License Plate Detection and Segmentation Using Cluster Run Length Smoothing Algorithm

Siti Norul Huda Sheikh Abdullah, Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, Malaysia

Muhammad Nuruddin Sudin, Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, Malaysia

Anton Satria Prabuwono, Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, Malaysia

Teddy Mantoro, Advanced Informatics School, Universiti Teknologi Malaysia, Malaysia

ABSTRACT

For the different types of license plates being used, the requirement of an automatic license plate recognition system is different for each country. In this paper, an automatic license plate detection system is proposed for Malaysian vehicles with standard license plates based on image processing and clustering. Detecting the location of license plate is a vital issue when dealing with uncontrolled environments and illumination difficulty. Therefore, a proposed algorithm called Cluster Run Length Smoothing Algorithm (CRLSA) was applied to locate the license plates at the right position. CRLSA consisted of two separate proposed algorithms which applied run length edge detector algorithm using 3×3 kernel masks and 128 grayscale offset plus a three-dimensional way to calculate run length smoothing algorithm, which can improve clustering techniques in segmentation phase. Six separate experiments were performed; Morphology, CRLSA, Clustering, Square/Contour Detection, Hough, and Radon Transform. From those experiments, analysis based on segmentation errors was constructed. The prototyped system has accuracy more than 96%.

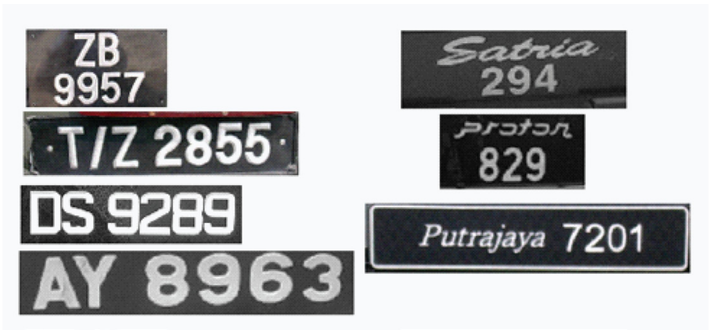
Keywords: *Clustering, Image Detection, License Plate Recognition, Run Length Smoothing Algorithm*

INTRODUCTION

Image detection is one of the crucial issues in image processing besides feature extraction and recognition (Bataineh, Abdullah, & Omar, 2011, 2012). Computer surveillance such as license plate recognition (Romero, Prabuwono, & Taufik, 2011) is an example of object detection application. To ensure that the selected and determined objects were correctly obtained, a very strategic way for object detection is highly required (Abdullah, PirahanSiah, Abidin, & Sahran, 2010; Prabuwono & Idris, 2008). Several prominent ways License Plate Detec-

DOI: 10.4018/jitr.2012070103

Figure 1. (a) Samples of common and (b) Samples of special Malaysia license plates



tion (LPD) were for contour detection (Han, Han, Wang, & Zhai, 2003), Hough transform (Soh, Chun, & Yoon, 1994), Radon transform (Shapiro, Gluhchev, & Dimov, 2006), Morphology (Xu & Zhu, 2007), clustering (Siah, 2000; Abdullah, Khalid, Yusof, & Omar, 2007; Abdullah et al., 2010; PirahanSiah, Abdullah, & Sahran, 2011; Abidin, Abdullah, Sahran, & PirahanSiah, 2011) and Speed Up Robust Features (SURF) (Bay, Tuytelaars, & Van Gool, 2006). However, this paper will elaborate on the theories and applications of only some of the mentioned techniques for object detection. Later, these techniques are experimented and compared.

The rest of the paper is organized as follows. Related works and proposed method are discussed in subsequent sections. Then, we justify our proposed work in the Results and Analysis section. This paper concludes our findings in the Conclusion section.

RELATED WORKS

Object Detections

SURF is a technique proposed for object detection (Bay et al., 2006). SURF applies Fast Hessian, Harris-Laplace and DoG Detectors to calculate its descriptors that identify the interest point and area based on image intensity pattern. The experiments achieved up to 85.7% recognition rate. However, this technique seems to be dangerous to apply in LPD because illumination can cause an object of interest's area broaden or lengthen due to lengthening or inconsistency of intensity pattern. Combining with thresholding process can reduce unwanted intensity but, unfortunately, it may destroy important features. Examples of image result using LPD is shown in Figure 2 and Figure 3.

Square detection, line detection (Han et al., 2003) and object detection have been widely applied to detect the location of license plates. Square detection is highly dependent on threshold value. Normally, Canny edge detector with a fixed upper and lower threshold is a powerful technique to produce gradient shading out of an image. Meanwhile, dilation operation is used to remove potential holes between edge segments. After applying these two techniques, contour search is executed. Typically, the searching is also highly threshold dependent which may deviate the actual square object.

Morphology technique is a common way to extract a region or object of interest. In practice, morphology techniques consists of several important classes that applies different operators which sometimes the region or object of interest may be lost or missing due to standardized of filtering operators for all image cases.

Hough transform is a second order operator which also depends on edge detector performance. If the resultant image of the edge detector shows clear lines of the license plate's border, therefore the line detection may be useful else otherwise. Applying Canny edge detector, the resultant lines in the image proved

Figure 2. Samples image result using SURF technique for Malaysia license plates: Example 1

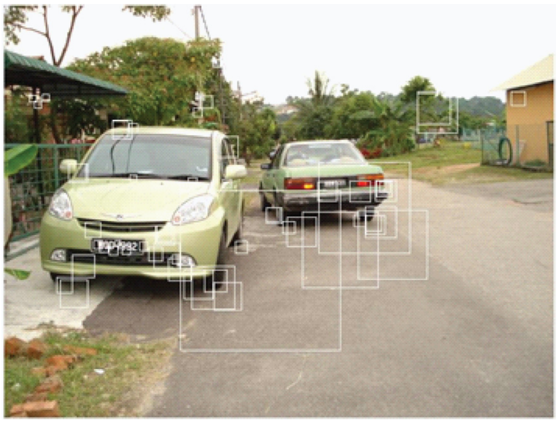
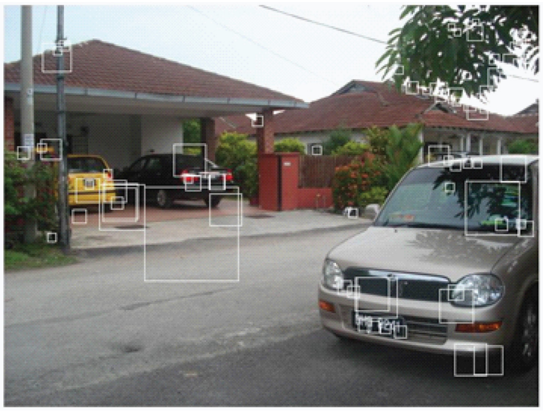


Figure 3. Samples image result using SURF technique for Malaysia license plates: Example 2



that Hough transform can detect the location of the license plate fairly well. Hough transform is powerful enough to identify the gaps or disjoints of straight lines. On the other hand, one distinct limitation of Hough transform is that it can hardly reconstruct the line in case of salt-pepper noise.

Additionally, Radon transform is a second derivative image which is similar to Hough transform, it requires the use of any edge detector operator to compute the skeleton of the image. Canny edge detector is normally applied because it considers inner and outer lines of an image. The disadvantage of Canny edge detector is that it is also a second derivative operator which correlates to threshold selective value and this may cause problems to the overall detection process.

In conclusion, Square detection and Hough transform is perhaps the better way to detect Malaysian special license plates because it will automatically capture all important information on the plate and is less prone to missing segmentations. Besides that, square detection prunes to detect license plates with a lighter background and the car body with a darker background because the low intensity color creates a better or clearer square object of the license plate. Meanwhile, Hough transform has limitations to noisy images like disjointed lines and requires enhancements first before executing. Despite that, illumination is the worst threat of this approach since it can distract the square or line shape detection. Radon transform can perform better results for license plate detection because it is powerful enough to detect salt and pepper problems.

Run Length Smoothing Algorithm

Run Length Smoothing Algorithm (RLSA) has been used widely in Optical Character Recognition (OCR) process especially in document analysis (Papamarkos, Tzortzakis, & Gatos, 1996). The previous method developed for the Document Analysis (Nagy, 1968) consists of two steps. Firstly, a segmentation procedure subdivides the area of a document into regions (blocks), each of which should contain only one type of data (text, graphic, halftone or image) and later some basic features of these blocks are calculated. Then, a linear classifier is adapted in verifying character heights and discriminating between text and images. Normally, RLSA technique is only applied after horizontal or vertical projection so as to recognize the block segmentation, and the transformations are only concerned with two dimensional images $(black - white)$ (Papamarkos et al., 1996). Technically, RLSA used in document analysis only involves binary image or black and white image after applying a threshold process (Papamarkos et al., 1996). The run length is calculated based on the distance between two bounding boxes after executing Connected Component Labeling (CCL) method. It comprises two components: horizontal distance and vertical distance. Apart from RLSA, it is well known for top-down segmentation approach, it is insensitive to document skew (Papamarkos et al., 1996). It also requires high memory storage to execute the algorithm. Sometimes RLSA fails to separate non-orthogonal blocks that are intermixed with graphics or halftones. This drawback is due to the use of global smoothing values that are less suitable for the document's local characteristics (Nagy, 1968).

Clustering Run Length Smoothing Algorithm (CRLSA)

The CRLSA is introduced as a segmentation technique in LPD. The LPD using CRLSA is continued after the image correction or image enhancement module. The CRLSA comprises two submodules: (1) The Modified Clustering Technique (MCT) submodule and (2) The Checking Technique (CT) submodule. The MCT submodule consists of two main processes: (a) Cluster all the blobs, (b) Sort all the clusters and clusters' members. The CT submodule comprises three main processes: (c) Run RGB Convolution and Check the First Run Length Smoothing Algorithm (RLSA1), (d) Run Threshold Value One (TV1) and Check the Second RLSA (RLSA2), (e) Get winner cluster indexes. All the processes in the MCT and CT modules are executed consecutively. This involves two consecutive checking processes in the CRLSA module. First is to check the RLSA1 and second is to check the RLSA2. A brief explanation of each process is given in the following paragraphs.

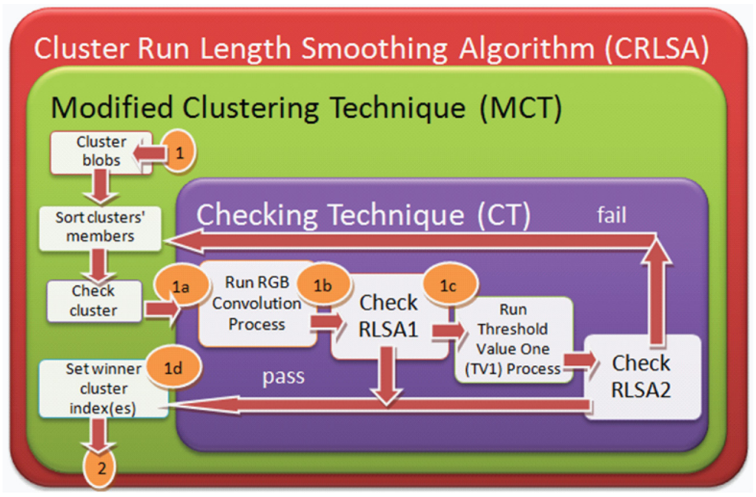
1. Cluster all the Blobs

State 1 is started. Normally, the size of the license plate characters are typically homogeneous. Thus, according to Figure 4, these blobs are retrieved from Blob Sequence Storage and are then clustered according to their sizes and distances from each other. The cluster conditions are also modified to accept any blob which have similar characteristics at any rotation. These modified clusters are kept in Cluster Sequence Storage.

2. Sort all the Clusters' Members

All blobs of each cluster are sorted twice: Firstly, the cluster is sorted according to the minimum-X, $minX_{B1_{(0,1,\ldots,Size-1)}}$, values where $Size$ is the total number of blobs or member size, and secondly, the cluster is sorted, $C1_{(0,1,\ldots,n-1)}$ based on $Size$, where n is the total number of clusters. This information is restored

Figure 4. The basic flow chart of CRLSA module in the LPD phase


Cluster Run Length Smoothing Algorithm (CRLSA)
Modified Clustering Technique (MCT)
Cluster blobs
1
Sort clusters' members
Checking Technique (CT)
fail
Check cluster
1a
Run RGB Convolution Process
1b
Check RLSA1
1c
Run Threshold Value One (TV1) Process
Check RLSA2
Set winner cluster index(es)
1d
pass
2

in Cluster Sequence Storage and Blob Sequence Storage.

3. Run RGB Convolution and Check RLSA1

State 1b is started. Firstly, the winner cluster, WC, is determined. The criteria of the first cluster to be checked, is the maximum number of member size. Secondly, the cluster image is transformed into a three dimensional image using RGB convolution process. Finally, the run length of the image result is checked using RLSA1 approach.

4. Run TV1 and Check RLSA2

Before performing RLSA2, the RGB Convolution image is transformed into a binary image using TV1 approach. RLSA2 is a second check process using a two dimensional image. If this check also fails, then the next maximum cluster is examined until the winner cluster is found.

5. Set Winner Cluster Indexes

If either the first or the second checking is successful then the winner cluster's reference is kept in an array called Winner Cluster Sequence. If neither the first nor the second checking is successful, therefore the segmentation has failed and the LPR system is terminated. Since the Malaysian license plate format can either be a single or double row, then the second "winner" cluster is looked for near the first winner cluster. Then, Steps (c) and (d) is repeated to check the second winner cluster's run length. If it is successful, then the reference of the second winner cluster is also kept in the same array. Lastly, State 2 is permitted to the next module which is referred to as the Character Segmentation.

The CRLSA approach is explained in detail later. Algorithm 1 is the algorithm of the CRLSA module based on the Figure 4.

The Modified Clustering Technique (MCT)

As mentioned earlier, the LPR system applies clustering technique to identify important blobs. After processing the image using a simple image enhancement technique, such as Homomorphic Filter for the Image Enhancement module and Fixed Thresholding, Blob Analysis and CCL in the LPD and Segmentation module (which are provided in the VSDP library and Figure 4), the image is segmented using horizontal scan line profiles and clustering technique. Thoroughly, each image is transformed into blob objects

Algorithm 1. The CRLSA algorithm module

```
Input: Retrieve the Blob Sequence Storage, $B1_{(0,1,\ldots,Size-1)}$. The CT class object, $S$
is either `successful' or `unsuccessful'. {State 1a}
Output: Develop a Cluster Sequence Storage, $C1_{(0,1,\ldots,n-1)}$. Obtain the total number
of winner cluster, $twc$, and winner cluster indexes, $WC_{index}$.
1: Execute the MCT submodule. {State 1}
2: $counter \Leftarrow n-1$.
3: $twc = 1$.
4: while $(counter >= 0) \cap (S(counter) \neq unsuccessful)$ is true do
5:    $counter \Leftarrow counter - 1$.
6:    $WC_{twc} \Leftarrow 0$.
7: end while {Search the first winner cluster in $C1_{(0,1,\ldots n-1)}$. }
8: $WC_{twc} \Leftarrow counter$.
9: if ($WC_{twc} = 0$) is true then
10:   Indicate the cluster as ``Segmentation Fail". {Finish}
11: else
12: for $((WC_{twc} - 1) \geq counter \geq 0)$ is true do
13:   if $H_{C1_{counter}} = H_{WC_{twc}}$ then
14:      if $S(counter) = successful$ is true then
15: $twc \Leftarrow 2$.
16:        Set the second winner cluster as $WC_{twc} \Leftarrow counter$.
17:      else
18:      $twc \Leftarrow 1$.
19:      endif
20:   endif
21: $counter \Leftarrow n-1$.
22: end for {Check the second winner cluster, $WC_{twc+1}$, based on two conditions: the
cluster's distance and the height is the same as the first winner cluster, $WC_{twc}$.}
23: return twc.
24: Execute the character segmentation module {State 2}
```

and important information such as location, height and width. Next, they are analyzed by the LPR system for the purpose of clustering and choosing the best winner cluster pertaining to the blobs' arrangements (Figure 5).

Suppose, $B_{counter} \in B_{(0,1,\ldots,m-1)}$ is the predetermined blob, $C_{counter1} \in C_{(0,1,\ldots,n-1)}$ is the predetermined cluster, $maxY$ is the maximum Y value, H is the height, and α value is 0.3, 0.5 or 0.7. These $alpha$ values are optimized through experiments and supported by Siah (2000). The blobs are clustered (Figure 6) as follows (Equation (1) and (2)):

$$maxY_{B_{counter}} - maxY_{C_{counter1}} < \alpha \times \mathrm{H}_{C_{counter1}}, \tag{1}$$

$$H_{C_{counter1}} - H_{B_{counter}} < \alpha \times \mathrm{H}_{C_{counter1}}. \tag{2}$$

This clustering algorithm is modified by changing the previous constants, α, and accepts all blobs with the same height and different width (Siah, 2000; Siah, Haur, Khalid, & Ahmad, 1999). The α value is increased from 0.3 to 0.7 in order to accept rotated or slanted license plates. Previously, Siah (2000) only consider blobs with similar width as members of the cluster. The MCT algorithm is described in Algorithm 2.

Examples of the original, threshold and labeled and clustered blobs image are depicted in Figure 7.

Figure 5. Image segmentation using the clustering approach

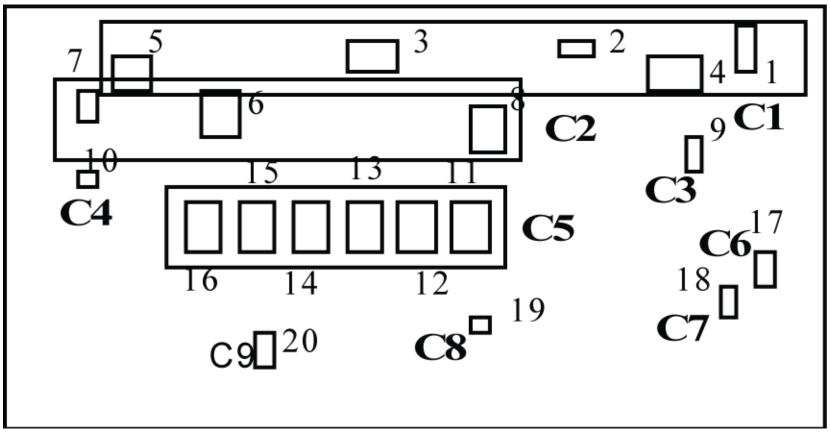
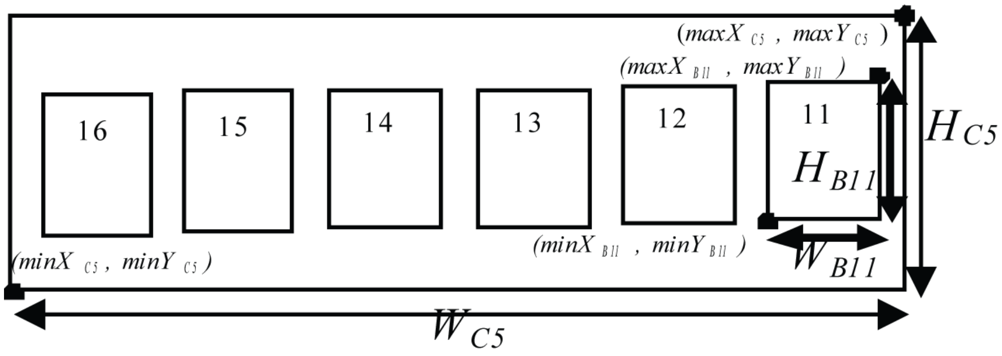


Figure 6. Important information of a cluster



The Checking Technique (CT)

The conventional RLSA approaches (Wong, Casey, & Wahl, 1982; Papamarkos et al., 1996) relied only on the two dimensional image (black-white). However, in the CT process, the image is first transformed into the three dimensional image (black-gray-white) using the RGB convolution sub-process and then the run length is calculated in the RLSA1 sub-process. If the three dimensional's run length calculation fails, only the second check is performed using the two dimensional (black-white) method. The two dimensional image is only generated by applying TV1 sub-process on the three dimensional image. Upon completion of the binary transformation of the TV1 image, the RLSA2 is successively performed.

As mentioned earlier, this sub-section consists of two main processes; (a) Run the RGB Convolution and Check the RLSA1, (b) Run TV1 and Check the RLSA2. The steps are elaborated below.

Run RGB Convolution and Check the First Run Length Smoothing Algorithm (RLSA1)

This process comprises of two sub-processes called RGB convolution and RLSA1. Firstly, the RGB convolution is executed. Next, the run length is checked in the RLSA sub-process. The

Algorithm 2. CRLSA: The MCT Algorithm

```
Input: Given, $C_{(0,1,\ldots,n-1)}$ are the clusters, where $n$ is the total number of cluster, $B_{(0,1,\ldots,m-1)}$ are the blobs, where $m$ is the total number of blobs, $B_{counter} \in B_{(0,1,\ldots,m-1)}$ is the predetermined blob, $C_{counter1} \in C_{(0,1,\ldots,n-1)}$ is the predetermined cluster, $maxX$ and $maxY$ are the maximum X and Y values, $minX$ and $minY$ are the minimum X and Y values, $H$ is the height, and $W$ is the width of each blob or cluster.
Output: Obtain $C1_{(0,1,\ldots,n-1)}$, a series of sorted member clusters, $C2_{(0,1,\ldots,n-1)}$, a series of $C1$ sorted according to member size of the cluster, and $B1_{(0,1,\ldots,Size-1)}$, a series of sorted blobs in the sorted clusters of $C1$.
1. Execute the MCT submodule. {State 1: Step 7-7}
2. for $counter \Leftarrow 0$. (0 $\leq counter \leq m-1$) is true do
3.     $counter1 \Leftarrow 0$.
4.    Cluster each blobs, $B_{counter}$, into groups, $C_{counter1}$, as in Equations (1) and (2).
5. $counter1 \Leftarrow counter1 + 1$.
6. end for {Obtain a series of clusters with specific member size, $C_{(0,1,\ldots,n-1)}$ and store them in Cluster Sequence Storage.}
7. for ($0 \leq counter \leq n-1]$) is true do
8.    The cluster, $C_{counter}$ is sorted descending or according to the total number of blobs or the cluster's member size, $Size_{counter}$.
9.     $counter \Leftarrow counter + 1$.
10. end for  {Obtain the sorted cluster, $C1_{(0,1,\ldots,n-1)}$.}
11. for ($0 \leq counter \leq n-1$) is true do
12.    for ($0 \leq counter1 \leq Size_{counter}$) is true do
13.       Sort all the members of the cluster, $C1$ according to minimum X value of the blobs, $minX_{B1_{counter1}}$.
14.       Update the cluster's height, $H_{B1_{counter1}}$ and width, $W_{B1_{counter1}}$.
15.  $counter1 \Leftarrow counter1 + 1$.
16. end for
17.  $counter1 \Leftarrow counter + 1$.
18. end for {Obtain a series of sorted blobs in the sorted clusters ($C1$), $B1_{(0,1,\ldots,Size-1)}$.}
19. Execute the RGB convolution subprocess.{State 1a}.
```

RGB convolution and RLSA1 algorithm are explained in the following paragraphs.

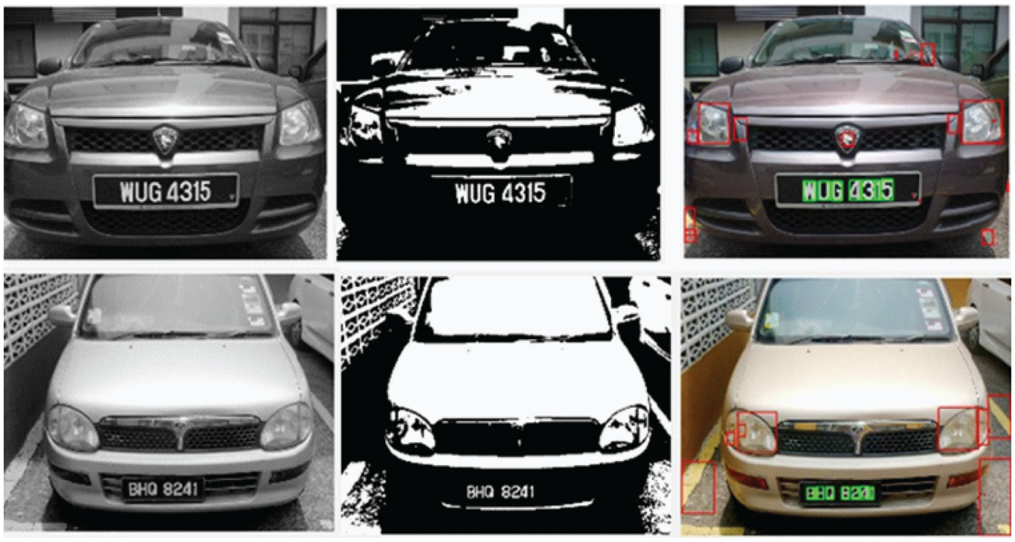
Run RGB Convolution

RGB Convolution is an approach to transform a RGB pixel value to determine the gray scale value. The gray scale value is a three dimensional image which appears as a *black-gray-white* image. Either Run Length Edge Detection (RED) or Kirsch Edge Detection (KED) with 128 gray scale offset technique is used in the approach to determine the gray scale value (Abdullah et al., 2007). The 3×3 kernel mask is applied as illustrated in Kernel matrix (3). Every pixel (in RGB format), RGB_{sum} in the image is summed and multiplied by any two of the kernel matrices given in Kernel matrix (4) and (5).

$$\begin{bmatrix} kernelV_{(0,0)} & kernelV_{(1,0)} & kernelV_{(2,0)} \\ kernelV_{(0,1)} & kernelV_{(1,1)} & kernelV_{(2,1)} \\ kernelV_{(0,2)} & kernelV_{(1,2)} & kernelV_{(2,2)} \end{bmatrix} \quad (3)$$

$$kernelV_{(x,y)_{RED_{vertical}}} = 128 + \begin{bmatrix} 3 & 0 & -3 \\ 5 & 0 & -5 \\ 3 & 0 & -3 \end{bmatrix} \quad (4)$$

Figure 7. Examples of (left) the original, (middle) and threshold, images. While (right) are the labeled and clustered blobs images using color blob labellings process.



$$kernelV_{(x,y)_{KED_{vertical}}} = 128 + \begin{bmatrix} 3 & 5 & 3 \\ 0 & 0 & 0 \\ -3 & -5 & -3 \end{bmatrix}. \quad (5)$$

Suppose, $Tpixel(i,j)$ represents a series of pixels from the original image. At the same time, K_{sum}, the summation of 3×3 kernel mask values, $kernelV_{(x,y)}$ where x and y values range are from 0 to 2 is calculated (Equation (6)).

$$RGB_{sum} = \sum_{y}\sum_{x} kernelV_{(x,y)} \times Tpixel_{(i,j)} \quad (6)$$

Next the RGB_{sum} is divided into K_{sum} as Equation (9) and added to a gray scale offset value, b. Finally, the RGB_{sum} is converted again into gray scale value (Equation (7)). It has applied Grassmann's law which involves human color perception of an experimental result on (approximately) linear chromatic response. It was founded by Hermann Grassmann (1980).

$$Gray_{(i,j)} = (0.299 \times RGB_{sum}) + (0.587 \times RGB_{sum}) + (0.114 \times RGB_{sum}) \quad (7)$$

This process is called RGB Convolution. The RGB convolution algorithm is described in Algo. 3. The compositions of all new gray scale values is transformed an original clustered image, $img0_{C1}$, as in Figure 8 (a) into a new $black - gray - white$ (b-g-w) image, $img6$, as in Figure 8(b).

1. Firstly, the three color images (Figure 8b) are transformed into a series of binary pixels where $b = black, g = gray, w = white$ are as shown in Figure 9.

For example, a series of pixel for row 6 is as follows:

Algorithm 3. CRLSA-CT: The RGB Convolution Algorithm

```
Input: Set the $3\times 3$ kernel mask value, $kernelV_{(x,y)}$ either kernel matrix RED or
KED as Kernel (8) where $(x,y)\in X=(0,1,2), Y=(0,1,2)$, the bias, $b$, as 128 value.
Suppose the RGB pixel from the original image, $C1$, is obtained at $(i,j)$ as
$Tpixel_{(i,j)}$ and $C1_{counter1}\in C1_{(0,1,\ldots,n-1)}$ where $i,j\in I=(0,1,\ldots,(W_{counter1})-1), J=(0,1,\ldots,(H_{counter1})-1)$,
$n$ is the total of cluster and $counter1$, is the predetermined cluster.
```

$$kernelV_{(X,Y)}=\begin{bmatrix}3&0&-3\\5&0&-5\\3&0&-3\end{bmatrix}\cup\begin{bmatrix}3&5&3\\0&0&0\\-3&-5&-3\end{bmatrix} \quad (8)$$

```
Output: Obtain the new gray scale value, $Gray_{(i,j)}$ and display the convolution
image and keep it into a buffer, $img6$.
1. Execute the RGB Convolution subprocess. {State 1a: Steps 10-10}
2. for ($0\le i\le I$) is true do
3.    for ($0\le j\le J$) is true do
4.       for ($0\le x\le X$) is true do
5.          for ($0\le y\le Y$) is true do
6.             Retrieve the $3\times 3$ kernel mask value, $kernelV_{(x,y)}$ and source RGB
pixel at $(i,j)$ as $Tpixel_{(i,j)}$.
7.             Calculate the $RGB_{sum}$ as in Equation (6).
8.             Set the $kernelV(x,y)=K_{sum}$ as in Equation (9)
9.             $K_{sum}=\sum_y\sum_x kernelV$                                        (9)
10.            Reset the $K_{sum}<0$ as condition below:
   $K_{sum}=\begin{cases}K_{sum} & if\ K_{sum}<0,\\ K_{sum} & otherwise\ K_{sum}.\end{cases}$      (10)
11.            end for
12.            Suppose $b=128$, calculate $RGB_{sum}$ as follows:
   $RGB_{sum}=(\frac{RGB_{sum}}{K_{sum}})+b.$                                     (11)
13.            Reset the $RGB_{sum}$ as condition below:
   $RGB_{sum}=\begin{cases}255 & if\ RGB_{sum}>255,\\ 0 & if\ RGB_{sum}\le 0,\\ RGB_{sum} & else\ 0<RGB_{sum}\le 255.\end{cases}$      (12)
14.         end for
15. end for
16. Derive the new gray scale value, $Gray_{(i,j)}$ as in Equation (7).
17. $img0 \xrightarrow{RGBconvolution} img6$. Obtain the threshold image, $img6$.
18. Execute the RLSA1 subprocess.{ State 1b}
```

$wggwwwbbbggwwwbbbggwwwggggggg\ldots.$

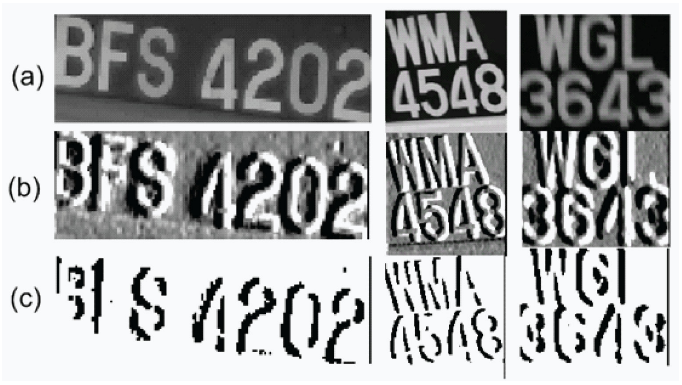
$bbbbgwwwwggggbbbbbbggggwww\cdots$

2. Secondly, the run length of the transformed binary image (Figure 9) is stored and calculated only when $black-gray-white$ $(b-g-w)$ pixels exists consecutively as depicted in Figure 9. The run length, $RunLength$, for each pixel is calculated as illustrated in the following example,

$$3b2g3w-3b2g3w-4b1g4w-6b4g6w\cdots$$

Using Equation (13), the average run length for this line, $aveRunLength$, is calculated as below,

Figure 8. Examples of (a) images of detected license plates (b) image results after applying only RGB Convolution process (c) Image after applying RGB Convolution and TV1 process correspondingly



$$aveRunLength = \frac{\sum_{0}^{countRL} Runlength}{countRL}, \quad (13)$$

where $Runlength$ is the total $b-g-w$ pixels consecutively for each line and $countRL$ is the total number of run length.
Next, the average $aveRunlength$ per line, $aveRLwhole$, as given in Equation (14), are computed.

$$aveRLwhole = \frac{\sum_{0}^{p} aveRunlength}{p}, \quad (14)$$

where $aveRunlength$ istheaverage $Runlength$ for each line and the number of lines or the cluster's height, p, is derived as Equation (15).

$$p = H_{C_{counter1}}. \quad (15)$$

The average number of $CountRL$ per line, $aveCountRLwhole$, are computed as follows:

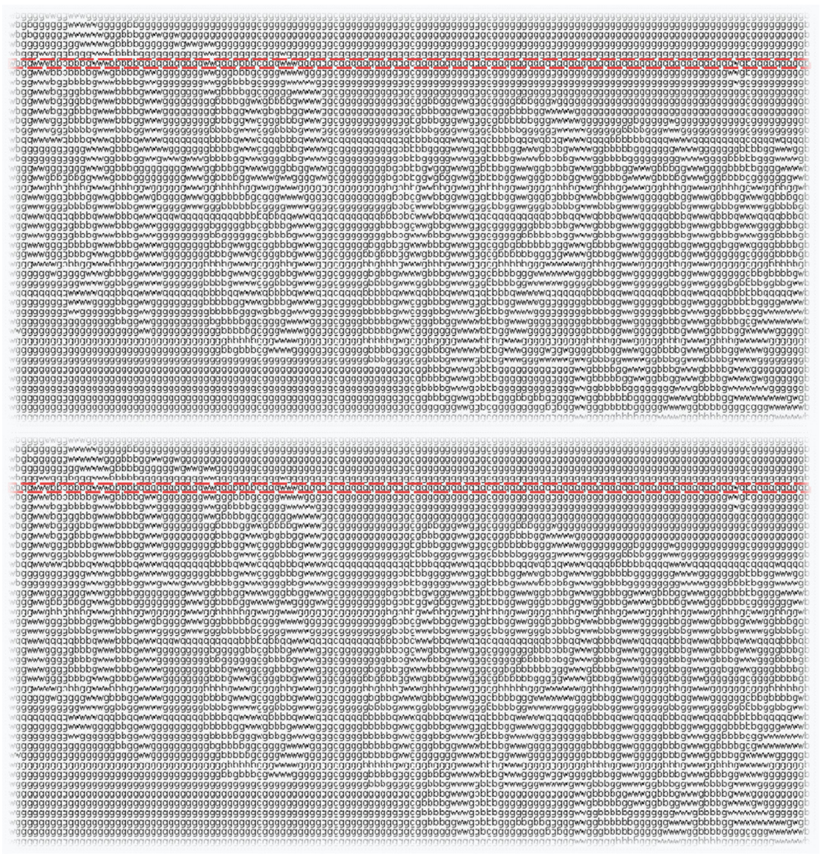
$$aveCountRLwhole = \frac{\sum_{0}^{p} CountRL}{p}, \quad (16)$$

where $CountRL$ is the total number of $RunLength$ for each line.

Lastly, the information obtained from Equation (14) and (16) serves two purposes. Firstly, it is used to construct a set of decision threshold rules at the early stages for the recognition of character and non-character clusters. Secondly, it is used to distinguish between a character and non-character cluster.

Check for the run length of the normal format or case when $(Size \geq 30)$ is true. The information of the run length is obtained after executing several experiments of the CRLSA module on 2500 data images. It is used for setting the decision threshold and also for distinguishing whether the cluster is a character or a non-character. In Figure 10 is shown the minimum, maximum, average values of $aveCountRLwhole$ and $aveRLwhole$ versus different error types, such as *pass* (success in detection), *miss1* (Missing one object/charac-

Figure 9. Example of a string of image representations using black, gray and white $(b-g-w)$ *pixels*



ter), *miss2* (missing two objects/characters), *miss>2* (missing more than two objects /character), *missPart* (Missing some of the part of the object) and *missall* (missing all object). Therefore the decision thresholds are set within minimum, average and maximum values of $aveCountRLwhole$ and $aveRLwhole$ and the member size of the cluster, $Size$, for Algo. 4. There are several cases where, due to illumination, the characters look connected after applying the fixed threshold process. Hence, the CRLSA module still accepts this less member size ($Size < 3$) of the cluster but is requires to permit the second check process called RLSA2 as described in the following section.

Run Threshold Value One (TV1) and Check the Second Run Length Smoothing Algorithm (RLSA2)

As mentioned before, this process covers two sub-processes: (1) Run TV1 and (2) Check the RLSA2. The Run TV1 sub-process is executed before the Check RLSA2 sub-process. Both sub-processes are elaborated in detail in the following paragraphs:

a. Run TV1

This submodule is the third step in the CRLSA module where $(b-g-w)$ image is transformed into $(b-w)$ image by using a simple threshold-

Algorithm 4. CRLSA-CT:The RLSA1 algorithm

```
Input: Set the original image, $img0$, and a RGB convolution image, $img6$, for
cluster $C1_{counter1}$ where $counter1$ is the predetermined cluster into two buffers
correspondingly. Retrieve the number of blobs in the cluster, $Size_{C1}$, and
calculate the average run length, $aveRLWhole$ and average count of the run
length, $aveCountRLwhole$ for $C1$ cluster image.
Output: Obtain the status of cluster $S$, either a character or non-character
cluster.
1. Execute The RLSA1 subprocess.{State 1b}
2. if($Size_{C1} < 3$) is true then
3.     if $((aveCountRLwhole < 2) \cup (aveCountRLwhole > 20) \cup (aveRLwhole < 3) \cup (aveRLwhole > 17)$ ) is
true then
4.        Execute the TV1 and RLSA2 subprocess subsequently. {State 1c}
5. else
6.         Set the status, $S$ as a non-character image.
7.         Execute the MCT submodule. {State 1d}
8.  end if {Check for the run length of cluster of the special format or case
when $(Size < 3)$ is true. Sometimes, some of the characters are connected to each
other and therefore the the member size is reduced than actual number of
characters. }
9. else if
10. $((aveCountRLwhole < 0.2) \cup (aveCountRLwhole >= 19) \cup (aveRLwhole < 3) \cup (aveRLwhole > 36))$ is true then
11.          Set the status, $S$ as a non-character image.
12.          Execute the MCT submodule. {State 1d}
13. else if
14.          $((aveRLwhole < 11) \cup (aveCountRLwhole < 7) \cup (aveCountRLwhole > 9))$ is true then
15.          Execute the TV1 and RLSA2 subprocess subsequently. {State 1c}
16. end if
17.          Set the status, $S$ as a character image and keep the winner cluster
index, $WC_{counter1}$ in the Winner Sequence array.
18.          Execute the MCT submodule.{State 1d }
19. endif {Check for the run length of the normal format or case when  is true}
20. end if
21. return S.
 Note $\cup$ is logical OR
```

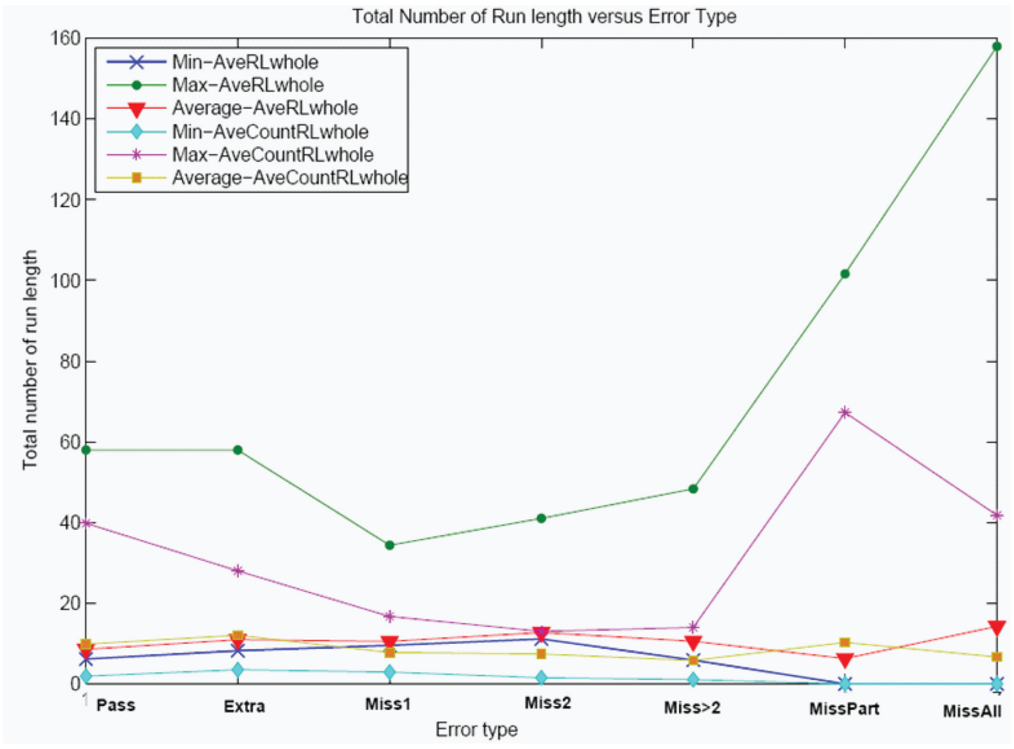
ing process. In other words, the threshold value one, Ω_1 is applied during the TV1 process. The resultant image is depicted in Figure 8(c).

b. Check RLSA2

Previous researchers only calculated the distance or the Run length based on a binary image of the original image (Zhong & Yan, 1999; Xi, Hu, & Wu, 2002; Wong et al., 1982). After applying CCL, they formulated the distance vertically and horizontally as mentioned above. On the other hand, this work introduces another way of checking character or non-character by calculating horizontal and vertical lines. The threshold image, $img7$, as depicted in Figure 8(c), contained *b*lack and white $(b - w)$ pixels. Only the white pixels are calculated as the run length. The white pixels are calculated horizontally and vertically as explained below:

1. Firstly, the `black and white' image is transformed into binary image with $b$ for `0' gray pixel value and w for `255' gray pixel value. For example, the result of a binary transformed image into a series of $b - w$ pixels of row 6 in Figure 11 is as follows, ***wbwwwwbbbbbbwwwwbbbbbw-wwwwwwwwwwwwwwwbbbbwww-ww... ...wwwwwwwwwwwwwwwwwwwww-wwwwwwwwwwwwwwwwwwwwwww... ...... wwwwwwwwwwwbwwwwwwwwwwwb.***

Figure 10. The graph of decision threshold settings for RLSA2 Algorithm



2. Secondly, the run length for each white and black pixels is represented as follows: ***1**w1b4w7b4w4b**16**w3b**84**w1b**10**w1b*.
3. Thirdly, suppose $counter1$ is the chosen clusterwhere $C1_{counter1} \in C1(0,1,\ldots n-1)$. Only the run lengths of the *white* pixels, $runlength$, are calculated based on Equation (17). The $runlength$ is divided into three categories: (i) Long lines, (ii) Medium lines, and (iii) Short lines. Examples of the horizontal decision rules, H_{Dec}, are summarized as below,

$$q = W_{C_{counter1}}, \tag{17}$$

where $C_{counter1}$ is chosen cluster and q is the number of columns or cluster's width,

$$H_{Dec} = \begin{cases} \mathrm{H}Long & if \quad runlength \geq 0.3 \times q, \\ \mathrm{H}Medium & if \quad 0.01 \times q \leq runlength \leq 0.3 \times q, \\ \mathrm{H}Short & if \quad runlength \leq 0.01 \times q, \end{cases} \tag{18}$$

4. Next, the number of occurrence for each H_{Dec} category for each line is also computed horizontally as follows:
   - The total of long horizontal lines per line, $totalHLong$ is calculated using Equation (19),

$$totalHLong = \sum_{0}^{q} HLong. \tag{19}$$

   - The total medium horizontal lines per line, $totalHMedium$ is calculated using Equation (20),

$$totalHMedium = \sum_{0}^{q} HMedium, \tag{20}$$

- The total short horizontal lines per line $totalHShort$ is calculated using Equation (21),

$$totalHShort = \sum_{0}^{q} HShort. \tag{21}$$

5. After calculating the horizontal run lengths for all lines, then the vertical run lengths are also continued. The vertical projection decision rules, V_{Dec}, are as follows,

$$\mathrm{V}_{Dec} = \begin{cases} \mathrm{V}Long & if \quad runlength \geq 0.4 \times p, \\ \mathrm{V}Medium & if \quad 0.05 \times p \leq runlength \leq 0.4 \times p, \\ \mathrm{V}Short & else \quad runlength \leq 0.05 \times p. \end{cases} \tag{22}$$

6. Then, the number of occurrence for each V_{Dec} category for each line is also computed vertically as follows:
   - The calculation for the total long vertical lines per column, $totalVLong$, is given as,

$$totalVLong = \sum_{0}^{p} VLong, \tag{23}$$

   - The total medium vertical lines per column, $totalVMedium$, is given as,

$$totalVMedium = \sum_{0}^{p} VMedium, \tag{24}$$

   - The total short vertical lines per column, $totalVShort$ is given as,

$$totalVShort = \sum_{0}^{p} VShort. \tag{25}$$

The TV1 and RLSA2 algorithm is described in Algorithm 5.

Similar to RLSA1 decision threshold settings, the RLSA2 is proposed to check whether the whites spaces are the majority pixels in the cluster image or otherwise. If yes, then the cluster should be indicated as a non-character image or otherwise. Therefore, the summation of the $totalHLong$, $totalHMedium$ and $totalHShort$ for horizontal lines and the summation of the $totalVLong$, $totalVMedium$ and $totalVShort$ for vertical lines are set based on a ratio of the cluster's height and width. After running several experiments, the constant values of 0.01 and 0.3 of the cluster's height while 0.4 and 0.05 of the cluster's width, that to be indicated as long, medium and short categories for horizontal and vertical respectively, are acceptable.

Some advantages of using the CRLSA approach are as follows:

1. This technique can distinguish successfully the structure and non-structure images (Abdullah et al., 2010).
2. It can still detect the license plate even though the image is blurred or is having a fusion problem (Abdullah et al., 2007),
3. All clusters are treated equally. However, Siah et al. (1999) and Siah (2000) were able to treat only the cluster with the most member size as the winner cluster.
4. The acceptable run lengths can distinguish the license plate characters (Abdullah et al., 2007).

Some limitations of the CRLSA approach are as follows:

1. Memory leak if the cluster size exceeds the memory limitation. However, this can be solved by resizing the cluster size.
2. Sometimes the run length of other characters such as the car trademark or the advertisement may also consider as the winner cluster. However, this can be solved in the Character Classification module. It

Algorithm 5. CRLSA-CT:The TV1 and RLSA2 Algorithms

```
Input: Suppose $C1_{counter1} \in C1_{(0,1,\ldots,n-1)}$ where $counter1$ is the chosen cluster and $n$ is
the total number of clusters, $img0$ is the original image and $img6$ is the RGB
convolution image of specific cluster.
Output: Obtain the TV1 image, $img7$, and the status of cluster $S$ image, either
a character or non-character cluster.
1. Execute the TV1 and RLSA2 subprocesses subsequently.{State 1c }
2. $img6_{C1} \xrightarrow{TV1} img7_{C1}$.
3. Retrieve the cluster's height, $H_{C1_{counter}}$ and cluster's size, $Size_{C1_{counter}}$. Calculate
the $totalHLong$, $totalHMedium$, $totalHShort$, $totalVLong$, $totalVMedium$ and $totalVShort$.
4. if $(Size_{C1_{counter1}} > 2)$ is true then
5.     if$(H_{C1_{counter}} > 11)$ is true then
6.        if $(totalHLong > totalHMedium)$ is true then
7.             Set the status, $S$ as a non-character image
8.        else
9.        if $((totalHShort < totalHMedium) \cap (totalHShort \neq 0) \cap ((totalHMedium - totalHShort) > 0.05 * totalHMedium))$ is true then
10.            Set the status, $S$ as a character image and keep the winner
cluster index, $WC_{counter1}$ in the Winner Sequence array.
11.       endif
12.       else
13.       if $((totalVShort < totalVMedium) \cap (totalVMedium - totalVShort > 0.01 * totalVMedium) \cap (totalVShort \neq 0))$
is true then
14.                 Set the status, $S$ as a character image and keep the winner
cluster index, $WC_{counter1}$ in the Winner Sequence array.
15.            endif
16.       else
17.            Set the status, $S$ as a non-character image.
18.       endif
19.    else
20.            Set the status, $S$ as a non-character image.
21.    endif {Check the run length of the cluster for the special case
($Size < 3$). Sometimes, some of the characters are connected to each other there-
fore the member size of cluster is reduced than actual number of character.}
22. else
23.    if  $(totalHLong > totalHMedium)$ is true then
24.       Set the status, $S$ as a character image and keep the winner cluster
index, $WC_{counter1}$ in the Winner Sequence array.
25.    else
26.       Set the status, $S$ as a non-character image.
27.    end if
28. endif {Check the run length of the cluster for the normal case of member size ($Size \geq 3$ blobs).}
29. return $S$.
30. Execute the MCT submodule. {State 1d }
   Note $\cup$is logical OR
```

can accept the winner blobs if they are recognized in the format of "$XXX.@@@@@@@$" where X represents characters and @ represents numbers. This module is called Post-Processing.

3. The processing time may be lengthy because it is linearly correlated to the size of the cluster. The processing time may decrease if the CRLSA source code is executed as vector programming instead of array programming.

RESULTS AND ANALYSIS

According to the preliminary experiments that were applied (Abdullah, Khalid, Omar, & Yusof, 2009), the segmentation phase which consists of detection and segmenting characters, contributes more errors than the classification phase. Therefore, an alternative technique to increase the detection rate was proposed. Since the conventional method using cluster is also an acceptable technique, the selection of the winner cluster using CRLSA is applied and the most maximum winning cluster would be chosen as the winner cluster. This proposal has been elaborated in previous sections. There are two objectives of these experiments. The main objective is to propose an enhanced technique to locate the license plate in an image. Meanwhile, the second objective is to compare the proposed technique (CRLSA) with other existing techniques.

Previously, all the techniques except CRLSA technique are used for LPD system, such as (1) multi-clustering by Siah (2000), Square/contour detection by Saldago, Menende, Rendon, and Garcia (1999), (2) Morphology by Han et al. (2003), (3) Hough transform by Miyamoto, Nagano, and Tamagawa (1991), and (4) Radon Transform by Kong, Liu, Lu, Zhou, and Zhao (2005). Table 1 has shown the accuracy rates of different researchers pertaining to license background color (Figure 12).

Eventually, six experiments were conducted based on: Cluster, CRLSA, Square/contour detection, Morphology, Hough Transform and Radon Transform. The Cluster and CRLSA used the fixed threshold technique and blob labellings concept; meanwhile other techniques applied the same edge detector, referred to as the Canny edge detector, to compute the contours in the image. All the techniques are first optimized for images of the Malaysian license plate. Table 2 concludes the overall system configurations.

Since Malaysian vehicles consists of standard and special license plate, therefore standard (with black background) and special (with white background) formats were chosen as the image datasets. Examples of Malaysian license plate format images can be seen in Figure 1. The percentage of special license plates is insignificant therefore the sample image datasets for the special plates were lesser than the standard fonts. Total image datasets for standard and special plates were 1235 and 20 respectively. The images were taken in normal conditions and the average distance of object from the camera is less than 5 meters. All six different detection techniques were run onto the same datasets.

Results of the six different experiments are shown in Figure 13 and Table 3. Meanwhile examples of license plate detection using different approaches namely Clustering, CRLSA, Square, Morphology, Hough and Radon Transform are shown in Figure 11 through Figure 14 respectively.

From the Table 3, it can be concluded that CRLSA has achieved the highest detection rate (96.43%) compared to other techniques for the standard Malaysian license plate format. Meanwhile, the techniques of conventional Cluster, Morphology, Hough and Radon Transform give 94.17%, 85%, 81.9%, and 73.7% accuracy rates

Table 1. Previous results using different approaches in objection detection phase

Techniques	Accuracy	Designed by	License Background Color
Clustering	89.63%-97.78%	Siah (2000)	Black
Square or contour	Nil	Jeong et al. (2006)	Yellow
Morphology	98.38%	Han et al. (2003)	Yellow
Hough Transform	92.20%	Soh et al. (1994)	Yellow
Radon Transform	96.20%	Kong et al. (2005)	Yellow

respectively for the standard format. However, previous research had improved from 89.63% up to 97.78%, 98.38%, 92.20%, and 96.20%, respectively, with mostly yellow color backgrounds (except for the Cluster technique). Unfortunately, the square/contour technique produced the worst result (54.79%) for the black background or standard Malaysian license plate images. These outstanding results in different perspectives are explained as shown.

Several advantages and disadvantages have been observed with the proposed approaches applying the RGB convolutions and PED with 128 gray scale offset. Some specific advantages are:

1. Even though the original image of the back or frontal car has a fusion problem, such as illumination, blur, rotated, skewed, CRLSA can still successfully detect the locations of the license plate as shown in Figure 14.
2. CRLSA is able to detect the precise location of the license plate even though the license plates's characters are connected or too close to each other.
3. CRLSA can quantify the run length's acceptance to be indicated as license plate number.
4. CRLSA can reduce the number of missing numbers and letters.

However, the disadvantages of CRLSA are:

1. **Passing rate.** The success rate for CRLSA were only slightly increased compared to Cluster. This is because CRLSA will consider all blobs in the image. However, the Cluster technique only takes the maximum cluster members as the winner cluster.
2. **Incorrect detection rate.** The number of incorrect detection for CRLSA is fairly high which may be due to cluster similarity in characteristics such as its run length value.
3. **Processing time.** RGB convolution with a new edge detector is very time consuming

Figure 11. Example of a string of image representations for threshold value one image using black and white $(b-w)$

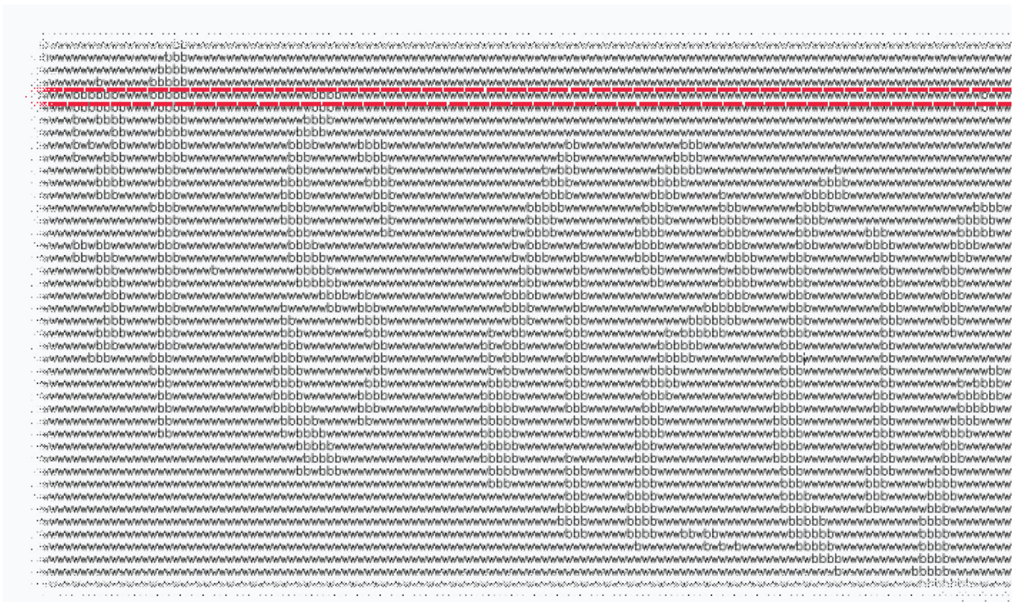
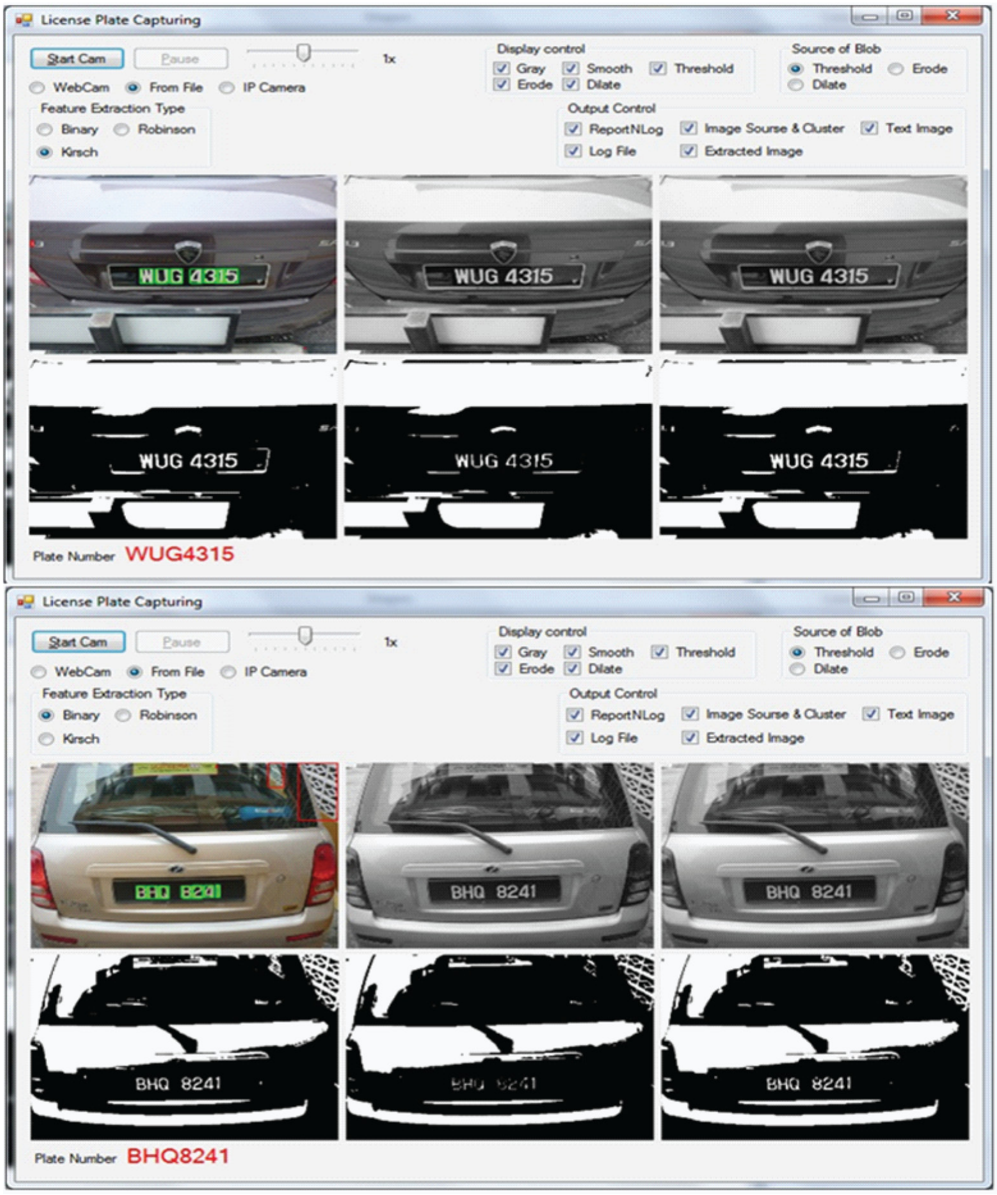


Figure 12. Examples of the correct LPD using CRLSA module



License Plate Capturing

Start Cam Pause 1x

WebCam [✓ From File] IP Camera

Feature Extraction Type: Binary Robinson [✓ Kirsch]

Display control: [✓ Gray] [✓ Smooth] [✓ Threshold] [✓ Erode] [✓ Dilate]

Source of Blob: [✓ Threshold] Erode Dilate

Output Control: [✓ ReportNLog] [✓ Image Sourse & Cluster] [✓ Text Image] [✓ Log File] [✓ Extracted Image]

Plate Number WUG4315

License Plate Capturing

Start Cam Pause 1x

WebCam [✓ From File] IP Camera

Feature Extraction Type: [✓ Binary] Robinson Kirsch

Display control: [✓ Gray] [✓ Smooth] [✓ Threshold] [✓ Erode] [✓ Dilate]

Source of Blob: [✓ Threshold] Erode Dilate

Output Control: [✓ ReportNLog] [✓ Image Sourse & Cluster] [✓ Text Image] [✓ Log File] [✓ Extracted Image]

Plate Number BHQ8241

because the calculations of getting new gray scale output requires every pixel of the original image to be analyzed.

4. **Memory allocation.** Quite often memory leaks when using RGB convolution because it requires high storage.

In conclusion, CRLSA can boost the detection rate of license plates by following the suggestions below:

1. Instead of using fixed thresholding in the blob labellings, adaptive zoning thresh-

Table 2. Configuration for LPS technique comparisons

Setting feature	Clustering	CLRSA	Square/contour
Image	CCL	CCL	
processing filter	Blob Analysis	Blob Analysis	Contour tracking
		Homormophic	Dilation filter
Thresholding filter	Fixed	Fixed	Repetitive
Edge detection	Nil	KED or RED	Canny Edge
Standard sample test	1235	1235	1235
Special sample test	20	20	20
Software / Library	C++, VSDP	C++, VSDP	OpenCV
Setting feature	**Morphology**	**Radon transform**	**Hough transform**
Image	Top hat filter		
processing filter	Bottom hat filter		
	Subtract filter	Radon transform	Hough transform
	Opening filter		
	Blob Analysis		
Thresholding filter	Fixed	nil	Adaptive
Edge detection	Canny Edge	Sobel Edge	Canny Edge
Standard sample test	1235	1235	1235
Special sample test	20	20	20
Software / Library	MATLAB	MATLAB	MATLAB

Table 3. Experiment results using clustering, CRLSA, square/contour, morphological, Hough Transform, and Radon Transform subsequently

Malaysian	CLUSTERING		CRLSA		SQUARE/CONTOUR	
License plate	**Standard**	**Special**	**Standard**	**Special**	**Standard**	**Special**
Non detect	72	8	44	3	916	6
Detect	1163	12	1191	17	319	14
Total image	1235	20	1235	20	1235	20
Detect %	0.94	0.6	0.96	0.85	0.26	0.7
Non detect %	0.06	0.4	0.04	0.15	0.74	0.3
Malaysian	**MORPHOLOGICAL**		**HOUGH TRANSFORM**		**RADON TRANSFORM**	
License plate	**Standard**	**Special**	**Standard**	**Special**	**Standard**	**Special**
Non detect	571	3	232	5	340	6
Detect	692	17	1050	15	956	14
Total image	1263	20	1282	20	1296	20
Detect %	0.55	0.85	0.82	0.75	0.74	0.7
Non detect %	0.45	0.15	0.18	0.25	0.26	0.3

Figure 13. Graph of accuracy versus standard and special format

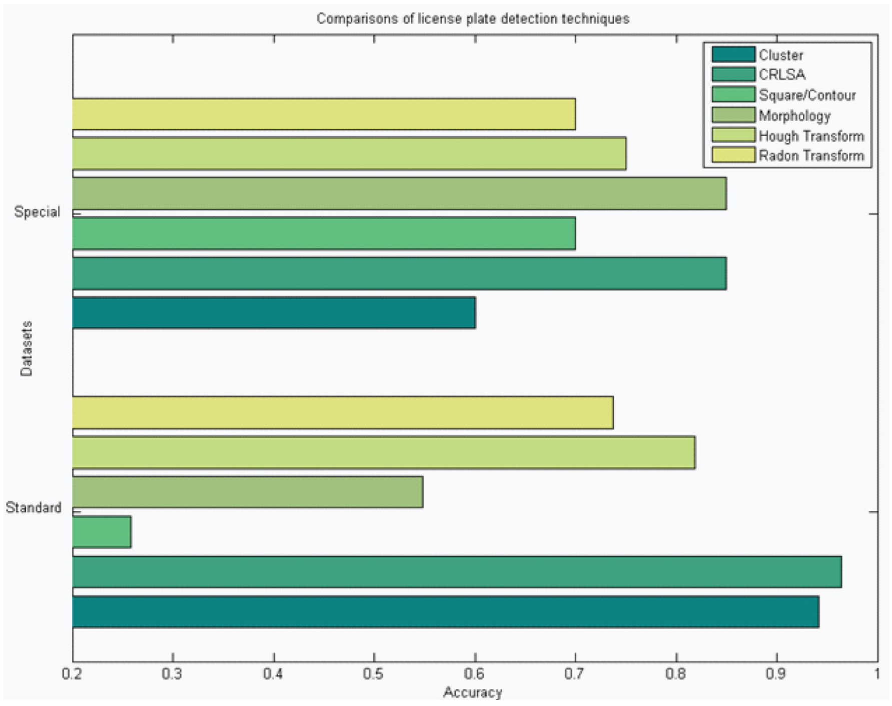
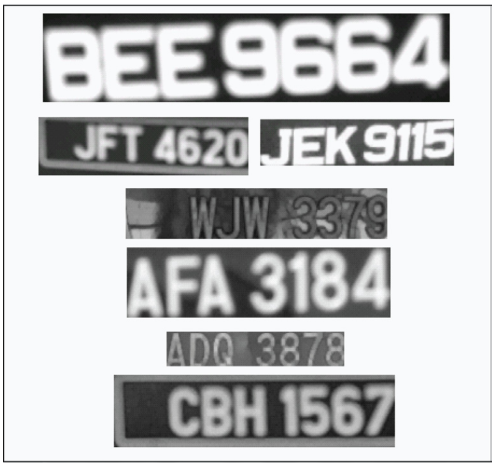


Figure 14. Fusion Images that has been successfully detected by LPD system



olding can also help CRLSA improve its detection rate.

2. RGB convolutions can apply other edge detectors with 128 gray scale offset.
3. Before applying RGB convolutions and Horizontal and Vertical Projections, checking the candidate winner cluster from the original image using binary projections may increase the whole performance.
4. Incorporating uncertainty value while calculating the maximum number of blobs and run length of each cluster may increase the overall performance.
5. Resizing only the image with over fitting candidate winner cluster before attempting RLSA checking can reduce the memory leak problem.

CONCLUSION

Even though previous researchers obtained remarkable results on their image data as noted in Table 1, this experiment had proved that with different formats of license plates, such as the Malaysian license plates, some techniques failed to detect the location of license plates as well as CRLSA (Table 3). Standard license plates with black backgrounds and letters with lighter color (normally white) can easily be detected using CRLSA and Cluster due to the following factors:

1. All objects, with any shape on the image, which are subjected to analysis by the thresholding concepts allow all blob candidates to distinguish the darker pixels as background meanwhile the lighter pixels as the candidates. Therefore, grouping similar blob identities into a group can simplify the searching process. Unlike other techniques, square/contour and morphology techniques only consider objects with four edges connected to each other. Hough Transform and Random Transform would only consider straight lines objects.
2. All known techniques are prone to the selection value of the thresholding process (the image transformation from grayscale pixels to only black and white pixels) but CLRSA and Cluster are less sensitive compared to other techniques because it applies fixed threshold values and only certain small blob candidates will lose the information.
3. Square/contour, Morphology, Hough and Radon Transform are most sensitive to the threshold selection value because Canny Edge operator is a second derivative operator which is highly dependent on the double threshold process. As a result, the required object shape by each technique may be destroyed or distracted and affects the overall detection performance.
4. Illumination is a major problem in image processing. Even though illumination can spoil the wanted blob candidates, normally only some portion of the license plates are affected when using CRLSA and Cluster, and the approximate location can still be located. Similar to CRLSA and Cluster, the detection rates are quite comparable using Hough Transform and Radon Transform technique. Not all information will be lost because the lines of the car body bonnet can still be traced which also leads to the correct license plate location. In contrast to Square/contour and morphology techniques, determining line or square object may eliminate all information (characters) on the license plate.

On the other hand, for special license plates (or white background), both CRLSA and Morphology techniques obtained the highest accuracy rate about 85%. This is followed by Hough transform, Square/Contour Technique and Radon Transform in a special format case. Lighter background are less prone to illumination sensitivity because during the thresholding process, the illumination may always be indicated as brighter pixels (grayscale between 0 until 127 pixel color) similar to its background colour. Therefore, the demanding object shape should be difficult to trace. Unlike CRLSA and Cluster, the white tranformed pixels due to illumination cause numbers and letters to look connected or destroys each blob candidates characteristics such as the height, width, distance between the blob candidates. Despite that, Hough Transform technique has the capability to fill gaps between lines. Unlike Square/contour and Morphology Technique, Hough transform's capability can definitely ease the line searching.

From the discussion above, CRLSA technique is better compared to others in black or white background license plates as accuracy rates for both cases are 96.43% and 85% correspondingly as shown in Table 3 and Figure 13.

ACKNOWLEDGMENT

Special thanks to late Allahyarham Marzuki Khalid, Khairuddin Omar (both are the supervisor and co-supervisor of the 1st author) and our colleagues: Norzi, Faridatul, Norzu, Dilah, Rosmizan and Hidayah who had worked hard in collecting the image data in various states. Not also forgetting to Christian Viard Gaudin from Institut de Research on Communication and Cybernetics, Ecole Polytechnique, Universite de Nantes, France and Ahmed Patel who has consistently given the best critics toward excellent journal writings. This project research is funded by FRGS/1/2012/SG05/UKM/02/8 grant entitled Generic Object Localization Algorithm for Image Segmentation and UKM-GGPM-ICT-119-2010 grant entitled Logo and Text detection based on moving object vision guided.

REFERENCES

Abdullah, S. N. H. S. PirahanSiah, F., Abidin, N. H. Z., & Sahran, S. (2010). Multi-threshold approach for license plate recognition system. In *Proceedings of the WASET International Conference on Signal and Image Processing* (pp. 804-808).

Abdullah, S. N. H. S., Khalid, M., Omar, K., & Yusof, R. (2009). Pengesanan nombor plat kenderaan menggunakan alkhwarizmi Gugusan dan Kelancaran Jarak Larian (GKJL) (License plate detection using Cluster Run Length Smoothing Algorithm (CRLSA)). *Jurnal Sains Malaysiana*, *38*(4), 577–587.

Abdullah, S. N. H. S., Khalid, M., Yusof, R., & Omar, K. (2007). Comparison of feature extractors in license plate recognition. In *Proceedings of the First Asia International Conference on Modelling & Simulation* (pp. 502-506).

Abidin, N. H. Z., Abdullah, S. N. H. S., Sahran, S., & PirahanSiah, F. (2011). License plate recognition with multi-threshold based on entropy. In *Proceedings of the International Conference on Electrical Engineering and Informatics* (pp. 1-6).

Bataineh, B., Abdullah, S. N. H. S., & Omar, K. (2011). An adaptive local binarization method for document images based on a novel thresholding method and dynamic windows. *Pattern Recognition Letters*, *32*(14), 1805–1813. doi:10.1016/j.patrec.2011.08.001

Bataineh, B., Abdullah, S. N. H. S., & Omar, K. (2012). A novel statistical feature extraction method for textual images: Optical font recognition. *Expert Systems with Applications*, *39*(5), 5470–5477. doi:10.1016/j.eswa.2011.11.078

Bay, H., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded up robust features. In *Proceedings of the 9th European Conference on Computer Vision* (pp. 404-417).

Han, P., Han, W., Wang, D.-F., & Zhai, Y.-J. (2003). Car license plate feature extraction and recognition based on multistage classifier. In *Proceedings of the International Conference on Machine Learning and Cybernetics* (pp. 128-132).

Jeong, T., Kang, J., & Choi, Y. S. (2006). Implementation of embedded system for intelligent image recognition and processing. In M. Gavrilova, O. Gervasi, V. Kumar, C. J. K. Tan, D. Taniar, A. Laganá, Y. Mun, & H. Choo (Eds.), *Proceedings of the International Conference on Computational Science and Its Applications* (LNCS 3980, pp. 993-999).

Kong, J., Liu, X., Lu, Y.-H., Zhou, X., & Zhao, Q. (2005). Robust license plate segmentation method based on texture features and Radon transform. In S. Zhang & R. Jarvis (Eds.), *Proceedings of the 18th Australian Joint Conference on Advances in Artificial Intelligence* (LNCS 3809, pp. 510-519).

Miyamoto, K., Nagano, K., & Tamagawa, M. (1991). Vehicle license-plate recognition by image analysis. In *Proceedings of the IEEE International Conference on Industrial Electronics, Control and Instrumentation* (pp. 1734-1738).

Nagy, G. (1968). Preliminary investigation of techniques for automated reading of unformatted text. *Communications of the ACM*, *11*(7), 480–487. doi:10.1145/363397.363417

Papamarkos, N., Tzortzakis, J., & Gatos, B. (1996). Determination of run-length smoothing values for document segmentation. In *Proceedings of the IEEE International Conference on Electronics, Circuits, and Systems* (pp. 684-687).

PirahanSiah. F., Abdullah, S. N. H. S., & Sahran, S. (2010). Adaptive image segmentation based on peak signal-to-noise ratio for a license plate recognition system. In *Proceedings of the International Conference on Computer Applications and Industrial Electronics* (pp. 468-472).

Prabuwono, A. S., & Idris, A. (2008). A study of car park control system using optical character recognition. In *Proceedings of the International Conference on Computer and Electrical Engineering* (pp. 866-870).

Romero, D. D., Prabuwono, A. S., & Taufik, H. A. (2011). A review of sensing techniques for real-time traffic surveillance. *Journal of Applied Sciences*, *11*(1), 192–198. doi:10.3923/jas.2011.192.198

Saldago, L., Menende, J. M., Rendon, E., & Garcia, N. (1999). Automatic car plate recognition through vision engineering. In *Proceedings of the IEEE 33rd Annual International Carnahan Conference on Security Technology* (pp. 71-76).

Shapiro, V., Gluhchev, G., & Dimov, D. (2006). Towards a multinational car license plate recognition system. *Journal of Machine Vision and Applications*, *17*(3), 173–183. doi:10.1007/s00138-006-0023-5

Siah, Y. K. (2000). *A design of an intelligent license plate recognition* (Unpublished master's thesis). Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia.

Siah, Y. K., Haur, T. Y., Khalid, M., & Ahmad, T. (1999). Vehicle licence plate recognition by fuzzy artmap neural network. In *Proceedings of the World Engineering Congress*.

Soh, Y. S., Chun, B. T., & Yoon, H. S. (1994). Design of real time vehicle identification system. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* (pp. 2147-2152).

Wong, K. Y., Casey, R. G., & Wahl, F. M. (1982). Document analysis system. *IBM Journal of Research and Development*, *26*(6), 647–657. doi:10.1147/rd.266.0647

Xi, J., Hu, J., & Wu, L. (2002). Page segmentation of Chinese newspapers. *Pattern Recognition*, *35*(12), 2695–2704. doi:10.1016/S0031-3203(01)00248-5

Xu, Z., & Zhu, H. (2007). An efficient method of locating vehicle license plate. In *Proceedings of the Third International Conference on Natural Computation* (pp. 180-183).

Zhong, D. X., & Yan, H. (1999). Pattern skeletonization using run-length-wise processing for intersection distortion problem. *Pattern Recognition Letters*, *20*(8), 833–846. doi:10.1016/S0167-8655(99)00047-1

Siti Norul Huda Sheikh Abdullah obtained her computing degree from University of Manchester Institute of Science and Technology (UMIST), UK and her master degree specializing in Artificial Intelligence from Universiti Kebangsaan Malaysia (UKM). She completed her PhD Degree in Image Processing at Universiti Teknologi Malaysia. Holding an associate professor post in Faculty of Information Sciences and Technology, she also serves as a research fellow at Center for Artificial Intelligence Technology, UKM. Her research interests are pattern recognition, computer vision and robotics, and document analysis and recognition.

Muhammad Nuruddin Sudin obtained his Artificial Intelligence degree in Universiti Kebangsaan Malaysia in September 2011. He is currently pursuing his master degree in Computer Science in the same university and is a member of the Pattern Recognition Group under Center for Artificial Intelligence Technology (CAIT). His research interests are robotic and computer vision.

Anton Satria Prabuwono received his PhD in Industrial Computing from Universiti Kebangsaan Malaysia where he is currently an Associate Professor. He is an author and co-author for more than 130 papers in book chapters, journals and conference proceedings. He is a member of IACSIT Systems, Man & Cybernetics Society and IEEE Robotics & Automation Society. In general, his research interests include machine vision, robotics and autonomous systems.

Teddy Mantoro obtained a PhD, an MSc, and a BSc, all in Computer Science. He was awarded a PhD from School of Computer Science, the Australian National University (ANU), Canberra, Australia. Currently, he is an Associate Professor in Advanced Informatics School, Universiti Teknologi Malaysia. He has filed four certificate Malaysian patents in credit to his name and published 3 books and more than 100 papers in book chapters, journals and conference proceedings. His research interest is in pervasive/ubiquitous computing, context aware computing, mobile computing and intelligent environment.