International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 06, June-2019

# Number Plate Detection and Recognition for Vehicle in Toll Gate by using GFNN

Neelraj Elangovan[1]
Department of EECE
St Joseph University in Tanzania
Dar es salaam, Tanzania

Vasanth Arumugam[2]
Department of EECE
St Joseph University in Tanzania
Dar es salaam,Tanzania

Ramalingam shanmugam[3]
Research Scholar, ACTECH
Karaikudi

***Abstract*:- In this study, a vehicle license plate is recognized using artificial neural networks. Recognition of a vehicle license plate is usually important for many security and control systems. Artificial neural networks (ANN) are explained, and used image processing algoritms towards recognition are given with their result in this paper.**

## **I.** INTRODUCTION

An artificial neural netwok is a parallel distributed information processing system. It stores the samples with distributed coding, thus forming a trainable nonlinear system. The main idea of neural network approach resembles the human brain functioning. Given the inputs and desired outputs, it is also self adaptive to the environment so as to respond different inputs rationally [1].

Traffic license plate recognition is usually important for many security and traffic systems. In these kinds of systems, it is wanted to understand the number of license plate of a vehicle on a captured image. Image processing has also a great duty on these kind systems.

The aim of this paper is to give information about recognition of a license plate using neural network towards the real-time implementation. Artificial neural networks are known good at online giving answer towards the given unknown input by the way of training previously. This feature of ANN makes it very useful in these kinds of recognition systems.

## **II.** IMAGE PROCESSING

The related image processing algorithms are given in this section towards recognition process. In prepared software, the image is processed off-line, but the real time implementation of the system is explained.

### NOISE REDUCTION

Noise reduction, in other words, filtering or smoothing is one of the most important processes in image processing systems. Images are often corrupted by positive and negative impulses stemming from decoding errors or noisy channels. An image can be degradated because of the undesirable effects due to illumination and other effects. Median filter is widely used for smoothing and restoring images corrupted by noise. It is a non-linear process useful especially in reducing impulsive or salt- and-pepper noise. In a median filter, a window slides along the image, and the median intensity of the pixels within the window becomes the output intensity of the pixel being processed. Different from linear filters such as the mean filter, median filter has attractive properties for supressing impulse noise while preserving edges [2][3][4][5]. It should be noted that in real time applications, it is necessary to use a noise reduction algorithm due to the undesired effects especially from system hardware [6].

### *THRESHOLDING*

Many applications require the differentiation of the objects or some characters from background in the image data. Thresholding is easily appliable method for this purpose. Thresholding is choosing a threshold value and assigning 0 to the pixels with values smaller or equal to T and 1 to the ones with greater values than T. We are interested in dark characters on a light background, the parameter T, called the brightness threshold, is choosen and applied to the image f (x,y) as follows:

If $f(x,y) \geq T$ f (x,y )= Background
Else f (x,y) = Character

Thresholding is a technique widely used in image segmentation [7][8].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 06, June-2019

## IMAGE SEGMENTATION

Thresholding is the main process towards image segmentation. The thresholded image has only the characters that belong to the license plates of a vehicle including numbers and alphabetic chararecters. To recognize these characters, an ANN is designed to give an answer for only one character. So, it is important to segment all characters in a vehicle license plate image. The segmentation process is implemented in prepared software using the information that the characters are different black pixels, and the character number is known for a country.

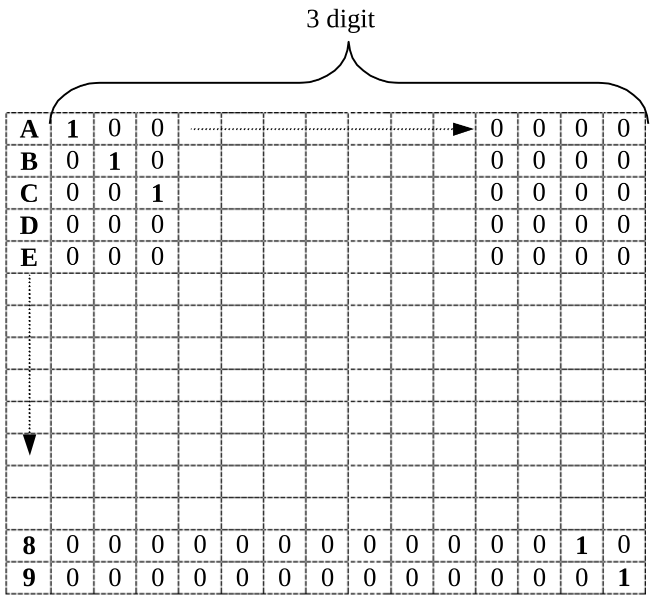
## IMAGE CODING

In an image processing application, it is necessary to code the images according to aim. There are many coding techniques being used in coding, such as Chain code, run code and crack code. In addition to these, sometimes some feature extraction techniques are used, such as moment invariants or Fourier Descriptors. According to the simplicity of the process, it is possible to code images in binary [9]. In this study, the characters are coded in binary to make them ready as an input to ANN. A sample binary coded character image is given below.

```
1 1 1 1 1 1 1 1 1 1 1
1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 0 1
1 1 1 1 1 1 1 0 0 0 1
1 1 1 1 1 1 0 0 0 1 1
1 1 1 1 1 1 0 0 1 1 1
1 1 1 1 1 0 0 1 1 1 1
1 1 1 1 1 0 0 1 1 1 1
1 1 1 1 0 0 1 1 1 1 1
1 1 1 1 0 0 1 1 1 1 1
1 1 1 0 0 1 1 1 1 1 1
1 1 0 0 0 1 1 1 1 1 1
1 1 0 0 1 1 1 1 1 1 1
1 0 0 0 1 1 1 1 1 1 1
1 0 0 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1
```

Figure:1 Sample image code for number 7 (input of ANN)

As seen in figure 1, each character is coded 11x19 pixels. These 209 bits are given as input vector set of ANN sequentially for each character. The binary technique is used to code the output of ANN expressing each character (figure 2). It is a known fact that binary codes are strong through the noise.



| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **A** | **1** | 0 | 0 | | | | | | | | 0 | 0 | 0 | 0 |
| **B** | 0 | **1** | 0 | | | | | | | | 0 | 0 | 0 | 0 |
| **C** | 0 | 0 | **1** | | | | | | | | 0 | 0 | 0 | 0 |
| **D** | 0 | 0 | 0 | | | | | | | | 0 | 0 | 0 | 0 |
| **E** | 0 | 0 | 0 | | | | | | | | 0 | 0 | 0 | 0 |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | |
| **8** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 |
| **9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** |

Figure:2 The representation of characters in binary (output of ANN)

## III. ARTIFICIAL NEURAL NETWORKS

Artificial neural networks are statistical models of real world systems which are built by tuning a set of parameters. These parameters, known as weights, describe a model which forms a mapping from a set of given values known as inputs to an associated set of

the correct values –training- is vehiclered out by passing a set of examples of input-output pairs through the model and adjusting the weights in order to minimise the error between the answer the network gives and the desired output. Once the weights have been set, the model is able to produce answers for input values which were not included in the training data. The models do not refer to the training data after they have been trained; in this sense they are a functional summary of the training data [1,10]. These mentioned processes for backpropagation algorithm are given step by step below.

## THE BACK PROPAGATION (BP) ALGORITHM

The back propagation is a widely used algorithm, and it can map non-linear processes. It is a feedforward network with the one or more hidden layers. The elementary architecture of the backpropagation network has three layers. There are no constraints about the number of hidden layers. Backpropagation is a systematic method for training multilayer artificial neural networks. It has a mathematical foundation that is strong if not highly practical. In figure 3, a sample multi-layer feedforward net structure is given, and all parameters are given according to this figure below.

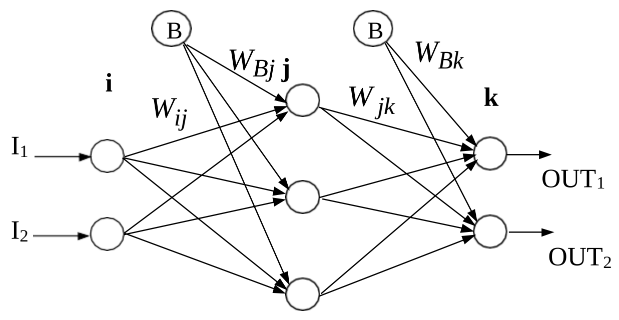


Figure 3. A sample multi-layer feedforward net structure

**_FORWARD PASS PROCESS IN BP ALGORITHM_**

NET values can be obtained by multiplying inputs and related weights. To calculate NET values out of perceptrons in hidden layer; the formula (1 and 2) given below is used.

$$NET_j = \sum I_i W_{ø-} + W_{Bj}^{NEW} \qquad (1)$$

$$OUT_j = \frac{1}{1+e^{-NET_j}} \qquad (2)$$

To calculate the OUT values, NET values are applied to an activation function. Similarly, the outs of hidden units are considered as inputs of the next hidden layer units or output layer units. $W_{Bj}^{NEW}$ and $W_{Bk}^{NEW}$ values present biases. OUT and NET are calculated by using formulas (3) and (4).

$$NET_k = \sum I_k W_{jk} + W_{Bk}^{NEW} \qquad (3)$$

$$OUT = \sum \frac{1}{1+e^{-NET_k}} \qquad (4)$$

**_ERROR PROPAGATION IN BP ALGORITHM_**

The outputs of neural network model are obtained from the output layer units. The difference between target values and actual values are considered as system error. The obtained error values are propagated back to connection weights. This process is applied using the following equations;

Firstly, from output layer to last hidden layer;

$$\delta_k = f'(NET_k)(TARGET_k - OUT_k) \qquad (5)$$

$$\delta_k = OUT_k(1 - OUT_k)(TARGET_k - OUT_k) \qquad (6)$$

$$\Delta W_{kj}(n+1) = \eta\delta_k OUT_k + \alpha.[\Delta W_{kj}(n)] \qquad (7)$$

$$W_{kj}^{NEW} = W_{kj}^{OLD} + \Delta W_{kj}(n+1) \qquad (8)$$

Where $TARGET_k$ presents desired output value. $\eta$ is learning rate, $\alpha$ is momentum coefficient, $f(NET_k)$ presents activation function, N presents iteration number and $\Delta W$ is change of related weight. This term is added to old weight of the related connection to obtain a new one.

Secondly, from hidden layer to input layer;

$$\delta_j = f'(NET_j). \sum_k \delta_k W_{kj} \qquad (10)$$

$$\delta_j = OUT_j(1 - OUT_j)\sum\delta_k W_{kj} \qquad (11)$$

$$\Delta W_{ji}(n+1) = \eta.\delta_j.OUT_j + \alpha.[\Delta W_{ji}(n)] \qquad (12)$$

$$W_{ji}^{NEW} = W_{ji}^{OLD} + \Delta W_{ji}(n+1) \qquad (13)$$

The bias effects the activation function in order to force the learning process, therefore the speed of learning process increases. Biases are recomputed as following, for the biases of output layer, where the letters and symbols have similiar meanings;

$$\Delta W_{BK}(n+1) = \eta\delta_k + \alpha.[\Delta W_{BK}(n)] \qquad (14)$$

$$W_{Bj}^{NEW} = W_{Bj}^{OLD} + \Delta W_{Bk}(n+1) \qquad (15)$$

and for the biases of hidden layer**;**

$$\Delta W_{Bj}(n+1) = \eta.\delta_j + \alpha.[\Delta W_{Bj}.(n)] \qquad (16)$$

$$W_{Bj}^{NEW} = W_{Bj}^{OLD} + \Delta W_{Bj}(n+1) \qquad (17)$$

The training of the neural network model, as may be understood from the previous process is vehicleried out in two steps. The first step is called forward pass, that is composed of calculation for NET and OUT values. The second step is called backward pass that is composed of error propagation throughout connection weights. The iterative process is repeated until a satisfactory learning level is achieved i.e. the differences between TARGET and OUT are minimized [11][12][13].

## IV. EXPERIMENTAL STUDY

In this section, the experimental studies are axplained towards training ANN. Training is one of the important processes that affect the system success directly in ANN based systems. The training and recognition processes are given below including its software architecture. ANN and image processing processes are coded in Delphi Programming Language.

### *TRAINING NEURAL NETWORK*

In off-line training of BP neural network, input and output vector sets are prepared for 36 character sets as mentioned in image segmentation section. The deformed character images are also used to prepare learning set to obtain sensitivity through the undesired effects. The designed ANN includes 209 nodes in the input layer, and 36 nodes in the output layer. The number of nodes in hidden layer is 36, and choosen experimentaly. Learning rate and the momentum rate are experimentally choosen as 0.2 and 0.8, respectively. Error at the end of the learning is 0.000763. The error is computed using the equation (19) known as average squared error [1]. Here, N denotes the total number of samples in training set, and the set C includes all the neurons in the output layer of the network.

$$e_k(n) = d_k(n) - y_k(n) \qquad (18)$$

$$\varepsilon_{av} = \frac{1}{2N} \sum_{n=1}^{N} \sum_{k \in C} e_k^2(n) \qquad (19)$$

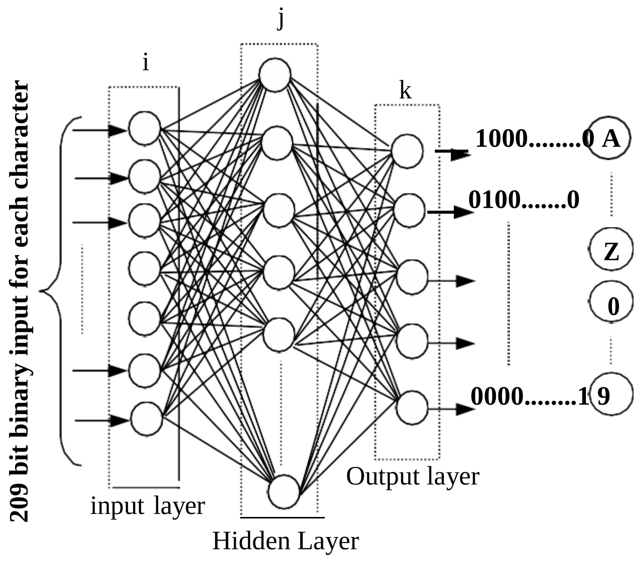


Figure:4 The topology of the designed neural network

The training process has been completed approximately in 20000 iterations. When off-line training is completed, a neural network is designed using the obtained weights as seen in figure 4.

## V. ON REAL-TIME IMPLEMENTATION OF THE SYSTEM

In this section, it is aimed to give information towards the real-time implementation process. This study is not real-time implemented, but also gives the necessary algorithms, and hardware implementations. The hardware of the system is explained and its block diagram is given in figure 5.

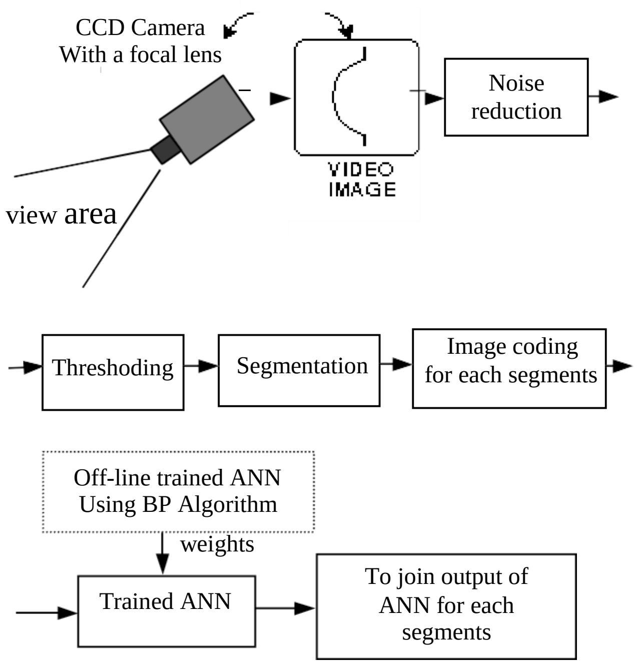

CCD Camera
With a focal lens
view area
VIDEO IMAGE
Noise reduction
Thresholding
Segmentation
Image coding for each segments
Off-line trained ANN Using BP Algorithm
weights
Trained ANN
To join output of ANN for each segments

Figure:5 The block diagram of the system towards real-time implementation.

This system needs three major components, a CCD camera, a capture card and a computer to use as a system laboratuary. The camera should have a focal lens to zoom the license plate on a vehicle, and also enough resolution. As a capture card, a frame grabber card, classical capture card or Tv-Capture card may be used in this study. It needs more software ability to control the Tv-Capture vehicled online using its drivers (DLL files). The aim of these capture cards are to capture image in a time and digitalize the image towards image processing [14,15].

## VI. RESULTS AND DISCUSSIONS

Training process is implemented using Pentium-III 750 Mhz. computer. The threshold unit is used in ANN to decide the result of character, such as a character 'V' may be found % 80 'V', and % 45 'Y'. So that, a threshold value should be given to decide object type.

The trained network is tested using different license plates, and % 100 correct recognition is observed for each

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 06, June-2019

character. The sample number in learning set makes system more reliable. Prepared simulation program seperates given license plate to characters, and it gives the result of test as keyboard characters. The sample recognized license plate is given in figure 6. The software gets given image from the user interface and segments it to obtain 8 character belonging the plate. Then, test process has given the correct license number. It is taken care of that the deformed characters are also thought to be recognized by the way of using them in training set..

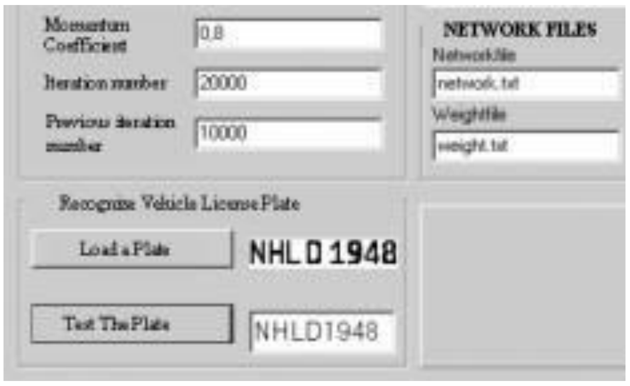

Momentum Coefficient: 0.8
Iteration number: 20000
Previous iteration number: 10000
NETWORK FILES
Networkfile: network.txt
Weightfile: weight.txt
Recognize Vehicle License Plate
NHLD1948

Figure: 6 A sample license image and its test result (a view from prepared software)

## VII. CONCLUSIONS

In this study, the success of vehicle license recognition system using neural networks is observed. If the system is wanted to be implemented in a real time application, the captured image will include some environmental factors. The license plate may be seperated by some basic image processing algorithms, such as template matching and the white pixels around the characters, etc., and then, the real time vehicle license recognition system may be obtained by using the software developed in this study.

## REFERENCES

[1] S. Haykin, Neural Networks, Macmillian College Publishing Company, Inc., 1994.
[2] J. S. Lim, Two-Dimensional and Image Processing, Prentice-Hall, 1990.
[3] X Yang, P. S. Toh, Adaptive Fuzzy Multilevel Median Filter, IEEE Transaction on Image Processing, Vol.4, No.5, pp.680-682, May 1995.
[4] H. Hwang, R. A. Haddad, Adaptive Median Filters: New Algorithm and Results, Transactions on Image Processing, Vol. 4, No. 4, pp.499-505, April 1995
[5] A. Rosenfeld, Digital Picture Processing, Academic Press Inc., 1982.
[6] R. Köker, C. Öz, $ )HULNR÷OX 2EMHFW 5HFRJQLWLRQ Based on Moment Invariants Using Neural Network, 3rd Intern. Symp. on Intelligent Manufacturing Systems, IMS 2001, Sakarya, Turkey, accepted paper.
[7] U. M. Erdem, 2D Object Recognition In Manufacturing Environment Using Implicit Polynomials DQG $OJHEUDLF ,QYDULDQWV 0DVWHU 7KHVLV %R÷D]LoL University, 1997.
[8] K. S. Fu, J. K. Mui, A Survey On Image Segmentation, Pattern Recognition, Vol.13, pp.3-16, Pergamon Press, 1981.
[9] G. J. Awcock, R. Thomas, Applied Image Processing, McGraw Hill, Inc., 1996.
[10] K. Swingler, Applying Neural Networks A Practical Guide, Academic Press, 1996.
> @ 7 dDNDU ø dLO $ .XUW )OH[LEOH 0DQXIDFWXULQJ System Design and Determination of Priority Rules by Using Artificial Neural Networks, 8th International Machine Design and Production Conference, September 9-11, 1998, Ankara TURKEY.
[12] P. D. Wasserman, Neural Computing, Van Nostrand Reinhold, New York, 1989.
[13] J. Freeman, D. Sakapura, Neural Networks: Algorithms, Applications and Programming Techniques, Reading, Mass., Addison Wesley, 1991.
> @ 5 .|NHU & g] $ )HULNR÷OX 'HYHORSPHQW RI D Vision Based Object Classification System For an Industrial Robotic Manipulator, The 8th IEEE Intern. Conf. on Electronics, Circuits and Systems, Malta, 2001, accepted paper.
[15] P.A. Laplante, A.D. Stoyenko, Real Time Imaging, IEEE Press, 1996, p.231.