

DESIGN AND DEVELOPMENT OF A LEARNING-BASED
LICENSE PLATE DETECTION ALGORITHM

HO WING TENG

MASTER OF COMPUTER SCIENCE

FACULTY OF ENGINEERING AND SCIENCE
UNIVERSITI TUNKU ABDUL RAHMAN
JUNE 2012

**DESIGN AND DEVELOPMENT OF A LEARNING-BASED LICENSE
PLATE DETECTION ALGORITHM**

By

HO WING TENG

A thesis submitted to the Department of Computer Science and
Information Systems,
Faculty of Engineering and Science,
Universiti Tunku Abdul Rahman,
In partial fulfillment of the requirements for the degree of
Master of Computer Science
June 2012

ABSTRACT

Design and Development of a Learning-Based Vehicle License Plate Detection Algorithm

Ho Wing Teng

The objective of this project is to research, design and develop a learning-based license plate detection framework with a new feature type named as Spatial Horizontal Edge Variation features. As the growing of vehicle usage, there are a lot of applications related in the Intelligent Traffic System. One of the applications that applied in vehicle is the Vehicle License Plate Recognition. There are few modules in Vehicle License Plate Recognition, and license plate detection is one of the essential modules that affect the overall performance. In this project we introduce a framework that will learn from the training samples and generates a set of classifiers to detect the license plate from the vehicle. Each of the classifiers constructs by a set of features that we named it as Spatial Horizontal Edge Variation feature. Spatial Horizontal Edge Variation feature is a simplified edge feature that can extract the edge information of the object through the horizontal orientation to differentiate between license plate and non license plate. Our framework selects a set of best features from a large set of feature set and generates a linear tree classifier to perform the object detection. From our experiments and analysis, we found that vehicle license plates have similarity with text that consist several characters. The changes of the edge information

from the left to right horizontally through the license plate can actually provide useful information to distinguish whether it is a license plate or otherwise. In our project, we did further testing on the framework for detecting text from natural scene and side view car detection. For our license plate detector, we tested with some of the Malaysian vehicle license plate and achieve recall rate of 0.93796 and precision rate of 0.55366. During our testing evaluation on our framework, we found that there are some situations that the classifiers unable to detect the license plate and analysis are done on the experiment results.

ACKNOWLEDGEMENT

First and foremost, I would like specially thank my supervisor Associate Professor Dr Tay Yong Haur for his continuous guidance and support. Throughout the whole process, he has provided me a lot of ideas, knowledge, advice, and inspiration. Besides being a good mentor, he is also showing his care like a brother, inspire and motivate me whenever there is a downtime during the research. All his efforts are truly appreciated and I would like to take this opportunity to thank him gratefully.

Besides, I would like to thanks my family, my lovely parents who always there for me, take care of me, supporting me and provides me a warm and sweet home, and believe in me for all my hard work. Then, I would like to show my appreciation to my love Elaine Yeong Pui Pui, who never give up and believing in me, supporting me, caring of me. She provides motivation and mentally support to me whenever I face any problems and obstacles either in life or during my research. Without all the loves and supports from my parents and my love, I doubt that I can come to this stage; therefore, I am greatly thankful for having them be by my side.

Next, not to forget, the helps and knowledge from my friends, seniors, colleagues, Mr Tan Xian Yi, Mr Tou Jing Yi, Mr Kenny Khoo Kuan Yew, Ms. Chan Siew Keng, Mr. Lim Hao Wooi, Mr. Richard Ng Yew Fatt, Mr. Heng Eu Sin and Mr. Lee Chee Wei from Computer Vision and Intelligent Systems (CVIS) group, and lectures as well as professors from Universiti Tunku Abdul Rahman for their efforts and guidance that lead me to the completion of the dissertation of my Master research.

Last but not least, for all my friends that I did not mentioned above as there are too many of them, thanks for the supports, helps for all these while. I really feel grateful to have them being part of my life and I would like take this opportunity to wish everyone all the best and healthy always. Cheers.

APPROVAL SHEET

This dissertation/thesis entitled “**Design and Development of a Learning-Based License Plate Detection Algorithm**” was prepared by HO WING TENG and submitted as partial fulfillment of the requirements for the degree of Master of Computer Science at Universiti Tunku Abdul Rahman.

Approved by:

(Associate Prof. Dr. TAY YONG HAUR)
Professor/Supervisor
Department of Internet Engineering
and Computer Science
Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

Date: 19 October 2011

FACULTY OF ENGINEERING AND SCIENCE
UNIVERSITI TUNKU ABDUL RAHMAN

Date: 5TH June 2012

SUBMISSION OF THESIS

It is hereby certified that Ho Wing Teng (ID No: 07UIM08453) has completed this thesis entitled “Design and Development of a Learning-based License Plate Detection Algorithm” under the supervision of Dr Tay Yong Haur (Supervisor) from the Department of Internet Engineering and Computer Science, Faculty of Engineering and Science.

I understand that University will upload softcopy of my thesis in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

(Ho Wing Teng)

DECLARATION

I hereby declare that the dissertation is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTAR or other institutions.

Name HO WING TENG

Date 05 June 2012

TABLE OF CONTENTS

	Page
ABSTRACT	i
ACKNOWLEDGEMENT	iii
APPROVAL SHEET	iv
DECLARATION	vi
TABLE OF CONTENTS	vii
LIST OF TABLES	x
LIST OF FIGURES	xiv
 CHAPTER	
 INTRODUCTION	1
1.1 Problem Statement	2
1.2 Motivation	4
1.3 Objectives	6
1.4 Scope of Works	7
1.5 Section Description	8
LITERATURE REVIEW	10
2.1 Introduction	10
2.2 Background & Previous Work	11
2.3 License Plate Detection	14
2.4 Learning Based Framework	25
2.5 Heuristic-based algorithm	28
2.6 Adaptive Boosting	32
2.6.1 Adaptive Boosting for Face Detection	32
2.6.2 Adaptive Boosting for License Plate Detection	36
2.6.3 Adaptive Boosting for Text Detection	38
2.7 Adaptive Boosting & Evolutionary Feature Search	40
2.8 Control Points Based Object Detection	41
2.9 Spatial Histogram Based Object Detection	43
2.10 Support Vector Machine for Object Detection	44
2.11 Support Vector Machine in Text Detection	45

2.13	Summary	47
SYSTEM DESIGN AND IMPLEMENTATION		50
3.1	System Frameworks & Architecture	50
3.2	Image Representative – Integral Horizontal Edge Variation Image	56
3.3	Generate Horizontal Edge Variation Image	57
3.3.1	Calculate Integral Horizontal Edge Variation Image	58
3.4	Feature Extraction	60
3.4.1	Spatial Horizontal Edge Variation	60
3.4.2	Spatial Horizontal Edge Variation Feature Value Extraction	62
3.4.2.1	Calculate Spatial Horizontal Edge Variation Feature Template	62
3.4.3	Spatial Horizontal Edge Variation Feature Normalization	64
3.4.4	Finding Feature Threshold	65
3.5	Genetic Algorithm for Evolutionary Feature Search	68
3.5.1	Applying Genetic Algorithm - Feature Searching and Selection	71
3.5.2	Hypothesis and population	71
3.5.3	Fitness function and Selection	71
3.5.4	Genetic Operators	72
3.5.4.1	Single-point crossover operator	73
3.5.5	Feature Repair Function	75
3.6	Cascading of Feature Construction	76
3.7	Support Vector Machine Classification	77
3.8	Image Sub-window Scanning with Cascaded Features	78
EXPERIMENTS SETTING AND RESULTS		89
4.1	Experiment Evaluation Function	90
4.2	Experiment Setting and Results	92
4.2.1	Experiment: Spatial Horizontal Edge Variation feature on License Plate Detection	96
4.2.2	Experiment: Spatial Horizontal Edge Variation feature on Text Detection	109
4.2.3	Experiment: Spatial Horizontal Edge Variation feature on side view car detection	117
4.3	Discussions	126
4.3.1	How numbers of Regions of Spatial Horizontal Edge Variation Feature Affect the Detection Result?	126
4.3.2	How Training Image Type Affect the Detection Result?	140
4.3.3	Cascaded Feature-thresholding Classification Vs Support Vector Machine Classification	145
4.3.4	How to Reduce False Detection?	146
4.3.5	Images that failed to be detected	148
CONCLUSION		156

LIST OF TABLES

Tables		Page
3.1	The learning algorithm for constructing cascade feature during training Stage in Object Detection using Spatial Horizontal Edge Variation Feature.	88
4.1	Definition of the parameters that is tested during the experiments of Spatial Horizontal Edge Variation Feature.	95
4.2	Experiment Setting for Testing Spatial Horizontal Edge Variation Feature in detecting Vehicle License Plate	96
4.3	Results for Classifiers with different number of regions corresponding to the same Horizontal Edge Variation Threshold	99
4.4	Results for Classifiers of Spatial Horizontal Edge Variation feature with 1 region corresponding to the different Horizontal Edge Variation Image Threshold	104
4.5	Results for Classifiers of Spatial Horizontal Edge Variation feature with 3 regions corresponding to the different Horizontal Edge Variation Image Threshold	105
4.6	Results for Classifiers of Spatial Horizontal Edge Variation feature with 6 regions corresponding to the different Horizontal Edge Variation Image Threshold	106
4.7	Results for Classifiers of Spatial Horizontal Edge Variation feature with 9 region corresponding to the different Horizontal Edge Variation Image Threshold	107
4.8	Experiment Setting for Testing Spatial Horizontal Edge Variation Feature in detecting Text from natural scene	109
4.9	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 30 pixels.	110

4.10	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 45 pixels.	111
4.11	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 60 pixels.	112
4.12	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 70 pixels and horizontal edge variation image testing threshold of 45 pixels.	113
4.13	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 70 pixels and horizontal edge variation image testing threshold of 70 pixels.	114
4.14	Experiment Setting for Testing Spatial Horizontal Edge Variation Feature in detecting UIUC Car Dataset.	117
4.15	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 30 pixels.	118
4.16	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 45 pixels.	119

4.17	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 60 pixels.	120
4.18	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 90 pixels and horizontal edge variation image testing threshold of 45 pixels.	121
4.19	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 95 pixels and horizontal edge variation image testing threshold of 70 pixels.	122
4.20	Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 90 pixels and horizontal edge variation image testing threshold of 90 pixels.	123
4.21	Table shows the images that we used for the analysis on how each feature in the cascaded feature react on each images.	129
4.22	Comparison between the original image and Horizontal Edge Variation Image (with threshold of 45 pixels)	134
4.23	Comparison between the original image (large image) and Horizontal Edge Variation Image (with threshold of 45 pixels)	134
4.24	Comparison between the original image and Horizontal Edge Variation Image (with threshold of 45 pixels)	135

4.25	Table shows the images that we used for the analysis on how each Spatial Horizontal Edge Variation feature in the cascaded feature react on each images (car and Non Car Image).	137
4.26	Table shows the images that we used for the analysis on how each Spatial Horizontal Edge Variation feature in the cascaded feature react on each images (text and Non text Image).	139
4.27	Results from the experiments of testing how edge image affecting the performance of the vehicle license plate detector trained with Spatial Horizontal Edge Variation feature. The threshold set for the canny edge function in OpenCV (CvCanny – threshold(minimum) = 128, threshold (maximum) = 255).	142
4.28	Results from the experiments of testing how edge image affecting the performance of the vehicle license plate detector trained with Spatial Horizontal Edge Variation feature. The threshold set for the canny edge function in OpenCV (CvCanny – threshold(minimum) = 0, threshold (maximum) = 255).	142
4.29	Results from the experiments of comparing both different classification method, Cascaded Feature Classification Vs Support Vector Machine Classification.	146
4.30	Results from the experiments of comparing Cascaded Feature Classification and the Cascaded Feature with Support Vector Machine as false removal.	148
4.31	Comparison between the original image (large image) and Horizontal Edge Variation Image (with threshold of 30, 45, and 60 pixels) (Side View Car Image).	149
4.32	Comparison between the original image (large image) and Horizontal Edge Variation Image (with threshold of 30, 45, and 60 pixels) (Vehicle license plate Image).	151
4.33	Comparison between the original image (large image) and Horizontal Edge Variation Image (with threshold of 30, and 45 pixels) (Image with text).	153

LIST OF FIGURES

Figures		Page
2.1	Detection results from the techniques proposed by Bai & Liu. Image adapted from (Bai & Liu, 2004)	16
2.2	Roberts edge operator Template	19
2.3	Sobel Operator	19
2.4	Prewitt Operator	20
2.5	Krisch operator	20
2.6	Diagrams adapted from the (Nikzad, Dehkordi, Ekhlas, & Azimifar, 2010) showing the proposed Peak-Valley filter (PVF) results where (a) Original gray image, (b) Gray level changes along the 321th row of the gray image (white line), (c) PVF results before the combining the same consecutive transitions, (d) PVF result after combining, (e) gray level changes of the line inside the plate, (d) PVF result of (e)	22
2.7	Gentle Adaboost procedure used to construct a strong classifier. Adapted from: (Viola & Jones, 2004). Robust Real-Time Face Detection. International Journal of Computer Vision 57(2), pg 142.	34
2.8	Simple Haar-like basis functions used as features in license plate detection scheme. Adapted from: (Viola & Jones, 2004). Robust Real-Time Face Detection. International Journal of Computer Vision 57(2), pg 139.	35
2.9	Diagram depicts of Cascade of a set of Weak Classifiers Selected by Adaboost Learning Algorithm Adapted from: Viola (Viola & Jones, 2004). Robust Real-Time Face Detection. International Journal of Computer Vision 57(2), pg 144.	36
2.10	Diagram shows the distribution of gradient density in the area of (a) license plate, and (b) Background image. Adapted from: (Zhang, Jia, He, & Wu, 2006). Learning-Based License Plate Detection Using Global and Local Features. 18th International Conference on Pattern Recognition (ICPR'06), pg 1102-1105.	37

2.11	Features that compute properties average within the region. Adapted from: (Chen & Yuille, 2004). Detecting and Reading Text in Natural Scenes. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04), pg 3.	39
2.14	Six Feature type with different geometrical layouts. Adapted from: (Treptow & Zell, 2004) Figure showing the 6 feature types that are implemented in Treptow & Zell Adaboost learning algorithm with Evolutionary Search.	41
2.13	Adapted from: (Abramson, Moutarde, Steux, & Stanciulescu, 2006). Figure showing the control points feature implemented in (Abramson, Moutarde, Steux, & Stanciulescu, 2006).	42
3.1	System Architecture of the Training Stage in License Plate Detection Learning Framework	50
3.2	System Architecture of the Testing Stage in License Plate Detection Learning Framework	54
3.3	Ways to calculate Horizontal Edge Variation image <i>ImgHEV</i> from Original Grayscale image, <i>Imgg</i> .	58
3.4	Adapted from (Viola & Jones, 2004). Diagram shows how to compute the sum of all pixels in any regions of the integral image.	59
3.5	Integral Image. Adapted from (Viola & Jones, 2004) The value of the integral image at point <i>x,y</i> is the sum of pixels above and to the left.	60
3.6	Spatial Horizontal Edge Variation Feature within the sub-window with the size of Width x Height.	65
3.7	Figure shows graphs of feature values computed from a single feature on two groups of samples, which are positive samples and negative sample. If the feature minimum hit rate is set 1.0, then the feature threshold is 0.75, as the feature is required to achieve 100% of acceptance rate.	66

3.8	Figure shows graphs of feature values computed from a single feature on two groups of samples, which are positive samples and negative sample. If the feature minimum hit rate is set 0.8, then the feature threshold is approximated to be 0.85, as the feature is only required to achieve 80% of acceptance rate.	67
3.9	adapted from (Mitchell, 1997), showing the single-point crossover operator that recombining two parent hypotheses and generating two offspring with the crossover mask of a single point.	69
3.10	Adapted from (Mitchell, 1997), showing the point mutation operator that modifying a value of a bit point selected randomly onto the two offspring .	70
3.11	Diagram adapted from (Mitchell, 1997), showing the general prototypical genetic algorithm with fitness proportionate selection, crossover operator, point mutation operator.	70
3.12	Bits String of the hypothesis	73
3.13	Selecting a random point for single-point crossover.	74
3.14	single-point crossover mask	74
3.15	Generate offspring from selected parent using single-point crossover	74
3.16	Examples of Generate offspring from selected parent using single-point crossover	75
3.17	Single Point Mutation	75
3.18	adapter and modified from (Viola & Jones, 2004), diagram showing the cascaded feature structure of processing each sub-window, each feature in the cascaded feature will determine whether to further process the sub-window or rejecting it if any of the feature classified the sub-window as non-target object.	77
3.19	Sub-window scanning Process, Move Sub-window based on some number of pixels, $mp \Delta$	80
3.20	Sub-window scanning Process, Rescale Sub-window based on rounding scale factor $[sf \Delta]$	80

3.21	Sub-window scanning Process, cascaded feature classification and SVM false detection removal.	81
3.22.1	System Architecture for Training Stage in Object Detection using Spatial Horizontal Edge Variation Feature.	82
3.22.2	System Architecture for Training Stage in Object Detection using Spatial Horizontal Edge Variation Feature.	83
3.22.3	System Architecture for Training Stage in Object Detection using Spatial Horizontal Edge Variation Feature	84
3.22.4	System Architecture for Training Stage in Object Detection using Spatial Horizontal Edge Variation Feature.	85
3.23.1	System Architecture for Testing Stage in Object Detection using Spatial Horizontal Edge Variation Feature.	86
3.23.2	System Architecture for Testing Stage in Object Detection using Spatial Horizontal Edge Variation Feature.	87
4.1	shows a brief testing framework for this project.	90
4.2	Training Dataset and Testing Dataset for each object detection problem	93
4.3	Positive Training Sample consist of Artificial Vehicle License plate images	94
4.4	Negative Training Sample consists of Non-Vehicle License plates images collected from World Wide Web.	94
4.5	shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers	98
4.6	Example of the Actual License Plate and the Description of the Spatial Horizontal Edge Variation Feature Region Distribution	100
4.7	Graph with feature value generated for the License Plate Image From the test images (Refer to Figure 4.8)	101

4.8	Actual License Plate extracted from the Test Images	102
4.9	Graph with feature value generated for the Non-License Plate Image From the test images (Refer to Figure 4.10)	103
4.10	Non License Plate extracted from the Images	103
4.11	shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers. The Graphs show the comparison of feature with 1 region on different Horizontal Edge Variation Image Threshold	105
4.12	shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers. The Graphs show the comparison of feature with 3 regions on different Horizontal Edge Variation Image Threshold	106
4.13	shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers. The Graphs show the comparison of feature with 6 regions on different Horizontal Edge Variation Image Threshold	107
4.14	shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers. The Graphs show the comparison of feature with 9 regions on different Horizontal Edge Variation Image Threshold	108
4.15	shows the recall rate and the corresponding precision rate of four text detectors [1 bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 30 pixels	111
4.16	shows the recall rate and the corresponding precision rate of four Text detectors [1 bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 45 pixels	112
4.17	shows the recall rate and the corresponding precision rate of four Text detectors [1 bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 60 pixels	113

4.18	shows the recall rate and the corresponding precision rate of three Text detectors [3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 70 pixels, and testing with threshold of 45 pixels	114
4.19	shows the recall rate and the corresponding precision rate of three Text detectors [3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 70 pixels, and testing with threshold of 70 pixels	115
4.20	shows the combinations of all the graphs from Figure 4.5 until Figure 4.9 The recall rate and the corresponding precision rate of Text detectors with different training threshold and testing threshold and different number of regions [1 bin, 3 bins, 6 bins, and 9 bins] are shown from the graph	115
4.21	shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 30 pixels	119
4.22	shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 45 pixels	120
4.23	shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 60 pixels	121
4.24	shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 90 pixels, and testing with threshold of 45 pixels	122
4.25	shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 90 pixels, and testing with threshold of 70 pixels	123

4.26	shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 90 pixels, and testing with threshold of 90 pixels	124
4.27	shows the combinations of all the graphs from Graph 1.14 until Graph 1.19. The recall rate and the corresponding precision rate of Side View Car detectors with different training threshold and testing threshold and different number of regions [1 bin, 3 bins, 6 bins, and 9 bins] are shown from the graph	125
4.28	Diagram shows the Spatial Horizontal Edge Variation features selected by the system, where each black rectangle is the sub-window with 85 x 25 with a sub rectangle internally (white color). These features are selected for vehicle license plate detection, with 6 regions of the Spatial Horizontal Edge Variation feature	128
4.29	The Graph shows the distributions of the feature values of each bin in the 1st Spatial Horizontal Edge Variation feature from the cascaded features (vehicle license plate detector)	130
4.30	The Graph shows the distributions of the feature values of each bin in the 2nd Spatial Horizontal Edge Variation feature the cascaded features (vehicle license plate detector)	131
4.31	The Graph shows the distributions of the feature values of each bin in the 3rd Spatial Horizontal Edge Variation feature from the cascaded features (vehicle license plate detector)	132
4.32	The Graph shows the distributions of the feature values of each bin in the 4th Spatial Horizontal Edge Variation feature from the cascaded features (vehicle license plate detector)	132
4.33	The Graph shows the feature values of each Spatial Horizontal Edge Variation feature from the cascaded features (vehicle license plate detector). The feature values from this graph is different from the graph previously, where the feature values for each feature is computed from comparing the 6 bins of each feature with the feature template (6 bins)	133

4.34	Diagram shows the Spatial Horizontal Edge Variation feature (9 bins) selected by the system for side view car detection, where each black rectangle is the sub-window with 100 x 40 with a sub rectangle internally (white color)	136
4.35	Graph shows the features values generated by each Spatial Horizontal Edge Variation Features (9 bins) from the cascaded features list. Any images that are classified as non car will not be processed by the rest of the features in the list.	137
4.36	Diagram shows the Spatial Horizontal Edge Variation features (1 bin) selected by the system for text detection, where each black rectangle is the sub-window with 65 x 25 with a sub rectangle internally (white color)	138
4.37	Graph shows the features values generated by each Spatial Horizontal Edge Variation Features (1 bin) from the cascaded features list. Any images that are classified as non car will not be processed by the rest of the features in the list	139
4.38	Images of original testing image 1(a), converted edge image with the minimum threshold of 0 and maximum threshold of 255 1(b), converted edge image with the minimum threshold of 128 and maximum threshold of 255	143
4.39	Diagram shows the results of the detection by the vehicle license plate detector trained with edge image. Images of original testing image 1(a), converted edge image with the minimum threshold of 0 and maximum threshold of 255 1(b), converted edge image with the minimum threshold of 128 and maximum threshold of 255	144
4.40	Images failed to be detected by the Text Detector trained by the System with (Spatial Horizontal Edge Variation Feature)	154
4.41	Images failed to be detected by the license plate Detector trained by the System with (Spatial Horizontal Edge Variation Feature)	155

CHAPTER ONE

INTRODUCTION

As the increase of vehicle usage among the nation, there are significant increases in the demands for automated vehicle license plate recognition system (ALPRS), and it can be applied as car park management system, police enforcement on tracking vehicle violating traffic rules, gated community security, automation for entrance access, and etc. There are many challenges in designing a robust ALPRS for the real world application as there are many variations that could affect the performance of the system. Variations such as lighting illumination, occlusion, noise, weather conditions, image quality, and etc become challenges and problems to the system. ALPRS consists of few main modules such as license plate detection, license plate segmentation, and characters recognition. License plate detection is one of the important modules in the license plate recognition system (Zhao, Gu, Liu, Han, Gao, & Hu, 2010). It is a two-class pattern classification problem that classifies between the license plate and non-license plate. There are many existing algorithms that specifically designed to solve the problem, and many research in this field to further enhance the existing problem, for example edge based vehicle detection with color aided on license plate detection (Abolghasemi & Ahmadyfard, 2009), License plate detection using SIFT Features by (Zahedia & Salehia, 2011). There are systems using

discrete wavelet transform (DWT) to extract license plate from the image. In (Wang, Lin, & Horng, 2011), they achieve accuracy rate of 97.33% under complex environments. In this project, we perform research on license plate detection problem, analyze the existing system with their advantages and drawbacks. After that, from the technical reviews, we introduced a new feature set as we named it as Spatial Horizontal Edge Variation feature to perform the license plate detection. With this feature set, we design a framework to generate the final classifier to perform the detection by selecting some of the best features through number of round of training on the image samples. For our license plate detector, we tested with some of the Malaysian vehicle license plate and achieve recall rate of 0.93796 and precision rate of 0.55366. Subsequently, we test our proposed framework onto the text detection problem, but we could not get relatively high accuracy for this problem. Analysis and discussions are further elaborated in this thesis for more details. Though we are not able to obtain the comparable same accuracy from the rest of the existing techniques, but yet we can observe the information extracted by our edged feature selected from the framework. Perhaps this could be an initiation or inspiration for the next ideas for better improvement.

1.1 Problem Statement

In the real world of vehicle license plate detection application, there are many challenges and problems in the industry, such as lighting illumination issue due to environment changes (A. L., Choong,, Wong, & K. T. K., 2011), license plate characters occluded from the plate, slanted or rotated license plate, size

orientation of the characters from the plate, color, blurring images, low light condition, and etc (Zhao, Gu, Liu, Han, Gao, & Hu, 2010). Many applications applied image pre-processing, noise filtering, image normalization and binarization to remove noises from the image and using edge based techniques to extract connected components (Satish, Lajish,, & Kopparapu,, 2011), and etc. All these techniques designed specifically to solve the license plate detection problem, and in our project we have come out with few problem statements that become the objectives of this research.

- How to design a license plate detector learning framework that requires minimum manual parameters tuning and setting. This framework will automate the process of finding the features for the classification whether it is a license plate or non license plate. The framework will learn the patterns and characteristic of the license plate from the image samples provided, and from there it generates classifiers to perform the detection.
- How to design a new features type to extract data information from the license plate image, and these features should be capable of generating features values that able to distinguish between license and non license plate accurately.
- How to design an algorithm to select efficient features from the large set of features generated automatically from the framework. The algorithm auto-generate random features at the beginning of the training process, and

after that it will learn and evolves the feature set and patterns, generating improved features for next iteration of feature selection.

1.2 Motivation

As the growing usage of vehicles in the nation, monitoring system is required to be functional for 24 hours in 7 days of a week. The monitoring system such as license plate recognition system is essential for the security tracking purpose. License plate detection is one of the core modules of the recognition engine, as it helps to locate the vehicle license plate correctly so that it can increase the rate of false alarm. By understanding the license plate image patterns characteristics in computer, perhaps we can obtain more information through the research and implementation. In business point of view, there are a lot of business opportunities in this industry such as building a license plate recognition system in the car park for managing the vehicle authentication that could increase the security of the community. Besides there are other system such as police enforcement usage to capture images of the vehicle moving in high speed or violating the traffic rules, by using the license plate recognition system, it aids to reduce the enforcement processing time by automate the process of recognizing the license plates from large set of images. This could then increase the effectiveness and efficiency.

Besides the business value and opportunities from the industry of automating vehicle license plate recognition, there are other motivations in this project. Previously we researched in the face detection framework by Viola and

Jones (Viola & Jones, 2004), they were able to generate a robust face detection system by using a framework named Adaboost, and their framework requires sufficient images as the samples to *train* a detector that could be used in the classification. Then, we had been wondering whether the framework would generate a detector that achieves relatively high accuracy in detecting vehicle license plate if we modify the image samples to become vehicle license plate images. By tuning different parameters in the framework and preparing different image samples, we applied some experiments to verify the feasibility of the existing framework could be used for the license plate problem. However, things did not turn out to be what we expected; the detector we generated failed to achieve high accuracy, then we further analyzed the results to understand what was not right in our implementation. From the framework, we found that there are some important modules and elements in the framework that could largely affect the detection accuracy. One of the major modules is the feature set, as the features derive the information about the object image patterns and characteristics, and the classifiers generated by the Adaboost framework will further decide whether the give object is the target object or non target object. However, determining the best feature set is very important; as not every feature set is adequate for solving every object detection problem. Some feature set might be achieving high accuracy in detecting face detection, but it might not be too feasible when it come to detecting other object such as text detection. (Chen & Yuille, 2004) Therefore, one of the motivations in this research is to determine the feature set that is adequate for different object detection problems.

A single feature might seem to be simple and difficult to make a complex decision whether the image is the target object or non target object. But yet, large sets of features could actually combine their hypotheses together to make the correct decision. However, the question is how to select the *right* features from the large set of feature set? There are many answers for the above question; we could apply hill climbing approach to heuristically search through the large set of features and discard all non-effective features. This technique is good that it will search through the whole large possible feature set and not missing any good feature, but this might not be efficient as it would be time consuming and resource consuming. Therefore, another motivation of this project is that we are interested to design and implement the architecture to automate the process of searching good features from the large feature set and with this type of architecture setting, user can reduce the time of selecting effective and efficient feature compare to manually searching or other heuristic-based searching. From all the learning approach, we apply this technique as it is simple to implement and it could provide an optimal solution through evolution of the features.

1.3 Objectives

The main objectives of this research are listed as below:

1. Design and implement a learning-based object detection framework that can be applied to license plate detection, and text detection.

2. Analysis on the feasibility of the feature set in extracting edge information from license plate and text from natural scene.
3. Implement Evolutionary Feature Searching techniques to decrease time in searching large feature set and using this technique to evolve and find features that provide the optimized solution.

1.4 Scope of Works

In this project the scope of works are described as below:

- Research in current existing object detection, license plate detection, text detection problems, what are the strengths and weaknesses of the existing systems.
- Research in the techniques and frameworks implemented by other people in object detection, face detection, license plate detection, and text detection.
- Analysis in the features used for face detection, license plate detection, text detection.
- Design and implement feature set to solve the license plate detection problem.
- Design and implement the learning based framework to generate the final classifier from the large set of features.
- Collect the training and testing samples for license plate detection problems and text detection problems.

- Perform experiments on the implemented framework by testing on different parameter settings.
- Analysis and discussions on the results obtained from the experiments.

1.5 Section Description

This project has divided into few chapters, whereby each chapters describe the whole process of this project from initial research to the design and implementation as well as testing. Below are the chapters and section description.

- Chapter 1 – Introduction
 - Introduction of the whole projects, depicting the background of previous research related to this project and the brief prologue of what are the tasks and modules achieved in this project and the findings.
- Chapter 2 – Literature Review
 - Studies and research of the current related projects and system. Understand how other similar techniques already applied to solve the problem. From each piece of information, reveal the steps of getting the idea of the evolved features and framework.
- Chapter 3 – System Description
 - Explanation of whole system architecture and framework designs. Providing details of the formulae and techniques to implement the conceptual idea of this research project.

- Chapter 4 – Experiment Setting and Results & Discussions
 - Descriptions of the experiments settings applied in this project to test the implemented framework. Image samples are shown in this project for reference. Each experiments objectives and results are recorded in this chapter. After obtaining the results from the experiments, we will further discuss the results analysis.
- Chapter 5 – Conclusion
 - Final Chapter of this research project, showing the conclusion and finding throughout the whole process. Success and failures from the experiments illustrate the challenges and achievements of the project. As a startup idea, further research and analytics in the future, perhaps could lead to more inspirations in the ideas of solving the problems whether in Computer Vision of any other fields.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

As the Intelligent Traffic System (ITS) industry starting to grow in the world, LPRS become one of the important ITS applications that applied to the car park ticketing system, car park security checking, gated community access control, toll collection, and etc. License plate detection is an important module in the automatic license plate recognition system (ALPRS) , as the system will start to search for the location of vehicle license plate from the image, and extract the region for further license plate character segmentation before passing the characters to any classifiers for recognition. There are many challenges in the license plate detection due to the environment changes, the lighting illumination variation that keep changing from time to time as the camera is installed at the outdoor environment. Besides, other challenges such as the size, orientation of the characters of the license plate, occluded or dirt that found from the license plate and etc. License plate recognition technology that applied in the industry require real time processing and this increase the complexity challenge for the algorithm to achieve high accuracy while maintaining short processing time. In our literature review, we will study the existing license plate detection techniques on their advantages and drawbacks, after that we research on different kind of features

type which are used in extracting license plate information from the image. Then we conclude our chapter 2 with the overall summary of the literature review that we conducted.

2.2 Background & Previous Work

Our previous works were to design the framework that can generally create classifiers that are able to distinguish between the target object and non target object based on specific training samples. Initially we research on face detection framework by Viola and Jones selects informative features from a large set of rectangle features. (Viola & Jones, 2004), after that we tried to further extend the research to implement the same framework to solve license plate detection problem. However, we failed to achieve relatively high accuracy from our experiments. During our experiments, we failed to obtain sufficient Malaysia actual vehicle license plates as our training samples; therefore, we implemented artificial vehicle license plates as the training samples. This is one of the reasons that caused the failure in generating the *good* license plate detector to achieve high accuracy rate using the framework. Then, we also found that feature also greatly affecting the detection results as some of the features might be efficient and effective in achieving high accuracy in detecting specific object, but yet failed to apply the same feature set onto other object detection problem. From our findings and researches, Haar-like features that show proficient in detecting face using Viola & Jones framework might not be so feasible when it comes to detect text and license plate. This is because according to (Chen & Yuille, 2004), Haar-

like features in the face detection framework is efficient in solving object with spatially similar characteristic for example human face that having eyes, nose, mouth and ears which are sharing the partially similar position among other human faces. By understanding the characteristics and patterns from the face and text as well as license plates, we can extract different information for each of them. From human eyes, we can actually observe the difference of the text and license plate, they are sharing the similar patterns as both also having the alphanumeric characters, and human face on the other hand is totally different problem domain.

Chen and Yuille (Chen & Yuille, 2004) in their natural scene text detection, they implemented Adaptive Boosting for selecting informative features. Chen and Yuille (Chen & Yuille, 2004) stated that the set of features used for frontal face detection by Viola and Jones (Viola & Jones, 2004) provide the advantage in computation time but are not suitable in text detection. This is because in face detection, every facial feature such as eyes, nose, and mouth share roughly the same spatially positions. This shows face is considered as a spatially similar object while the text is spatially dissimilar object. Texts in license plate are different from one car to another that will become a challenge for the learning based algorithm. This is because for learning based algorithm, training samples are the essential factor in constructing a robust object detection application. However, the questions are how to determine which training samples are suitable for certain problems, and how many training will be appropriate.

Providing large training samples might improve the performance but this would not be an efficient method in solving the problem. Rather, we need to determine a way to efficiently improve the detector performance. Kobi Levi and Yair Weiss (Levi & Weiss, 2004) in their paper stated the importance of good and efficient features for learning a robust object detector with a small number of training samples compare to (Viola & Jones, 2004). They implemented local edge orientation histograms (EOH) as features compared to standard linear features (Levi & Weiss, 2004). Referring to Wang, Li and Zhang in (Wang, Li, & Zhang, 2006), they applied histogram feature as the novel features for face detection, and they had showed that the selected histogram features are much representative in face detection problem comparing to Haar-like features implemented in (Viola & Jones, 2004). In (Zhang, Gao, Chen, & Zhao, 2005), they implemented spatial histogram as the features in their object detection frameworks as spatial histogram provides discriminative domain knowledge about the shape, texture, local patches of the objects. However, not every feature set feasible and robust for different objects detection problems. Therefore, one of the problems we will encounter is to determine robust features set that is feasible in detecting license plate and other object detection, such as face, text, car detection, and so forth.

Selecting a set of good and efficient features from the exhaustive set of features can be time consuming and computational expansive. In (Viola & Jones, 2004), their framework providing 160000 of rectangle features during the training of a face detector with the base resolution size of 24×24 . Although they introduce integral image as image representation for fast feature computation but

yet calculating the complete set of features still relatively expensive. Therefore, in (Treptow & Zell, 2004), André Treptow and Andreas Zell showed the replacement of evolutionary search algorithm in feature selection can improve the detection rate as it can automate the learning of features and able to search for better features from the larger set in a shorter time. Another technique in automating feature learning and selection is combining Adaboost with hill climbing and evolutionary search for object detection. (Abramson, Moutarde, Steux, & Stanciulescu, 2006) In order to select better features in comprehensive and large feature set, exhaustively feature searching might not be feasible. Therefore, we need to determine methods to efficiently automate the feature searching and combining the selected features into a structure that can robustly classify between the target object and the non target object.

2.3 License Plate Detection

There are many techniques in license plate detection; one of the techniques is the hybrid of license plate extraction method based on edge statistic and morphological techniques (Bai & Liu, 2004). In their research they proposed the system that consists of four steps, including the vertical edge detection, edge statistical analysis, hierarchical-based license plate location, and morphology-based license plate extraction. After they implemented the normalization and filtering to reduce the noise caused by the lighting illumination, they perform a vertical edge detection to generate the image edge map from the normalized gray image. Then, in their system they have few steps to extract all the possible region

of interest that contains license plate. These steps including gathering the feature points and combine them into lines and by using thresholding to retain all the lines that are having density within the minimum and maximum threshold value. All the lines that are close to each other within the distance threshold will then be joined to form into rectangles. Reduce the number of rectangles, only rectangles with more than the minimum lines density is preserved. With all these rectangles, many of them are actually false detection which they do not contain any license plate; therefore, Bai & Liu proposed extra steps to further remove the false detection. The technique they used is the morphological based to further find the license plate from the image by getting area with high edge density. Equation 2.1 shows the formulae to calculate the density of the edge pixels from the 3 x 15 block, where $d(i, j)$ is representing the edge density map, $g(i, j)$ is the center of the block.

$$d(i, j) = \frac{1}{45} \sum_{x=-1}^{x=1} \sum_{y=-1}^{y=7} g_v(i + x, j + y) \text{Mask}(i + x, j + y) \quad (2.1)$$

From their experiments, they showed an impressive good accuracy for their proposed technique, whereby they are able to achieve a detection rate of 99.6% on the image of 768 x 534 resolutions.



Figure 2.1 Detection results from the techniques proposed by Bai & Liu. Image adapted from (Bai & Liu, 2004)

Besides this algorithm, there are many other algorithms that implementing the techniques to extract information from the edges as the contrast between the background and the license plate characters contain the prerequisite information whether the area consists of any license plate. Zhang and Zhang in their paper shows they implemented a license plate detection algorithm that able to perform well under different conditions and robust against illumination changes, color difference, vehicle positions variance, and etc. They proposed a multi-feature such as mathematical morphology, rectangle features, edge statistics and character features that claimed to be able to perform well even in several regional license plates (Zhang & Zhang, 2010). Their morphological techniques consist of Sobel Orientation (generating vertical edge map), edge density and OSTU method to create the binarization map, and then they obtain bigger size of blobs through dilations. Next, they used Connected Component Analysis (CCA) to extract all the blobs that their pixels are connected to each other, and combine the blobs into

rectangles. Similar to (Bai & Liu, 2004), Zhang (Zhang & Zhang, 2010), they using the technique of filtering rectangles that are not enough lines density. This vertical edge line filtering method does not able to remove line-type of object from the image such as the barrier at the front of the vehicle. So, from the edge image, the barrier at the front of the vehicle might be joined together with the license plate after the binarization and this results a larger blob, where they solve it by separating them into smaller rectangle, and from there they perform character detection onto each candidate rectangles by verifying whether it is a license plate or non license plate region. Their approach able to achieve 99.3% of detection rate for American plates detection and 99.65% of detection rate on Chinese character plates, as they claimed that American plates have more samples that are very unique. This algorithm although it is able to perform well under different condition and complex environment, but yet it still needs lots of works to do in orders to solve other exceptional, unique and personal license plate.

Another example of locating license plate through detecting lines is by (A. Ravi, Shashank, Abhishek, & Kandadvli, 2010), where they proposed the use of adaptive threshold and contours to extract the lines and they proved that their algorithm implementing adaptive threshold is able to outperform Canny Edge detection in extracting license plate. From the research, we found that generating the edge map of the original image to find the license plate becoming one of the simplest techniques and commonly use in many system. As the characters from the license plate are having higher contrast compare with their background. In (Kim, Han, & Hahn, 2008), they presented a license plate method that uses edge

detection and closed loop detection. From the closed loop detection, they extracted character candidates and they perform the vertical and horizontal edge lines detection from the binary images to further verify the candidate plates is a license plate or not. Their approach of closed loop is able to achieve 96.5% detection rate on finding the candidate characters, then from there the correct license plate is obtained by using vertical and horizontal edge lines, and the accuracy of getting the plate is 95.3% of detection rate.

While we being talking about how edge information becomes a common and simple way to extract license plate from the image, in (Guo, Hu, Hu, & Tang, 2010), they provide the analysis of few edge detection algorithm and 5 different operators in license plate recognition. The five convolution operators that showed in their paper given us the idea of how to extract edge information from the image, those operators are

- Roberts edge operator

This operator uses a partial differential method to find the edge operator, and their formula is shown from Equation 2.2. Then, Figure 2.2 shows the operator template to convolute over the image and generate the edge image.

$$g(x, y) = \left\{ [\sqrt{f(x, y)} - \sqrt{f(x+1, y)}]^2 + [\sqrt{f(x, y)} - \sqrt{f(x, y+1)}]^2 \right\}^{1/2} \quad (2.2)$$

1	
	-1

	1
-1	

Figure 2.2 Roberts edge operator Template

- Sobel operator

This is an operator that uses discrete differentiation methods to calculate the estimated of the gradient from the original image by convoluting the image with a filter shown at Figure 2.3 in vertical and horizontal direction to get the estimates of the derivatives changes of these two directions. This operator is rather simple and easy in computation, that making it fast in computing the edge of the image in short period of time, and it even applied in vehicle recognition. (Wang W. , 2009)

1	2	1
-1	-2	-1

-1		1
-2		2
-1		1

Figure 2.3 Sobel Operator

- Prewitt operator

Prewitt operator uses two 3 x 3 kernel to calculate the estimation of the derivatives for the vertical changes and the horizontal changes. This will generate the gradient of the image intensity and provide the direction of the highest changes of the lighting from bright to dark and as well as the rate of changes in that direction.

-1	-1	-1
1	1	1

-1		1
-1		1
-1		1

Figure 2.4 Prewitt Operator

- Krisch operator

Krisch operator is an edge detector that locates the maximum edge in eight different directions. Each of the kernels convolutes through the image and determine the maximum magnitude of the edge values and from all the images generated, the edge direction is determined by the kernel that produce the largest magnitude of the edge value.

+5	+5	+5
-3		-3
-3	-3	-3

-3	+5	+5
-3		+5
-3	-3	-3

-3	-3	+5
-3		+5
-3	-3	+5

-3	-3	-3
-3		+5
-3	+5	+5

-3	-3	-3
-3		-3
+5	+5	+5

-3	-3	-3
+5		-3
+5	+5	-3

+5	-3	-3
+5		-3
+5	-3	-3

+5	+5	-3
+5		-3
-3	-3	-3

Figure 2.5 Krisch operator

- Canny Edge Detection operator

Canny Edge operator uses the Gaussian filter to perform the smoothing on the image and it is a multi-scale edge detection that able to extract wide range of edge from the image. It implemented the Gaussian filter first derivative technique to remove noise that present from the image. Then it calculates the edge from the

image (after the noise remove using Gaussian filter) through horizontal, vertical, and diagonal.

Besides all approaches that use the edge extraction for license plate detection, there are other techniques such as the texture-based approach by (Nikzad, Dehkordi, Ekhlasi, & Azimifar, 2010) for their license plate detection system. In their paper they categorized license plate detection methods into few types, edge-based, texture-based, and transformation-based. This approach uses a filter called Peak-Valley filter to extract the distinctive gray level intensity changes and remove noises such as leaves, lighting illuminations. This Peak-Valley filter is applied to each line of the image and segmented into different partitions based on the periodicity structure and extract some features from each partition so that the feed-forward neural networks will be used as the machine learner classifier to determine whether the candidate lines belong to a vehicle license plate region and filter all the non license plate regions. The variance of the width and height of the peak and valley of each partition are used to generate the feature set to detect the possible true candidate.

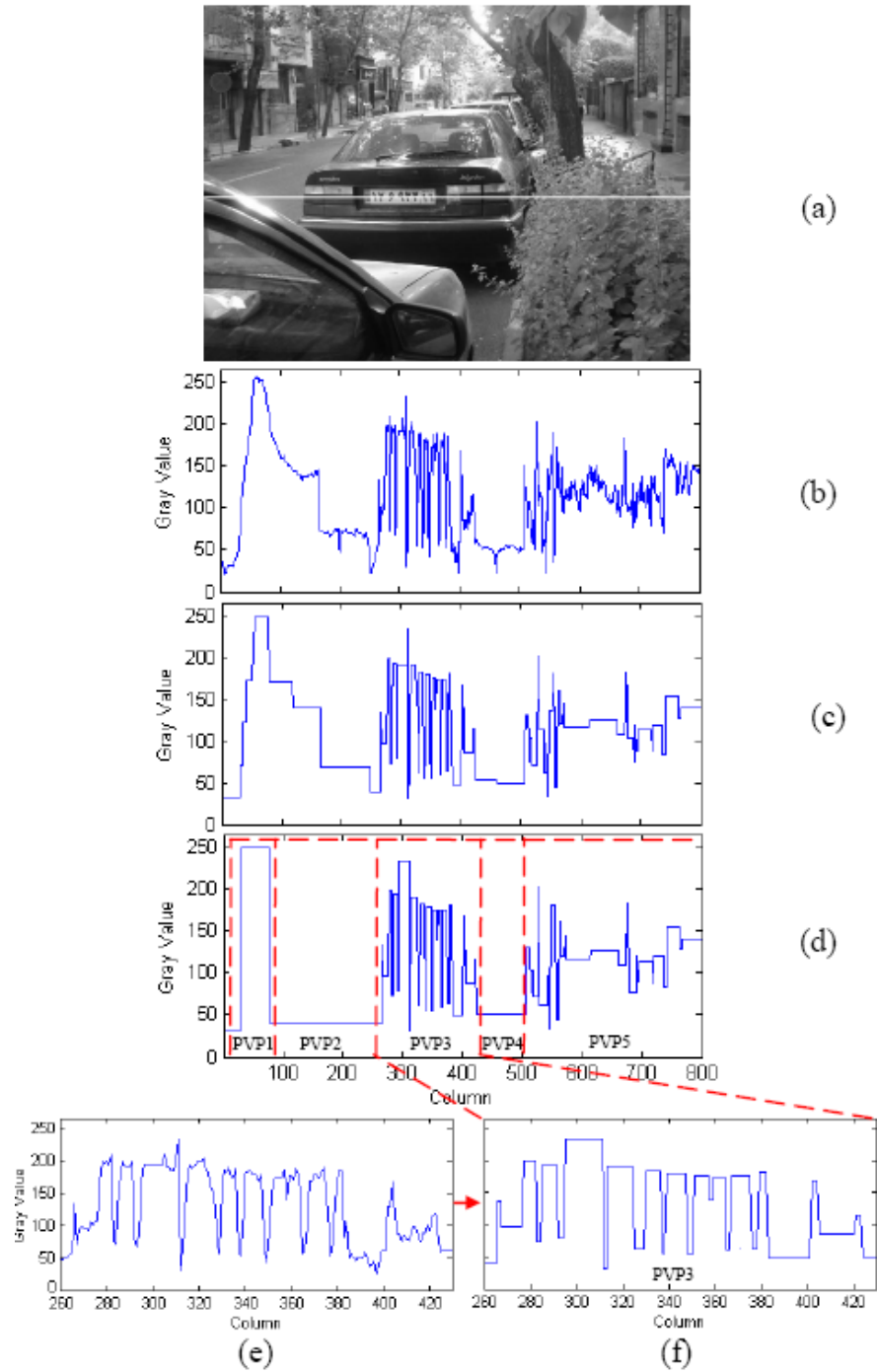


Figure 2.6 Diagrams adapted from the (Nikzad, Dehkordi, Ekhlasi, & Azimifar, 2010) showing the proposed Peak-Valley filter (PVF) results where (a) Original gray image, (b) Gray level changes along the 321th row of the gray image (white line), (c) PVF results before the combining the same consecutive transitions, (d) PVF result after combining, (e) gray level changes of the line inside the plate, (f) PVF result of (e)

From edge-based to texture based of license plate detection techniques, here we proceed to study on the license plate detection that implementing color-based technique. One of the approach uses color as the feature to perform the detection is by (Deb & Jo, 2008) in their paper presenting HSI (Hue, Saturation, and Intensity) color based vehicle license plate detection. Their proposed algorithm consists of the license plate color-properties extraction, shape-based verification and position histogram. In their system, they design their color arrangement based on Korean vehicle license plate and focuses on three different color for the plate during the license plate searching. They used the HSI model to search for the candidate region that having the predefined color. After that, the algorithm will be able to extract license plate candidates and further process the region by binary and gray scale the image and through connected component analysis, it able to further filter candidates that do not have the shapes and ratio similar to the license plate characters. This technique is able to obtain the detection rate of 89% on 90 test images, and from their testing analysis, they found that this approach is sensitive to the lighting and the system will be having problem when the license plate background is having similar color with the vehicle body. (Deb & Jo, 2008)

Although color-based technique might seems to be quite sensitive in performing the license plate detection in (Deb & Jo, 2008), there are hybrid technique that uses the edge and color information to solve the license plate problem. In (Miao, Wang, & Wang, 2009), they presented an automatic license plate detection based on edge density and color model; whereby their approach

implemented the image binarization using adaptive thresholding technique and remove some of the area that do not contain license plate. After that, the algorithm will use connected component technique to extract license plate characters from the binarized image, and combine them into a plate candidates by some predefined average height, ratio and direction of each connected components. This however, does not fully eliminate all the non license plate candidates; therefore, they implied further process to verify the candidates in order to remove those non license plate regions. Through the combination of the edge density and color feature, they able to remove regions that are not license plate while retaining those real license plates and achieve the detection rate of 97.7% from their testing.

The following method is having different approach comparing to the above discussed approaches, whereby this method is called the multi-layer weak filter (MWF) by (Zhang & Zhang, 2009) in their license plate detection. This MWF is having the idea similar to the Haar-like Classification by (Viola & Jones, 2004) in their face detection; where (Viola & Jones, 2004) presented the approach of combining a cascade of weak classifiers into a strong classifier and each weak classifiers will decide whether the candidate is a license plate or not, if the first weak classifier already rejected the candidate as a true license plate, then it will not be processed by the following classifiers, and this will indirectly increase the speed of processing.

Then the MWF presented by (Zhang & Zhang, 2009) has few weak filters for their license plate detector. The algorithm starts with the transformation of the color image from RGB to HSV and filter as much as possible for the non-license

plate through Hue and Saturation information. Therefore, at this stage, regions that do not contain license plate background color will be removed as much as possible.

Then their second weak filter is used to generate the chart of marginal density region based on the characters' space on vehicle license plate using the edge operator of Sobel from the vertical direction. After that, the third weak filter is to label the region of binary image from the marginal density, and remove regions that are non license plate based on the area shape, and color of the region. This approach achieve the error rate of 7.74% of license plate detection, and claimed to be more effective than the color characteristic detection and texture characteristic detection. However, the approach failed to detect license plates that are tilted too much or containing too much of dirt onto the plates.

2.4 Learning Based Framework

In this section we research and discuss regarding the learning based frameworks that applied for different kind of object detection such as face detection, pedestrian detection, car detection, license plate detection, and etc. All these frameworks has the common characteristic which is their approach is to generate the classifiers that learned from the samples. There are few researches of learning based and example based system for object detection (Viola & Jones, 2004) (Papageorgiou & Poggio, A Trainable System for Object Detection, 2000) (Mohan, Papageorgiou, & Poggio, 2001) (Papageorgiou & Poggio, A Trainable Object Detection System: Car Detection in Static Images, 1999). Papageorgiou

and Poggio have presented a trainable and universal system for object detection in cluttered environments without changing the general architecture (Papageorgiou & Poggio, A Trainable System for Object Detection, 2000) (Papageorgiou & Poggio, A Trainable Object Detection System: Car Detection in Static Images, 1999). They have developed systems for face, pedestrian, and car detection. They implemented Haar wavelet as the features that provide a lot of useful information in particular pattern descriptions. Haar wavelet is capable of describing the structure of the class for the object of interest and disregards the noises in an image. The research applied support vector machine (SVM) classifier as the learning algorithm to distinguish between the object of interests and other patterns. It is a supervised learning method that minimized the boundary of the empirical error for the classifier.

Mohan, Papageorgiou, and Poggio presented example-based object detection in images by locating the object constituent components and combining the detected components if the configuration is valid (Mohan, Papageorgiou, & Poggio, 2001). The system will detect the basic components for a particular object and combining all the related components with a classifier if the geometrical configuration is met. They implemented the architecture to detect pedestrian and the system is different from the pedestrian detector by Papageorgiou and Poggio (Papageorgiou & Poggio, A Trainable Object Detection System: Car Detection in Static Images, 1999). Their system is able to trace the frontal, rear, and side view of the pedestrian using wavelet-based features. However Mohan, Papageorgiou, and Poggio claimed that the system is unable to locate occluded pedestrian with

some features missing from the image. Therefore, they proposed a components-based object detection architecture that detects an object by identifying the significant components for the object rather than the whole object.

The component-based detectors process the regions in an image by applying Haar wavelet and classifying the data vector. They used quadratic Support Vector Machine (SVM) as the component classifier and the candidate component that generates high component score value will be identified as a strong candidate. The candidate with the highest score will be fed into the combination classifier to further verify if the pattern is a pedestrian. They used a linear SVM classifier for the component classifier, and hierarchically detect different components at one level and similar combination classifier at another level to identify the object as pedestrian or non-pedestrian.

Tuzel, Porikli, and Meer (Tuzel, Porikli, & Meer, 2006) presented a region descriptor that used in pattern classification problem, such as object detection and texture classification. They proposed a region descriptor to calculate the covariance of several images inside a region of interest. They use covariance as their feature and compute it using integral image to detect the object of interest. Porikli and Kocak (Porikli & Kocak, Robust License Plate Detection Using Covariance Descriptor in a Neural Network Framework, 2006) presented a license plate detection algorithm that applies an image descriptor and calculate the covariance matrix of low-level pixel-wise features to locate the position of the license plate for a given image. Their approach is robust against the illumination,

rotation, noise, and yet to achieve a fast detection speed with less computation of the covariance matrix.

Some of the algorithms are designed specifically for a particular object detection problem such as the adaptive algorithm for text detection from natural scenes by Gao and Yang (Gao & Yang, An Adaptive Algorithm for Text Detection from Natural Scenes, 2001) . They developed a prototype system that can recognize Chinese sign inputs. The algorithm is designed in hierarchical structure with different conditions regulated in each layer. The algorithm can be applied to text in other languages by modifying the layout constraints.

For some learning-based algorithms, we need to determine suitable features for a particular object detection problem, yet, the architecture proposed by Papageorgiou and Poggio (Papageorgiou & Poggio, A Trainable System for Object Detection, 2000), has proven to be successful in face, pedestrian, car detection. They claimed that it is a general, trainable system for object detection using the same architecture.

2.5 Heuristic-based algorithm

Compare with the learning based framework, in this section we study different frameworks for the object detection; one of the approach is frontal view face detection by (Balasuriya & Kodikara, 2001). They implemented a deformable template algorithm to do face detection. The template is based on the bright and dark intensities invariant since there are few regions of the face provide significant gray scale intensities invariance, such as cheek, forehead, and nose.

When the system searches through the images, all the located dark and bright invariant regions are passing into a Kohonen Feature Map and generate two network weight topologies that are used as A-units for a perceptron. Then, their deformable template is generated by the tuning the weights of the two network topologies into array indexes. This technique claimed to be yielding high detection rate compare to learning based algorithm such as Neural Networks and yet does not required extensive sample training or preparing large training sample. (Balasuriya & Kodikara, 2001)

Besides deformable template face detections, there are many other approaches such as face detection based on nonlinear color transformation and locate region of face from the image, then all the candidates are verified through eyes, mouth and boundary map (Hsu, Abdel-Mottaleb, & Jain, 2002). With the nonlinear color transformation technique, this approach is able to detect dark skin-tone face as well as light skin-tone face. The approach might fail to detect the face if the eyes and mouth are not visible and occluded in the image.

Some of the heuristic techniques and frameworks implemented in face detection might not be feasible if apply to solve other object detection problems such as text detection. Edge has always being one of the common methods used in solving text detection problems, as edge detection provides information about the connected curves and pixels which indicating the boundaries of an object (text/characters). Few papers presented the implementation of edge detector in locating text from image, Canny edge detection (Shivakumara, Huang, & Tan, 2008), Sobel edge detection on color image in (Ezaki, Bulacu, & Schomaker, 2004),

edge map generated using Sobel edge detector (Zhu, Ouyang, Gao, Chen, & Xiong, 2009), multi-scale edge detector (Liu & Samarabandu, 2006), multi-resolution edge detector by (Gao, Yang, Zhang, & Waibel, 2001).

In (Shivakumara, Huang, & Tan, 2008) Shivakumara et al. assumed that the text is appeared to be in the horizontal direction with uniform spacing between words. Firstly they dividing an image into 16 equal sized blocks, and each text blocks candidates is selected based on the subtraction of Canny edge block from the Sobel edge block to determine whether the block is text or non text candidates. Then, each of the text block candidates is segmented and further filtered using edge elimination. Their proposed approach achieves detection rate of 89.5% on ICDAR 2005 text dataset. However, there still some limitations where the boundaries specifying the text region might be inaccurate labeled in complex background.

Gao et al. has shown an approach of text detection and translation from natural scenes on Chinese signs using adaptive searching method. Instead of searching through the whole image, their approach start with obtaining text candidates regions using multi-resolution edge detection algorithm, and then search through the candidates neighborhood for any other possible text candidates. Further on, the system will analyze layout of the candidates to align the characters together and later the content of the text will be recognized using OCR engine. (Gao, Yang, Zhang, & Waibel, 2001)

License plate detection is another two class problem that is similar to text detection, where there are many approaches implementing techniques such as edge detection to extract candidate regions that might consists of text, after that through analyzing on the connected components of the edges and construct possible license plate candidates from the image. (Saha, Basu, Nasipuri, & Basu, 2009) (Nguyen, Ardabilian, & Chen, 2008) Other techniques using learning based algorithm to detect license plate such as (Zhang, Jia, He, & Wu, 2006) (Chen, Hsieh, Wu, & Chen, 2006) generates classifiers that able to distinguish between license plate and non license plate through learning on the training image samples.

In (Nguyen, Ardabilian, & Chen, 2008) , they presented a hybrid approach of combining edge and line segments information to detect license plate. Instead of using color information, they used intensity based image to reduce the illumination changes. During locating candidate license plate, they included steps like vertical edges detection, line detection, and grouping perceptual lines together. After that, they will further verify the license plate through geometrical and statistical checking to remove false detection.

Another review of license plate detection using edge information is the approach proposed by Saha et al. (Saha, Basu, Nasipuri, & Basu, 2009). They had proposed approaches for detecting license plate that are not formalized in India. Their system started with preprocessing of grayscale conversion, median filtering and contrast enhancement to sharpen the image and then use Sobel edge operator to extract edge from the image. Based on the distribution of vertical edge pixels,

license plate candidates are extracted. After that, prominent vertical edges will be applied to further verify and filter noises.

2.6 Adaptive Boosting

Adaboost stands for *adaptive boosting*, a machine learning technique introduced by (Freund & Schapire, 1995). It is used primarily to boost the classification performance of a simple algorithm (e.g. a simple perceptron) by combining collection of weak classifiers to form one stronger classifier. A weak classifier means any simple classifier that delivers performance slightly better than chance and preferably with low computation time requirement. Freund and Schapire discovered that a *committee* of weak classifiers when combined properly often outperforms strong classifiers such as Support Vector Machines (SVM) and Neural Networks. Boosting algorithm comes with many variants such as Discrete Adaboost, Real Adaboost, and Gentle Adaboost (Freund & Schapire, 1995). During the learning of classifier, instead of providing large sample, Freund and Schapire has showed a ‘boosting by sampling’, selecting a small samples from the large sample through uniform distribution. Adaboost produced the final hypothesis based on all the accuracy of all hypotheses from each weak classifier.

2.6.1 Adaptive Boosting for Face Detection

Viola and Jones presented a framework for face detection that achieves high detection rate and yet with extremely rapid image processing speed (Viola & Jones, 2004). Motivated by Papageorgiou and Poggio (Papageorgiou & Poggio, A

Trainable System for Object Detection, 2000) (Papageorgiou & Poggio, A Trainable Object Detection System: Car Detection in Static Images, 1999), Viola and Jones introduced a new image representative known as the integral image that allows the features used in the detection to be calculated very rapidly. They introduced Haar-like features, reminiscent of Haar wavelet used by Papageorgiou and Poggio to classify the patterns for an image. Then, they use Adaboost learning algorithm to select a small simple Haar-like features from the over-complete features set. In Viola and Jones face detection framework, each feature is known as a weak classifier, and the weak classifiers will be combined to become a strong classifier. Through number of iterations during the training, all strong classifiers are cascaded to increase the speed of detection. The face detector is able to process an image with the size of 384 x 288 in about 0.067 seconds and achieves detection rate of 94.1 percent with 422 false detections. (Viola & Jones, 2004)

- Given example image $(x_1, y_1), \dots, (x_n, y_n)$, where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialise weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negative and positive respectively.
- For $t = 1, \dots, T$:
 1. For each feature, j , choose a classifier, h_t by minimizing a weighted squared error

$$\varepsilon_j = \sum_i w_i |h_j(x_i) - y_i|$$
 2. Update the classifier $F(x) \leftarrow F(x) + h_t(x)$
 3. Update weights by $w_i \leftarrow w_i e^{-y_i h_t(x_i)}$
- Final strong classifier is $F(x) = \text{sgn}\left(\sum_{i=1}^T \alpha_i \bullet h_i(x) + b\right)$

[Note: T is the number of weak classifiers. The final strong classifier is a weighted linear combination of the T weak classifiers $h_i(x)$ with biased offset b .]

Figure 2.7 Gentle Adaboost procedure used to construct a strong classifier.
Adapted from: (Viola & Jones, 2004). Robust Real-Time Face Detection.
International Journal of Computer Vision 57(2), pg 142.

Haar-like features are *overcomplete* in certain sense because for an associated window, the number of rectangle features is far larger than the number of pixels, for instance a 24x24 window has 45,936 possible Haar-like rectangular features (Viola & Jones, 2004) compared to 576 (24x24) pixels. Even though these rectangle features can be calculated efficiently using integral image, Viola and Jones postulated that only small number of these features is useful for classification purpose. The challenge now is to find these features. Note that, boosting algorithms only differ in the procedure on how to re-weight training

examples after the training iteration. To boost the performance of a strong classifier, AdaBoost algorithm search over a pool of weak classifiers to find one with the lowest classification error for the subsequent combination. This learning method is also known as greedy feature selection process.

While training a classifier, it is called upon to classify training examples so that these examples can be re-weighted to emphasize those which were incorrectly classified for the next training iteration. After training, we have a weighted combination of weak classifiers in the form of perceptrons and a simple binary threshold value determined automatically. Every weak classifier has an associated weight where good classifiers assigned larger weight while poorer classifiers assigned smaller weight. Figure 2.7 shows the learning algorithm (Viola & Jones, 2003); Figure 2.8 and Figure 2.9 shows the Haar-like features that are used in their framework and the cascade of weak classifiers selected by the Adaboost learning algorithm.

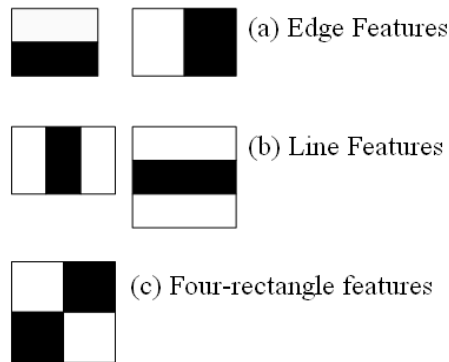


Figure 2.8 Simple Haar-like basis functions used as features in license plate detection scheme. Adapted from: (Viola & Jones, 2004). Robust Real-Time Face Detection. *International Journal of Computer Vision* 57(2), pg 139.

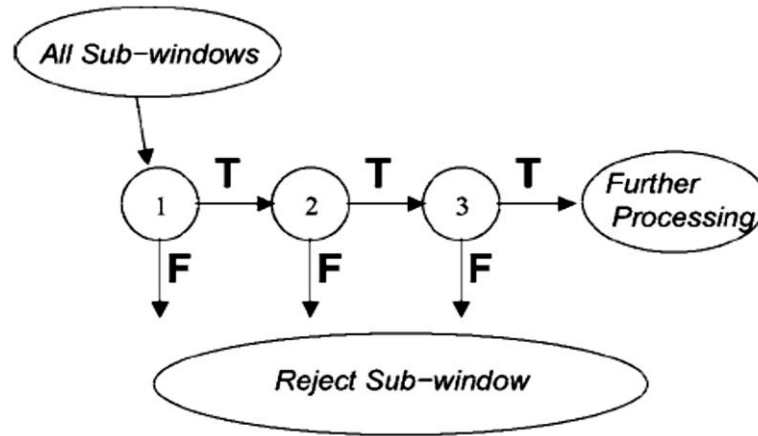


Figure 2.9 Diagram depicts of Cascade of a set of Weak Classifiers Selected by Adaboost Learning Algorithm Adapted from: Viola (Viola & Jones, 2004). Robust Real-Time Face Detection. *International Journal of Computer Vision* 57(2), pg 144.

2.6.2 Adaptive Boosting for License Plate Detection

There are many approaches and techniques in solving license plate detection, one of approaches is applying global and local features by (Zhang, Jia, He, & Wu, 2006). Adaboost learning algorithm with Haar-like feature set has achieved high detection rate in face detection (Viola & Jones, 2004), after that, many researches and extension of works from the Adaboost learning framework. One of the further extensions will be (Zhang, Jia, He, & Wu, 2006) of their learning based license plate detection using global and local features. Their global features are constructed by the gradient density and density variance. Gradient density is used in their paper as they claimed that license plates tend to have high density and it is easier to implement and obtain information of gradient density rather than edge. After that, they found that the distribution of gradient density in the area of license plate is equally even comparing to background. Therefore, they implemented density invariance as their second global feature to distinguish

between license plate and non license plate. They added these two global features at the first two stages and able to remove 70% of the background image while the local Haar-like features make their classifier invariant to brightness, color, size, and position of the license plate.



Figure 2. 10 Diagram shows the distribution of gradient density in the area of (a) license plate, and (b) Background image. Adapted from: (Zhang, Jia, He, & Wu, 2006). Learning-Based License Plate Detection Using Global and Local Features. *18th International Conference on Pattern Recognition (ICPR'06)*, pg 1102-1105.

Chen et al. (Chen, Hsieh, Wu, & Chen, 2006) presents a hybrid method combining morphological operations and Adaboost algorithm for detecting license plate. As license plate is having high contrast between the characters and the background. They apply a morphological feature extraction to determine the high contrast area from the image and eliminate a lot of non-license plate region which will significantly reduce the computation for large amount of possible candidates. In their paper, they categorized license plate with characters horizontally proper aligned as normal license plate, while those without horizontal aligned is known as inclined in (Chen, Hsieh, Wu, & Chen, 2006). Adaboost is able to trace the normal license plate and failed to detect those inclined license plate. In order to solve this problem, morphological operation is implemented. Every candidate extracted using morphological operation will be rectified to a

proper pattern and verify whether it is a license plate or non license plate using Adaboost algorithm.

2.6.3 Adaptive Boosting for Text Detection

Chen and Yuille (Chen & Yuille, 2004) demonstrated an algorithm for detecting text in natural images. They used Adaboost learning algorithm to train a strong classifier, but they claimed that the set of features used for face detection by Viola and Jones (Viola & Jones, 2004) might not suitable for detecting text. This is because there is less spatial similarity for the text compared to face; a face can be regarded as spatially similar object since it consists features such as eyes, nose, and mouth, which are approximately the same relative spatial position for any face. They did PCA analysis on the text and the result shows faces have less non-zero Eigen values than text. Therefore, they provide informative features which give similar results for all text. The informative features are the features which are good in discriminating between text and non-text and Figure 2.11 shows some of the features that are used to detect text in natural scenes. Then, they implemented Adaboost learning algorithm to perform the feature selection and cascade all the strong classifiers to make the detection faster.

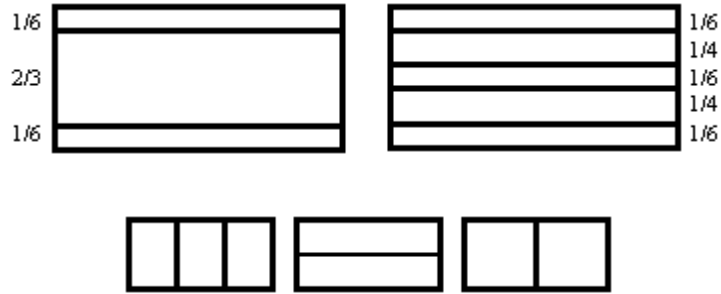


Figure 2.11 Features that compute properties average within the region. Adapted from: (Chen & Yuille, 2004). *Detecting and Reading Text in Natural Scenes*. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04), pg 3.

In (Zhu, et al., 2005), they proposed a nonlinear NiBlack Method (NLNiBlack) to decompose the image into a candidates Connected Components (CC) based and applied 12 distinguish features to differentiate between text and non text. After that, the implementation of Adaboost as a machine learning technique in this system to automatically setting threshold for each features and cascaded them in a form for fast computation. The 12 features proposed in the system are geometric feature (area ratio, length ratio, and aspect ratio), edge contrast feature, shape regularity feature (holes, contour roughness, compactness and occupy ratio), Stroke Statistics Feature (Mean Stroke Width, Normalized Stroke Deviation), and Spatial Coherence Features (area ratio, boundary touching).

Each feature in the system acted differently, for example, 3 geometrical features (area ratio, length ratio, and aspect ratio) are the common simple features which can be fast to remove large non-text regions. This is an advantage for the system, as the execution time decrease when the system already removing those

large non-text region at the beginning of the processing. Then, edge contrast feature is the most essential feature in the whole algorithm, as edge comprises significant information of text. While Sharp Regularity features used to remove noises with unequal shape but strong texture information. After that, 3 Stroke Statistics features acted as the Character Stroke Checker, checking the irregular character strokes of the connected components from the previous classifier. Finally the two Spatial Coherence features are applied to remove noises with less spatial coherence ratio. All the features implemented in (Zhu, et al., 2005) is playing different role as some of the features contain the texture information about text while some features acting as the noise removal.

2.7 Adaptive Boosting & Evolutionary Feature Search

As an extend of (Viola & Jones, 2004), Treptow & Zell present an evolutionary algorithm that is applied into Adaboost to search for new features. In (Viola & Jones, 2004), exhaustive feature search is time consuming for large feature set, so in (Treptow & Zell, 2004), they improve the approach with addition of evolutionary algorithm to perform feature searching, and their approach is prominence to real time processing as they will implement the object detector to the robots to locate and trace any targeted object. The improvement they had introduced in their approach is through the modification of existing Haar-like features set from (Viola & Jones, 2004) using crossover, and mutation. In addition, they employed a new feature type with two regions of interest to the existing feature set. This newly added feature type is claimed to detect regions

with non symmetric dependencies. From their paper, they had show the comparison of training time for Adaboost learning exhaustive search and their Adaboost learning with evolutionary search. Significant difference is observed, where Adaboost learning with evolutionary search is having shorter training time compare to exhaustive search. Meanwhile, this approach is able to achieve detection rate of 96.9% compare to detection rate of 96.1% by the classifier selected using Adaboost learning with exhaustive search.

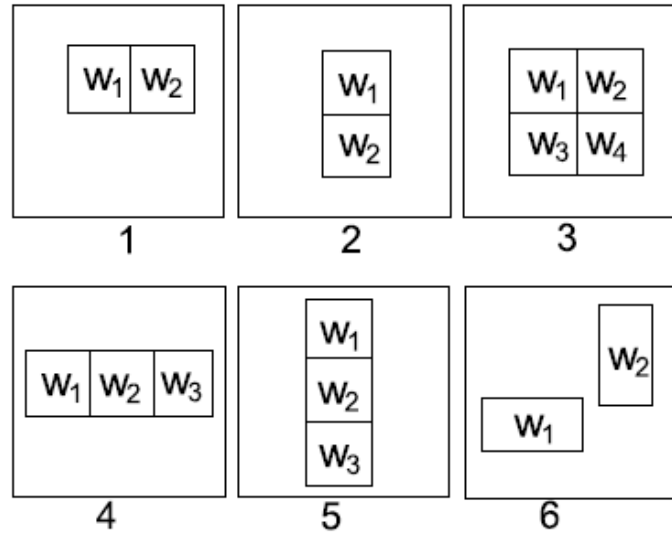


Figure 2.12 Six Feature type with different geometrical layouts. Adapted from: (Treptow & Zell, 2004) Figure showing the 6 feature types that are implemented in Treptow & Zell Adaboost learning algorithm with Evolutionary Search.

2.8 Control Points Based Object Detection

Another approach based on Adaboost learning algorithm is introduced by (Abramson, Moutarde, Steux, & Stanculescu, 2006). However, this approach does not apply nor adding new feature to the existing Haar-like feature

implemented in (Viola & Jones, 2004). Yet, they bring in the idea of Control Point as feature set in solving object detection problem. This idea was first proposed in (Abramson & Steux, 2005). This feature is based on the statistical information of individual pixel value and how relation between pixels value response. It does not need to depend on the difference of the pixels (Viola & Jones, 2004) and also insensitive to any histogram equalization, where it is affected only by the *sign* of the changes among the two groups of pixels. Each point for the feature is actually the one pixel values of the given image. The feature will say the scanning window as a target object if and only if every points in the first group is larger than every points in the second group. However, it might not be easy to determine which points should be in first group and which points should be in the second. Therefore, similar to (Treptow & Zell, 2004), (Abramson, Moutarde, Steux, & Stanciulescu, 2006) they implemented a Genetic-like algorithm to choose the best feature, then use Adaboost to select the weak classifier and updating the sample weights for next round of training.

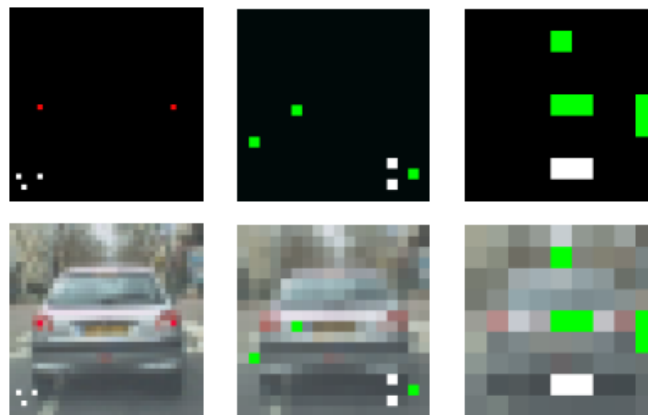


Figure 2.13 Adapted from: (Abramson, Moutarde, Steux, & Stanciulescu, 2006). Figure showing the control points feature implemented in (Abramson, Moutarde, Steux, & Stanciulescu, 2006).

Figure 2.13 shows the control points features implemented by (Abramson, Moutarde, Steux, & Stanciulescu, 2006) . The first row is depicting the control points features of different scales, while the second row is showing the images extracted from the sub-scanning window of the video frames. The left upper corner is the full resolution (36x36) of the control points feature where 3 white colour points are the negative points and 2 red colour points are the positive points. The middle upper shows the detection window that is scaled to 18x18 which is half of the full resolution. It has 3 positive points with green colour and 3 negative points in white. Then, at the right upper corner, this is the detection window that is scaled to 9 x 9, half of the resolution of the middle detection window. This feature is having 5 positive green points and 2 negative points in white.

2.9 Spatial Histogram Based Object Detection

Comparing to (Viola & Jones, 2004) in their face detection frameworks that implementing Adaboost learning algorithm with Haar-like feature, in (Zhang, Gao, Chen, & Zhao, 2005) they had implemented another type of informative feature called Spatial Histogram. Spatial Histogram consists of marginal distribution of image over local area and information about the texture and shape of the object. Instead of using a set of Spatial Histogram feature which are predefined for the object detection, they improve their previous system by implementing Fisher Criterion to analyze the discriminative of each Spatial Histogram during the training, and select a set of the Spatial Histogram features

based on their correlation information. Finally all the selected Spatial Histogram features are arranged in a cascade form for histogram matching. This cascade histogram matching is similar to the cascade of weak classifiers in (Viola & Jones, 2004) where it can obtain high detection rate while removing many false detection. However, the false positive rate still high after the cascade of histogram matching, so, (Zhang, Gao, Chen, & Zhao, 2005) they employ Support Vector Machine (SVM) as a classifier that will remove the false positive. With this framework architecture, they had applied to face, car, and text detection. They obtained detection rate of 96.5% and precision rate of 81.10% for UIUC car test set, after that they also trained a text detector using the same approach and attained detection rate of 90.63% and precision rate of 87.88% on a video text set.

2.10 Support Vector Machine for Object Detection

Feature selection is one of the important factors in solving object detection problem as stated in (Sun, Bebis, & Miller, 2004). They presented an approach using Genetic Algorithm to select the feature and then using Support Vector Machine to perform further classification on the two class object detection problem. They first preprocessing by normalizing their input image to compensate with noise, illuminations, size, and orientations. They applied Principle Component Analysis (PCA) to extract the eigenvector of the distribution from the training samples as feature for their classification. Since there are many eigenvector available during the training, so they implemented Genetic Algorithm to efficiently determine the good subset of eigenvector for improving detection

rate. Similar to (Oh, Lee, & Moon, 2002) (Treptow & Zell, 2004), Genetic algorithm is implemented for feature selection as this approach provides fast and nearly optimal solution in a large search space. While Genetic Algorithm is responsible for feature selection, Support Vector Machine is applied here as a classification mechanism to distinguish a set of points belongs to which class of object. Support Vector Machine will determine the hyperplane with the maximum number of points of the same class at one side and maximizing the distance of the other class from the hyperplane. In (Sun, Bebis, & Miller, 2004), they had tested their approach on rear-view vehicle detection from gray-scale image and frontal-view face detection. With the Genetic algorithm approach, the performance for both vehicle and face detection improved compare to SBFS approach stated in (Sun, Bebis, & Miller, 2004), where SBFS is the sequential backward floating search for feature selection heuristically.

2.11 Support Vector Machine in Text Detection

In (Kim, Jung, & Kim, 2003), they implemented a technique combining continuously adaptive mean shift algorithm (CAMSHIFT) and Support Vector Machine for texture-based Text Detection. In their system, they did not apply any texture feature extractions methods, but using raw pixels of the images and passed to SVM for determining the textural properties of the text and non text. The advantage of the system is providing fast processing as it does not required feature extraction. Firstly, SVM is used as a classification stage that transforms the input image into a representative and CAMSHIFT is proposed to locate the

text region from the classification result of SVM. Yet, there are still weaknesses in this system, as it failed to detect text region that is too small or low contrast.

Besides that, there are other studies in locating text in complex background using Support Vector Machine (Chen, Bourlard, & Thiran, 2001). Chen et al. analyze the image with edge-based method and extract the text regions after that segment them into lines and it then further processed into feature vector using distance map feature space and feed into Support Vector Machine to perform text and non-text classification. They applying Canny Edge detection to extract edge information of the image and using morphological dilation to combine the edges into groups. There are many text detection techniques implementing Edge – based method to extract the edges information of text regions in the image. For examples, (Liu & Samarabandu, 2006) presented multi-scale edge based text extraction from complex images and (Zhu, Ouyang, Gao, Chen, & Xiong, 2009) implementing four Sobel operator to extract edges information of the image and converted into feature vectors and classify whether the given region is text or non-text using Support Vector Machine.

2.12 Genetic Algorithm for Feature Selection

In this project, we selected genetic algorithm for the feature selection rather than the AdaBoost learning framework. We had chosen the genetic algorithm because the evolutionary searching that provided by the genetic algorithm inspired me on the possibilities to search for the best features. Compare to the Adaboost implemented in (Viola & Jones, 2004), they search through the

large features set that predefined in the framework before the training process started. By doing so, it would be time consuming for the feature selection, therefore, in order for me to improve on the feature selection process, I picked the genetic algorithm for its simplicity in implementation and the evolutionary characteristics of genetic algorithm enable it to evolve the features based the number of learning iterations on the samples.

From (Treptow & Zell, 2004), they combined evolutionary algorithm and Adaboost to search for new features in their system proven to be efficient and effective compare to solely using Adaboost for the learning framework. In our project we did not implement both evolutionary algorithm and Adaboost together into our system because I want to make the developments and implementation simple and easy to be applied for future works.

2.13 Summary

There are many techniques and methods in license plate detection and learning framework, such as the hybrid of license plate extraction method based on edge statistic and morphological techniques (Bai & Liu, 2004), Zhang and Zhang (Zhang & Zhang, 2010) with their multi-feature such as mathematical morphology, rectangle features, edge statistics and character features, (A. Ravi, Shashank, Abhishek, & Kandavli, 2010) detecting license plate with the help of lines generated from contours, (Kim, Han, & Hahn, 2008) with their edge detection and closed loop detection, (Zhang, Jia, He, & Wu, 2006) with their adaptive boosting based license plate detector and etc. From each technique, we

get inspirations from the features type and methods that they used to solve license plate detection problem. Obviously, edge information becomes essential in extracting important features of the license plate, and this inspired us to further design our feature type in this research. Then, from the literature, we studies on some approaches for designing our learning framework. Papageorgiou and Poggio have presented a trainable and universal system for object detection in cluttered environments without changing the general architecture (Papageorgiou & Poggio, A Trainable System for Object Detection, 2000). After that, motivated by their works, Viola and Jones presented a framework for face detection that achieves high detection rate and yet with extremely rapid image processing speed (Viola & Jones, 2004); Then, more researches and analysis extended of Viola and Jones works in face detection, for example, in (Chen & Yuille, 2004), they modified on the feature type and added statistical features to solve text detection problem. Subsequently, (Zhang, Jia, He, & Wu, 2006) added global feature into existing Adaboost framework with Haar-like feature as local feature and tested on license plate detection problem. Instead of focusing on adding new features, rather, (Treptow & Zell, 2004) implemented Evolutionary Search within Adaboost face detection framework by (Viola & Jones, 2004) as well as added one extra feature type to the existing 5 Haar-like feature set. Their approach is able to speed up the training time while attaining high detection rate. While these researches making improvements in changing the feature type, feature searching approach, another feature type is introduced in (Abramson, Moutarde, Steux, & Stanculescu, 2006) called control points feature. This type of feature does not depend on the

difference of pixels summated from different regions, but is depends on how each pixels sign different between two groups of pixels. Further on, spatial histogram is introduced by (Zhang, Gao, Chen, & Zhao, 2005), informative feature type that is implemented in face, car, text detection.

CHAPTER THREE

SYSTEM DESIGN AND IMPLEMENTATION

3.1 System Frameworks & Architecture

In this project, we have developed an object detection system framework consists of two stages. Here in this chapter we shall further describe the entire system framework and the learning algorithm. There are two stages in our system framework; The First stage is the *training* stage, and the next stage is the *testing* stage. The Training stage here is defined as the stage where we generate an object detector through an iteration of learning process on the training image sample prepared before the learning started. There are six major components in the training stage, which are

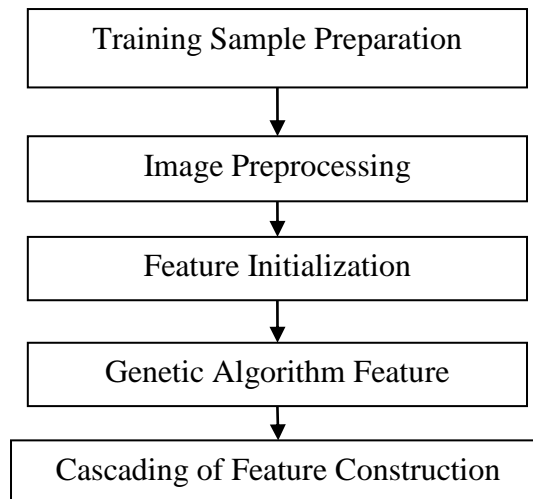


Figure 3.1 System Architecture of the Training Stage in License Plate Detection Learning Framework

Each of these modules has different role in constructing the final classifier, and is started with Training Sample Preparation module. This module is to collect images consist of prior knowledge for the learning process. Training images are categorized into two groups, positive samples, and negative samples. Positive sample is the group of images consists of the target object of interest; while negative sample is the group of images consists of non-target object of interest or we called them as the background images.

The next component after preparing the training sample is Image Preprocessing module. This module consists of a few components, i.e. Image Grayscale Conversion, and Integral Horizontal Edge Variation Calculation. In this module, images are first converted from color images to grayscale images. After that, Integral Horizontal Edge Variation Image is calculated as an image representative for fast processing. In this project, we introduced Spatial Horizontal Edge Variation feature to find the changes of pixel's sign when crossing from left to right of the image horizontally. Further explanations of the Spatial Horizontal Edge Variation Feature are described in next few sections.

Then, the system started the training with feature initialization module. Inspired by (Zhang, Gao, Chen, & Zhao, 2005), they applied Spatial Histogram to perform face, car and text detection, so in this project, we modify the feature to become a new feature type in our license plate detection framework; we called this feature type as Spatial Horizontal Edge Variation feature. From our research and studies on the previous work on Adaboost Framework in Face Detection by (Viola & Jones, 2004), we found that the feature sets that implemented for the

face detection in their works are able to provide optimal results for objects with spatially similarity, and we was not be able to obtain a high detection rate when we try to implements this learning frameworks and features set to our license plate detection problems. Therefore, from our studies on literature reviews, there are many others ways to solve the license plate detection problems, and different kind of features were used such as color, shapes, connected components analysis on the alphanumeric text and characters, customized Haar-like features to search for the license plate regions and etc. From there, Malaysian vehicle license plate is constructed with multiple alphanumeric and each of the characters combined and become one license plate. All of these alphanumeric characters are placed on top of the black color background plate that actually provides a contrast in between the plate and the numbers. Therefore, this give us an inspiration of creating features that can extract the region with highly contrast. However, if we defined a static feature template to find the regions with high contrast will be providing us a lot of false detection as the possibility to get regions with high contrast from an image is very high. We found that the changes of the pixels from the starting characters to the end of the number plates provide similar patterns among the samples. But one feature might not able to give a good result to much different kind of license plate images as it might be able to detect only some of license plate. Then, from there we start to design some of the features that might be good in detecting the license plate regions, but the question is how many features do we need to manually design and how to manually determine whether which features are *good*? Linking with all these issues, we come out with the

Genetic algorithms to help us do the searching and generating the each features thresholds to make decision between license plate and non license plate.

After the initialization of the features, the system starts to select the best features from the feature set. Due to the reason that the feature set is relatively large, we implemented Genetic Algorithm as the feature selection method to search for the best features and expand the feature search space. We did not implement Adaboost for our feature selection is because genetic algorithm is simpler to implement and easier to tune the training parameters. While Adaboost is to generating the list of sequence classifiers that learned and adjust based on the training samples that misclassified by the previous classifiers. In our projects, we picked Genetic Algorithm as it helps to solve the optimization and searching for the best solution for the problem, and our hypotheses on this project that we able to implement Genetic Algorithm to search for the best features with the optimal training time. Genetic algorithm applied in this project consists of three main parts, Selection, Crossover, and Mutation. Further explanations these three parts are described in next few sections.

In each generation of the Genetic Algorithm, we select one of the features with the minimum errors and combined it into the list of features in a linear structure. Motivated by (Viola & Jones, 2004), they combine their weak classifiers into a cascading structure and becoming a strong classifier to perform the face detection. Here, we applied a similar technique to combine the selected features into a linearly cascading structure for fast processing. Using this method,

the processing time can be reduced as many false images are removed by the initial features in the cascaded feature during the scanning.

However, there is still a lot of false detection from the output of our Cascaded features; Therefore, Support Vector Machine is applied for training and generating classifier to remove false alarms detected from the output of cascaded features. Further explanations on Support Vector Machine technique are described in next few sections. After generating cascaded features and Support Vector Machine classifier in *training* stage, we can further apply our object detector through the processes in *testing* stage. There are six modules in the testing stage, which are

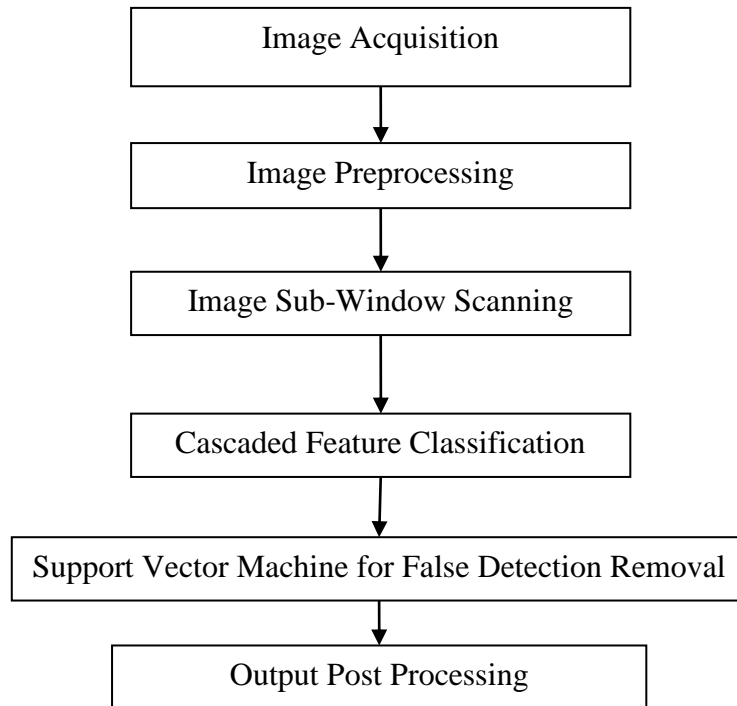


Figure 3.2 System Architecture of the Testing Stage in License Plate Detection Learning Framework

Image Acquisition is the first module in testing stage, as the system will obtain an image from the user to further process it and locate the target object using the trained classifiers. In this project, we tested on images with size of 384 widths and 288 heights, as the increase of the size for the image will increase the processing time. This is because when the size of image increase, the number of sub-window scanning increase which increase the computation time.

After that, the original color image is converted to grayscale and the system will generate the integral Horizontal Edge Variation image. With this image representative, it can reduce the time of computing the feature values for every scanning window.

In this project, we implemented the approach of window scanning method to finds the target object. The system starts with locating the license plate from the image by scanning the image with a rectangle-like window that is defined during the training stage. The window moved with a step size of single pixel from the left to the right and from the top to bottom of the image. At each position, the window will be processed by the cascaded feature to detect any license plate within the window, and Support Vector Machine classifiers are included in this project to further remove the false detection. Then the scanning window size will increase by a *scale factor* and continue the same process whereby the purpose of increasing the size of the scanning window is to locate larger license plates.

In this project, the learning framework selects a list of features and combines them in linear structures to perform the classification. Each scanning window is processed by the list of the selected features linearly, whereby, the

features will determine whether the window containing any license plate or not, if the feature classify it as a possible candidate, then it will proceed with the following features in the linear structure. This process will continue until the final feature in the list classify that the window is a license plate candidate, then the algorithm will assume that this region containing license plate. However, if any of the features classify the window as a non license plate candidate, then the algorithm will not continue to process the window with the following features in the list, and move the window to the next position for processing. Here in this project we named the feature in the linear structure as cascaded features.

All the output from the cascaded features is categorized as the possible candidates that containing license plate. Therefore, the system will further verify the window using Support Vector Machine classifier. This module is implemented in order to remove those false detections from the cascaded features.

Finally, all the results generated from the previous modules is gathered and feed to the final module for post processing. This module removed any duplicated windows or overlapping windows, and labels the final results with rectangles bounding the target object. Details for each module will be discussed in coming next sections, and figure 3.22 shows the summary of the system architecture for *training* stage of the object detection framework in this project, and figure 3.23 shows the summary of the system architecture for *testing* stage.

3.2 Image Representative – Integral Horizontal Edge Variation Image

Viola and Jones implemented Integral Image for their face detection framework as a way to improve the processing time by reducing the computation of the feature values (Viola & Jones, 2004). Inspired by their technique, we implement the Integral Horizontal Edge Variation Image for our algorithm for faster processing time. Firstly, we calculate the Horizontal Edge Variation of the image (detailed explanations of Horizontal Edge Variation feature is described in section 3.4). After that we generate the Integral image from the Horizontal Edge Variation image, and all the features extraction is based on the integral image.

3.3 Generate Horizontal Edge Variation Image

First of all we pre-process the image by converting the original image to grayscale image. After that, we generate the Horizontal Edge Variation image from the grayscale image by convoluting the image from the first pixel start with coordinate(0,0), and compare the first pixel with the second pixel of its right(1,0). A heuristic constant value $threshold_{pc}$ is set for this Horizontal Edge Variation image generation. If the subtraction of the two pixels at (x,y) and $(x+1,y)$ is more than the $threshold_{HEV}$, then the Horizontal Edge Variation image at position (x,y) is assign with a constant value of 255 value to indicate there is difference of the two pixels value.

$$\begin{aligned}
 &Img_{HEV}(x,y) \\
 &= \begin{cases} 255, & |Img_g(x,y) - Img_g(x+1,y)| \geq threshold_{HEV} \\ 0, & otherwise \end{cases} \quad (3.1)
 \end{aligned}$$

Where $Img_{HEV}(x, y)$ is the Horizontal Edge Variation image at coordinate (x, y) , $Img_g(x, y)$ is the grayscale image at coordinate (x, y) , and $threshold_{pc}$ is the heuristic constant value.

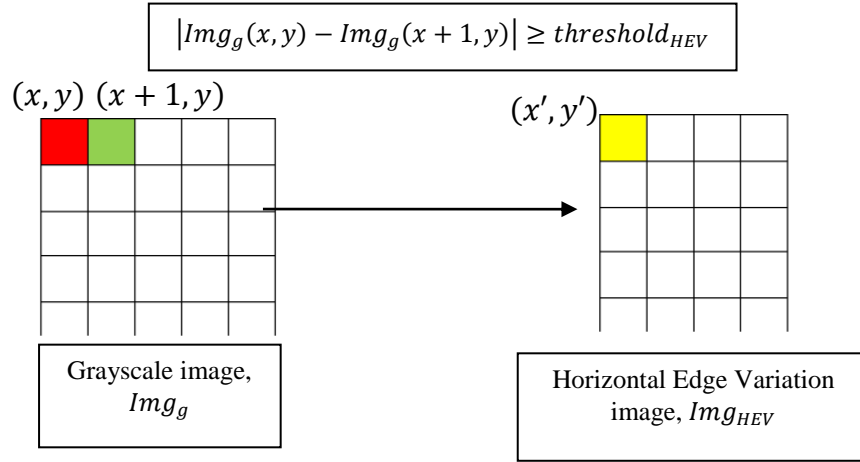


Figure 3. 3 Ways to calculate Horizontal Edge Variation image Img_{HEV} from Original Grayscale image, Img_g .

3.3.1 Calculate Integral Horizontal Edge Variation Image

Viola and Jones (Viola & Jones, 2004) introduced integral image as an image representation which provide fast processing. Integral image at coordinate (x, y) is summation of the pixels above and to the left of (x, y) shown as below:

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \quad (3.2)$$

Where $ii(x, y)$ is the integral image and $i(x, y)$ is the original image. (Viola & Jones, 2004)

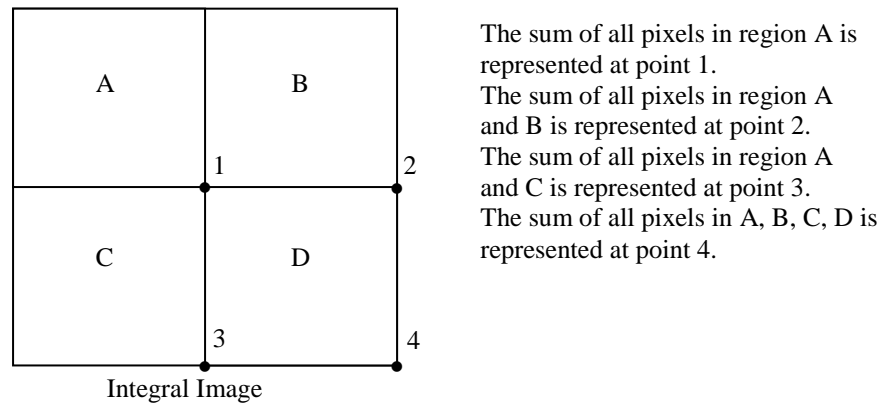
To compute the integral image in one pass over the original image, we can use the formulae below:

$$s(x, y) = s(x, y - 1) + i(x, y) \quad (3.3)$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \quad (3.4)$$

Where $s(x, y)$ is the cumulative row sum, $s(x, y - 1) = 0$, and $ii(x, y) = 0$.

Figure 3.4 shows the formulae to calculate the sum of pixels for any region within the sub-window.



- Therefore, the Sum of Pixels in Region D,

$$Sum_D = Point_4 - Point_3 - Point_2 + Point_1$$

Figure 3.4 Adapted from (Viola & Jones, 2004). Diagram shows how to compute the sum of all pixels in any regions of the integral image.

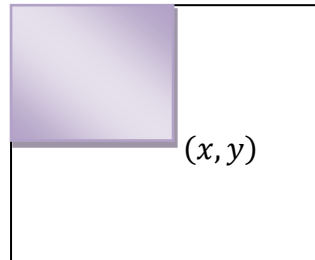


Figure 3.5 Integral Image. Adapted from (Viola & Jones, 2004) The value of the integral image at point (x, y) is the sum of pixels above and to the left.

3.4 Feature Extraction

In this project, we introduce the newly feature type named Spatial Horizontal Edge Variation Feature. In the following sections, we will explain on the details of each feature and techniques to compute the features values for this feature type.

3.4.1 Spatial Horizontal Edge Variation

For our research in detecting license plate, we proposed a feature type called Spatial Horizontal Edge Variation feature for detecting the changes of pixels across the image horizontally. This feature is similar to edge detection and gradient filter, as the feature shows the pattern distributions of how the pixels changed from left to right at the horizontal line. Figure 3.6, showing the Spatial Horizontal Edge Variation feature with three regions. Each region denotes a value represents the average of the total of pixels changes from left to right of the image horizontally. However, Spatial Horizontal Edge Variation feature does not limit to three regions, the *number_of_region* can be varies, further experiments will be conducted to examine the effect of *number_of_region* in overall performance.

$$rect'(x', y') = \begin{cases} 255, & |rect(x', y') - rect(x' + 1, y')| \geq threshold_{HEV} \\ 0, & otherwise \end{cases} \quad (3.5)$$

Where, $rect(x', y')$ is the rectangle within the sub-window with the coordinate of x' and y' , $rect'(x', y')$ is the spatial horizontal edge variation rectangle within the sub-window with the coordinate of x' and y' , $threshold_{HEV}$ is a heuristic constant value we manually set to determine a boundary for separating two groups of values. After that, for each horizontal line, we accumulate the total sign of the pixel changes.

$$RowSum(y') = \sum_{i=x'}^{x'+w-1} rect'(i, y') \quad (3.6)$$

Where $RowSum(y')$ is the summation of the total sign of the pixel changes for row y' , given $y \leq y' < y + h$, x' and y' are the coordinate x and coordinate y respectively within the rectangle of the sub-window. w is the width of the rectangle of the sub-window and $rect'(x', y')$ is the spatial horizontal edge variation rectangle within the sub-window with the coordinate of x' and y' .

Then, for each Spatial Horizontal Edge Variation region, we compute the summation for each region of the horizontal edge variation rectangle $rect'(x', y')$.

$$RegionSum(r) = \sum_{i=r \times number_row_in_region}^{number_row_in_region} RowSum(i) \quad (3.7)$$

$$number_row_in_region = \frac{h}{number_of_region} \quad (3.8)$$

Where $RegionSum(r)$ is the summation of each row $RowSum$ of the region r .

Given that $0 \leq r < number_of_region$

$$RegionAverage(r) = \frac{RegionSum(r)}{number_row_in_region} \quad (3.9)$$

Where $RegionAverage(r)$ is the average of each region in the horizontal edge variation rectangle feature. Given that $0 \leq r < number_of_region$.

3.4.2 Spatial Horizontal Edge Variation Feature Value Extraction

The feature values for each Spatial Horizontal Edge Variation feature are denoted as below:

$$SpHEV_{rect(x,y,w,h)}RegionAverage(r) \quad (3.10)$$

Where $SpHEV_{rect(x,y,w,h)}RegionAverage(r)$ is the average of pixel changes in each region for the Spatial Horizontal Edge Variation feature with a rectangle, $rect(x,y,w,h)$, where x coordinate, y coordinate, w width, and h height respectively. Given that $0 \leq r < number_of_region$.

3.4.2.1 Calculate Spatial Horizontal Edge Variation Feature Template

For Spatial Horizontal Edge Variation feature, we implement a simple formula to calculate the feature template. For n number of positive image, we calculate the template for the spatial horizontal edge variation feature that will represent the object over the n number of training positive image. The template of

a spatial horizontal edge variation feature is obtained through computing the average of the feature vector over the n number of positive training samples.

$$\widehat{SpHEV_{rect(x,y,w,h)}} = \frac{1}{n} \sum_i^n SpHEV_{rect(x,y,w,h)}(P_j) \quad (3.11)$$

Where $\widehat{SpHEV_{rect(x,y,w,h)}}$ is the feature template for the Spatial Horizontal Edge Variation feature with (x, y) coordinates, w of width, and h of height of the rectangle, $rect(x, y, w, h)$ inside the sub-window, and (P_j) is the positive training sample.

To compare the difference of two horizontal edge variation features, we implemented the Chi-Square distance.

$$\begin{aligned} & D(SpHEV_1, SpHEV_2) \\ &= \sum_i \frac{(SpHEV_1RegionAverage(i) - SpHEV_2RegionAverage(i))^2}{SpHEV_1RegionAverage(i) + SpHEV_2RegionAverage(i)} \end{aligned} \quad (3.12)$$

Where i is the number of region for the spatial horizontal edge variation feature $D(SpHEV_1, SpHEV_2)$ is the distance between two spatial horizontal edge variation $SpHEV$, and $SpHEV_2$.

After that, for any sample (positive or negative) P , we denote its Spatial Horizontal Edge Variation feature $f_{rect(x,y,w,h)}(P)$ as its distance to the feature template,

$$f_{rect(x,y,w,h)}(P) = D(SpHEV_{rect(x,y,w,h)}(P), \widehat{SpHEV_{rect(x,y,w,h)}}) \quad (3.13)$$

3.4.3 Spatial Horizontal Edge Variation Feature Normalization

In this project, we normalized each Spatial Horizontal Edge Variation feature and its template during training and testing stage. Normalization is applied to remove some of the outliers that might affect the performance of the detector, and normalization makes the Spatial Horizontal Edge Variation feature invariant to the sub-window size.

Firstly, we normalize each Spatial Horizontal Edge Variation feature template $\widehat{SpHEV_{rect(x,y,w,h)}}$ with the size of the rectangle within the feature template. Normalization method is shown as below:

$$\begin{aligned} & \widehat{SpHEV_{rect(x,y,w,h)}}RegionAverage(i) \\ &= \frac{\widehat{SpHEV_{rect(x,y,w,h)}}RegionAverage(i)}{w \times h} \end{aligned} \quad (3.14)$$

Where $\widehat{SpHEV_{rect(x,y,w,h)}}$ is the Spatial Horizontal Edge Variation feature template with $rect(x, y, w, h)$, i is the i -th region of the feature, w is the width and h is the height of the rectangle within the feature template.

Then, for each Spatial Horizontal Edge Variation feature $SpHEV_{rect(x,y,w,h)}(P)$, we apply normalization as shown below.

$$\begin{aligned} & SpHEV_{rect(x,y,w,h)}RegionAverage(i)(P) \\ &= \frac{SpHEV_{rect(x,y,w,h)}RegionAverage(i)(P)}{w \times h} \end{aligned} \quad (3.15)$$

Where $SpHEV_{rect(x,y,w,h)}RegionAverage(i)(P)$ is the Spatial Horizontal Edge Variation feature template with $rect(x, y, w, h)$ and P is the image of interest, i is

the i -th region of the feature, w is the width and h is the height of the rectangle within the feature template.

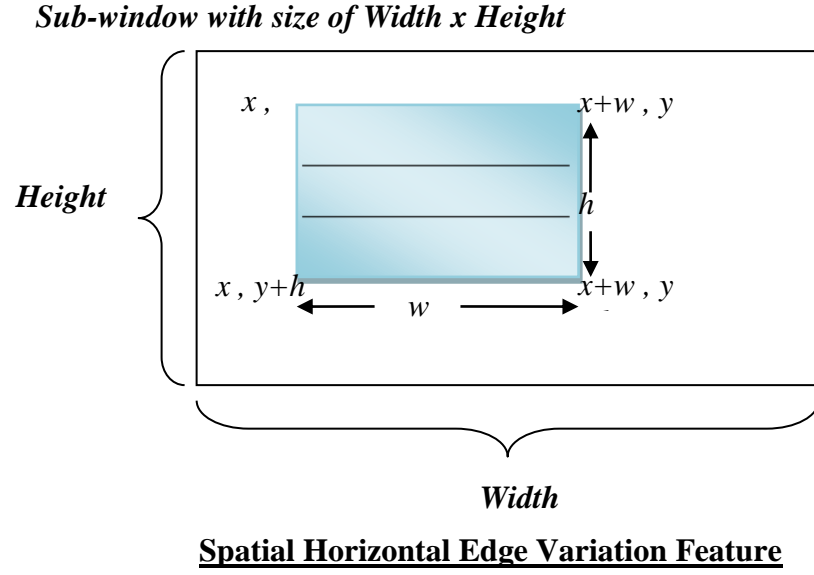


Figure 3.6 Spatial Horizontal Edge Variation Feature within the sub-window with the size of Width x Height.

3.4.4 Finding Feature Threshold

In this section, we explain the ways that we used for finding the threshold for each features. In this algorithm, each feature has its own distinct threshold value which is used for classification, and the Figure 3.7 below shows how to obtain the threshold for each feature:

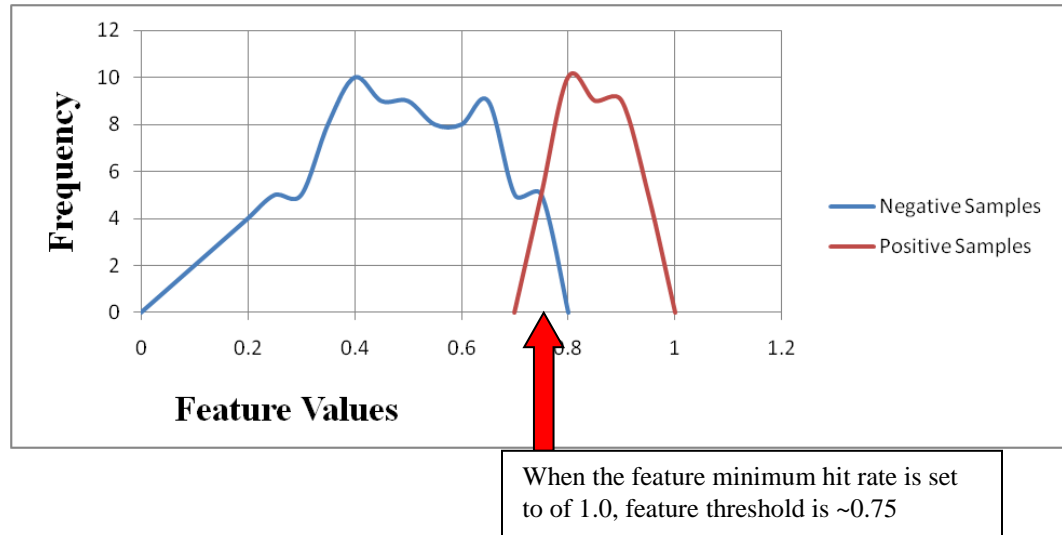


Figure 3.7 Figure shows graphs of feature values computed from a single feature on two groups of samples, which are positive samples and negative sample. If the feature minimum hit rate is set 1.0, then the feature threshold is 0.75, as the feature is required to achieve 100% of acceptance rate.

From the graph in Figure 3.7, we can observe that the distribution of the Positive samples feature values generated by this feature is from the 0.75 and above; this means that if we set the feature threshold value to be 0.75, then any values less than 0.75 will be classified as *non* target object and any values more than 0.75 will be classified as target object. Here we can observe that if the training framework is set to achieve 100% hit rate for each feature, then we will set this feature threshold to 0.75. However, setting this threshold, can increase the hit rate of the feature, on the contrary it increase the false detection of the feature. Some of the negative images might be wrongly classified as true positive.

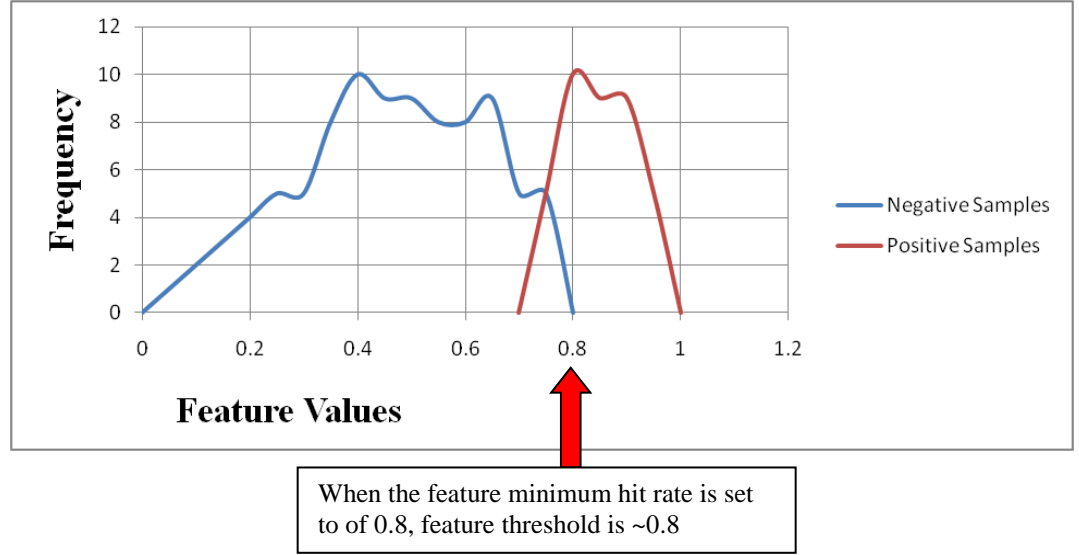


Figure 3.8 Figure shows graphs of feature values computed from a single feature on two groups of samples, which are positive samples and negative sample. If the feature minimum hit rate is set 0.8, then the feature threshold is approximated to be 0.85, as the feature is only required to achieve 80% of acceptance rate.

When we obtained the feature threshold value, $f_{rect(x,y,w,h)}\theta$, we can do classification through the equation 3.16 as shown below:

$$h_i(f_{rect(x,y,w,h)}, \theta, x_j) = \begin{cases} 1, & f_{rect(x,y,w,h)}(x_j) \leq f_{rect(x,y,w,h)}\theta \\ 0, & otherwise \end{cases} \quad (3.16)$$

Where $h_i(f_{rect(x,y,w,h)}, \theta, x_j)$ is the i -th hypothesis with feature, $f_{rect(x,y,w,h)}$, feature threshold θ , and j -th sample x_j . The hypothesis will return value 1 as a target object if the feature $f_{rect(x,y,w,h)}$ is having feature values less than or equal to the feature threshold $f_{rect(x,y,w,h)}\theta$; the hypothesis will return 0 as a non target object if the feature $f_{rect(x,y,w,h)}$ is having feature values more than feature threshold $f_{rect(x,y,w,h)}\theta$.

3.5 Genetic Algorithm for Evolutionary Feature Search

In this research, we implemented Genetic Algorithm for our feature selection. Genetic algorithm (GA) is a learning algorithm that is inspired by biological evolution which increasing hypothesis search space through iteratively *selecting* and *recombining* of the current *hypotheses* with best *fitness* and *mutating* parts of the newly generated hypothesis. In GA, different applications or problems can have different interpretations of the *Hypothesis*. Hypothesis can be interpreted as one of the candidate solutions to solve the problems; or it can be a feature that used to solve a detection problem. A collection of hypothesis is called *population* and it is updated in every iteration of generations by the *offspring* with the best *fitness* values. At the beginning, GA will initialize the population with the hypotheses generated randomly. After that, each hypothesis in the population is evaluated with the *fitness function*. (Mitchell, 1997)

Fitness function is the formulae that used to measure the chromosomes in the population, where it can be varied for different problems or applications. GA calculate the fitness value of each chromosomes and from there it can determine how fit is the chromosome. Formula 3.17 shows the formula to calculate the fitness probability of the chromosome among the population.

$$\Pr(h_i) = \frac{Fitness(h_i)}{\sum_{j=1}^p Fitness(h_j)} \quad (3.17)$$

where $\text{Pr}(h_i)$ is the probability of the i -th hypothesis, and p is the population of the current generation. Hypothesis with high fitness value will get high probability among the hypotheses in the current population.

Based on the hypothesis's probability, GA selects two hypotheses (parent chromosomes) and further recombined them with the common operator called *Crossover operator*, producing two offspring from the two parent chromosomes. These newly produced offspring chromosomes will be added to the population and continue with the reproduction for finding the best chromosomes. Figure 3.9 showing the single-point crossover operator that recombining two parent hypotheses and figure 3.10 is the further process of one-point mutation on the child offspring.

Diagram in figure 3.11 shows the overall design architectures of the Genetic algorithm adapted from (Mitchell, 1997), showing the general prototypical genetic algorithm with *fitness proportionate selection*, *crossover operator*, *point mutation operator*.

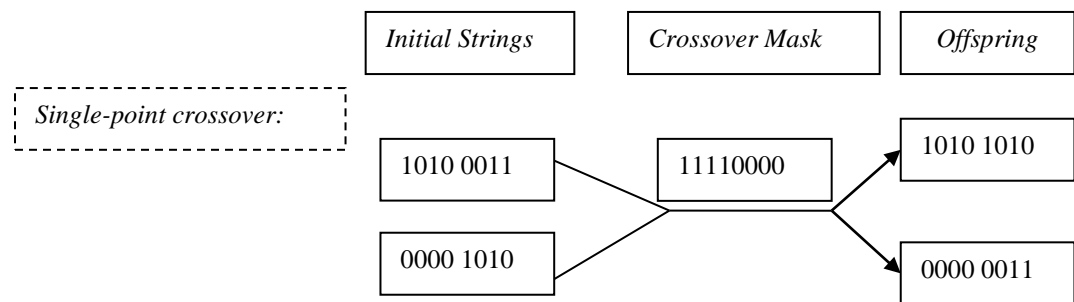


Figure 3.9 adapted from (Mitchell, 1997), showing the *single-point crossover* operator that recombining two parent hypotheses and generating two offspring with the crossover mask of a single point.

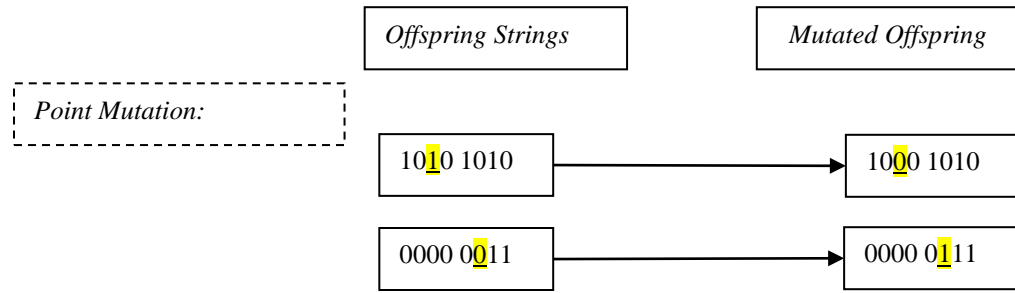


Figure 3.10 Adapted from (Mitchell, 1997), showing the *point mutation* operator that modifying a value of a bit point selected randomly onto the two offspring .

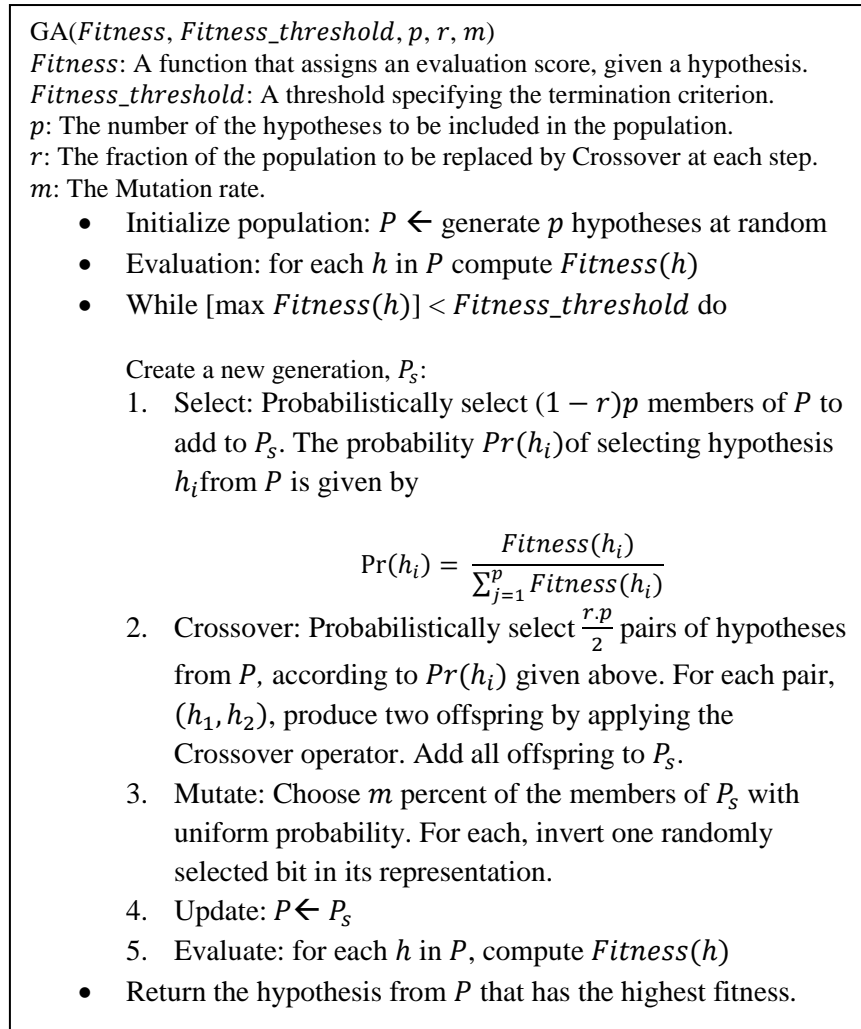


Figure 3.11 Diagram adapted from (Mitchell, 1997), showing the general prototypical genetic algorithm with *fitness proportionate selection*, *crossover operator*, *point mutation operator*.

3.5.1 Applying Genetic Algorithm - Feature Searching and Selection

After discussion of the general theory on how genetic algorithm is actually functioning, in this section, we describe how genetic algorithm is applied to our system frameworks. Genetic algorithm is applied here with some changes on the structures, used as the feature selection and feature searching technique that provides fast and optimal solution.

3.5.2 Hypothesis and population

In this project, we apply genetic algorithm as a technique that can improve the feature searching and selection, therefore, each of the hypothesis in the population is defined as a single feature. By using the genetic algorithm, we able to speed up the process of finding the efficient and best feature from the relatively large Spatial Horizontal Edge Variation feature set.

3.5.3 Fitness function and Selection

Fitness functions that we applied in this project are referring the error functions from (Viola & Jones, 2004) to compute the error for each feature during the evaluation. Instead of multiplying the error with the sample weight in (Viola & Jones, 2004), we remove the multiplication of sample weight from the error function and the formula is shown below:

$$\epsilon(h_i) = \sum_j^{n+m} |(h_i(f^{rect(x,y,w,h)}, \theta_i, x_j) - y_j)| \quad (3.18)$$

Where $\epsilon(h_i)$ is the error of the hypothesis h_i , $h_i(f^{rect(x,y,w,h)}, \theta_i, x_j)$ is the hypothesis feature value with $f^{rect(x,y,w,h)}$ is the feature and $rect(x,y,w,h)$ is the rectangle defined within the sub-window. θ_i is the feature threshold of the feature $f^{rect(x,y,w,h)}$, x_j is the j-th sample, y_j is the teaching signal of the j-th sample where $y_j = 0, 1$ respectively. Then n and m are the positive and negative sample respectively.

From this error function, we can obtain an error value for each feature in the range of

$$[0, n + m] \quad (3.19)$$

After that, the feature fitness value is calculated based on the error value:

$$Fitness(h_i) = 1 - \frac{\epsilon(h_i)}{n + m} \quad (3.20)$$

The probability $Pr(h_i)$ of selecting hypothesis h_i from P population is given by

$$Pr(h_i) = \frac{Fitness(h_i)}{\sum_{j=1}^P Fitness(h_i)} \quad (3.21)$$

3.5.4 Genetic Operators

Following the steps in the GA structure, after computing the probability for each feature, we implement the *fitness proportionate selection* method to choose two parent features from the population, and perform the genetic operators, *crossover operator* and *mutation operator*.

3.5.4.1 Single-point crossover operator

The bit strings for each hypothesis is defined by the feature rectangle $rect(x, y, w, h)$, figure 3.15 shows how two parent crossover to produce two new offspring and mutate randomly on changing the value of one of the bit.

Selection: two parent hypotheses are selected based on their fitness values.

Parent chromosome A: $h_A(f^{rect(x,y,w,h)})$

Parent chromosome B: $h_B(f^{rect(x,y,w,h)})$

Crossover: Single Crossover operator is applied to recombine both parent hypotheses.

Each hypothesis is represented by a feature with a rectangle of x , y coordinates, w width and h height within the sub-window. The rectangle $rect(x, y, w, h)$ is defined as the bit string of the hypothesis:

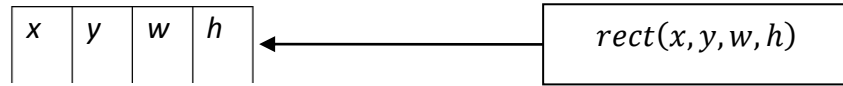


Figure 3.12 Bits String of the hypothesis

During the single-point crossover, GA will randomly choose a point position from the bit string.

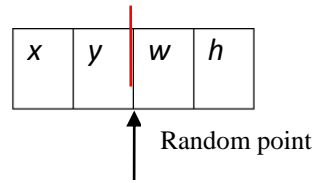


Figure 3.13 Selecting a random point for single-point crossover.

Then the single-point crossover mask is generated based on the point selected.

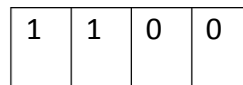


Figure 3.14 single-point crossover mask

Then the bit of the mask which is set to 1 will be copying bit string from the first parent, and the bit which is set to 0 will copy bit string from the second parent.

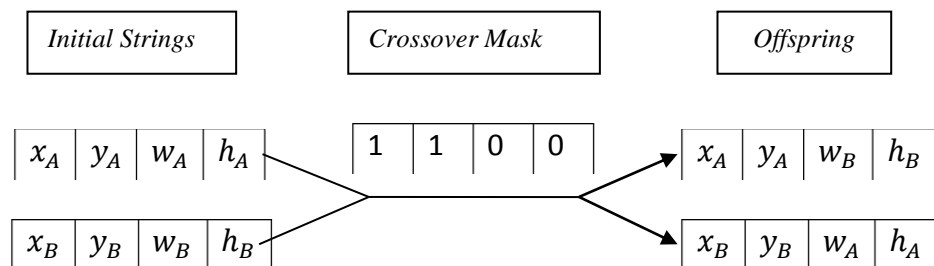


Figure 3.15 Generate offspring from selected parent using single-point crossover

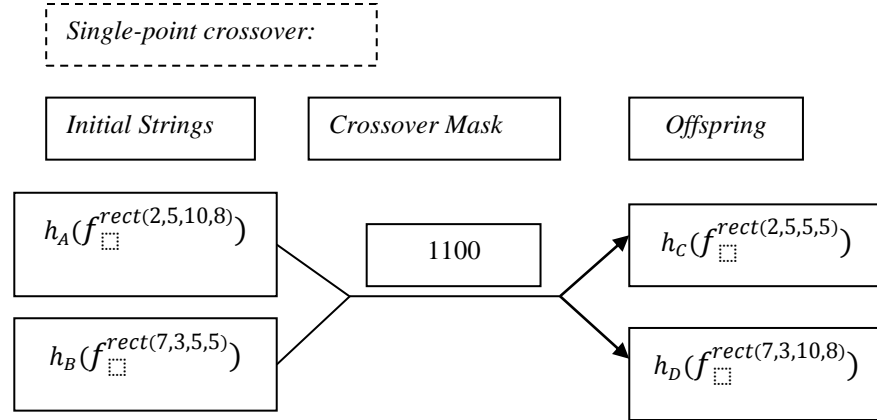


Figure 3.16 Examples of Generate offspring from selected parent using single-point crossover

Mutation: Point Mutation operator is applied to randomly change the value of one of the bit.

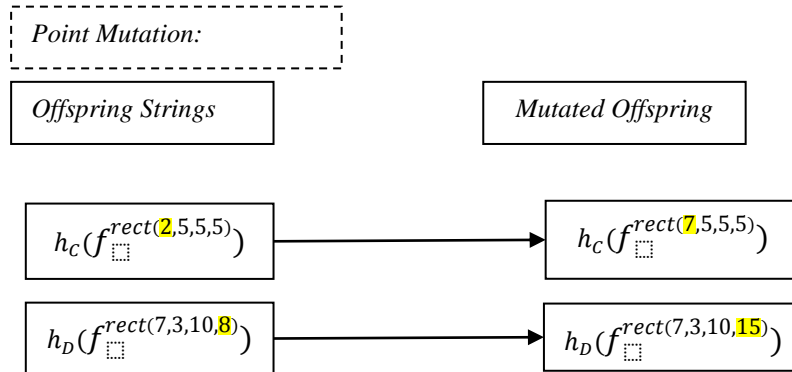


Figure 3.17 Single Point Mutation

3.5.5 Feature Repair Function

After applying crossover operation and mutation operation, we obtain two new offspring. However, there are probabilities that the newly generated offspring has invalid value on the $rect(x, y, w, h)$, where the rectangle might be outside the minimum boundary. For example, if the resolution size of the sub-window is 40 x 100 width height, and one of the offspring is $rect(15, 9, 55, 16)$, then the offspring

has an invalid value for the width size. This is because 55 is already out of the sub-window width size of 40. Therefore, we apply a feature repair function to check and repair any feature that is out of boundary or having any invalid values.

3.6 Cascading of Feature Construction

In this section, we present the technique of combining each selected features from GA into a linear cascade structure. We implement the same cascading method from (Zhang, Gao, Chen, & Zhao, 2005). It is a simple cascaded of features with classification thresholds. The decision rule of the cascaded feature is shown below:

$$\begin{aligned} & Cascade(P) \\ &= \begin{cases} 1, & \text{if } (f_{rect(x,y,w,h)_1}(P) \geq T_1 \wedge f_{rect(x,y,w,h)_2}(P) \geq T_2 \dots \wedge f_{rect(x,y,w,h)_l}(P) \geq T_l) \\ 0, & \text{otherwise} \end{cases} \quad (3.22) \end{aligned}$$

Where $Cascade(P)$ is the cascaded features with P sample, $f_{rect(x,y,w,h)_1}$ is first feature in the cascaded feature, $f_{rect(x,y,w,h)_2}$ is second feature in the cascaded feature; T_1 , T_2 , and T_l are the feature threshold for feature $f_{rect(x,y,w,h)_1}$, $f_{rect(x,y,w,h)_2}$, until $f_{rect(x,y,w,h)_l}$, respectively. If each feature in the cascaded features, $f_{rect(x,y,w,h)}(P) \geq T$, then the cascaded features will classify the sample P as target object, otherwise the sample P is classified as non-target object. Figure 3.18 shows the details of how the cascaded feature works. If any of the features classified the sub-window as non-target object, then the algorithm will not further process the sub-window, and continue with the next

sub-windows. This structure will improve the processing speed, as many false images are rejected by the features at the beginning of the cascaded features.

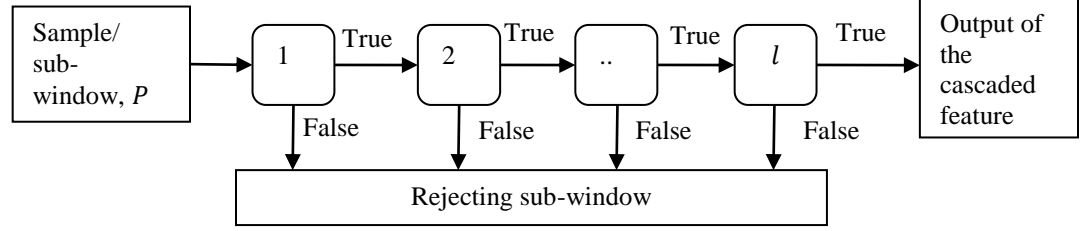


Figure 3.18 adapter and modified from (Viola & Jones, 2004), diagram showing the cascaded feature structure of processing each sub-window, each feature in the cascaded feature will determine whether to further process the sub-window or rejecting it if any of the feature classified the sub-window as non-target object.

3.7 Support Vector Machine Classification

This project uses SVM as a classifier (Burges, 2004) (Cristianini & Shawe-Taylor, 2000), which is provided by the library LIBSVM (Chang & Lin, 2001). SVM is a linear perceptron that solves binary classification problem. Some supervised learning algorithm is usually applied to find an optimal N-dimensional hyperplane. The decision function of a SVM is defined as,

$$f(x) = \text{sgn} \sum_{i \in S} y_i \alpha_i K(x, s_i) + b \quad (3.23)$$

where x is the input to be classified into $y \in \{-1, +1\}$ and $S = \{i | \alpha_i \in \mathbb{R}^+\}$ is a set of positive coefficients. Support vectors s_i , $i \in S$ are a subset of the training vector extracted during the optimization process. K is a user-chosen kernel function.

In reality, problems are seldom linearly separable. Thus, a non-linear kernel function that meets the Mercer's theorem (Vapnik, 1995) is needed to map input space to higher dimensions so that one can solve a non-linear problem using linear classification. In this paper, radial basis kernel is used. It is defined as,

$$K(x, z) = \exp(-\gamma \|x - z\|^2) \quad (3.24)$$

Given training vectors $x_i \in \mathbb{R}^n$, $i = 1, \dots, l$ such that $y_i \in \{-1, +1\}$, the following primal problem must be minimized (Vapnik, 1995)

$$Q(w, \xi_i, b, p) = \frac{1}{2} w^T w - vp + \frac{1}{l} \sum_{i=1}^l \xi_i \quad (3.25)$$

Where $y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i$, $\xi_i \geq 0$ ($i = 1, \dots, l$), $p \geq 0$ Its dual formulation is defined as,

$$Q(\alpha) = \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j y_i y_j K(x_i x_j) \quad (3.26)$$

Where $0 \leq \alpha_i \leq 1$ ($i = 1, \dots, l$), $e^T \alpha \geq vl$, $y^T \alpha = 0$

$v \in (0, 1]$ is a user specified parameter to control the number of support vectors and training errors. It is an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors.

3.8 Image Sub-window Scanning with Cascaded Features

After searching and selecting the best features using Genetic Algorithm feature search, cascading those selected feature and adding Support Vector

Machine classifier as the false alarm removal; the final detector is tested through scanning an image with the moving sub-window start from the beginning location (0,0). Instead of using image pyramid scanning method from (Zhang, Gao, Chen, & Zhao, 2005), we implemented the feature scanning from (Viola & Jones, 2004). We rescale the sub-window size with a rounding scale factor $[sf \Delta]$, as the feature values within the sub-window will be normalized in conjunction with the changing size of the sub-window. The $[sf \Delta]$ values cannot set to a high value as the detection rate might decrease and decreasing the false positive. Another parameter that affecting the speed and performance of the image sub-window scanning is the moving pixel changes, $mp \Delta$. The sub-window is shifted to the right by some number of pixel $mp \Delta$. If the $mp \Delta$ is high, then it can improve on the detector speed, as the number of scanning window reduce if the scanning window shifted a lot to the right. However, this might affect the detection rate of the detector, as some of the target objects might left out if the sub-window moves at high $mp \Delta$.

For each sub-windows extracted from the image during the scanning, we applied the cascaded features to classify whether the sub-window is target object or non target object. If the sub-window is classified as a target object, then this sub-window is further verified by the Support Vector Machine classifier. Support Vector Machine classifier will filter those false detection from the cascaded features. From here, both Cascaded feature and Support Vector Machine constructed a two-stage object detector. Finally the results from the Support

Vector Machine is further post-processed and labeled with the rectangle specifying it is the target object.

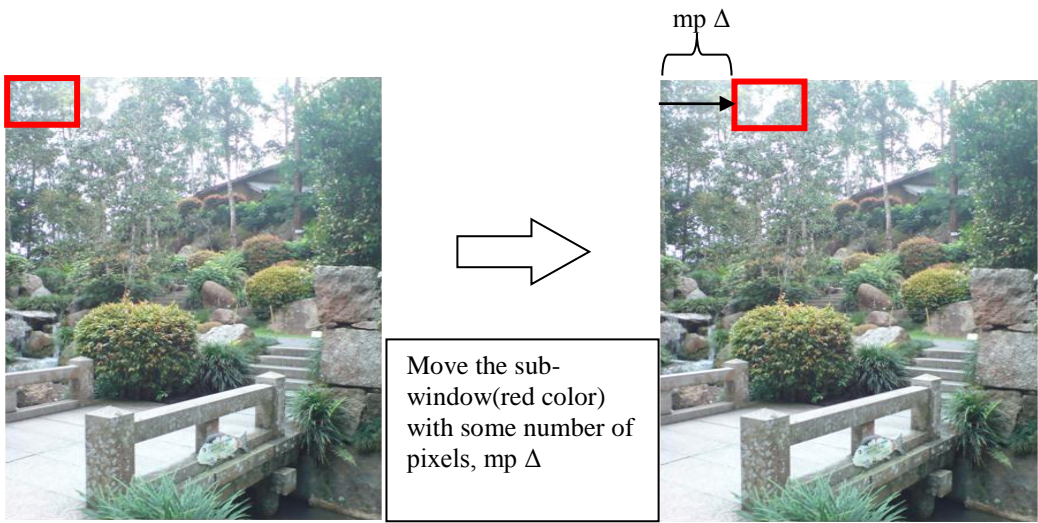


Figure 3.19 Sub-window scanning Process, Move Sub-window based on some number of pixels, $mp \Delta$

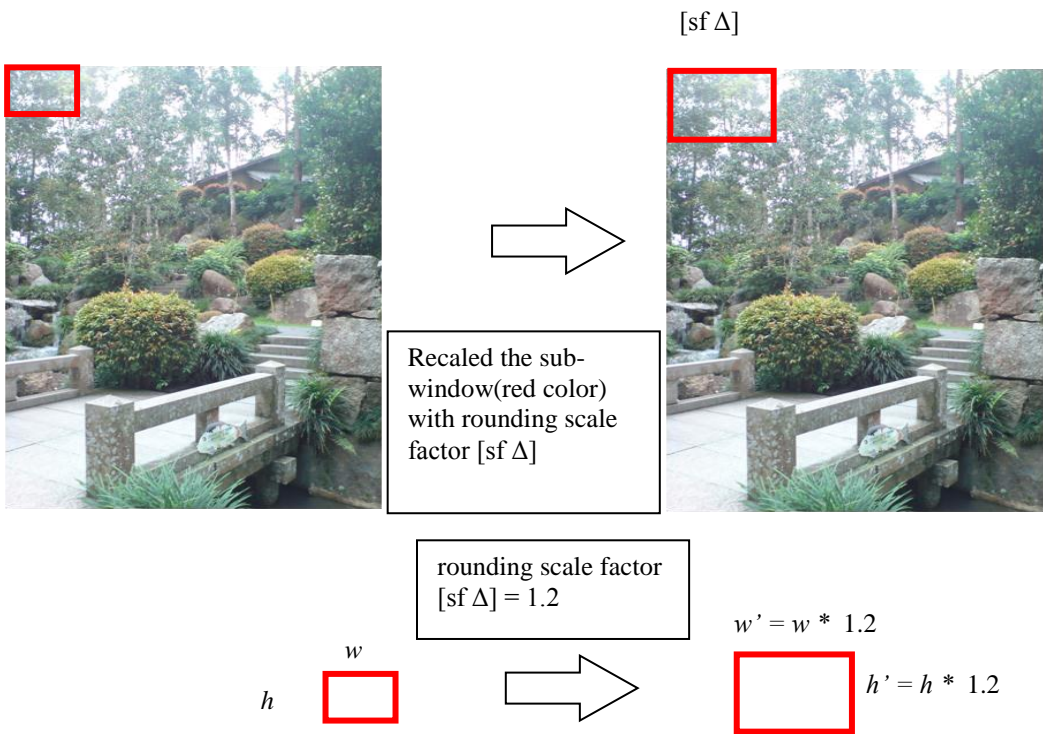


Figure 3.20 Sub-window scanning Process, Rescale Sub-window based on rounding scale factor $[sf \Delta]$

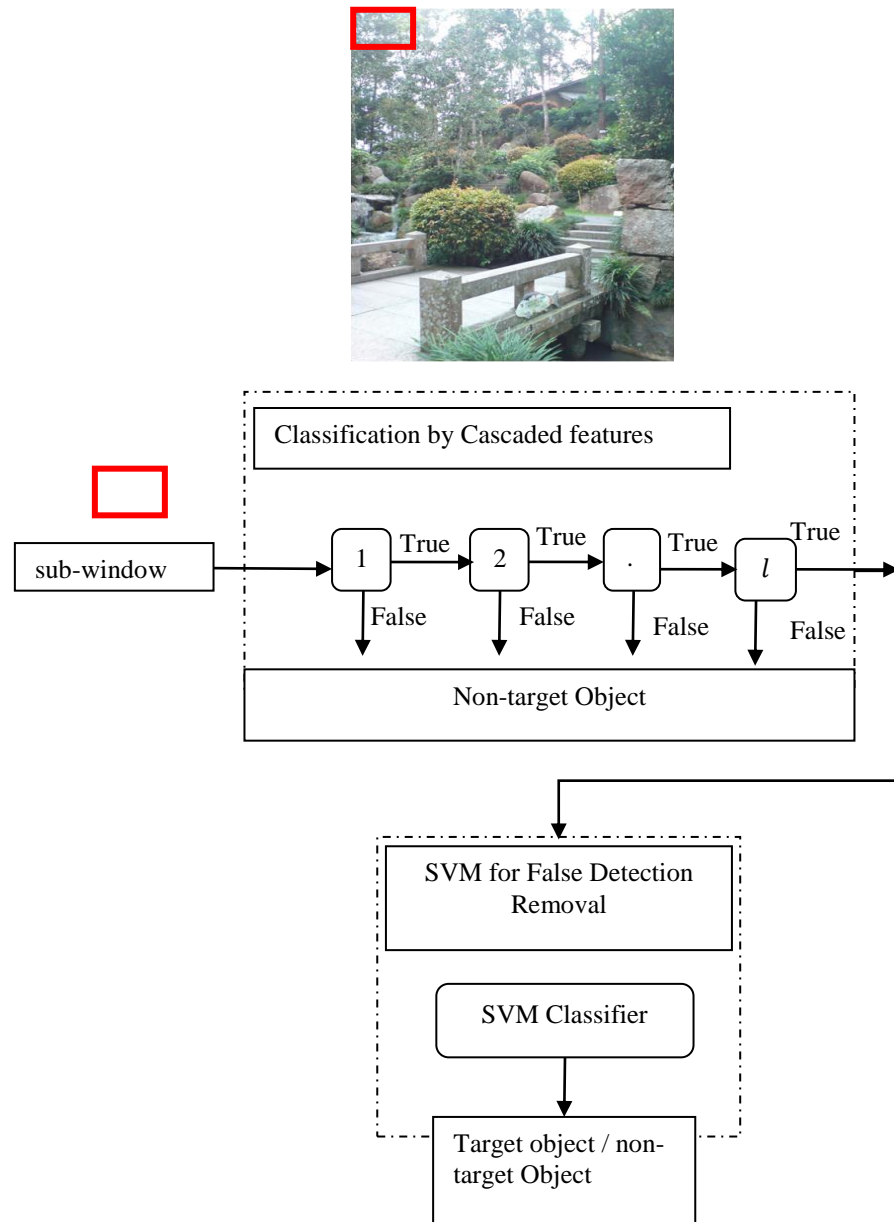


Figure 3.21 Sub-window scanning Process, cascaded feature classification and SVM false detection removal.

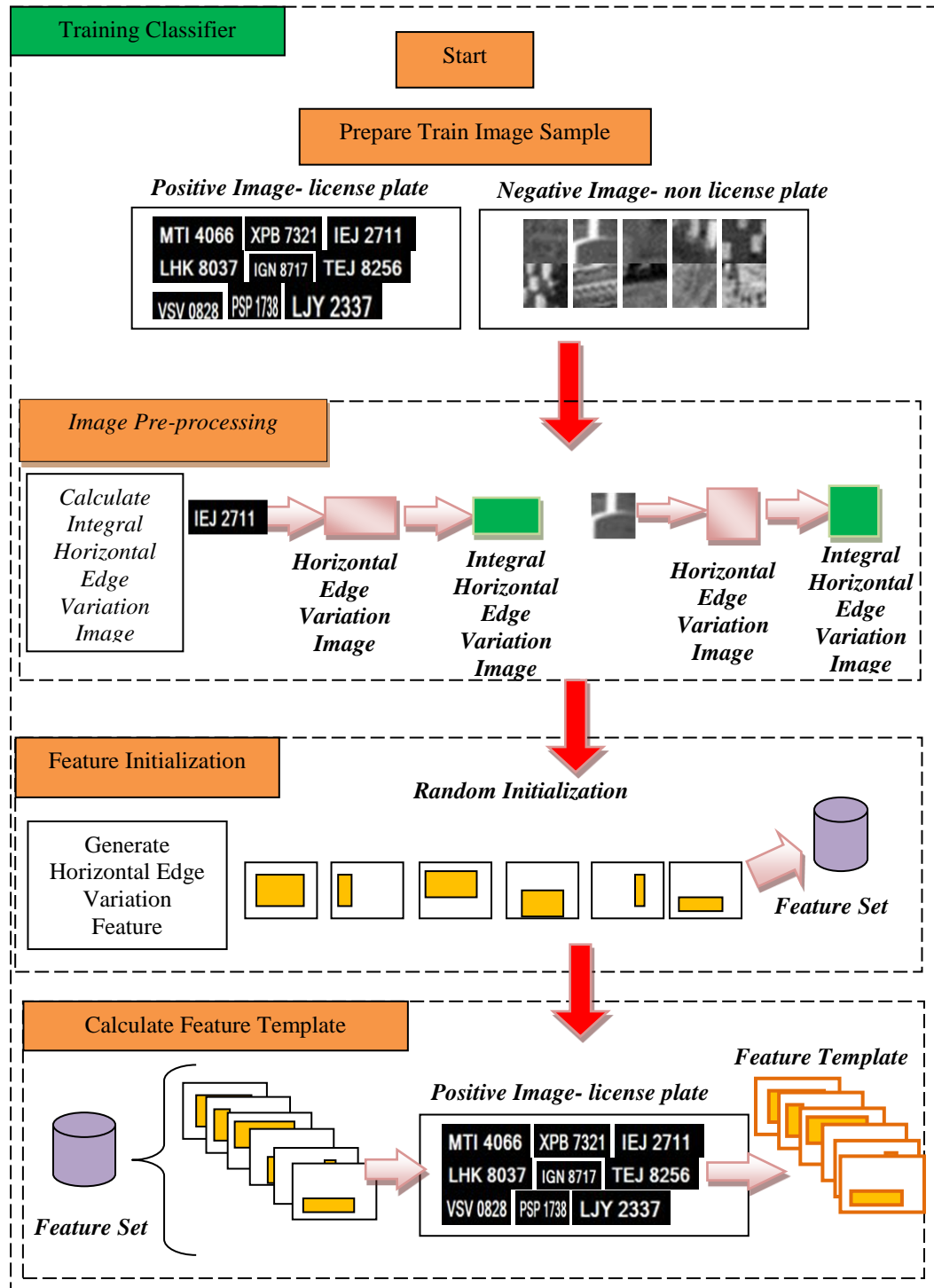


Figure 3.22.1 System Architecture for *Training* Stage in Object Detection using Spatial Horizontal Edge Variation Feature.

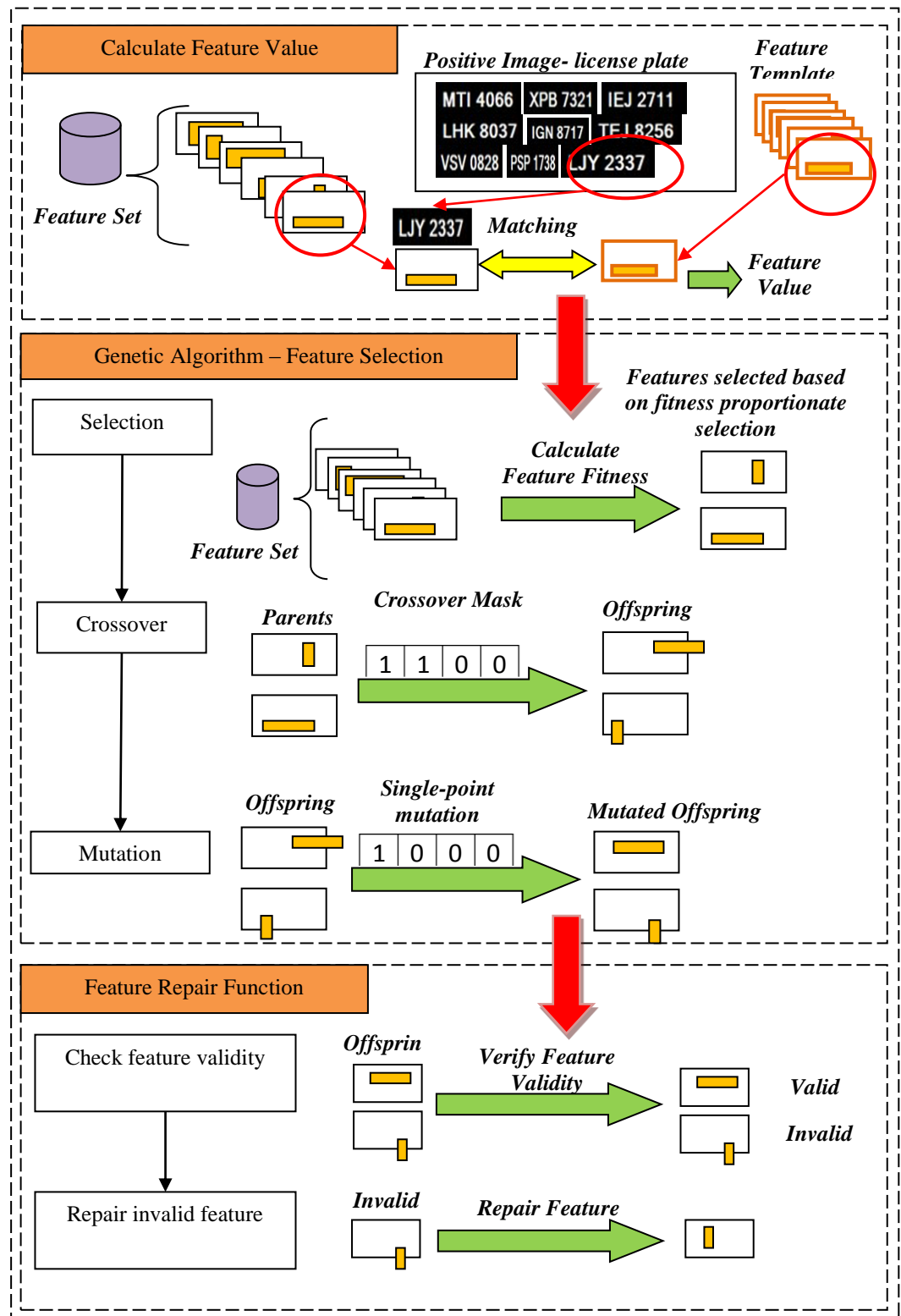


Figure 3.22.2 System Architecture for *Training* Stage in Object Detection using Spatial Horizontal Edge Variation Feature.

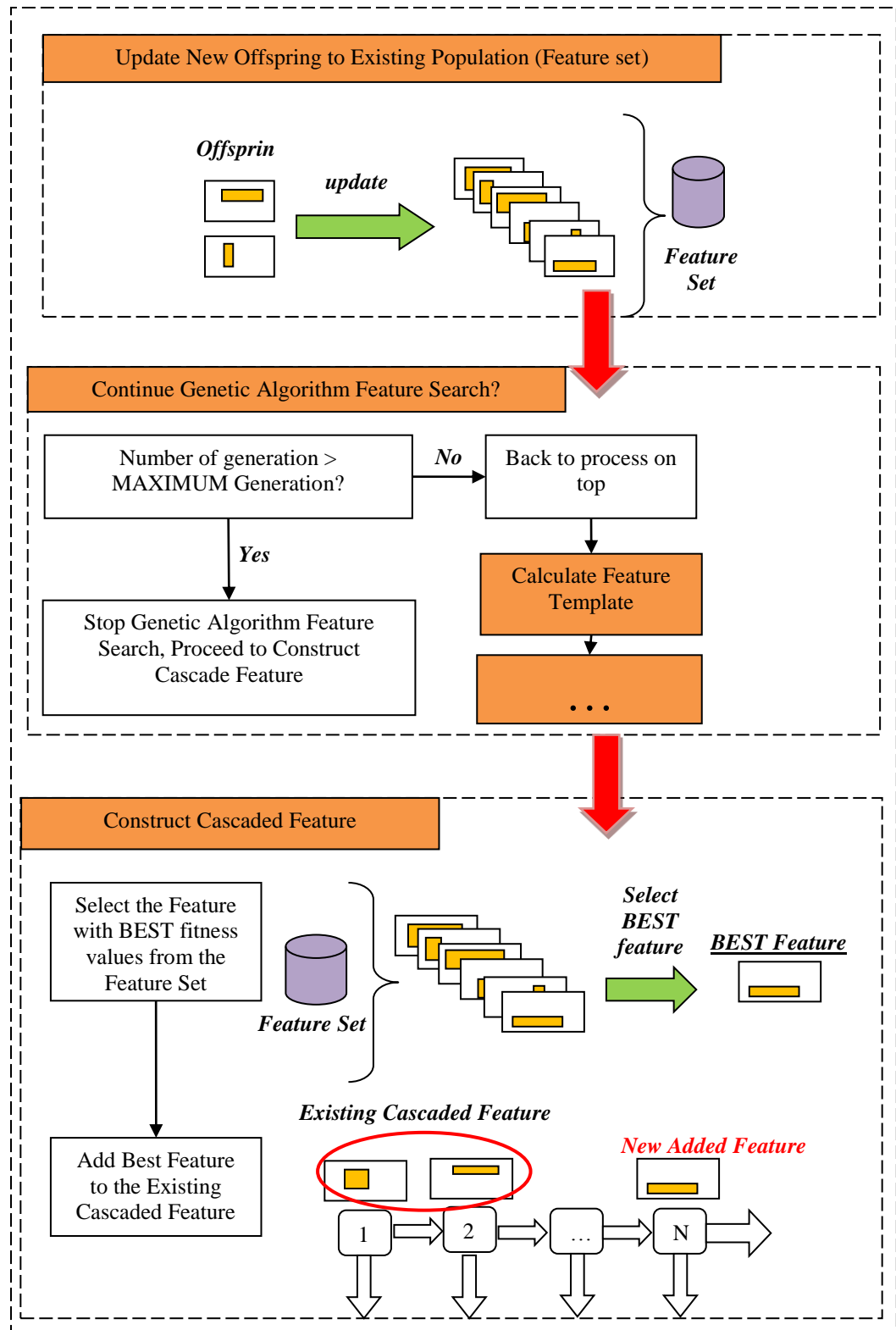


Figure 3.22.3 System Architecture for *Training* Stage in Object Detection using Spatial Horizontal Edge Variation Feature.

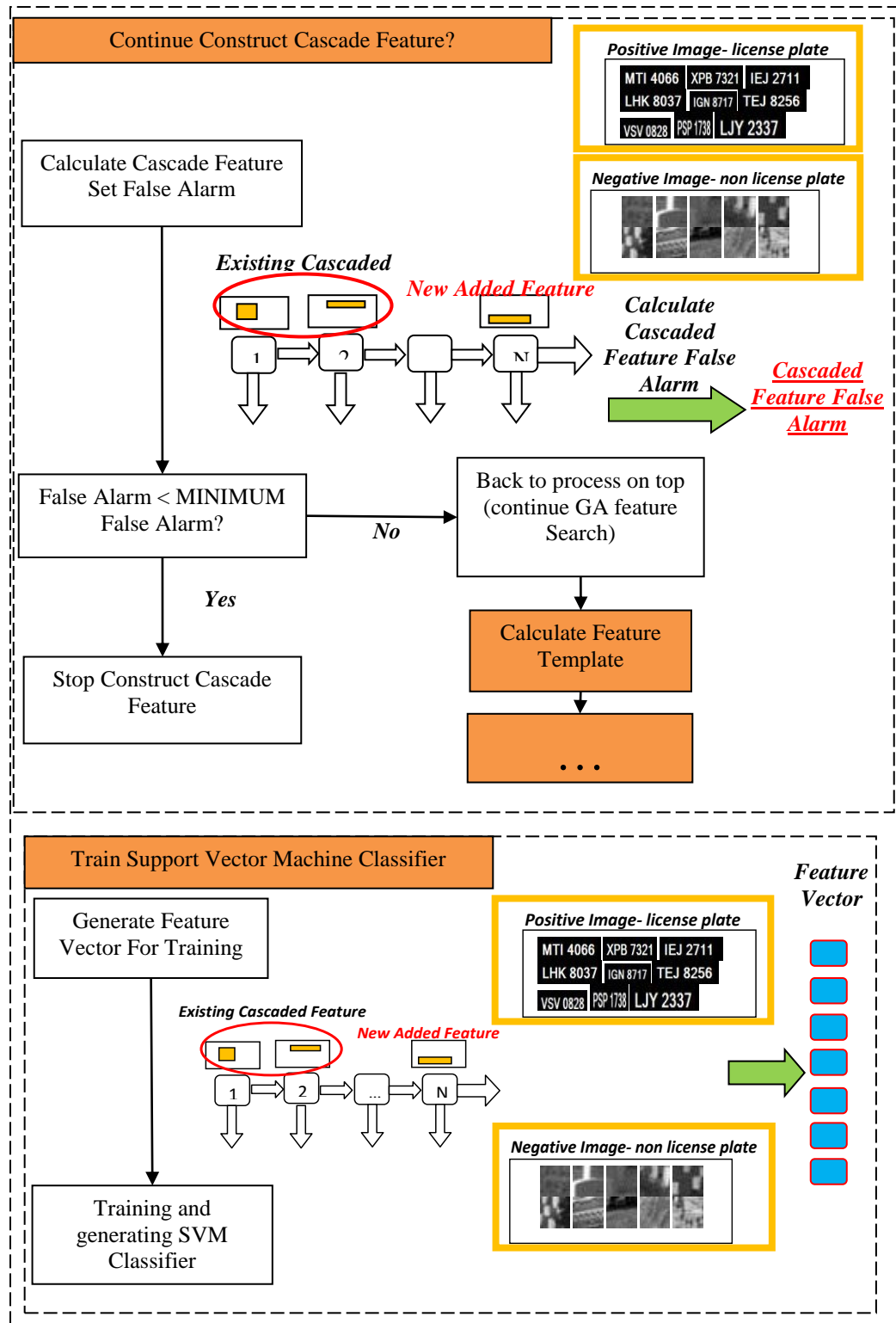


Figure 3.22.4 System Architecture for *Training Stage* in Object Detection using Spatial Horizontal Edge Variation Feature.

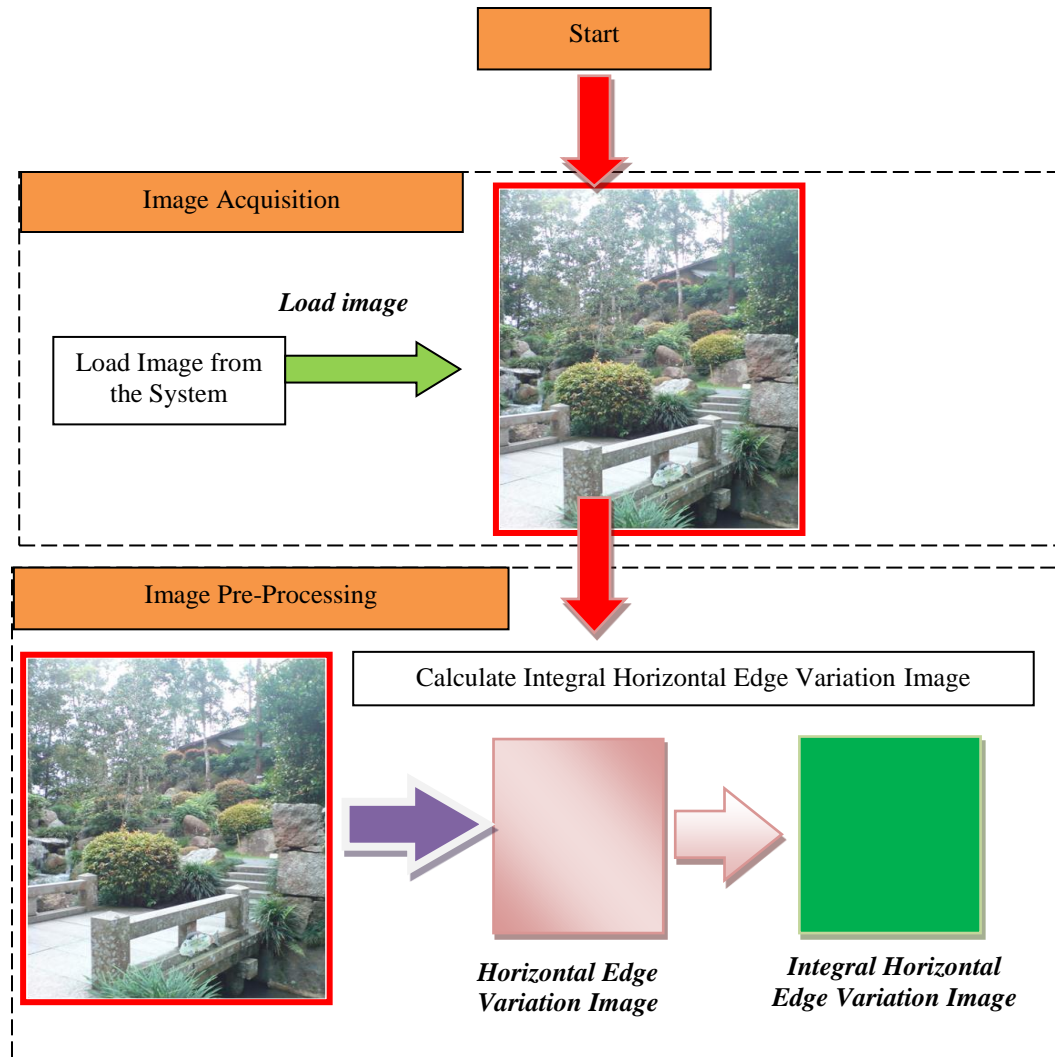


Figure 3.23.1 System Architecture for *Testing* Stage in Object Detection using Spatial Horizontal Edge Variation Feature.

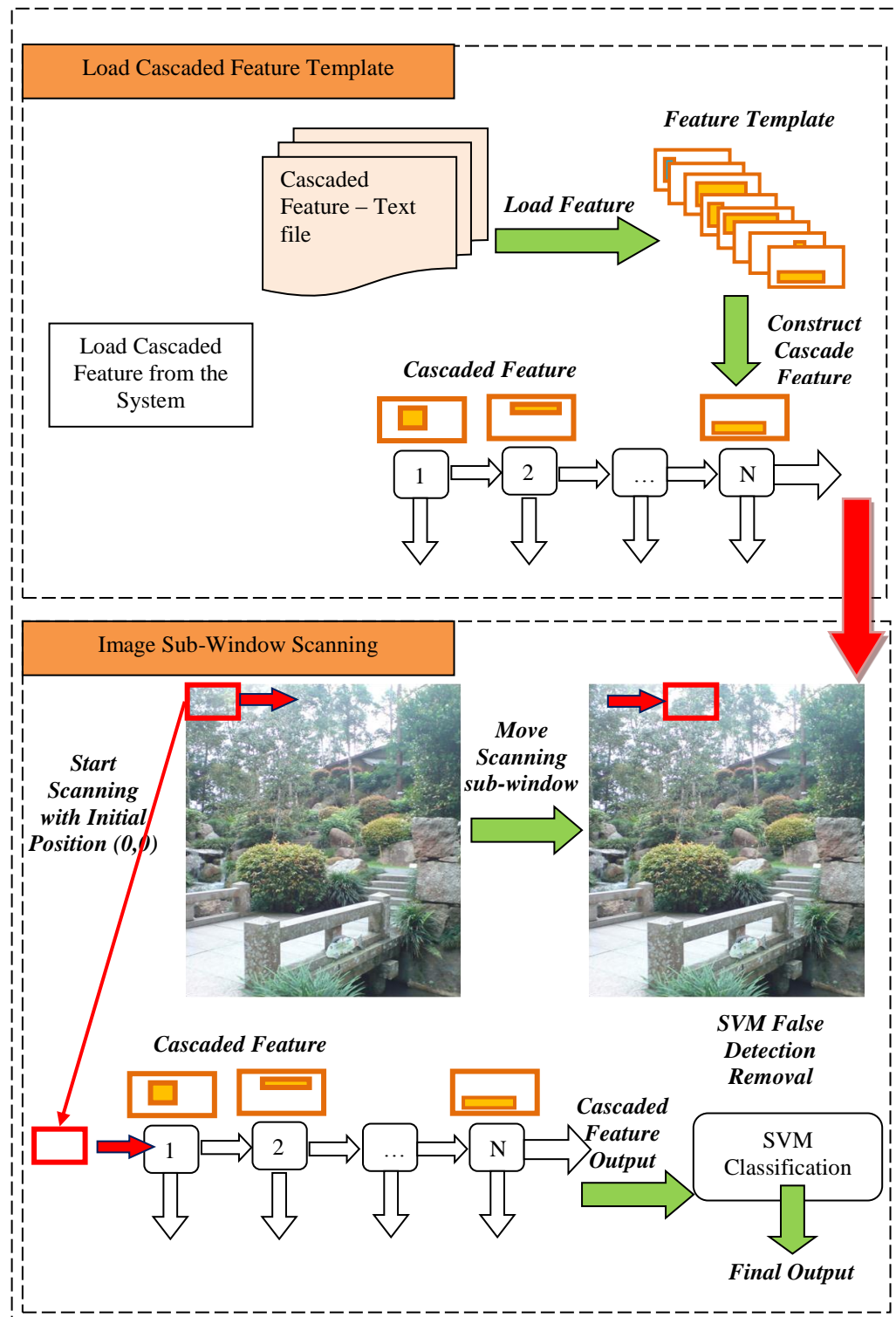


Figure 3.23.2 System Architecture for *Testing* Stage in Object Detection using Spatial Horizontal Edge Variation Feature.

Table 3.1 The learning algorithm for constructing cascade feature during *training* Stage in Object Detection using Spatial Horizontal Edge Variation Feature.

- Given Training Sample. $TrainSample = \{(x_1, y_1), \dots, (x_n, y_n)\}$, where x_n is the sample, n is the total number of positive and negative sample, and $y_n = 1, 0$ for positive samples and negative samples respectively.
- Random Initialization of Feature Set, Spatial Horizontal Edge Variation, $SpHEV_{rect(x,y,w,h)}$.
- Calculate Integral Horizontal Edge Variation Image ii_{hev} .
- Initialize Cascaded of Feature, $CascadedFeat$
- while $CascadedFeatureFalseAlarm > MaximumFalseAlarm$ do
 - Define each hypothesis by $SpHEV_{rect(x,y,w,h)}$
 - Initialize population: $P \leftarrow$ generate p hypotheses at random
 - Evaluation: for each h in P compute $Fitness(h)$
 - While $t < Max_Generation$ do
 - i. Create a new generation, P_s :
 - ii. Select: Probabilistically select $(1 - r)p$ members of P to add to P_s .
The probability $Pr(h_i)$ of selecting hypothesis h_i from P is given by

$$1. \quad Pr(h_i) = \frac{Fitness(h_i)}{\sum_{j=1}^p Fitness(h_j)}$$
 - iii. Crossover: Probabilistically select $\frac{r \cdot p}{2}$ pairs of hypotheses from P , according to $Pr(h_i)$ given above. For each pair, (h_1, h_2) , produce two offspring by applying the Crossover operator. Add all offspring to P_s .
 - iv. Mutate: Choose m percent of the members of P_s with uniform probability. For each, invert one randomly selected bit in its representation.
 - v. Repair Invalid Feature that is out of boundary
 - vi. Update: $P \leftarrow P_s$
 - vii. $t \leftarrow t + 1$
 - Return the hypothesis from P that has the highest fitness.
 - $CascadedFeat \leftarrow$ hypothesis, h_j
 - Evaluate $CascadedFeat$ onto $TrainSample$, compute $CascadedFeatureFalseAlarm$
- Final Cascaded Features are:

$Cascade(P)$

$$= \begin{cases} 1, & \text{if } (f_{rect(x,y,w,h)_1}(P) \geq T_1 \wedge f_{rect(x,y,w,h)_2}(P) \geq T_2 \wedge \dots \wedge f_{rect(x,y,w,h)_l}(P) \geq T_l) \\ 0, & \text{otherwise} \end{cases}$$

Where h_1 is the hypothesis $h_1(f_{rect(x,y,w,h)})$ with feature, $f_{rect(x,y,w,h)}$, and its feature threshold T_1 , so on and so forth.

CHAPTER FOUR

EXPERIMENTS SETTING AND RESULTS

In this chapter, we show some of the experiments we had done for analyzing the classifier generated from the license plate detection framework using Spatial Horizontal Edge Variation feature. Few objectives of the experiments are:

- Analysis on the feasibility of Spatial Horizontal Edge Variation feature in detecting license plate with artificial generated license plate images as training samples.
- Analysis on the feasibility of Spatial Horizontal Edge Variation feature in detecting text with ICDAR 03 text dataset as testing samples.
- Analysis on the feasibility of Spatial Horizontal Edge Variation feature in detecting side-view car with UIUC car dataset as training samples and testing samples.
- Analysis on effects of the number region of Spatial Horizontal Edge Variation feature on the performance of the classifier.
- Analysis on the effects of the threshold used to generate Spatial Horizontal Edge Variation image.
- Analysis on the performance of the classifier trained with training images that are converted to Canny Edged Images.
- Analysis on the performance of the learning algorithm

- Cascaded feature classification
- Support Vector Machine classification
- Analysis on how implementation of Support Vector Machine improving the accuracy of the detector.

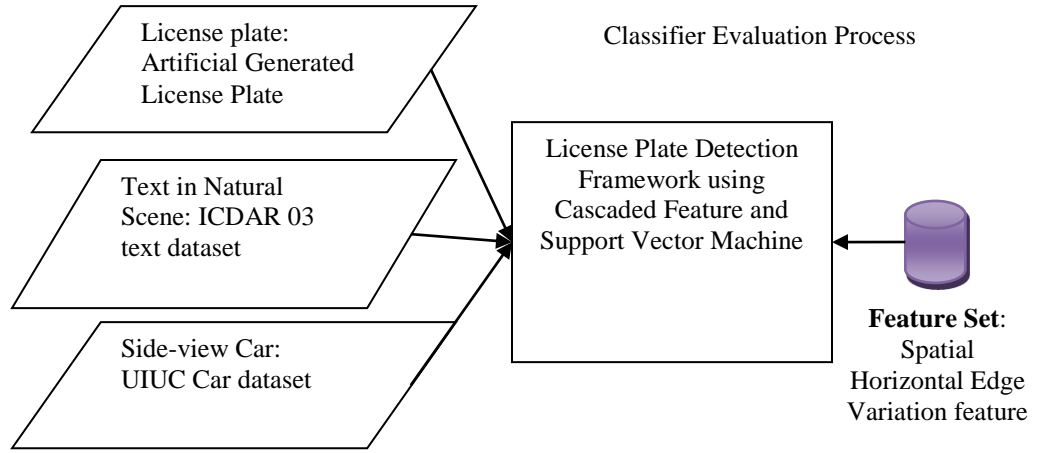


Figure 4.1 shows a brief testing framework for this project.

4.1 Experiment Evaluation Function

We implemented the evaluation methods from (Lucas, Panaretos, Sosa, Tang, Wong, & Young, 2003) to measure the performance of the object detector generated by our approach. For a given image, there are numbers of target object, T , and the output generated from the detector is known as estimates, E . The recall rate is the number of correct estimates divided by the total number of targets.

$$RecallRate = \frac{Correct_{estimates}}{|T|} \quad (4.1)$$

The precision rate is the number of correct estimates divided by the total number of estimates where correct estimate is the estimated output with the target objects.

$$PrecisionRate = \frac{Correct_{estimates}}{|E|} \quad (4.2)$$

High recall rate and high precision rate shows the detector is having high detection rate and low false detection. However, if the detector detecting many regions that are not the target objects, then the detector will be having low precision rate.

After that, we referred to another evaluation methods from (Viola & Jones, 2004) to generate the ROC (Receiver Operating Characteristic) Curve, a graph that represents the True Positive Rate VS the False Positive Rate of the features from different threshold setting. Such graph representation enables us to overview the performance results of a classifier and select the optimal model (threshold setting) that provide the optimal results in the detection accuracy.

TPR also know as True Positive Rate is defined as the test result that how many correct results detected from the classifier over a set of positive test sample. Formulae 4.3 shows the method to calculate the True Positive Rate where True Positive (TP) is when the estimated results generated by the classifier is the same as the actual result, False Negative (FN) is when the estimated result is classified as the non-target object but the actual result is a target object.

$$TPR = \frac{TP}{(TP + FN)} \quad (4.3)$$

FPR also known as False Positive Rate, is defined as the rate of how many incorrect false positive that detected during the testing, where FP is the false positive and TN is the true negative.

$$FPR = \frac{FP}{(FP + TN)} \quad (4.4)$$

4.2 Experiment Setting and Results

In this section, we start our experiments with collecting positive training samples and testing samples. We measure the performance of the classifier on three types of object detection problem.

- License Plate Detection
- Text Detection in Natural Scene Images (Nobuo, 2004)
- Side-View Car Detection

In our project, we tested our approach with a few benchmark datasets on text, car, and our own artificial generated license plate.

Object	Training Dataset Name	Testing Dataset Name
Vehicle License Plate	Artificial Generated License Plates	Real world Image with Malaysian Vehicle License Plate
Text in Natural Scene Images	Artificial Generated Text	ICDAR 03 text testing dataset
Side-view Car	UIUC Car training dataset	UIUC Car Testing dataset

Figure 4.2 Training Dataset and Testing Dataset for each object detection problem

In our experiment of solving license plate detection problems, we collected 150 real world vehicle license plate testing samples and our training samples consist of two groups of images. These two groups of images are categorized as positive samples and negative samples, where positive samples are images of the target object, and negative samples are images without any target object. For this experiment, we created a program that randomly generates artificial license plate as our training samples. All the artificial license plate samples are following Malaysian vehicle license plate standard, which constructed of alpha numeric. Figure 4.3 shows some images from the artificial license plates generated from our program, and Figure 4.4 shows some images collected from the World Wide Web which are the negative training sample consists of non vehicle license plate.

V 058	V 264	PG 360	NF 655
RM 467	VRL 2485	BED 8757	T 634
LX 2	GPG 6147	OJ 087	OB 480
C 60	V 204	FH 542	P 14
W 866	YD 28	YB 833	K 5
S 15	GB 647	U 52	DW 607

Figure 4.3 Positive Training Sample consist of Artificial Vehicle License plate images

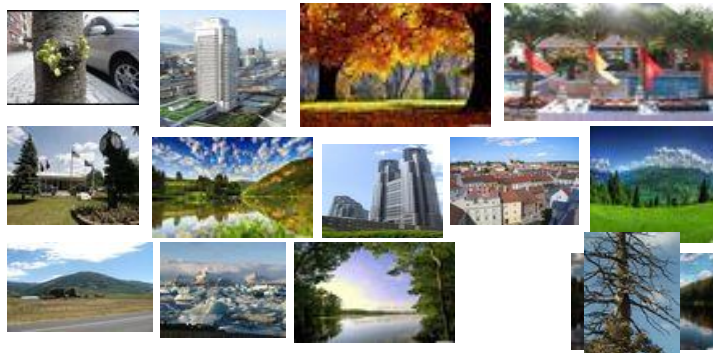


Figure 4.4 Negative Training Sample consists of Non-Vehicle License plates images collected from World Wide Web.

Table 4.1 Definition of the parameters that is tested during the experiments of Spatial Horizontal Edge Variation Feature.

Parameter	Definitions
Number of Regions in Spatial Horizontal Edge Variation Feature	Number of regions that divide the sub-rectangle of Spatial Horizontal Edge Variation feature into few sections. Each section generates the average of the total pixel changes from the left to right of the image.
Horizontal Edge Variation Threshold – Training	This parameter is set during the training of the detector, where this threshold parameter is to generate the Horizontal Edge Variation Image; the threshold value is to set the boundary of the two groups of values. For example, if the threshold is set to 45, then if the difference of the left pixel and right pixel is larger than 45, then 1 will be set at (x, y) pixel of the Horizontal Edge Variation image.
Horizontal Edge Variation Threshold – Testing	This parameter is the same as the threshold set during training, where it is also used to generate the Horizontal Edge Variation Image; the difference is the threshold is set during evaluating the detector. By setting this threshold differently, we can observe the changes of the detector accuracy.

4.2.1 Experiment: Spatial Horizontal Edge Variation feature on License Plate Detection

Table 4.2 Experiment Setting for Testing Spatial Horizontal Edge Variation Feature in detecting Vehicle License Plate

Experiment	Descriptions
Hypothesis	Feasibility of Spatial Horizontal Edge Variation Feature in detecting license plate
Objective	To test whether Spatial Horizontal Edge Variation Feature is able to provide high accuracy in detecting license plate using artificial license plate
Training Samples	Artificial Generated License Plates
Testing Samples	Real world Image with Malaysian Vehicle License Plate
Static Variables	Feature Set: Spatial Horizontal Edge Variation Feature Training sample (grayscale image) Positive Sample: artificial license plate Number of positive sample: 1900 Negative Sample: Non license plate sample from internet Number of negative sample: 1000 Testing sample (grayscale image) Real world license plate captured from outdoor environment Number of samples:150 Single-point crossover Single-point mutation Minimum Hit rate: 0.95 Horizontal Edge Variation Threshold: 45, 90

We conducted some experiments to test on the effect of the number of regions of Spatial Horizontal Edge Variation feature in providing better result. We trained our vehicle license plate detector with the artificial generated vehicle license plates, and tested the vehicle license plate detector with some testing images consist of real license plate. We generated 1900 of positive samples of

artificial vehicle license plate, and 1000 of negative samples of non vehicle license plate from internet.

During the training, we implemented a boosting-like technique to load our negative samples where those negative samples that are classified correctly will not be used in the next round of the training. Firstly, the algorithm will generate some random features, using genetic algorithm feature search to select a feature with best fitness value. After that, the best feature will be added to the cascading list, and the cascaded feature will be used to select another 1000 negative samples from the negative images. Any negative samples that are correctly classified will not be included in the next iteration of training. Therefore, each feature in the cascaded feature can help the algorithm to ensure that the next selected feature is responsible to focus more on any negative sample that the existing cascaded feature failed to classify correctly.

The following are some results and graphs of the detection rate and precision rate by each license plate detector generated based on different setting. All the results from the experiments show how the numbers of regions in Spatial Horizontal Edge Variation feature affecting the performance of the detector. Meanwhile, we can observe that the training threshold and testing threshold are also becoming one of the factors that affect the accuracy.

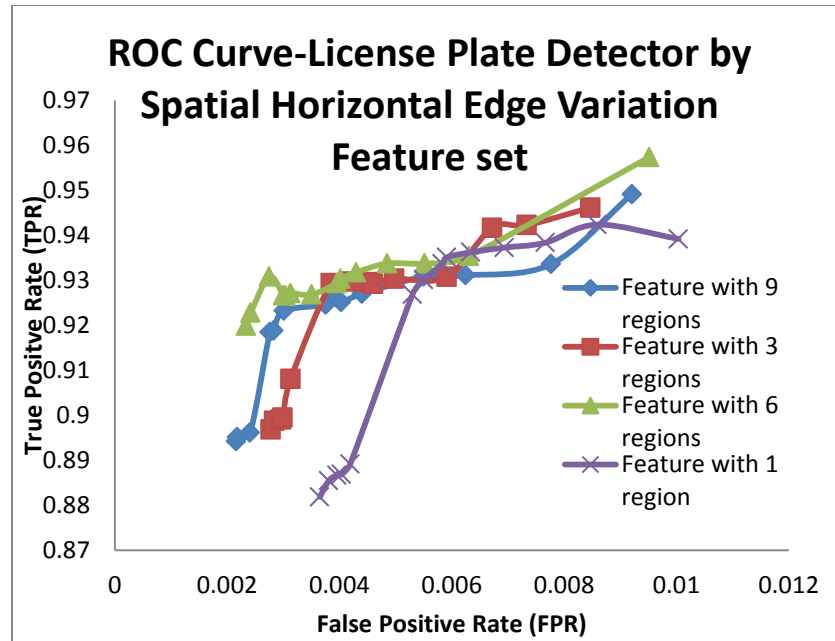


Figure 4.5 shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers.

From Figure 4.5 we plot the graph of the ROC curve showing the true positive rate and false positive rate of classifiers with different number of Spatial Horizontal Edge Variation feature region. Overall we found that the classifier with Spatial Horizontal Edge Variation feature (6 regions) generates the highest optimal true positive rate among the rest of the classifiers such as (1 region, 3 regions, and 9 regions). Although the classifier with Spatial Horizontal Edge Variation feature (6 regions) has the highest true positive rate among them, but yet the difference of the true positive for the 4 classifiers does not have a big gap between each other. From the results, we found that the true positive rate actually increasing when we increase the number of regions for the Spatial Horizontal Edge Variation feature. From the graph, we shows that the classifiers is generating high true positive rate while at the same time it also generating higher

false positive rate. Our final classifiers that consist of multiple Spatial Horizontal Edge Variation features are designed in a linear structure format that the initial features in the list normally allow more false detection and pass to the subsequent features to further filter the false alarm. As more features are added to the final classifier, the true positive rate will decrease and at the same time it reduce the false positive rate.

Table 4.3 Results for Classifiers with different number of regions corresponding to the same Horizontal Edge Variation Threshold.

Number of regions in Spatial Horizontal Edge Variation feature	Horizontal Edge Variation Threshold Training	False Positive Rate	True Positive Rate
1	45	0.00365	0.88186
3	45	0.00278	0.89686
6	45	0.00234	0.91988
9	45	0.00216	0.89423

Next we proceed to examine the feasibility of the Spatial Horizontal Edge Variation feature in solving the License Plate Detection Problem. As referring to previous chapter in explanations on the Spatial Horizontal Edge Variation feature, this feature is sensitive to the pixel changes horizontally from the left to the right of the image. Therefore, for our next experiment, we plot two graphs to show how our features are sensitive to the alphanumeric license plate image while able to filter non-license plate image. From the graph in figure 4.7, there are 6 types of data such as

- Feature Value – 1st Region
- Feature Value – 2nd Region
- Feature Value – 3rd Region
- Feature Value – 4th Region

- Feature Value – 5th Region
- Feature Value – 6th Region

We can see from the graph that the 1st region and the 6th region is having feature value close to zero is because these two regions are having less pixel changes from the left to right of the image. Meanwhile, we found that the 3rd region and 4th region of the feature are generating high feature values for each license plate image consistently. This is because these regions are in the middle of the Vehicle License plate image where the alphanumeric placed. We can refer to the figure 4.6 on the actual license plate image and how the regions of the Spatial Horizontal Edge Variation feature can be applied to the image.

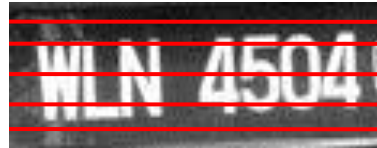


Figure 4.6 Example of the Actual License Plate and the Description of the Spatial Horizontal Edge Variation Feature Region Distribution.

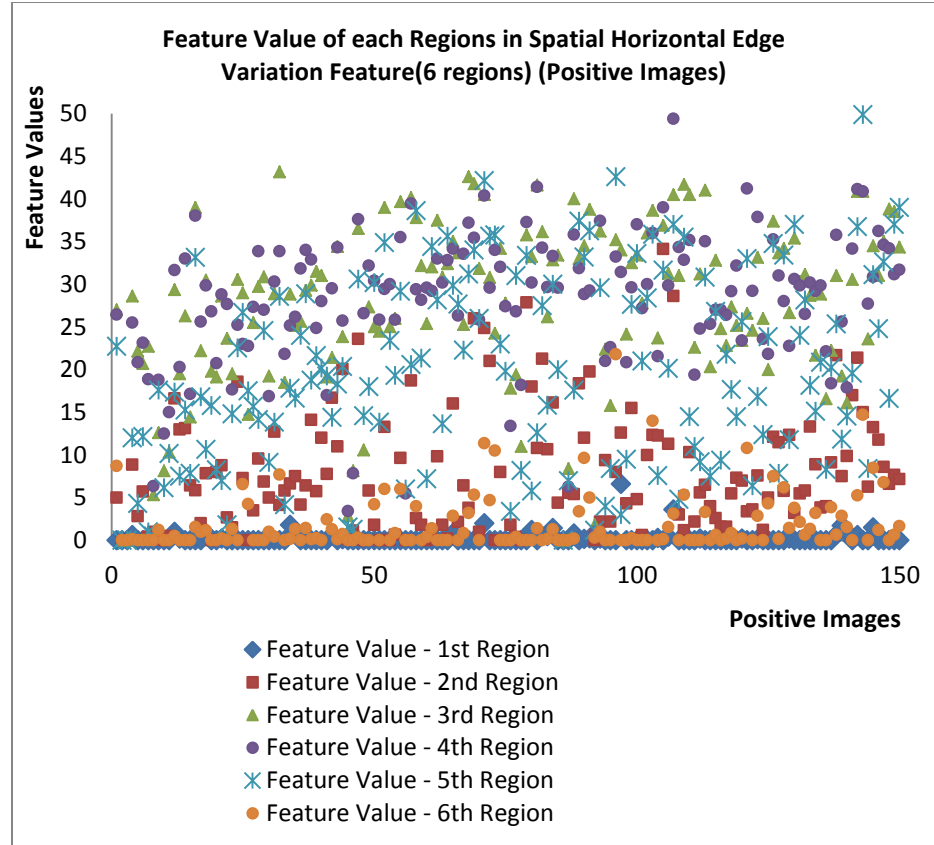


Figure 4.7 Graph with feature value generated for the License Plate Image From the test images (Refer to Figure 4.8)

After that, we perform the same actions to calculate the feature values for the non-license plate images and have a comparison among the two graphs. Refer to figure 4.9 where it shows the graph with feature values of the non-license plate image. From the graph we found that, on average most of the images are having low feature values for each regions of the Spatial Horizontal Edge Variation feature. One of the reasons is the image does not have high pixel changes from the left to the right of the image horizontally. Another reason will because of the threshold for generating the Horizontal Edge Variation image. This threshold will affect the results of the classifier when the image is having low illumination and low contrast. When the contrast of the image is low, then the difference among

the pixels will not be so obvious. This becomes one of the drawbacks of this technique whereby the threshold is fixed during the processing. To overcome this problem, we could normalize the image and provide a dynamic threshold during the processing in the future development.



Figure 4.8 Actual License Plate extracted from the Test Images.

The graph in figure 4.9 shows data scattered different from the graph in figure 4.7. This shows that our designed feature in this situation is able to distinguish between the license plate and non license plate. However, there are still cases where the classifier might misclassify many false positive. So, we could implement our linear classifier that consists of multiple features to further eliminate the false positive as much as possible.

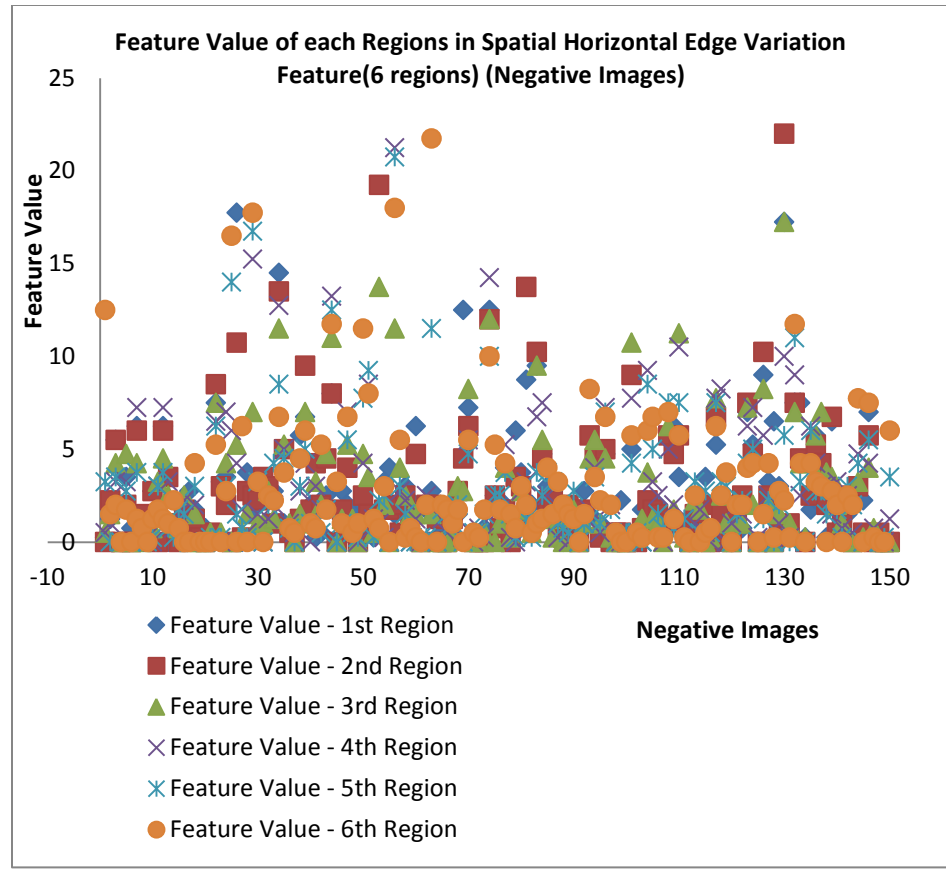


Figure 4.9 Graph with feature value generated for the Non-License Plate Image From the test images (Refer to Figure 4.10)



Figure 4.10 Non License Plate extracted from the Images.

Further on, we do another experiment on the threshold that used to generate the Horizontal Edge Variation Image, and analyze the impact of the threshold on the overall performance for the classifier. Table 4.4 until table 4.7 and figure 4.11 until figure 4.14 shows the results comparison of the Spatial Horizontal Edge Variation feature with different Horizontal Edge Variation Image threshold. Through the observation, we found that the true positive rate actually increase when the Horizontal Edge Variation Image Threshold value is reduced. One of the reasons is because some of the test images that we have are having low contrast and illumination. When this threshold is low, the system will eventually more sensitive as this is similar to binary threshold, where at the same time, the classifier can detect more license plate that are low contrast but it also have more possibilities to detect false positive.

Table 4.4 Results for Classifiers of Spatial Horizontal Edge Variation feature with 1 region corresponding to the different Horizontal Edge Variation Image Threshold.

Number of regions in Spatial Horizontal Edge Variation feature	Horizontal Edge Variation Threshold	False Positive Rate	True Positive Rate
1	30	0.00564	0.9135
1	45	0.00365	0.88186
1	60	0.00217	0.76574

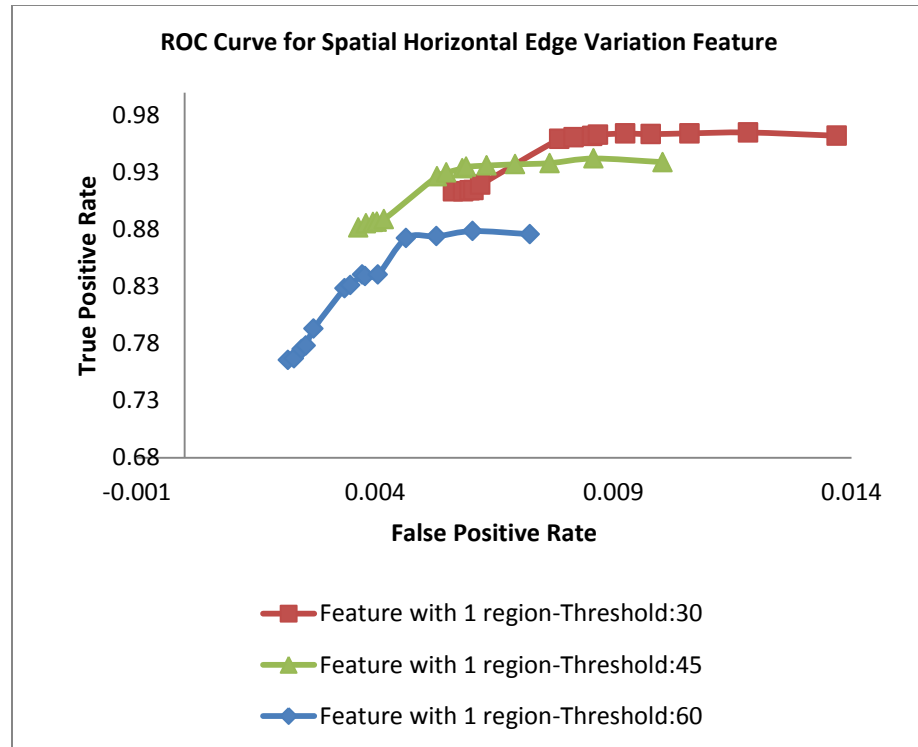


Figure 4.11 shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers. The Graphs show the comparison of feature with 1 region on different Horizontal Edge Variation Image Threshold.

Table 4.5 Results for Classifiers of Spatial Horizontal Edge Variation feature with 3 regions corresponding to the different Horizontal Edge Variation Image Threshold.

Number of regions in Spatial Horizontal Edge Variation feature	Horizontal Edge Variation Threshold	False Positive Rate	True Positive Rate
3	30	0.00379	0.917
3	45	0.00278	0.89686
3	60	0.00182	0.79187

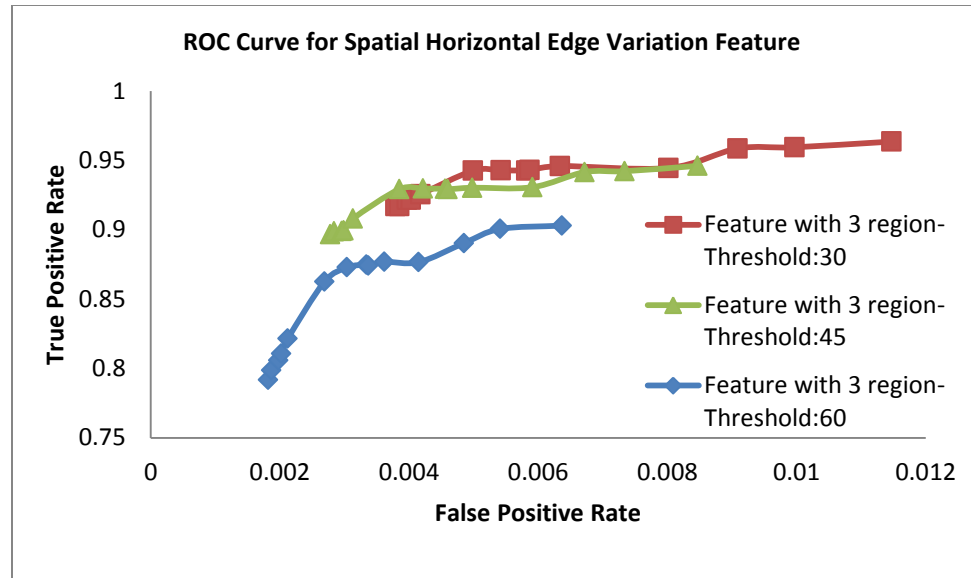


Figure 4.12 shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers. The Graphs show the comparison of feature with 3 regions on different Horizontal Edge Variation Image Threshold.

Table 4.6 Results for Classifiers of Spatial Horizontal Edge Variation feature with 6 regions corresponding to the different Horizontal Edge Variation Image Threshold.

Number of regions in Spatial Horizontal Edge Variation feature	Horizontal Edge Variation Threshold	False Positive Rate	True Positive Rate
6	30	0.00282	0.92666
6	45	0.00234	0.91988
6	60	0.00167	0.81963

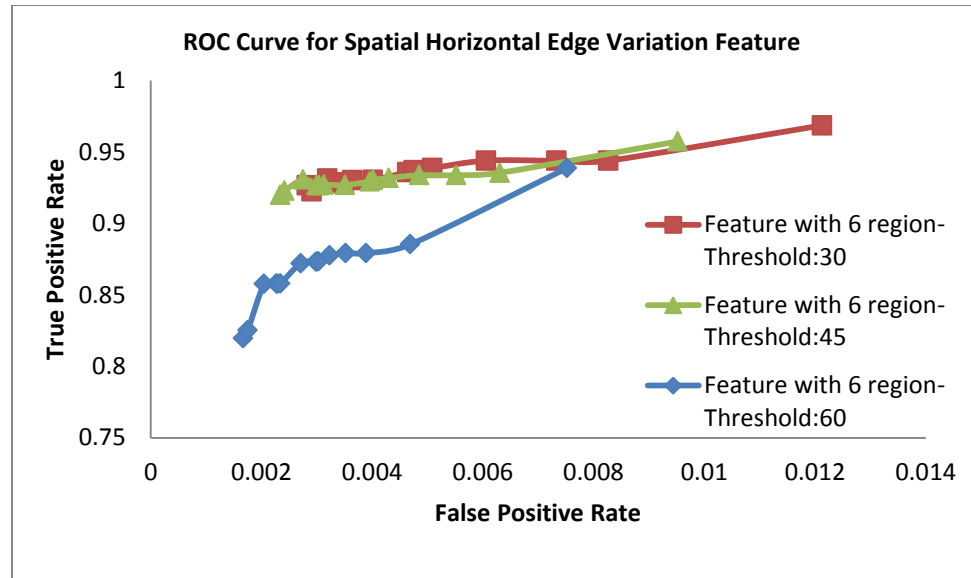


Figure 4.13 shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers. The Graphs show the comparison of feature with 6 regions on different Horizontal Edge Variation Image Threshold.

Table 4.7 Results for Classifiers of Spatial Horizontal Edge Variation feature with 9 region corresponding to the different Horizontal Edge Variation Image Threshold.

Number of regions in Spatial Horizontal Edge Variation feature	Horizontal Edge Variation Threshold	False Positive Rate	True Positive Rate
9	30	0.00259	0.90581
9	45	0.00216	0.89423
9	60	0.00155	0.79209

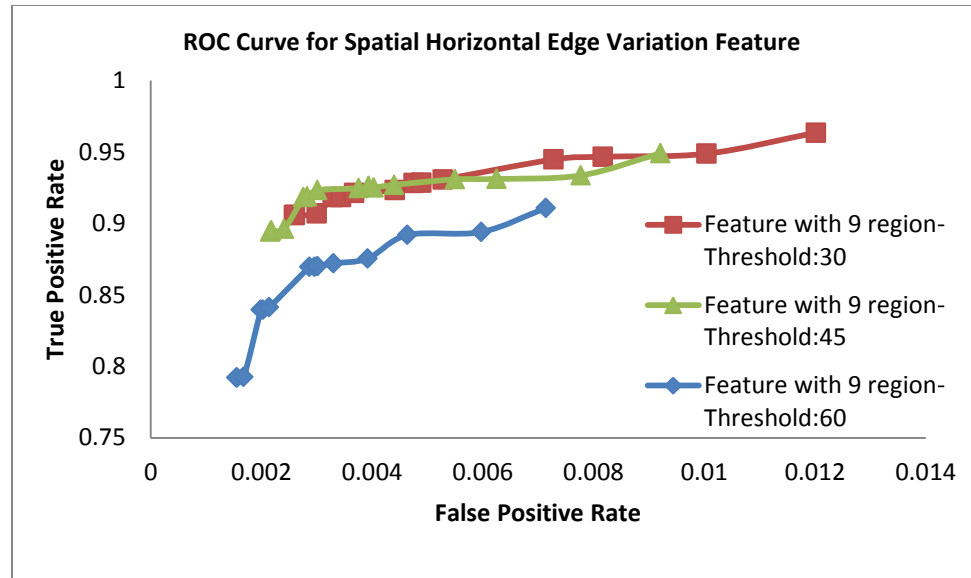


Figure 4.14 shows the ROC curve for the license plate detector with Spatial Horizontal Edge Variation feature as the classifiers. The Graphs show the comparison of feature with 9 regions on different Horizontal Edge Variation Image Threshold.

4.2.2 Experiment: Spatial Horizontal Edge Variation feature on Text Detection

Table 4.8 Experiment Setting for Testing Spatial Horizontal Edge Variation Feature in detecting Text from natural scene.

Experiment	
Hypothesis	Feasibility of Spatial Horizontal Edge Variation Feature in detecting text from natural scene
Objective	To test whether Spatial Horizontal Edge Variation Feature is able to provide high accuracy in detecting text from natural scene
Training Samples	Artificial generated text training set
Testing Samples	ICDAR 03 text testing dataset
Static Variables	Feature Set: Spatial Horizontal Edge Variation Feature Training sample (grayscale image) Positive Sample: artificial generated text Number of positive sample: 1500 Negative Sample: Non license plate sample from internet Number of negative sample: 1000 Testing sample (grayscale image) ICDAR 03 text testing dataset Number of sample:~250 Single-point crossover Single-point mutation Minimum Hit rate: 0.95 Horizontal Edge Variation Threshold: 45, 70

After experiment the Spatial Horizontal Edge Variation feature on vehicle license plate, we trained the text detector with the artificial generated text, and tested the text detector with testing images from ICDAR 03 text testing dataset. We generated 1500 of positive samples of artificial text, and 1000 of negative samples of non text images from internet.

Similar to the experiments we did for the license plate detector with Spatial Horizontal Edge Variation feature, we also selected 2 random thresholds (Horizontal Edge Variation Threshold) for our experiments on text detection. Then, we used the similar setting, and features set are randomly initialized by the framework based on the minimum resolution window size defined before the training started. The major changes in this training will be the Text samples, and providing these text samples to the framework, it will select the suitable features for generating the final classifier. The following are tables and graphs illustrating the results obtained for the text detector created by our framework.

Table 4.9 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 30 pixels.

Number of Regions in Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
1 bin	45	30	0.56256	0.32939
3 bins	45	30	0.28876	0.30128
6 bins	45	30	0.34254	0.34732
9 bins	45	30	0.35290	0.33286

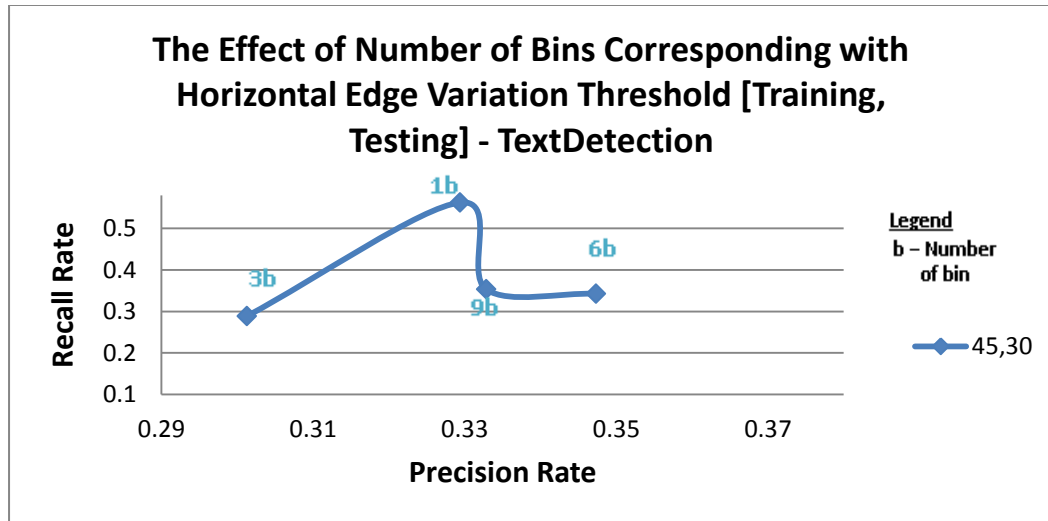


Figure 4.15 shows the recall rate and the corresponding precision rate of four text detectors [1 bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 30 pixels.

Table 4.10 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 45 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
1 bin	45	45	0.48538	0.37093
3 bins	45	45	0.26694	0.30552
6 bins	45	45	0.31903	0.35583
9 bins	45	45	0.32033	0.34599

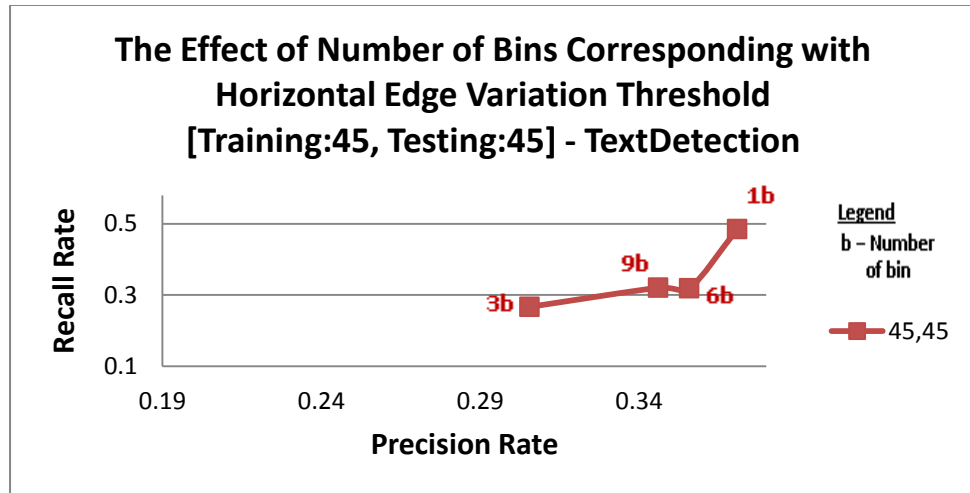


Figure 4.16 shows the recall rate and the corresponding precision rate of four Text detectors [1 bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 45 pixels.

Table 4.11 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 60 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
1 bin	45	60	0.27843	0.27713
3 bins	45	60	0.13911	0.19718
6 bins	45	60	0.18074	0.24187
9 bins	45	60	0.19703	0.26405

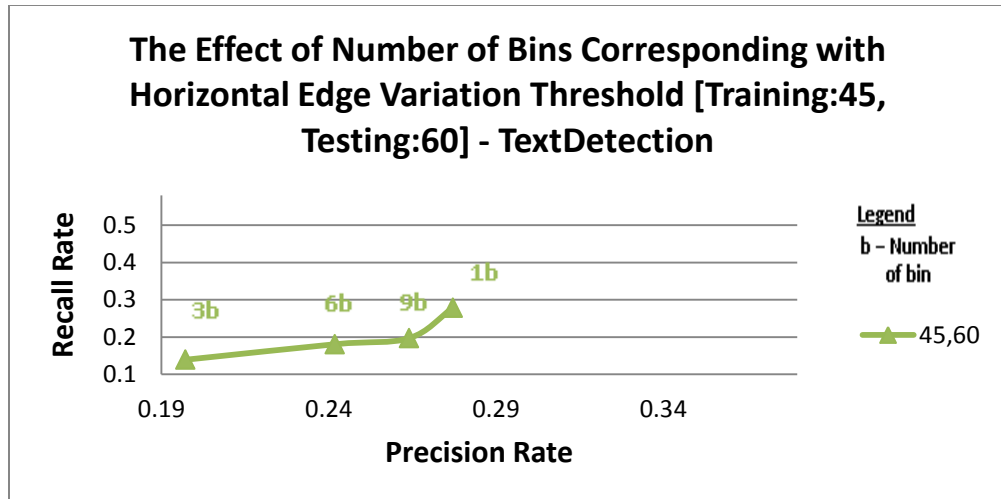


Figure 4.17 shows the recall rate and the corresponding precision rate of four Text detectors [1 bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 60 pixels.

Table 4.12 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 70 pixels and horizontal edge variation image testing threshold of 45 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
3 bins	70	45	0.41955	0.35427
6 bins	70	45	0.49162	0.37597
9 bins	70	45	0.51605	0.36407

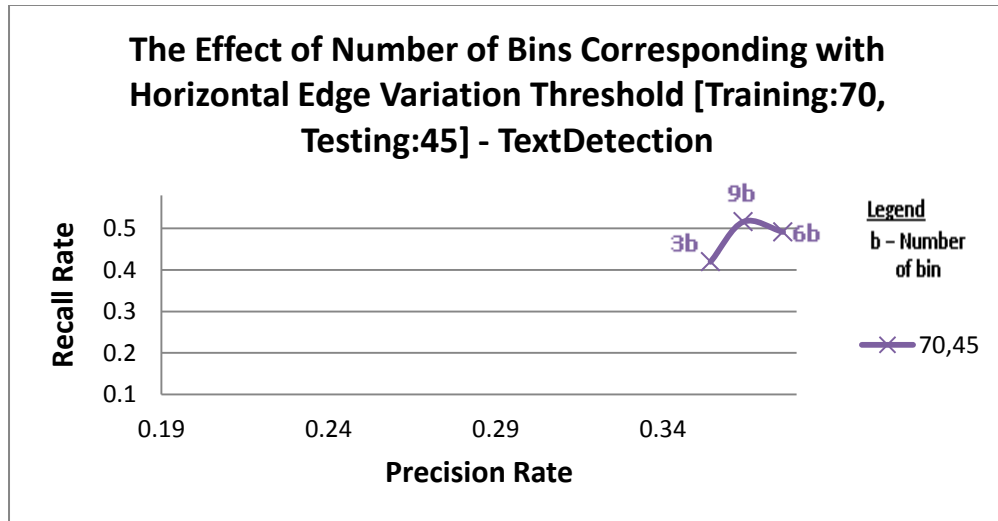


Figure 4.18 shows the recall rate and the corresponding precision rate of three Text detectors [3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 70 pixels, and testing with threshold of 45 pixels.

Table 4.13 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector. The results show the difference between horizontal edge variation image training threshold of 70 pixels and horizontal edge variation image testing threshold of 70 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
3 bins	70	70	0.28238	0.29896
6 bins	70	70	0.36924	0.34194
9 bins	70	70	0.42393	0.34972

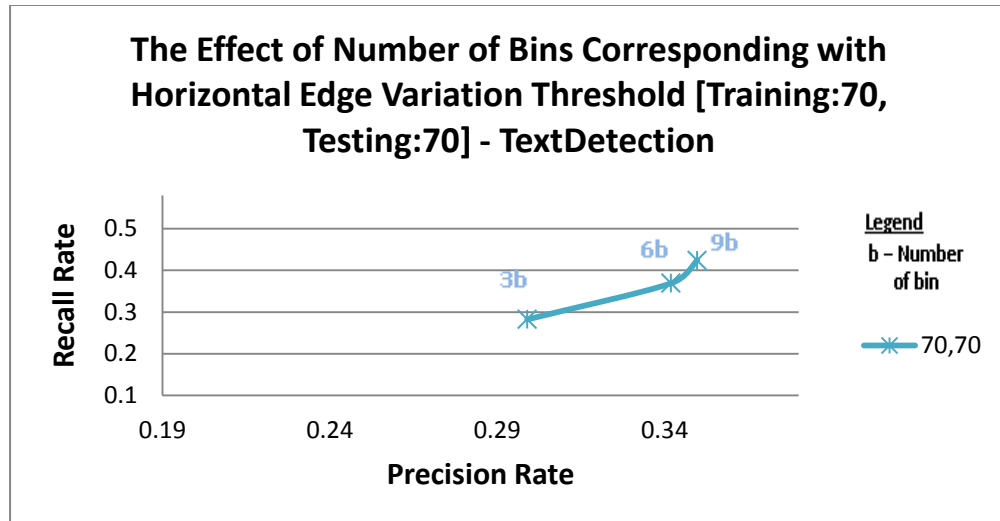


Figure 4.19 shows the recall rate and the corresponding precision rate of three Text detectors [3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 70 pixels, and testing with threshold of 70 pixels.

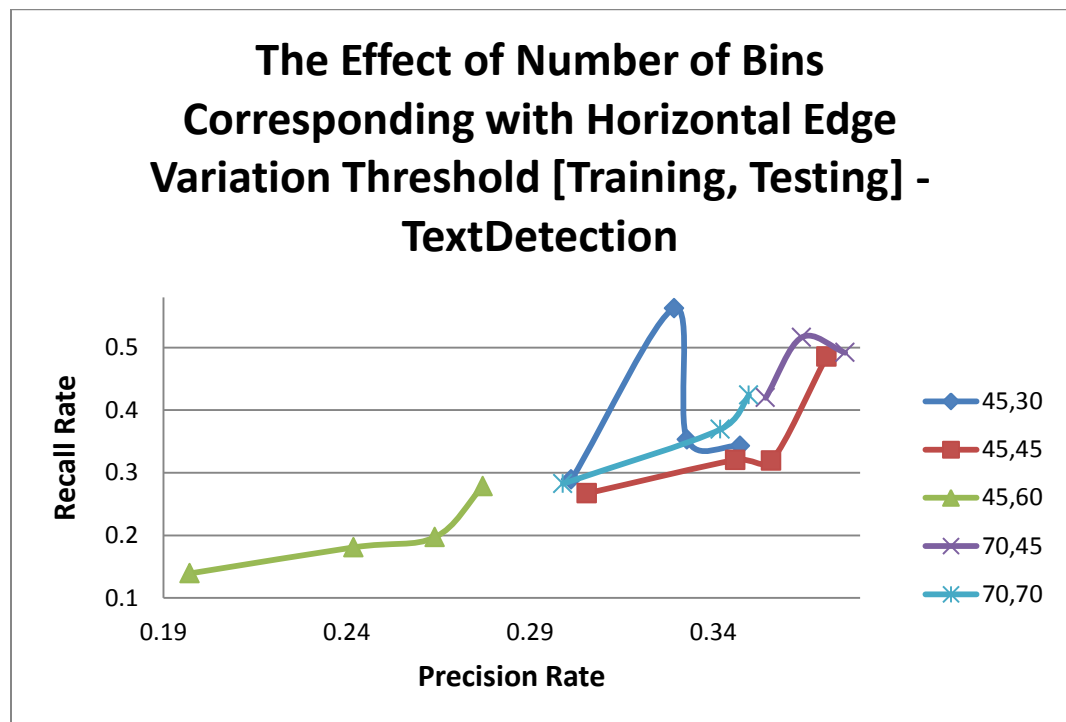


Figure 4.20 shows the combinations of all the graphs from Figure 4.15 until Figure 4.19. The recall rate and the corresponding precision rate of Text detectors with different training threshold and testing threshold and different number of regions [1 bin, 3 bins, 6 bins, and 9 bins] are shown from the graph.

From the results, we observe that the text detector created by our framework unable to outperform other approaches such as the method by (Ezaki, Bulacu, & Schomaker, 2004) , they implemented different techniques and combined them together to improve the text detection. As the size and variation of the text from the natural scene images can be varies for different scenes, and complex objects from the natural scene can be a challenge for the system to extract the text accurately. The other challenge of the natural scene is the changes of the environment that will affect the lighting and illumination, which increase the false or missed detection. They proposed the four methods to extract the characters from the natural scene images and their morphological method is able to obtain recall rate of 55% and precision rate of 38%. Compare with our approach, our technique failed when the image is only containing a single character or character with low brightness and lighting reflection. One of the reasons that caused the failure is the scanning window resolution size for our algorithm is fixed with the same height width ratio. Therefore, during our training process, we only provide text samples with similar height width ratio. From here we can actually notice that our technique might not be so adequate for the text detection in natural scene image due to the variety changes on the text in term of size, format, background, color, height width ratio and etc.

4.2.3 Experiment: Spatial Horizontal Edge Variation feature on side view car detection

Table 4.14 Experiment Setting for Testing Spatial Horizontal Edge Variation Feature in detecting UIUC Car Dataset.

Experiment	
Hypothesis	Feasibility of Spatial Horizontal Edge Variation Feature in detecting side view car
Objective	To test whether Spatial Horizontal Edge Variation Feature is able to provide high accuracy in detecting side view car
Training Samples	UIUC Car Training Dataset
Testing Samples	UIUC Car Testing Dataset
Static Variables	Feature Set: Spatial Horizontal Edge Variation Feature Training sample (grayscale image) Positive Sample: UIUC Car Training Dataset Number of positive sample: 550 Negative Sample: Non license plate sample from internet Number of negative sample: 550 Testing sample (grayscale image) UIUC Car Testing Dataset Number of sample:~170 Single-point crossover Single-point mutation Minimum Hit rate: 0.95 Horizontal Edge Variation Threshold: 45, 90

After we tested our proposed approach on the vehicle license plate problems and detecting text from natural scene image, we do a simple experiment on side-view car detection problem to determine whether the technique we apply on license plate detection can be applied on side-view car detection. Similar to the experiment previously on license plate detector and text detector, in this

experiment on side view car detection, features set are randomly initialized by the framework based on the minimum resolution window size defined before the training started. We used the UIUC Car dataset as the training samples for the framework; it selects the suitable features to generate the final classifier. The following are tables and graphs illustrating the results obtained for the side-view car detector created by our framework.

Table 4.15 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 30 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
1 bin	45	30	0.86984	0.37680
3 bins	45	30	0.85686	0.38155
6 bins	45	30	0.68986	0.34566
9 bins	45	30	0.72799	0.34426

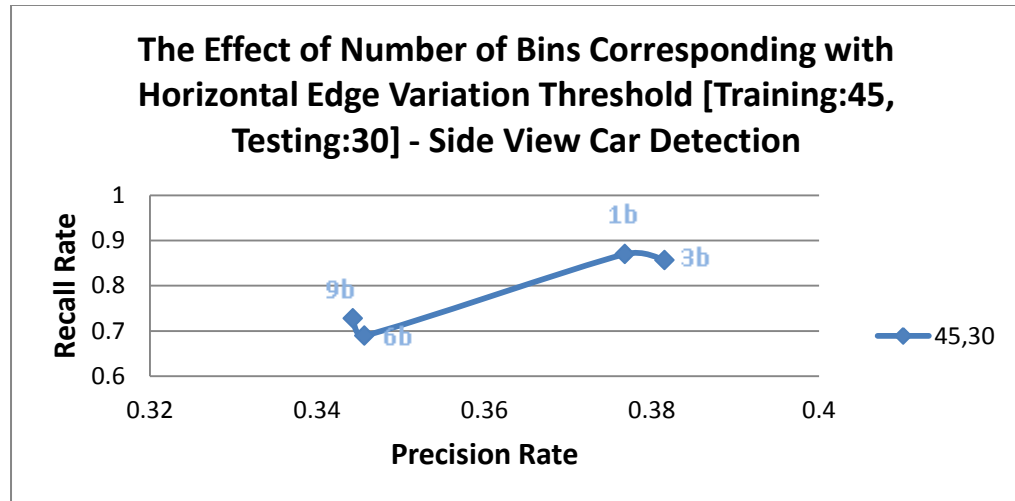


Figure 4.21 shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 30 pixels.

Table 4.16 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 45 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
1 bin	45	45	0.85497	0.32193
3 bins	45	45	0.81654	0.31578
6 bins	45	45	0.77164	0.33511
9 bins	45	45	0.76227	0.31087

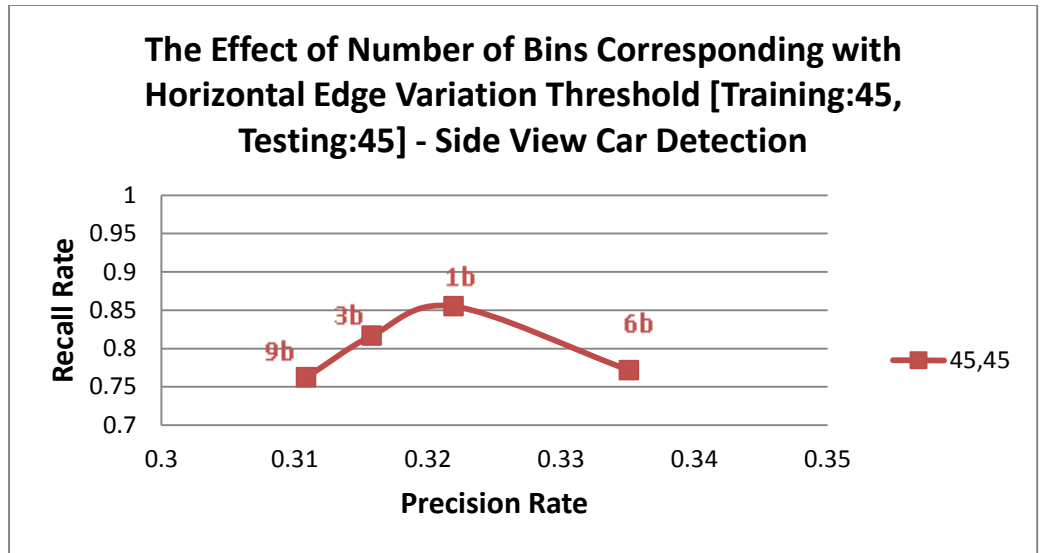


Figure 4.22 shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 45 pixels.

Table 4.17 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 45 pixels and horizontal edge variation image testing threshold of 60 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
1 bin	45	60	0.79529	0.32293
3 bins	45	60	0.74452	0.31158
6 bins	45	60	0.66439	0.31911
9 bins	45	60	0.65930	0.31242

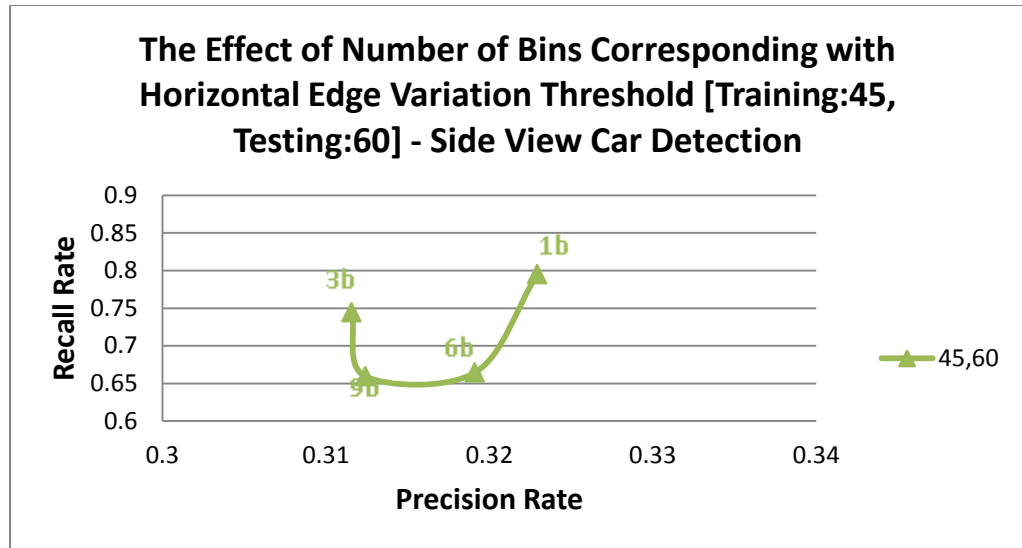


Figure 4.23 shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 45 pixels, and testing with threshold of 60 pixels.

Table 4.18 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 90 pixels and horizontal edge variation image testing threshold of 45 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
1 bin	90	45	0.90289	0.37834
3 bins	90	45	0.92962	0.35769
6 bins	90	45	0.92850	0.33422
9 bins	90	45	0.92986	0.33599

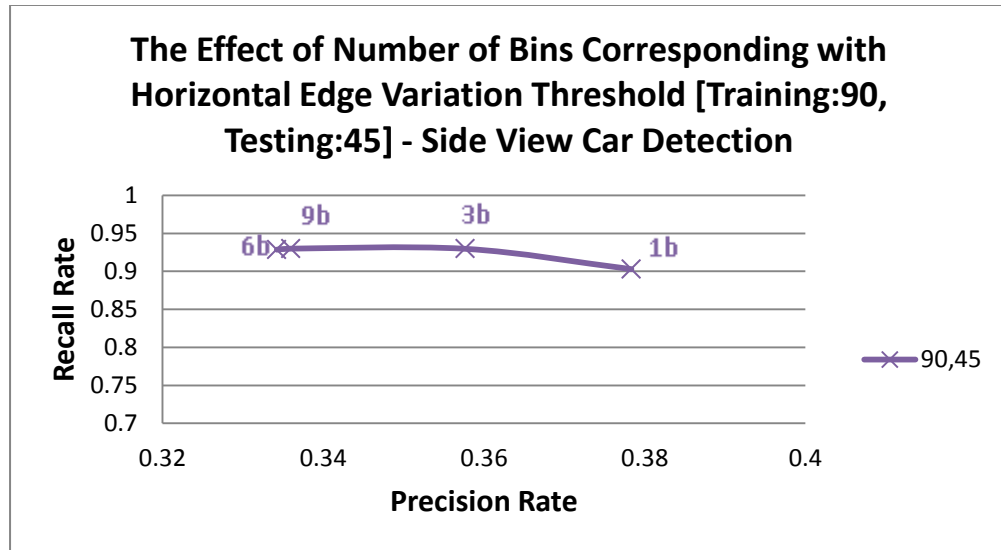


Figure 4.24 shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 90 pixels, and testing with threshold of 45 pixels.

Table 4.19 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 95 pixels and horizontal edge variation image testing threshold of 70 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
1 bin	90	70	0.91348	0.35122
3 bins	90	70	0.92305	0.35202
6 bins	90	70	0.90611	0.34240
9 bins	90	70	0.64767	0.20037

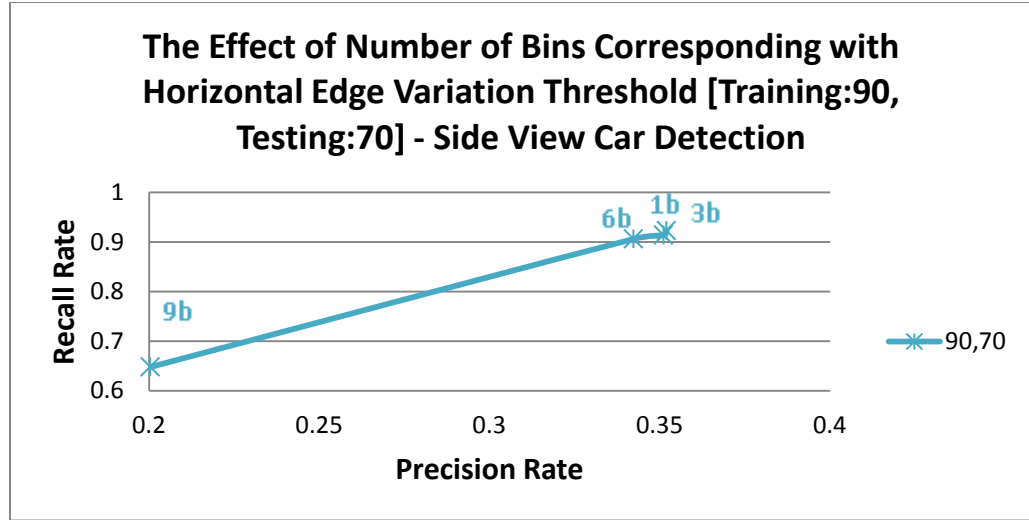


Figure 4.25 shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 90 pixels, and testing with threshold of 70 pixels.

Table 4.20 Results from the experiments of testing how the number of regions in Spatial Horizontal Edge Variation features affecting the performance of the detector (UIUC Car Dataset). The results show the difference between horizontal edge variation image training threshold of 90 pixels and horizontal edge variation image testing threshold of 90 pixels.

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
1 bin	90	90	0.85065	0.36397
3 bins	90	90	0.85724	0.36676
6 bins	90	90	0.84675	0.36479
9 bins	90	90	0.84872	0.36571

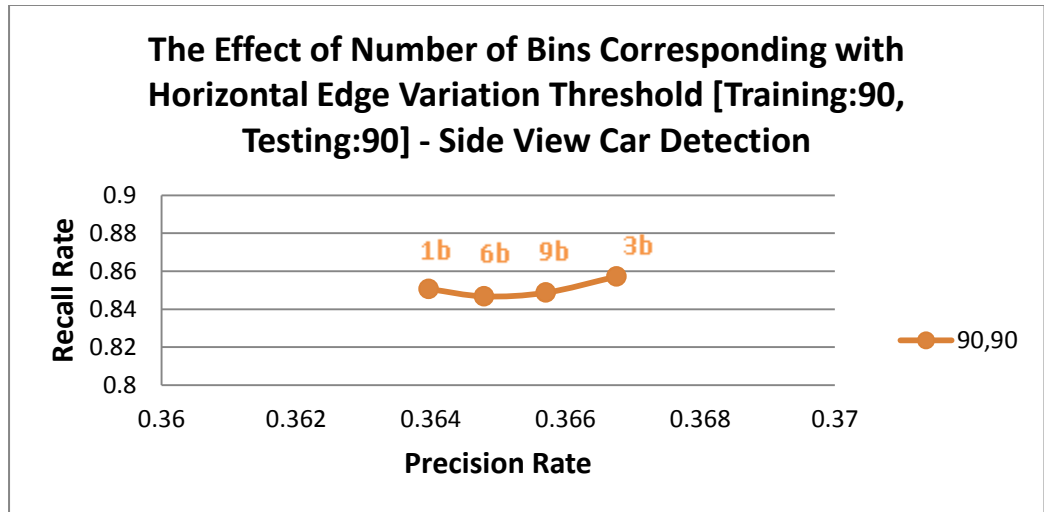


Figure 4.26 shows the recall rate and the corresponding precision rate of four Side View Car detectors [1bin, 3 bins, 6 bins, and 9 bins]. The detectors were trained with the training threshold of 90 pixels, and testing with threshold of 90 pixels.

Overall, the Spatial Horizontal Edge Variation feature with 9 regions and trained with threshold of 90 pixels and test with threshold of 45 pixels obtained the highest recall rate of 0.92986 and precision rate of 0.33599. The precision rate is low showing that the detector generates a lot of false detection, and from the results we found that most of the false detections are actually overlapped of the detection window onto the areas of the car, and duplicates of the detected window were not combined for this preliminary test. Our main purpose of testing the license plate detection learning framework onto text detection and side view car detection is to observe whether the Spatial Horizontal Edge Variation features set can be applied for other object detection besides license plate detection. For this preliminary test on text and side-view car detection, we found that the training sample is very important as the framework based on the training sample to create and select the features from the large feature set and based on the training samples to set the threshold for the features. However, there are still a lot of improvements

can be done on the framework to increase the recall rate and precision rate for the license plate detection. In the next section will be further discussions on the algorithm framework as well as the feature set.

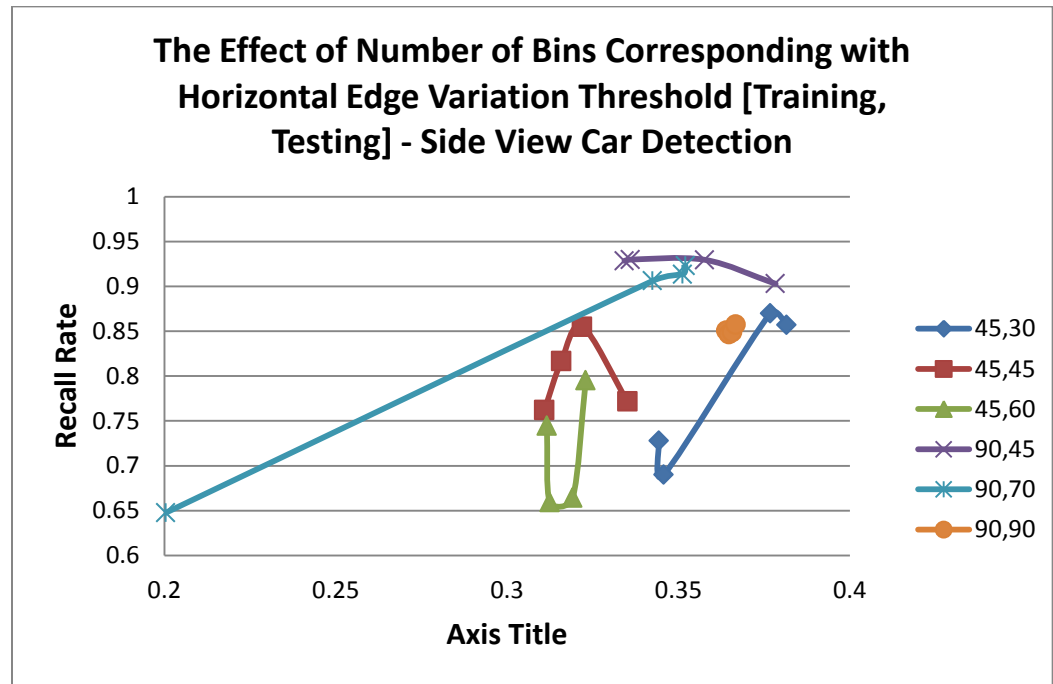


Figure 4.27 shows the combinations of all the graphs from Graph 1.14 until Graph 1.19. The recall rate and the corresponding precision rate of Side View Car detectors with different training threshold and testing threshold and different number of regions [1 bin, 3 bins, 6 bins, and 9 bins] are shown from the graph.

4.3 Discussions

In this section, we are going to have some discussions on the approach, experiment results, and some analysis on the failure detection. From previous experiments, we obtained some results that can help us to further explain in this section. Some of the discussions include how the number of regions in Spatial Horizontal Edge Variation feature can be varies in the detection results, and then we discuss the differences between the threshold that we set for Horizontal Edge Variation Image during the training and testing. Further on, we will analyze those failure cases where the detector failed to locate the target object, and why some of the detector failed to detect the target object. Analysis will be done in understanding the features selected by the system for each detector, and identify any techniques to tune the system in order to select more feasible features during genetic algorithm feature search. After that, we also determine any techniques that are able to improve the precision rate by removing more false detection, for example by implementing Support Vector Machine as the second stage false detection removal.

4.3.1 How numbers of Regions of Spatial Horizontal Edge Variation Feature Affect the Detection Result?

The definition of the number of regions for the Spatial Horizontal Edge Variation feature is number of regions that divide the sub-rectangle of Spatial Horizontal Edge Variation feature into few sections. Each section generates the average of the total pixel changes from the left to right of the image. From the

values we get through Spatial Horizontal Edge Variation feature; it extracts the patterns of the pixels changes from the sub window feature. It shows the wave of the pixels changes inside the rectangle of the feature. Spatial Horizontal Edge Variation feature is showing the density of the pixel changes from the left to right of the image horizontally.

For the vehicle license plate detection problem, our experiments show that the detector trained with 6 bins of Spatial Horizontal Edge Variation feature and training threshold of 45 pixels provide the highest accuracy among the others. It obtained a recall rate of 0.91988 and a precision rate of 0.53373. Figure 4.28 shows the Spatial Horizontal Edge Variation feature selected for the 6 bins of regions. Each of the features selected plays a role of removing different false images from the sub-window; for example, the image starts the window scanning by moving the sub-window through the image and for each sub-window, the cascaded features will determine whether the particular sub-window contains any possible license plate. For all the features selected during the learning process, they will be combined into a linear structure. Then, the first feature in the list start to determine whether the sub-window containing any license plate, if the sub-window contains any license plate, then the next feature will continue to further verify the sub-window; however, if any features classifies that the sub-window as a non license plate, then it will directly reject the sub-window for any further processing and the sub-window will move to the next positions. This method can reduce the processing time as they do not need to execute all the features in the cascaded list.

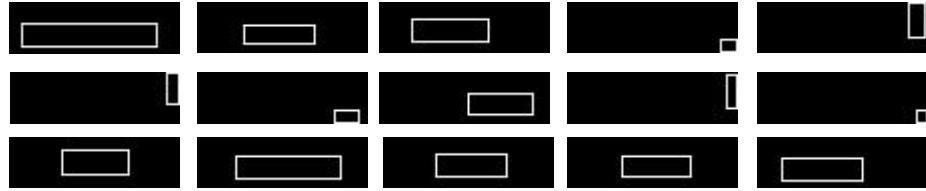







Figure 4.28 Diagram shows the Spatial Horizontal Edge Variation features selected by the system, where each black rectangle is the sub-window with 85 x 25 with a sub rectangle internally (white color). These features are selected for vehicle license plate detection, with 6 regions of the Spatial Horizontal Edge Variation feature.

Now we will analyze on one of the vehicle license plate detector trained. We select the vehicle license plate detector trained on artificial license plate samples. The final cascaded feature constructed by 15 Spatial Horizontal Edge Variation features. We can review how each feature reacts on the different sub-window from the images. The 15 features are showed at Figure 4.28, where each of the features is used to filter all the non-license plate and remain those candidates' license plate regions. So, we had taken few samples of images to analyze how each feature reacts to those images. Table 4.21 shows the images that we used for the analysis, where they consist of artificial vehicle license plate (used for training), a real vehicle license plate, 3 false images we extracted from the real world testing image.

Table 4.21 Table shows the images that we used for the analysis on how each feature in the cascaded feature react on each images.

Image Type	Image
Artificial Vehicle License Plate	
Real Vehicle License Plate	
1 st False Images from real world testing image	
2 nd False Images from real world testing image	
3 rd False Images from real world testing image	

Here, we are going to explain how all these work. First of all, we provide all the images shown at Table 4.21 and used the vehicle license plate detector generated by our learning framework to classify whether the image contain any license plate. The vehicle license plate detector consists of a list of features combined in a cascaded format, where the detector start the license plate detection process with the first feature in the list and if the first feature classify the image as a non license plate then the detector will stop the processing. However, if the first feature classifies the image as a possible license plate candidate, then the detector will continue to process by passing the image to the second feature in the cascaded feature. This process continue until the last feature of the detector, if all the features classify the image as a license plate, then the detector will conclude that the image as a license plate. The drawback of this technique is that, if the

license plate image is rejected by any of the feature, then the detector will make the wrong decision.

Figure 4.29 shows the graph of the distribution for the feature values of each bin in the first feature (Spatial Horizontal Edge Variation feature). There are 6 values generated by the feature for each images, this is because there are 6 regions in the Spatial Horizontal Edge Variation feature. From the graph pattern, we found that the first false image has the highest feature values for the 6 regions among all the images. The first false image in the list is removed as the first feature classifies it as a non license plate. Therefore, the rest of the features in the cascaded feature list do not need to perform the classification onto the first false image.

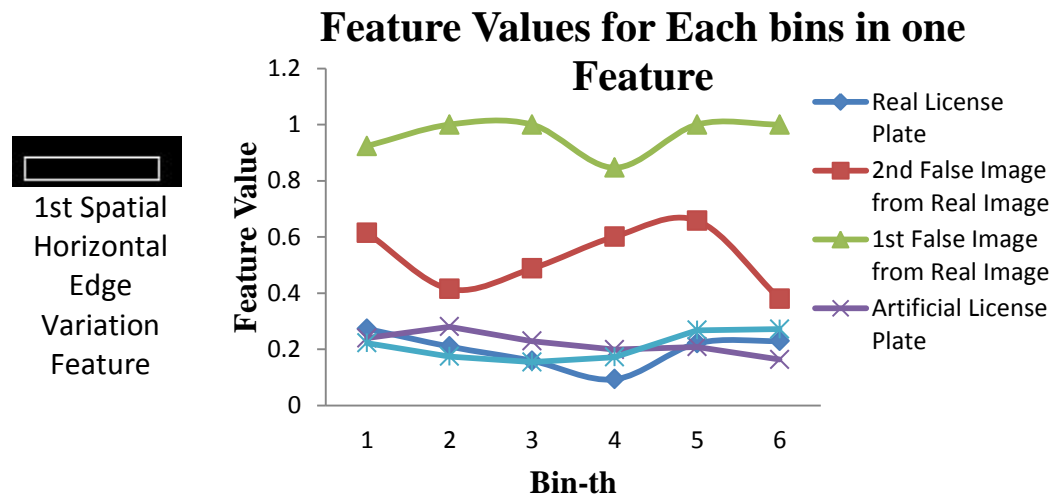


Figure 4.29 The Graph shows the distributions of the feature values of each bin in the 1st Spatial Horizontal Edge Variation feature from the cascaded features (vehicle license plate detector)

From the graph pattern, we found that the feature generate the highest feature value for the second false image compare with the rest of the images. However, the second false image in the list is not removed from the processing as

the second feature still classifies it as a possible license plate. Therefore, the next feature in the cascaded feature list will continue to classify the second false image and the others.

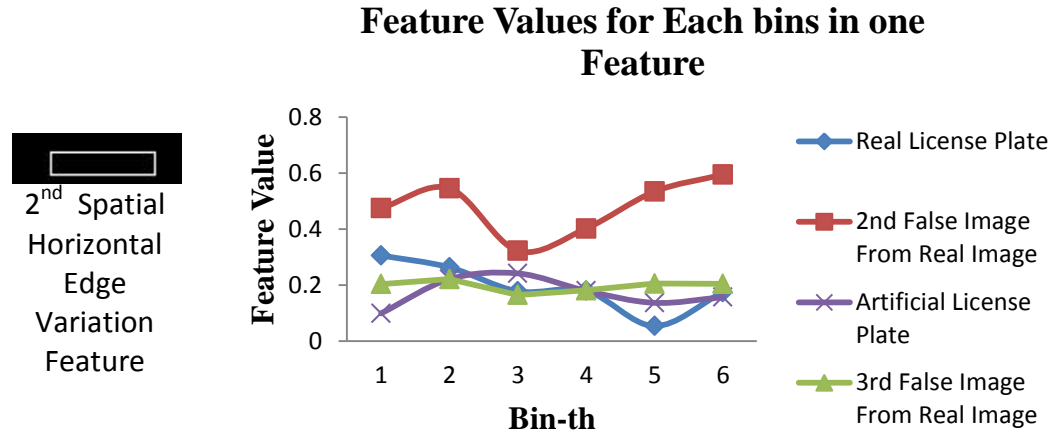


Figure 4.30 The Graph shows the distributions of the feature values of each bin in the 2nd Spatial Horizontal Edge Variation feature the cascaded features (vehicle license plate detector)

Next, we continue to analyze 3rd Spatial Horizontal Edge Variation feature onto the all the images, and Figure 4.31 is showing the graph of the distribution for the feature values of each bin in the third feature (Spatial Horizontal Edge Variation feature). From the graph pattern, we found that all the images is classified as possible license plate, so the next feature in the cascaded feature list will continue to classify all the image.

After that, when come to the 4th Spatial Horizontal Edge Variation feature in the cascaded feature list, the Figure 4.32 shows that the values for every regions is the highest among the others when classifying the 2nd false image. Meanwhile, the rest of the image gives lower feature values for each bin. Therefore, the 4th feature in the cascaded feature list based on the threshold,

classify the 2nd false image as the non license plate, and remove it from continue been execute by the rest of the feature.

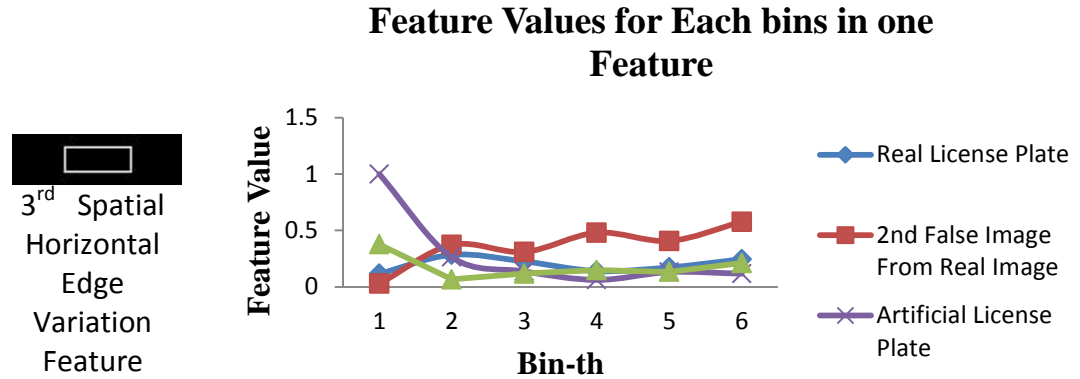


Figure 4.31 The Graph shows the distributions of the feature values of each bin in the 3rd Spatial Horizontal Edge Variation feature from the cascaded features (vehicle license plate detector)

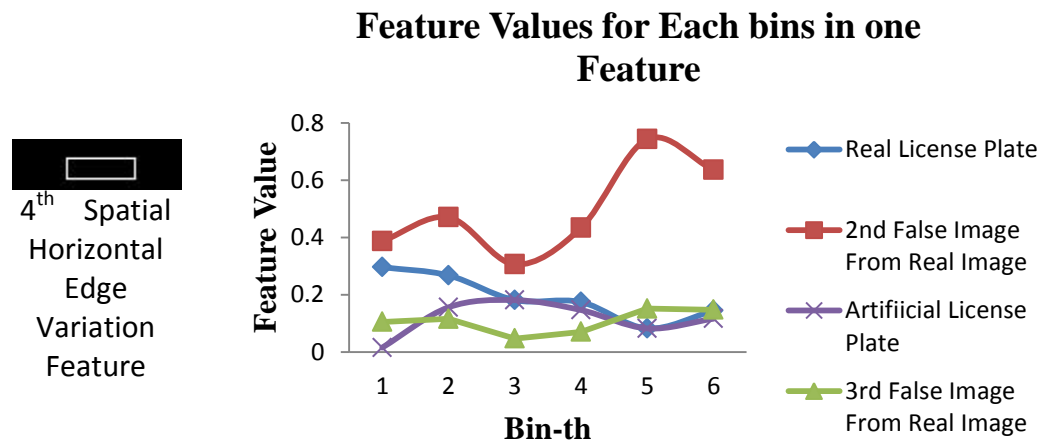


Figure 4.32 The Graph shows the distributions of the feature values of each bin in the 4th Spatial Horizontal Edge Variation feature from the cascaded features (vehicle license plate detector)

Finally, we observe the overall feature value distributions patten from the graph in Figure 4.33. This graph is different from the graph previously, where the x axis of the graph shows each Spatial Horizontal Edge Variation features in the cascaded feature sequentially, and the y axis of the graph is the feature value

extracted by each feature from the images. This feature value is generated from comparing the 6 values of each feature with the feature template (6 bins). This graph is the summary of the previous graphs, where we can see that the 1st false image is removed by the first feature as the feature value computed on this image is more the feature threshold.

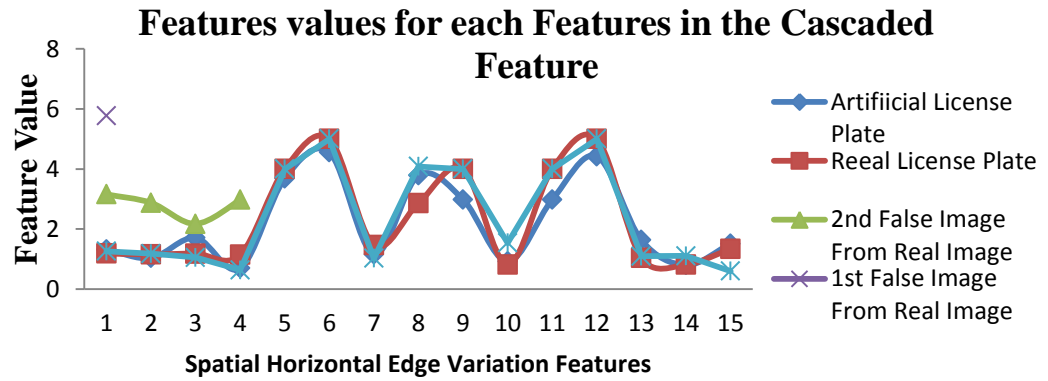


Figure 4.33 The Graph shows the feature values of each Spatial Horizontal Edge Variation feature from the cascaded features (vehicle license plate detector). The feature values from this graph is different from the graph previously, where the feature values for each feature is computed from comparing the 6 bins of each feature with the feature template (6 bins).

Then for the 2nd false image, the 4th features classify it as non license plate and remove it from the list. 2 of the false images in the analysis are already removed by the 1st and 4th feature from the cascaded feature list, but the 3rd false image passed through the entire feature and is classified as the license plate. This is a false positive generated from the detector, and from the graph pattern we found that the 3rd false image is having a similar pattern like the artificial license plate and the real license plate.

Table 4.22 Comparison between the original image and Horizontal Edge Variation Image (with threshold of 45 pixels)

Original Image					
Horizontal Edge Variation Image					

For the images from Table 4.21 , we observe that the image is having high pixel changes from the left to the right of the image horizontally, which might caused the detector wrongly classify the object as the target object. Table 4.22 shows the horizontal edge variation image generated by the system before the classification started. For the cascaded feature constructs of Spatial Horizontal Edge Variation features, we need to pre-process the original image into a special image type named horizontal edge variation image. This image type is specifically design for the feature where the edges of the license plate are much clearer for the detection, and at the same time noise can be removed through this image type.

There are more examples of how the horizontal edge variation image affects the detection results at Table 4.23 and Table 4.24. Both tables shows the conversion of the original image to the horizontal edge variation image type and we can observe that after the processing and conversion, the image removes a lot noises from the image and remain with the license plate edges. So, from the analysis, we found that the pattern similar to the license plate image will be classified as the license plate, where they have the high density of the pixel changes from the left to right of the image horizontally.

Table 4.23 Comparison between the original image (large image) and Horizontal Edge Variation Image (with threshold of 45 pixels)



Original Image	Horizontal Edge Variation Image (threshold – 45 pixels)
	

Table 4.24 Comparison between the original image and Horizontal Edge Variation Image (with threshold of 45 pixels)

Original Image	Horizontal Edge Variation Image (threshold – 45 pixels)
	

Next, we continue to have a similar discussion on how the Spatial Horizontal Edge Variation feature performing the classification on side view car and non side view car. Figure 4.34 shows twenty Spatial Horizontal Edge Variation feature with 9 regions and trained with a training threshold of 90 pixels. These twenty features are cascaded into a sequence order and used to perform classification on side view car and non side view car. Again, we selected some images for us to do analysis on each of the feature, refer to table 4.26.

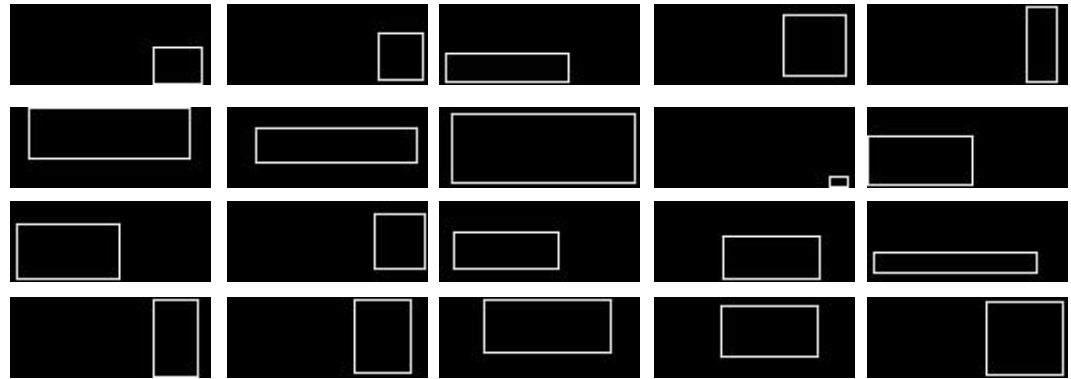






Figure 4.34 Diagram shows the Spatial Horizontal Edge Variation feature (9 bins) selected by the system for side view car detection, where each black rectangle is the sub-window with 100 x 40 with a sub rectangle internally (white color).

The results of the analysis are shown from the Figure 4.35 , where the graph shows the feature values generated by each feature onto the different images we provided for analysis. From the graph, we actually found that one of the features is providing a feature value of 0 for all the images tested, where we can assume that this feature is not improving the accuracy of the classification and we can even remove it from the cascaded feature list. Non car image 1 is removed by the last feature of the cascaded feature, and non car image 2 becomes the false

detection of the result, this is because the entire features classify this image as the target object.

Table 4.25 Table shows the images that we used for the analysis on how each Spatial Horizontal Edge Variation feature in the cascaded feature react on each images (car and Non Car Image).

Image Type	Image
Car Image 1	
Car Image 2	
Non Car Image 1	
Non Car Image 2	

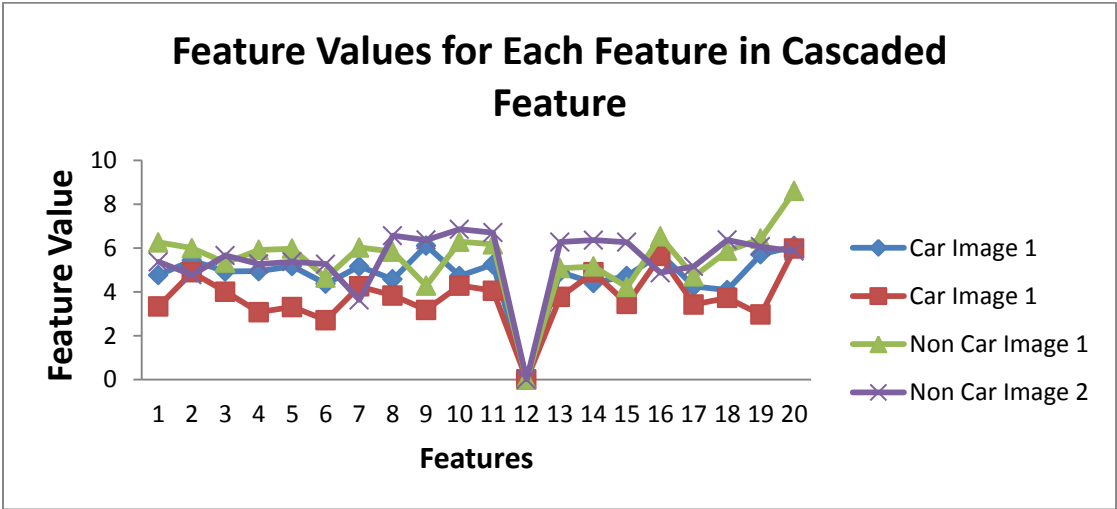


Figure 4.35 Graph shows the features values generated by each Spatial Horizontal Edge Variation Features (9 bins) from the cascaded features list. Any images that are classified as non car will not be processed by the rest of the features in the list.

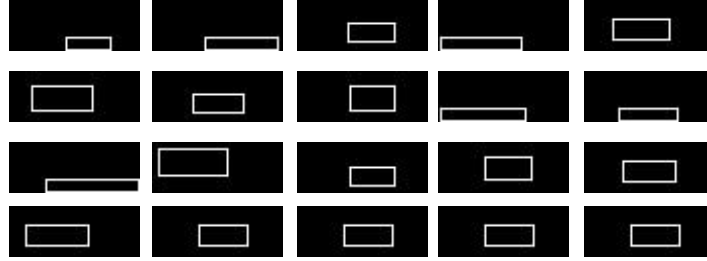


Figure 4.36 Diagram shows the Spatial Horizontal Edge Variation features (1 bin) selected by the system for text detection, where each black rectangle is the sub-window with 65 x 25 with a sub rectangle internally (white color).

After analyzing the performance of the Spatial Horizontal Edge Variation feature on the vehicle license plate and side view car detection, here we continue analysis on the Spatial Horizontal Edge Variation feature selected by the learning framework for text detection. The Spatial Horizontal Edge Variation feature that we selected is 1 region and Figure 4.36 shows all the features from the cascaded feature list for text detection. Table 4.26 shows the images we selected for the analysis on the Spatial Horizontal Edge Variation feature for text detection.





Image Type	Image
Text Image 1	
Text Image 2	
Text Image 3	
Non Text Image 1	
Non Text Image 2	
Non Text Image 3	

Table 4.26 Table shows the images that we used for the analysis on how each Spatial Horizontal Edge Variation feature in the cascaded feature react on each images (text and Non text Image).

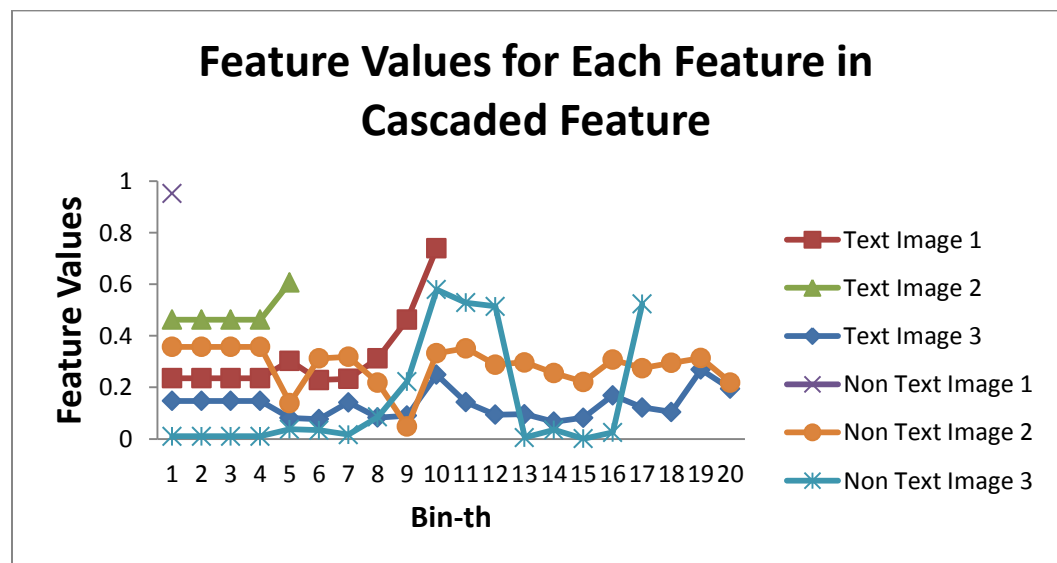


Figure 4.37 Graph shows the features values generated by each Spatial Horizontal Edge Variation Features (1 bin) from the cascaded features list. Any images that are classified as non car will not be processed by the rest of the features in the list.

From the results, we found that the detector failed to classify the text image 1 and text image 2 as the target object, and they were classified as non text by the fifth and tenth feature respectively. One of the reasons maybe is because the training samples for the text detection are not normalized and the training samples maybe not sufficient. Then, only the text image 3 is classified correctly as the target, and there is a false detection which is the non text image 2. This non text image 2 is wrongly classified by Spatial Horizontal Edge Variation feature. The variation of the brightness and the differences of the background and foreground text is affecting the results, from the test samples we can observe that the text that having similar color tone or brightness with the surrounding background can make the detector to classify wrongly. This is because our Spatial Horizontal Edge Variation feature is specifically extracting the information of the area that has high pixel changes from the left to right of the image horizontally. Therefore, we can see one of the weaknesses of this feature is the lightness illumination of the object and its background can give an impact to the detection result.

4.3.2 How Training Image Type Affect the Detection Result?

In this section, we will have a simple discussion on how different image type will affect the accuracy of the detector. Previous experiment we tested the algorithm by training the detector with the training samples consists of grayscale images. Here, we modify the grayscale image by further process the grayscale image into an edge image. The purpose we are doing this is to check how the

algorithm react through the edge images. We will observe the changes on our artificial license plate detection. First of all, we implemented Canny Edge function from OpenCV, where there are two thresholds for adjusting the edges in the image, the threshold with the minimum value is used to edge linking, and the threshold with the maximum value is used to locate the initial segment of the strong edges. (Intel Open Source Computer Vision Library, 1999-2001)

We did an experiment for the Spatial Horizontal Edge Variation feature on the Canny Edge function from OpenCV with a minimum threshold value of 128 and a maximum threshold of 255. The results of the experiment shows from Table 4.27, and we can see that the recall rate of the license plate detector trained with Spatial Horizontal Edge Variation feature, where the recall rate of the detector trained with 9 regions in the Spatial Horizontal Edge Variation feature with a minimum value of 0 and maximum value of 255 is 0.88764 and precision rate of 0.24098. When the threshold is tuned to 128 from the 0, the results shows some slight changes, where the recall rate increase to 0.89499 and precision rate increase to 0.35283, refer to table 4.29 for more details on the accuracy.

Table 4.27 Results from the experiments of testing how edge image affecting the performance of the vehicle license plate detector trained with Spatial Horizontal Edge Variation feature. The threshold set for the canny edge function in OpenCV (CvCanny – threshold(minimum) = 128, threshold (maximum) = 255).

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
3 bins	45	45	0.36343	0.19114
6 bins	45	45	0.86999	0.18348
9 bins	45	45	0.88764	0.24098

Table 4.28 Results from the experiments of testing how edge image affecting the performance of the vehicle license plate detector trained with Spatial Horizontal Edge Variation feature. The threshold set for the canny edge function in OpenCV (CvCanny – threshold(minimum) = 0, threshold (maximum) = 255).

Number of Regions in Spatial Horizontal Edge Variation Feature	Horizontal Edge Variation Threshold - Training	Horizontal Edge Variation Threshold - Testing	Recall Rate	Precision Rate
3 bins	45	45	0.46561	0.30491
6 bins	45	45	0.88909	0.29365
9 bins	45	45	0.89499	0.35283




Original Image	Edge Image Minimum Threshold = 0 Maximum Threshold = 255	Edge Image Minimum Threshold = 128 Maximum Threshold = 255
		
1(a)	1(b)	1(c)

Figure 4.38 Images of original testing image 1(a), converted edge image with the minimum threshold of 0 and maximum threshold of 255 1(b), converted edge image with the minimum threshold of 128 and maximum threshold of 255

Next, we take one of the testing images as the example of how the experiment parameters were tuned. Figure 4.38 shows the diagram of images from the original testing dataset, the converted edge images with two different parameters. From there we can visually observe the differences between the two edge images. The image with less minimum threshold will have more edges detected; and the image with higher minimum threshold reduces the edges detected, which will reduce some of the noise from the image.


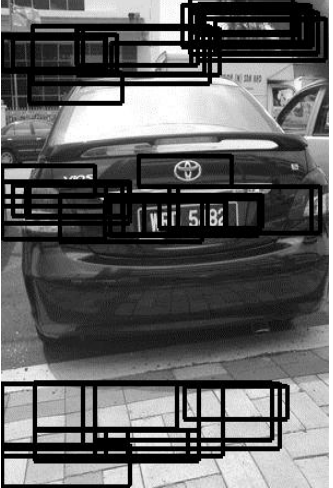

Original Image	Edge Image Minimum Threshold = 0 Maximum Threshold = 255	Edge Image Minimum Threshold = 128 Maximum Threshold = 255
		
1(a)	1(b)	1(c)

Figure 4.39 Diagram shows the results of the detection by the vehicle license plate detector trained with edge image. Images of original testing image 1(a), converted edge image with the minimum threshold of 0 and maximum threshold of 255 1(b), converted edge image with the minimum threshold of 128 and maximum threshold of 255

After that, we have a look at the results of the detection through Figure 4.39; where we found that the detection result of the edge image with smaller minimum value detected more false regions compare to the edge image with a higher minimum value. This scenario could be explained through the edges detected from the conversion with a smaller minimum is higher. This could lead to more noises in the image that might be wrongly classified as the license plate edges. However, by tuning these two parameters might not be sufficient to increase the accuracy of the detector, as these two parameters might be affected by others problems, such as brightness, contrast of the image. This is because

brightness could affect the clearness of the edges and reduce the edges detected as they might be removed when the threshold is set too high.

4.3.3 Cascaded Feature-thresholding Classification Vs Support Vector Machine Classification

In chapter 3, we define our system architecture into two stages, training and testing. During the training stage, we provide the positive and negative training samples for the system to execute an iterative of loops to search for the best features based on the samples, and construct a list of cascaded features from the selected best features. Each feature in the cascaded feature consists of a threshold which is used to classify whether the sub-window is a target object or non target object. By using this simple cascaded threshold-based classification, the classification process accelerates where it reduces the processing time by filtering most of the non-target object. However, the decision boundary made by the cascaded feature might not be good enough to separate the target object and non-target object. Therefore, in this section we conduct another simple experiment on examine the difference between cascaded feature thresholding classification and Support Vector Machine classification. Basically, our purpose is to see if there is any improvement in the accuracy when we implement Support Vector Machine as the classification method, where the same feature extraction is applied.

Table 4.29 Results from the experiments of comparing both different classification method, Cascaded Feature Classification Vs Support Vector Machine Classification.

	6 bins – Cascaded Feature Classification	6 bins – SVM Classification
Recall Rate	0.93796	0.85206
Precision Rate	0.55366	0.31581
Processing Speed (average)	~7 sec	~38 sec

Here, we used the vehicle license plate that we previously trained with Spatial Horizontal Edge Variation feature and combine all the selected feature in a cascaded form, where uses simple thresholding as the classification methods. Therefore, we are interested to see what the changes are when we modify the classification method from cascaded feature thresholding to Support Vector Machine Classification. Table 4.29 shows the experiments results that compare both classification methods. For the Support Vector Machine Classification, we apply the same features selected from cascaded feature, as the feature extraction in the SVM model. From the experiment results, we found that the recall rate for cascaded feature classification is still achieving high recall rate of 0.91988 and precision rate of 0.53373. On the contrary, when we evaluate on the SVM classification, the recall rate dropped to 0.85206, and precision rate too dropped to 0.31581.

4.3.4 How to Reduce False Detection?

From the results of the detector we trained for vehicle license plate, text from natural scene, side-view car. The precision rate of each of the detector was not high where a lot of false detections are detected during the experiment on the

test dataset. In this discussion, we will conduct a simple experiment to compare whether by including Support Vector Machine as the second stage in the system architecture will improve the performance of the detector or not. The SVM classifier is trained based on the same feature from the cascaded feature. We used the vehicle license plate detector (6 regions in Spatial Horizontal Edge Variation feature) as the comparison. The initial recall rate achieved by this detector is the highest among the rest of the detector trained with different bins and thresholds. Therefore, we used all the features from this detector and become the feature extraction of the SVM training. From table 4.4 we can observe there are some differences between the two classifications. The recall rate dropped from 0.93796 to 0.86698 for the 2 stages (cascaded feature classification & SVM for false removal), and the precision rate increased from 0.53373 to 0.55718. This however cannot conclude that by adding SVM classification as the false removal fails to improve the precision rate. We can assume that the features from the cascaded feature already remove all the false image as much as possible. Maybe we need to increase the negative image sample (non license plate) for the training, or modifying the feature type for further improvement of the accuracy.

Table 4.30 Results from the experiments of comparing Cascaded Feature Classification and the Cascaded Feature with Support Vector Machine as false removal.

	6 bins – Cascaded Feature Classification	6 bins – Cascaded Feature Classification + SVM for false removal
Recall Rate	0.91988	0.86698
Precision Rate	0.53373	0.55718
Processing Speed (average)	~7 sec	~6 sec

4.3.5 Images that failed to be detected

In this section, we will discuss on what are the images that we failed to detect for our test dataset. First and foremost, the experiments we conducted in this project is to analyze the algorithm performance on





- Spatial Horizontal Edge Variation Feature
- Cascading feature thresholding classification
- Edge image and grayscale image
- Support Vector Machine Classification

Spatial Horizontal Edge Variation feature in this project is introduced specifically in solving vehicle license plate detection problem by providing artificial license plate images as the training samples. However, there are some limitations with this feature type, as the contrast and brightness could cause the feature unable to classify the image correctly. Referring to Table 4.31 and Table 4.32, these two tables are showing the horizontal edge variation image generated by different threshold values, 30, 45, and 60 pixels. From the images, we can actually see that the image has more edges when the threshold value is set to a

lower value like 30 pixels. This is because when the threshold value is set to 30 pixels, based on our formula, for all the pixels comparing the pixel on their right, if the difference between both pixels is more than 30 pixels then the horizontal edge variation image will update based on the difference.

With this parameter, the benefit is we able to lower down the noise from the original image and focus on the target object that is having clearer edge view. Yet, this could become the drawback of the algorithm, where the system needs to determine the *right* threshold to set. Different threshold could generate different classification results, as the threshold could enlighten the edges while it could remove the edges too.



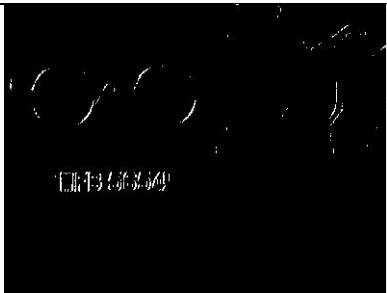
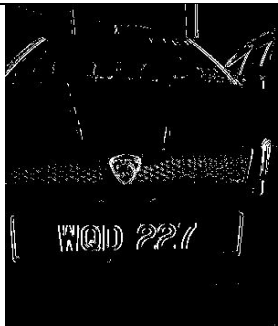

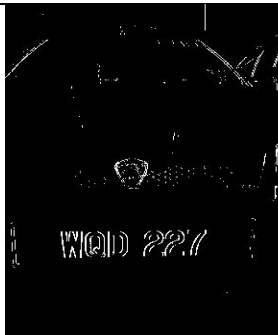
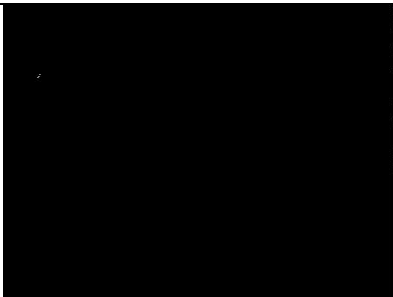
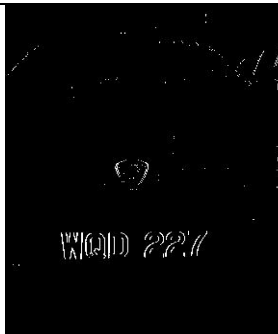
Table 4.31 Comparison between the original image (large image) and Horizontal Edge Variation Image (with threshold of 30, 45, and 60 pixels) (Side View Car Image).

Original Image	Spatial Horizontal Edge Variation Image – Threshold : 30	Spatial Horizontal Edge Variation Image – Threshold : 45	Spatial Horizontal Edge Variation Image – Threshold : 60
			

Setting a high value for the threshold during generating the horizontal edge variation image, could cause the target object (license plate) disappear as the threshold already remove the edges. We can observe this through the images from the Table 4.32, where one of the image with a higher brightness but low contrast







(left), and the other image with a normal color tone and brightness. The image on the left in Table 4.32, after setting the threshold value to 45 pixel during the horizontal edge variation image generation, the license plate edge disappear, as the color tone for the background and the license plate is similar.

Table 4.32 Comparison between the original image (large image) and Horizontal Edge Variation Image (with threshold of 30, 45, and 60 pixels) (Vehicle license plate Image).

Image Type	Image	Image
Original Image		
Horizontal Edge Variation Image (threshold – 30 pixels)		
Horizontal Edge Variation Image (threshold – 45 pixels)		
Horizontal Edge Variation Image (threshold – 60 pixels)		

Then, with the same method, we observe how the horizontal edge variation image affects the performance of the detector during the text detection. Table 3.4 shows two original images and its horizontal edge variation image. The text image on the left of the table shows that the image after the conversion contains some noise around the text; this could affect the detection, where all these noise could lead the feature to classify those regions as non license plate. Then, we have a look at another text image from the right, an image with some text and brightness effects. After the conversion, the horizontal edge variation image failed to retain any text edges as the brightness caused the image required lower threshold in order to keep the text edges in the image.

Table 4.33 Comparison between the original image (large image) and Horizontal Edge Variation Image (with threshold of 30, and 45 pixels) (Image with text).

Image Type	Image	
Original Image		
Horizontal Edge Variation Image (threshold – 30 pixels)		
Horizontal Edge Variation Image (threshold – 45 pixels)		

From our experiments on the ICDAR 03 text dataset, we found that we failed to achieve high accuracy on the testing samples. Some of the failure cases are due to the brightness and contrast of the image that cause the edges to be

unclear. Our algorithm failed to perform best with the existing techniques, as the algorithm could not train the classifier based on text with a single character mixing with the text with few characters. The dissimilarities of the training samples could cause the training process failed. Referring to Figure 4.40, some of the images selected from the ICDAR 03 text dataset, some of the images consist of just a single character or the text are similar with the background. Furthermore, some of the text could not be detected due to other factors, including the shape, size, so on and so forth.



Figure 4.40 Images failed to be detected by the Text Detector trained by the System with (Spatial Horizontal Edge Variation Feature).

Finally, we shows some of the images failed to be detected by the license plate detector trained with Spatial Horizontal Edge Variation feature at Figure 4.41. Some of these images failed to be detected when the threshold for the horizontal edge variation image generation is set too high, so when the threshold is set to a lower values, can help to improve on the detection rate.



Figure 4.41 Images failed to be detected by the license plate Detector trained by the System with (Spatial Horizontal Edge Variation Feature).

However, there are some images with a low brightness, failed even we tried to set the threshold to a very low value. The background of the license plate and the text of the license plate are having the similar brightness and contrast that is hard to be differentiated by the feature. In addition, rotation not more than 45 degree still acceptable for the detector to classify whether the candidate is the target object or non target objects.

CHAPTER FIVE

CONCLUSION

In conclusion, we presented a learning framework that consists of a genetic algorithm for feature searching, and a simple cascading algorithm for combining all the selected features into a cascaded feature for license plate detection. We introduced a feature type called Spatial Horizontal Edge Variation feature to solve license plate detection problem. This feature type is used to find the object that is having strong edges difference from the horizontal direction, where it calculates the difference of the pixel and its neighbour pixel on its right. In conjunction with this feature, we introduced an image type called horizontal edge variation image, to increase the processing speed, we further improve on the image by converting it into an integral horizontal edge variation image. With this integral image technique, the sub-window scanning could be execute very fast as the image already pre-calculate the image pixel. After that, we apply the sub-window scanning in this project to exhaustively search for the target object from the image. The sub-window scanning method start to find for the target object from the beginning (x, y) coordinate of (0, 0), and the move the sub-window with a step factor, so on and so forth until the whole image is scanned through. Each Spatial Horizontal Edge Variation feature selected has different shape, size, and coordinates inside the sub-window. Then the feature compute the feature value based on all the pixels changes from the left to right within the sub-window.

Therefore, whenever the sub-window moves to any part of the image, it will extract the feature values and classify whether that particular region is a target object or non target object.

From the experiment results, we found that Spatial Horizontal Edge Variation feature is able to generate high recall rate for the license plate detection problem, where the best recall rate on the license plate detection is 0.91988 and precision rate of 0.53373. However, applying the same algorithm to search for the Spatial Horizontal Edge Variation from the large feature set does not provide a high accuracy for the text detection and side view car detection. From our experiments, the text detector on spatial horizontal edge variation feature with the best accuracy among the others is only generating a recall rate of 0.56256 and low precision rate of 0.32939. Then, we observe the feasibility of spatial horizontal edge variation feature on the side view car detection problem and the best accuracy is the detector generating recall rate of 0.92986, but yet this detector failed to achieve high precision rate. Through all the experiments, we realize that the modification of the number of bin (defined as number of region in this project) in Spatial Horizontal Edge Variation feature is affecting the accuracy of the detector. For this moment we manually adjust the number of regions (sections) in the feature. Referring back to chapter 3, Spatial Horizontal Edge Variation features values is calculated through comparing the average from each region in the feature with the template feature. Therefore, we could improve this in the future by adding function to automatically search for the suitable number of regions for different object detection problem.

The introduction of horizontal edge variation image is to enable the feature extraction for spatial horizontal edge variation feature. This image type is able to remove noises from the original image and generate a whole new image representation for the classification. This image type is able to make the edges (horizontal) to be seen clearer. Although it can remove noises from the image, but this become its weakness too, as it is very dependent on the threshold tuning. Different threshold can generate different horizontal edge variation image. Images with low contrast and low brightness, or the target object is having pixel values similar to its surrounding background could fail the detection. A lot future works can be done to solve this problem, where we can design the system in a way that will dynamically set the threshold by analyzing the brightness and contrast level of the image.

Besides, the text detector (spatial horizontal edge variation feature) that we trained in this project failed to achieve high accuracy. This is because, we found that the detector could not detect text with different orientation of the shape patterns, different size of each characters in a string, text with a single characters, text that are having similar pixel intensities with its background, text from logo, and etc. The research we had done for detecting text from natural scene using our spatial horizontal edge variation feature is still new. Preliminary results show that there are still a lot improvements can be done in enhancing the existing algorithm to locate the text from natural scene.

Further, we compared between the detector trained with normal grayscale images and detector trained with edged images. The results show that the detector trained with edged image can be improved if we can determine a best way to compute the suitable thresholds used for generating the edged image. This is because, if we are able to get the appropriate thresholds for different image, we can remove noises and make the edges of the image looked even much clearer. This is similar with our horizontal edge variation image as both also depend on setting the right threshold.

In addition, we examine the difference between the simple cascaded feature threshold classification and the support vector machine classification. We apply the same feature extraction for both classification techniques, and compare their performance. We found that the result from the cascaded feature threshold is better than the support vector machine classification. This however does not conclude that support vector machine classification cannot perform well in this problem, as we can further research on the samples type, adding more training samples, and modifying the feature type with other feature extraction such as Scale Invariant Feature Transformation (SIFT) or Speeded Up Robust Feature (SURF) or Histogram of Oriented Gradient feature, and etc. Subsequently, cascaded feature classification is much faster than support vector machine classification as the cascaded feature is designed in a way where once any features in the cascaded feature rejected the image, the rest of the feature will no longer required to further processed the image.

Last but not least, from all the results we observed, there are still a lot of research can be done to improve current results. There are still a lot of false detections from the results; we could improve further on the samples image, the feature search algorithms. Besides that, further modification on the feature could be done by adding new feature type that could find the pixel changes from the top to the bottom and from the left to the right of the image. This type of feature will extract the information about the object from vertical and horizontal. with the introduction of the feature extractions and feature searching techniques, perhaps more research and analysis could be done through this direction, so that more improved techniques can be introduced in future to solve more challenging object detection problems.

REFERENCES

- A. L., C. M., Wong, F., & K. T. K., T. (2011). Tracking and localisation of moving vehicle license plate via signature analysis. *Mechatronics (ICOM), 2011 4th International Conference On*, (pp. 1 - 7). Kuala Lumpur, Malaysia.
- A. Ravi, T., Shashank, J., Abhishek, A., & Kandavli, V. K. (2010). License Plate Extraction Using Adaptive Threshold and Line Grouping. *Signal Processing Systems (ICSPS), 2010 2nd International Conference*, (pp. V1-211 - V1-214).
- Abolghasemi, V., & Ahmadyfard, A. (2009). An edge-based color-aided method for license plate detection. *Image and Vision Computing, Volume 27, Issue 8* , 1134-1142.
- Abramson, Y., & Steux, B. (2005). YEF Real-Time object Detection. *Proc. of Intl. Workshop on Automatic Learning and Real-Time (ALART'05)*.
- Abramson, Y., Moutarde, F., Steux, B., & Stanculescu, B. (2006). Combining adaboost with a hill-climbing evolutionary feature-search for efficient training of performant. in *Proceedings of the 7th International FLINS Conference on Applied Artificial Intelligence (FLINS '06)*. Genova, Italy.
- Agarwal, S., Awan, A., & Roth, D. (2004). Learning to Detect Objects in Images via a Sparse, Part-Based Representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (pp. 1475-1490).
- Bai, H., & Liu, C. (2004). A hybrid license plate extraction method based on edge statistics and morphology. *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference*, (pp. 831 - 834).
- Balasuriya, L., & Kodikara, N. (2001). Frontal view human face detection and recognition. *Proceedings of the 20th National Information Technology Conference*. Colombo, Sri Lanka Colombo.
- Bay, H., Ess, A., Tuytelaars, T., & Gool, L. V. (2008). SURF: Speeded Up Robust Features. *Computer Vision and Image Understanding (CVIU)*, (pp. 346-359).
- Bdiri, T., Moutarde, F., Bourdis, N., & Steux, B. (2009). AdaBoost with “Keypoint Presence Features” for Real-Time Vehicle Visual Detection. *16th ITS World Congress and Exhibition on Intelligent Transport Systems and Services*. Stockholm, Sweden.
- Boughorbely, S., Tarel, J.-P., & Boujemaay, N. (2005). GENERALIZED HISTOGRAM INTERSECTION. *Proceedings of IEEE International Conference on Image Processing (ICIP'05)*, (pp. 161-164).
- Burges, C. J. (2004). A Tutorial on Support Vector Machines for Pattern Recognition . *Data Mining and Knowledge Discovery* , 121-167.
- Chang, C.-C., & Lin, C.-J. (2001). *LIBSVM: a library for support vector machines*.

- Chen, C.-C., Hsieh, J.-W., Wu, J.-C., & Chen, Y.-S. (2006). Detecting license plates from videos using morphological operations and adaboosting algorithm. *International Computer Symposium*, (pp. 1201–1204).
- Chen, D., Bourlard, H., & Thiran, J.-P. (2001). Text Identification in Complex Background Using SVM. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'01)*, (p. 621).
- Chen, X., & Yuille, A. L. (2004). Detecting and Reading Text in Natural Scenes. *Computer Vision and Pattern Recognition, 2004*, (pp. II-336 - II-373).
- Cristianini, N., & Shawe-Taylor, J. (2000). *Support Vector Machines and other kernel-based learning methods*. Cambridge: The Press Syndicate of the University of Cambridge.
- Deb, K., & Jo, K.-H. (2008). HSI Color based Vehicle License Plate Detection. *International Conference on Control, Automation and Systems 2008* , (pp. 687-691). Korea.
- Ezaki, N., Bulacu, M., & Schomaker, L. (2004). Text Detection from Natural Scene Images: Towards a System for Visually Impaired Persons. *International Conference on Pattern Recognition (ICPR 04)*, (pp. 683-686).
- Freund, Y., & Schapire, R. E. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. *Proceedings of the Second European Conference on Computational Learning Theory*, (pp. 23 - 37).
- Gao, J., & Yang, J. (2001). An Adaptive Algorithm for Text Detection from Natural Scenes. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'01)*, (p. 84).
- Gao, J., Yang, J., Zhang, Y., & Waibel, A. (2001). *Text Detection and Translation From Natural Scenes*. Pittsburgh: School of Computer Science, Carnegie Mellon University.
- Guo, L., Hu, Y., Hu, Z., & Tang, X. (2010). The Edge Detection Operators and Their Application in License Plate Recognition. *Computational Intelligence and Software Engineering (CiSE), 2010 International Conference*, (pp. 1 - 4).
- Hsu, R.-L., Abdel-Mottaleb, M., & Jain, A. K. (2002). Face Detection in Color Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (pp. 696-706).
- (1999-2001). *Intel Open Source Computer Vision Library*. USA: Intel Corporation.
- Khammari, A., Lacroix, E., Nashashibi, F., & Laugeau, C. (2005). Vehicle detection combining gradient analysis and AdaBoost classification. *Proceedings. 2005 IEEE Intelligent Transportation Systems*, (pp. 66-71).
- Kim, J., Han, Y., & Hahn, H. (2008). License Plate Detection Using Topology of Characters and Outer Contour. *Future Generation Communication and Networking Symposia, 2008. FGCNS '08. Second International Conference*, (pp. 171 - 174).

- Kim, K. I., Jung, K., & Kim, J. H. (2003). Texture-Based Approach for Text Detection in Images Using Support Vector Machines and Continuously Adaptive Mean Shift Algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (pp. 1631 - 1639).
- Levi, K., & Weiss, Y. (2004). Learning Object Detection from a Small Number of Examples: The Importance of Good Features. *Computer Vision and Pattern Recognition*, (pp. 53-60).
- Liu, X., & Samarabandu, J. (2006). Multiscale Edge-Based Text Extraction from Complex Images. *IEEE International Conference on In Multimedia and Expo 2006*, (pp. 1721-1724).
- Lucas, S. M., Panaretos, A., Sosa, L., Tang, A., Wong, S., & Young, R. (2003). ICDAR 2003 Robust Reading Competitions. *Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR 2003)*.
- Miao, L., Wang, F., & Wang, H. (2009). Automatic License Plate Detectio based on Edge Density and Color Model. *Control and Decision Conference, 2009. CCDC '09. Chinese* , (pp. 3718 - 3721). Guilin.
- Mitchell, T. M. (1997). Chapter 9: Genetic Algorithm. In T. M. Mitchell, *Machine Learning* (pp. 249-273). MIT Press and The McGraw-Hill.
- Mohan, A., Papageorgiou, C., & Poggio, T. (2001). Example-Based Object Detection in Images by Components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (pp. 349 - 361).
- Nguyen, C.-D., Ardabilian, M., & Chen, L. (2008). Real-time license plate localization based on a new scale and rotation invariant texture descriptor. *11th International IEEE Conference on Intelligent Transportation Systems (ITSC)*.
- Nikzad, M., Dehkordi, M. Y., Ekhlasi, V. R., & Azimifar, Z. (2010). A Novel Approach for Fast and Robust Multiple on license plate detection. *Machine Vision and Image Processing (MVIP), 2010 6th Iranian*, (pp. 1-6).
- Nobuo, E. (2004). Text Detection from Natural Scene Images: Towards a System for Visually Impaired Persons. *In Int. Conf. on Pattern Recognition*, (pp. 683--686).
- Oh, I.-S., Lee, J.-S., & Moon, B.-R. (2002). Local Search-Embedded Genetic Algorithms for Feature Selection. *16th International Conference on Pattern Recognition (ICPR'02)*, (p. 20148).
- Papageorgiou, C., & Poggio, T. (1999). *A Trainable Object Detection System: Car Detection in Static Images*. A.I. Memo No. 1673, C.B.C.L Paper No. 180, Massachusetts Institute of Technology.
- Papageorgiou, C., & Poggio, T. (2000). A Trainable System for Object Detection. *International Journal of Computer Vision* , 38, 15-33.

- Porikli, F. (2005). Integral Histogram: A Fast Way to Extract Histograms in Cartesian Spaces. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 829-836).
- Porikli, F., & Kocak, T. (2006). Robust License Plate Detection Using Covariance Descriptor in a Neural Network Framework. *Proceedings of the IEEE International Conference on Video and Signal Based Surveillance*, (p. 107).
- Rubner, Y., Tomasi, C., & Guibas, L. J. (2000). The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision* , 2000.
- Saha, S., Basu, S., Nasipuri, M., & Basu, D. K. (2009). License Plate Localization from Vehicle Images: An Edge Based Multi-stage Approach. *International Journal of Recent Trends in Engineering (IJRTE)*, (pp. 284-288).
- Satish, M., L. V., & K. S. (2011). Edge assisted fast binarization scheme for improved vehicle license plate recognition. *Communications (NCC), 2011 National Conference on* , (pp. 1 - 5).
- Scott, D. W. (1979). On optimal and data-based histograms. *Biometrika* , 605-610.
- Shivakumara, P., Huang, W., & Tan, C. L. (2008). Efficient Video Text Detection using Edge Features. *International Conference on Pattern Recognition ICPR 2008*.
- Sun, Z., Bebis, G., & Miller, R. (2004). Object Detection Using Feature Subset Selection. *Pattern Recognition* , 2165-2176.
- Treptow, A., & Zell, A. (2004). Combining Adaboost Learning and Evolutionary Search to select Features for Real-Time Object Detection. *Evolutionary Computation CEC*, (pp. 2107 - 2113).
- Tuzel, O., Porikli, F., & Meer, P. (2006). Region Covariance : A Fast Descriptor for Detection and Classification. *9th European Conference on Computer Vision*, (pp. 589-600). Graz, Austria.
- Vapnik, V. N. (1995). *The nature of statistical learning theory*. New York, United States of America: Springer-Verlag.
- Venables, W. N., & Ripley, B. D. (2002). section 5.6: Density Estimation. In W. N. Venables, & B. D. Ripley, *Modern Applied Statistics with S* (p. 126). Springer.
- Viola, P., & Jones, M. J. (2004). Robust Real-Time Face Detection. *International Journal of Computer Vision* , 57 (2), 137-154.
- Wang, H., Li, P., & Zhang, T. (2006). Histogram Feature-Based Fisher Linear Discriminant for Face Detection. *Asian Conference on Computer Vision*, (pp. 521-530).
- Wang, W. (2009). Reach on Sobel Operator for Vehicle Recognition. *Artificial Intelligence, 2009. JCAI '09. International Joint Conference*, (pp. 448 - 451).

- Wang, Y.-R., Lin, W.-H., & Horng, S.-J. (2011). A sliding window technique for efficient license plate localization based on discrete wavelet transform. *Expert Systems with Applications: An International Journal* , 3142-3146.
- Zahedia, M., & Salehia, S. M. (2011). License Plate Recognition System Based on SIFT Features. *Procedia Computer Science, Vol. 3 (2011), World Conference on Information Technology* , 998-1002.
- Zhang, H., Gao, W., Chen, X., & Zhao, D. (2005). Learning Informative Feature for Spatial Histogram-Based Object Detection. *Proc. of International Joint Conference on Neural Networks*, (pp. 1806-1811). Montreal, Canada.
- Zhang, H., Jia, W., He, X., & Wu, Q. (2006). Learning-Based License Plate Detection Using Global and Local Features. *18th International Conference on Pattern Recognition (ICPR'06)*, (pp. 1102-1105).
- Zhang, R., & Zhang, Y. (2009). Car Number Plate Detection Using Multi-layer Weak Filter. *Business Intelligence and Financial Engineering, 2009. BIFE '09. International Conference on* , (pp. 228 - 232).
- Zhang, X., & Zhang, S. (2010). A Robust License Plate Detection Algorithm Based on Multi-features. *Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference*, (pp. 598 - 602).
- Zhao, Y., Gu, J., Liu, C., Han, S., Gao, Y., & Hu, Q. (2010). License Plate Location Based on Haar-Like Cascade Classifiers and Edges. *2010 Second WRI Global Congress on Intelligent Systems*, (pp. 102-105).
- Zhu, C., Ouyang, Y., Gao, L., Chen, Z., & Xiong, Z. (2009). An Automatic Video Text Detection, Localization and Extraction Approach. In L. N. Science, *Advanced Internet Based Systems and Applications* (pp. 1-9). Springer Berlin / Heidelberg.
- Zhu, K., Qi, F., Jiang, R., Xu, L., Kimachi, M., Wu, Y., et al. (2005). Camera-Based Document Analysis and Recognition. *First International Workshop on Camera-Based Document Analysis and Recognition (CBDAR2005)*.