DEVELOPMENT OF VISION AUTONOMOUS GUIDED VEHICLE BEHAVIOUR USING NEURAL NETWORK

HUSNUL ‘ASYIYYAH BT MOHAMAD @ AWANG

BACHELOR OF MANUFACTURING ENGINEERING

UNIVERSITI MALAYSIA PAHANG

2012

UNIVERSITI MALAYSIA PAHANG

BORANG PENGESAHAN STATUS TESIS♦

JUDUL: **DEVELOPMENT OF VISION AUTONOMOUS GUIDED VEHICLE BEHAVIOUR USING NEURAL NETWORK**

SESI PENGAJIAN: **2011/2012**

Saya **HUSNUL 'ASYIYYAH BT MOHAMAD (900502-11-5064)**

mengaku membenarkan tesis (Sarjana Muda/~~Sarjana~~ /~~Doktor Falsafah~~)* ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Tesis adalah hakmilik Universiti Malaysia Pahang (UMP).
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. **Sila tandakan (√)

☐	**SULIT**	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)
☐	**TERHAD**	(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)
√	**TIDAK TERHAD**	

Disahkan oleh:

(TANDATANGAN PENULIS)

(TANDATANGAN PENYELIA)

Alamat Tetap:

134-F JALAN SEEMERAK,
22300 KUALA BESUT,
TERENGGANU

PROF. DR. ZAHARI TAHA

Tarikh: _____

Tarikh: _____

CATATAN:
- * Potong yang tidak berkenaan.
- ** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh tesis ini perlu dikelaskan sebagai atau TERHAD.
- ♦ Tesis dimaksudkan sebagai tesis bagi Ijazah doktor Falsafah dan Sarjana secara Penyelidikan, atau disertasi bagi pengajian secara kerja kursus dan penyelidikan, atau Laporan Projek Sarjana Muda (PSM).

DEVELOPMENT OF VISION AUTONOMOUS GUIDED VEHICLE BEHAVIOUR USING NEURAL NETWORK

HUSNUL ‘ASYIYYAH BT MOHAMAD @ AWANG

Report submitted in partial fulfilment of the requirements

for the award of the degree of

Bachelor of Manufacturing Engineering

Faculty of Manufacturing Engineering

UNIVERSITI MALAYSIA PAHANG

JUNE 2012

SUPERVISOR'S DECLARATION

I hereby declare that I have checked this project and in my opinion, this project is adequate in terms of scope and quality for the award of the degree of Manufacturing Engineering or Bachelor of Manufactuirng Engineering.

Signature : _____

Name of Supervisor : PROF. DR. ZAHARI TAHA

Position :

Date :

STUDENT'S DECLARATION

I hereby declare that the work in this project entitled "Development of Vision Autonomous Guided Vehicle Behaviour Using Neural Network" is the results of my own research except as cited in the references. The project has not been accepted for any degree and is not concurrently submitted for award of other degree.

Signature : _____

Name : HUSNUL 'ASYIYYAH BT MOHAMAD @ AWANG

ID Number : FA08016

Date :

To my beloved parents

Mr Mohamad @ Awang bin Muda

Madam Zahrah @ Jamilah bt Haji Ahmad

and

my fellow friends

ACKNOWLEDGEMENTS

Alhamdulillah, thanks to Allah s.w.t for giving me strength to finish this report and projecy within time given without any unsolved difficulties.

I am grateful and would like to express my sincere gratitude to my supervisor Prof. Dr. Zahari Taha for his invaluable guidance, continuous encouragement, advices, constant support, constructive criticisms and suggestion throughout this project in the progression and smoothness of the project.

I would also like to express my deepest appreciation to my beloved parents who always support me and motivate me to complete this final year project.

I am also very thankful to Faculty of Manufacturing Engineering. My sincere thanks go to those lecturers and staff who helped me in many ways and shared their precious knowledges and experiences with me. My sincere appreciation to all my course mate with their sharing and help during my difficulties completing my job for this project. I would like to thank to my friends in supporting me in finish up this project.

Finally to individuals who has involved neither directly nor indirectly in succession of this project and during my thesis writing. Thank you very much.

ABSTRACT

This project is motivated by an interest in promoting the use of artificial neural network in manufacturing. Automated guided vehicle (AGV) is used in advanced manufacturing system that can help to reduce cost and increase efficiency. The application of neural network in the AGV is to help in increasing the AGVs performance and efficiency. The objectives of this project are to develop a line recognition algorithm for automated guided vehicle and to understand two types of neural networks that can be use in manufacturing. The types of guidelines used in this project are straight guideline, turn right guideline, turn left guideline and stop guideline. The line recognition algorithm involved the pre-processing images of the guideline captured by a camera and extracts the feature of the images by using first order statistics to calculate the values of mean, variance, skewness and kurtosis and train the image recognition by using neural networks. Neural network process involved setup the two types of neural network, trained and tested the network and compared the result. There are two types of neural network that used in this project namely, Feedforward Backpropagation and Radial Basis. In Feedforward Backpropagation Network the parameter involves are transfer function and number of neurons. Mean Squared Error (MSE) is used as performance function. Radial Basis Network with spread constant one give significantly better performance compared to Feedforward Backpropagation Network. It produced much lower error compared to Feedforward Backpropagation Network. This project used MATLAB software which able to perform image processing tasks, train and simulate neural networks.

ABSTRAK

Projek ini adalah didorong oleh kepentingan dalam mempromosikan penggunaan rangkaian neural buatan dalam pembuatan. Kenderaan berpandu automatik (AGV) digunakan dalam sistem pembuatan termaju yang boleh membantu mengurangkan kos dan meningkatkan kecekapan sistem. Penggunaan rangkaian neural dalam AGV adalah untuk membantu dalam meningkatkan prestasi dan kecekapan AGV. Objektif projek ini adlah untuk membangunkan satu algoritma pengecaman garisan untuk kenderaan berpandu automatik dan memahami dua jenis rangkaian neural yang boleh digunakan dalam sektor pembuatan. Jenis-jenis garis panduan yang digukan dalam projek ini adalah garis panduan lurus, garis panduan kanan, garis panduan kiri dan garis panduan berhenti. Algoritma pengecaman garis yang terlibat ialah pemprosesan imej garis panduan yang ditangkap oleh kamera, pengekstrakan ciri imej dengan menggunakan statistik tertib pertama untuk mengira min, perbezaan, kecondongan dan kurtosis dan melatih pengecaman imej dengan menggunakan rangkaian neural. Terdapat dua jenis rangkaian neural yang digunakan dalam projek ini iaitu Feedforward Backpropagation dan Radial Basis. Parameter yang terlibat dalam rangkaian Feedforward Backpropagation ialah bilangan neuron dan fungsi pindah. Mean Squared Error (MSE) digunakan sebagai fungsi prestasi. Fungsi latihan yang digunakan adalah trainlm. Rangkaian Radial Basis dengan pemalar penyebar satu memberikan prestasi yang jauh lebih baik berbanding dengan rangkaian Feedforward Backpropagation. Ia menghasilkan ralat yang lebih rendah berbanding dengan rangkaian Feddforward Backpropagation. Projek ini menggunakan perisian MATLAB yang mampu melaksanakan tugas-tugas pemprosesan imej, melatih dan mensimulasikan rangkaian neural.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

μ_n	Moments of the gray level histogram
μ	Mean
P_k	Normalized histogram
σ^2	Variance
γ^3	Skewness
γ^4	Kurtosis
δ_j	Error between output and input in backpropagation network
δ_k	Error between hidden layer and output layer in backpropagation network
y_{net}	Output of the Radial Basis Neural Network

LIST OF ABBREVIATIONS

AGV	Automated Guided Vehicle
ANN	Artificial Neural Network
FMS	Flexible Manufacturing Systems
JPEG	Joint Photographic Experts Group
MSE	Mean Squared Error
NN	Neural Network
RGB	Red Green Blue

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION OF STUDY

Automated Guided Vehicle (AGV) is a kind of intelligent mobile robot, which can move along the guideline. It can operate independently, which means it is able to perform their operations without human direction. In the development of AGV, there are two classification of AGV that are guiding with lines and without lines (Sulaiman Sabikan et al., 2010). AGV also can follow markers or wires in the floor or use laser or vision. The number of AGV use is increasing from year to year. The application of AGV has been expended and no longer restricted to industrial environments. AGVs are widely use in industrial field such as automotive, manufacturing and chemical. With the implementation of the AGV system, it will help to reduce costs and increase efficiency especially in advanced manufacturing system. Usually the implementation of AGV is in the Flexible Manufacturing Systems (FMS) in order to integrating machinery or manufacturing cells, which need material transfer. Generally, the AVG systems consist of a computer software and technology that are the brain behind AGV (Sulaiman Sabikan et al., 2010).

1.2 PROJECT BACKGROUND

An artificial neural network (ANN), usually called neural network (NN) is a data processing system consisting of a large number of simple and highly interconnected processing elements (artificial neurons) inspired by the structure of the cerebral cortex of the brain (Lefteri, H.T. and Robert, E.U., 1997). ANN is a type of artificial intelligence that attempts to imitate the way of human brain works (Sivanandam, S.N. et al., 2011). Basically, neural network deal with cognitive tasks such as learning, adaptation, generalization and optimization. Certainly, recognition, learning, decision making and action represent the principal navigation problems (Janglova, D., 2004). Neural networks perform two major functions that are learning and recall. Learning is the process of adapting the connection weights in an artificial neural network to produce the desired output vector in response to a stimulus vector presented to the input buffer. Recall is the process of accepting an input stimulus and producing an output response in accordance with the network weight structure (Lefteri, H.T. and Robert, E.U., 1997). Learning rules enable the network to gain knowledge from available data and apply that knowledge to assist a manager in making key decisions. Neural networks also able to compute any computational function. It also can be defined as parameterized computational nonlinear algorithms for data, signal and image processing (Sivanandam, S.N. et al., 2011).

Table 1.1 shows the comparison between artificial and biological neural network. Biological neural network or nerve cell consists of cell body, dendrite, soma and axon while artificial neural network consists of neurons, weights or interconnection, net input and output (Sivanandam, S.N. et al., 2011).

Table 1.1: Comparison between artificial neural network and biological neural network

Characteristics	Artificial Neural Network	Biological Neural Network
Speed	Faster in processing information.	Slow in processing information.
Processing	Sequential mode operations.	Massively parallel operations.
Size and complexity	Not involve as much computational neurons. Hence it is difficult to perform complex pattern recognition.	Have large number of computing elements, and the computing is not restricted to within neurons. The size and complexity of connections give the brain power of performing complex pattern recognition tasks.
Storage	In a computer, the information is stored in the memory, which is addressed by its location. Any new information in the same location destroys the old information. Hence here it is strictly replaceable.	Store information in the strengths of the interconnections. Information in the brain is adaptable, because new information is added by adjusting the interconnection strengths, without destroying the old information.
Fault tolerance	Artificial nets are inherently not fault tolerant, since the information corrupted in the memory cannot be retrieved.	Exhibit fault tolerance since the information is distributed in the connections throughout the network.
Control mechanism	There is a control unit, which monitors all the activities of computing.	There is no central control for processing information in the brain. No specific control mechanism external to the computing task.

Source: Sivanandam, S.N. et al. (2011)

Table 1.1 shows that artificial neural network are faster in processing information compare to the biological neural network. Processing for artificial neural network is operating in a sequential mode while for biological neural network can perform massively parallel operations. The size and complexity of connection in biological neural network gives the brain the power of performing complex pattern recognition tasks, which cannot be realized on artificial neural network. For storage,

artificial neural network stored information in the memory, which is addressed by its location where new information in the same location will destroys the old information, while biological neural network store information in the strengths of the interconnection where new information is added by adjusting the interconnection strengths without destroying the old information. There is a control unit, which monitors all the activities of computing for artificial neural network while there is no central control for processing information in the brain.

Inspired by biological neural networks, artificial neural networks are massively parallel computing systems consisting of an extremely large number of simple processors with many interconnections. Device based on biological neural networks will posses some of these desirable characteristics such as learning ability, adaptivity, fault tolerance, low energy consumption, generalization ability, massive parallelism and distributed representation and computation. Hence it is reasonable to expect a rapid increase in our understanding of artificial neural networks leading to improved network paradigms and a host of application opportunities. Neural network have remarkable ability to derive meaning from complicated or imprecise data, to extract patterns and detect trends that are too complex to be noticed. A trained neural network can be thought of as an expert in the category of information it has been given to analyze. The basic building blocks of the artificial neural network are network architecture, setting the weights and activation function (Sivanandam, S.N. et al., 2011). Advantages of neural networks are good pattern recognition technique, the system developed through learning rather than programming that consume more time for analyst, flexible in changing environment, can build informative models and can operate well with modest computer hardware (Symeonidis, K., 200).

1.3 PROBLEM STATEMENT

In order to increase AGVs efficiency, the line guideline must be detected and recognized by vision sensor accurately (Sulaiman Sabikan et al., 2010). Neural network is employ in its controller algorithm and vision system as ranging sensor. Therefore, there is need to study the performance of AGV recognized the line by using neural network behaviour algorithm. It is to determine the most suitable type of neural network that can allow the most efficient line recognition algorithm.

1.4 PROJECT OBJECTIVES

The objectives of this project are:

(i) To develop a line recognition algorithm for automated guided vehicle (AGV).
(ii) To understand two types of neural networks that can be use in manufacturing.

1.5 PROJECT SCOPES

Line recognition algorithm for vision AGV is important because it can be a main reference throughout navigation. Meanwhile, guideline is needed as important characteristic for the line recognition. This guideline will be placed on the flat floor surface and it is white colour. The types of guidelines used in this project are:

(i) Straight guideline
(ii) Turn right guideline
(iii) Turn left guideline
(iv) Stop guideline

This project used supervised training where it is a process of providing the network with a series of sample inputs and comparing the output with the expected

responses. The training continues until the network is able to provide the expected response (Sivanandam, S.N. et al., 2011). Development of vision AGV behaviour using neural network for this research involves in comparing two types of neural networks which are:

(i) Feedforward backpropagation
(ii) Radial basis

The purpose of comparing these two types of neural network is to find the best type for line recognition besides learn the recognition analysis using neural networks. This is important to improve the AGVs capabilities and increase it efficiency. This project use camera based vision for the vision sensor. Camera based vision system is useful in order to recognize the line guideline and allow line recognition algorithm.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

The purpose of this chapter is to provide a review of past research related to this project. Some of the contents of the research that related to this project are Vision-Based Automated Guided Vehicle (V-AGV), statistical feature extraction and neural networks. The idea of this project is developed from the related article and journal.

2.2 VISION-BASED AUTOMATED GUIDED VEHICLE

A navigation control system for a Vision-Based Automated Guided Vehicle (V-AGV) by detecting and recognizing line tracking can be done by using Universal Serial Bus (USB) camera (Sabikan, S. et al., 2010). The main components used are laptop and low cost USB camera. The vision-based navigation system structure is composed of guideline detection, sign detection and obstacle detection. Through USB camera three algorithms that are guideline detection, sign detection and obstacle detection gain some predictive of knowledge from environment. Line detection algorithm consists of seven types of guidelines that are straight, crossing, turn left, turn right, straight and left, straight and right, and lastly is junction guideline. Besides that, this line detection

algorithm is divided into four steps, they are system initialization, image pre-processing, measuring the width of the guideline and recognition and classification of guideline. Sign symbols have been placed on the floor for sign detection algorithm that is used as a direction in the V-AVG navigation (Sabikan, S. et al., 2010).

The experimental results from above research have shown that V-AVG navigation control system have been successfully implemented on the real guideline system. A low cost of USB camera can be use for vision based line recognition and detection algorithm. The USB camera has performed well in executing the proposed algorithm. This control system do not need the destination target to be programmed, it depends on the guideline.

2.3 FEATURE EXTRACTION

Feature extraction is the process of defining a set of features or image characteristics which will most efficiently or meaningfully represent the information that is important for analysis and classification. Much of the information in the data set may be of little value for discrimination. Indeed, pattern recognition using the original measurements is frequently inefficient and may even obscure interpretation (Nurhayati, O.D. et al., 2011). Feature extraction is a special form of dimensionality reduction for pattern recognition and image processing. It can be used in image processing which involves the use of algorithms to detect and isolate various desired portions or shapes (features) from an image or video.

Statistical feature extraction can be used to calculate the value of mean, variance, skewness and kurtosis from first order statistics. First order statistics or moments of the gray level histogram are the nth moment of the (normalized) gray level histogram is given by:

$$\mu_n = \sum_{i=1}^{L} (k_i - mean)^n \, p(k_i) \tag{2.1}$$

where

k_i = gray value of the ith pixel

mean = mean gray value of the pixel set

L = the number of distinct gray levels

$p(k_i)$ = normalized histogram (probability density function of the pixel set)

2.3.1 Mean

Mean is the average of the values in the set of data, obtained by summing the values and dividing by the number of values. Mean also can be defined as a measure of the center of the distribution.

2.3.2 Variance

The variance will tell how much the gray level of pixels differs from the mean value to detect if there are any substantial light or dark spots in the image.

2.3.3 Skewness

Skewness is a measure of the asymmetry of distribution. If the skewness is negative, the data are spread out more to the left. If skewness is positive, the data are spread out more to the right. The skewness of the normal distribution (or any perfectly symmetric distribution) is zero. Data that are skewed left mean that the left tail is long relative to the right tail. Similarly, data that are skewed right means that the right tail is long relative to the left tail (Matthews, 2010).

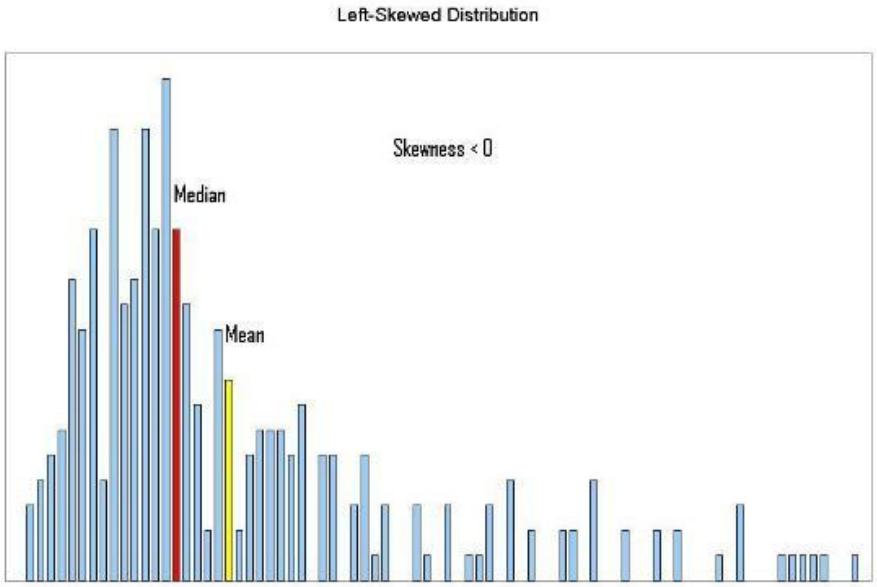


Figure 2.1: Left skewed distribution

Source: Patrick G. Matthews (2010)

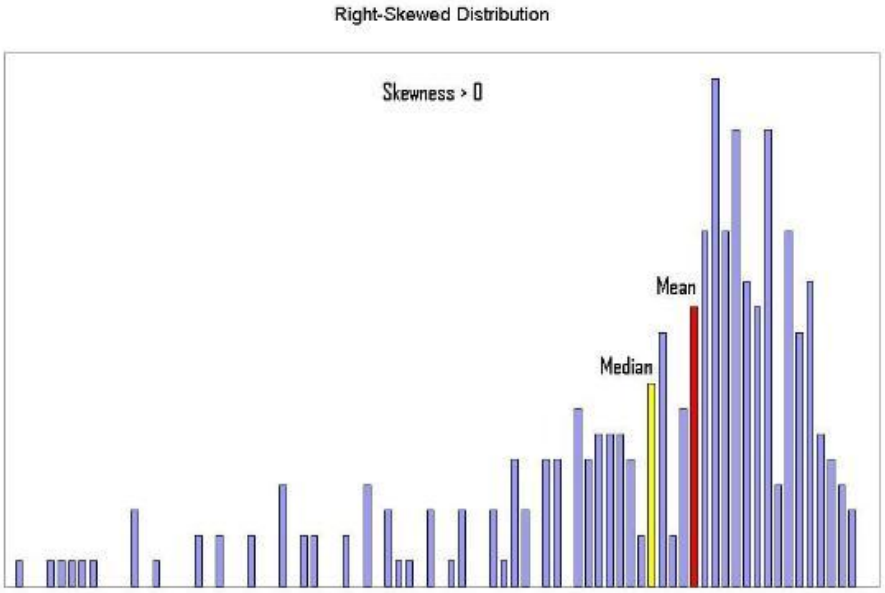


Figure 2.2: Right skewed distribution

Source: Patrick G. Matthews (2010)

2.3.4 Kurtosis

Kurtosis is a measure of whether the data are peaked or flat relative to a normal distribution. That is, data sets with high kurtosis tend to have a distinct peak near the mean, decline rather rapidly and have heavy tails. Data sets with low kurtosis tend to have a flat top near the mean rather than a sharp peak. Standard normal distribution has a kurtosis of zero. Positive kurtosis indicates a peaked distribution and negative kurtosis indicates a flat distribution (Matthews, 2010).

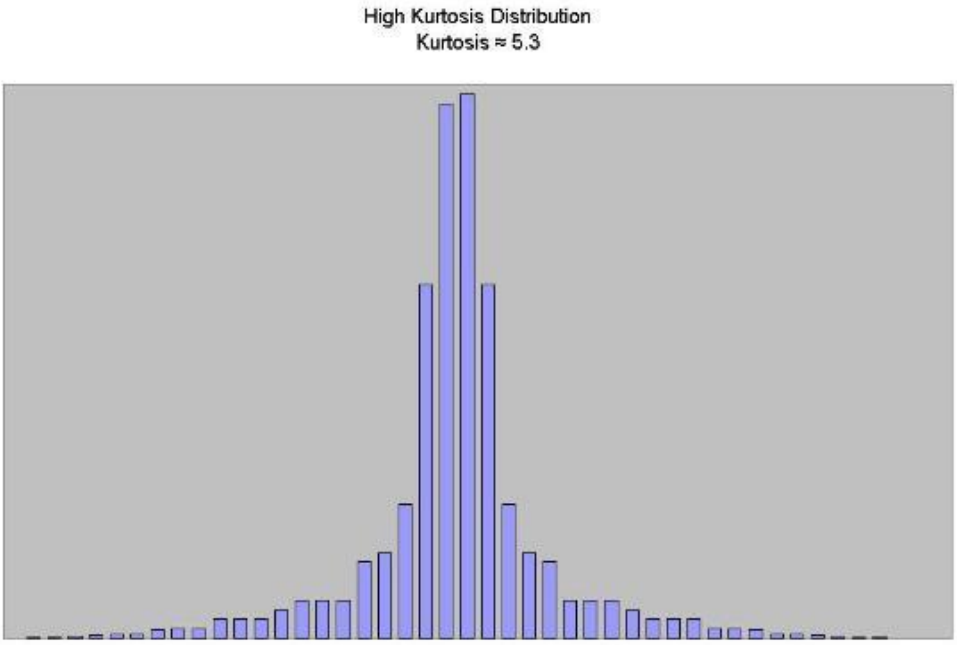


Figure 2.3: High kurtosis distribution

Source: Patrick G. Matthews (2010)

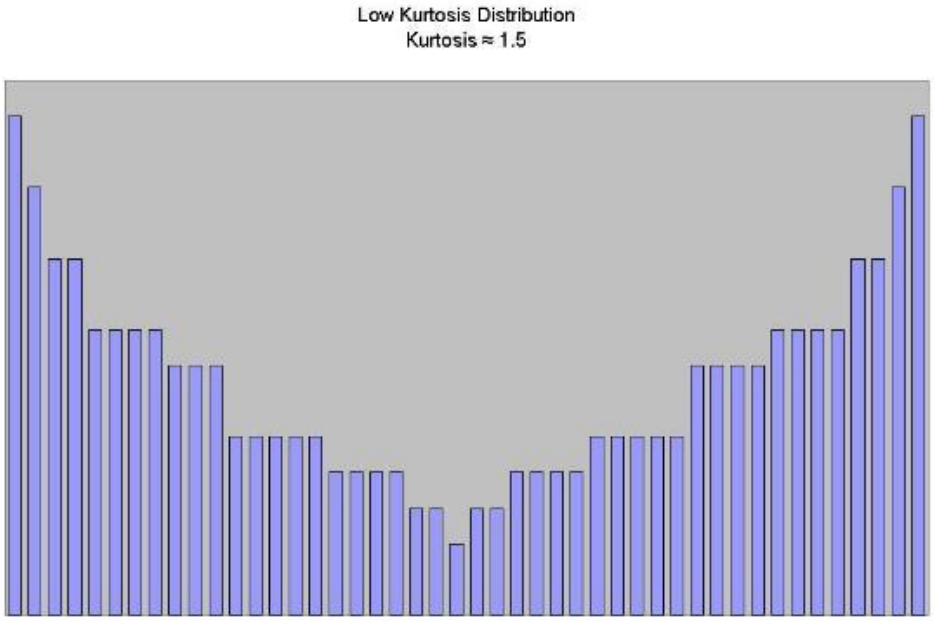


Figure 2.4: Low kurtosis distribution

Source: Patrick G. Matthews (2010)

2.4 NEURAL NETWORK

Neural networks are nonlinear information (signal) processing device, which are built from interconnected elementary processing devices called neurons. It is inspired by the way of biological nervous system, such as brain process information. Neural network is composed of a large number of highly interconnected processing elements (neurons) working in union to solve specific problem. It is configured for a specific application, such as pattern recognition or data classification through a learning process. Through a learning process, knowledge is acquired by the network from its environment. Learning involves the adjustments of the synaptic connections that exist between the neurons. The interneuron connection strengths, known as synaptic weight are used to store the acquired knowledge (Sivanandam, S.N. et al., 2011).

2.4.1 Structure of an Artificial Neural Network

Artificial Neural Networks is an information-processing system. In this information-processing system, the elements called as neurons, process the information. The signals are transmitted by means of connection links. The links posses as associated weight, which is multiplied along with the incoming signal (net input). The output signal is obtained by applying activations to the net input.

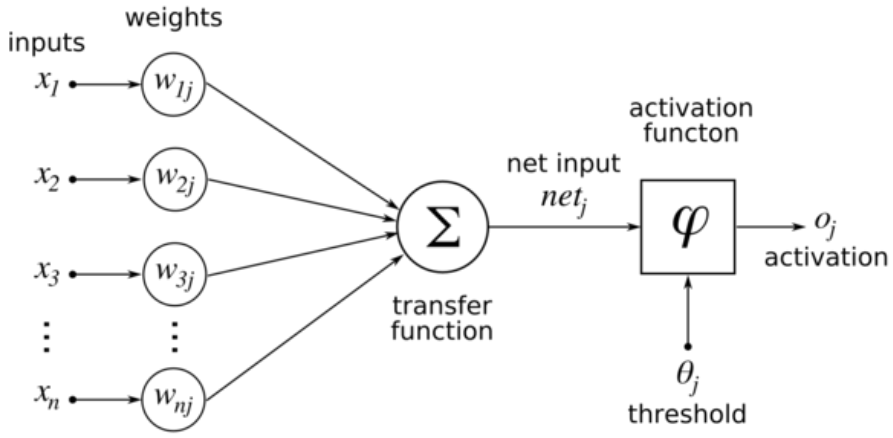


Figure 2.5: Structure of an artificial neural network

Source: Konar Amit (2009)

An artificial neuron is characterized by:

(i) Architecture (connection between neurons)
(ii) Training or learning (determining weights of the connections)
(iii) Activation function

2.4.2 Neural Network Architecture

The arrangement of neurons into layers and the pattern of connection within and in-between layer are generally called as the architecture of the net. The neuron within a layer is found to be fully interconnected or not interconnected. The number of layer in the net can be defined to be the number of layers of weighted interconnected links

between the neurons. If two layers of interconnected weights are present, then it is found to have hidden layers (Sivanandam, S.N. et al., 2011). There are various types of network architectures such as Feedforward Net, Competitive Net and Recurrent Net.

Feedforward Networks can be divided into Single layer and Multilayer. Single layer Feedforward Networks has only one layer of weighted interconnections. This type of network consists of only two layers, namely input layer and the output layer. The inputs are directly connected to the outputs. It is strictly a feedforward type and it is called single layer because only the output layer performs the computational. Multilayer Feedforward Networks is consists of multiple layers which it has hidden layers between input and output layer. The hidden layer helps in performing useful computational by extracting progressively more meaningful features from input pattern before directing the input to the output layer. This network also exhibits high degrees of connectivity determined by the synapses of the network. This is advantageous over single layer that it can be used to solve more complicate problems (Sivanandam, S.N. et al., 2011).

Competitive Networks is similar to a Single layer Feedforward Network except that there are connections usually negative between the output nodes. These connections cause the output nodes tend to compete to represent the current input pattern. Sometimes the output layer is completely connected and sometimes the connections are restricted to the units that are close to each other. This type of network has been used to explain the formation of topological maps that occur in many animal sensory systems including vision, audition, touch and smell (Sivanandam, S.N. et al., 2011).

Recurrent Networks is different from Feedforward Networks where it has at least one feedback loop. It is also allow networks to process sequential information. Processing in Recurrent Networks depends on the state of the network at the last time step. Consequently, the response to the current input depends on previous inputs. For Fully Recurrent Networks, all units are connected to all other units and every unit is both an input and an output (Sivanandam, S.N. et al., 2011).

Table 2.1 shows the summary of architectures of neural networks types for Perceptron, Associative Reward-Penalty, Backpropagation, Cohen-Grossberg, Learning

Vector Quantization, Counterpropagation Network and Adaptive Resonance Theory. Table 2.2 shows the summary of the application for some of neural networks. Some of the applications are adaptive nonlinear control, optimal control, nonlinear modelling, adaptive filtering and prediction and adaptive control using generic index.

Table 2.1: Summary of the architectures of neural networks types

Architecture	Type of connection	Main characteristics
Perceptron	1 layer feedforward	Linear system with on-off output
Associative Reward-Penalty	3 layers feedforward	Use of low quality information as feedback
Backpropagation	Several layers feedforward	Nonlinear system decision regions
Cohen-Grossberg	1 layer feedback	Dynamic nonlinear system
LVQ (Learning Vector Quantization)	Lateral connections	Quantization
CPN (Counterpropagation Network)	Lateral connections feedforward	Nonlinear system
ART II (Adaptive Resonance Theory)	2 layers feedback lateral connections	Adaptive classifer

Source: K. J. Hunt et al. (1992)

Table 2.2: summary of the application of the neural networks

Application	Architecture	Learning
Adaptive control using generic index	BP (Backpropagation)	Reinforcement Learning
Adaptive filtering and prediction	BP (Backpropagation)	BP learning alg.
Adaptive nonlinear control	BP (Backpropagation)	BP learning alg.
	CMAC	Delta rule
	Kohonen and linear system	Kohonen alg. and delta rule
	CPN	Kohonen alg. and delta rule
Optimal control	Kohonen and linear system	Kohonen alg. and delta rule
Nonlinear modelling	BP (Backpropagation)	BP learning alg.

Source: K. J. Hunt et al. (1992)

2.4.3 Training Methods

The method of setting the value for the weights enables the process of learning or training. The process of modifying the weights in the connections between network layers with the objective of achieving the expected output is called training a network. The internal process that takes place when a network is trained is called learning. Training methods on Neural Network can be classified into supervised training, unsupervised training and reinforced training (Sivanandam, S.N. et al., 2011).

Supervised training is the process of providing the network with a series of sample inputs and comparing the output with the expected responses. The training continues until the network is able to provide the expected response. A teacher is assume to be present during the learning process when a comparison is made between the network's computed output and the correct expected result to determine the error. The error then is used to change the network parameters which will improvement the

performance of the application. In a neural net, for a sequence of training input vectors there may have target output vectors. The weights may be adjusted according to a learning algorithm. Supervised training is adopted in pattern association as well. Some of the supervised learning algorithms include Backpropagation, Counter Propagation and Pattern association memory (Sivanandam, S.N. et al., 2011).

In the unsupervised training, the target output is not presented to the network. It seems no teacher to present the desired patterns. The system learns by its own by discovering and adapting to structural features in the input patterns. It may modify the weight so that the most similar input vector is assigned to the same output unit. Unsupervised networks are far more complex and difficult to implement. It involves looping connections back into the feedback layers and iterating through the process until some sort of stable recall can be achieved. Unsupervised networks are also called self-learning networks or self-organizing networks because of their ability to carry out self-learning. This is the method adopted in the case of self-organizing feature maps and adaptive resonance theory (Sivanandam, S.N. et al., 2011).

In reinforced training, a teacher is assumed to be present, but the right answer is not present to the network. Instead, the network is only presented with an indication of whether the output answer is right or wrong. The network then must use this information to improve its performance. Reinforced learning is a very general approach to learning that can be applied when the knowledge required to apply supervised learning is not available (Sivanandam, S.N. et al., 2011).

2.5 TYPES OF NEURAL NETWORK

2.5.1 Feedforward Networks

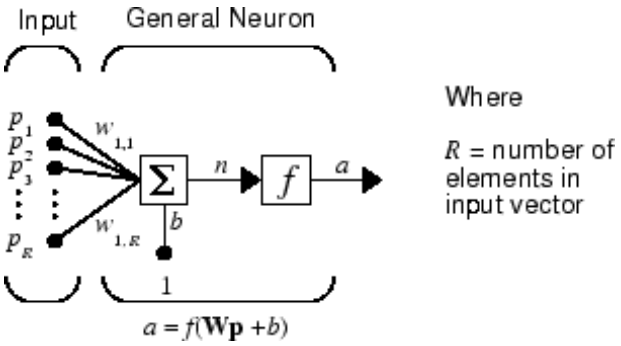


Figure 2.6: Neuron model for Feedforward Network

Source: Beale M.H. et al. (2011)

Figure 2.6 shows the neuron model for Feedforward Network with an elementary neuron of *R* input. Each input is weighted with an appropriate *w*. The sum of the weighted inputs and the bias forms the input to the transfer function *f*. Neurons may use any differentiable transfer function *f* to generate their output.

A mobile robot used Feedforward control in its neural network approach. In this work a Multilayer Perceptron Feedforward Neural Network using an on-line weight tuning is involved. This neural network has an input layer with 6 neurons, a hidden layer with 8 neurons and the output layer has 2 neurons. The most important feature of neural network is its capability to approximate multivariate nonlinear continuous function. For feedforward control, they use the velocity control from the kinematic controller. Due to absence of a complete knowledge of the parameters involved, the neural network is used to perform the nonlinear mapping. The simulations were performed under real time that is parking maneuver and a path following task (Oliveira, V.M. et al., 2000).

From this research, they used neural network to handle the incomplete knowledge of the dynamics and physical parameters of a mobile robot. Neural network

cannot have a large number of neurons in the hidden layer in order to respect the time restrictions.

Another research that involved this type of neural network is a research on characterization of a neural network-based trajectory recognition optical sensor for an automated guided vehicle. Characterization of a relatively simple optical sensor system used for recognition of the desired fixed trajectory for an automated guided vehicle, painted on an industrial shop floor. The optical sensor consists of 14 infrared (IR) emitter-detector pairs arranged in two columns and id fixed underneath the vehicle chassis. The AGV is designed to track a fixed route and an optical marking in the form of a strip painted on the floor is the route guide or the desired trajectory. The colour of the painted strip contrasts against that of the floor. The objective of the navigation control of the AGV is to keep the vehicle along the painted route guide during its motion. The navigation system of the vehicle consists of three units that are the image acquisition unit (IAU), the image processing unit (IPU) and the trajectory control unit (TCU). A microcomputer based test platform for evaluation of the microcomputer sensor is used. The sensor performance is evaluated using geometrical algorithms and one based on neural networks. Two Artificial Neural Networks (ANNs) is used to estimate the vehicle position deviation to the painted route guide. The input to each of the ANNs is the 14 IR detector conditioning circuit output voltages. The NNs are Forward Multilayer Perceptron type and implement identification of a nonlinear model. These employ nonlinear neurons in the hidden layer and linear output neurons. This implementation provides global approximations or generalizations of the input patterns not used during the training phase. These generalizations are desirable once we recognize that we have a limited number of valid test points due to the discrete nature of positioning procedure employ in the test platform. Both the ANNs contain three layers. The training of NNs is done to enable them to perform a nonlinear mapping procedure. The training is based on backpropagation algorithm (Borges, G.A. et al., 1998).

From this research, the use of ANNs has given the lowest values of the cost functions. The performance curve of the ANNs is much smoother than those of the geometrical algorithms. The smoothness of the curve is an advantage for a trajectory controller with large sensitivity at high frequencies. The ANN based method has

demonstrated much better results compared to the geometrical formulations although it takes more time for processing the acquired image. The two ANNs can be replaced by only one ANN with two outputs. This should reduce the image processing time.

A work in robot end-effector recognition using modular neural network for autonomous control is designed from several modules of neural networks and each of them employs a Multilayer Feedforward Neural Network. The modular neural network is used to recognize the robot's end effector precisely and to minimizing the processing time. The designed recognition system consists of four parts. An image obtained from a CCD camera is preprocessed and transformed to a smaller size by the preprocessing neural network. The recognition neural network searches the position of end-effector from a down-sampled, preprocessed image. The size information is identified by the size indicator neural network. The transformed images are used to train and test the recognition system. Due to the variety of end-effector images, a recognition system with one neural network may not be appropriate since it can degrade the accuracy of recognition. The training patterns are classified into several groups according to the shape of end-effector. To obtain each group, a variation of LVQ (Learning Vector Quatization) clustering technique is used. The clustering is achieved through the iterative and hierarchical approach using the variance and the mean of each group. Each module is independently trained to recognize different shapes with a common property using the backpropagation algorithm. The designed neural network is used to recognize the end-effector from the local search space with a predefined size. The modules of neural networks are to recognize the end-effector even if it shows various types of shapes. The weight of one module is locally tuned to represent the local property. The output can represent the closeness of the input to the model that the module represents. The outputs of modules are fully interconnected to final output node. Once the recognition neural network finds the position of the end-effector, the size information is scanned through the size indicator neural network. The output is mapped to the value from -1 to +1 linearly in proportion to size of the end-effector. The training patterns are generated from edges of the end-effector which are scaled for the size information (Park, D.S. et al., 1999).

The performance of the modular neural network is better than the single module neural network. The control system using modular neural network can perform the recognition with high precision even under hard conditions. The training using a series of the backpropagation and the clustering algorithm is successful in supporting the capability of the invariant recognition. The modular neural network can be used for line detection algorithm.

Saman Razavi and Bryan A. Tolson presented a research on a new formulation for feedforward neural networks. Feedforward neural networks, also known as multilayer perceptrons, with one hidden layer have been proven capable of approximating any function with any desired accuracy provided that associated conditions are satisfied. It also is one of the most commonly used function approximation techniques and has been applied to a wide variety of problems arising from various disciplines. However, neural networks are black-box models having multiple challenges/difficulties associated with training and generalization. Saman Razavi and Bryan A. Tolson had look into the internal behaviour of neural networks and develops a detailed interpretation of the neural network functional geometry. Based on this geometrical interpretation, a new set of variables describing neural networks is proposed as a more effective and geometrically interpretable alternative to the traditional set of network weights and biases. They develop a new formulation for neural networks with respect to the newly defined variables; this reformulated neural network (ReNN) is equivalent to the common feedforward neural network but has a less complex error response surface. To demonstrate the learning ability of ReNN, two training methods involving a derivative-based (a variation of backpropagation) and a derivative-free optimization algorithms are employed. Moreover, a new measure of regularization on the basis of the developed geometrical interpretation is proposed to evaluate and improve the generalization ability of neural networks. Multiple problems are test to show the value of the proposed geometrical interpretation, the ReNN approach, and the new regularization. Results show that ReNN can be trained more effectively and efficiently compared to the common neural networks and the proposed regularization measure is an effective indicator of how a network would perform in terms of generalization (Razavi, S. and Tolson, B.A., 2011).

The reformulate neural networks can enhance the neural network usefulness and learning ability. Neural networks training results shows that the ReNN approach can enhance the efficiency and effectiveness of a training algorithm by reducing the complexity of the network error function surface. Moreover, the new approach to neural network internal geometry was utilized to develop a new measure of regularization as an estimate indicator of how well a neural network would perform in terms of generalization. The ReNN formulation is less concise and more complex compared to ANN formulation, however, once ReNN is efficiently coded, it gives benefit to users such as ReNN can be trained much faster than ANN saving the user considerable training time and ReNN variables can be directly interpreted helping the users develop more reliable networks.

2.5.2 Perceptron Networks

Perceptron Network is implemented in a research on hand gesture recognition using neural networks. A pattern recognition system used a transform that converts an image into a feature vector, which then be compared with the feature vectors of a training set of gestures. The perceptron is a program that learns concepts. The structure of a single perceptron is very simple. There are two inputs, a bias and an output. Each of the inputs and the bias is connected to the main perceptron by a weight. A weight is generally a real number between 0 and 1. When the input number is fed into the perceptron, it is multiplied by the corresponding weight. After this, the weights are all summed up and fed through a hard-limiter. The way a perceptron learns to distinguish patterns is through modifying its weights. The concept of a learning rule must be introduced. In the perceptron, the most common form of learning is by adjusting the weights by the difference between the desired output and the actual output. The perceptron convergence algorithm is first initialization, second is activation, third computation of actual response, and fourth is adaption of weight vector. Perceptrons can only solve problems where the solutions can be divided by a line and this called linear separation. The research has developed algorithm using supervised learning where the learning rule is provided with a set of examples of proper network behaviour and it is an input to the network and also the corresponding target output. As the inputs are applied to the network, the network outputs are compared to the targets. The learning rule is

then used to adjust the weights and biases of the network in order to move the network output closer to the targets. Adapt is another function in MATLAB for training a neural network, the main difference with train is that only batch training can be used while adapt you have the choice of batch and incremental training. Adapt supports far less training functions. This research use train which is more flexible and no need for incremental training. Train applies the inputs to the new network, calculates the outputs, compares them to the associated targets, and calculates a mean square error. If the error goal is met, or if the maximum number of epochs is reached, the training is stopped and train returns the new network and a training record (Klimis Symeonidis, 2000).

From this research, Perceptron Network not fully stable and there are so many parameters that one can play to find the optimal setting. The network is adjusted based on a comparison of the output and the target until the network output matches the target. MATLAB is perfect for speeding up the development process but it can be very slow on execution when bad programming practices have been employed.

Another work that related to this type of neural network is a research on Multilayer Perceptron (MLP) Neural Networks to control a mobile robot using multiple goal points. A new method to track a path based on an Artificial Neural Network (ANN) by using Multilayer Perceptron ANN as controller that computes the steering command to follow a previously recorded path. A new method to track a path is presented. The method takes into account the same idea used by the RRF Controller where the Radial Basis Function (RBF) Controller calculates the steering command using a RBF Neural Network, whose inputs are the distance between the current position and the selected goal point, and the angle defined by the line that joins both points and one of the axis of the local reference system, but several goal points are selected to get the most adequate steering command to follow a previously recorded path. Instead of using single goal point placed at a given distance of the vehicle, the proposed method considers a segment of the path ahead of the vehicle by defining several goal points. Thus, the steering command is obtained by using the set of goal points, in such a way that the characteristics of the segment of the path are taken into account. A recorded path consists of a set of records. If a recorded path is given, the steering commands to track the path are calculated using the control loop. When a

driver drives a vehicle, he doesn't look at only one goal point. He usually looks at a piece of the path located at a given distance in order to select the appropriated curvature to follow the path. This is the path tracking problem. The implemented controller by a MLP ANN where a fixed distance and the goal point located at this distance is selected. The other successive points are selected near this first point using a threshold. Different values and number of points have been used in several experiments to obtain the driver's behaviour tracking a path (Pavon, N. et al., 2001).

This research implemented MLP ANN into the autonomous path tracking. The controller implements the nonlinear function that relates the steering command with the vehicle's relative position and orientation with respect to the path to be followed. Instead of using a single point on the path to define the position and orientation errors of the control loop, a piece of the path consisting of a set of goal points is selected. The controller by using MLP ANN gives efficiency to autonomous driving.

2.5.3 Radial Basis

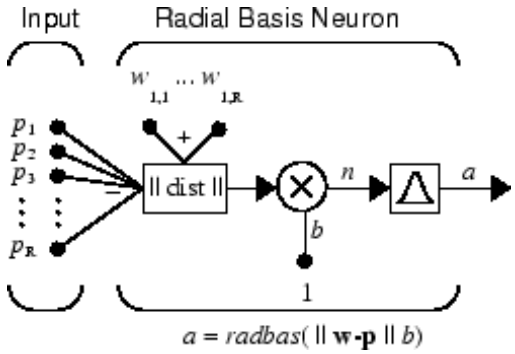


Figure 2.7: Neuron model for Radial Basis Network

Source: Beale M.H. et al. (2011)

Figure 2.7 shows the neuron model for Radial Basis Network. The net input to the radbas transfer function is the vector distance between its weight vector w and the input vector p, multiplied by the bias b.

A Radial Basis Function Neural Network (RBFN) is used for classification of fused images for human face recognition. Fusion of visual and thermal images has been done to take the advantages of thermal images as well as visual images. Method fused images are generated using visual and thermal face images in the first steps. For the second step, fused images are projected into eigenspace. Finally, RBFN is used for classified. Neural network have been employed and compared to conventional classifiers for a number of classification problem. The results have shown that the accuracy of the neural network approaches is equivalent or slightly better than other methods (Bhowmik, M.K. et al., 2009)

Radial Basis Function (RBF) Neural Networks are found to be very effective because they are universal approximates, have a very compact topology and their learning speed is very fast because of their locally tuned neurons. An important property of RBFN is that they form a unifying link between many different fields such as function approximation, regularization, noisy interpolation and pattern recognition. RBFN is excellent types of neural network for pattern classification where attempts have been carried out to make the learning process in this type of classification faster than normally required for the multilayer feed forward neural networks.

Radial Basis Function Networks (RBFN) approach is used for the tracking problem of mobile robots. The use of RBFN based controllers is to add robustness in the control system and to avoid the nonlinear optimization techniques used in the learning algorithm of the Multilayer Neural Networks (MNN) and related problems of local minimum. The structure of these networks allow the on line adaption of the networks parameters for improving the tracking performances. RBFN is a special two layer network which is linear in the parameters by fixing all radial basic function centers and nonlinearities in the hidden layer. The hidden layer performs a fixed nonlinear transformation with no adjustable parameters and it maps the input space onto a new space. The output layer then implements a linear combiner on this new space and the only adjustable parameters are the weights of this linear combiner. These parameters can therefore be determined using standard least squares techniques, which is an important advantage of this approach because of these techniques are faster than the gradient methods used in MNN. The particular structure of RBFN allows their ability to

adaptive control schemes. The RBF centers are fixing in number and position and adapting the weights of the network second layer in line with the plant. This adaption allows the control scheme to cope with parameters uncertainties and external disturbances. The output layer of RBFN is only a linear combination of the hidden layer signals and there is only one hidden layer. RBFN allow for a much simpler weights updating procedure with respect to MNN and subsequently open up greater possibilities for stability proofs and network robustness. The performances of RBFN critically depend upon the chosen centers. The centers are often chosen to be a subset of the NN input data. The fixed centers should suitably sample the NN input domain, in general the centers are arbitrarily selected from NN input data points. This mechanism is an unsatisfactory method for building RBFN. The resulting RBFN often perform poorly or have a large size and numerical ill-conditioning frequently occurs (D'Amico, A. et al., 2001).

The research shows that RBFN based controllers required lower computation efforts with respect to MNN based controllers and the adaptive RBFN controller allowed improving the performances of the control scheme in the presence of disturbances on the mobile robot. RBFN based controllers showed comparable performances with respect to MNN based control schemes confirming RBFN as alternative to avoid MNN and their gradient methods for adjustment of the parameters.

2.5.4 Self-Organizing Map

Self-Organizing Neural Network approach is used for the single AGV routing problem. The objective is to find the shortest tour for a single, free-ranging AGV that has to carry out multiple pick and deliver (P&D) requests. This combinatorial optimization problem is very similar to the asymmetric travelling salesman problem (ATSP) which is known to be NP-complete. An artificial neural network algorithm based on Kohonen's self-organizing feature maps is developed to solve the problem, and made several improvements on the basic features of self-organizing maps in winner selection, learning procedures and insertion-deletion mechanism. First phase is initialization and the first iteration, second phase is winner selection, third phase is node insertion, fourth phase is weight update, fifth phase is node deletion, sixth phase is stopping rule and last phase is grouping. Winner selection and node insertion are

considered simultaneously at a single step. The best interval is search for a job rather than selecting a single node as the winner. Neighbourhood strength is defined based on the Euclidean distance between weight vectors rather than position of the nodes in the output layer. A new weight update procedure is proposed for the neighbouring nodes of a winner, which moves origin (destination) of the nodes towards destination (origin) of the winner only if they are close. A new deletion scheme and a grouping mechanism are developed to provide convergence and improve solution quality. Performance of the algorithm is tested under various parameter settings for different P&D request patterns and problem sizes, and compared with the optimal solution and the nearest neighbour rule (Soylu, M. et al., 2000).

From this research, the initial neighbourhood effect should relatively small and allow it to decrease faster as iteration proceeds to improve the solution quality. This algorithm provides good solutions within reasonable computational time for certain request patterns. The solution qualities for other patterns are acceptable but not as good. The self-organizing feature map is efficient in producing good solution for large scale TSP problem.

Another research that used this type of neural network is a research on self-organizing behaviour of a multi-robot system by a neural network approach. The goal is to assign a team of robots to every target location and minimize the total cost in the sense of shortest total robot path. The objectives are to assign the robots to the target locations and how to control the motion of mobile robot. The robots are homogeneous mobile robots with the basic capabilities for navigation, obstacle avoidance, and location recognition. A neural network approach is for controlling a group of mobile robots to achieve multiple tasks at several different locations. The model is based on SOM neural network. It combines the robot task requirement and motions planning together, which are normally handled separately in some conventional approaches. By modifying the initial weights of the neural network, the rule to select the winner, the rule of the neighbourhood function, and the rule to update the weights are extended from a conventional SOM neural network algorithm. The original SOM neural network approach, proposed by Kohonen combines a competitive learning principle with a topological structure of nodes such that adjacent nodes tend to have similar weight

vector, that means to put the associated output of similar input as close as possible to each other. SOM is a two layer neural network, first layer is the input layer including the number of pattern features, and second layer is the output layer. The connection among nodes in the second layer represents the neighbourhood relationship. The main steps of the algorithm are the first step is to select an input sample, calculate the activity nodes of the output layer, and get a winner. The second step is to decide the neighbourhood function. The third step is to modify weights of the winner and its neighbours. Repeat the first to the third step until all of the weights do not change (Zhu, A. and Yang, S.X., 2003).

This proposed approach can conduct the task assignment and path planning of multi-robots to target locations. With the self-organizing process, the proposed approach has several interesting features and advantages. It combines the target assignment and motion planning for a multi-robot system, which allows the robots stat moving before their destinations are finalized. It is an error resistant, and can deal with the sudden change of the environment such as the breakdown of few mobile robots. It can deal with some complicated case such as assigning more than one robots to a target location.

2.5.5 Learning Vector Quantization

Learning Vector Quantization (LVQ) Neural Network based target differentiation method is used for mobile robot. LVQ neural network is employed to differentiate targets based on sonar time-of-flight (TOF) and amplitude data. LVQ neural network combines the competitive learning with supervised learning and it can realize nonlinear classification effectively. In the hidden layer, only the winning neuron has an input of one and other neurons have outputs of zero. The weight vectors of the hidden layer neurons are the prototypes. The number of the hidden neurons is defined before training and it depends on the complexity of the input-output relationship. The learning phase starts by initiating the weight vectors of neurons in hidden layer. The input vectors are presented to the network in turn (Xin Ma et al., 2005).

LVQ neural network is very applicable to classification problem, from this research it is employed to process sonar data to differentiate the typical targets in indoor environment. LVQ neural network can differentiate the typical targets in a large ranging scope in indoor environment effectively and rapidly with high robustness to noise and partial removal of amplitude and time-of-flight data and different targets in some extent. Moreover the multi-level hierarchical configuration, which was motivated by the traditional statistical target differentiation method, decentralizes and relieves the training and differentiation force for a single neural network. This configuration makes the architecture of every LVQ neural network simple and the time for training neural network decreasing.

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

This chapter will explain detail about process and method used in this project. This project used MATLAB software for image processing, create and training neural network. In this project, statistical feature extraction method is used to get the values of mean, variance, skewness and kurtosis. These values are calculated from the first order statistics. These values are then used as input data to train the neural network together with the target output values. The two types of neural networks used in this project are Feedforward Backpropagation and Radial Basis. Both types of neural network are trained with same input and target. This project used supervised training where the networks are provided with inputs and target outputs. The purpose of comparing two types of neural networks is to find the best type of neural network for line recognition besides learn the recognition analysis using neural networks. This is to improve the AGVs capabilities and increase it efficiency. Camera based vision is used in this project for the vision sensor. Camera based vision system is useful in order to recognize the line guideline and allow line recognition algorithm. The types of guidelines used in this project are straight, right, left and stop guidelines.

3.2 OVERALL METHODOLOGY

Below is the flow chart of overall methodology for this project.

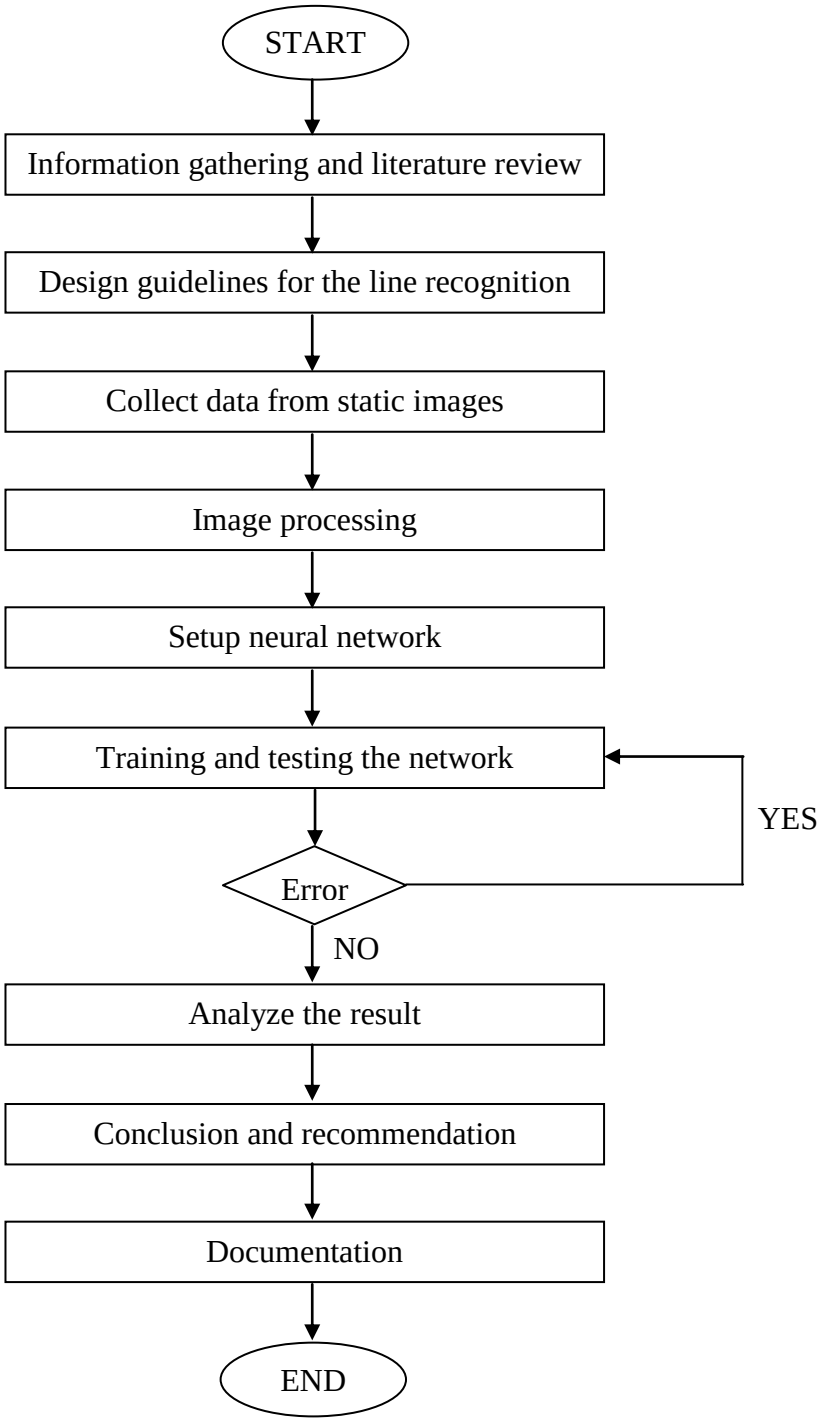


Figure 3.1: Flow chart of overall methodology

3.2.1 Guideline for the line recognition

A low cost guideline has been designed and implemented as a guiding system for this project. The guideline consists of one layer and it is white colour. It has been placed on a flat floor surface. There are four types of guidelines designed and constructed for the used of the experimental work. They are:

(i) Straight guideline
(ii) Turn right guideline
(iii) Turn left guideline
(iv) Stop guideline

Table 3.1: Images and types of the guidelines

Images	Types
	Straight guideline
	Turn right guideline
	Turn left guideline

Table 3.1: Continued

Images	Types
	Stop guideline

3.3 LINE RECOGNITION ALGORITHM

The process flow of line recognition algorithm is as follow:

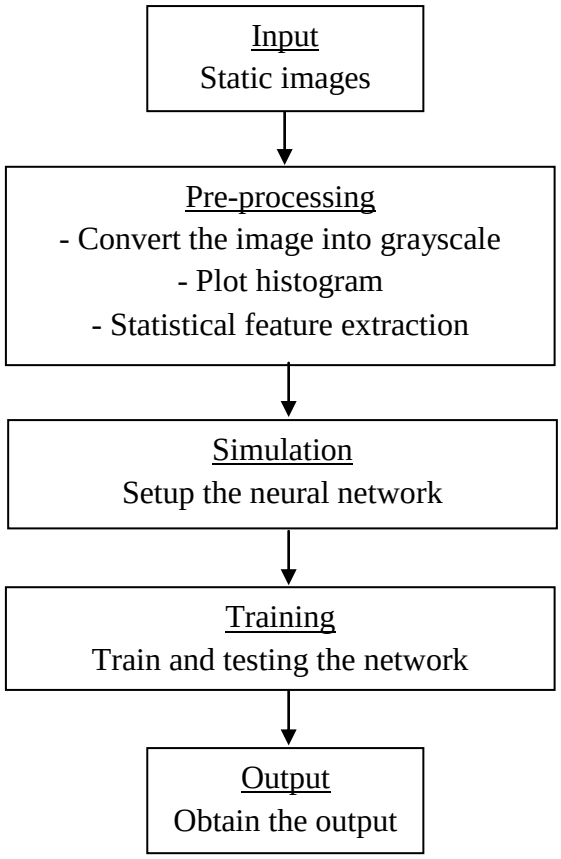


Figure 3.2: Flow chart of line recognition algorithm

There are several steps that need to be done to achieve the objectives of this project. Line recognition algorithm can be divided into four steps. First is the system initialization where the static images of the guidelines are captured by the camera. This includes parameters setting for USB camera. Second is pre-processing where the image captured by the camera will undergoes three processes. These three processes will be explained detail below. Third process is setup the neural network. Fourth is training and testing the network to get the output for the recognition of the guideline.

Two main stages were identified that are pre-processing and neural network. It is necessary to get the data input from the real image. This is to allow the real line recognition algorithm. The project is started with the image processing from the static image captured by the camera. USB2.0 camera is used in this project. The static images will undergoes pre-processing which has three stages that are convert the image into grayscale, plot histogram and statistical feature extraction which are to calculate mean, variance, skewness and kurtosis from the histogram. The process continues with neural network that does the training, testing and recognition process. The output from the two types of the neural network will be compared and analyzed.

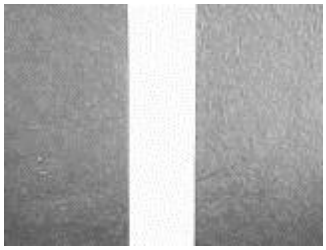
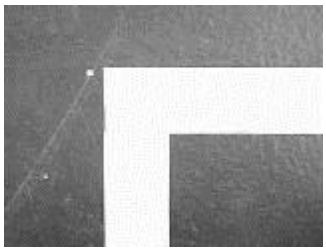
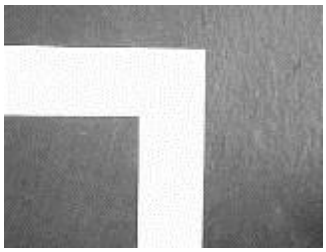
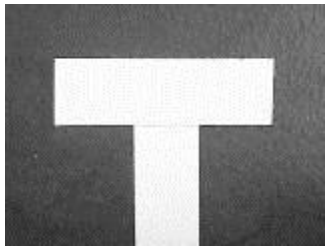
3.4 PRE-PROCESSING

The static images captured by the USB2.0 camera with size 160 x 120 pixels. The images are RGB (truecolour) image and save in JPEG format. To read the image, MATLAB coding as follow is used:

```
a=imread('C:\Users\user\Desktop\PSM\DATA\data_straight\s1.jpg');
```

The image processing is continued with converting the true colour image to grayscale image. Figure 3.2 shows the original images and grayscale images.

Table 3.2: Original images and grayscale images of the guidelines

Original images	Grayscale images	Types
		Straight guideline
		Turn right guideline
		Turn left guideline
		Stop guideline

The objective in converting true colour to grayscale is to enhance the computational with the program and to make it easier in calculating statistical feature extraction example mean value from gray level histogram. MATLAB coding as follow is used to convert the true colour image to grayscale image and plot histogram.

```
%convert to grayscale image
b=rgb2gray(a);

%create histogram
c=imhist(b);
```

Below is the example of coding and figure of original image, grayscale image and histogram from static image of straight guideline.

```
figure(1),subplot(1,2,1),imshow(a),title('original image');
figure(1),subplot(1,2,2),imshow(b),title('grayscale image');

figure(2),subplot(1,2,1),imshow(b),title('grayscale image');
figure(2),subplot(1,2,2),imhist(b),title('histogram grayscale image');
```

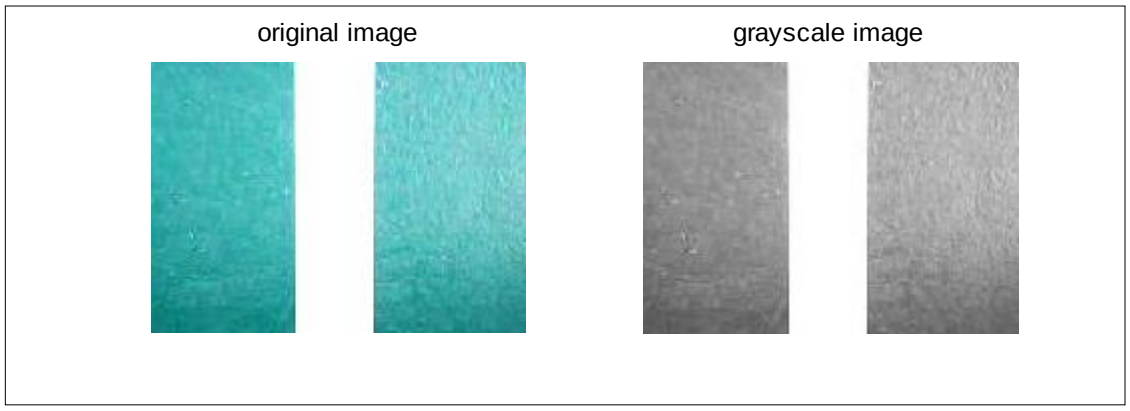


Figure 3.3: Original image and grayscale image

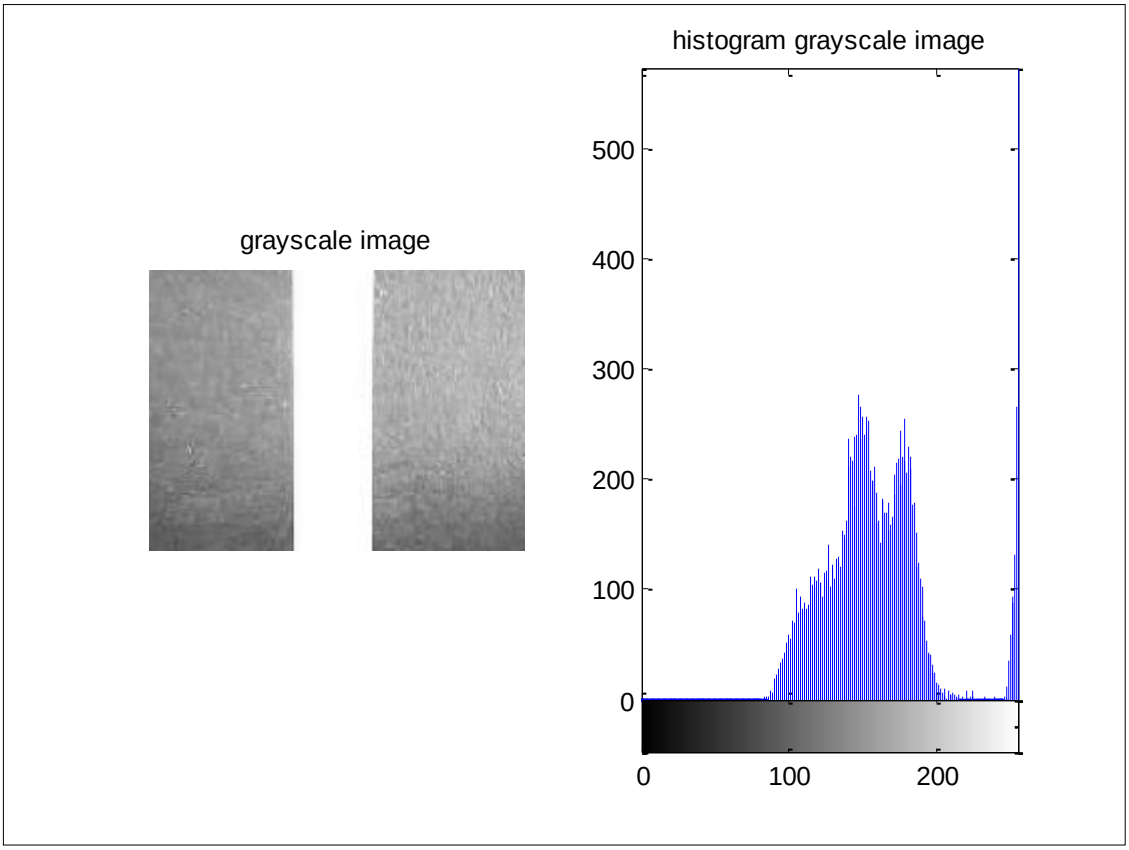


Figure 3.4: Grayscale image and histogram

From the histogram, statistical feature extraction method has been used to calculate the value of mean, variance, skewness and kurtosis from first order statistics. The mean, variance, skewness and kurtosis are calculated based on the following equation:

The equation of mean is given by:

$$\mu = \sum_{k=1}^{K} kP_k \tag{3.1}$$

where

μ = mean gray value of the pixel set

K = the number of distinct gray levels

k = gray value of the ith pixel

Pk = normalized histogram (probability density function of the pixel set)

with

$$P_k = \frac{pixels\ \ with\ \ value\ \ I}{total\ \ \ pixels} \tag{3.2}$$

The equation of variance is given by:

$$\sigma^2 = \sum_{k=1}^{K} (k-\mu)^2 P_k \tag{3.3}$$

where

σ^2 = variance value and it is square of standard deviation

μ = mean value

K = the number of distinct gray levels

k = gray value of the ith pixel

Pk = normalized histogram (probability density function of the pixel set)

The equation of skewness is given by:

$$\gamma^3 = \frac{1}{\sigma^3} \sum_{k=1}^{K} (k-\mu)^3 P_k \tag{3.4}$$

where

γ_3 = variance value

σ^3 = variance value

μ = mean value

K = the number of distinct gray levels

k = gray value of the ith pixel

Pk = normalized histogram (probability density function of the pixel set)

The equation of kurtosis is given by:

$$\gamma^4 = \frac{1}{\sigma^4}\sum_{k=1}^{K}(k-\mu)^4 P_k - 3 \tag{3.5}$$

where

γ_4 = kurtosis value

σ^4 = variance value

μ = mean value

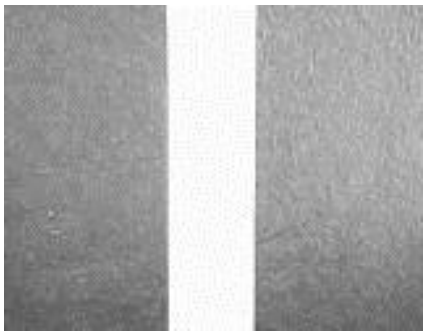
K = the number of distinct gray levels

k = gray value of the ith pixel

Pk = normalized histogram (probability density function of the pixel set)

Table 3.3 below shows an example of straight guideline with values of mean, variance, skewness and kurtosis.

Table 3.3: Values of mean, variance, skewness and kurtosis for straight guideline

Straight Guideline	Mean	Variance	Skewness	Kurtosis
	75	4.70e+04	12.62485218	1.86e+02
	75	6.18e+04	11.63830847	1.65e+02
	75	6.09e+04	11.60169962	1.64e+02
	75	4.05e+04	9.210651458	1.17e+02
	75	5.88e+04	12.46153747	1.82e+02
	75	5.71e+04	12.44068465	1.82e+02
	75	6.25e+04	11.46884835	1.61e+02
	75	5.31e+04	12.41856007	1.82e+02
	75	5.87e+04	11.65002095	1.65e+02
	75	6.48e+04	11.21113539	1.56e+02

The guidelines have four types. Each type of the guideline is captured ten times. These made the total images are forty. All these forty images are calculated their mean, variance, skewness and kurtosis values. The value of mean is not used for training and testing in the neural network because the value is same for all forty images.

3.5 NEURAL NETWORK

The data from the values of variance, skewness and kurtosis of the static images are then applied in the neural network training and testing. The total static images are forty. Twenty of the images are used for training and other twenty are used for testing. The data for each static image is 1 row and 3 columns. For the input data it is 20 rows and 3 columns. The data input is represented as a row vector enclosed by squared brackets and separated by semicolon. The data is defined as variable called input. The input is transposed to get 3 rows and 20 columns and defined as follow:

```
input = [4.70E+04 12.62485218 1.86E+02; 6.18E+04 11.63830847 1.65E+02;
         6.09E+04 11.601699 1.64E+02; 4.05E+04 9.210651458 1.17E+02;
         5.88E+04 12.461537 1.82E+02; 4.08E+04 9.491299923 1.21E+02;
         4.19E+04 9.2304750 1.14E+02; 4.43E+04 9.483165912 1.15E+02;
         4.55E+04 9.7195829 1.22E+02; 5.15E+04 10.26267141 1.33E+02;
         4.46E+04 10.390226 1.35E+02; 3.73E+04 8.295582481 98.34061;
         5.16E+04 10.052764 1.32E+02; 4.29E+04 9.162574051 1.16E+02;
         3.70E+04 8.0491890 93.36156; 4.74E+04 11.66197149 1.63E+02;
         4.88E+04 9.6509691 1.25E+02; 5.17E+04 10.11116596 1.34E+02;
         7.05E+04 11.855885 1.69E+02; 6.55E+04 11.41434627 1.60E+02]';
```

It is also necessary to define the target output with a variable called target. The target is defined as 20 rows and 1 column. Target output is also defined as row vector same as the input vector. The target is transposed to get target data 1 row and 20 columns. The target output has been defined as follow:

```
target=[1;1;1;1;1;2;2;2;2;2;3;3;3;3;3;4;4;4;4;4]';
```

After training the network it is continued with simulate testing data. The testing data defined as 20 rows and 3 columns. The testing data is defined as variable called test. The testing data is transposed to get 3 rows and 20 columns.

```
test = [5.71E+04 12.44068465 1.82E+02; 6.25E+04 11.46884835 1.61E+02;
        5.31E+04 12.4185600 1.82E+02; 5.87E+04 11.6500209 1.65E+02;
        6.48E+04 11.2111353 1.56E+02; 6.57E+04 12.2208201 1.77E+02;
        4.00E+04 8.92110301 1.10E+02; 4.39E+04 8.56186287 1.02E+02;
        4.01E+04 8.88375332 1.07E+02; 7.09E+04 12.6851916 1.86E+02;
        4.61E+04 9.93190857 1.30E+02; 4.45E+04 9.44783783 1.19E+02;
        3.45E+04 7.37125827 79.51295; 4.18E+04 8.39742722 99.62106;
        3.87E+04 8.95611901 1.11E+02; 5.70E+04 10.6448516 1.44E+02;
        5.15E+04 10.6152340 1.44E+02; 5.68E+04 11.4698601 1.62E+02;
        4.78E+04 9.79464618 1.26E+02; 5.21E+04 10.0117442 1.33E+02]';
```

The two types of neural networks used are Feedforward Backpropagation and Radial Basis.

3.5.1 Feedforward Backpropagation

The network is constructed one hidden layer and one output layer. For hidden layer and output layer, tansig transfer function is used. Training function used is "trainlm" which is Levenberg-Marquardt algorithm. It is the fastest training algorithm for networks of moderate size and has memory reduction feature for use when the training set is large. There is only one training function associated with a given network. Performance function used is Mean Squared Error (MSE). At the first hidden layer, 10 neurons are taken by guesswork. If the network has trouble learning which cannot produce the desired output, the neurons can be added to the hidden layer. Increasing the large number of neurons also needs to consider the learning time and the output result. If the network solves the problem well, a fewer could be tried.

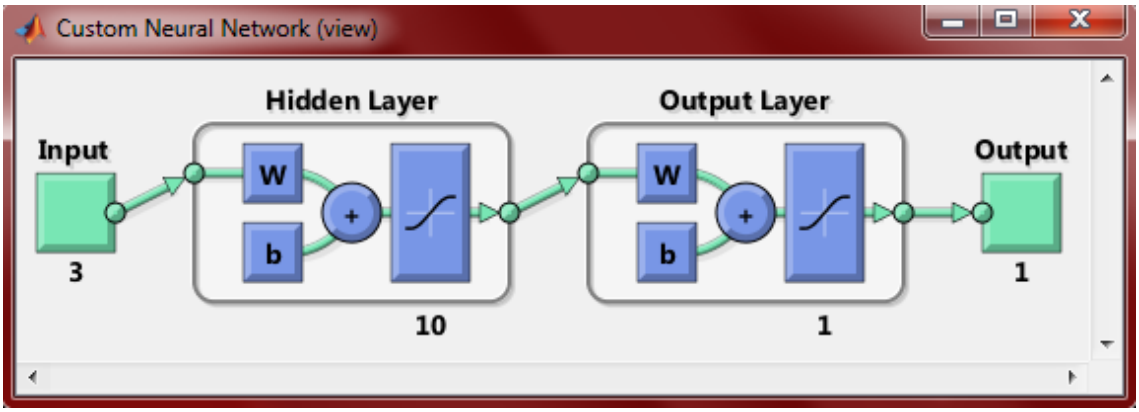


Figure 3.5: Feedforward Backpropagation Network

This network training if performed by mean squared error performance function that calculates the average squared error between the network output, o and the target output, t. For neural network training other parameter as below are used:

Performance: Mean Squared Error (mse)
Mean Squared Error goal: 0
Maximum number of epoch to train: 1000 epochs
Epochs between displays: 25 epochs

For the Backpropagation algorithm, between output (*k*) and input (*i*) layer, error is calculated by using generalized delta rule that can be expressed as:

$$\delta_j = o_j(1-o_j)\sum_k \delta_k W_{kj} \tag{3.6}$$

where

δ_j = error signal through hidden layer

o_j = output at hidden layer

δ_k = error signal at the output layer

W_{kj} = weight between output layer and hidden

Error between hidden layer (*j*) and output layer (*k*) is calculated by using equation below:

$$\delta_k = o_k(t_k - o_k)(1-o_j) \tag{3.7}$$

where

δ_k = error signal between output layer and hidden layer

o_k = output at output layer

t_k = target output at the output layer

o_j = output at the hidden layer

Since it use weight adaption rule, all layer update weight for each unit by using equation below:

$$\Delta W_{ij}(n+1) = \eta(\delta_j o_j) + \Delta W_{ij}(n) \tag{3.8}$$

where

δ_j = error signal through hidden layer

o_j = output at hidden layer

ΔW_{ij} = adaption weight between input (*i*) layer and hidden layer (*j*)

η = learning rate

n = iteration number

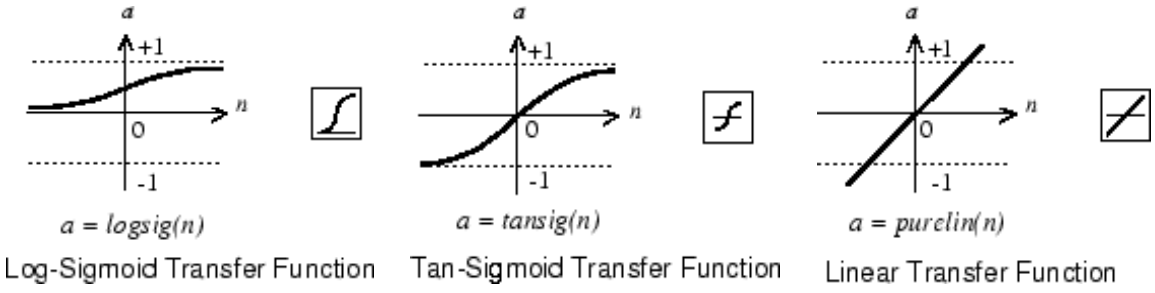


Figure 3.6: Transfer function, *f* in Feedforward Network

Source: Beale M.H. et al. (2011)

There are three transfer functions, *f* that can be used in Feedforward Network namely, log-sigmoid transfer function logsig, tan-sigmoid transfer function tansig and linear transfer function purelin. The transfer function logsig generates outputs between 0 and 1 as the neuron's net input goes from negative to positive infinity. The transfer function tansig are often used for pattern recognition problems while linear transfer function purelin are used for function fitting problems (Beale M.H. et al., 2011).

3.5.2 Radial Basis

Radial basis function network can be used for approximating functions, series prediction and recognizing patterns. It uses Gaussian Potential functions or also known as Radial Basis transfer function. The Gaussian Potential functions are also used in networks called regularization networks. The architecture of radial basis function network consists of two layers, the hidden radial basis layer and an output linear layer as shown in Figure 3.7.

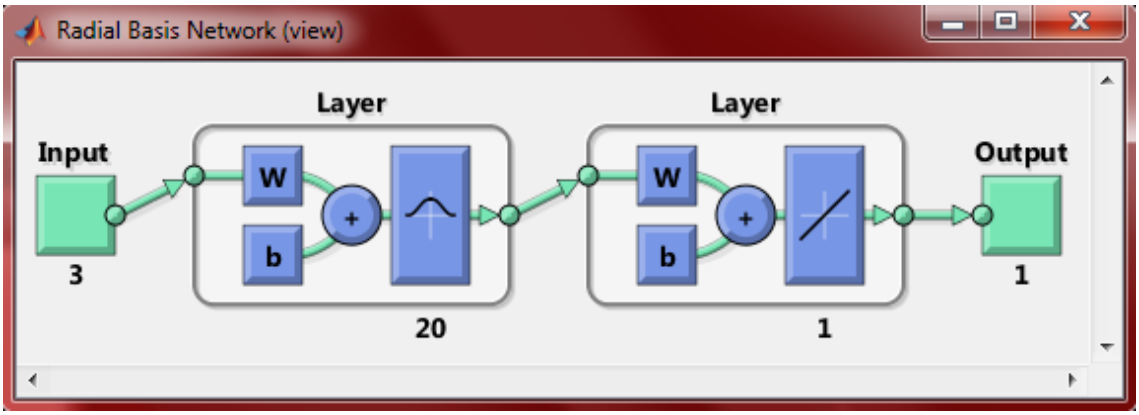


Figure 3.7: Radial Basis Network

Other parameters are used as bellow in training of radial basis network:

Performance goal: 0
Spread constant: 1.0

The output of the Radial Basis neural network is calculated by using equation below:

$$y_{net} = \sum_{i=1}^{H} w_{im} v_i(x_i) + w_0 \tag{3.9}$$

where

H = number of hidden layer nodes (RBF function)

y_{net} = Output value of m_{th} node in output layer for the n_{th} incoming pattern

w_{im} = Weight between i_{th} RBF unit and m_{th} output node

w_0 = Biasing term at n_{th} output node

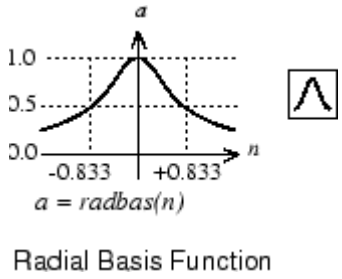


Figure 3.8: Radial Basis transfer function

Source: Beale M.H. et al. (2011)

The plot of the Radial Basis transfer function is as shown in Figure 3.6. The transfer function for a radial basis neuron is:

$$radbas(n) = e^{-n^2} \tag{4.0}$$

The Radial Basis function has a maximum of 1 when its input is 0. As the distance between weight vector and input vector decreases, the output increases. Thus, a Radial Basis neuron acts as detector that produces 1 whenever the input vector is identical to its weight vector (Beale M.H. et al., 2011).

3.6 MATRIX LABORATORY (MATLAB)

MATLAB is a high level technical computing language and interactive environment for algorithm development, data analysis, data visualization and numeric computation. It has many built-in tools for solving problems and developing graphical illustrations. MATLAB is widely used in many applications such as signal and image processing, communication, control design, test and measurement, financial analysis and modelling and computational biology. Toolboxes such as Image Processing Toolbox and Neural Network Toolbox can extend the MATLAB environment with tools for designing, implementing, visualizing and simulating neural networks (Sivanandam, S.N. et al., 2011).

Image Processing Toolbox is very useful in analyzing image and it is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image processing operations example morphological operations, image analysis and enhancement, linear filtering and spatial image transformations.

Neural Networks Toolbox is very useful for applications where formal analysis would be difficult, such as pattern recognition and nonlinear system identification, classification, vision and control systems. Neural networks can be trained to solve problems which difficult for conventional computers or human beings. The toolbox emphasizes the use of neural network to build up and can be used in engineering, financial and other practical applications (Sivanandam, S.N. et al., 2011).

For this project, MATLAB is used for image analysis and process the input image by using statistical feature extraction. Neural Networks Toolbox was used to setup the neural networks, training, compute the input, output for the neural network learning.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 INTRODUCTION

The purpose of this chapter is to show all the results obtained from this project. Tables of results, graphs and figures are all included. Detailed explanation of the results, tables, graphs and figures are also provided. The data is collected after the input data is trained regarding to the types of the neural network. The optimization method usage and interpretation of its result are obtained based on detailed study of the usage of the software involved.

4.2 RESULTS

These results are collected after the training is completed and after simulate the testing data. It is based on the two types of neural networks used that are Feedforward Backpropagation and Radial Basis.

4.2.1 FEEDFORWARD BACKPROPAGATION

This Feedforward Backpropagation Network has been tested with different transfer function for it hidden layer and output layer. The results are compared in terms of best validation performance (MSE) and R value for training, validation, test and all data. Performance function for feedforward networks is mean squared error (MSE). It is the average squared error between the network outputs and the target outputs. The line that gives the least mean squared error will be the line with the best fit. The R value is an indication of the relationship between outputs and targets. If R value is equal to one, this indicates that there is an exact linear relationship between outputs and targets. If R value is close to zero, then there is no relationship between outputs and targets.

Table 4.1: Feedforward Backpropagation Network tested with different types of transfer function

Transfer function		**R value**				**Best validation performance (MSE)**
Hidden layer	**Output layer**	**Training**	**Validation**	**Test**	**All**	
Tansig	Tansig	0.96763	1	1	0.97445	$6.5084e^{-10}$
Tansig	Purelin	0.98065	0.99946	0.22248	0.89137	0.46976
Tansig	Logsig	0.79377	0.69141	0.98829	0.84658	0.24564
Logsig	Logsig	0.7135	0.94491	-0.88451	0.73766	0.83333
Logsig	Tansig	0.94444	0.98374	0.81049	0.87016	0.90631
Logsig	Purelin	0.90001	0.89904	0.50979	0.81053	0.65622
Purelin	Purelin	0.44553	0.31188	-0.35948	0.37573	1.53093
Purelin	Tansig	0.71987	-0.94413	-0.76634	0.67372	1.36513
Purelin	Logsig	0.64336	0.86603	0.94491	0.70741	0.83333

Table 4.1 shows the results of the Feedforward Backpropagation Network tested with different types of transfer function. From the table, tansig transfer function for hidden and output layer gives the best R value for training, validation, test and all data. It is also gives the lowest mean squared error which means it gives the best line fit.

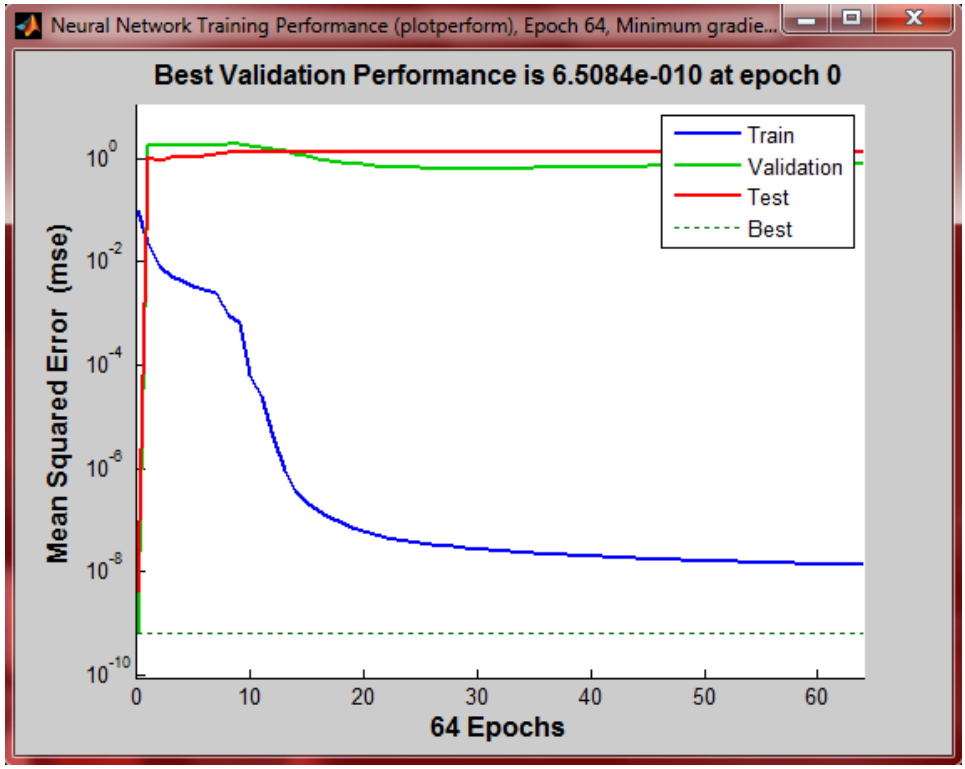


Figure 4.1: Performance for Feedforward Backpropagation Network

The process of training a neural network involves tuning the values of the weights and biases of the network to optimize network performance. The performance function for Feedforward Backpropagation networks is Mean Squared Error (MSE) where it is the average squared error between the network outputs and the target outputs. Figure 4.1 show that the best validation performance is $6.5084e^{-10}$, at epoch 0 from the total 64 epochs. This figure does not indicate any major problems with the training. The test curve and validation curves are very similar.

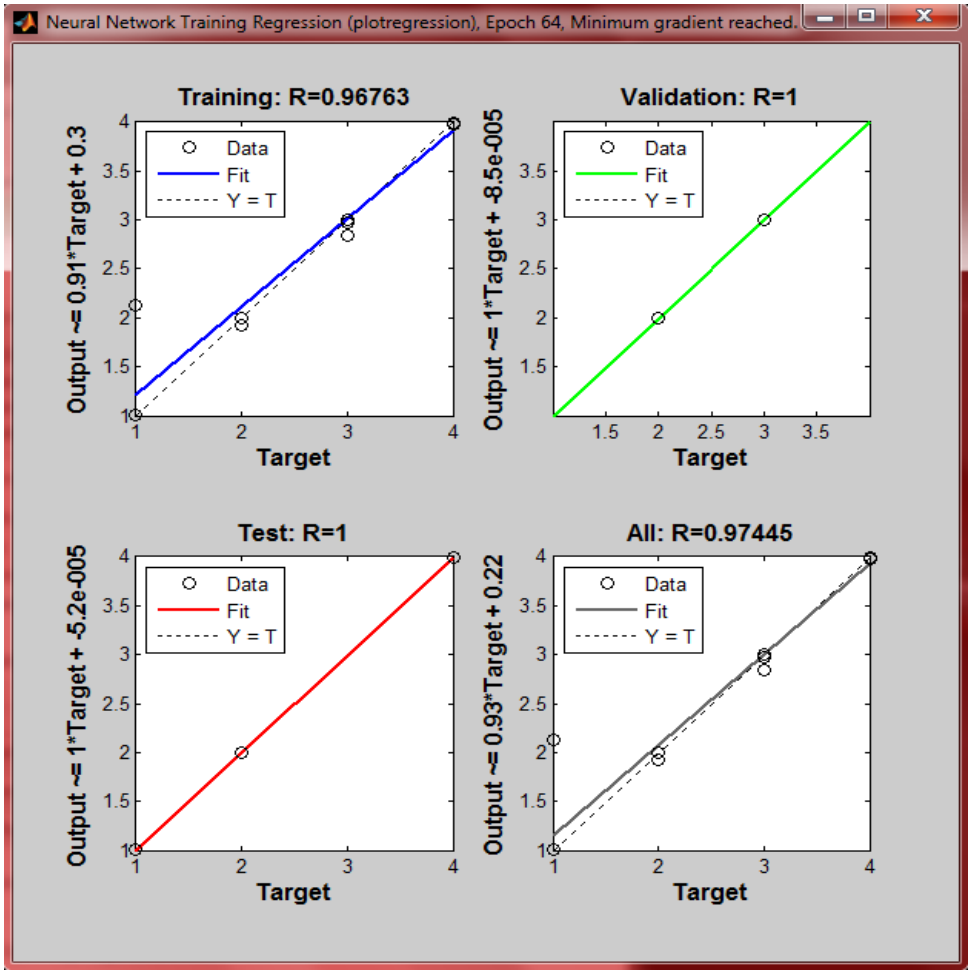


Figure 4.2: Regression plot for Feedforward Backpropagation

Regression plot is created in order for the next step in validating the network, which shows the relationship between the outputs of the network and the targets. The results shown in the Figure 4.2 represent the four axes that are training, validation, testing and all data. The dashed line in each axis represents the perfect result – outputs = targets. The solid line represents the best fit linear regression line between the outputs and targets. The R value is an indication of the relationship between the outputs and targets. If R = 1, this indicates that there is an exact linear relationship between outputs and targets. If R is close to zero, then there is no linear relationship between outputs and targets. From Figure 4.2 it shows that the R values for training is 0.96763, validation is 1, test is 1 and all is 0.97445. The training, validation, test and all data show R values that greater than 0.9. This shows that it indicates a good fit. The scatter plot is helpful in showing the certain data points that have poor fits.

Table 4.2: Feedforward Backpropagation Network tested with different number of neurons

Number of neurons	R value				Best validation performance (MSE)
	Training	Validation	Test	All	
1	0.8194	-0.50003	0.94495	0.73868	1.5038
5	0.81196	0.96934	0.78803	0.85201	0.7471
10	0.96763	1	1	0.97445	$6.5084e^{-10}$
15	0.64807	0.96313	0.77306	0.71286	0.23126
20	0.5446	0.99403	0.14869	0.50601	0.31563
25	0.27134	-0.75722	0.40224	0.10828	1.4867
30	0.74188	0.13885	0.45901	0.77575	0.26125

Feedforward Backpropagation Network had been tested with different number of neurons. From Table 4.2, number of neurons 10 shows the best R value and gives better performance or mean squared error which is lower error.

4.2.2 RADIAL BASIS

Radial Basis Networks can be used for approximating functions and recognizing patterns. It adds neurons to the hidden layer of a Radial Basis network until the maximum number of neurons has been reached. The network generates random weight to map the input to output using radial basis function.

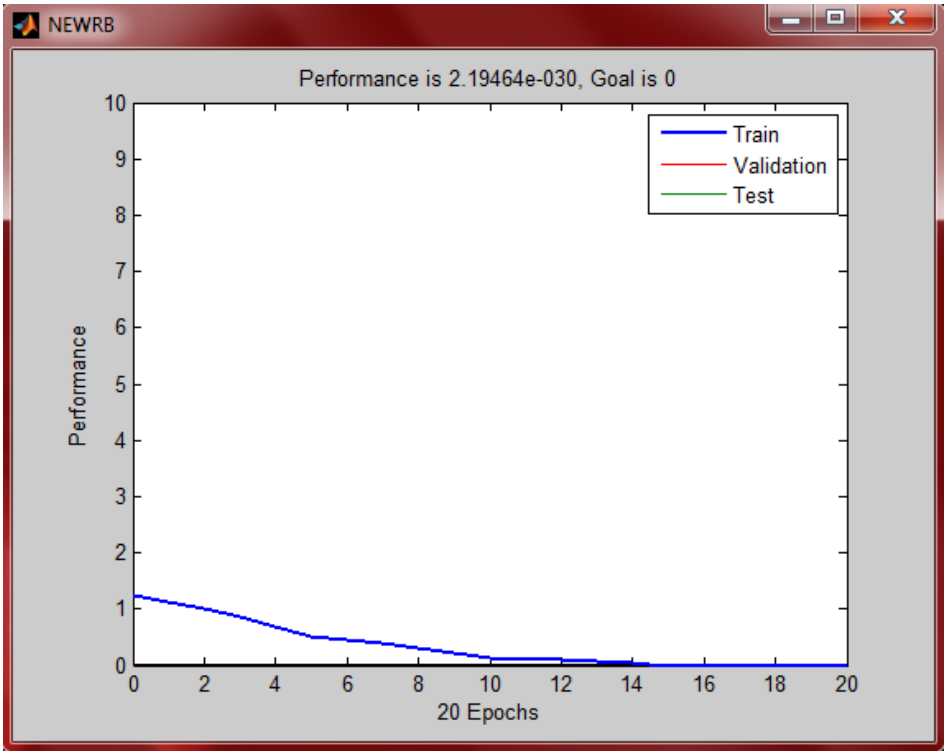


Figure 4.3: Performance for Radial Basis Network

Table 4.3: Neurons and Mean Squared Error (MSE) for Radial Basis

Neurons	Mean Squared Error (MSE)
0	1.25
2	1
3	0.852941
4	0.6875
5	0.5
6	0.446429
7	0.384615
8	0.3125
9	0.227273
10	0.125
11	0.111111
12	0.09375
13	0.0714286
14	0.0416667
15	9.09655e-030
16	4.56553e-030
17	1.4446e-030
18	1.73549e-030
19	0
20	2.19464e-030

Figure 4.3 shows that the performance for Radial Basis is 2.19464e-030 and it consists of 20 epochs. Table 4.3 shows that the neurons are added to the network until maximum number of neurons has been reached. It creates one neuron at a time. At each iteration, the input vector that results in lowering the network error the most is used to create a Radial Basis neuron. The error of the new network is checked. The next neurons are added until the maximum number of neurons is reached.

4.3 COMPARISON BETWEEN FEEDFORWARD BACKPROPAGATION AND RADIAL BASIS

Table 4.4: Performance of Feedforward Backpropagation and Radial Basis by comparing the value of mean squared error

Performance by using mean squared error (MSE)	
Feedforward Backpropagation	**Radial Basis**
$6.5084e^{-10}$	$2.19464e^{-30}$

The best performance of Feedforward Backpropagation by using mean squared error is $6.5084e^{-10}$ and for Radial Basis is $2.19464e^{-30}$ as shown in Table 4.4. Radial Basis gives better performance which resulted in significantly lower error (MSE) compared with the Feedforward Backpropagation.

Radial Basis Network is designed efficiently and it tends to have many times more neurons than a comparable Feedforward Network with tansig or logsig in the hidden layer. This is because sigmoid neurons can have outputs over a large region of the input space, while radial basis neurons only respond to relatively small regions of the input space. The result is that the larger the input space the more radbas neurons required.

Further more, designing a Radial Basis Network take much less time than training a sigmoid or linear network. Sometimes it can give result with fewer neurons

used. Radial Basis Networks may require more neurons than standard Feedforward Backpropagation Networks, but they can be designed in a fraction of the time it takes to train standard feedforward networks. It creates as many radial basis neurons as there are input vectors in the training data. It is also able to reduce the training time.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 INTRODUCTION

This chapter provide the summary of the whole project. The conclusion can be made from the outcomes, observation of results, analysis and discussion throughout the project to see whether the objectives are achieved or not. Recommendation involved the improvement of this project in the further research.

5.2 CONCLUSION

As a conclusion, the line recognition algorithm has been developed. It started with capture the static image of the guideline, pre-processing, simulate the neural network finally train and testing the network. The first objective of this project is to develop a line recognition algorithm for automated guided vehicle (AGV). This objective was achieved.

Second objective is to understand two types of neural networks that can be use in manufacturing. Based on this project, two types of neural networks have been studied. The two types of the neural networks are Feedforward Backpropagation and Radial Basis. These two types of neural networks have good ability in pattern recognition and classification. The second objective of this project was also achieved.

Feedforward Backpropagation with tansig transfer function for hidden and output layer and also with ten numbers of neurons gives the best performance of the R value and lower error. The R value is the indication of the relationship between the outputs and the targets. The performance function used is mean squared error (MSE). Radial Basis gives significantly lower error compared to Feedforward Backpropagation and it takes less time training.

5.3 RECOMMENDATION

Some recommendation can be made to increase and improve this project. In improving the result if the network is not sufficiently accurate, the network can be initialized again and do the training again. Each time initialize a network, the network parameters will be different and might produce different solutions.

Another approach in improving the result is increase the number of the hidden neurons above 20. Larger numbers of neurons in the hidden layer give the network more flexibility because the network has more parameters it can optimize. If the hidden layer is too large, it would cause the problem under-characterized and the network must optimize more parameters than there are data vectors to constrain these parameters.

Besides that, the result can be improved with try or change different training functions. Sometimes this can produce better generalization capability. Another method is use additional training data to improve the result if the network is not sufficiently accurate. Providing additional data for the network can produce a network that generalizes well to new data but this need to consider for the processing time. When there is more additional data, the longer time taken to process the data.

For define the structure of the network, there is need to choose the most suitable activation function for the network. The training function also needs to choose correctly. The structure of the network architecture such as the hidden layers and neurons in each layer must be decided carefully. Networks are also sensitive to the numbers of neurons in their hidden layers. Too few neurons can lead to underfitting. Too many neurons can contribute to overfitting.

In improving the line recognition and for the robustness of the neural network, there is need to consider the orientation of the camera and illumination. The image needs to improve if there are too much noisy and too much illumination. For the image of the guideline, crossing guideline and junction guideline can be add for improving the robustness of the neural network.

REFERENCES

A. D'Amico, G. I. 2001. A Radial Basis Function Networks approch for the tracking problem of mobile robots. *International Conference on Advanced Intelligent Mechatronics Proceedings* (pp. 498-503). Como, Italy: IEEE.

Amit, K. 2009. *Neural networks: A requirement for intelligent systems*. Retrieved May 15, 2010, from Neuro AI Intellihent systems and neural networks: http://www.learnartificialneuralnetworks.com/references.html

Dong Sun Park, S. Y. 1999. Robot end-effector recognition using modular neural network for autonomous control. 2032-2036.

G.A. Borges, A. L. 1998. Characterization of a neural network-based trajectory recognition optical sensor for an automated guided vehicle. *Instrumentation and Measurement Technology Conference.* St. Paul, Minnesota, USA.

Janglova, D. 2004. Neural networks in mobile robot motion. *International Journal of Advanced Robotic Systems*, 15-22.

K. J. Hunt, D. S. 1992. Neural network for control study - a survey. 1083-1112.

Mark Hudson Beale, M. T. 2011. Neural Network Toolbox User's Guide. United States.

Matthews, P. G. 2010, May 24. *Median. mode, skewness and kurtosis in MS access*. Retrieved May 15, 2011, from Experts exchange: http://www.experts-exchange.com/Microsoft/Development/MS_Access/A_2529-Median-Mode-Skewness-and-Kurtosis-in-MS-Access.html

Mrinal Kanti Bhowmik, D. B. 2009. Clasiification of fused images using radial basis function neural network for human face recognition.

Mustafa Soylu, N. E. 2000. A self-organizing neural network approach for the single AGV routing problem. *European Journal of Operational Research*, 124-137.

Oky Dwi Nurhayati, D. A. 2011. Principal component analysis combined with first order statistical method for breast thermal images classification. *International Journal of Computer Science and Technology*, 12-18.

Pavon N., S. O. MLP neural networks to control mobile robot using multiple goal points. 395-398.

Sulaiman Sabikan, M. S. 2010. Vision-based automated guided vehicle for navigation and obstacle avoidance. *The second International Conference on Engineering and ICT.* Universiti Teknikal Malaysia Melaka (UTEM).

Symeonidis, K. 2000. Hand gesture recognition using neural network.

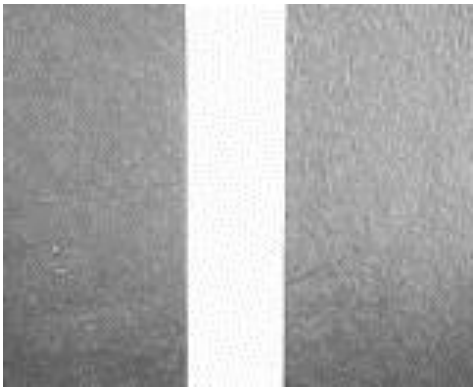
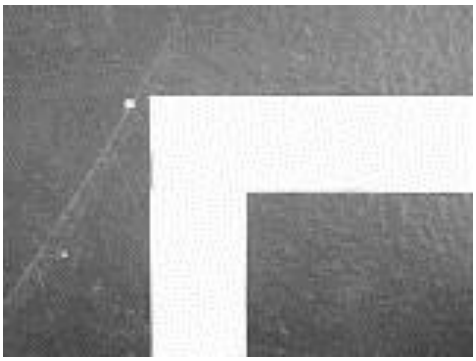
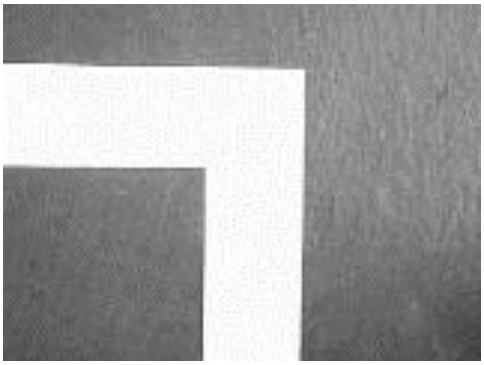
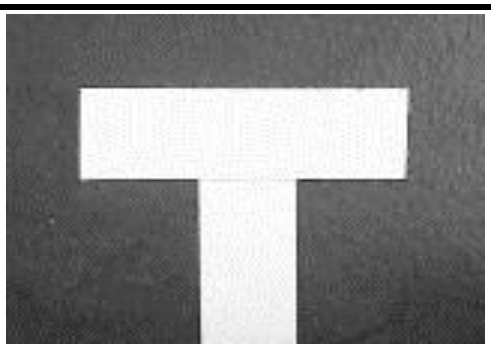
Tolson, S. R. 2011. A new formulation for feedforward neural networks. 1588-1598.

Vinicius M. de Oliveira, E. R. 2000. Feedforward control of a mobile robot using neural network. 3342-3347.

Xin Ma, W. L. 2005. LVQ neural network based target differentiation method for mobile robot. 680-685.

Yang, A. Z. 2003. Self-organizing behaviour of a multi-robot system by a neural network approch. *International Conference on Intelligent Robots and Systems* (pp. 1204-1208). Las Vegas, Nevada: IEEE.

APPENDIX A
Data of mean, variance, skewness and kurtosis of images

	Mean	Variance	Skewness	Kurtosis
 Straight	75	4.70E+04	12.62485218	1.86E+02
	75	6.18E+04	11.63830847	1.65E+02
	75	6.09E+04	11.60169962	1.64E+02
	75	4.05E+04	9.210651458	1.17E+02
	75	5.88E+04	12.46153747	1.82E+02
	75	5.71E+04	12.44068465	1.82E+02
	75	6.25E+04	11.46884835	1.61E+02
	75	5.31E+04	12.41856007	1.82E+02
	75	5.87E+04	11.65002095	1.65E+02
	75	6.48E+04	11.21113539	1.56E+02
 Right	75	4.08E+04	9.491299923	1.21E+02
	75	4.19E+04	9.230475033	1.14E+02
	75	4.43E+04	9.483165912	1.15E+02
	75	4.55E+04	9.719582934	1.22E+02
	75	5.15E+04	10.2626714	1.33E+02
	75	6.57E+04	12.22082015	1.77E+02
	75	4.00E+04	8.921103019	1.10E+02
	75	4.39E+04	8.561862879	1.02E+02
	75	4.01E+04	8.883753322	1.07E+02
	75	7.09E+04	12.68519168	1.86E+02
 Left	75	4.46E+04	10.39022679	1.35E+02
	75	3.73E+04	8.295582481	98.34061296
	75	5.16E+04	10.052764	1.32E+02
	75	4.29E+04	9.162574051	1.16E+02
	75	3.70E+04	8.049189097	93.36156958
	75	4.61E+04	9.931908576	1.30E+02
	75	4.45E+04	9.447837837	1.19E+02
	75	3.45E+04	7.371258274	79.51295967
	75	4.18E+04	8.397427229	99.62106461
	75	3.87E+04	8.95611901	1.11E+02
 Stop	75	4.74E+04	11.66197149	1.63E+02
	75	4.88E+04	9.650969195	1.25E+02
	75	5.17E+04	10.11116596	1.34E+02
	75	7.05E+04	11.85588578	1.69E+02
	75	6.55E+04	11.41434627	1.60E+02
	75	5.70E+04	10.64485163	1.44E+02
	75	5.15E+04	10.61523407	1.44E+02
	75	5.68E+04	11.46986011	1.62E+02
	75	4.78E+04	9.794646188	1.26E+02