

# The Use Of Gabor Filter And Back-Propagation Neural Network For The Automobile Types Recognition

Gregorius Satia Budhi  
Informatics Department  
Petra Christian University  
Siwalankerto 121-131  
Surabaya 60236, Indonesia  
(62-31) 2983455  
greg@petra.ac.id

Rudy Adipranata  
Informatics Department  
Petra Christian University  
Siwalankerto 121-131  
Surabaya 60236, Indonesia  
(62-31) 2983455  
rudya@petra.ac.id

Fransisco Jimmy Hartono  
Informatics Department  
Petra Christian University  
Siwalankerto 121-131  
Surabaya 60236, Indonesia  
(62-31) 2983455  
fr8jimmy@yahoo.com

## ABSTRACT

Type of automobile is the general factor that makes automobile different from each other. But conventional sensor cannot detect the automobile and recognize its type. Because of those reasons, we made an experiment application that can count the number and recognize automobile based on its type. This application uses gabor filter for feature extraction and back-propagation neural network for training and recognizing type of automobile. The experiment was done using various parameters for back-propagation neural network and gabor filters. The experimental result shows that the best error rate of recognition results is 16%. It's done with the brightness condition not too low or high.

## Keywords

Gabor Filter, Back-propagation Neural Network, Automobile Type Recognition.

## 1. INTRODUCTION

Along with technological advances, nowadays computer science relating to digital image processing and computer vision also has been implemented to help human's work. Therefore in this experiment, we developed an application which can calculate the number of cars and identify them based on their types. Input of this application is a video of the car entering parking area and being taken by using a video camera / webcam. The beginning process of this application is separating car objects from the other objects that have been captured by the video camera / webcam. Furthermore, this application will count the number of cars that has been successfully detected and recognize its type. The number of cars passing by and the type of the cars will be the output from this application.

This research was inspired from another research that has been done by our colleagues majoring in electrical engineering from Petra Christian University in 2001 [9]. On that research, they have examined the same topic, namely Vehicle Type Recognition. On that research, a recognition process is done by using gabor filter and template matching method. In our research, we change the method of recognition using back-propagation neural network, and reevaluated the best parameter / configuration of gabor filters.

## 2. IMAGE PROCESSING

### 2.1 Grayscale

Grayscale is the one technique in digital image processing to flatten pixel values of the three RGB values into one same value. This technique can be done using YUV color system, by converting RGB to YUV and then take the Y component (illumination) [1]. This is done by using equation 1.

$$gray = Y = 0.2989 \times R + 0.5870 \times G + 0.1140 \times R \quad (1)$$

### 2.2 Low Pass filter

Low pass filter is used to give the smoothing (blurring) effects in the image. Besides, low pass filter also can be used to eliminate noise in the image [8]. The template 3x3 of low pass filter is shown at Formula (2) [4].

$$template = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (2)$$

### 2.3 Background Subtraction

Background subtraction can be used to identify a motion between two images or more. In order to use the background subtraction technique, all images must be capture in time as close as possible without changes in light conditions and use a fixed background. By doing the background subtraction then a moving object can be detected [2].

$$g(x,y) = \begin{cases} f(x,y) & \text{if } |f(x,y) - b(x,y)| > threshold \\ 0 & \text{if otherwise} \end{cases} \quad (3)$$

Descriptions:

$g(x, y)$  = Result of pixel's value

$f(x, y)$  = Foreground pixel's value

$b(x, y)$  = Background pixel's value

### 2.4 Median filter

Median filter is used to reduce random noise without give any blurring effect at the line of the image like low pass filter. The process that occurs in the median filter is as follows, for each pixel value in several areas that contained in the template on the image will be sorted first, and then will be searched for its median

value. This median value is become the gray level value for the result image [4].

## 2.5 Gabor Filter

Gabor filter is made under the concept of the mammalian cortical simple cells [11]. By using gabor filter each point on the image will be processed to obtain its vector feature. 2-D gabor filter is harmonic oscillator that obtained by modulating 2-D sine wave at a particular frequency and orientation with a Gaussian envelope.

Gabor filter can be used to capture frequency information. By adjusting gabor filter at the specific frequency and direction, then the information of local frequency and orientation can be obtained [3]. Gabor filter can be divided into two different equations, one for describing the real part and the other to describe the imaginary part of gabor filter [10].

$$\Psi_r(x, y, w_0, \theta) = \frac{1}{2\pi\sigma^2} \exp\left\{-\left(\frac{x'^2 + y'^2}{\sigma^2}\right)\right\} \times \left[\cos w_0 x' - e^{-w_0^2 \frac{\sigma^2}{4}}\right] \quad (4)$$

$$\Psi_i(x, y, w_0, \theta) = \frac{1}{2\pi\sigma^2} \exp\left\{-\left(\frac{x'^2 + y'^2}{\sigma^2}\right)\right\} \times \sin w_0 x' \quad (5)$$

where:

$$x' = x \cos \theta + y \sin \theta \quad (6)$$

$$y' = -x \sin \theta + y \cos \theta \quad (7)$$

$$\theta_m = \frac{\pi}{8} m \quad (8)$$

$$w_m = \frac{\pi}{2(2)^{\frac{m}{N}}} \quad (9)$$

Descriptions:

$x, y$  : Pixel position in spatial domain

$w_0$  : Radial middle frequency

$\theta$  : Gabor rotation

$\sigma$  : Gaussian deviation standard of  $x$  and  $y$

## 3. NEURAL NETWORK

### 3.1 Multilayer Neural Networks

A multilayer neural network is a feed forward neural network with one or more hidden layers. Typically, the network consists of an input layer of source neurons, at least one middle layer or hidden layer of computational neurons, and an output layer of computational neurons [7].

The input layer accepts input signals from the outside world and redistributes these signals to all neurons in the hidden layer. The output layer accepts output signals, or in other words a stimulus pattern, from the hidden layer and establishes the output pattern of the entire network. Neurons in the hidden layer detect the features; the weights of the neurons represent the features hidden in the input patterns. These features are then used by the output layer in determining the output pattern [7]. A multilayer neural network with two hidden layers is shown in Figure 1.

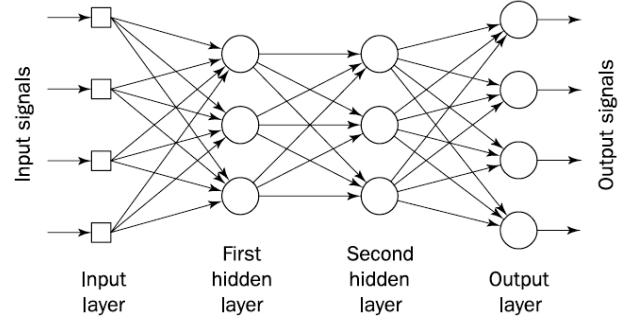


Figure 1. Multilayer Neural Network With 2 Hidden Layers [7]

### 3.2 Back-propagation Learning Algorithm

#### Step 1: Initialization

Set all the weights and threshold levels of the network to random numbers uniformly distributed inside a small range [5, 7]:

$$\left(-\frac{2.4}{F_i}, +\frac{2.4}{F_i}\right) \quad (10)$$

where  $F_i$  is the total number of inputs of neuron  $i$  in the network. The weight initialization is done on a neuron-by-neuron basis.

#### Step 2: Activation

Activate the back-propagation neural network by applying inputs  $x_1(p), x_2(p), \dots, x_n(p)$  and desired outputs  $y_{d,1}(p), y_{d,2}(p), \dots, y_{d,n}(p)$ .

- (a) Calculate the actual outputs of the neurons in the hidden layer:

$$y_j(p) = \text{sigmoid} \left[ \sum_{i=1}^n x_i(p) \times w_{ij}(p) - \theta_j \right] \quad (11)$$

where  $n$  is the number of inputs of neuron  $j$  in the hidden layer, and sigmoid is the sigmoid activation function.

- (b) Calculate the actual outputs of the neurons in the output layer:

$$y_k(p) = \text{sigmoid} \left[ \sum_{j=1}^m x_{jk}(p) \times w_{jk}(p) - \theta_k \right] \quad (12)$$

where  $m$  is the number of inputs of neuron  $k$  in the output layer.

#### Step 3: Weight training

Update the weights in the back-propagation network propagating backward the errors associated with output neurons.

- (a) Calculate the error gradient for the neurons in the output layer:

$$\delta_k(p) = y_k(p) \times [1 - y_k(p)] \times e_k(p) \quad (13)$$

$$e_k(p) = y_{d,k}(p) - y_k(p) \quad (14)$$

Calculate the weight corrections:

$$\Delta w_{jk}(p) = \alpha \times y_j(p) \times \delta_k(p) \quad (15)$$

Update the weights at the output neurons:

$$w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p) \quad (16)$$

- (b) Calculate the error gradient for the neurons in the hidden layer:

$$\delta_j(p) = y_j(p) \times [1 - y_j(p)] \times \sum_{k=1}^l \delta_k(p) \times w_{jk}(p) \quad (17)$$

Calculate the weight corrections:

$$\Delta w_{ij}(p) = \alpha \times x_i(p) \times \delta_j(p) \quad (18)$$

Update the weights at the hidden neurons:

$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p) \quad (19)$$

#### Step 4: Iteration

Increase iteration  $p$  by one, go back to Step 2 and repeat the process until the selected error criterion is satisfied.

## 4. APPLICATION DESIGN

This application is designed to be able to receive a stream image input from a video camera / webcam device in real time. The design of this application is shown in Figure 2.

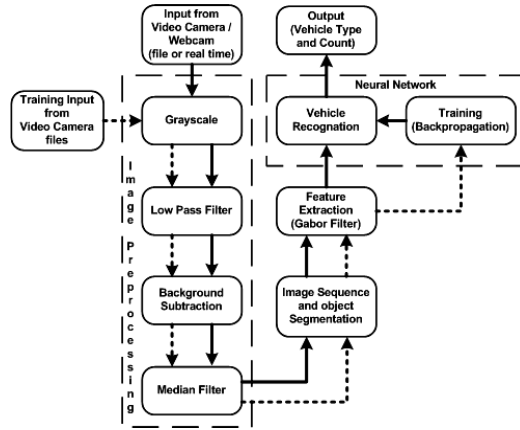


Figure 2. The Application Design.

### 4.1 Image Preprocessing Phase

In this phase, all the frames from video files or data streams will be processed using grayscale, low pass filter to eliminate noise, background subtraction to detect a moving object and the last process is the median filter to eliminate noise that may still remain. The example results of the preprocessing phase are shown in Figure 3.



Figure 3. Examples of Image Preprocessing Phase Result.

### 4.2 Segmentation Phase

The next phase is the segmentation phase. At this phase, each image frame in the video file (data stream) will be checked first on the most left, most right and middle vertically. If not all pixel in the middle position is black, while the most left and right area is black, it can be assumed that the frame in the checked area contained a complete car object (not truncated). Furthermore, in the frame containing the complete car object will be measured to ensure that large object that has been detected is not a passing pedestrian. Frame that pass from two stage of examination will be stored into the image file (\*.jpg) and will be used for neural network training process after going through feature extraction process using gabor filter or instantly recognizable when the module that used is Automobile Recognition module. Examples of frames that are successful and not successful passing the segmentation phase are shown in Figure 4 and 5.

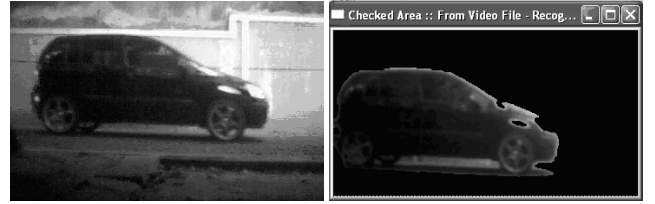


Figure 4. Example of Frame That is Successfully Passing The Segmentation Phase



Figure 5. Example of Frame That is Failed to Pass The Segmentation Phase

### 4.3 Feature Extraction Phase

In this phase, the result image from segmentation process will be convolved using gabor kernel for extracting the feature of image. The number of images generated from convolution process using gabor filter is equal to the number of gabor kernels that used to process an image. Gabor kernels are created based on orientation and frequency parameter that used. Suppose by using two types of orientation and five kinds of frequency will result ten kinds of gabor kernel. Image produced from this convolution process is called gabor response image. Example of this convolution process can be seen in Figure 6.

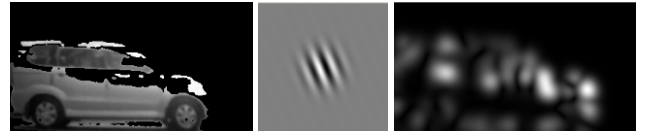


Figure 6. Image Before Gabor Filtering, The Gabor Kernel and Result Image After The Convolution Process.

In this experiment, we used five different parameters of frequency ( $n = 0, \dots, 4$ ), namely:  $W_n = 2^{-1}\pi, 2^{-1.5}\pi, 2^{-2}\pi, 2^{-2.5}\pi, 2^{-3}\pi$  and eight different parameters of orientation ( $m = 0, \dots, 7$ ), namely:  $\theta_m = 0^0$ ,

22.5°, 45°, 67.5°, 90°, 112.5°, 135°, 157.5°. This is adjusted by the research conducted by Moreno et.al. [6]. The variations of gabor kernel are shown in Figure 7.

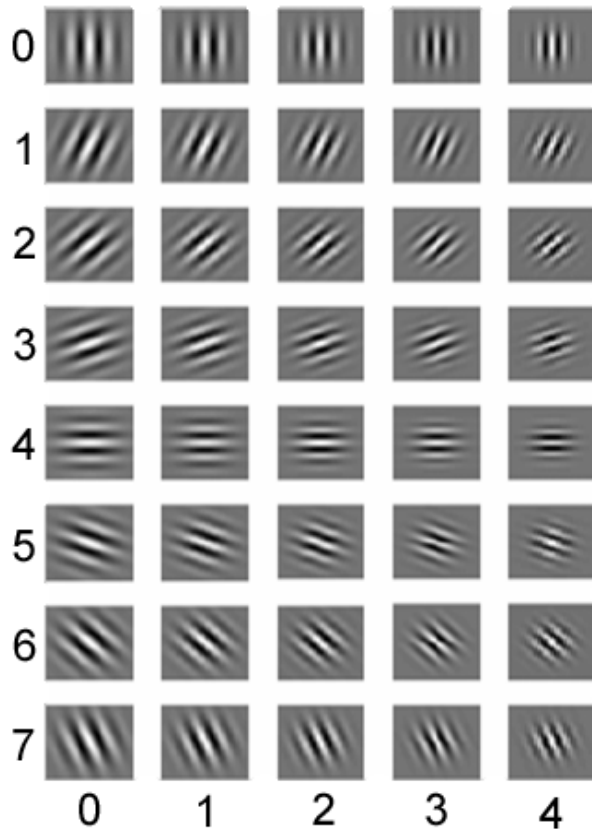


Figure 7. Forty Types of Gabor Kernel [6].

#### 4.4 Feed Forward Neural Network Design

Feed forward neural network architecture that is used is dynamic. Number of input neuron in input layer determined from the number of gabor response in feature extraction phase. After feature extraction process, this system will collect a sample point from result image that has been convolved by generated gabor kernel by using a grid. For example we used grid in the amount of ten, and from feature extraction phase we used four kinds of gabor kernel then the number of neurons in input layer is  $10 \times 10 \times 4 = 400$  neurons.

Hidden layer that is used is also dynamic, where we can change it freely according to the number of neuron in hidden layer and the number of hidden layer will be built.

Output layer consists of three neurons. All of three neurons represent the amount of automobile type that is processed. It is assumed the number of automobile-type in this world is less than or equals to seven kinds. Example formats of desire target: 000 = Sedan, 001 = City Car, 010 = MPV, etc.

#### 4.5 Application Output

Output from this application is a list consists of automobile type that recognized successfully and their number. In real time mode, the number of automobile type is the number of vehicles that have been passed until then, since the application started. Meanwhile, if the mode is video file, this application will calculate the number

and type of automobile that successfully identified. There are two kinds of displaying output: General List will display the recognition results based on the processing time, and Specific List will display the recognition results based on the automobile type. General and Specific List are shown in Figure 8.

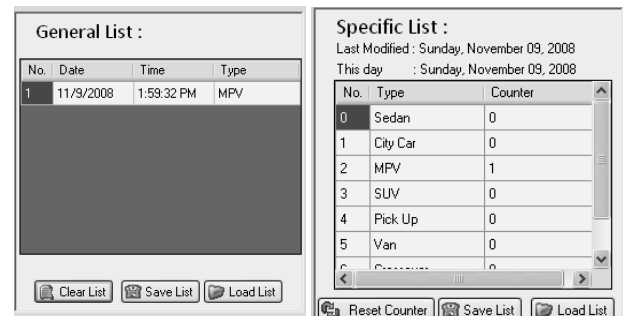


Figure 8. General List and Specific List examples.

### 5. TESTING AND ANALYSIS

The device specification used for testing is:

**Processor :** Pentium Core2Duo 2.4GHz

**RAM :** 2Gb DDR2

**Harddisk :** 250Gb

**O/S :** Windows XP SP2

**Compiler :** Ms. Visual Studio 2005 C++.Net

There are several kinds of experiments are performed, namely:

1. Test on the influence of the number of hidden layers and neurons in a hidden layer to the speed of training process until achieve convergence. The results of the tests are shown in Figure 9 and 10.

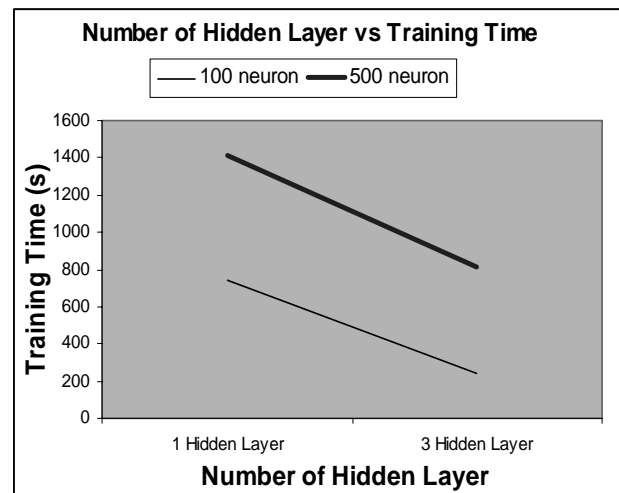
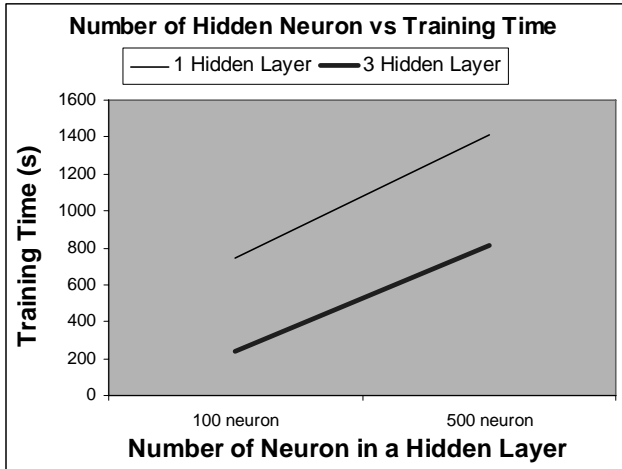


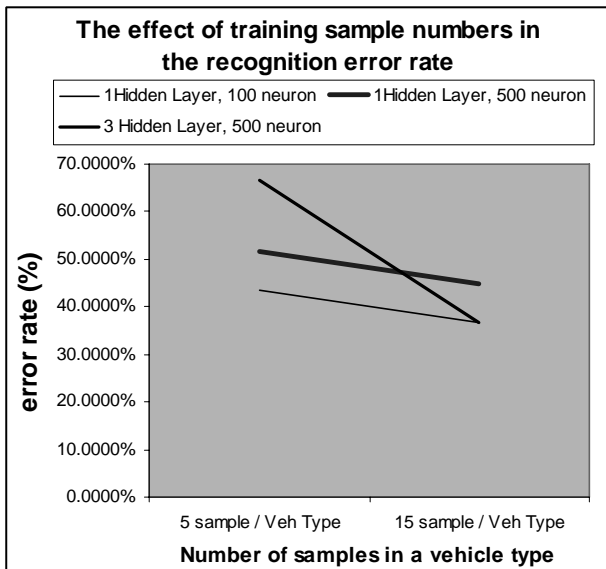
Figure 9. Number of Hidden Layer vs Training Time.



**Figure 10. Number of Hidden Layer Neurons vs Training Time.**

From the experiments, we can conclude that the addition of hidden layers will speed up training time, while increasing number of neurons in the hidden layer will slow down the training time.

- Tests for the error rate in recognition using data that has been trained to Neural Network. In this test, error rate that generated is 0% on the other words all the object has been recognized successfully. Gabor filter used frequency 0,1 and orientation 2,4.
- Tests for the error rate in recognition using data that has not been trained to Neural Network. Gabor filter used frequency 0,1 and orientation 2,4. The results of the test are shown in Figure 11.



**Figure 11. The Effect of Training Sample Numbers in The Recognition Error Rate**

From the tests, we can see that the quantity of training data for each automobile type can affect the accuracy of recognition for testing data that has not been trained yet. The percentage

of lowest error rate for this test is 36%. From further analysis, we found that there is a type of automobile that is often wrongly recognized, it is the Crossover, because the shape of Crossover is similar to the shapes of Sedan, MPV or City Car. This application often made a mistake in recognized this type of automobile. If this type (Crossover) is removed from training data and testing data, then error rate will reduced to 24%.

- The next test is to find the proper parameter setting of gabor filter. Various parameters of combination and frequency were attempted on the neural network with fixed configuration, one hidden layer and one hundred neurons in hidden layer. This back-propagation neural network setting is selected because it produces the lowest error rate (24%), with lowest number of hidden layer and neurons in hidden layer. From the test results, we found some parameter combinations of gabor filter which have the smallest recognition error rate, with percentage of error rate is 16% are:
  - Configuration 1: Orientation: 2; Frequency: 0, 2, 4, 6, 8
  - Configuration 2: Orientation: 3; Frequency: 0, 2, 4
  - Configuration 3: Orientation: 2, 3; Frequency: 0, 2, 4
- The fifth test was used to see if the color of the car affects the rate of recognition. The result of the test is shown in Table 1.

**Table 1. The Result of Automobile Color Test**

Movie ID	Num. of Samples	Vehicle Color	Error rate
1	12	Gray(2), Silver(3), Black(3), Brown(2), Red, Blue	41.6667%
2	12	Blue(5), Silver(3), Black(2), Gray(2)	41.6667%
3	12	Silver(4), Blue(3), Green, Black, Brown, Gray, White	41.6667%

The error rates from several car colors that have been tested are same, which is 41.66%. It can be concluded that the color of the car does not affect the recognition rate.

- The sixth test used to see if the lighting condition on the object that captured by a web cam can affect recognition rate or not. From the results of the tests known that if the lighting is too dark or too bright will cause the wrong silhouette object that will be processed as an input for the application. Examples of 'Too Dark' and 'Too Bright' object silhouette are shown in Figure 12 and 13.



**Figure 12. An Object and Its 'Too Dark' Object Silhouette**



**Figure 13. An Object and Its 'Too Bright' Object Silhouette**

## 6. CONCLUSION

In general, an application that is made in this study has achieved its purpose. By using gabor filters with orientation: 3 and the frequency: 0, 2, 4, and configuration of back-propagation neural network with one hidden layer and one hundred neurons in a hidden layer, we get the lowest error rate: 16%. This application is vulnerable to the effects of lighting when the lighting is too bright or too dark. This application is also experiencing difficulties in recognizing the shape of the hybrid-type automobile, such as crossover that is a mix of sedan and MPV or City Car.

## 7. ACKNOWLEDGMENTS

Our thanks to Mr. Resmana Lim and Mr. Thiang, who has helped us to get a better understanding of the research that they have done and also provide any important advice for the research we've done.

## 8. REFERENCES

- [1] Burdick, H. E. 1997. Digital imaging: Theory and Applications. London: McGraw-Hill.
- [2] Castleman, Kenneth R. 1996. Digital image processing. New Jersey: Prentice-Hall.
- [3] Fadlisyah. 2007. Computer Vision dan Pengolahan Citra. Yogyakarta: Andi.
- [4] Gonzales, Rafael C., and Woods, Richard E. 2002. Digital image processing 2<sup>nd</sup> edn. New Jersey: Prentice Hall.
- [5] Haykin, S. 1999. Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall, Englewood Cliffs, NJ.
- [6] Moreno, Plinio., Bernardino, Alexandre., and Santos-Victor, Jose. 2005. Gabor Parameter Selection for Local Feature Detection. IBPRIA - 2nd Iberian Conference on Pattern Recognition and Image Analysis, Estoril, Portugal, June 7-9, 2005.
- [7] Negnevitsky, Michael. 2005. Artificial intelligence: a guide to intelligent systems 2<sup>nd</sup> edn. Pearson Education Limited. Addison-Wesley.
- [8] Pratt, William K. 1991. Digital image processing 2<sup>nd</sup> edn. New York: John Wiley & Sons, Inc.
- [9] Thiang, Guntoro, Andre Teguh., and Lim, Resmana. 2001. Type of Vehicle Recognition using Gabor Filter Representation and Template Matching Method. Proceeding, Seminar of Intelligent Technology and Its Applications (SITIA 2001).
- [10] Tou, Jing Yi., Tay, Yong Haur., and Lau, Phooi Yee. 2007. Gabor Filters and Grey Level Co-occurrence Matrices in Texture Classification. Malaysia: Computer Vision & Intelligent Systems (CVIS) Group, Faculty of Information & Communication Technology Universiti Tunku Abdul Rahman (UTAR).
- [11] Yap, W. H., Khalid, M., Yusof, R. 2007. Face Verification With Gabor Representation And Support Vector Machines. IEEE Proc. of the First Asia International Conference on Modeling & Simulation.