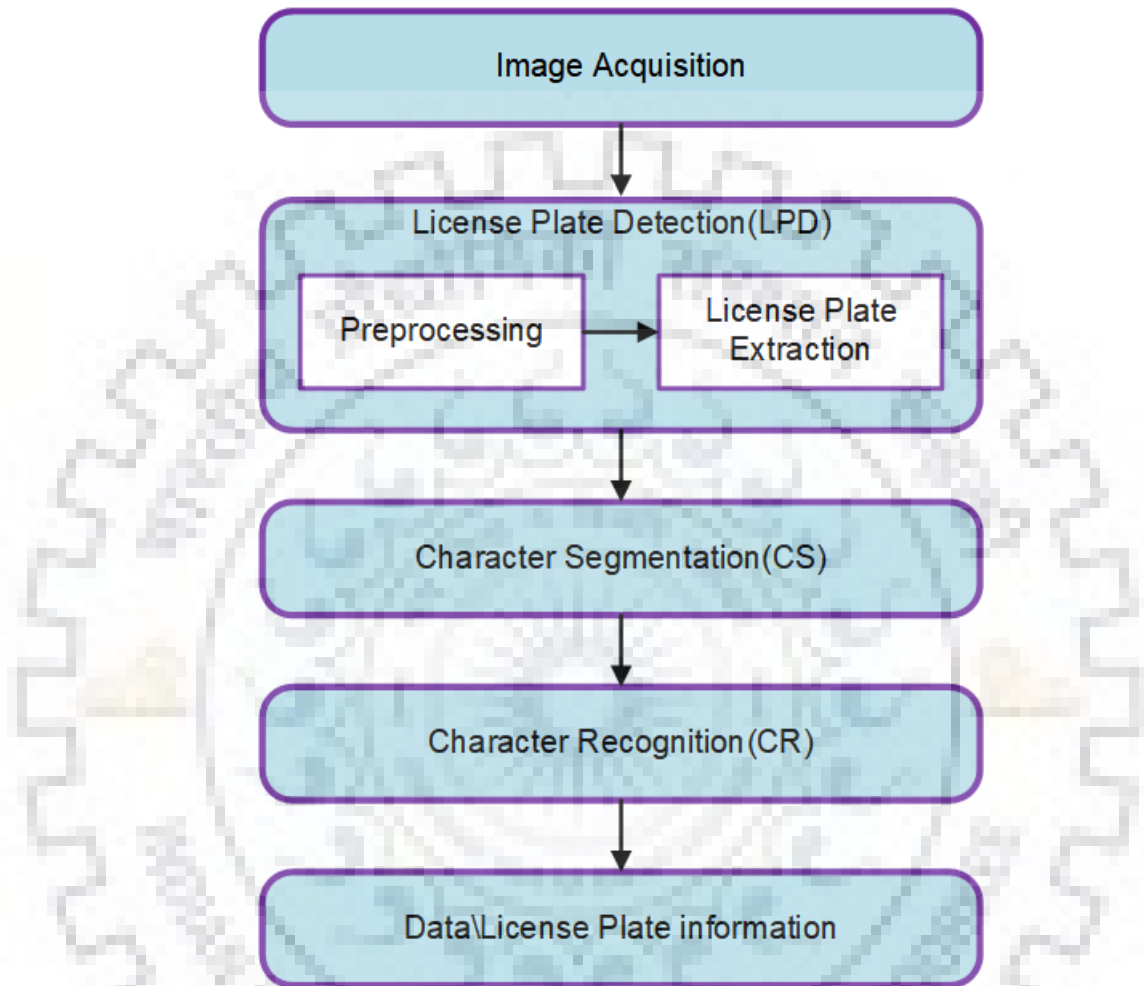


Fig. 1.1. A Typical LPR System.

Amongst all the three components LPD is the most critical one since, the net precision and speed of processing of the overall system is majorly contributed by LPD. This in turn dictates the computation time efficiency of subsequent modules of CS and CR, in turn impacting the effectiveness of the technology in real world applications as mentioned above. Over and above this, the accuracy and robustness of character segmentation and recognition modules becomes critical for effectively extracting the license plate number from the license plate and utilize it in standardized text formats for further analysis, storage and processing.

1.2 Motivation

A plethora of LPD techniques have been proposed and implemented over the past twenty years, and some of them have delivered reasonable and desirable results. However, it has come to focus that most of the previous techniques have performed up to the mark only in predefined, favorable and task-oriented conditions. Certain experimental limitations encompass uniform and rigid illumination conditions, input images with little or no blur or distortion, changes in viewpoint angles or trajectories, simplistic backgrounds and the presence of a singular license plate in an input sample. The latest state of art techniques, concentrate more on detection accuracy, while compromising on reduction of computational time, and still finding it challenging to accurately detect license plates in complicated environments. Furthermore the techniques in existence are rigid in terms of hardware deployed and are sensitive to inputs. Experimental observations and simulation results have brought out that, although some of the state of art methods are quite time efficient, they lack the flexibility to accept any input irrespective of the resolution it possesses. The impact on ground of such a problem results in replacement/redeployment of the entire hardware infrastructure in turn resulting in cost escalation as well as wastage of time and human resource. With this as the primary motivation, this dissertation brings out an idea of a hybridized approach towards license plate detection. This contributes to the key issue of flexibility particularly keeping in mind a large number of camera devices of different specifications deployed today.

To further augment the flexibility and the computation time efficiency offered by the hybridized plate detection there was a requirement to integrate it with an efficient accurate and robust character segmentation module, which should be adaptable to variety existing plate formats, as well as variable backgrounds and illumination. This served as a motivation to design a novel technique for character segmentation which has been brought out in subsequent chapters.

1.3 Objectives Achieved

The objectives attained, along with the theoretical and experimental contributions of the dissertation has been broadly summarized in the succeeding paragraphs.

- *Flexibility and Cross-Platform Integration* : The biggest strength of the model proposed is the flexibility it would offer at the detection stage, which was grossly lacking in the other existing state of art techniques, as discussed in the subsequent chapters. The input images to the model can be of any resolution, which in turn means that they can be extracted from camera devices of any configuration. This further entails that camera grids of different specifications can be seamlessly converged and true cross platform integration can take place.

- *Faster Convergence and Accuracy*: Since the algorithm has been hybridized utilizing the best sub-modules of the state of art techniques, the computation time achieved is of a very reduced value and is comparable to the techniques existing. This contributes for faster convergence towards plate detection. Moreover by utilizing two edge detection techniques instead of one, and utilizing a novel technique called Redundant Details Reduction Filter (RDRF), the accuracy of the proposed model emerges to be better/comparable to the techniques in existence.
- *Novelty in CS Module* : The suggested model integrates a novel technique for character segmentation. The novel CS module proposed counters a key issue of adaptability, keeping in mind variable license plate backgrounds and variable font colours of the characters. At the same time the technique is designed keeping the core issue of reduction of computational complexity and hence further contributes in the reduction of processing time. The improved performance of the proposed CS technique further augments the accuracy of the subsequent CR module, and has been brought out in both qualitative and quantitative sense through the results included in this dissertation.

1.4 Thesis Organization

The dissertation thesis has been structured as follows:

- *Chapter 1* provides a brief introduction to the topic, to include the basic concept and background behind its genesis. It touches upon the the various challenges linked to LPR systems, and the motivation behind pursuing the research. It also brings out the original contribution of the dissertation.
- *Chapter 2* brings out the work carried out over the years related to this thesis and the significant observations obtained through it. It also highlights state of art LPD and CS techniques which contributed towards faster detection and less computation complexity without compromising accuracy and robustness.
- *Chapter 3* describes the complete architecture, of the proposed hybridized LPR model including detail methodology of individual modules.
- *Chapter 4* puts forward simulation results, comparative analysis, and experimental observations obtained.
- *Chapter 5* concludes the thesis by bringing out the impact of the proposed model on modern day LPR systems and discusses the aspects of future research work.

Chapter 2

Literature Survey

2.1 Work Related to LPD

This section touches upon several LPD methods that have assisted in the maturing of this area of research over several years. Considering the reality that number plates are distinguished by available edge data, a variety of techniques utilize the information content of edges for detecting license plates. In [4], the values of magnitudes produced by vertical gradients are availed to detect probable license plate candidates in a segmented image. Shapiro et al. [5] used the Robert's edge operator to extract the vertical edges and then projected this edge information to extract license plates. Zheng brought out a method [6] that hunts for a license plate in a convolved digitized output image utilizing a rectangular shift operator of a fixed window size. However, the technique was restricted by the fact that only singular license plates could be detected in a frames containing multiple vehicles. Jia et al. [7] put forward a region-based methodology for license plate extraction that leverages the mean-shift technique to segment an input colour image of a vehicle, followed up by the analysis of edge density values for accurate verification. In [8], edge density prediction was carried out utilizing a block-based mechanism, which further aided to find out candidate license plate regions. Notwithstanding the fact that, the detection carried out in this model was fast, the accuracy of the system to predict the location of the license plate in the image depended majorly on the block size.

Linking the regions of individual characters is yet another important instrument for accurate license plate detection. Donoser et al. [9] suggested a model of LPD utilizing the idea of Maximally Stable Extremal Region (MSER) [10], enabling the parallel identification of positions and segmentation of distinctive characters of the license plate. Again, the method was able to acquire high detection rates in comparatively simple input samples, and not that with higher complexity. Morphology [11] is yet another important technique that has been applied widely by many researchers for successful license plate detection. In particular it is utilized to extract information related to structural data of license plates. Nevertheless, the morphology techniques again impinge considerably on

the detection time and are unfit for license plate detection in complicated surroundings. Furthermore, quite a few methods have exhaustively used color characteristics for plate detection, based on the observations that a typical license plate generally possesses an appearance of regular colours of both its backdrop and its content. In the model introduced in [12], a technique utilizing neural networks extracted colour values separately from all the hue, saturation and lightness channels. In [13] a proposal for fusing in colour and texture features for extraction of plate regions in images was derived. In [14], Tian proposed a plate demarcation method based on fixed values of colour pair for the content as well as the background areas of a license plate.

Most of the techniques discussed above were able to generate a fairly high detection rate but at the cost of computational time and computational complexity. Moreover, the above mentioned techniques were very effective only for trivial backgrounds and fixed resolutions. The work highlighted in this thesis concentrates on improving this gap thus making License Plate Detection faster while not compromising on the accuracy. The thesis also focuses on implementation of LPD and CS modules without compromising key issues of flexibility and adaptability, considering variable resolution of inputs and a large variety of existing plate formats and backgrounds.

2.2 Work Related to CS

In this section we focus on the work carried out over the years related to character segmentation of car license plate. The CS step is of prime importance as the accuracy of the recognition step is primarily dependent upon identification of the correct character regions. In [15] a character segmentation technique based on blob extraction was proposed. In this initially the image was converted into a binary image using Niblack's binarization algorithm. Thereafter, by analyzing the connected set of pixels (blobs), removing the blobs containing noise, and finally merging and splitting the blobs brings out the required character regions. Using Connected Component Analysis (CCA) of detected license plate image [16] is another methodology to carry out effective character segmentation. Using CCA for filtration of the components with undefined properties, correct character segmentation was carried out. Both the techniques mentioned above lacked the accuracy when it came to complex backgrounds and variable plate formats. In [17], after binarizing and pre-processing of the detected license plate image, character segmentation using a combination of vertical and horizontal projection method is carried out. The technique gives an average accuracy of about 97%, but suffers high value errors in case of fuzzy images, where the difficulty to separate characters from background becomes extremely high.

Using Hough transformation combined with prior knowledge in horizontal and vertical segmentation [18] is another robust technique to achieve good results. The method utilizes information related to intensities to negate absorption so as to overcome loopholes of

binarization. Although the segmentation achieved has high accuracy and robustness, the technique grossly fails with license plates having two-row alignment of characters and variable backgrounds. In [19], character segmentation was carried out utilizing a novel line scanning technique. This involved scanning the detected license plate image from left to right by using several innovative morphological operations. The technique brought out in [20] utilized a rule based segment analysis engine to derive the correlating regions containing license plates from the edge map images. However the weakness of this technique was that vertical edge components detected in the non-plate regions were also segmented, thereby reducing accuracy.

The technique brought out in [21] used an image scissoring algorithm to locate characters present in a license plate. For this purpose, a division of the license plate into several images takes place and then utilizing several morphological steps, character segmentation is performed. In [22], a CS technique based on morphology and partitioning is proposed. Input image samples of size 640x480 was utilized and tested under variable lighting environments at different distances. The method achieved a good accuracy of 94%, but at the cost of robustness and flexibility in input resolution. The model described in [23] consists of a prior knowledge based character segmentation technique. The prior knowledge comprises of parameters such as license plate size, character size, average gap between characters etc. The major drawback of this technique is the rigidity it offers when it comes to variable sizes of license plates and variable character sizes.

In [24], to negate the problems of diverse lighting conditions at different times of the day, as well as shadows in the image and adhesive characters, a template matching combined with a projection based technique is used. Projection is carried out in horizontal direction to find out the edges of a single license plate in vertical direction. A more fine tuned character segmentation is thereafter carried out by a template matching technique. The templates used are designed by taking into consideration the prior knowledge of the license plate. Thereafter by calculating the similarity measures the required characters are segmented using projection techniques as described above. Finding the required regions containing characters by utilizing horizontal and vertical smearing algorithms is brought out in [25]. The process comprised of four parts. Firstly enhancement is carried out by removing noise and parts which are unrelated. In the second part, a dilation operation takes place to separate the characters which are close to each other. Smearing algorithm is the third step and final step consisting of extraction of the final characters. In [26], a novel algorithm based on vector quantization is put into effect to solve the segmentation problem. The binary split tree is utilized for the vector quantization, in order to detect the boundary accurately and perform the plate region segmentation. The three methods mentioned above were able to achieve an impressive accuracy rate, but at the cost of again compromising computation time and increased complexity.

All the previously designed techniques brought out in this section are able to deliver accurate results with a related drawbacks as mentioned above. However a common loop-

hole of all them is that of adaptability with different formats of license plates, font colour and font styles. As mentioned above template matching can solve the problem to an extent, but with a trade off of either making the the algorithm too bulky and complex or too rigid for a particular type of plate. The CS technique proposed in this thesis tries to solve this problem and bring out true adaptability with any kind of plate format, yet at the same time not compromising on accuracy. Also, focus has been given to keep the algorithm lean enough, so as to not increase the computational time and complexity.

2.3 Real Time State of Art LPD Models

This section reviews two state of art LPD techniques to counter the challenges faced by modern day license plate recognition systems, in particular of that of minimum computational time, without compromising accuracy. Various important sub-modules of both the algorithms are briefly explained using a method flowchart. This is an important step as it helps in the understanding the comparative analysis of the results described in Chapter 4. The description of both the algorithms, will also assist us in conceptualizing and designing our own novel LPD model as described in Chapter 3.

2.3.1 LPD Using Vertical Edge Detection Algorithm (VEDA)

This model utilizes several segmentation techniques of image processing to get the desirable result. Initially, gray scale conversion of the low-resolution input image containing the target license plate takes place to facilitate easier segmentation. Thereafter, in order to binarize the image the process of Adaptive Thresholding (AT) [27] is undertaken. Followed by binarization, a novel module called Unwanted Line Elimination Algorithm (ULEA) [28] is executed to filter out the unwanted noise produced by AT and to enhance the binarized image. Subsequently, information related to vertical edge densities is obtained by utilizing the VEDA [28]. VEDA forms the heart of the method and is the primary contributor to reduce the computational complexity. The subsequent step is to localize the license plate and obtain the details required. To achieve this, initially the VEDA output is analyzed and plate information is highlighted using the net pixel value. Thereafter, operations related to statistical and logical nature are implemented to obtain probable candidate regions in the segmented image, followed by a converging procedure to localize the actual region. In the end, the correct plate region is extracted using the Plate Region Selection (PRS) and Plate Detection (PD) modules [28]. The flowchart of the proposed VEDA method is shown in fig 2.1.

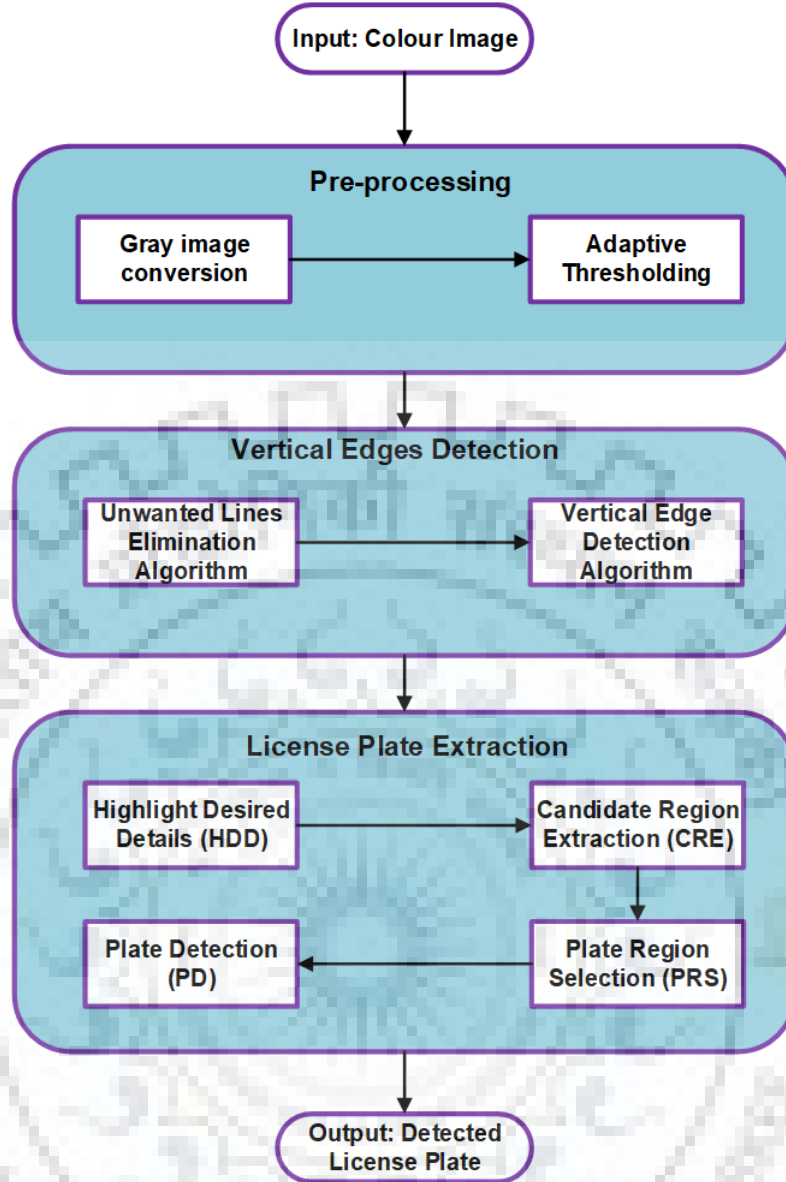


Fig. 2.1. Flowchart of VEDA based LPD method.

2.3.2 LPD Using Extended Sobel and Line Density Filter (LDF)

In this subsection we review the second state of art technique for efficient and accurate license plate detection. The method comprises of three main stages, namely, image pre-processing, candidate extraction and license plate verification. As given by the flowchart in fig 2.2, the input high resolution image first undergoes downscaling operation for easier pre-processing without impinging on computation time. Thereafter, the down scaled output undergoes gray scale conversion for easier segmentation.

Subsequently, a group of regions as possible candidates are obtained utilizing edge detection using novel extension of Sobel operator [29]. After edge detection an adaptive thresholding (AT) operation is applied for binarization of the processed image for enhanced and easy convergence of the required license plate region. The operation has

been executed the same way as used in the previous method. After the digitized image is generated by AT, there is a need to bring to focus the desired license plate region and exterminate unwanted details in the binary image. For this purpose, a novel technique called Line Density Filtering [29] is used. The underlying concept of LDF is to connect regions of high edge density and remove sparse regions in each row and column in a binary edge image derived from adaptive thresholding.

Once Line Density Filtering is carried out, the resultant image is then processed through the candidate filter module, which using the bounding box technique extracts the required license plate. The license plate is then verified for its colour saliency using a SVM classifier [29] which verifies and reconfirms the license plate region using its RGB characteristics.

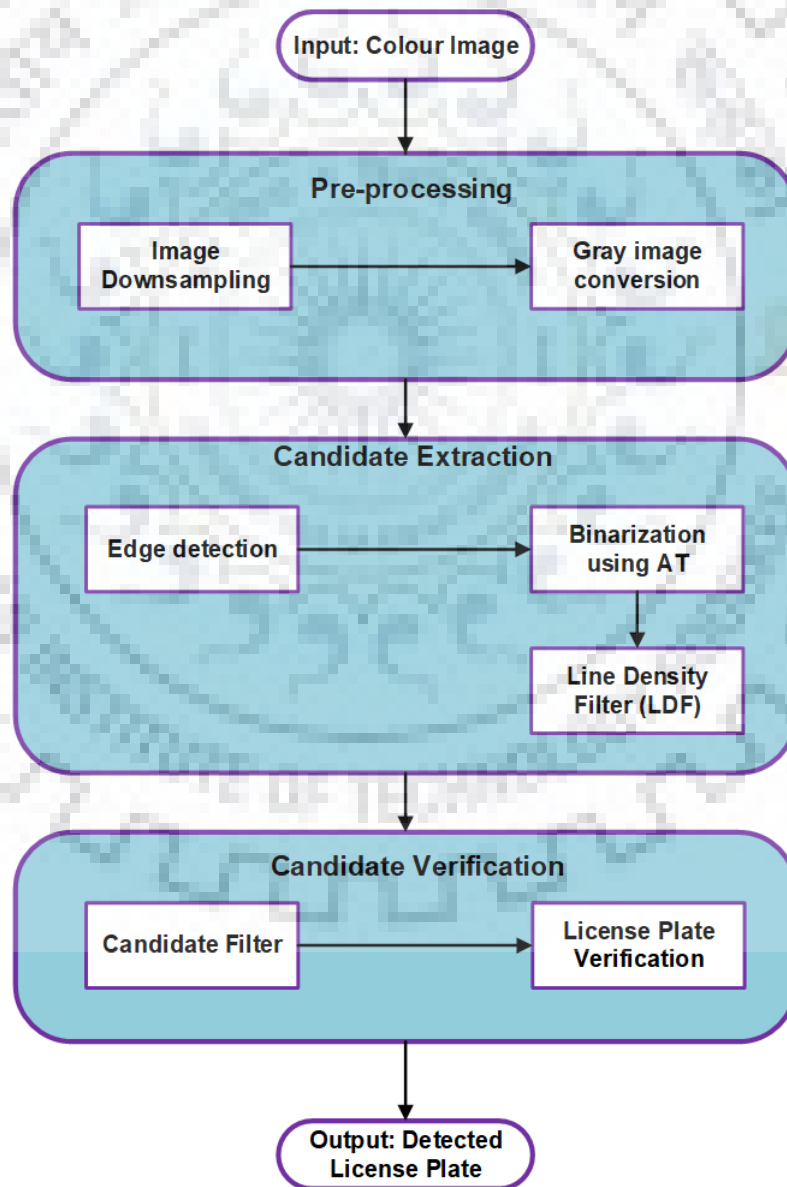


Fig. 2.2. Flowchart of method using Extended Sobel and LDF.

2.4 State of Art CS Algorithms

In this section two state of art algorithms used character segmentation, i.e. the *MSER* and *Watershed* algorithm have been described. The MSER algorithm has been implemented in model based approach for simulation and comparative analysis purposes. The two algorithms will be briefly described the subsequent subsections.

2.4.1 CS Using MSER Algorithm

MSER is a technique of blob detection in images. The MSER algorithm retrieves a variety of co-variant regions from an image called MSERs. In other words MSER is a steady connected component of a few gray level groups of an image. The idea is of taking regions which remain nearly the same through a wide range of thresholds. The word extremal is linked to the property that every pixel within the MSER possess either higher intensity (bright extremal regions) or lower intensity (dark extremal regions), as compared to all pixels on its outer boundary.

Initially, MSER was put forward to resolve the matching problem in wide baseline stereo vision as brought out in [30]. The advantage accrued was of outstanding affinity invariance. This typicality can also be utilized in the CS module of a LPD algorithm. Another reason for MSER working well in case of CS is the high contrast and consistent colour of text leading to stable profiles of intensity.

To describe it broadly, some variable thresholds are utilized to binarize the target image and thereafter the regions derived from the binarized image with the highest stable area variation are categorized as MSERs. To put it across more formally, at first, utilizing the thresholds from 0 to 255, the image undergoes binarization. Thereafter, assuming that gray level pixels less than a particular threshold are converted to white, and the ones higher than the threshold are converted to black we obtain a series of black and white regions called the extremal regions, which can be shown as $Q_i, \dots, Q_{i-1}, Q_i, \dots, (Q_i \subset Q_{i+1})$. When the area within a particular extremal region is stable for a wide range levels of gray, then this region is called MSER. Utilizing the mathematical expression, that is applicable if and only if $i = i^*$,

$$q(i) = \frac{|Q_{i+\Delta} \setminus Q_{i-\Delta}|}{|Q_i|} \quad (2.1)$$

one gets the local minimum, and MSER Q_{i^*} is derived. Here Δ denotes the symbol giving the gray level threshold step length. The other details and definitions related to the above mentioned mathematical formulation is given in [30]. Further, using the fact that gray levels can be tuned in two opposite directions, and after applying the two operations, we obtain the contrasting bright and dark extremal regions. The obtained difference of the two extremal regions can be effectively utilized to segment character regions out of the

detected license plate region.

The universal implementation of MSER is described above. In the model implemented for the purpose of stimulation and comparative analysis with the proposed adaptive CS module, additional morphological and statistical sub-modules have also been integrated. The flowchart of the implemented MSER model is given in fig 2.3.

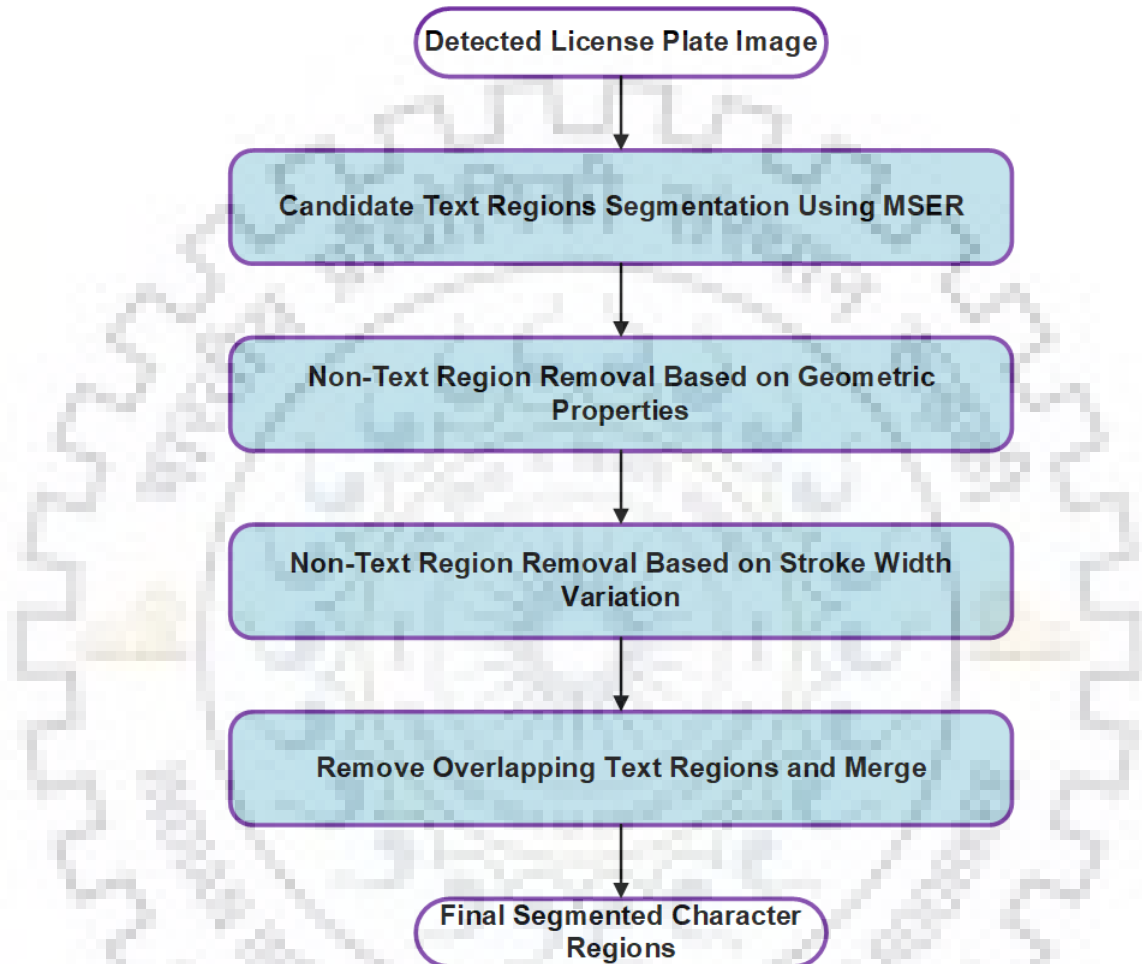


Fig. 2.3. Implemented model for CS using MSER

As given in the flowchart above, the detected license plate image region segmentation using the basic MSER algorithm. However there are many non text regions detected along with the desired character regions. To further converge to the desired character regions, initially a rule based approach is applied. In this the properties related to the geometry of the characters is utilized to further filter out the non-character stable regions. In this particular implementation, we have used simple thresholds for geometric properties filtering out non-character regions. As an alternative one can utilize a machine learning approach for training a classifier to separate a text from a non text region. A combination of the two can also be used [31]. There are several geometric properties like *aspect ratio*, *eccentricity*, *Euler number*, *extent*, *solidity* etc. which are used for differentiating text

from non text regions [32,33].

After removal of non-text regions utilizing geometric properties, a module to further remove the non-text regions based on *stroke width* variation is applied. *Stroke width* is a common parameter utilized for distinguishing between non-character and character regions. It is a measure of lines and curves that make up a character. Non-character regions tend to have higher stroke width variation than character regions. The stroke width of the MSER regions can be calculated using binary thinning as well as distance transform operation [33]. In order to remove non-character regions using stroke width, the complete MSER region under consideration, a stroke width variation metric should be computed first. Thereafter we apply a threshold for removal of the non text regions. The threshold value may be varied depending on different styles of fonts. The above mentioned procedure is then applied to each and every geometrically filtered MSER region separately to get the overall result.

After using the stroke width module, the output comprises of individual characters. To enable the individual OCR module to be effective, the individual characters should be merged together. The approach used in this particular case is first finding adjacent regions and then utilizing expanding bounding boxes merge the character regions. For this, an overlap ratio in the middle of all expanding bounding box pairs is calculated. The grouped overlap ratio is then fed to a graph function to merge all character regions linked by overlap ratios which are non zero. This module greatly enhances the accuracy of the subsequent CR module and successfully delivers the final segmented character regions.

2.4.2 CS Using Watershed Algorithm

Watershed Algorithm (WA) [38] is another celebrated tool in image processing for effective segmentation of images. The name watershed is drawn from the geological feature of watershed, which is nothing but a divide between two side-by-side drainage basins. WA treats a particular image having multiple objects as a topographical map with brightness levels of each pixel corresponding to a particular height. Subsequently, similar to a topographical map where we have contour lines, the algorithm connects these heights corresponding to the pixel values and is able to find a divide between objects similar to a watershed.

To understand WA, let us consider the fig given below. The basic concept of WA revolves around the geological concept of *catchment basins* and the demarcation between these catchment basins called the *watershed*. These catchment basins correspond to the local minimum in a topographical image and the idea of WA is to find the corresponding local maximum.

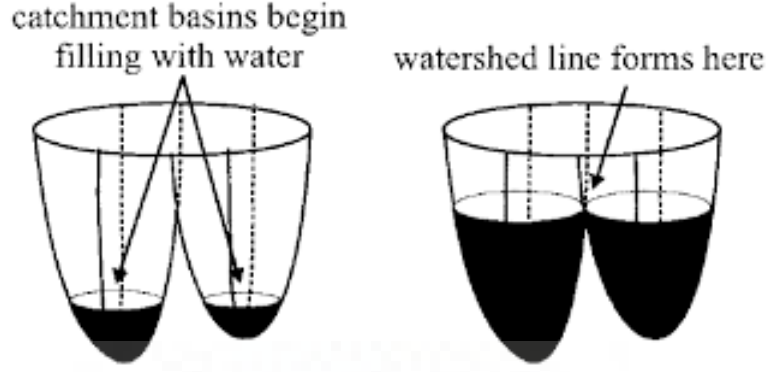


Fig. 2.4. Illustration of Watershed Algorithm

To understand the working of the algorithm, let us consider that a hole is enforced in each of the local minimum and the entire topographical feature is flooded from the bottom, through the hole and the water is allowed to rise at uniform rate. The pixels that are present just below the water level at any given point of time are marked as flooded and start defining the object regions. As the water level increases, so does the size of the flooded regions. This will keep happening to a point where the water level in each of the catchment basin will equalize and merge. When this merger happens the algorithm constructs a one pixel value thick dam that separates the two image regions. This process continues till the point that all objects in the image in the form of catchment basins are separated by watershed ridge lines, thus resulting in effective and clear cut segmentation.

To put across the concept more formally and mathematically, let us assume that the maximum and minimum gray values in the image under consideration, namely I , are given by $\max(I)$ and $\min(I)$ respectively. Also, the set of co-ordinates in the local minimums existing in I are given by $Min_1, Min_2, \dots, Min_i, \dots, Min_R$. Now, $C(Min_i)$ is the set of coordinates linked with Min_i , $g(x, y)$ is the value of gray level of individual pixel (x, y) , and $C_{g(x,y)}(T)$ is the set corresponding to coordinate (x, y) and $f(x, y) < T$ and is given by the expression as under:

$$C_{g(x,y)}(T) = \{(x, y) | g(x, y) < T\} \quad (2.2)$$

Now $J(T)$ is the pixel set in the recognized catchment basins when the value of gray level is T and is given by the following expression as:

$$J(T) = \bigcup_{i=1}^R \left\{ C(Min_i) \cap C_{g(x,y)}(T) \right\} \quad (2.3)$$

Now $R(T)$ is the pixel set in the watershed given by the equation as under:

$$R(T) = I - J(T) \quad (2.4)$$

Taking the above equations into account, the WA in the form of a iteration process acting circularly is given in Algorithm 1 as under:

Algorithm 1: Watershed Algorithm

Input: Grayscale image \mathbf{I}_{gray}

Output: $\mathbf{J}(T)$ which is the pixel set in the catchment regions and $\mathbf{R}(T)$ which is the pixel set of the watershed

Step 1. Initialization: $\mathbf{C}(Min_i)$ is originally constructed and $T = min(\mathbf{I}) + 1$ is calculated ;

Step 2. $\mathbf{C}_{g(x,y)}(T)$ is calculated using (2.2);

Step 3. $\mathbf{J}(T)$ is calculated using (2.3);

if $T \leq max(\mathbf{I}) + 1$ **then**

 | go to *Step 2* ;

else

 | $\mathbf{R}(T)$ is calculated using (2.4) and pixel set belonging to watershed boundary is derived;

end

The algorithm explained above is used in many state of art LPR systems such as one used in [39]. The overall concept of WA is quite similar to that of MSER algorithm and delivers a very robust and accurate performance when it comes to character segmentation and augmenting optical character recognition.

Chapter 3

Proposed Hybridized LPR Model

In this chapter we describe in detail the proposed LPR model, including all major modules, to include all stages and techniques used within each module. The chapter will start with a brief overview of the entire LPR system. Thereafter each module i.e. the *Hybrid LPD module*, the *Adaptive CS module* and the *CR module* will be discussed in detail in subsequent sections.

3.1 Brief Overview

The flowchart of the proposed LPR model is given in fig 3.1. The brief overview of the detection, character segmentation and recognition modules is given in subsequent subsections.

3.1.1 Detection

The hybrid LPD module is divided into three main stages i.e., the *pre-processing stage*, *candidate segmentation stage*, and finally the *candidate extraction and verification stage*. A point to note here is that the input to the algorithm can be of any resolution or in other words from any camera device of any specification. All three stages of the algorithm are briefly described below.

- *Pre-processing*: In this stage a novel idea of a *decision module* has been introduced. The function of the decision module is to calculate and analyze the resolution of the input image and then take the decision for the subsequent pre-processing and segmentation steps. Once the decision module has analyzed the input image then depending upon the decision taken, the image is first downsampled and then gray scale converted (in case of high resolution) or else is directly gray scale converted. The downsampling algorithm is designed depending upon the dimensions of the license plate and the characters being considered. Using the dimensions of the input image, and pre-defined downscaling factors for both the dimensions, downscaled proportions are calculated using bi-linear interpolation process.

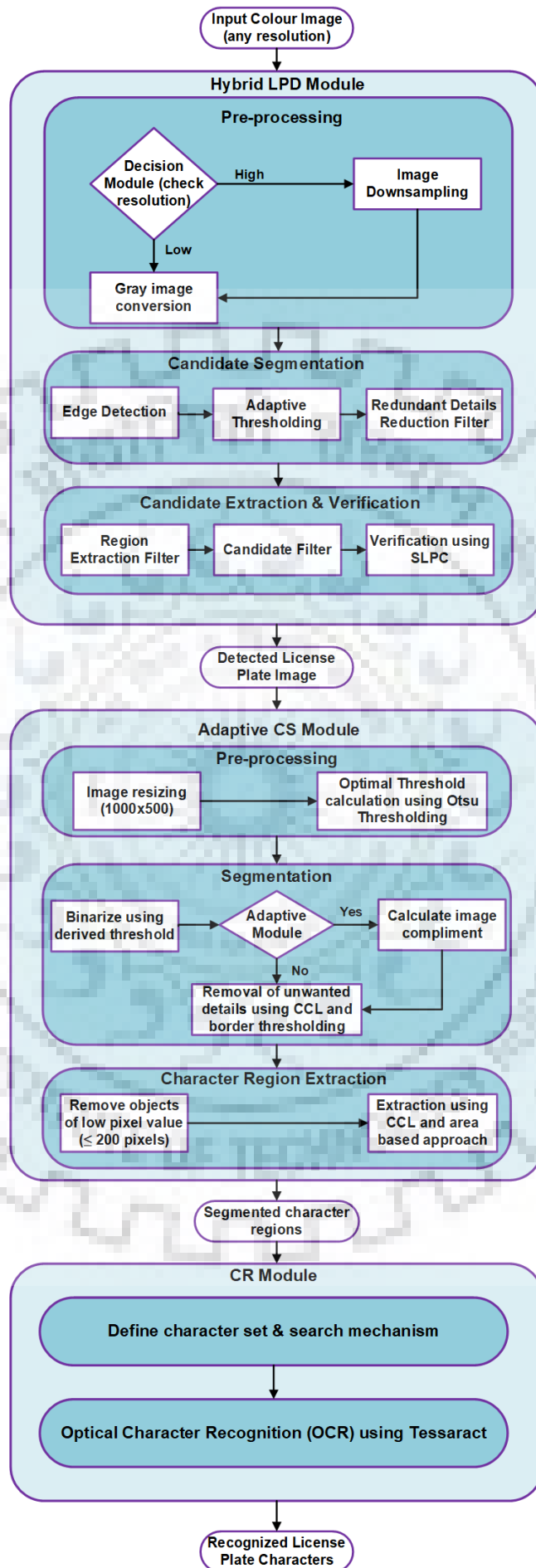


Fig. 3.1. Proposed Hybridized LPR Model

- *Candidate Segmentation:* At this stage the gray scale converted image from the previous stage, initially undergoes edge detection. The detection technique applied will also be decision based depending upon the resolution of the image i.e., VEDA [28] based in case of low resolution and extended Sobel [29] based in case of high resolution. The edge detection will then be followed by adaptive thresholding (AT) [27], by using the concept of an integral image. The final step would be to further segment the image by using the RDRF. In this technique we try to remove the redundant one pixel value lines produced by thresholding. To achieve this, while taking the values of black pixels as background, a 3x3 mask, is utilized to test only the black pixel values. Once the pixel value under consideration, which is located at the center of the mask, is detected to be black, then testing of eight-neighbor pixels is executed. If two subsequent values are collectively white, then the pixel under consideration is transformed into a white pixel matching the foreground. Similarly, all the line which are unwanted are systematically deleted from the image. This in turn assists in faster convergence of the algorithm at the extraction and verification stage.
- *Candidate Extraction & Verification:* In the final stage, we first apply the *Region Extraction Filter* (REF) on the segmented image. The main function of this filter will be to find the most probable regions in the image which can qualify to be the license plate. Region extraction filtering is driven by the observation that license plates contain a comparatively higher value of edge intensity, with the characters printed on the license plate having horizontal orientation, and yet at the same time the size of each character is almost alike. Another fact is that if an input sample comprises of more than one license plate then sufficient territorial gap will be present between them. Using the factors described above, the underlying concept of Region extraction filter is to connect regions of high edge density and remove sparse regions in each row and column in a binary edge image derived from adaptive thresholding. The output of the Region Extraction Filter will be then fed to the candidate filter which using connected component labeling and a voting based process will detect the license plate. Once the license plate is detected, the verification process takes place, which is done by a *Surged License Plate Classifier* (SLPC) based on linear SVMs. Considering the motivating fact that license plates contain fixed background colours (white, black, blue and yellow), the technique utilizes feature vectors from the original image are extracted from both the RGB and HSV colour spaces to arrive at a more robust and accurate verification result.

3.1.2 Character Region Segmentation

The proposed adaptive CS module is subdivided into three stages i.e. the *Pre-processing stage*, *Segmentation stage* and the *Character Region Extraction stage*. The module was

designed with an aim to make the CS process comparable and in some aspects better than methods utilizing state of art techniques like MSER [30]. The three stages are briefly described below.

- *Pre-processing:* In this stage, initially the detected license plate image is re-sized to 500x1000 pixels. This is done to normalize the subsequent segmented character regions and achieve compatibility with the template characters in the CR module. After this *Otsu thresholding* [34] is performed to calculate the optimal threshold required for binarization operation. The motivating factor behind using Otsu thresholding comes from the fact that an accurately detected license plate image will contain two class of pixels i.e. the foreground containing the license plate number and the background contain the license plate colour. Thus in such a scenario, a successful bi-modal histogram can be obtained and by minimizing the intra-class variance or in equivalent terms, maximizing the inter-class variance (as defined by Otsu's method), optimal threshold value can be derived.
- *Segmentation:* At this stage, firstly the detected license plate image is binarized using the optimal threshold derived from Otsu's method and then given as input to a novel algorithm called the *Adaptive Module*. The fact that license plates can exhibit a dark or light coloured background and contain black or white coloured characters, as well as can suffer from illumination problems due to environmental conditions, the results of binarization may not be consistent and may result in complete failure. With this as the primary motivation, the task of the *adaptive module* is to decide whether the binarized image is fit for further processing and segmentation or not. This is in turn is decided by identifying (using specific morphological operations) the exact foreground and background regions, irrespective of their colour and intensities. Thereafter, a decision is taken whether to take the compliment of the binarized image or not, so as to make it compatible for further processes. The output obtained by the *adaptive module*, is then applied with a *border threshold*. The border threshold is applied keeping in mind the logic that invariably, the license plate number in the detected image will lie around the centroid. Hence to eliminate extra objects in the background and concentrate only on the character regions, first Connected Component Labeling (CCL) is applied and thereafter all the components touching the defined, and further compressed border, are removed.
- *Character Region Extraction:* To further prepare the segmented image for successful extraction, initially a filter is applied to remove objects of low pixel values (≤ 200 pixels). The filtered image is applied to extraction sub-module, wherein first using CCL and then area based decision approach the individual character regions are extracted.

3.1.3 Recognition Using OCR

Once the characters are successfully segmented they are fed to the CR module for conversion into a viable text format. For this particular model we have used *Tesseract*, which is an open source tool for optical character recognition. To refine the recognition, the character set is defined to include the English alphabet and the standard number set. Also the search mechanism is tuned to line wise search to further increase the accuracy of recognition. The important point to focus here is that for the accurate and efficient functioning of this module, it is very important to have an efficient and robust CS module.

3.2 Hybrid LPD Module

This section discusses in detail all the stages and the techniques required to execute the hybridized approach towards the license plate detection. The flowchart for the Hybrid LPD module is given in fig 3.2. The various sub modules of the algorithm have been described in detail in the subsequent sub-sections.

3.2.1 Decision Module

As mentioned earlier the basic motivation of incorporating this module is to provide much needed flexibility towards accurate detection of license plates. Initially, the module at first measures the actual width and height of the input colour image in pixels. Once measured it carries out resolution calculation by simply multiplying the height with the width. Thereafter, a decision is taken to downsample the image and then grayscale convert it or to directly apply grayscale conversion process to the input image. This depends on the simple threshold relation given as $\mathfrak{R} \leq 150000$. Here the threshold value of 150000 has been empirically derived after exhaustive experimentation and observing the performance of the subsequent sub-modules. The symbol \mathfrak{R} denotes the resolution calculated. The detection module not only provides flexibility but also reduces the computation time in case of low resolution images by completely avoiding the downsampling step.

3.2.2 Image Downsampling

In the scenario when $\mathfrak{R} > 150000$ i.e., the image is of a high resolution, the next step towards LPD is the downscaling operation. The basic idea behind the downscaling operation is to reduce the high cost of computation imposed on the overall system for effectively extracting the license plate from the input image. Sadly, the downscaling operation may result in a losing information and significantly reduced license plate detection performance. For fixing this issue, an innovative process to downscale an input sample has been put forward. The implemented downscaling process considerably decreases the size

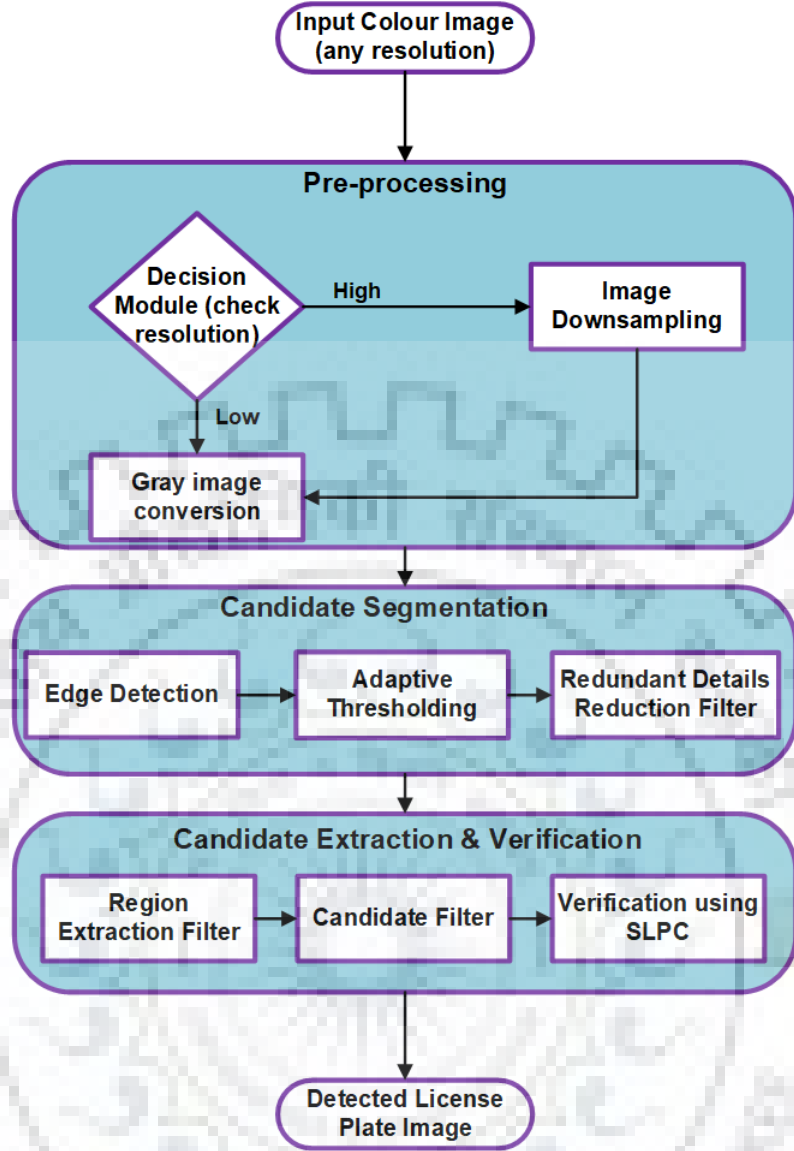


Fig. 3.2. Flowchart of Hybrid LPD Module

of the input sample, and at the same time the overall performance in terms of accuracy is similar, as compared to that obtained by an input sample of high resolution. The technique is basically linked to license plate geometry. There are two main considerations. The primary consideration is that the length of a license plate always has a larger value as compared to the height. The secondary consideration is that the license plate information is embossed horizontally. Exploiting these considerations, we create scaling factors for downscaling process in terms of vertical and horizontal orientation. These are given as follows

$$w_s = \frac{w_i}{d_w} \quad (3.1)$$

$$h_s = \frac{h_i}{d_h} \quad (3.2)$$

Here, w_i and h_i denote the width and height, of the input sample and w_s and h_s give out the converted downscaled proportions. The variables, d_w and d_h (such that $d_h < d_w$) are the downscaling factors, for both the dimensions. A point to note is that d_w , which is used for downscaling of the original input image along the horizontal direction, is taken to be larger as compared to d_h . There are two reasons for this adoption. Firstly, more compression of the image data can be carried out in the horizontal direction. This is because of the fact that the height of the license plate is generally less than the width. Secondly, taking d_w larger contributes to more compactness of characters on the license plate, which in turn contributes to more accurate results at REF module. In the implementation of the downsampling module, $d_w = 3$ and $d_h = 2$. Utilizing these values, bilinear interpolation involving computing output in terms of pixel value by averaging out the weights of the closest pixels in a 2×2 neighborhood is carried out to get the desired result. In the case when $\mathfrak{R} \leq 150000$, the downsampling process is completely skipped and the input image is directly grayscale converted and supplied to the next stage for further segmentation and processing.



Fig. 3.3. Downsampling and Grayscale Conversion. (a) Original high resolution image. (b) Downsampled image. (c) Grayscale converted image.

In the results related to one input shown in fig 3.3 the resolution of the original image is 3984×2241 . Whereas the resolution of the image after the downscaling operation is 1192×747 , justifying the significant impact of downscaling without loss of information. The downscaled image then undergoes gray image conversion as shown in fig 3.3(c).

3.2.3 Edge Detection

The generated gray scale image then undergoes the process of edge detection. As mentioned earlier, the edge detection process is again flexible in nature depending upon the resolution of the image. For low resolution image the VEDA [28] based edge detection technique is applied. For high resolution images an extended Sobel [29] based edge detection technique is used. The application of the edge detection technique also depends on the output of the decision module. The two edge detection techniques are explained in detail in the subsequent paragraphs.

VEDA

The basic concept of VEDA is converging on the crossings of high and low intensity pixel regions in the grayscale image. For this purpose, a 2×4 mask is introduced, as shown in fig 3.4, where the variables x depicts the row and y depicts the columns of the grayscale sample. The pixel under consideration is kept at coordinates $(0, 1)$ and $(1, 1)$. Masking operation is carried out from left hand side to right hand side to find the abrupt intersections between low intensity and high intensity pixels, keeping only the last two pixels. Similarly, for intersections between high intensity and low intensity pixels, the initial low intensity pixel is kept.

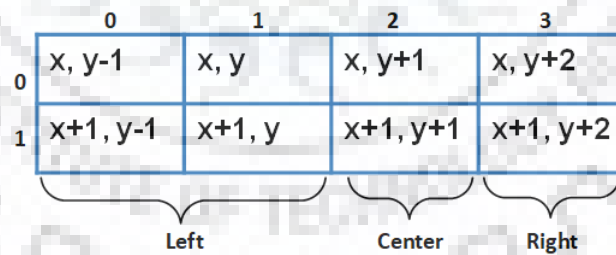


Fig. 3.4. VEDA mask

The size of the VEDA mask is of the size 2×4 and is divided into three individual sub-masks. This is designed to achieve the following two objectives:

- The mask on the extreme left hand side is the first sub-mask of the size 2×2 . The mask in the center of the size 2×1 is the second sub-mask of size 2×1 and the sub-mask on the extreme right is the third one and is again of the size 2×1 . All three sub-masks are marked individually in fig 3.4. After checking two pixels together,

the application of the first sub-mask takes place which considers width of two pixels for detection. This particular procedure is applied for detection of edges of vertical orientation at the intersections of low-high intensity regions. In the same way the third sub-mask is applied on the intersections of high-low intensity regions. Due to this the process the edge detected has a pixel width of 1.

- Since two rows are considered in one go, the time consumption using VEDA is half as compared to each row being singularly checked.

For edge detection, the pixels located at $(0, 1)$ and $(1, 1)$ are considered to be the center of the VEDA mask. Two value pixels and one value pixel by considering low-high intensity and high-low intensity intersecting regions respectively are kept. The mask starts traversing from left to right and top to bottom to find the relevant edges. The result of a single low resolution sample applied with VEDA operator is given in fig 3.5.

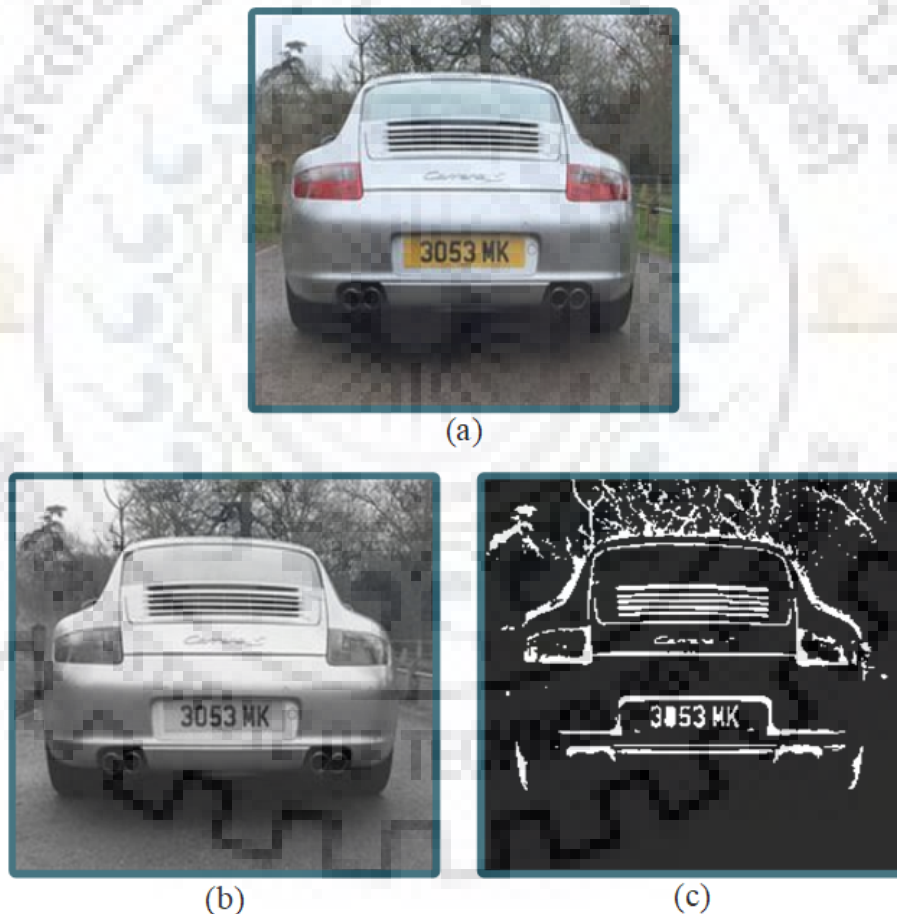


Fig. 3.5. VEDA output. (a) Original low resolution image. (b) Grayscale conversion. (c) Edges generated by VEDA.

Extended Sobel Operator

This edge detection technique is used for the high resolution images i.e. $\mathfrak{R} > 150000$. For the purpose of edge detection and edge density enhancement, a simple extension of Sobel

operator as shown in fig 3.6 is used.

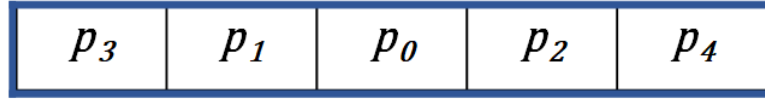


Fig. 3.6. Extended Sobel Mask

Here p_0 depicts the pixel under current consideration and p_1, p_2, p_3 and p_4 denote the nearest neighborhood pixels of p_0 . The edge intensity e_0 for pixel p_0 is given by the following expression:

$$e_0 = \begin{cases} \Gamma & \text{if } d_0 \geq \Gamma \\ d_0 & \text{else } d_0 < \Gamma \end{cases} \quad (3.3)$$

where Γ is the pre-defined limiting value and d_0 is computed as under:

$$d_0 = |p_1 + p_2 - 2 \times p_0| + |p_3 + p_4 - 2 \times p_0| \quad (3.4)$$

Extended Sobel operator utilizes a single 1×5 mask, in comparison to a typical Sobel operator utilizing a couple of 1×3 or 3×3 masks to execute convolution. The standard Sobel operator does gradient estimation in both horizontal and vertical directions. Whereas the extended Sobel does gradient estimation in the vertical direction only. It was found through exhaustive experimentation that 1×5 mask offered more robustness when it came to edge detection than a 1×3 convolution mask. The limiting value of Γ was defined to be 48 in the experimental implementation, to steer clear from stack overflow of the succeeding AT process. Result of the application of extended Sobel operator on a single sample for visual interpretation has been shown in fig 3.7.



Fig. 3.7. Result of edge detection using Extended Sobel operator.(a) Downscaled grayscale image. (b) Edge detection result.

3.2.4 Adaptive Thresholding

To further augment the resultant estimated edges produced by edge detection and for generation of a binarized image E_{bin} , an AT operation [27] is performed. Using the earlier edge generated grayscale image E_{gray} , the AT technique initially is utilized for the generation of an integral image E_{intgrl} . This done by summation of all pixel values from the left upper corner with regards to every pixel in E_{gray} . Thereafter a binarized edge image E_{bin} is produced by executing thresholding operation over every pixel $p(x, y)$ present in the integral image E_{intgrl} . This is done utilizing a threshold value calculated adaptively using a *local window* in E_{intgrl} . Explicitly, the value of $E_{bin}(x, y)$ is calculated as given by the following expression:

$$E_{bin}(x, y) = \begin{cases} 255 & \text{if } E_{intgrl}(x, y) \geq \beta\omega(x, y) \\ 0 & \text{else } E_{intgrl}(x, y) < \beta\omega(x, y) \end{cases} \quad (3.5)$$

where the coefficient β is utilized for controlling the threshold value and $\omega(x, y)$ signifies the average value of the entire pixels contained in the $h_w \times h_w$ local window around the current pixel $p(x, y)$. The value of $\omega(x, y)$ is given by the expression:

$$\omega(x, y) = \frac{1}{h_w \times h_w} \sum_{\substack{-h_w/2 < k < h_w/2 \\ -h_w/2 < j < h_w/2}} E_{intgrl}(x + k, y + j) \quad (3.6)$$

For the implementation of the proposed hybrid LPD model the value the variables β and h_w was assigned the values 0.7 and 20 respectively. For better assimilation and visualization the output of AT process for a single input is given in fig 3.8.



Fig. 3.8. Output of AT.(a) Result of edge detection. (b) Binarized image.

3.2.5 Redundant Details Reduction Filter

To further refine the output of AT and remove the redundant details for better execution of candidate extraction and verification stage, the module of RDRF is applied. RDRF can be summarized as a *morphological technique* and *intensification process*. It considers four cases for unwanted pixel values in continuity (unwanted lines). In the first case, horizontal line with an angle of 0° is considered. This is followed up by vertical line with 90° angle in second case. Next case includes a line inclined at an angle of 45° . The final case considers an inclined line of 135° . The final aim of RDRF is to remove all the above cases with one-pixel value. To achieve this, while taking the values of black pixels as background, a 3×3 mask, as given in fig 3.9, is utilized to test only the black pixel values. After the pixel value under consideration, which is located at the center of the mask, is detected to be black, then testing of eight-neighbor pixels is executed. If two subsequent values are collectively white, then the pixel under consideration is transformed into a white pixel matching the foreground. Similarly, all the line which are unwanted are systematically deleted from the image. A point to note here is that in case the thresholded image undergoes a compliment conversion (i.e. the white pixels become black or background becomes foreground), then the RDRF operation also undergoes complimentary conversion and instead of considering black pixel value at the center of the mask, white pixel value is considered. RDRF not only provides an easier image to handle for subsequent region extraction and verification, but also assists the subsequent modules to detect the license plate in a faster manner.

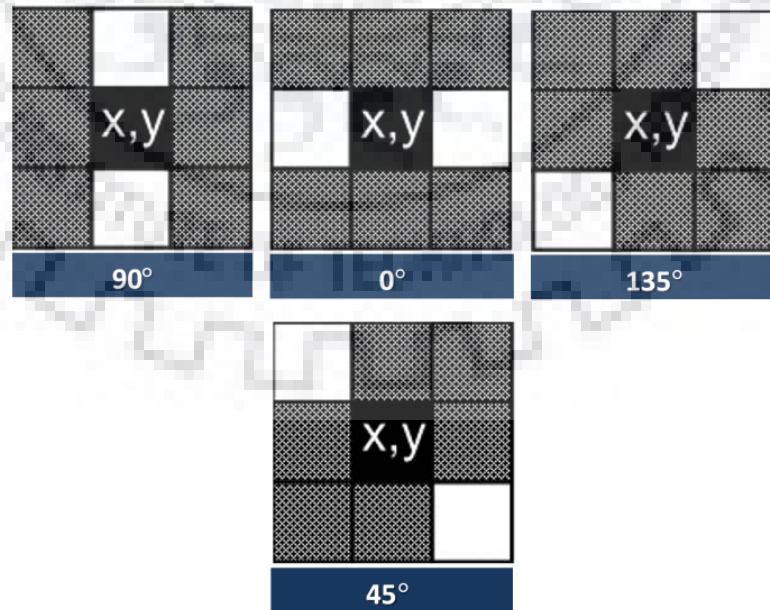


Fig. 3.9. Cases of pixel conversion using RDRF.

3.2.6 Region Extraction Filter

The task of the REF module is to generate the most probable candidates which can qualify to be the actual license plate region in the supplied image. The REF module is executed in two main parts. In the first part the process of *Candidate Region Extraction (CRE)* [28] is executed. This process is then cascaded by the process of LDF [29], which forms the second part. The idea of using both techniques is motivated by the primary requirement of accuracy of detection clubbed with computation time efficiency. The two techniques are described in the subsequent paragraphs.

CRE

The main objective of the CRE is to demarcate the probable regions in binarized image which can be further supplied to LDF for further processing and extraction of the most probable candidates. The process of CRE is further divided into four steps:

- *Step 1: Counting the Drawn Lines in every Row.* The total drawn lines in every row are calculated. Thereafter, this value is assigned to the variable having matrix structure called $HwMnyLines[a]$, where $a = 0, 1, \dots, height - 1$.
- *Step 2: Dividing the Image into Multiple sets.* The large quantity of rows will increase the time taken for processing in the subsequent modules. Hence in order to lessen the time consumption, bunching multiple rows into one entity is executed in this step. The division of the image values into multi-groups could be done using the following expression:

$$how_many_groups = \frac{height}{\xi} \quad (3.7)$$

where how_many_groups signifies the net value of groups, $height$ depicts the quantity of image rows, and ξ gives the extraction constant. In other words, ξ represents a single group (bunch of rows). In the implemented model brought out in this thesis, the value of ξ is taken to be 10. This was chosen because empirically it was found out that by using the value 10 the desired details were not lost much, as well as, computation time efficiency was not compromised.

- *Step 3: Counting and Storing Approved Group Indices and Regions.* Almost all of the line groups do not form part of the target region. Hence, it is handy to utilise a limiting value to abolish unsatisfactory grouping and to hold the approved grouping in which the license plate details are present. A total of 15 lines will be checked per group. If it meets the limit of 15, then it is taken as a chunk of the license plate region. Eventually, the net groups to include the slices of license plate regions will be calculated and kept.

- *Step 4: Choose Boundaries of Probable Candidates.* In this process the demarcation of each of the candidate region is carried out. This is done to narrow down the region of application of the LDF for more time efficient execution.

LDF

After the most probable regions are highlighted by CRE, there is a need to bring to focus the desired license plate region and exterminate unwanted details in the further segmented binary image. For this purpose, a technique called LDF is used. Line Density Filtering is driven by the observation that license plates contain a comparatively higher value of edge intensity, with the characters printed on the license plate have horizontal orientation, and yet at the same time the size of each character is almost alike. Another motivating fact is that if an input sample comprises of more than one license plate then sufficient territorial gap will be present between them. Using the factors described above, the underlying concept of line density filtering is to connect regions of high edge density and remove sparse regions in each row and column in a binary edge image derived from adaptive thresholding, RDRF and subsequently segmented by CRE.

Now consider the binary image E_{bin} obtained from adaptive thresholding. Supposing that there are two continuous black line pieces represented by l_1 and l_2 . Each are having lengths in pixels represented by l_{blk1} and l_{blk2} respectively. The part of the line targeted, containing a combination of edge, as well as non edge pixels, is represented by l_3 . It is this part (l_3), which requires either connection or removal of the white pixels. The decision to remove or keep the white pixels will depend on the line density represented by d_{line} . The value of d_{line} is given by the following relation:

$$d_{line} = \frac{w_{l_3}}{w_{l_3} + b_{l_3}} \quad (3.8)$$

where, w_{l_3} and b_{l_3} represent the value of white pixels and black pixels in l_3 .

In an ideal condition, we take the assumption that non-license plate regions exist on either side of the actual license plate region. Putting it across in another way, there has to be comparatively lengthy pieces of black lines (l_1 and l_2) present on either side of the sporadic line segment i.e. l_3 , which has to undergo processing. In order to establish a relation between the three pieces of lines under consideration i.e. l_1 , l_2 and l_3 , a unique process for calculating line segment lengths is applied. A single black pixel is taken as 1 and similarly a single white pixel is taken as -1. If uninterrupted black pixels are existing, they are grouped together and cumulative length is given by a positive number. In the same way, uninterrupted white pixels are calculated, given by a negative number, and cumulative length of the white line is derived.

Utilizing the line-extent map containing each pixel of E_{bin} generated with horizontal orientation, LDF is then applied to extract the exact license plate candidate regions. The

pseudo code for LDF oriented in horizontal direction is given in Algorithm 2. Four input variables contained in the pseudo code are given as under:

- The threshold for minimum length of a black line (l_1 or l_2) represented by T_{min} .
- Threshold depicting length of the gap (gap between l_1 and l_2 or in other words length of l_3), represented by T_{gap}
- Threshold of line density d_l for the piece of line l_3 , designated by $T_{d_{line}}$

For the purpose of implementation of the algorithm the value of T_{min} , T_{gap} , and $T_{d_{line}}$ is set to 20, 8 and 0.05 respectively. A point to note that in a special case, a candidate could be placed too near to either left or right of the boundary of the image i.e., either l_1 or l_2 have a length $< T_{min}$. In such a case, re-application of the algorithm should be done for processing l_3 . The three cases for most probable candidates are given by variables X_1 , X_2 and X_3 .

Algorithm 2: Line Density Filter

Input: Binarized image E_{bin} and threshold parameters T_{min} , T_{gap} and $T_{d_{line}}$
Output: Binarized image E_{bin} containing license plate candidates.
 Calculate length map M_{hlen} of horizontal orientation using E_{bin} as input;
for Every line in length map M_{hlen} **do**
 Identify two pieces of line l_1 and l_2 with respective lengths l_{blk1} and l_{blk2} ;
 $X_1 = l_{blk1} > T_{min}$ and $l_{blk2} > T_{min}$;
 $X_2 = l_1$ is the leftmost piece of line and $l_{blk1} < T_{min}$ and $l_{blk2} \geq T_{min}$;
 $X_3 = l_2$ is the rightmost piece of line and $l_{blk1} \geq T_{min}$ and $l_{blk2} < T_{min}$;
 if X_1 or X_2 or X_3 **then**
 Calculate d_{line} for l_3 ;
 if $d_{line} \geq T_{d_{line}}$ **then** set all values of pixels of l_3 to white in E_{bin} ;
 else set all values of pixels of l_3 to black in E_{bin} ;
 end
end

Once both the stage of the REF are executed, we get the most probable candidate regions highlighted with white lines. To visualize the output of REF the result obtained from one input sample is given in fig 3.10.

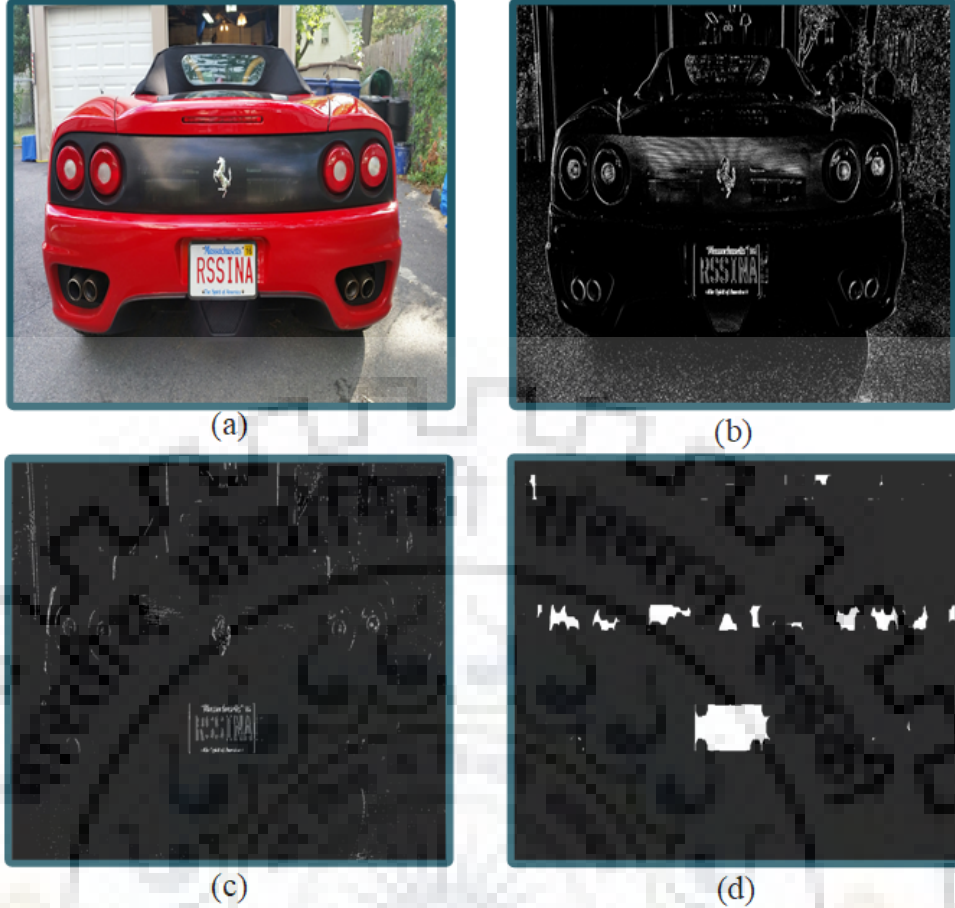


Fig. 3.10. Visualization of functioning of REF. (a) Original image. (b) Result of edge detection. (c) Output of AT. (d) Output of REF.

3.2.7 Candidate Extraction Filter

After acquisition of the most probable candidate regions utilizing the REF, there is a requirement to further filter out the actual license plate area from other regions. To accomplish this task a module called the Candidate Extraction Filter (CEF) is utilized. The functioning of the CEF is divided into two stages. In the first stage, CCL is put into effect to remove to look for most probable candidates by removing areas that are not exhibiting geometrical properties of a license plate, defined by the the following:

- $24 < w_c < 256$
- $6 < h_c < 32$
- $256 < w_c \times h_c < 4096$
- $w_c/h_c < 6$

where w_c and h_c denote the width and height limits, of the target candidates. The above given specifications are selected on the basis of minimum and maximum dimensions

(widths and heights of a large variety of license plates, covering the prospective properties of most practical settings. These parameters are modifiable to suit dimensional characteristics of a variety of countries and administrative regions. Once CCL is carried out successfully, the using bounding box technique the actual license plate is extracted. The final output with the input image and intermediate results is given in fig 3.11.



Fig. 3.11. Final result of detection. (a) Original image. (b) Result of edge detection. (c) Output of AT. (d) Output of REF. (e) Detected license plate region.

3.2.8 Verification Using SLPC

To further verify and ensure that the detected license plate is correct one, the detected license plate region is then checked for its colour. This technique is motivated by the fact that the license plate consists of two dominant colours the one in the background and that in the foreground. To verify that the license plate region is the correct one or not, the background colour is checked. For this purpose four colours are considered in

the suggested classifier module, i.e., blue, yellow, black and white. Rest all colours are grouped in the as non plate regions.

For the execution of the above mentioned requirement a Surged License Plate Classifier (SLPC) based on linear SVMs [37] and motivated by colour saliency verification techniques [35,36] is used. The basic idea behind the suggested method is to use both the RGB and HSV are used in a cascaded manner to derive the feature vector for classification. Further, to exterminate the regressive effect of variation in illumination and at the same time to keep computational complexity as less as possible, a quantization approach for colour feature depiction is used. In the colour space specific to HSV, the hue is quantized to 8 bin and the saturation and value each are quantized to 3 bin. This in turn generates a colour feature vector of 72 ($8 \times 3 \times 3$) dimensions. For every colour level $c \in (0, 72)$ we derive a saliency figure represented by $\delta_{saliency}$ and given by the expression given below:

$$\delta_{saliency} = \frac{\sum_{x=\frac{2w_{can}}{3}, y=h_{can}}^{x=\frac{2cw_{can}}{3}, y=0} \phi(p(x, y))}{h_{can} \times w_{can}/3} = c \quad (3.9)$$

where, ϕ denotes the processing for quantization and h_{can} denotes the height and w_{can} denotes the width of the detected license plate region. To augment the HSV colour space for achieving a more accurate and robust result, the values of the mean (denoted by μ_j), calculated individual for all three R, G, B channels are imbibed in the feature vector. The expression used for the calculations is given below:

$$\mu_j = \frac{\sum_{x=\frac{2w_{can}}{3}, y=h_{can}}^{x=\frac{2cw_{can}}{3}, y=0} \mu_j(x, y)}{255 \times h_{can} \times w_{can}/3}, j \in \{R, G, B\} \quad (3.10)$$

By surging the the feature vector of HSV space comprising of 72 dimensions within the 3 colour values obtained in the RGB space, each detected license plate region can be depicted by a feature vector of 75 dimensions. Utilizing this modified and improved feature vector a more accurate and reliable colour classification and hence license plate verification is executed.

3.3 Adaptive CS Module

Once the successful detection of the license plate region takes place, the next requirement is to efficiently and accurately derive the individual character regions in the detected license plate region, which in turn will support and augment the character recognition process. This section brings out the a novel adaptive CS module to fulfill the above mentioned requirement. All the stages and sub-modules involved in the CS module have been explained in detail in the subsequent subsections. The flowchart of the adaptive CS module is given in fig 3.12.

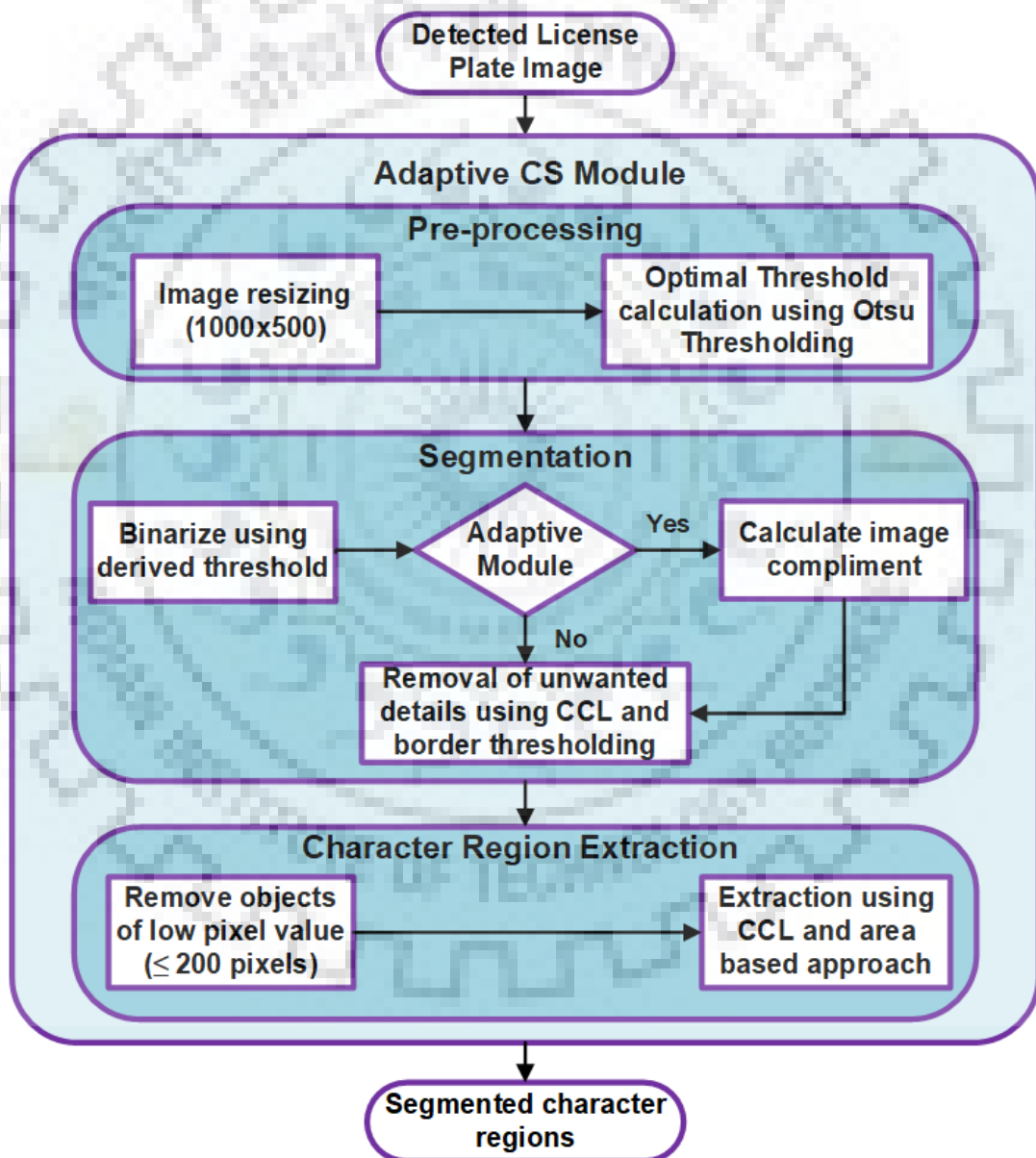


Fig. 3.12. Flowchart of Adaptive CS Module.

3.3.1 Image Re-sizing and Otsu Thresholding

This subsection brings out the pre-processing involved prior to the actual segmentation and extraction of the character regions. Initially, the detected license plate image is re-sized to a factor of 100×500 . This re-sizing operation is done using bicubic interpolation in a 4×4 neighborhood (16 pixels). This image re-sizing operation is carried out with the purpose of not missing out on any wanted details in the character region. After re-sizing operation, Otsu thresholding process [34] is applied to the re-sized detected license plate image.

Otsu thresholding is motivated by the fact that there are two prominent colour regions in the detected image i.e., the foreground containing the character regions and the background contain the plate on which the characters are embossed. The Otsu thresholding algorithm works on the assumption that the target image has only two pixel classes i.e., the pixels in the foreground and the pixels in the background, generating a bi-modal histogram. Using this assumption, the algorithm derives the optimal threshold dividing the two pixel classes making their overall spread or in other words the intra-class variance minimum. This in turn ensures that the inter-class variance is maximum.

Otsu thresholding is not looking at edges inside an image, but is trying to find regions inside the image to segment out objects. The basic idea is simple, i.e., we have minimize the within class variance (when two classes are available). The within class variance is defined by the following expression:

$$\sigma_w^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t) \quad (3.11)$$

where t is the optimal threshold we are looking for and whose value is varied accordingly. $\sigma_w^2(t)$ is the within class variance we are trying to minimize. Also, $q_1(t)$ and $q_2(t)$ is the probability of class 1 and class 2 respectively and is calculated as under:

$$q_1(t) = \sum_{i=1}^t P(i) \quad (3.12)$$

$$q_2(t) = \sum_{i=t+1}^I P(i) \quad (3.13)$$

where $P(i)$ is the probability of every pixel value obtained from the histogram and I is the largest pixel value (usually 255). Now once the individual class probabilities are calculated, we obtain the mean of the individual classes as given as:

$$\mu_1(t) = \sum_{i=1}^t \frac{iP(i)}{q_1(t)} \quad (3.14)$$

$$\mu_2(t) = \sum_{i=t+1}^I \frac{iP(i)}{q_2(t)} \quad (3.15)$$

where $\mu_1(t)$ and $\mu_2(t)$ are the means of class 1 and 2 respectively. After obtaining the mean we calculate the variance of both the classes by the following expressions:

$$\sigma_1^2(t) = \sum_{i=1}^t [i - \mu_1(t)]^2 \frac{P(i)}{q_1(t)} \quad (3.16)$$

$$\sigma_2^2(t) = \sum_{i=t+1}^I [i - \mu_2(t)]^2 \frac{P(i)}{q_2(t)} \quad (3.17)$$

where $\sigma_1^2(t)$ and $\sigma_2^2(t)$ are the variance of class 1 and 2 respectively. By calculating all the above given values we obtain the weighted within class variance by using equation 3.11. Now it is adequately clear that $\sigma_w^2(t)$ is a measure for the compactness of the classes. If we put the threshold in the wrong way, there is going to a large variance within the classes. Thus the threshold needs to be optimized which is exactly what Otsu thresholding does.

Now one of the ways to obtain the optimized threshold, is by putting the values of t recursively in equation 3.11 and compute till the time minimum value of $\sigma_w^2(t)$ is obtained. However, this involves cumbersome calculation and will slow down the thresholding process. This can be resolved by a little modification given by the following expression:

$$\sigma^2 = \underbrace{\sigma_w^2(t)}_{Term1} + \underbrace{q_1(t)[1 - q_1(t)][\mu_1(t) - \mu_2(t)]^2}_{Term2} \quad (3.18)$$

where σ^2 represents the total variance of the image and $Term2$ represents the between class variance. Now from equation 3.18 it is clear that minimizing $Term1$ is equivalent to maximizing $Term2$. This is because σ^2 is a constant term. Also looking at $Term2$, we observe that while updating the threshold t , the probability and mean are easier to update, as we move across the entire histogram. Thus $Term2$ can be computed very easily in a recursive fashion, thus making the implementation extremely effective and fast. In order to make Otsu thresholding failure free in non-uniform illumination cases, where we do not get a bi-modal histogram, we have utilized for the implemented adaptive CS module, a local thresholding mechanism involving the moving average windowing technique.

The obtained optimal threshold in the re-sized detected license plate image is then utilized for binarizing the image into equivalent values of 0s and 1s. The image is then

further given to the adaptive module for further segmentation and processing. The visualization of the Otsu thresholding and binarization output is given in fig 3.13.

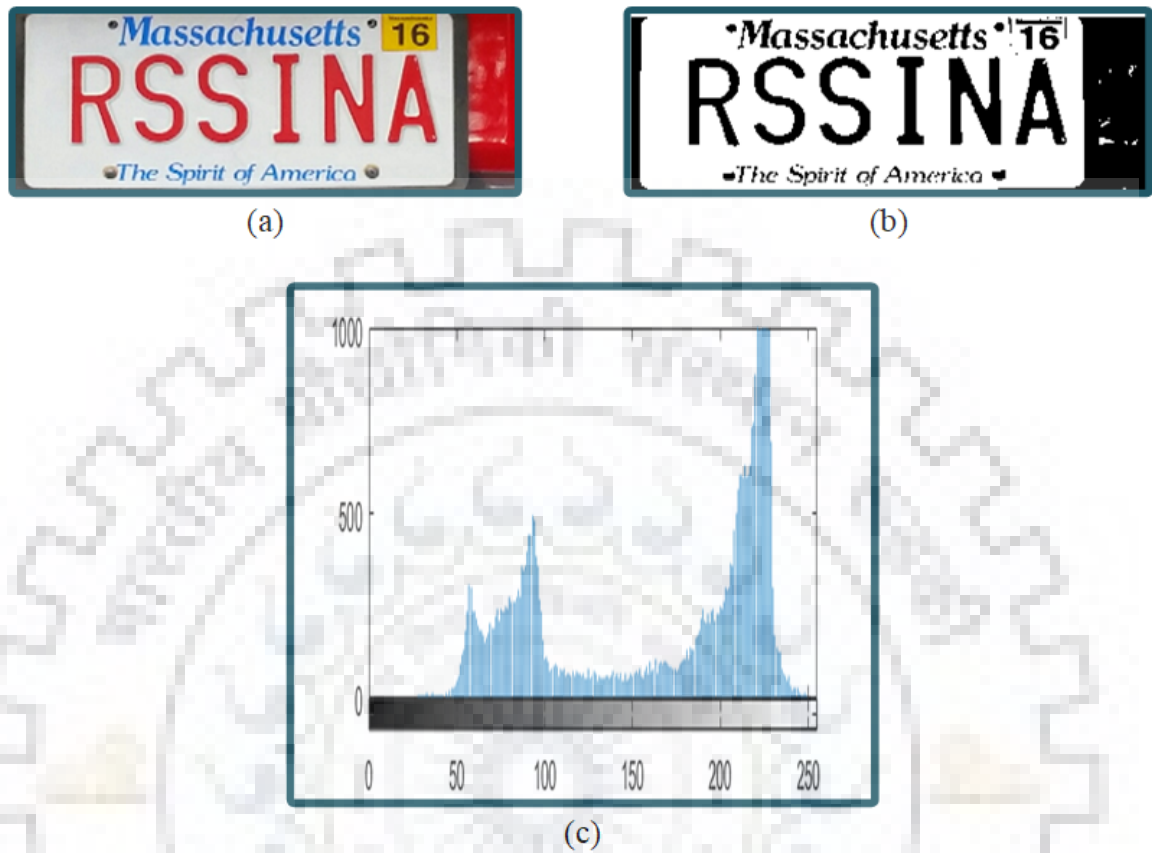


Fig. 3.13. Visualization of binarized output using Otsu thresholding. (a) Detected license plate image. (b) Binarized image. (c) Bi-modal histogram of the detected license plate image indicating potential for Otsu thresholding.

3.3.2 Adaptive Module

The use of Otsu thresholding to obtain the desired segmentation result is not enough. This is because the thresholding result can be a success or a failure depending on the variation in colour and intensity of license plate background and characters might vary. For example, if the license plate comprises of a white background with black characters then there will be confusion in the treatment of the foreground and background pixels thus resulting in the losing of the information of the character regions in turn resulting in total failure at the character recognition stage. One solution to the problem in hand is to take the compliment of the thresholded image to derive the clearly demarcated candidate regions. However, taking the compliment will have to be done on a case to case basis and decision to take the compliment would have to be taken manually. This is not a true implementation of thresholding as successful implementation of the process is not universal for all types of detected license plate images and has to be intervened manually.

Hence to make the thresholding process successful and infuse adaptability in the overall CS algorithm the *adaptive module* is proposed. The adaptive module in fact forms the heart of the entire proposed CS module. The pseudo code depicting the functioning of the adaptive module is given in Algorithm 3 as given below.

Algorithm 3: Adaptive Module

Input: Binarized image Ω_{bin}
Output: Adaptively chosen correct binarized image Φ_{bin} .
Calculate Ψ_{bin} using Ω_{bin} ;
Implement array padding of Ω_{bin} and Ψ_{bin} to generate $P_{\Omega_{bin}}$ and $P_{\Psi_{bin}}$ respectively;
Using $P_{\Omega_{bin}}$ and $P_{\Psi_{bin}}$ generate $F_{\Omega_{bin}}$ and $F_{\Psi_{bin}}$ respectively;
if $sum(F_{\Omega_{bin}} - P_{\Omega_{bin}}) > sum(F_{\Psi_{bin}} - P_{\Psi_{bin}})$ **then**
 | $\Phi_{bin} = \Omega_{bin}$;
else
 | $\Phi_{bin} = \Psi_{bin}$;
end

The variable description used in the algorithm is given as follows:

- Ω_{bin} representing the binarized image derived from Otsu thresholding and serving as input to the adaptive module
- Ψ_{bin} denoting the compliment of the binarized image.
- $P_{\Omega_{bin}}$ and $P_{\Psi_{bin}}$ are the images generated by the padding up with white pixel values images Ω_{bin} and Ψ_{bin} respectively. The padding is carried out in a 2×2 boundary. This process is carried out for successful implementation of the hole filling operation.
- $F_{\Omega_{bin}}$ and $F_{\Psi_{bin}}$ are the images generated by hole filling operation on $P_{\Omega_{bin}}$ and $P_{\Psi_{bin}}$ respectively.
- Φ_{bin} representing the correctly processed binarized image and is the output of the algorithm.

For better assimilation the flowchart of the Adaptive module is given in fig 3.14 below.

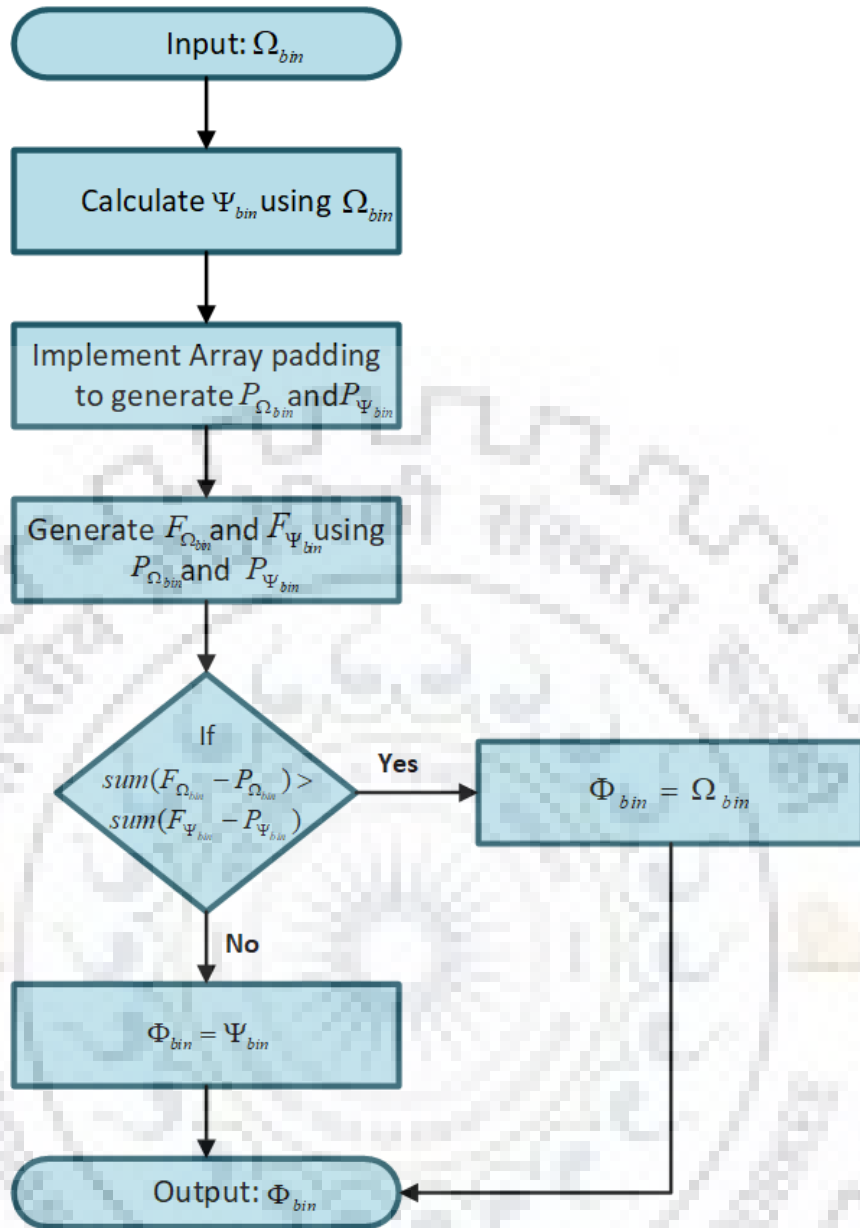


Fig. 3.14. Flowchart of Adaptive Module

This simple yet highly effective module provides a high level of adaptability to the entire CS module and provides the capability for character segmentation for a large variety of license plates. Fig 3.14 displays the qualitative result obtained for one sample, depicting the output when the adaptive module is applied and when simple thresholding with compliment is applied. Through the result it is observed that if we take output as displayed in Fig 3.14(b), the subsequent CR module will fail in detection as it will not be able to differentiate the foreground and background part correctly. This is because in a binarized image the foreground is depicted by white pixel value and background by black pixel value and the OCR module will only concentrate on the foreground. This in turn will result in total failure at the recognition stage.



Fig. 3.15. Visualization of binarized output using Otsu thresholding. (a) Detected license plate image. (b) Otsu thresholded image. (c) Correct binarized image derived from adaptive module.

3.3.3 Unwanted Details Removal Using Border Thresholding

After the successful binarization of the detected license plate image is carried out, it is important to filter out the other unwanted objects on the detected license plate image except for the required character regions. This process is partly carried out by the border thresholding module. The main logic behind this module is that the license plate number is generally present in the centroid region of the detected license plate image. Therefore if the search mechanism only concentrates on the centroid and filters out the details touching the border of the license plate image, then the unwanted details can be filtered out. This is exactly what the proposed border thresholding module does. If we take a particular threshold limit of the border, then we define a function based on CCL which removes all objects touching or beyond the defined border limit. By default we have taken the value of the border threshold to be 0 as we assume that the detection module has done accurate detection and the characters as a result are concentrated at the centroid. The output of the border thresholding module for one sample is given in fig 3.15. If we compare outputs of the adaptive module and the border thresholded image, then we see that the extra information touching the border have been removed.

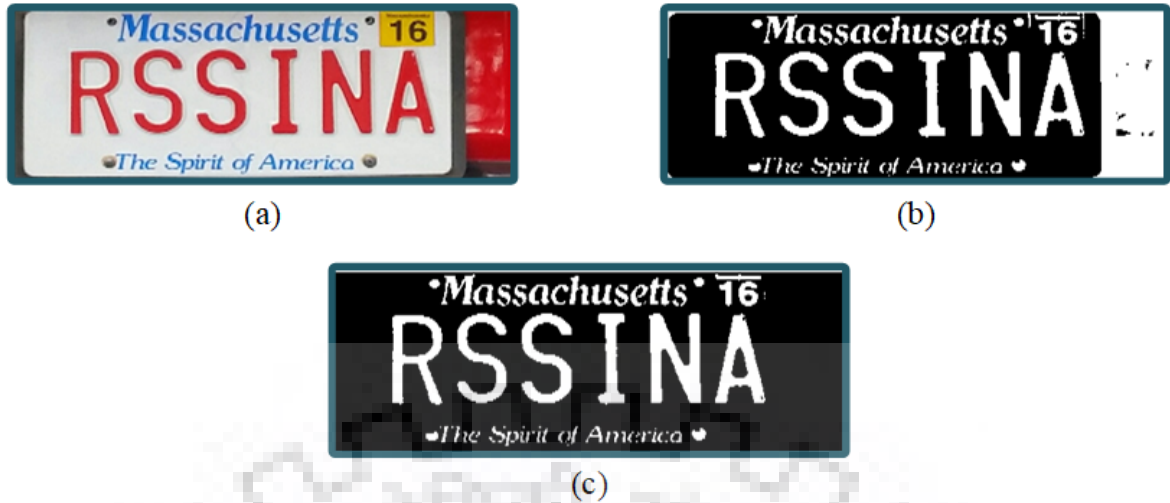


Fig. 3.16. Output of Border thresholding. (a) Detected license plate image. (b) Correctly binarized image using adaptive module. (c) Border thresholded image.

3.3.4 Removal of Objects of Low Pixel Value

There are cases in which after the binarizing and border thresholding there are still objects on the the detected license plate area that do not belong to the required character regions. To eliminate such unwanted areas and converge purely on the required character regions we use this module. For removal we define a pixel limiting value L_{pix} . After exhaustive experimentation the limiting value for our particular model i defined as $L_{pix} \leq 200$. Thus by applying CCL and using a simple function all objects in the border thresholded 500×1000 re-sized images which meeting the condition $L_{pix} \leq 200$ will be eliminated, providing a more refined and accurate information to segment. The output of this module is given in fig 3.16.



Fig. 3.17. Removal of Objects of low pixel value. (a) Detected license plate image. (b) Correctly binarized image using adaptive module. (c) Border thresholded image. (d) Low pixel value objects removal.

If we observe the case in fig 3.16(c), we observe that there are still several unwanted characters in the plate region above and below the main number which are unwanted. This is removed by low pixel value removal module as shown in fig 3.16(d).

The binarized image which is now prepared for segmentation is then again applied with CCL and by using an area threshold, which means recognizing objects which lie above a particular pixel value are segmented. An area threshold > 200 is taken in our case to obtain the individual characters. Thus obtaining the segmented characters. These character regions are then supplied to the fine tuned Tesseract OCR module to get the desired character recognition result.



Chapter 4

Results, Comparative Analysis and Experimental Observations

This chapter brings out the simulation results, analysis and experimental observations with respect to the proposed hybridized LPR model. To obtain the required result data, all the algorithms were tested with a standard data-set of 152 images. This data-set consists of a mix of 64 low resolution and 88 high resolution images. The images were also selected keeping in mind variable plate formats, with different colours and dimensions. Care was also taken to have a generous mix of license plates of various countries and regions. Apart from this, to check the robustness of the algorithm, various environmental and practical scenarios, to include, lighting conditions, multiple vehicles, complex background containing multiple objects that can deliver higher edge information, variable illumination, variable angles, variable distances etc. were also included in the data-set. The software simulation was carried out in MATLAB for all the algorithms.

In the following sections and sub-sections, initially the qualitative results will for both the hybrid LPD and adaptive CS modules will be discussed. Thereafter, quantitative results in terms of computation time and accuracy will be brought out. This will be followed by a comparative analysis with the state of art techniques in existence (mentioned in chapter 2). Finally, certain peculiar failure cases will also be brought out which serve as a bed rock for future work and research.

4.1 Qualitative Results for Hybrid LPD Module

This section brings out the various qualitative results obtained from Hybrid LPD module in the form of intermediate images and detected license plates. Emphasis has been given to depict the robustness and accuracy of the module through the results brought out. The various results obtained from the intermediate modules for a variety of input samples of both high and low resolution images are given in two parts in fig 4.1 and fig 4.2 respectively.

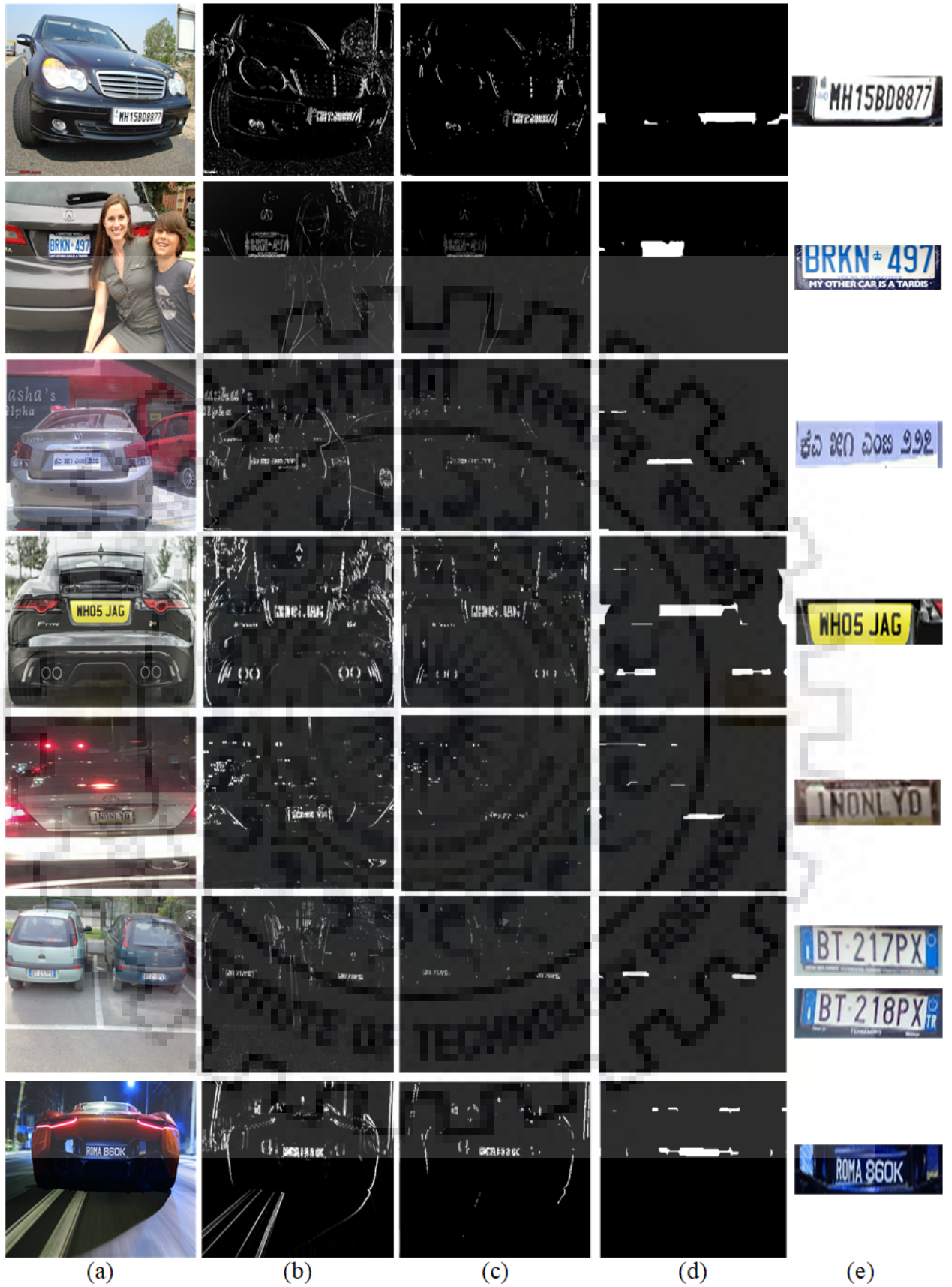


Fig. 4.1. Qualitative Results (Part 1). (a) Input images. (b) Results of Edge Detection. (c) Results of AT combined with RDRF. (d) Results of REF. (e) Detected License plates.

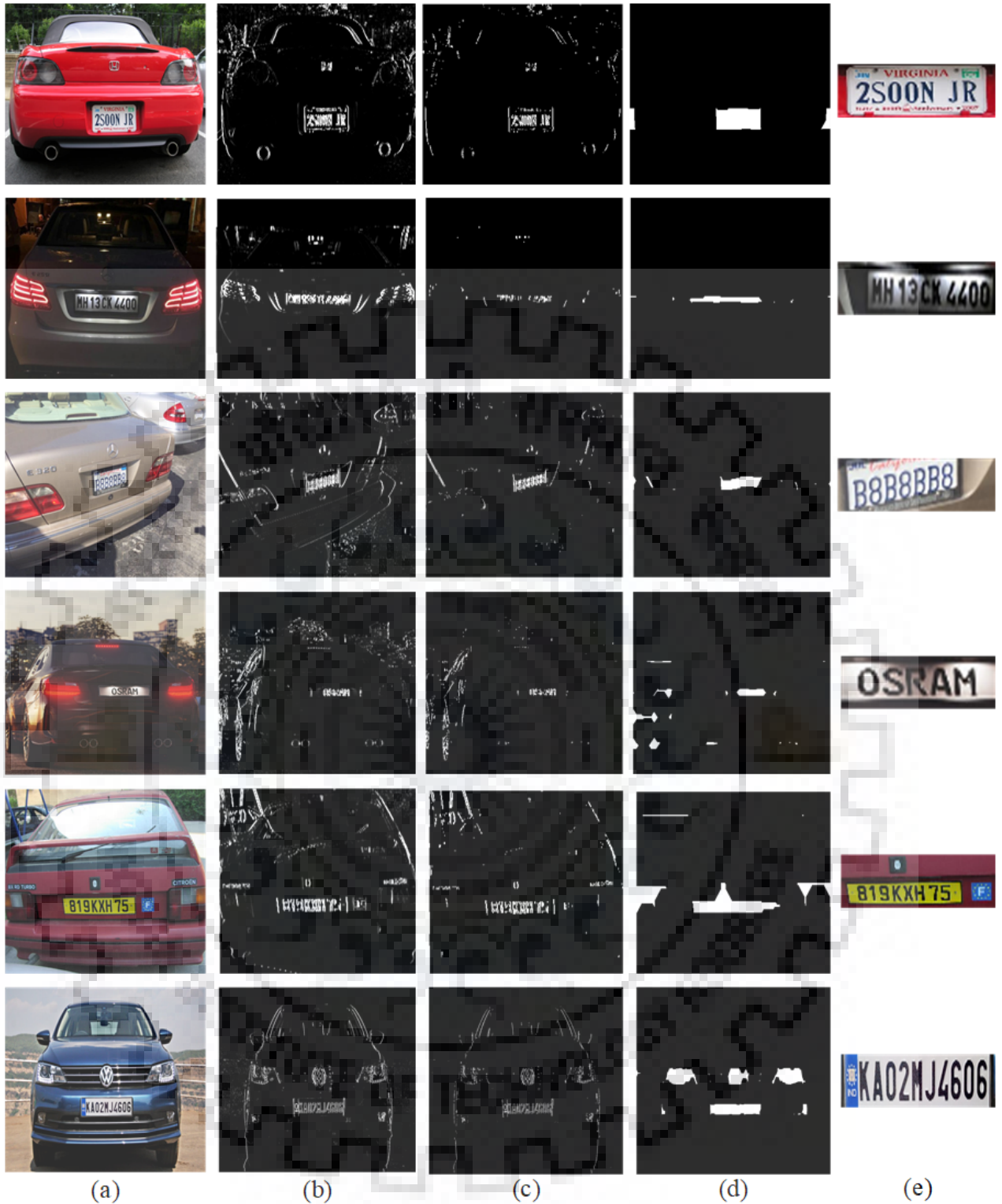


Fig. 4.2. Qualitative Results (Part 2). (a) Input images. (b) Results of Edge Detection. (c) Results of AT combined with RDRF. (d) Results of REF. (e) Detected License plates.

The input samples used for obtaining the results as shown above, comprise of images containing a variety of scenarios to include, complex environments contributing to a lot of edge information, non-uniform illumination, dim lighting conditions, multiple vehicles in the same frame, variable viewing angles, variable plate formats etc. This depicts the

robustness and accuracy of the proposed hybrid LPD model.

4.2 Qualitative Results for Adaptive CS Module



Fig. 4.3. Visual Results of Adaptive CS Module. (a) Detected license plate images. (b) Binarization using Otsu thresholding and adaptive module. (c) Result of Border thresholding. (d) Final segmented characters after removal of low pixel objects.

The various qualitative results with respect to the adaptive CS module have been brought out in this section. The stage wise results for the proposed module is given in fig 4.3 above. The input samples have been taken from the already detected license plate images obtained from the previous LPD module. Again the input samples displayed in fig 4.3, consisting of the detected license plate images, have been chosen keeping in mind the performance of the algorithm in variable backgrounds, font styles, colours, illumination, variable license plates and character dimensions etc. The intermediate results of the various stages of the adaptive CS module displayed through fig 4.3, brings out the robustness, accuracy and flexibility of the proposed CS module.

4.3 Quantitative Results of Hybrid LPD Module

In this section we bring out the quantitative results with respect to the Hybrid LPD module. The results are broadly divided into two aspects i.e., *computation time* and *accuracy*. For the purpose of comparative analysis and the two state of art methods mentioned in chapter 2 have been simulated and tested for the standard data sets along with the proposed hybrid LPD module and accordingly results have been drawn out.

4.3.1 Computation Time Results and Analysis

For the purpose of deriving results with regards to computation time, the three algorithms (VEDA, Extended Sobel and proposed) were exhaustively tested for 152 samples containing both high and low resolution images. The computation time testing was done for each each stage for each of the samples. Accordingly average stage wise computation time for both high and low resolution has been calculated. The details of average stage wise computation time analysis has been given in table 4.1.

Table 4.1: Stage Wise Breakdown of Average Computation Time (in secs)

Stages	High resolution	Low resolution	Entire Dataset
Decision Module	0.00795753	0.005755	0.00703037
Downsampling	0.0013934	0	0.000806688
Gray image Conversion	0.00201	0.0006003	0.001552095
Edge Detection	0.0048626	0.22882	0.099161751
AT	0.0107	0.417	0.181883
RDRF	0.08153	0.14592	0.108639
REF	0.01636	0.00818	0.0129176
Candidate Filter	0.079995	0.0709	0.07616626
Verification	0.33347	0.2527	0.2994763
Total Average Time	0.5385	1.13023	0.7876339

The visualization of the above given data in table 4.1 is represented in the graph given in fig 4.4.

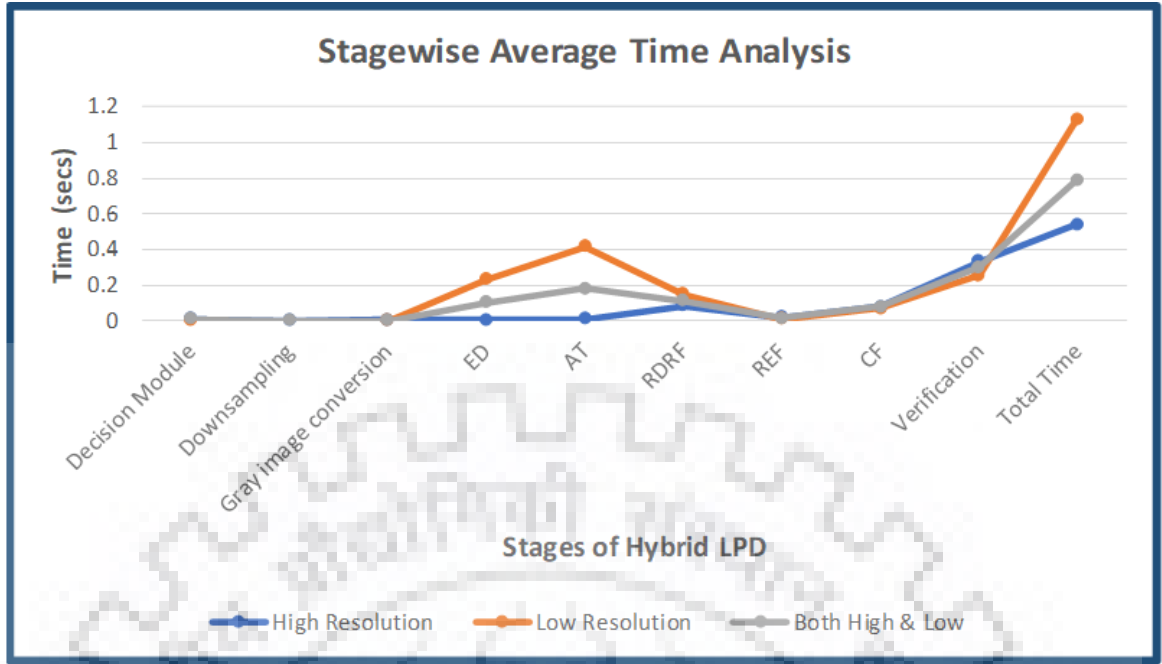


Fig. 4.4. Visualization of stage wise computation time analysis for Hybrid LPD Module.

From the graph it can be observed that the edge detection operator in case of low resolution images is taking more time as compared to high resolution images. Also the qualitative results given out by the edge detection is in turn increasing the time taken by the AT process. This is one grey area which can be part of future scope. However, when we take the entire dataset comprising of both high and low resolution images the net average time of 0.7876339 secs which is comparative quite competitive when we compare the VEDA, Extended Sobel and Hybrid LPD with each other. The overall average computation time for the three algorithms to bring out a comparison for average computation time efficiency is given in table 4.2.

Table 4.2: Comparative Analysis for Overall Average Computation Time (in secs)

LPD Algorithm	Average Computation Time (for entire dataset)
VEDA	2.0283
Extended Sobel	0.4381504941
Hybrid	0.78763397

It is observed from the above result that the computation time performance of Hybrid LPD lies somewhere in between the other two state of art techniques. Though the performance of VEDA drops due to its bad processing time results in case of high resolution images, the Hybrid LPD model makes up for this flaw by taking the best of the detection techniques from the extended Sobel algorithm and hence is able to mitigate the extra processing time required to process high resolution images.

Another interesting way to compare the computation time of the three techniques is

by concentrating on the common processes used in all three algorithms. This will bring a more clear picture of the processing efficiency of the three techniques. Table 4.3 gives the computation time analysis of the common processes executed in all three algorithms.

Table 4.3: Comparative Analysis for Overall Average Computation Time for Common Processes (in secs)

Common Processes	VEDA	Extended Sobel	Hybrid
Grayscale conversion	0.051074	0.005926	0.001552095
Edge Detection	0.23099	0.003534	0.099161751
Adaptive Thresholding	1.53022	0.00807139	0.181883
Plate Detection	0.18677	0.075737	0.07616626

The visualization in terms of a graph of depicting the data linked to computation time results for common processes is given in fig 4.5 below.

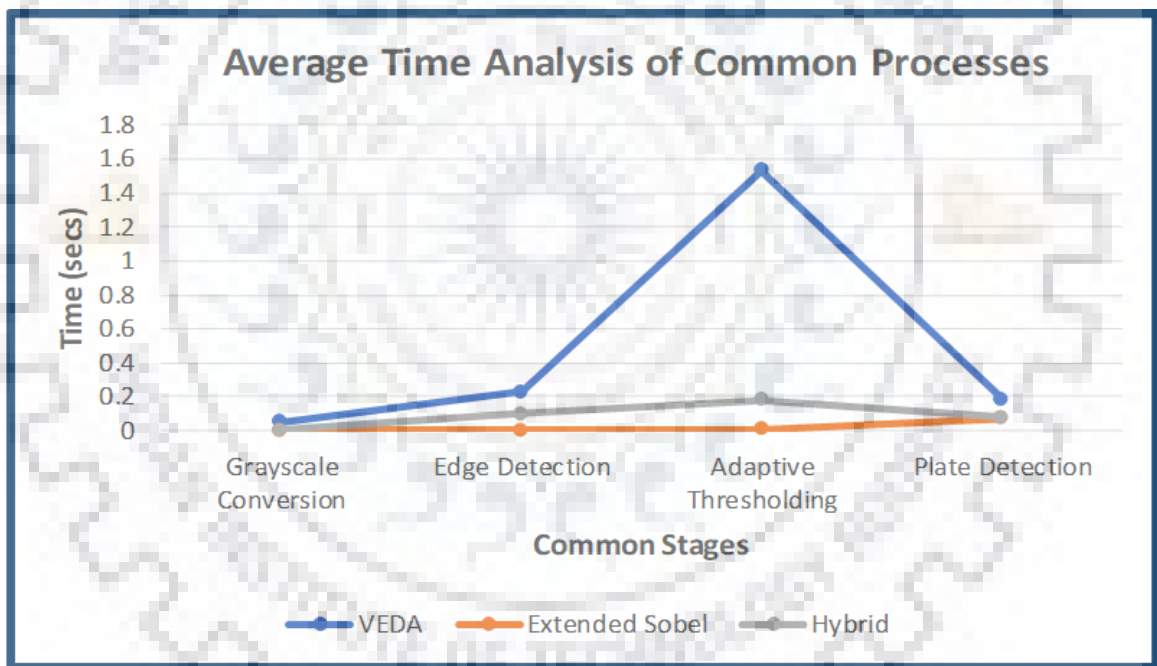


Fig. 4.5. Visualization of comparative computation time analysis for common processes.

The above figure brings out that adaptive thresholding in case of VEDA contributed maximum to the computation time when it came to common processes. To mitigate this in the Hybrid LPD algorithm, AT was applied after the downscaling and edge detection processes. Thus instead of AT being applied to the entire original input image, it was applied on the edge detected downscaled image containing less details and hence contributed towards less computational time.

A point to note in the entire computation time analysis mentioned above is that, the entire simulations have been carried out in MATLAB and hence the actual real time

analysis in languages like C++ will reveal a much less computation time for the algorithm.

4.3.2 Results and Analysis of Detection Accuracy

In this subsection we bring out the results and observations linked to detection rates and overall accuracy of the proposed algorithm. The detection rate or the rate of successful detection was analyzed individually for the high and low resolution images in order to assist in the overall conceptualizing and designing of the hybrid LPD module. The data set used, as mentioned above, is kept as homogeneous as possible to test the true aspects of flexibility while checking detection accuracy. The results of percentage detection rates obtained for various LPD algorithms is given below in table 4.4.

Table 4.4: Comparative Analysis Depicting Detection Rate in Percentages

Detection Rates(%)	VEDA	Extended Sobel	Hybrid
High Resolution	17%	94.31%	95.45%
Low Resolution	92.1875%	20.3%	93.75%
Overall Rate	48.684%	64.47%	94.736%

For a more visual depiction, fig 4.6 gives the detection accuracy in terms of number of correct detections out of a total sample space of 152.

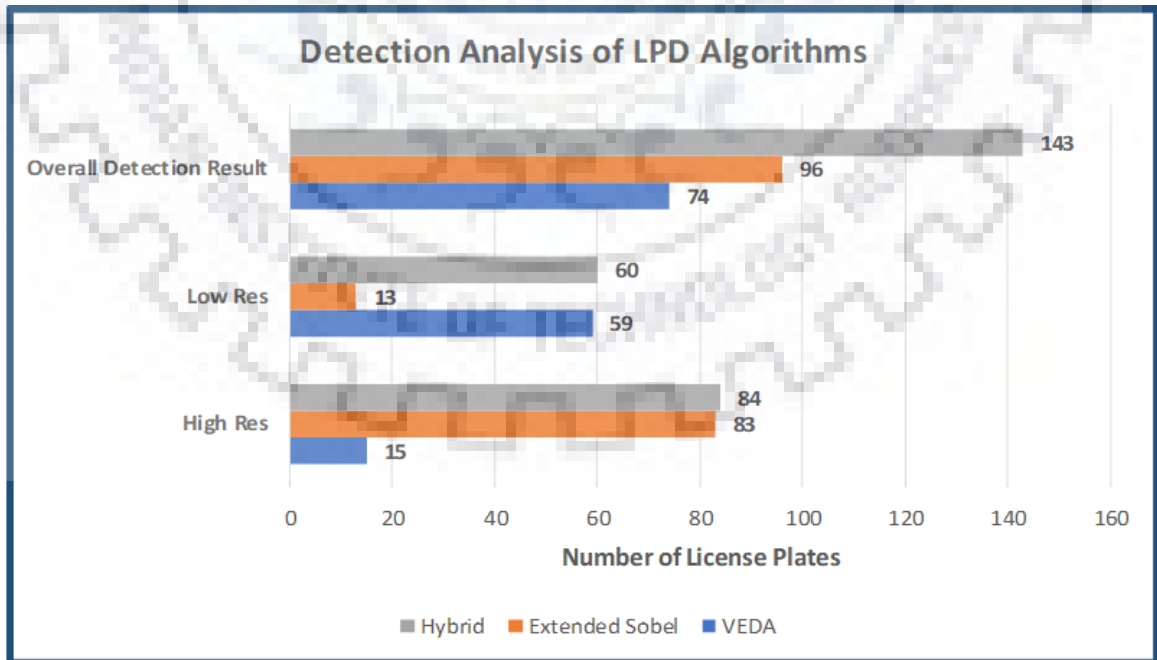


Fig. 4.6. Graphical depiction of Detection Accuracy.

The graph brought out in fig 4.6 above adequately brings forward the success of the

hybridized approach in terms of accuracy of detection. Over and above this, combining the strong points of both VEDA and extended Sobel, we have been able to achieve a high level of flexibility surpassing the state of art techniques in focus.

4.4 Quantitative Results of Adaptive CS Module

To analyze the performance of the adaptive CS module quantitatively, the results obtained from exhaustive experimentation are divided again into two broad categories, i.e., the *segmentation accuracy* and the *processing time*. To obtain the comparative analysis, the MSER based CS model as mentioned in chapter 2 was implemented and exhaustive tested for the standard dataset containing 152 (88 high resolution and 64 low resolution) images. A total 1081 characters were tested out of which 626 were from high resolution detected plate images and 455 were from low resolution detected plate images. A similar exercise was carried out for the proposed adaptive model. Accordingly segmentation, recognition and processing time results were obtained for both the algorithms. As a final step Tesseract was independently tested, without any segmentation module preceded by it and tested for recognition accuracy and processing time. This was done to get a clear picture of the effect of segmentation on the recognition module and subsequently on the overall performance of the LPR algorithm. The detection output supplied to all the three algorithms is obtained by the hybrid LPD module to get a clear picture of the combined working.

4.4.1 Results and Analysis of Segmentation Accuracy

In this subsection we bring out individually the performance in terms segmentation accuracy of the three algorithms, i.e., MSER, adaptive CS, and Tesseract. Thereafter a combined analysis of the three techniques is carried out. The segmentation performance of MSER and adaptive CS modules in terms of number of characters segmented and recognized is given in the form of bar graphs in fig 4.7 and 4.8 respectively. The graphs depict the performance of both the algorithms first for low resolution and then high resolution detected license plate images, in terms of number of character segmented and recognized. It also brings out the overall segmentation result in terms total number of characters segmented and recognized.

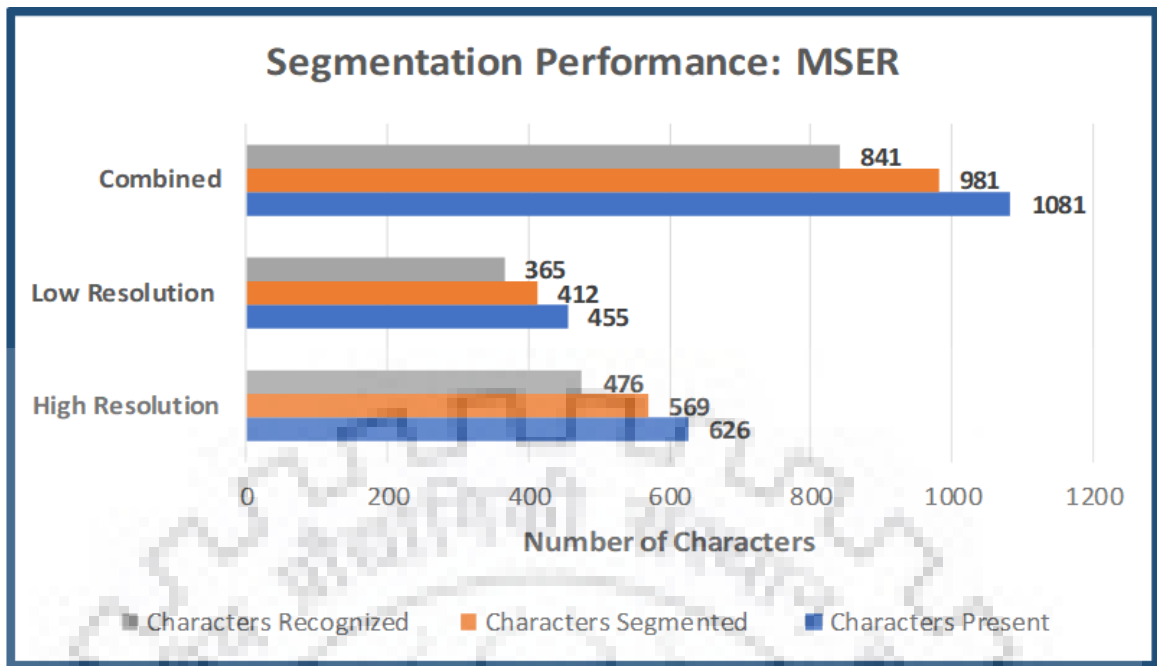


Fig. 4.7. Segmentation performance of MSER.

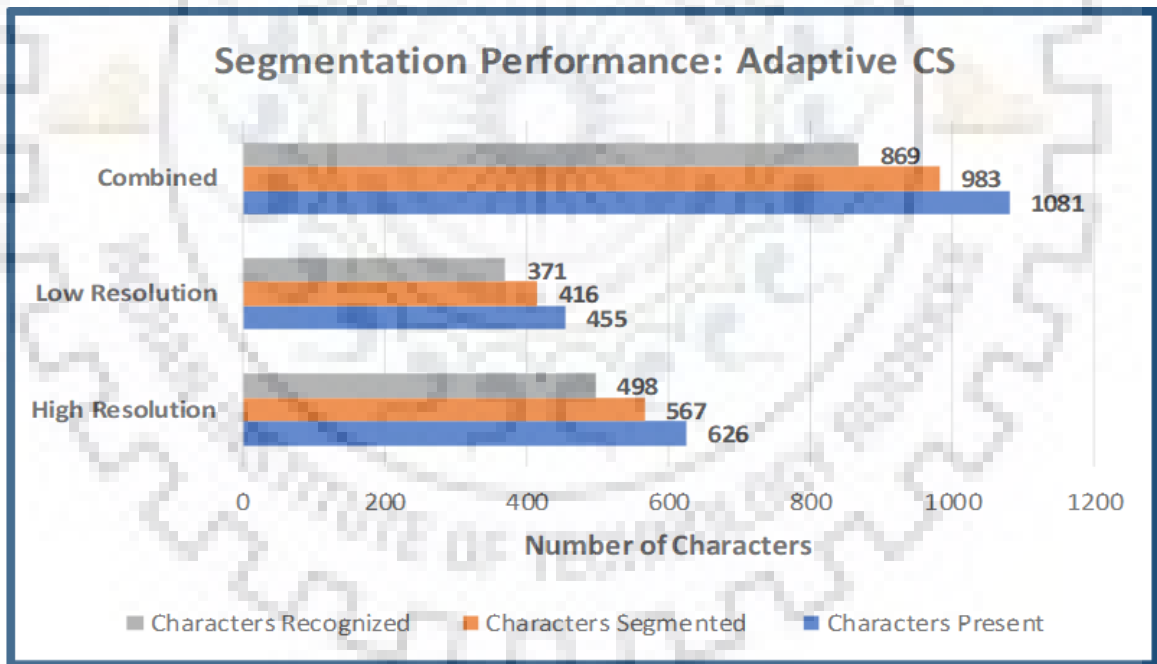


Fig. 4.8. Segmentation performance of Adaptive CS module.

The recognition performance of the OCR based Tesseract algorithm is given in graphical form in fig 4.9. The graph clearly brings out the massive dip in performance in recognition without a CS module clearly indicating the importance of a CS module in a LPR system and its role in augmenting the accuracy of recognition process.

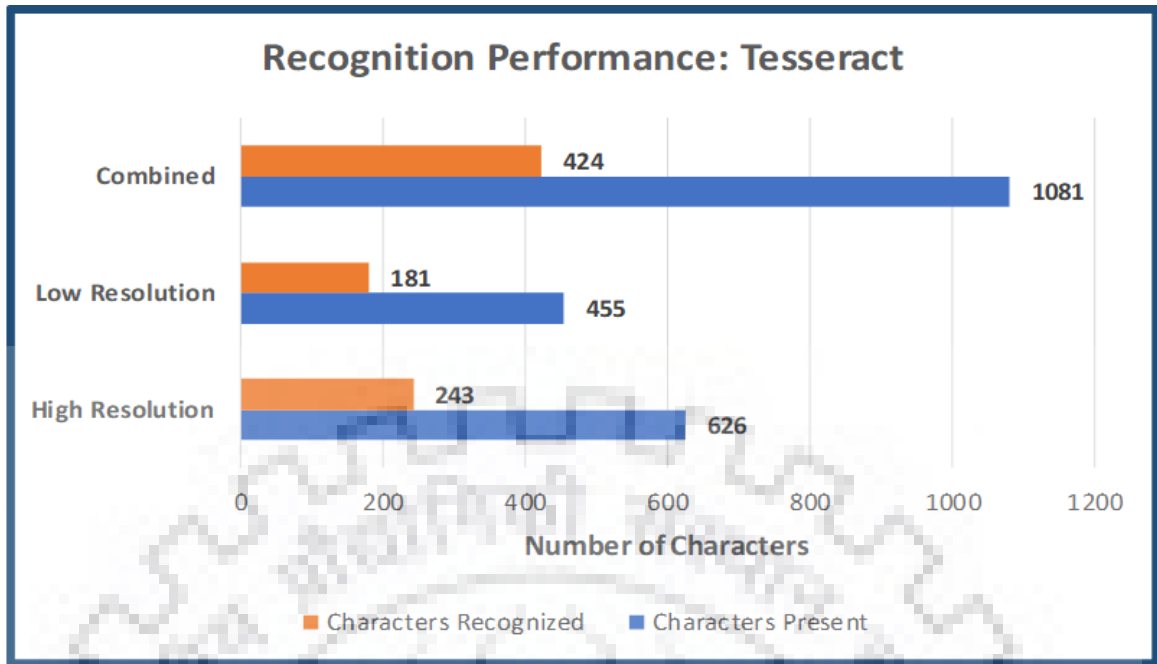


Fig. 4.9. Recognition performance of Tesseract.

Now that the algorithms have been individually analyzed, the comparative analysis in terms of segmentation and recognition success rates is given in percentage form in table 4.5.

Table 4.5: Comparative Analysis of Accuracy in Terms of Percentage Success Rates.

Success Rates(%)	MSER	Adaptive CS	Tesseract
Segmentation (low res)	90.5%	91.4%	-
Recognition (low res)[wrt segmentation]	88.59%	89.18%	-
Recognition (low res)[wrt total char present]	80.219%	81.53%	39.78%
Segmentation (high res)	90.89%	90.575%	-
Recognition (high res)[wrt segmentation]	83.65%	87.83%	-
Recognition (high res)[wrt total char present]	76.038%	79.552%	38.81%
Overall Segmentation	90.74%	90.93%	-
Overall Recognition [wrt segmentation]	85.72%	88.402%	-
Overall Recognition [wrt total char present]	77.79%	80.38%	39.22%

In the above table it has been adequately brought out that not only has the adaptive CS module matched the state of art MSER technique in terms of segmentation accuracy, but has also outperformed the MSER in terms of augmenting the accuracy of recognition of Tesseract. In other words the adaptive CS module has provided better qualitative results to Tesseract to act upon and carry out a more accurate recognition. It has also been brought out that both the CS modules (MSER and adaptive CS) have boosted the accuracy of Tesseract in a big way, thereby proving the fact that for successful recog-

dition, accurate and robust segmentation is a must. The graphical visualization of the performance of the three techniques is given in fig 4.10.

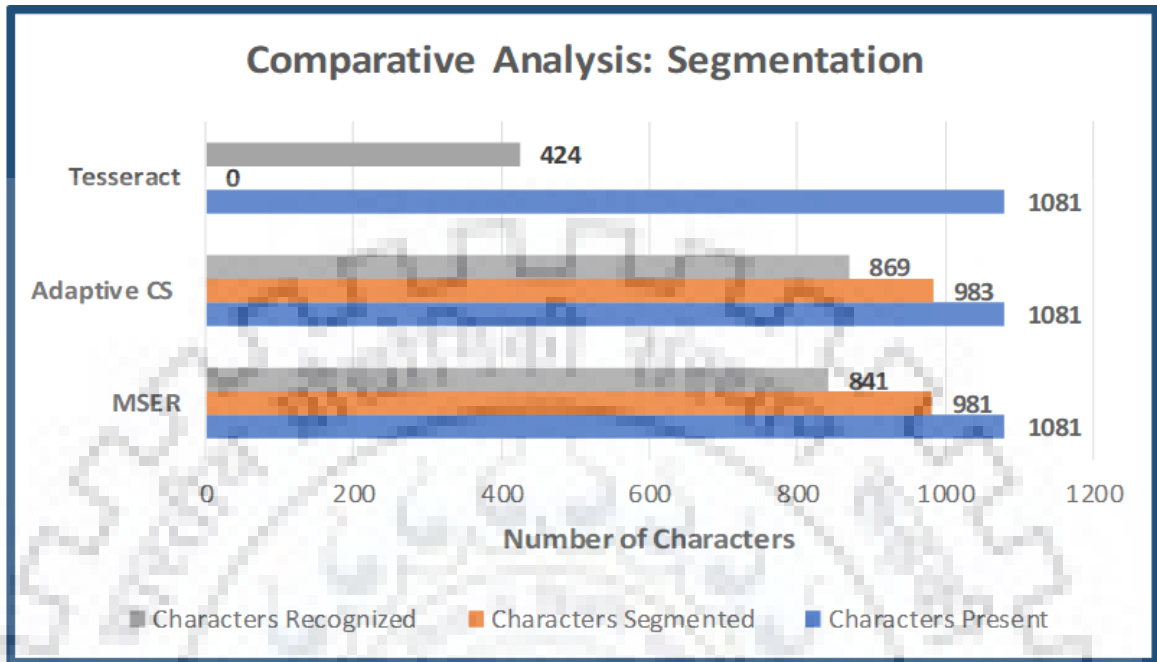


Fig. 4.10. Comparative analysis of segmentation and recognition performance.

From the above figure it is worthy to note that due to the adaptive CS module, the CR module i.e. Tesseract has recognized 28 characters more as compared to when MSER is applied. Also, adaptive CS module has substantially increased the capability of Tesseract by more than two times than its original result of 424 characters out of a total of 1081.

4.4.2 Results and Analysis of Processing Time

In this sub-section we bring out the results and comparative analysis with respect to the processing time performance of the three algorithms in focus i.e., MSER, adaptive CS and Tesseract. The processing time analysis for the three algorithms is given in table 4.6 below.

Table 4.6: Comparative Analysis of Average Processing Times (in secs).

Average Processing Time(secs)	MSER	Adaptive CS	Tesseract
Segmentation (low res)	1.1625	0.54857	-
Recognition (low res)	0.05793	0.067997	0.06858
Segmentation (high res)	1.19968	0.584901	-
Recognition (high res)	0.0702723	0.067247	0.07852
Overall Segmentation	1.184	0.5696	-
Overall Recognition	0.065074	0.067563	0.074334

From the results brought out in table 4.6 above, it comes to light that processing time efficiency of the proposed CS algorithm is much higher than that of the state of art MSER algorithm. The overall segmentation time taken by the adaptive CS algorithm is 0.5696 secs as compared to 1.184 secs of MSER. In a nutshell the processing time of adaptive CS is literally half of that of MSER. Over and above this the adaptive CS algorithm has given matching results in terms of augmenting the avg processing time of the subsequent CR module. This capability demonstrated by the adaptive CS module further makes it a more suitable and obvious choice for the enhancing the real time detection and recognition capability of the overall LPR system.

4.5 Analysis of Failure Cases: Hybrid LPD Module

In this section we briefly discuss certain cases of failed detection of the hybrid LPD module. These cases are important to acknowledge since they will form the basis of future research and work in this direction. Some interesting cases of failure of the hybrid LPD module are given in fig 4.11 above.

In the first first row of fig 4.11, we have brought out a case of of a hazy image. It is interesting to note that though the edge detection delivers some amount of edges, the subsequent stages totally fail to deliver an output, ultimately resulting in total failure. Thus it is necessary to carry out a de-hazing operation in such a scenario.

The result demonstrated in the second row of fig 4.11, displays the image of a car with the front license plate focus light of a very high brightness level, thereby totally overpowering the structure of the license plate region. Though the edge detection is able to deliver some edge information, it is not able to bring out the pixel density and edges of the required license plate region in such a case. Subsequently the AT and RDRF stage completely removes the required target region. This is followed by the failure of the REF and finally candidate filter detects a wrong image region.

The third and fourth row of fig 4.11 brings out the cases of glare caused by the headlights of the vehicles such that it eclipses the license plate region. The result is again catastrophic with total or partial failure of the subsequent stages and subsequently total failure of detection as displayed by the results of the candidate filter.

The fifth row of fig 4.11 brings out the important and real world case of blur. Though the proposed algorithm has robustly performed in case of certain images which have a mild blur, however for certain highly blurred images the overall algorithm totally fails to converge to the required region. Hence, it is observed that robust anti-blur techniques can be incorporated in the hybrid LPD algorithm, without compromising on the computation time efficiency and can form part of future work.



Fig. 4.11. Some typical failure cases of Hybrid LPD module. (a) Original input image. (b) Edge detection. (c) AT with RDRF. (d) REF output. (e) Detected region using Candidate filter.

4.6 Analysis of Failure Cases: Adaptive CS Module

This section brings out the typical cases of failure in case of adaptive CS module. The failure cases serve as the basis to counter the future challenges with respect to accuracy and robustness of this module. Some of the intriguing failure case of adaptive CS module

is given in fig 4.12.

If we take into consideration the first row of fig 4.12, we see that we are inputting into the adaptive CS module a correctly detected license plate. However if we observe carefully, the bounding box for the detected region is touching the upper edge of the license plate characters. Because of this, though we get a very good thresholding result, the border thresholding module completely fails as the characters touching the considered border and are removed as unwanted objects. Thus this indicates that the LPD module preceding the adaptive CS module has to be highly accurate.

Considering the input in the second row we see the effect of non-uniform illumination on the accuracy of segmentation. We see that due to the non uniform illumination, the thresholding and as a result the subsequent stages completely fail. Thus we can modify our adaptive technique to become more localized in terms of thresholding in order to generate better results in non-uniform cases. This can also form part of future work.



Fig. 4.12. Some typical failure cases of Adaptive CS module. (a) Detected license plate image. (b) Otsu thresholding using Adaptive module. (c) Border thresholding. (d) Final segmented image after removal of low pixel value objects.

The results brought out in the third column of fig 4.12 brings out yet another interesting practical observation. If we observe the detected license plate image carefully,

we see that there are two rivets inserted in between the characters of the license plate. These rivets due to their statistical size and texture are detected as part of the character set by the thresholding carried out by the adaptive module. As a result the Tesseract module recognizes as character 'O' and hence an error in overall recognition is caused. Thus more the rivets, more the recognition error. Over and above this, in this particular input, we again see the partial failure of border thresholding since again the character edges of two characters touch the upper boundary of the detection bounding box and are again removed as unwanted objects.

The results displayed in the fourth and fifth row of fig 4.12 brings out the importance of accurate detection of the area of the license plate. As we can see that the detected license plate image not only contains the license plate region but also the area containing the body of the vehicle. This results in a multi-modal histogram and hence results in failure of accurate thresholding and subsequently failure of border thresholding and final segmentation.

All these observations brought out as part of partial or complete failures can form a foundation for future work in this direction and designing of of a more result adaptive algorithm. The failure results also bring out the fact that the accuracy and success rate of the adaptive CS module depends heavily on the accuracy of the hybrid LPD module.



Chapter 5

Conclusions and Future Work

5.1 Conclusions

License plate recognition systems are poised to play an instrumental role in many modern applications, specially related to security and real time intelligence gathering both at national and international levels. The state of art LPD and CS methods discussed in the initial part of this dissertation have made it adequately clear that run time efficiency and accuracy are two major areas of research and refinement to make this technology a potent tool for real world practical scenarios of the future. However, it has also emerged that the imperative aspect of flexibility for seamless cross platform integration is missing in the techniques in existence. To counter this, through this dissertation we have introduced a novel hybridized approach to flexible, robust and accurate LPD augmented by an accurate, fast and robust adaptive CS module.

Utilizing a comprehensive data set of 152 (88 high resolution and 64 low resolution) images, the performance figures and qualitative results brought out in this dissertation clearly indicate the utility and effectiveness of both these algorithms in diverse scenarios of LPR. The hybrid LPD module proposed gives an impressive average computation time performance of 0.786 secs which is comparable and in some cases better than the state of art techniques in existence. This computation time efficiency is clubbed with a impressive overall accuracy of 94.763%. This combination along with the inherent flexibility it possesses makes the hybrid LPD a very potent algorithm in the domain of license plate recognition.

Coming to the adaptive CS module, we have already observed its overall segmentation accuracy of 90.93% which matches the accuracy of state of art techniques like MSER and Watershed. However, apart from this the adaptive CS module has been able to outperform the MSER algorithm in terms of boosting the accuracy of the corresponding Tesseract module in terms of character recognition. In terms of computation time it again outperforms the MSER algorithm by reducing the overall processing time by half of that of MSER. This makes the adaptive CS algorithm an extremely effective tool for

character segmentation tasks specific to LPR systems.

Thus to holistically conclude we can safely say that the techniques introduced in this dissertation can play an instrumental role in providing true cross platform integration combined with accuracy and real time efficiency. However, at the same time there is still some scope of improvement in both these algorithms which has been brought out in the next section.

5.2 Future Work

As clearly brought out by analyzing some of the key failure cases of both the algorithms, future work for improving this model further will include exhaustive testing and refinement utilizing a variety of data sets under various conditions. Detail comparative observations have to be also obtained for in depth analysis of the performance of the techniques proposed to enable their successful implementation in real world applications.

Considering the case of Hybrid LPD algorithm, there is a requirement to further increase the robustness of the algorithm by improving its performance in environments possessing haziness, high degree of glare, non-uniform illumination, and high degree of blur. At the same time the real challenge lies in keeping computation time efficiency and flexibility to the maximum, in order to utilize it for real time applications.

Coming to Adaptive CS module, the primary area of future work will be to make it less dependent on the accuracy of its preceding LPD module and to make it more robust for challenging environments and scenarios like non-uniform illumination, haze, variable size of characters and non-character objects like rivets present in license plates. However, to preserve the purpose for which this technique has been designed for, the processing time efficiency should not be compromised.

Bibliography

- [1] R.C. Gonzalez and R.E. Woods, “Digital Image Processing -Third Edition”, *Pearson*, 2009.
- [2] A.K. Jain, “Fundamentals of Digital Image Processing”, *Prentice Hall of India*, 2002.
- [3] R.C. Gonzalez, R.E. Woods and S.L. Eddins, “Digital Image Processing using MATLAB”, *Pearson Education*, 2004.
- [4] S.-Z. Wang and H.-J. Lee, “Detection and recognition of license plate characters with different appearances”, in *Proc. IEEE Conf. Intell. Transp. Syst. (ITSC)*, vol. 2, Oct. 2003, pp. 979–984.
- [5] V. Shapiro, G. Gluhchev, and D. Dimov, “Towards a multinational car license plate recognition system”, *Mach. Vis. Appl.*, vol. 17, no. 3, pp. 173–183, 2006.
- [6] D. Zheng, Y. Zhao, and J. Wang, “An efficient method of license plate location”, *Pattern Recognition. Lett.*, vol. 26, no. 15, pp. 2431–2438, 2005.
- [7] W. Jia, H. Zhang, and X. He, “Region-based license plate detection”, *J. Netw. Computer. Appl.*, vol. 30, no. 4, pp. 1324–1333, 2007.
- [8] Y. Zhao, Y. L. Yuan, S. B. Bai, K. Liu, and W. Fang, “Voting-based license plate location”, in *Proc. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2011, pp. 314–317.
- [9] M. Donoser, C. Arth, and H. Bischof, “Detecting, tracking and recognizing license plates”, in *Proc. Aisian Conf. Computer. Vis. (ACCV)*, 2007, pp. 447–456.
- [10] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide-baseline stereo from maximally stable extremal regions”, *Image Vis. Computer.*, vol. 22, no. 10, pp. 761–767, 2004.
- [11] R. M. Haralick, S. R. Sternberg, and X. Zhuang, “Image analysis using mathematical morphology”, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-9, no. 4, pp. 532–550, Jul. 1987.
- [12] E. R. Lee, P. K. Kim, and H. J. Kim, “Automatic recognition of a car license plate using color image processing”, in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Nov. 1994, pp. 301–305.

- [13] K. I. Kim, K. Jung, and J. H. Kim, "Color texture based object detection: An application to license plate localization", in *Pattern Recognition with Support Vector Machines (Lecture Notes in Computer Science)*, vol. 2388. Berlin, Germany: Springer-Verlag, 2008, pp. 293–309.
- [14] J. Tian, R. Wang, G. Wang, and F. Yang, "A new algorithm for license plate localization in open environment using color pair and stroke width features of character", *Proc. SPIE*, vol. 8921, pp. 909–927, Oct. 2013.
- [15] Youngwoo Yoon, Kyu-Dae Ban, Hosub Yoon, and Jaehong Kim, "Blob Extraction based Character Segmentation Method for Automatic License Plate Recognition System" *IEEE*, 978-1-4577-06530/11/\$26.00 , 2011.
- [16] Khalid Maglad, Dzulkifli Mohamad and Nureddin A. Abulgasem, "Saudian Car License Plate Number Detection and Recognition Using Morphological Operation and RBF Neural Network", 2011
- [17] Shuang Qiaol, Yan Zhul , Xiufen Li , Taihui Liu and Baoxue Zhangl, "Research of improving the accuracy of license plate character segmentation", *Fifth International Conference on Frontier of Computer Science and Technology*, 2010.
- [18] Yungang Zhang and Changshui Zhang, "A New Algorithm for Character Segmentation of License Plate", *IEEE* 0-7803-7848-2/03/\$17.00 0, 2003
- [19] Anuja P. Nagare, "License Plate Character Recognition System using Neural Network", *International Journal of Computer Applications*, (0975 – 8887) Volume 25–No.10, July 2011.
- [20] Satadal Saha, Subhadip Basu, Mita Nasipuri and Dipak Kr. Basu, "Localization of License Plates from Indian Vehicle Images Using Iterative Edge Map Generation Technique", *Journal Of Computing*, Volume 3, Issue 6, June 2011, Issn 2151-9617, 2010.
- [21] C. Nelson Kennady Babu, Siva Subramanian T and Kumar Parasuraman Member, IEEE, "A Feature Based Approach for License Plate-Recognition of Indian Number Plates" *IEEE International Conference on Computational Intelligence and Computing Research*, 2010.
- [22] Seyed Hamidreza Mohades Kasaei and Seyed Mohammadreza Mohades Kasaei, "Extraction and Recognition of the Vehicle License Plate For Passing under Outside Environment", *European Intelligence and Security Informatics Conference*, 2011.
- [23] Jing-Ming Guo, Senior Member, IEEE, Yun-Fu Liu, Student Member, IEEE, and Chih-Hsien Hsia, Member, IEEE, "Multiple License Plates Recognition System",

International Conference on System Science and Engineering, 30 June - 2 July, 2012.

- [24] Deng Hongyao and Song Xiuli, “License Plate Characters Segmentation Using Projection and Template Matching”, *International Conference on Information Technology and Computer Science*, 2009.
- [25] Serkan Ozbay, and Ergun Ercelebi, “Automatic Vehicle Identification by Plate Recognition”, *World Academy of Science, Engineering and Technology - 9*, 2007.
- [26] Fatih Kahraman, Binnur Kurt, and Muhittin Gökmen, “License Plate Character Segmentation Based on the Gabor Transform and Vector Quantization”, *A. Yazici and C. Sener (Eds.): ISCIS*, 2003, Incs 2869, pp. 381388, 2003.
- [27] D. Bradley and G. Roth, “Adaptive thresholding using the integral image,” *J. Graph., GPU, Game Tools*, vol. 12, no. 2, pp. 13–21, 2007.
- [28] A. M. Al-Ghaili, S. Mashohor, A. R. Ramli, and A. Ismail, “Vertical edge-based car-license-plate detection method,” *IEEE Trans. Veh. Technol.*, vol. 62, no. 1, pp. 26–38, Jan. 2013.
- [29] 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*, vol. 26, no. 3, march 2017.
- [30] J. Matas, O. Chum, M. Urba and T. Pajdla, “Robust wide baseline stereo from maximally stable extremal regions,” in *Proc. of British Machine Vision Conference*, 2002, pp. 384-396.
- [31] Neumann, Lukas, and Jiri Matas. ”Real-time scene text localization and recognition.” *Computer Vision and Pattern Recognition (CVPR)*, 2012 IEEE Conference on. IEEE, 2012.
- [32] Gonzalez, Alvaro, et al. ”Text location in complex images.” *Pattern Recognition (ICPR)*, 2012 21st International Conference on. IEEE, 2012.
- [33] Li, Yao, and Huchuan Lu. ”Scene text detection via stroke width.” *Pattern Recognition (ICPR)*, 2012 21st International Conference on. IEEE, 2012.
- [34] Zhong Qu and Li Hang ”Research on Iimage Segmentation Based on the Improved Otsu Algorithm.”, 2010
- [35] 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, Nov. 2015.

- [36] M.-M. Cheng, G. X. Zhang, N. J. Mitra, X. Huang, and S. M. Hu, "Global contrast based salient region detection," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2011, pp. 409–416.
- [37] R.E. Fan, K.W. Chang, C.-J. Hsieh, X.-R. Wang, and C.J. Lin, "LIBLINEAR: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jun. 2008.
- [38] L. Vincent and L. Soille, "Watersheds in digital space: an efficient algorithm based on immersion simulations," *IEEE Transactions On Pattern Analysis and Machine Intelligence*, vol. 13, pp. 583–598, 1991.
- [39] Chao-yang and Liu Jun-hua, "Vehicle License Plate Character Segmentation Method Based on Watershed Algorithm," *International Conference on Machine Vision and Human-machine Interface*, 2010.

