

DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**CONTROLLING A NON-HOLONOMIC VEHICLE
VIA ARTIFICIAL NEURAL NETWORKS**

by
Aytaç GÖREN

November, 2007

IZMIR

CONTROLLING A NON-HOLONOMIC VEHICLE VIA ARTIFICIAL NEURAL NETWORKS

**A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in
Mechanical Engineering, Machine Theory and Dynamics Program**

**by
Aytaç GÖREN**

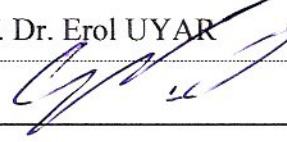
November, 2007

İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled “**CONTROLLING A NON-HOLONOMIC VEHICLE VIA ARTIFICIAL NEURAL NETWORKS**” completed by **AYTAÇ GÖREN** under supervision of **PROF. DR. EROL UYAR** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Erol UYAR



Supervisor

Yard. Doç Dr. Zeki KIRAL



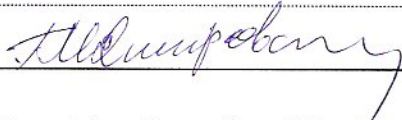
Thesis Committee Member

Yard. Doç.Dr. Zafer DİCLE



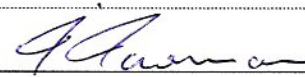
Thesis Committee Member

Prof. Dr. Georgi M. DIMIROVSKI



Examining Committee Member

Prof. Dr. İsmail H. TAVMAN



Examining Committee Member

Prof.Dr. Cahit HELVACI
Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

Firstly, I would like to thank to my supervisor Prof. Dr. Erol UYAR for supervising as flexible as possible so that I could have the chance of widening not only the scope of this research but also my research areas.

I would also like to thank Assist. Prof. Dr. Zeki KIRAL and Assist. Prof. Dr. Zafer DİCLE for their help with valuable suggestions and discussions during periodical meetings of this research and Prof. Dr. Georgi M. DIMIROVSKI for his valuable suggestions in expressing research results of DS-Theory which I presented in IFAC DECOM-TT '07.

I would also like to thank Faz Elektrik A.Ş. for supporting me by giving the base and motors of the model.

Special thanks are also extended to Osman KORKUT for his extraordinary effort of helping me in experiments, Cuma POLAT for building the mechanical model, my colleagues Levent ÇETIN and Özgün BAŞER for their encouragements and helping for finding solutions for problems, Necdet YILDIZ, Mustafa DAL, M. Rıfat AKAL for helping me in building the motor drivers, Abdullah ADIYAN, Eyüp KERVANCIOĞLU for helping me in building the 3D model and producing the real model, Prof. Dr - Ing. Ernst SCHNEIDER, Mrs. SCHNEIDER, P. VIEHHAUSER and Z. LEPOJEWITSCH for their encouragement in writing my thesis. I am still remembering them very clearly saying 'close your eyes and straight go!'

I want to dedicate my thesis to my wife Meltem GÖREN, my mother Gülsüm GÖREN and my father İsmail GÖREN for their unlimited patience through this project. Without their encouragements, it was not possible for me to motivate myself during hard working hours and days.

Aytaç GÖREN

CONTROLLING A NON-HOLONOMIC VEHICLE VIA ARTIFICIAL NEURAL NETWORKS

ABSTRACT

The use of learning autonomous robots is inevitable in modern production technologies. Modern industries require efficient production, précised measuring and robust control systems due to the hard competition of perfect product manufacturing whereas human wants more comfort in life. Various kinds of robot vehicles for various tasks have been developing increasingly not only in production industry but also in daily life.

In this research, a vehicle model with four wheels is built and equipped with actuators, different types of sensors, communication devices, data acquisition and control units in Automatic Control Laboratory of Mechanical Engineering Department of Dokuz Eylul University. On this developed autonomous wheeled mobile robot model, Dempster-Shafer evidence theory is tested at first step in means of sensor fusion for having more reliable data from sensors.

Fuzzy logic and most types of artificial neural networks architectures which are popular on autonomous mobile robots are explained starting with basic equations and control parameters, revealing the advantages of ANNs use on autonomous mobile robots in second chapter of this thesis. Chapter three covers the kinematic analysis of mobile robot model. It is the fourth chapter, in which the experimental data and results are demonstrated. The conclusion part, which is named as chapter five, is the part in which the results are evaluated in not only in means of technical or scientific research but also in means of daily life use.

Keywords: Artificial neural networks, autonomous mobile robots, wheeled mobile robots, non-holonomic vehicles, fuzzy logic, Dempster-Shafer Theory.

HOLONOMİK OLMAYAN ARAÇLARIN YAPAY SİNİR AĞLARI İLE KONTROLÜ

ÖZ

Öğrenebilen otonom robotların modern üretim teknolojilerinde kullanımı kaçınılmazdır. Modern endüstriler, mükemmel mamül üretim yarışı içerisinde hızlı ve verimli üretim, hassas ölçme ve sağlam kontrolü gerektirirken; insan da günlük hayatta daha fazla konfor aramaktadır. Değişik görevler için değişik çeşit robot araçlar sadece üretim endüstrisinde değil, günlük hayatta da hızla ortaya çıkmaktadır.

Bu çalışmada, dört tekerlekli bir araç modeli Dokuz Eylül Üniversitesi Makina Mühendisliği Bölümü Otomatik Kontrol Laboratuvarı'nda üretilmiş ve üzerine hareket elemanları, çeşitli algılayıcılar, iletişim cihazları, veri toplama ve kontrol üniteleri yerleştirilmiştir. Üretilen bu otonom tekerlekli gezgin robot modelinde ilk aşamada sensör füzyonu ve algılayıcılardan daha güvenilir veri alma konusunda Dempster-Shafer teoremi denenmiştir.

Otonom gezgin robotlarda sıklıkla kullanılan bulanık mantık ve yapay sinir ağları mimarileri temel denklem ve kontrol parametrelerinden başlanılarak, gezgin robotlarda YSA kullanımının avantajları ortaya konularak bu tezin ikinci bölümünde anlatılmıştır. Üçüncü bölümde gezgin robot modelinin kinematik analizine yer verilmiştir. Dördüncü bölümde ise deneysel veriler ve sonuçlar gösterilmiştir. Bölüm beş olarak adlandırılan sonuç bölümünde sadece teknik ve bilimsel araştırma açısından değil, günlük hayatta kullanım açısından da değerlendirilmiştir.

Anahtar Sözcükler: Yapay sinir ağları, otonom gezgin robotlar, tekerlekli gezgin robotlar, holonomic olmayan araçlar, bulanık mantık, Dempster-Shafer Teoremi.

CONTENTS

	Page
THESIS EXAMINATION RESULT FORM	ii
ACKNOWLEDGEMENTS.....	iii
ABSTRACT.....	iv
ÖZ	v
CHAPTER ONE – INTRODUCTION.....	1
1.1 Introduction	1
1.2 Artificial Neural Networks	2
1.2.1 Types of ANNs	4
1.2.2 Learning Types in ANNs	5
1.2.2.1 Unsupervised Learning	5
1.2.2.2 Supervised Learning	5
1.2.2.3 Reinforcement Learning	5
1.2. 3. Autonomous Mobile Robots	6
1.2. 3.1 Autonomous Mobile Robot Model	8
1.2. 4. ANNs Applications in Autonomous Mobile Robots	9
1.2.4.1 Application of Sensor Data Directly to ANNs.....	10
1.2. 4.2 Image Processing with ANNs on AMRs	11
1.2.4.3. Approximation to Optimal Path Following and Obstacle Avoidance ...	12
CHAPTER TWO -FUZZY LOGIC AND ARTIFICIAL NEURAL NETWORKS ...	15
2.1 Dempster – Shafer Theory and Sensor Fusion	15
2.2 Artificial Neural Networks	16
2.2.1 Types of Artificial Neural Networks	20
2.2.2 Learning Types in Artificial Neural Networks	20
2.2.2.1 Unsupervised Learning	20
2.2.2.2 Supervised Learning	20
2.2.2.3 Reinforcement Learning	21
2.2.3 Types of Connections in Artificial Neural Networks	21
2.2.4 Architectures and Learning Paradigms in Artificial Neural Networks	23
2.2.4.1 Hopfield Model	23

2.2.4.2	The Perceptron Model	24
2.2.4.3	Cellular Neural Networks Model	25
2.2.4.4	Winner Takes All Model	27
2.2.4.5	Back-Propagation Model	27
2.2.4.6	McCulloch-Pitts Model	28
2.2.4.7	Adaptive Resonance Theory (ART) Model	29
2.2.4.8	Radial Basis Function (RBF) Model	30
2.2.4.9	Widrow-Hoff Model (ADALINE and MADALINE)	32
2.2.4.10	Kohonen (Self Organizing Map- SOM) Model	33
2.2.4.11	Learning Vector Quantization (LVQ) Model	34
2.2.4.12	Boltzmann Machines	36
2.2.4.13	Hebbian Learning Rule	38
2.2.4.14	Principal Component Analysis	39
2.3	Fuzzy Logic	40
2.3.1	Introduction	40
2.3.2	Membership Function	40
2.3.3	Fuzzy Logic Control.....	41
2.3.3.1	Fuzzy Logic Operations	41
2.3.3.2	Control	43
CHAPTER THREE AUTONOMOUS NON-HOLONOMIC MOBILE ROBOTS..		48
3.1	Non-Holonomic Vehicles	48
3.2	Autonomous Wheeled Mobile Robots	49
3.3	Dead Reckoning	53
3.4	Architecture of Autonomous Wheeled Mobile Robots	54
3.4.1	Mobile Robot Navigation	55
3.5	Braitenberg Vehicles	56
CHAPTER FOUR SYSTEM AND EXPERIMENTS		58
4.1	Kinematics of the Experimental Mobile Robot	58
4.2	Experimental System Model	60
4.2.1	Ultrasonic Sensors for Navigation	61
4.2.1.1	Ultrasonic Sensors Installation	62
4.2.2	Wireless Communications on Model	64

4.2.2.1 Radio Frequency Modems	64
4.2.2.2 Bluetooth.....	65
4.2.2.3 Wireless Network	66
4.2.3 Motors and Drivers	67
4.2.3.1 Motors of the Research Model.....	67
4.2.3.2 Motor Drivers	68
4.2.4 Controllers and Input – Output Cards	69
4.2.4.1 PC104: Embedded Computer on the Mobile Robot Model	69
4.2.4.2 Microcontrollers on the Mobile Robot Model.....	71
4.3 Block Diagrams	72
4.3.1 First Models of the Robot	72
4.3.2 Control of the System	72
4.4 Dempster – Shafer Evidence Theory Experiments	74
4.5 Fuzzy Logic Experiments (Find Target)	77
4.5.1 Fuzzy Logic Controller Experiment 1	77
4.5.2 Fuzzy Logic Controller Experiment 2	81
4.5.3 Fuzzy Logic Controller Experiment 3	82
4.5.4 Fuzzy Logic Controller Experiment 4	85
4.5.5 Fuzzy Logic Controller Experiment 5	87
4.6 Artificial Neural Networks Experiments (Find Empty Space)	89
4.6.1 ANN and Fuzzy Logic Combined Experiments	91
CHAPTER FIVE - CONCLUSION	103
5.1 Overview	103
5.2 Conclusions About Control Techniques and Future Work	103
REFERENCES	106
APPENDICES	110
Appendix 1 – Nomenclature	110
Appendix 2 – Abbreviations	112
Appendix 3 – Definitions in Computer Networks	114
Appendix 4 – Schematics	119
Appendix 5 – Programs	129

CHAPTER ONE

INTRODUCTION

1.1 Introduction

It is the starting point of a hard target to build a robot that can decide. However, it is harder to build a robot that can sense environment and decide. The aim of most of the autonomous mobile robots is to move towards the target without any help from a man and without hitting any obstacle around using its own decision mechanism. Since the parameters which affect decision mechanism are variable, it is a great advantage for the control unit of an AMR to have learning skill.

The most important advantage of the use of artificial neural networks on AMR is the increase of the flexibility of adaptation of control algorithm in varying environment conditions. In other words, the main advantage of using ANN in mobile robots is the ability of developing solutions for changing environment and limitations problem. For instance, robot can complete its tasks in not only the predefined environments and conditions or in the workspace that is recorded in the memory but also in totally changed conditions without using huge environmental data in memory. In solving the two main problem of finding free space in workspace and finding the shortest way to target problem, many different ANN techniques are being used (Janglova, 2004; Hamdi, A.A. & Al-Zorkany, M.A., 2004).

ANN analysis and solutions techniques have been used in industrial applications in last years. Some application fields are: Production planning, optimization of production for increasing productivity in various conditions with changing parameters, process monitoring, modeling and control researches (Lennox, B., Montague, G. A., Frith, A. M., Gent, C., Bevan, V., 2001).

1.2 Artificial Neural Networks

Human nerve cell consists of four parts: 1. Dendrite, the part which accepts inputs, 2. Soma, the part that process inputs, 3. Axon, the part which changes inputs into outputs, 4. Synapse, the part that carries the data to the next cell (Figure 1.1).

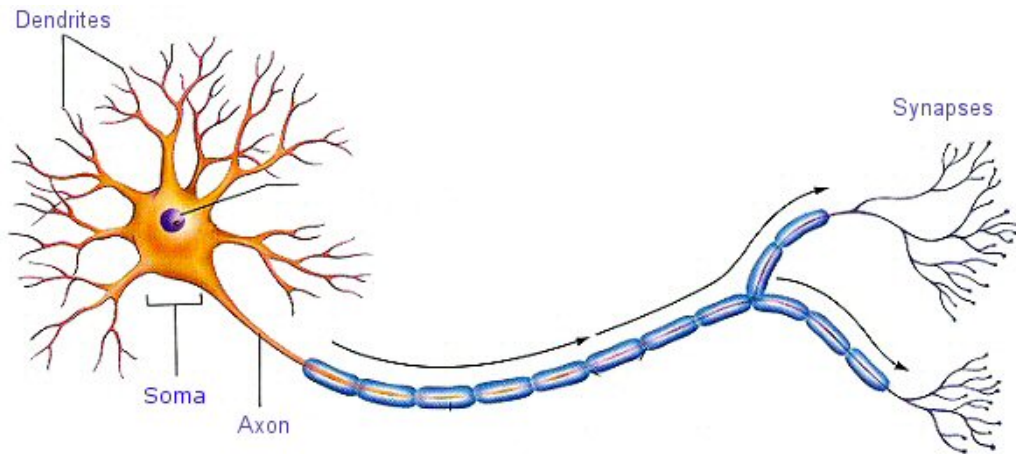


Figure 1.1 Parts of a biological nerve cell (neuron).

Artificial Neural Networks (ANN) are the control methods those are inspired from biological neuron and its parts. Each input (X_1, X_2, \dots, X_n) is connected to a certain cell or a cell (s) which is selected after process with a weight that defines the connection strength or the influence of a neuron to the next one. Weights (W_1, W_2, \dots, W_n) change to optimal values in program cycles during training period. Thus, the convenient output(s) is (are) obtained. In other words, the ANN learned. In equations below you may see the output of ANN, output with feedback, error and updated weight value respectively. In equations, d_i is the desired value, y_i is the output value, η is the learning rate and δ_i is the local slope.

$$f(x) = \sum_{i=0}^n x_i \cdot w_i ; \quad i=0, 1, 2, \dots, N \quad (1.1)$$

$$y_i = f\left(\sum_{i=1}^n (w_i x_i) - \theta_i\right) ; \quad i=0, 1, 2, \dots, N \quad (1.2)$$

$$e_i(n) = d_i(n) - y_i(n) \quad (1.3)$$

$$w_{ij}(n+1) = w_{ij}(n) + \eta \delta_i(n) x_j(n) \quad (1.4)$$

ANNs models are generally simple mathematical models those define $f: X \rightarrow Y$.

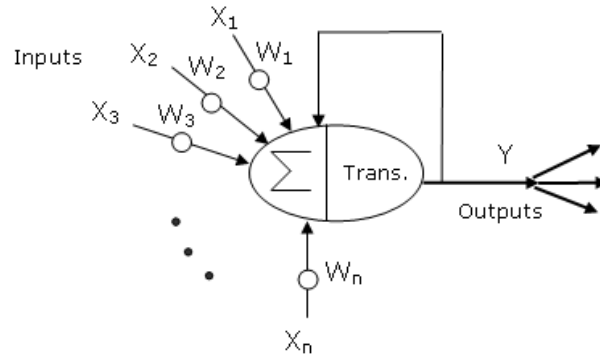


Figure 1.2 Artificial Neuron 1(feedback)

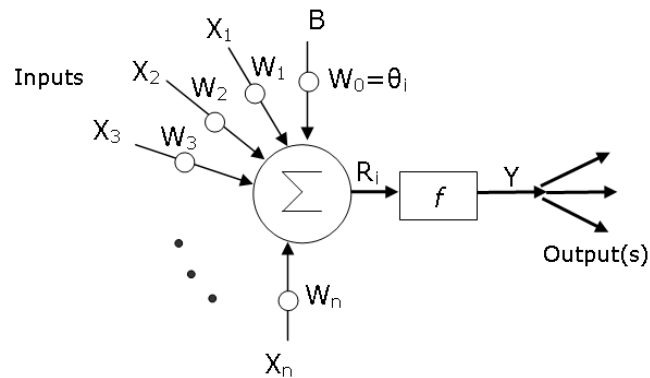


Figure 1.3 Artificial Neuron 2

ANNs have also disadvantages in applications. Time need in training period, more upgraded hardware need than generally for processing, more complex than classical methods are some clues which show that ANNs methods are not the optimal solution for every control problem. Some application fields of ANNs in AMR are: Classification, image and sound data processing, non-linear mapping, optimization, coordination of robot parts and associative memory applications.

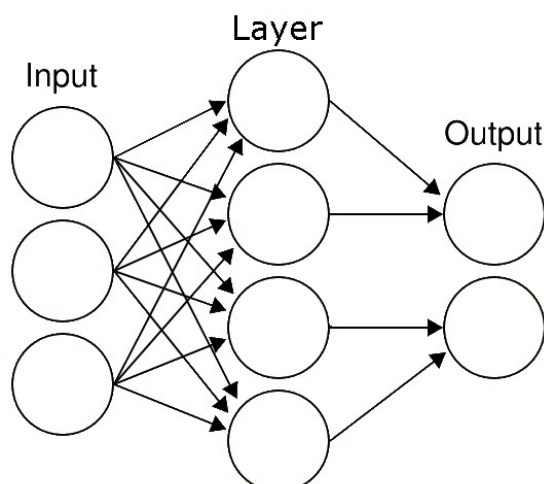


Figure 1.4 Artificial Neural Networks

1.2.1 Types of ANNs

The two main criteria for classification of ANNs are whether the ANN has feedforward or feedback and whether the ANN is supervised or unsupervised (Mori, H., Tamaru, Y., Tsuzuki, S. 1992). Some ANN algorithm types can be seen in Table 1.1

Table 4.1. Classification of ANN algorithms.

	Recurrent	Feedforward
Unsupervised	ART (Adaptive Resonance Theory), Hopfield, Bidirectional associative memory (BAM), Trilateral associative memory (TAM), Boltzmann Machine (can be also supervised),...	Linear associative memory, Fuzzy Associative Memory (FAM), LVQ, CPN, SOM, ...
Supervised	Fuzzy Cognitive Map (FCM), Boltzmann Machine (can be also unsupervised), ...	Backpropagation (BP), Adaline, Perceptron, ...

1.2.2 Learning Types in ANNs

1.2.2.1 Unsupervised Learning

It is learning with doing. Without any interference, the neurons in hidden layer organize themselves. No sample output according to relevant input is given to the network.

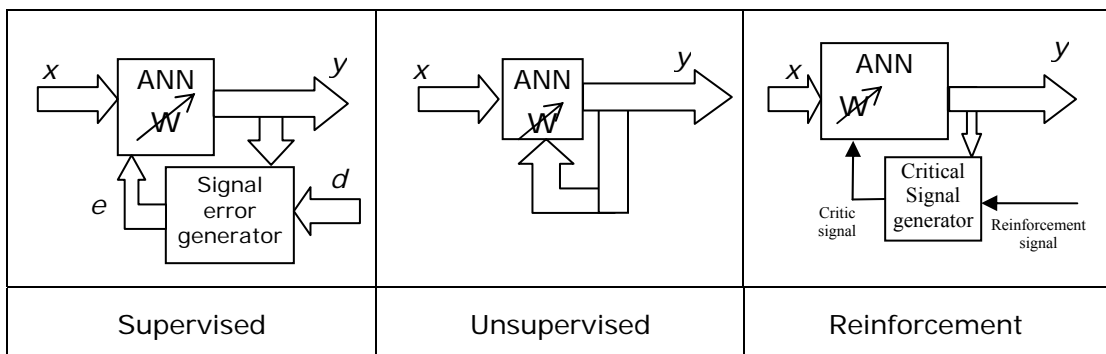
1.2.2.2 Supervised Learning

If the desired output value is y and network output is o , the difference between y and o is the error value. Error value is used to update the weights in supervised learning. Desired output values are connected to input values and are given as vectors to the network. These sample values matrix is called as training set.

1.2.2.3 Reinforcement Learning

It is also a type of supervised learning. Neurons in hidden layer are random connected to each other. Approximation to solution of the problem is the main evaluation criteria. There can be a training data set or a performance observer.

Table 1.2 Learning types in ANN



In addition to the ANN types above, in some problem solutions supervised and unsupervised learning are combined (Janglova, 2004).

1.2. 3. Autonomous Mobile Robots

An autonomous mobile robot is a robot which is moving and changing its workspace aiming to complete its tasks in limitations of the rules given or the rules it develops. The mobile robots which are studied in this research are wheeled mobile robots.



Figure1.5 Some AMR models in this research.

In kinematic analysis of mobile robots, there are four main differences for kinematic analysis of robot manipulators (Muir, 1986).

- Stationary manipulators only form closed chains when they are contact with fixed objects whereas; wheeled mobile robots form many closed chains at the same time.
- The contact between a wheel and plane forms a higher-pair, but stationary manipulators contain only lower-pair joints.

- In wheeled mobile robots, only some degrees of freedom of a wheel are actuated. However, all DOFs of each joint of a stationary manipulator has at least one actuator.
- Each joint in stationary manipulator has position and velocity sensors. In wheeled mobile robots, only some degrees of freedom of a wheel have position or velocity sensors.

Left and right motor velocity equations of a four wheeled mobile robot whose two wheels are actuated can be seen in figure below.

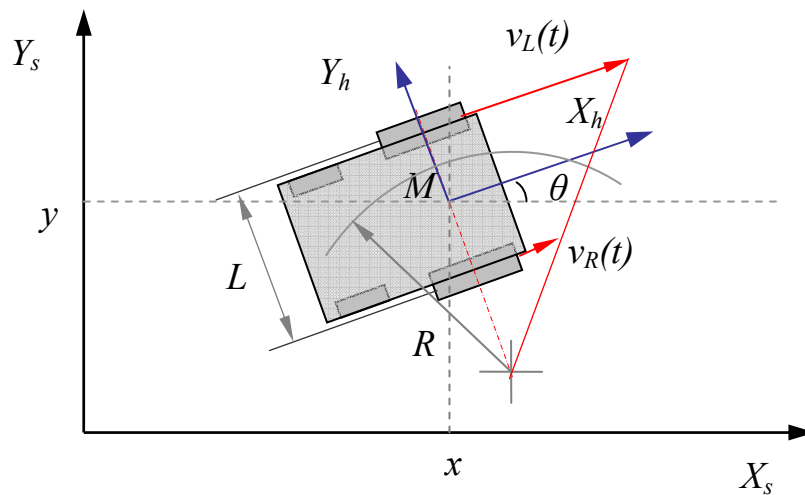


Figure.1.6 Velocities of left and right motors of the mobile robot model in this research.

r : Dynamic radius of wheel [m],

R : Radius of curvature, [m],

L : Distance between the middles of two front wheels [m],

v : Linear velocity of the mobile robot, [m/s],

$v_R(t)$: Linear velocity of the right front wheel, [m/s],

$v_L(t)$: Linear velocity of the left front wheel, [m/s],

θ : Heading angle, [rad],

$w(t)$: Angular velocity in z coordinate of the mobile robot, [rad/s],

$\{X_h, Y_h\}$: Moving coordinate axes,

$\{X_s, Y_s\}$: Stationary coordinate axes.,

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos \theta(t) & 0 \\ \sin \theta(t) & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v(t) \\ w(t) \end{bmatrix} \quad (1.5)$$

1.2. 3.1 Autonomous Mobile Robot Model

Autonomous mobile robots in this research are heeled mobile robots which can perform their tasks without guidance of human.

If the structure of a mobile robot is studied many external or internal sensors and sensor data, lots of priority levels in algorithm regarding the task priorities should be studied. But generally, interaction of mobile robot with three concepts can be seen. These are user, object and environment (Kopacek, 2006). In Figure 1.7, structure of an autonomous mobile robot can be seen. Extracting some blocks, this structure can be used also for an open loop controlled mobile robot. Of course when it is open loop controlled, it is not classified as autonomous. In Table 1.3, the paradigm of an autonomous mobile robot movement is demonstrated (Murphy, R.R., 2000).

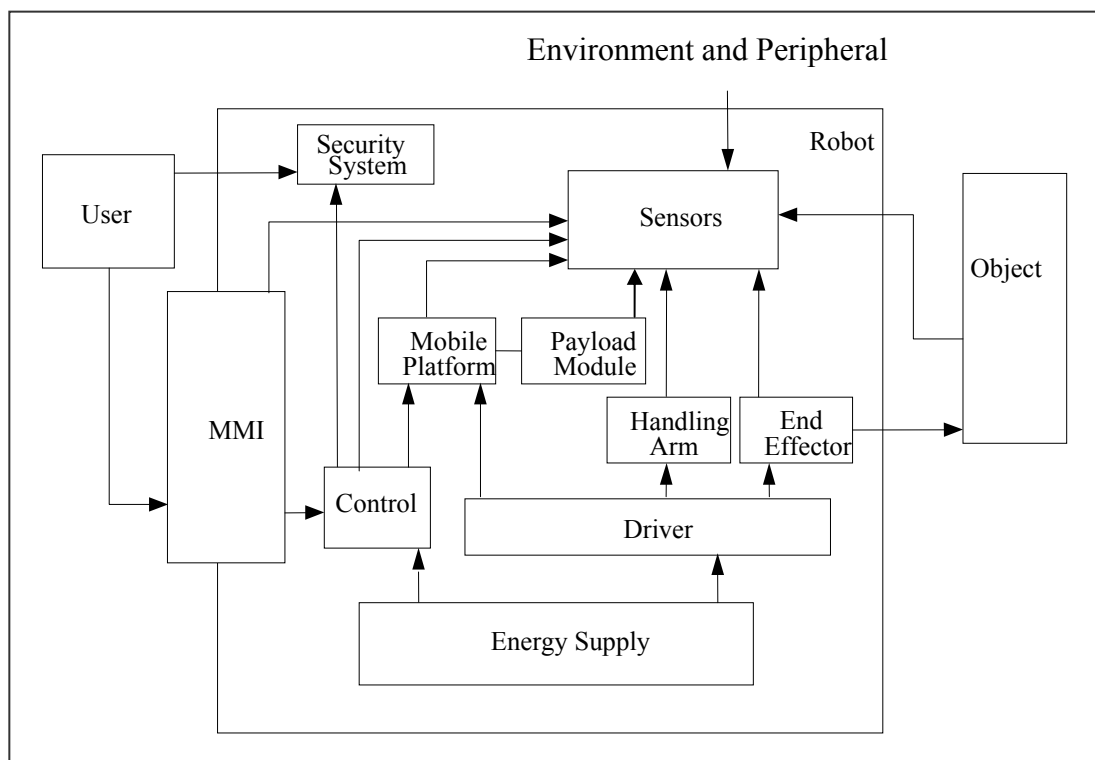


Figure 1.7 Structure of a mobile robot.

Table 1.3. Movement mechanism of an autonomous mobile robot.

Movement	Input	Output
Sense	Sensors	Sensor Data
Plan	Data (sensed and/or logic)	Commands
Do	Sensed data or commands	Actuators

1.2. 4. ANNs Applications in Autonomous Mobile Robots

Starting point of ANNs applications study can be simulation programs. MatLab NN toolbox and NN blockset, BrainCom, Stuttgart University's SNNS, EasyNN-Plus are some examples. In simulations or in programming basic steps to ANN study is:

- Determination of number of inputs, initial values of weights (some types do not need), number and types of outputs, number of layers.
- Training period.
- When the ANN achieves the desired values, it is tested.
- If it is convenient, it is used. If it is not, training and testing periods are repeated. If it is needed convenient changes are made and steps are repeated.

Most researches need more than simulations. In this case, if there is a NN library in the programming language, which will be used, it can be an advantage. Fast Artificial Neural Networks library is one of the widespread libraries. It not only has different versions for different programming languages (GCC, C++, .NET, Perl, PHP ...), but also has different versions for different operating systems (Linux, MacOS, POSIX, Windows ...). In figure below, FANN Explorer, the interface for FANN can be seen.

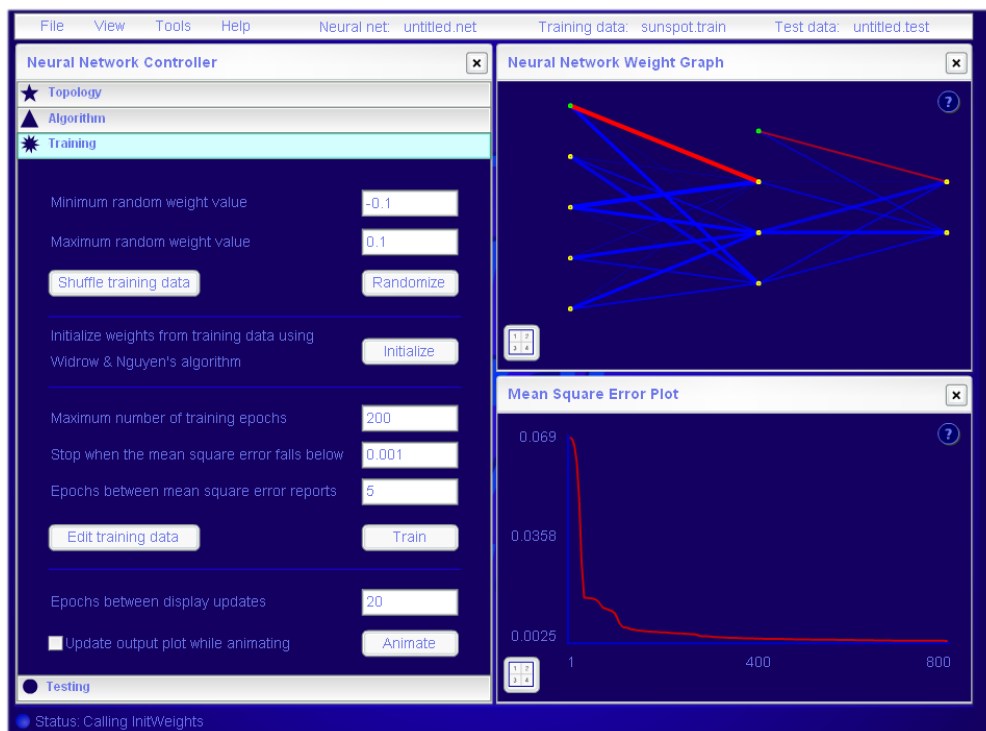


Figure 1.8 Interface of FANN library and FANN explorer

1.2.4.1 Application of Sensor Data Directly to ANNs

ANNs can be applied in all stages of movement paradigm of AMR. For instance, in sense stage ANNs can be used to have more reliable data from sensors; in planning stage, they can be used to find free space or path or for mapping and in doing stage, they can be used to solve non-linearity problems or finding optimal operating conditions.

In sensing stage, for the purpose of having reliable data, some theories like, Dempster-Shafer can be applied (Gören, A., Uyar, E., Dicle, Z., 2007). In Figure 1.9, dividing regions as for having data from sensors with priority levels can be seen. ANNs have some advantages in applications in sensing stage based on ‘behaving more sensible for the sensor that is stimulated before’. Each sensor data can be accepted as each neuron input. Taking into consideration of heading angle, limitations and target coordinates, weights can increase or decrease. In result of this, AMR can act more optimal. In sensing stage, ANNs also can be used to have reliable data.

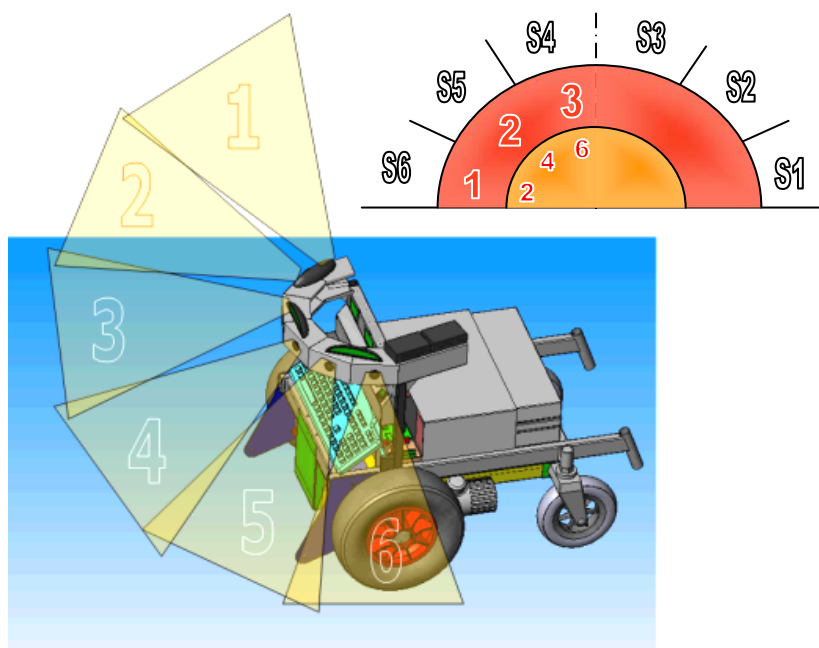


Figure 1.9 Sensor installations and regions on model

1.2. 4.2 Image Processing with ANNs on AMRs

Realtime image processing is one of the popular techniques which are used on mobile robots. AMR can detect an object from environment, map surround, classify objects, head towards a target, detect motion, detect the distance from an object or associate information with an object via image processing, On the other hand, realtime image processing code in control algorithm should be fast and should not make the behavior of the robot clumsy. In this case, *cellular neural networks* can be a solution method. Instead of taking into consideration of the whole data in an image, using just some important data in a region of an image the same result can be found. CNN (*cellular neural networks*) use this information. Using CNN, control algorithm becomes faster and AMR does not need advanced control unit hardware. So, realtime image processing on an AMR can be processed without high cost of control unit and without any slowness that is caused by image processing. Linux operating systems (Slackware, RT Linux, Debian, Fedora, etc...) with realtime kernel and ‘comedi drivers’ can be a very effective choice not only for low cost realtime control and also for developing open source codes.

Structure of CNN and definition as equations can be seen below.

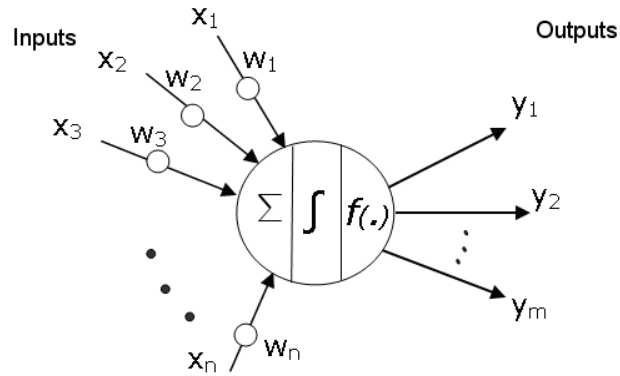


Figure 1.10. CNN Structure (Sum, Dynamic part and activation).

$$y_k = \frac{1}{2} (|x_k + 1| - |x_k - 1|) \quad (1.6)$$

$$\dot{x}_1 = -x_1 + \sum_{k \in N} a_k y_k + \sum_{k \in N} b_k u_k + z \quad (1.7)$$

$$\dot{X} = -X + A * y(X) + B * u + I \quad (1.8)$$

1.2.4.3. Approximation to Optimal Path Following and Obstacle Avoidance

One part of researches on WMR is finding the optimal path. Using classical methods or Fuzzy Logic in AMR control isn't the optimum solution especially in adaptation to different environments. Path to the target can be quite longer, movements of AMR may have discontinuity or because of long rule lists, the decision mechanism, consequently, the robot can be slow.

If it is considered that the velocities of the wheels of WMRs is simply dependent to the path following function of the robot. Therefore, a discontinuity on the path, which means also a discontinuity in velocity functions and also means high acceleration and deceleration values. In Figure 1.11, types of path curves can be seen. Equation 1.9, 1.10, 1.11 and 1.12 denote velocities of left and right side motors.

$r(s)$, is the radius of curvature at s and $\kappa(s)$ is the curvature at s . θ is the angle between the stationary coordinate axis and the tangent of the curve at point P.

$$\kappa(s) = r(s)^{-1} \quad (1.9)$$

$$v_L = v \left(1 - \frac{L \cdot \kappa(s)}{2} \right) \quad (1.10)$$

$$v_R = v \left(1 + \frac{L \cdot \kappa(s)}{2} \right) \quad (1.11)$$

$$v_R = v \frac{r(s) + \frac{L}{2}}{r(s)} \quad (1.12)$$

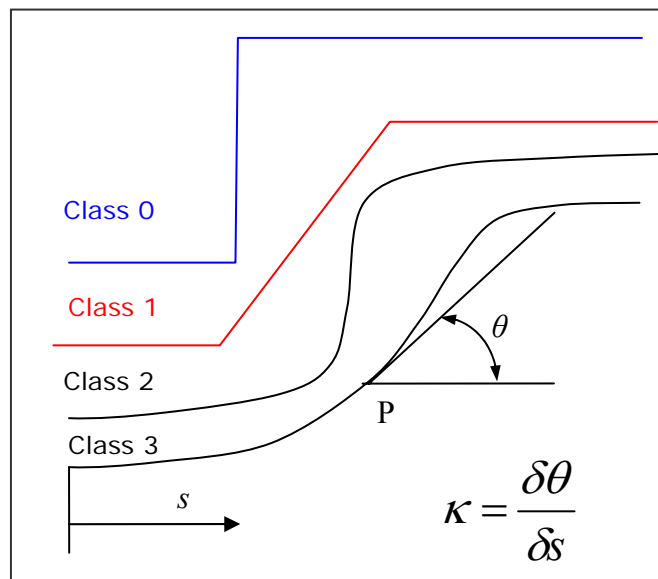


Figure 1.11 Class 0, class 1, class 2 and class 3 curve types.

In Figure 1.11, class 0 type curve is a discontinuous curve whereas a continuous curve is in class 1. In addition to continuity of the curve, if the direction tangent is also continuous it is a class 2 type curve. And in addition to all, if it has also curvature continuity, it is a class 3 type curve. Class 3 type curve is the optimal path following curve in most of the problems. And it is hard for an AMR to act in a class 3 type ways, especially in not known environments. However, using modern control

techniques like ANNs in AMR, the path for the robot can be smoother. Another point that has to be considered is that the performance criterion for an AMR is quite different than a stationary manipulator. Path or function following of a stationary manipulator is generally considers how accurate the ideal theoretical curve is followed whereas path following for an AMR generally means how successful, fast, optimal, accurate and how intelligent the tasks are completed. ANNs use is important for this reason.

In this research, some ANNs usage applications in AMR is explained with basic equations and control parameters revealing the advantages of ANNs use on autonomous mobile robots.

CHAPTER TWO
FUZZY LOGIC AND ARTIFICIAL NEURAL NETWORKS

2.1 Dempster – Shafer Theory and Sensor Fusion

Let X is universal set and $P(X)$ is the set of all possible subsets of X , including the empty set, \emptyset , in other words the power set. The mass of the empty set is zero;

$$m(\emptyset)=0 \tag{2.1}$$

$$1 = \sum_{A \in P(X)} m(A) \tag{2.2}$$

The mass of a given member of the power set, A , expresses the proportion of all relevant and available evidence that supports the claim that the actual state belongs to A but to no particular subset of A and denoted by , $m(A)$.

From the mass assignments, the upper and lower bounds of a probability interval can be defined. This interval is bounded by two non-additive continuous measures called *belief* (or support) and *plausibility*:

$$pl(A) = \sum_{B|B \cap A \neq \emptyset} m(B) \tag{2.3}$$

The sum of all the masses of subsets of the set of interest is the belief of a set and denoted by $bel(A)$.

$$bel(A) = \sum_{B|B \subseteq A} m(B) \tag{2.4}$$

Whereas the plausibility is the sum of all the masses of the sets that intersect the set of interest:

$$pl(A) = \sum_{B|B \cap A \neq \emptyset} m(B) \tag{2.5}$$

$$pl(A) = 1 - bel(\bar{A}) \quad (2.6)$$

Dempster's rule of combination is a generalization of Bayes rule that emphasizes the agreement between multiple sources and ignores all the conflicting evidence through a normalization factor.

$$m_{1,2}(\emptyset) = 0 \quad (2.7)$$

$$m_{1,2}(A) = \frac{1}{1 - K} \sum_{B \cap C = A \neq \emptyset} m_1(B)m_2(C) \quad (2.8)$$

$$K = \sum_{B \cap C = \emptyset} m_1(B)m_2(C) \quad (2.9)$$

In equations 8 and 9, K is a measure of the amount of conflict between the two mass sets whereas the normalization factor $(1-K)$, has the effect of completely ignoring conflict and attributing any mass associated with conflict to the null set.

To have more reliable data from sensors, sometimes sensory data or data derived from sensory data from disparate sources are combined. The reason is that resulting information is in some sense more accurate or more dependable than would be possible when these sources were used individually. This is called sensor fusion.

2.2 Artificial Neural Networks

For many scientists, artificial neural networks research started in 1943, when Warren McCulloch, a neurophysiologist, and a young mathematician, Walter Pitts, wrote a paper on how neurons might work and modeled a simple neural network with electrical circuits. Donald Hebb wrote *The Organization of Behavior* in 1949. The important point in this book was pointing out that neural pathways are strengthened each time that they are used. Nathaniel Rochester from the IBM research laboratories led the first effort to simulate a neural network in 1950's. But,

this attempt failed. . In 1956 the Dartmouth Summer Research Project on Artificial Intelligence provided a boost to both artificial intelligence and neural networks. In the years following the Dartmouth Project, John von Neumann suggested imitating simple neuron functions by using telegraph relays or vacuum tubes. Frank Rosenblatt, a neuro-biologist of Cornell, began work on the Perceptron. He was intrigued with the operation of the eye of a fly. Much of the processing which tells a fly to flee is done in its eye. The Perceptron, which resulted from this research, was built in hardware and is the oldest neural network still in use today. A single-layer perceptron was found to be useful in classifying a continuous-valued set of inputs into one of two classes. The perceptron computes a weighted sum of the inputs, subtracts a threshold, and passes one of two possible values out as the result. Unfortunately, the perceptron is limited and was proven as such during the "disillusioned years" in Marvin Minsky and Seymour Papert's 1969 book *Perceptrons*. It is 1959 that Bernard Widrow and Marcian Hoff developed models and called ADALINE and MADALINE (Multiple ADaptive LINear Elements). Eliminating echoes on phone lines, MADALINE was the first neural network to be applied to a real world problem. But, until John Hopfield of Caltech presented a paper to the national Academy of Sciences and Teuva Kohonen presented *Self-Organizing Maps* in 1982, the years passed without any success in application, but just frustration because of the insufficient computers. Hopfield's approach was not to simply model brains but to create useful devices. (History of Neural Networks, n.d.).

Inspiration of Artificial Neural Networks research is the human brain. Similar to biological neural networks, simple processing elements are called neurons. Each neuron is connected to certain of its neighbors with varying coefficients of connectivity. These coefficients are called as weights and represent the strengths of these connections. The most popular advantage for artificial neural networks is learning. Learning is accomplished by adjusting these weights to cause the overall network to output appropriate results. Collective and synergistic computation, asynchronous operation, robustness are other characteristics of ANNs. Intelligent machines, parallel processing, distributed computing, learning, generalization,

adaptation are some terms which are popular and mostly related with artificial neural networks.

Biological neuron consists of four parts:

- i. Dendrite, the part which accepts inputs,
- ii. Soma, the part that process inputs,
- iii. Axon, the part which changes inputs into outputs,
- iv. Synapse, the part that carries the data to the next cell (Figure 2.1).

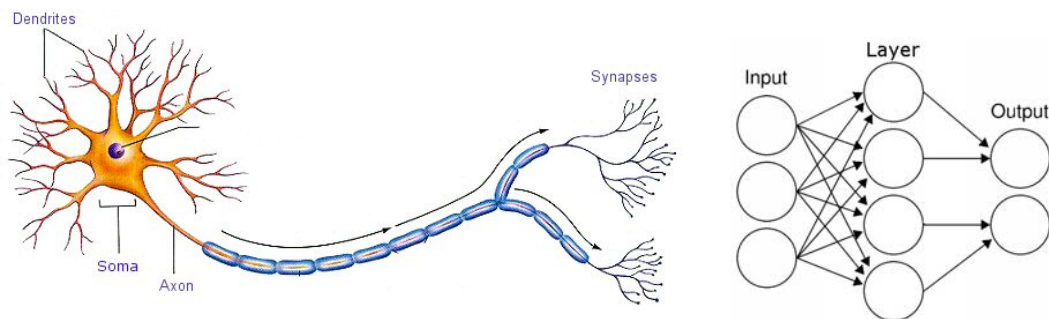


Figure 2.1. Parts of biological nerve cell (*The Major Structures of the Neuron*, n.d.) and artificial neurons.

If we compare the parts of biological and artificial neurons, the dendrite in biological neuron is input in artificial neuron, synaptic efficacy is weight, excitation level is noise (u) and signal in biological neuron is output in artificial neuron (Dohnal, V., Kuca, K. & Jun, D., 2005).

Each input (X_1, X_2, \dots, X_n) is connected to a certain cell or a cell (s) which is selected after process with a weight that defines the connection strength or the influence of a neuron to the next one. Weights (W_1, W_2, \dots, W_n) change to optimal values in program cycles during training period. Thus, the convenient output(s) is (are) obtained. In other words, the ANN learned. In equations below you may see the output of ANN, output with feedback, error and updated weight value respectively. In equations, d_i is the desired value, y_i is the output value, η is the learning rate and δ_i is the local slope.

$$f(x) = \sum_{i=0}^n x_i \cdot w_i; \quad i = 0, 1, 2, \dots, N \quad (2.10)$$

$$y_i = f\left(\sum_{i=1}^n (w_i x_i) - \theta_i\right); \quad i = 0, 1, 2, \dots, N \quad (2.11)$$

$$e_i(n) = d_i(n) - y_i(n) \quad (2.12)$$

$$w_{ij}(n+1) = w_{ij}(n) + \eta \delta_i(n) x_j(n) \quad (2.13)$$

ANNs models are generally simple mathematical models those define $f: X \rightarrow Y$. In Figures 2.2 and Figure 2.3 you may see examples of structures of artificial neural networks.

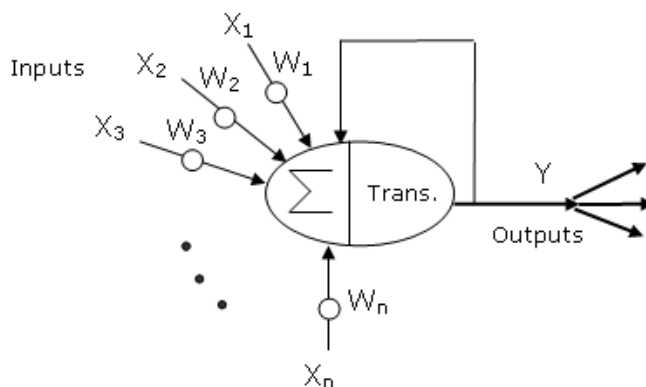


Figure 2.2 Artificial neuron model 1(feedback)

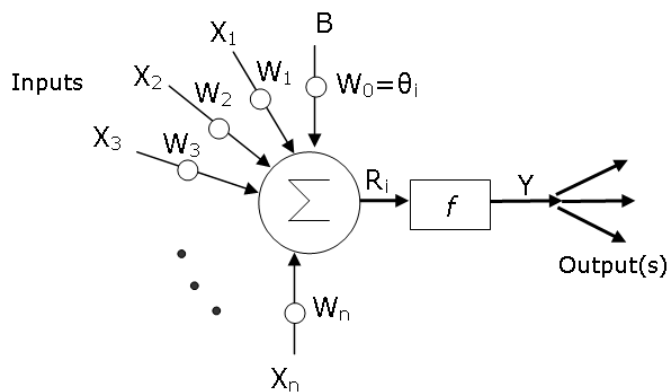


Figure 2.3 Artificial neuron model 2

2.2.1 Types of Artificial Neural Networks

The two main criteria for classification of ANNs are whether the ANN has feedforward or recurrent and whether the ANN is supervised or unsupervised (Mori, H., Tamaru, Y., Tsuzuki, S. 1992). Some ANN algorithm types can be seen in Table 2.1.

Table 2.1 Classification of algorithms of ANNs .

	Recurrent	Feedforward
Unsupervised	ART (Adaptive Resonance Theory), Hopfield, Bidirectional associative memory (BAM), Trilateral associative memory (TAM), Boltzmann Machine (can be also supervised),...	Linear associative memory, Fuzzy Associative Memory (FAM), LVQ, CPN, SOM, ...
Supervised	Fuzzy Cognitive Map (FCM), Boltzmann Machine (can be also unsupervised), ...	Backpropagation (BP), Adaline, Perceptron, ...

2.2.2 Learning Types in Artificial Neural Networks

2.2.2.1 Unsupervised Learning

It is learning with doing. Without any interference, the neurons in hidden layer organize themselves. No sample output according to relevant input is given to the network.

2.2.2.2 Supervised Learning

If the desired output value is d and network output is y , the difference between d and y is e , the error value. Error value is used to update the weights in supervised learning. Desired output values are connected to input values and are given as vectors to the network. These sample values matrix is called as training set.

2.2.2.3 Reinforcement Learning

It is also a type of supervised learning. Neurons in hidden layer are random connected to each other. Approximation to solution of the problem is the main evaluation criteria. There can be a training data set or a performance observer. Reinforcement learning and unsupervised learning have the relative disadvantage of slowness and inefficiency relying on random shuffling to find the proper weights.

In addition to the ANN types above, in some problem solutions supervised and unsupervised learning are combined (Janglova, 2004). For some literature, there are some other learning types in addition to the three types above. Some of these are: Associative recall, competitive learning, delta rule, Hebbian learning, gradient descend rule (Kartalopoulos, 1996, Şenol, 2002).

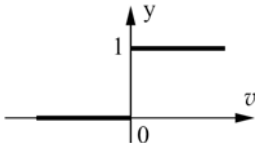
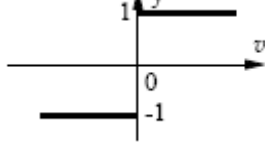
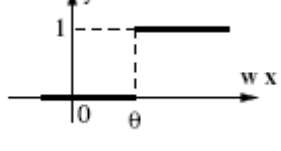
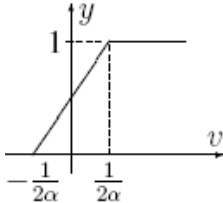
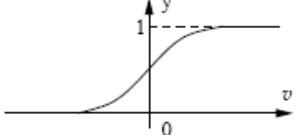
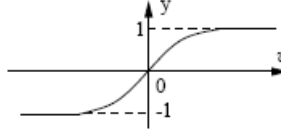
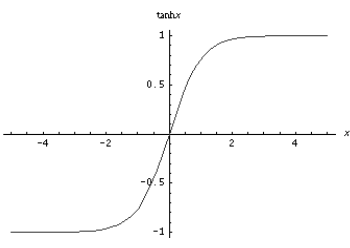
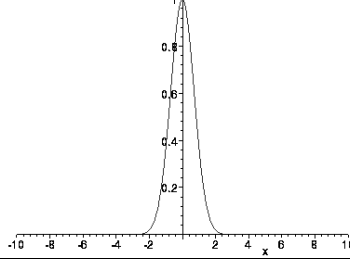
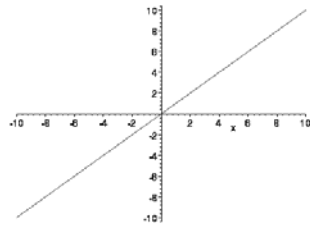
2.2.3 Types of Connections in Artificial Neural Networks

Connections between layers in ANNs can be fully or partially, feed forwarded, bi-directional, hierarchical connected or limited with certain conditions (resonance).

Table 2.2 Connections between neurons.

Interlayer connections		Intralayer Connections	
Fully Connected	Each neuron in previous layer is connected to the every neruon in next layer	Recurrent	Neurons in a layer are fully or partially connected to each other. After receiving inputs from neurons of other layers, they send data to each other before sending data to other layers.
Partially Connected	Not all neuron has to be connected to all neurons in next layer.		
Feedforward	Neurons in previous layer send data to the neurons in next layer.		
Bi-directional	Neurons in previous layer send data to the neurons in next layer and vice versa.	On-center / off surround	Neighbor neurons communicate with each other and update their weights before sending output to other layer neurons.
Hierarchical	Neurons of a lower layer can only communicate with the next level layer neurons.		
Resonance	Neurons have bidirectional connection and they continue to communicate until a certain condition is achieved.		

Table 2.3 Some activation functions.

Step Function (unipolar)	Step Function (bipolar)	Step Function with Bias
		
$y = \sigma(v) = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases}$	$y = \sigma(v) = \begin{cases} +1 & \text{if } v \geq 0 \\ -1 & \text{if } v < 0 \end{cases}$	$y = \sigma(v) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} \geq \theta \\ 0 & \text{if } \mathbf{w} \cdot \mathbf{x} < \theta \end{cases}$
Piecewise-linear Function	Sigmoidal Function (Logistic Function) (unipolar)	Sigmoidal Function (bipolar)
		
$y = \sigma(v) = \begin{cases} 0 & \text{if } v \leq -\frac{1}{2\alpha} \\ \alpha v + \frac{1}{2} & \text{if } v < \frac{1}{2\alpha} \\ 1 & \text{if } v \geq \frac{1}{2\alpha} \end{cases}$	$\sigma(v) = \frac{1}{1 + e^{-v}} = \frac{1}{2}(\tanh(v/2) + 1)$	$\sigma(v) = \tanh(\beta v) = \frac{2}{1 + e^{-2\beta v}} - 1$
Hyperbolic Tangent Function	Radial Basis Function	Identity Function
		
$\tanh x = \frac{e^{2x} - 1}{e^{2x} + 1}$	$f(x) = e^{-ax^2}$	$f(x) = x$

2.2.4 Architectures and Learning Paradigms in Artificial Neural Networks

2.2.4.1 Hopfield Model

Hopfield network is a single layer, symmetrically weighted autoassociative fully connected network. The network takes two-valued activations bipolar (+1,-1) or binary (0, 1). See equation (2.14) and equation (2.15). Mathematical analysis is easier with bipolar inputs. Hopfield model is described in terms of an energy function (Equation 2.16).

The Hopfield neural network is a simple artificial network which is able to store certain memories or patterns. The full pattern can be recovered if the network is presented with only partial information. Because of this feature, it is similar to the brain. Training a Hopfield net involves lowering the energy of states. The net serves as a content addressable memory system, so if it is given only part of the state, the network will converge to a remembered state. Hopfield model can be seen in Figure 2.4.

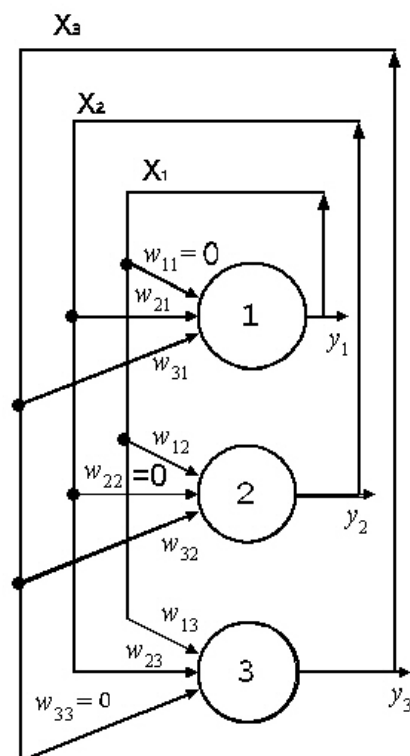


Figure 2.4 Hopfield Model

$$f(\text{net}) = \begin{cases} 1 & \text{if net} \geq 0 \\ 0 & \text{if net} < 0 \end{cases} \quad (2.14)$$

$$f(\text{net}) = \text{sgn}(\text{net}) = \begin{cases} 1 & \text{if net} \geq 0 \\ -1 & \text{if net} < 0 \end{cases} \quad (2.15)$$

$$E = -\frac{1}{2} \sum_{i \neq j} \sum w_{ij} s_i s_j + \sum_i \tau_i s_i \quad (2.16)$$

An activation function is simply a function that is used to introduce nonlinearity to the network. The choice of activation function can change the behavior of the ANN considerably. Some activation functions can be seen in Table 2.2.

2.2.4.2 The Perceptron Model

The perceptron is invented in 1957 at the Cornell Aeronautical Laboratory by Frank Rosenblatt. It can be seen as the simplest kind of feedforward, supervised neural network. The perceptron is a kind of linear binary classifier that maps its inputs x to an output value $f(x)$. x is a binary vector whereas $f(x)$ is a single binary value. It is calculated as:

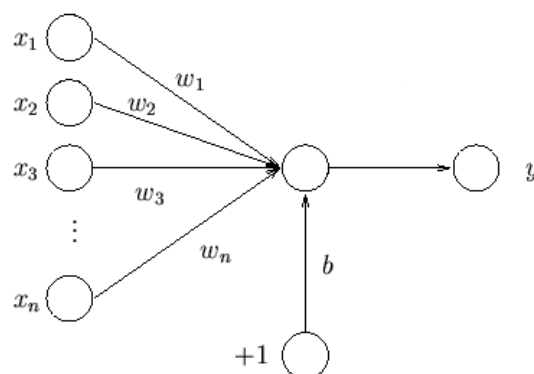


Figure 2.5 Perceptron model.

$$f(\text{net}) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{else} \end{cases} \quad (2.17)$$

The Hopfield Net is a neural network that is a lot simpler to understand than the Multi Layer Perceptron.

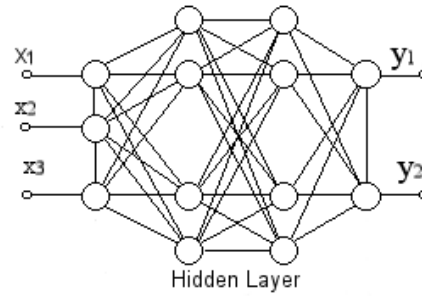


Figure 2.6 Multi layer perceptron.

2.2.4.3 Cellular Neural Networks Model

Cellular neural networks (CNN) are a parallel computing paradigm defined in discrete N-dimensional spaces. Communication is only allowed between neighbor units. There are two templates for iteration; control template and feedback template. CNN paradigm was first proposed by Chua and Yang in 1988. The two most fundamental ingredients of the CNN paradigm are: The use of analog processing cells with continuous signal values and local interaction within a finite radius (Destri, G., n.d.; Moser, 1998). Architecture of CNN can be seen in Figure 2.8.

$$y_k = \frac{1}{2} (|x_k + 1| - |x_k - 1|) \quad (2.18)$$

$$\dot{x}_1 = -x_1 + \sum_{k \in N} a_k y_k + \sum_{k \in N} b_k u_k + z \quad (2.19)$$

$$\dot{X} = -X + A * y(X) + B * u + I \quad (2.20)$$

CNN architecture and block-scheme of a generical CNN iteration can be seen in Figure 2.8.

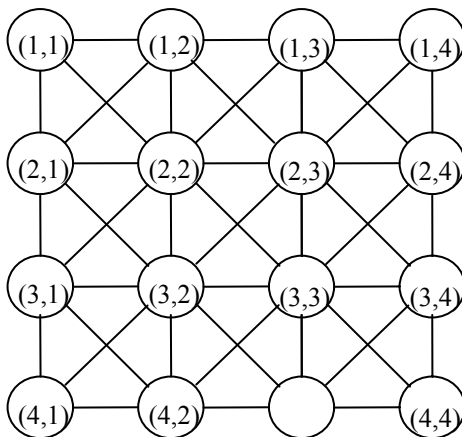


Figure 2.7 4x4 cellular neural network model.

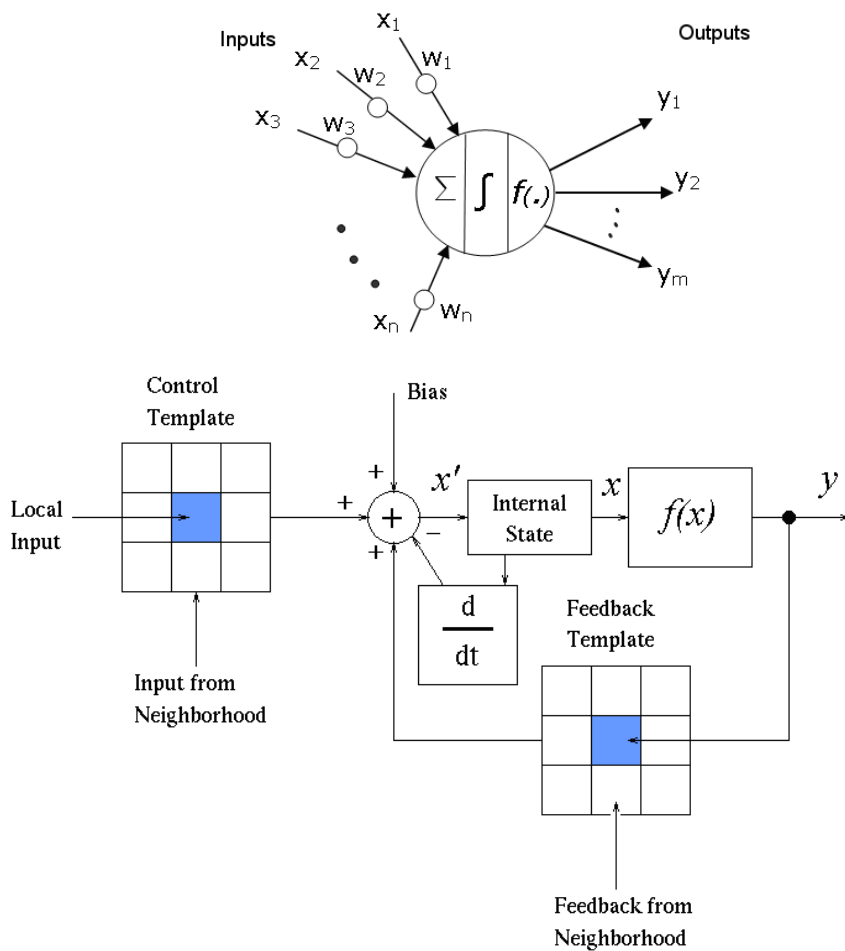


Figure 2.8 Block-scheme of a general CNN iteration and architecture of CNN

2.2.4.4 Winner Takes All Model

Winner-takes-all (WTA) network is an unsupervised competitive learning network. The desired output vector in winner-takes-all model has a single active unit because the output nodes are said to compete with each other to be the one to fire (see Figure 2.9). An input vector (\mathbf{X}) is applied to all nodes; the best response is declared the inner according to the winner selection criterion below,

$$y_n = \max (w_n X); i=1,2, \dots, N \quad (2.21)$$

$$\Delta w_n(k+1) = w_n(k) + \eta(k)(x - w_n) \quad (2.22)$$

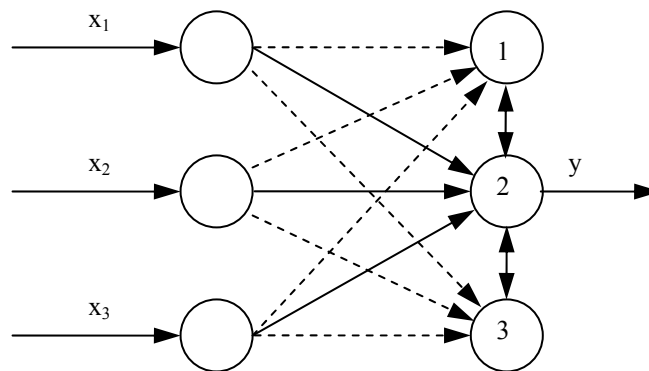


Figure 2.9 Winner-takes-all model with node 2 winning

2.2.4.5 Back-Propagation Model

The backpropagation algorithm was first introduced in 1974 by Paul Werbos in his Ph. D. Thesis. Back propagation neural networks are one of the most common neural network structures. The back propagation algorithm uses feedforward supervised learning. It is simple and effective. The network receives inputs by neurons in the input layer, and the output of the network is given by the neurons on an output layer. The training begins with random weights, and the goal is to adjust them so that the error will be minimal. There may be one or more intermediate

hidden layers. Character recognition is one of the most popular applications of BPNNs.

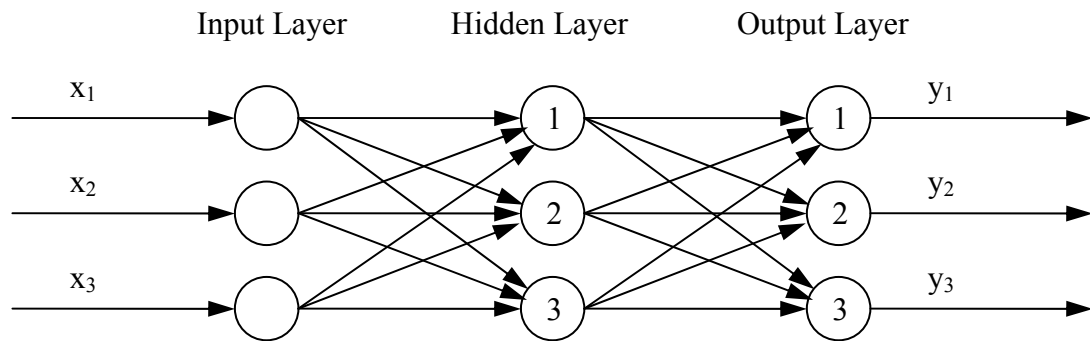


Figure 2.10 Backpropagation algorithm applied one layer feedforward neural network

$$y_j = \sigma \left(\sum_{i=1}^n w_{ji} x_i + b_j \right) \quad (2.23)$$

2.2.4.6 McCulloch-Pitts Model

The early model of an artificial neuron or the threshold logic unit first proposed by Warren McCulloch and Walter Pitts in 1943. It is a neuron of a set of inputs $x_1, x_2, x_3, \dots, x_n$ and one output y . The inputs and outputs are both binary. The linear threshold gate simply classifies the set of inputs into two different classes. Thus, the output y is also binary.

$$y_i = f \left(\sum_{j=1}^N (x_{ij} w_{ij}) - b_i \right) \quad (2.24)$$

As a transfer function, it employs a threshold or Heaviside step function taking on the values 1 or 0 only. Model equation can be seen as equation 2.24 and model can be seen in Figure 2.11.

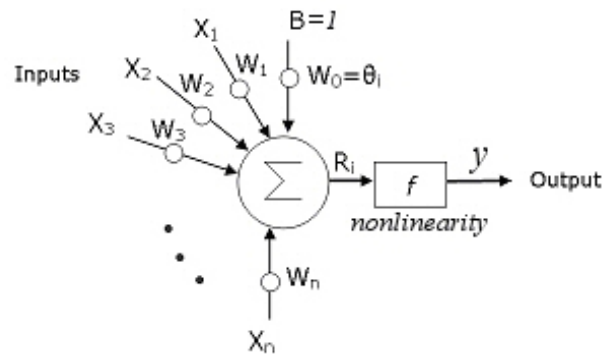


Figure 2.11 McCulloch-Pitts model

2.2.4.7 Adaptive Resonance Theory (ART) Model

Adaptive Resonance Theory (ART) is a neural network architecture developed by Stephen Grossberg in 1976. It was developed to solve the instability problem of feedforward systems. The basic ART system is an unsupervised learning model, a vector classifier. It accepts a vector as input and classifies it into a category depending on the stored pattern it most closely resembles. If the input pattern is found, it is trained to resemble the input vector. If the input vector does not match any stored pattern within a certain tolerance, then a new category is created by storing a new pattern similar to the input vector.

There are different types of adaptive resonance theory models. ART1 performs unsupervised learning for binary input patterns. The ART1 architecture has two layers: The input - comparison layer with N nodes and output - recognition layer with M nodes. Output layer of ART1 model is a winner-takes-all layer. ART2 is modified to handle both analog and binary input patterns. ART3 performs parallel searches of distributed recognition codes in a multilevel network hierarchy. ARTMAP combines two ART modules to perform supervised learning whereas fuzzy ARTMAP represents a synthesis of elements from fuzzy logic, neural networks, and expert systems.

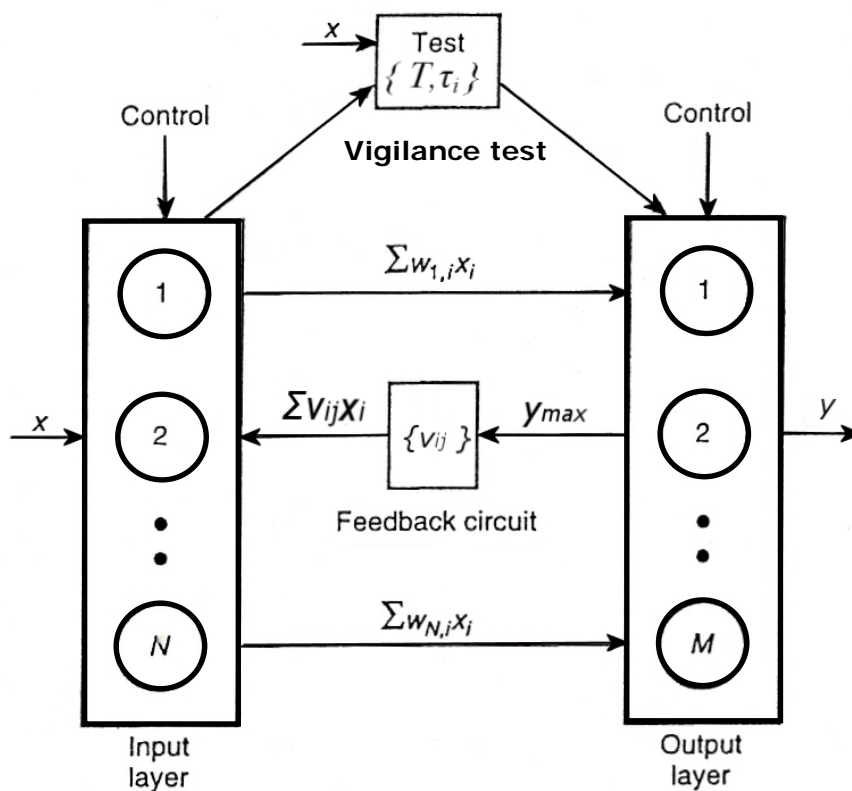


Figure 2.12 Adaptive resonance theory model

2.2.4.8 Radial Basis Function (RBF) Model

A radial basis function network is an artificial neural network which uses radial basis functions as activation functions. It is a classification and function approximation algorithm. If the architecture of the radial basis function studied (see Figure 2.13), there are no connection weights between input layer and hidden layer. The weights are between hidden layer and output layer. RBF may be used to simulate the nonlinear relationship between the sensors measurement and the ideal output value (Noguchi, 1997).

$$f(x) = \sum_{j=1}^m w_j h_j(x) \quad (2.25)$$

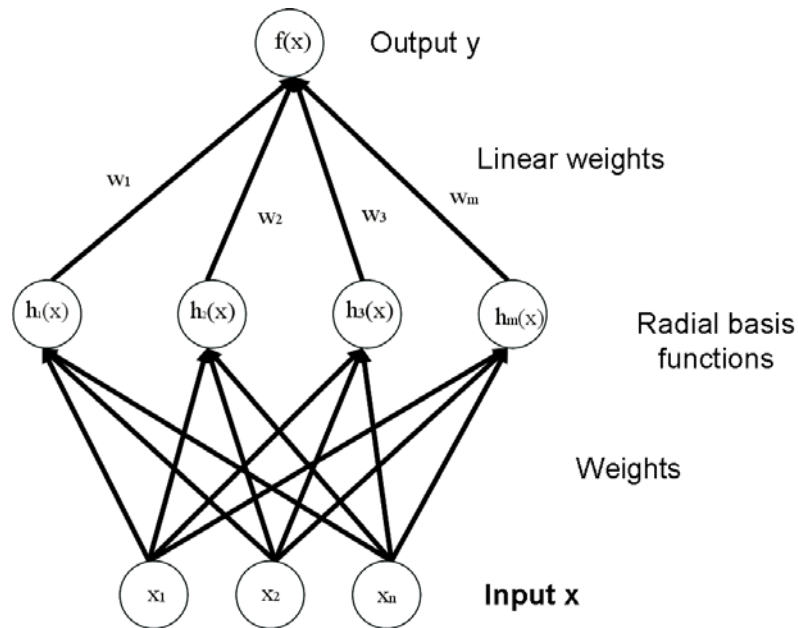


Figure 2.13 Radial basis function network architecture. The weights are between hidden and output layers.

RBF has three layers: Input layer, hidden layer and output layer. The input layer is fully connected to hidden layer. Hidden layer consists of units those are activated by radial basis activation functions. In Table 2.3, some activation functions can be seen. Equation 2.29 is the output of the RBF model.

Table 2.4 Commonly used RB activation function types

Name	Equation
Gaussian	$h(x) = \exp\left(-\frac{(x-c)^2}{r^2}\right)$
Multiquadric	$h(x) = \frac{\sqrt{r^2 + (x-c)^2}}{r}$
Euclidean distance	r
Thin plate splines	$r^m \log r$; $m = 2,4,6,\dots$
Smooth splines	rm ; $m=1,3,5,\dots$

2.2.4.9 Widrow-Hoff Model (ADALINE and MADALINE)

ADALINE (adaptive linear element, adaptive linear neuron) was developed by Professor Bernard Widrow and his graduate student Ted Hoff at Stanford University in 1960. It is similar to McCulloch-Pitts neuron. This model is supervised and feedforward. It is a single neuron whose weights are updated according to LMS (Least Mean Square) algorithm. The LMS algorithm is an adaptive algorithm that computes adjustments of the neuron synaptic weights.

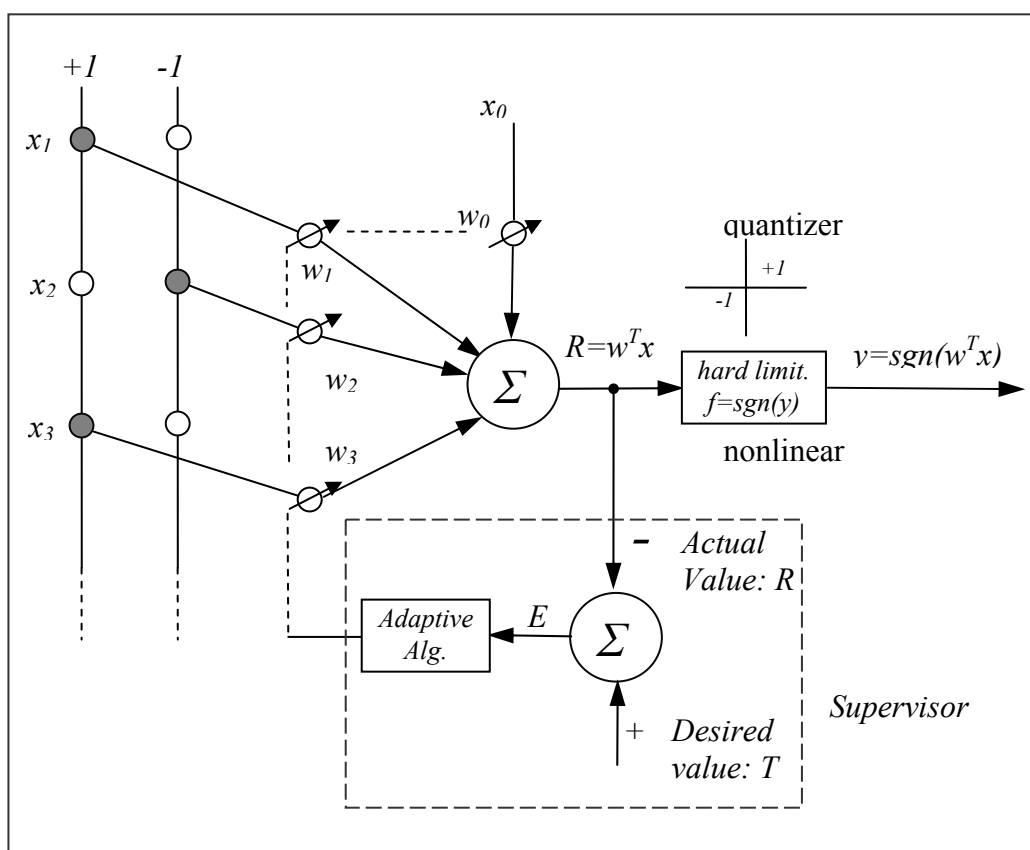


Figure 2.14 Adaline architecture.

$$y = \sum_{i=1}^n x_i w_i + \theta \quad (2.26)$$

$$w_i(n+1) = w_i(n) + \eta(n)x_i(n) \quad (2.27)$$

Using equations (2.27) is another form of equation (2.13) with the slope value of 1. Δw_i is the synaptic adjustment value and can simply be obtained from equation (2.27) as $\eta(n)x_i(n)$. In Figure 2.14 a detailed model of ADALINE may be seen.

MADALINE (multiple ADALINE) is a solution to nonlinear separation problem (see Figure 2.15). MADALINE has two types: MADALINE I is a multiple ADALINE with single output whereas MADALINE II has multiple outputs.

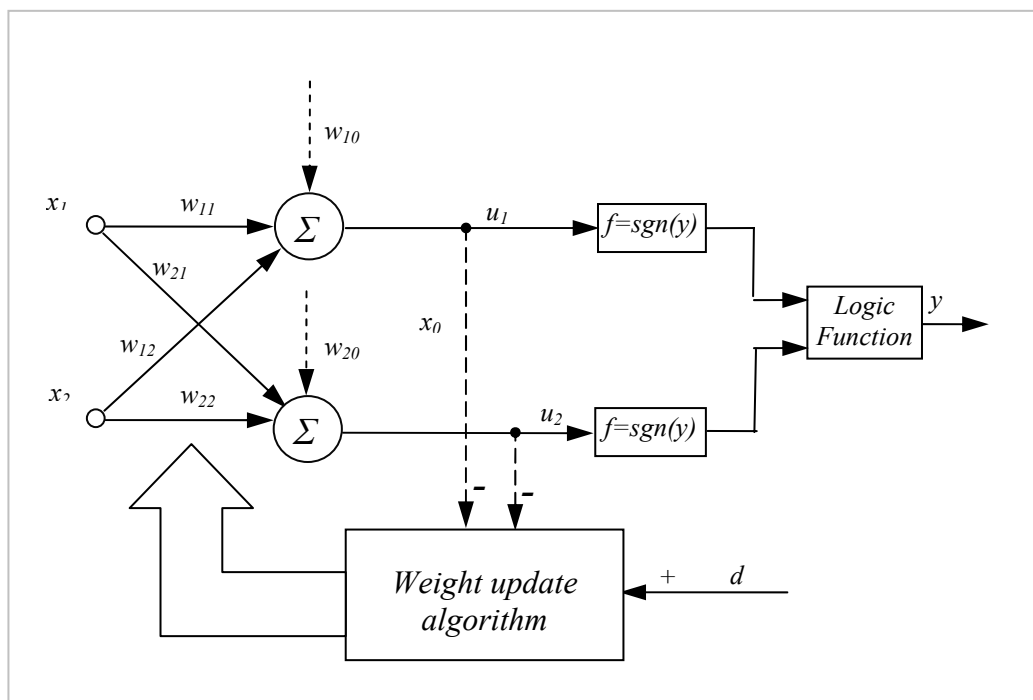


Figure 2.15 Madaline I architecture. MADALINE I is a multiple ADALINE with single output.

2.2.4.10 Kohonen (Self Organizing Map- SOM) Model

Self organizing map is a feedforward unsupervised network. A Kohonen network is composed of a grid of output units and N input units. The input pattern is fed to each output unit with weights. These weights are initialized as small random numbers. A set of artificial neurons learn to map points in an input space to coordinates in an output space. SOM is called a topology-preserving map because there is a topological structure imposed on the nodes in the network. A topological

map is simply a mapping that preserves neighborhood relations. The goal is to train the net so that nearby outputs correspond to nearby inputs.

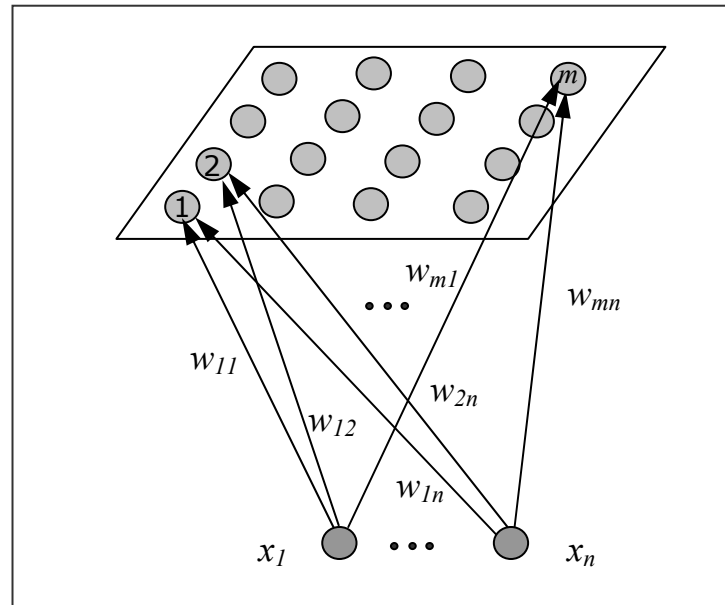


Figure 2.16 SOM conventional feature mapping architecture.

$$d_j = \sum_{i=0}^{N-1} (x_i(t) - w_{ij}(t))^2 \quad (2.28)$$

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)(x_i(t+1) - w_{ij}(t)) \quad (2.29)$$

Linear vector quantization net and SOM net are similar to each other since both have a single layer of nodes and use a distance metric to find the output node closest to input pattern. SOM uses equation (2.28) to find the closest distance and equation (2.29) to update the weights.

2.2.4.11 Learning Vector Quantization (LVQ) Model

Learning Vector Quantization is a supervised feedforward net which is developed by Teuvo Kohonen, like SOM algorithm. It is a modified SOM algorithm for classification. It has one hidden layer of neurons, fully connected with the input layer. The weights of the network are changed by the network in order to classify the

data correctly. LVQ LVQ1, OLVQ1 (Optimized Learning Vector Quantization 1), LVQ2.1, LVQ3, OLVQ3 are some different types of LVQ's. LVQ algorithms do not approximate density functions of class samples. Using a nearest-neighbor rule and a winner takes all paradigm, they directly define class boundaries based on prototypes. Some application areas for LVQ algorithms are: Pattern recognition, multi-class classification and data compression tasks. LVQ architecture can be seen in Figure 2.17.

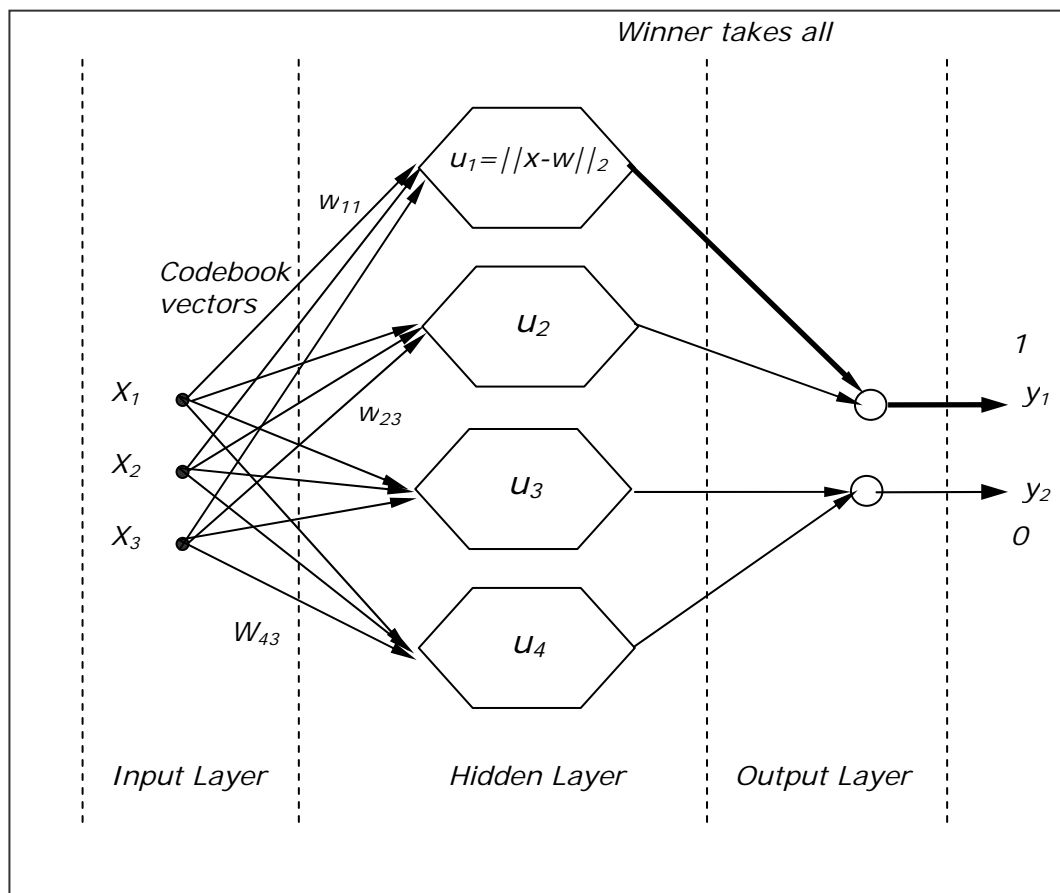


Figure 2.17 LVQ architecture

LVQ1 paradigm is as follows (Van Laerhoven, K., 1999):

- i. m_c is the codebook vector that is closest to the input x , so this will define the classification of x ,
- i. Update the codebook vectors $m_i = m_i(t)$,
- ii. If x is classified correctly: $m_c(t+1) = m_c(t) + \eta(t) \cdot [x(t) - m_c(t)]$,

- ii. If the classification is incorrect: $m_c(t+1) = m_c(t) - \eta(t) \cdot [x(t) - m_c(t)]$, and $m_i(t+1) = m_i(t)$ for $i \neq c$.

For each input pattern, LVQ finds the output node with the best match to the training pattern in training period. If the class of the training pattern differs from the class of output node, it finds the next best match. If the next best match is the appropriate class, LVQ moves the best match node farther from the training pattern.

The second version of LVQ adds a symmetric window of nonzero width between two codebook vectors.

2.2.4.12 Boltzmann Machines

The Boltzmann Machine is an artificial neural network of units with an "energy" defined for the network which is developed by Hinton and Sejnowski. It has stochastic binary units, unlike Hopfield nets have. Some application fields of Boltzmann Machines are: Combinatorial optimization, classification, and association. The global energy, E , in Boltzmann Machines is:

$$E = -\sum_{i < j} w_{ij} s_i s_j + \sum_i \tau_i s_i \quad (2.30)$$

where $s_i \in \left\{ \begin{matrix} -1 \\ +1 \end{matrix} \right\}$ and $w_{ii}=0$; $w_{ij}=w_{ji}$.

Probability that a cell is in a given state depends on the *synthetic temperature* of the system (Te):

$$p_i(+1) = \frac{1}{1 + e^{\frac{\Delta E_i}{Te}}} \quad (2.31)$$

$$p_i(0) = 1 - p_i(+1) \quad (2.32)$$

$$P_{\alpha} = \frac{e^{-\frac{E_{\alpha}}{T_e}}}{\sum \beta e^{-\frac{E_{\beta}}{T_e}}} \quad (2.33)$$

Boltzmann distribution can be seen in equation (2.33) and Boltzmann Machines architecture can be seen in Figure 2.18.

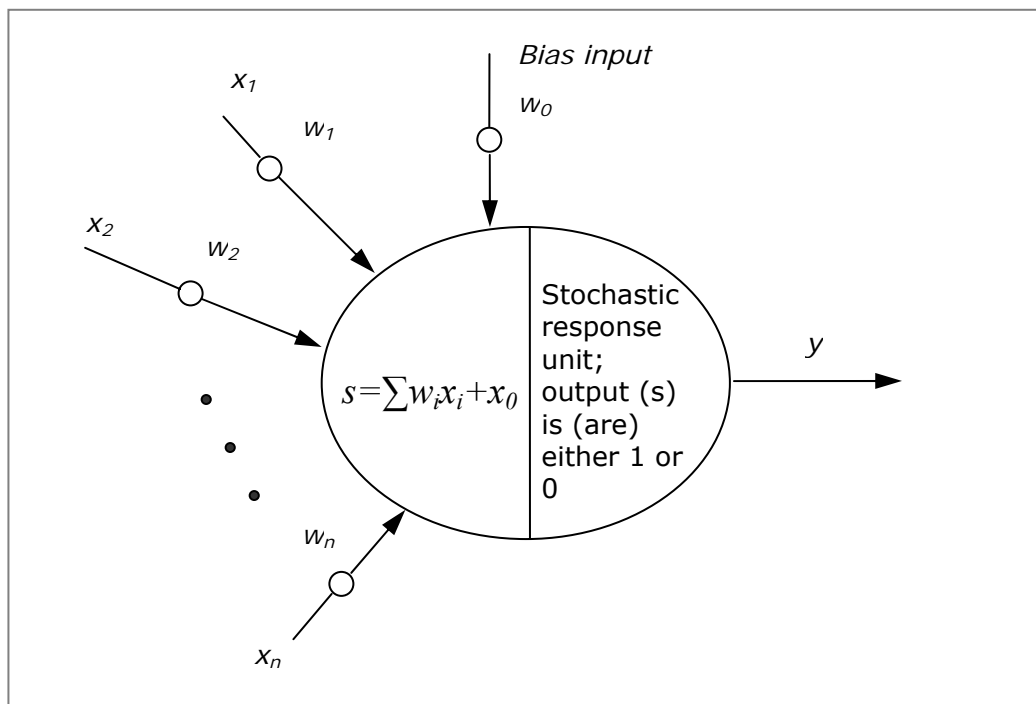


Figure 2.18 Architecture of Boltzmann Machines which is inspired from annealing in metallurgy.

Boltzmann learning is based upon a simulated annealing technique. Compared to other learning algorithms such as backpropagation, BM is significantly slower. The name and inspiration come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. The heat causes the atoms to become unstuck from their initial positions (a local minimum of the internal energy) and wander randomly through states of higher energy; the slow cooling gives them more chances of finding configurations with lower internal energy than the initial one (*Simulated Annealing*, 2007).

2.2.4.13 Hebbian Learning Rule

Hebbian learning rule is developed by D.O. Hebb in 1949. Hebb's original proposal was worded as:

When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased (Hebb, 1949, p. 62).

Hebbian learning is an unsupervised training algorithm in which the synaptic weight is increased if both the source neuron and target neuron are active at the same time.

$$w_{ij} = \frac{1}{n} \sum_{k=1}^p x_i^k x_j^k \quad (2.34)$$

where w_{ij} is the weight of the connection from neuron j to neuron i , n is the dimension of the input vector, p the number of training patterns, and x_j^k is the k^{th} input for neuron i . Hebbian learning rule is:

$$w_{ij}(t+1) = w_{ij}(t) + \eta x_i(t) y_j(t) \quad (2.35)$$

The discrete time standard Hebbian learning rule using energy function is:

$$w(k+1) = w(k) + \mu[y(k)x(k) - \alpha x(k)] \quad (2.36)$$

The continuous time standard Hebbian learning rule is:

$$\frac{dw}{dt} = \mu(yx - \alpha x) \quad (2.37)$$

2.2.4.14 Principal Component Analysis

Principal components analysis (PCA) is an artificial neural network technique which is used to reduce multidimensional data sets to lower dimensions for analysis (see Figure 2.19).

PCA is used to compress data, like image compressing. PCA can be also used as finding free spaces on mobile robot control as it condenses the input data down to a few principal components (Janglova, 2004).

The outputs of the PCA networks can be found using the equation below:

$$y_i(n) = \sum_{j=0}^{p-1} w_{ij}(n)x_j(n) \quad j=0,1,2,\dots,m-1 \quad (2.38)$$

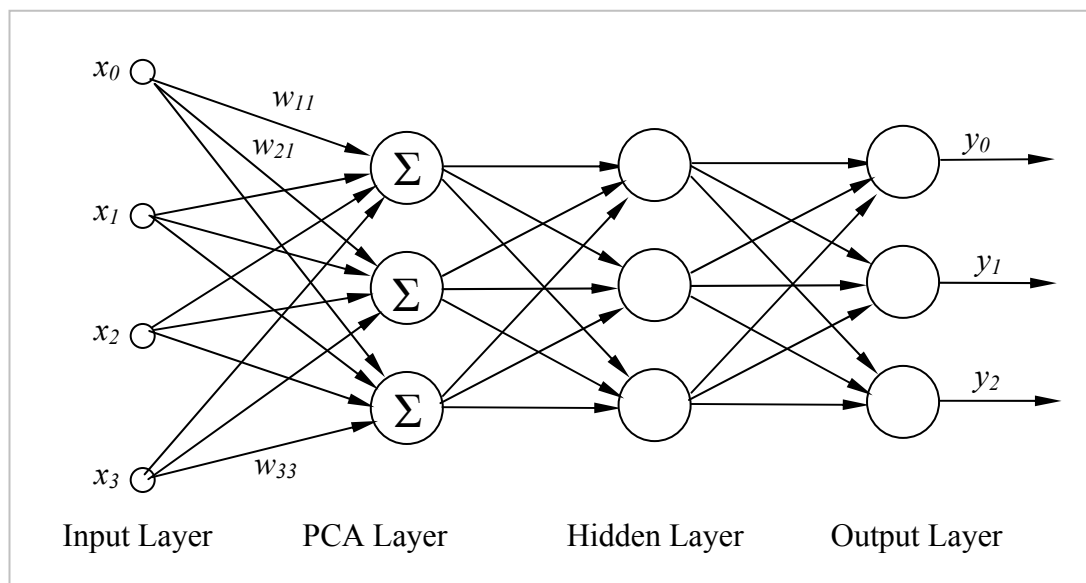


Figure 2.19 An application of ANN with one layer of principal component analysis.

Some other models are: Restricted Coulomb Energy- RCE Model, Culbertson's Model, Encephalon Model, Logicon Projection Network- LPN Model, Probabilistic RAM Model, Neural Accelerator Chip Model, Cerebellum Model Articulation Controller (CMAC) Model, Memory Type Models, Probabilistic Neural Network –

PNN Model, Time-Delay Neural Net (TDNN) Model, Linear Associative Memory (LAM) Model, Cognitron and Neocognitron Models, Real-time Models.

2.3 Fuzzy Logic

2.3.1 Introduction

The concept of Fuzzy Logic (FL) has been introduced by Lotfi Zadeh (1965), a professor at the University of California at Berkley. He presented fuzzy logic not as a control methodology, but as a way of processing data. In contrast to binary classical set memberships, fuzzy sets are sets whose elements have degrees of membership. Fuzzy set theory was not applied to control systems until the 70's because of insufficient small-computer capability.

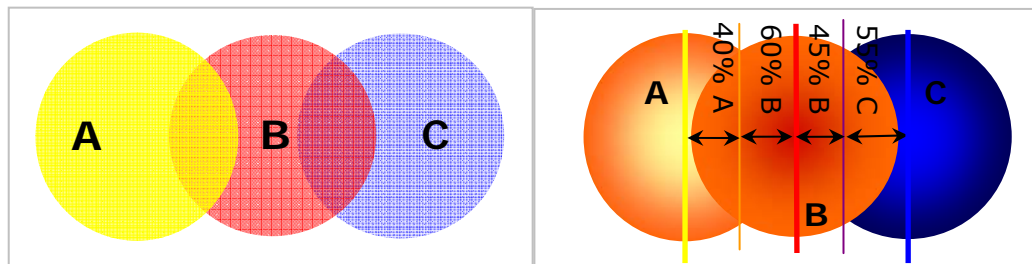


Figure 2.20 Memberships in crisp sets and in fuzzy sets.

Today, with the low cost use of microcomputers and microcontrollers, fuzzy logic is a very effective solution in many control problems ranging from embedded microcontrollers to personal computers as implemented in hardware, software or in both. Providing a simple solution for a desired conclusion from noisy, ambiguous, imprecise, or missing input information is the advantage of fuzzy logic. It can be used in hardware, software, or a combination of both.

2.3.2 Membership Function

For any set A , a membership function on A is any function from A to the real unit interval $[0, 1]$. For an element x of the fuzzy set \tilde{A} , the value $\mu_A(x)$ is called the

membership degree of x in \tilde{A} . The value 0 means that x is not a member of the fuzzy set whereas the value 1 means that x is fully a member of the fuzzy set. The values between 0 and 1 characterize fuzzy members, which belong to the fuzzy set partially.

2.3.3 Fuzzy Logic Control

2.3.3.1 Fuzzy Logic Operations

The standard definitions in fuzzy logic as suggested by Prof. Zadeh are:

- **Negate (negation criterion)** : $\text{truth}(\text{not } x) = 1.0 - \text{truth}(x)$
- **Intersection (minimum criterion)**: $\text{truth}(x \text{ and } y) = \text{minimum}(\text{truth}(x), \text{truth}(y))$
- **Union (maximum criterion)**: $\text{truth}(x \text{ or } y) = \text{maximum}(\text{truth}(x), \text{truth}(y))$

In order to clarify this, a few examples are given. Let A be a fuzzy interval between 4 and 6 and B be a fuzzy number about 3. The corresponding figures are shown below.

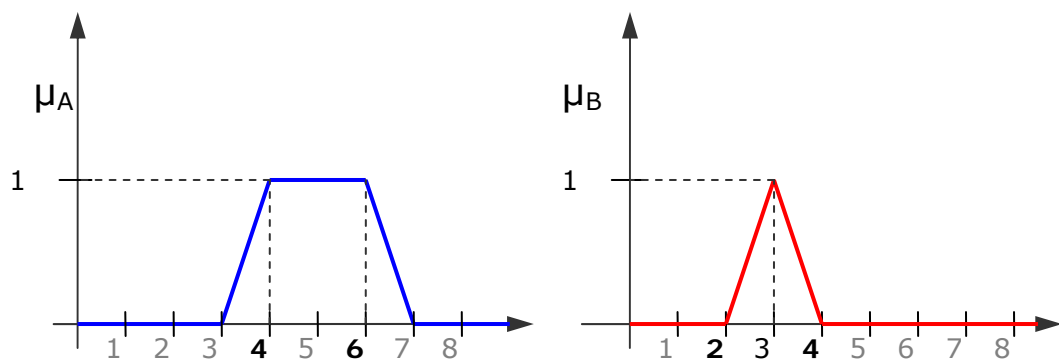


Figure 2.21 Fuzzy set μ_A and fuzzy number μ_B

The figure below gives an example for a negation. The red line is the negation of the fuzzy set μ_A , in other words $\mu_{\bar{A}}$.

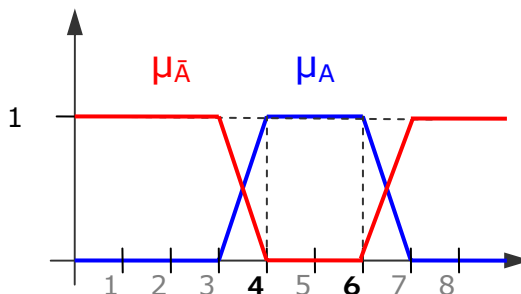


Figure 2.22 Negation of μ_A

The following figure shows the fuzzy set between 4 and 6 AND about 3 (red line). The minimum criterion is used. Intersection of μ_A and μ_B is in plum color.

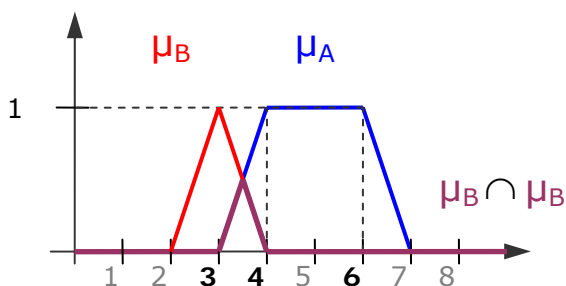


Figure 2.23 μ_A , μ_B and intersection of μ_A and μ_B

Finally, the Fuzzy set between 4 and 6 OR about 3 is shown in the next figure (green line). The maximum criterion is used.

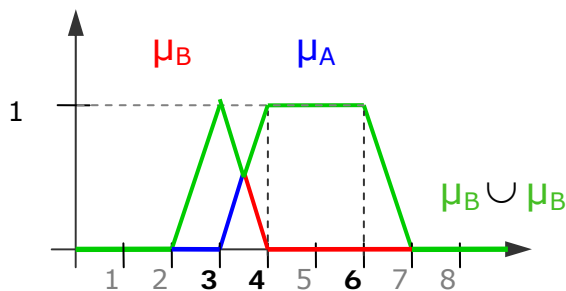


Figure 2.24 μ_A , μ_B and union of μ_A and μ_B

These basic operations provide guidelines to construct more complex ones which in turn can be used to create fuzzy machines. The following rules are common in classical set theory and fuzzy set theory.

Table 2.5 Some rules which are applicable to classical set and fuzzy set theories.

De Morgans law:	$\overline{(A \cap B)} = \bar{A} \cap \bar{B}$ $\overline{(A \cup B)} = \bar{A} \cup \bar{B}$
Associativity	$(A \cap B) \cap C = A \cap (B \cap C)$ $(A \cup B) \cup C = A \cup (B \cup C)$
Commutativity	$A \cap B = B \cap A$ $A \cup B = B \cup A$
Distributivity	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$

2.3.3.2 Control

Fuzzy logic control (FLC), which directly uses fuzzy rules, is the most important application in fuzzy logic theory. It can be either open or closed loop control. There is no learning in fuzzy control as in artificial neural networks, but if the main criterion is time, fuzzy control can be a very effective solution even on mobile robots. It is widely used on microcontrollers to force the control system to behave more naturally. FLC block diagram can be seen in Figure 2.25.

Inverted pendulum and crane problems are very popular examples of fuzzy logic control (Abdul Aziz, 1996; Becerikli, 2007; Schneider, 2007; Zhang, 1992). The problem is to balance an inverted pendulum on a mobile platform that can move to the left or to the right. The angle between the platform and the pendulum and the angular velocity of this angle are chosen as the inputs of the system whereas the speed of the platform is chosen as the corresponding output. FLC is also used in some researches about mobile robots mainly guidance (Gharieb, 2000).

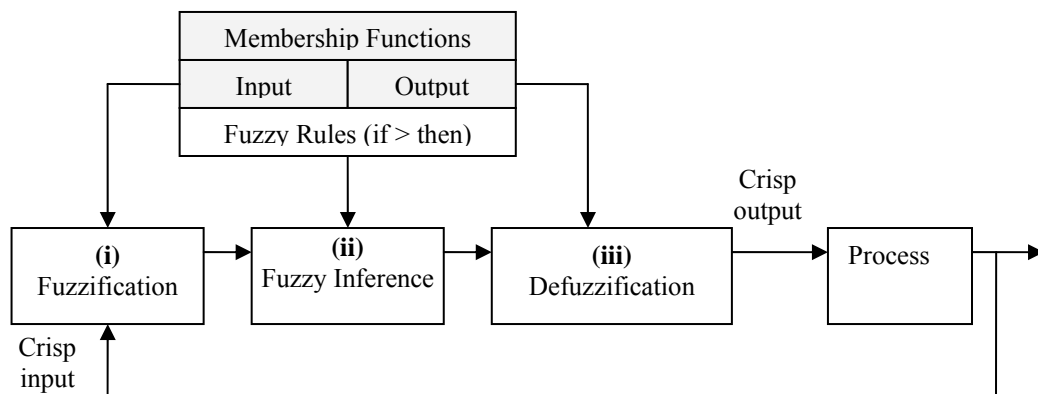


Figure 2.25 Fuzzy logic control block diagram

Three steps are taken to create fuzzy logic control:

- i. **Fuzzification:** Graphically description using membership functions. Levels of output of the platform are defined by specifying the membership functions for the fuzzy sets. These levels are named as high, medium, low, etc... Similarly, the different angles between the platform and the pendulum and the angular velocities of specific angles are also defined.

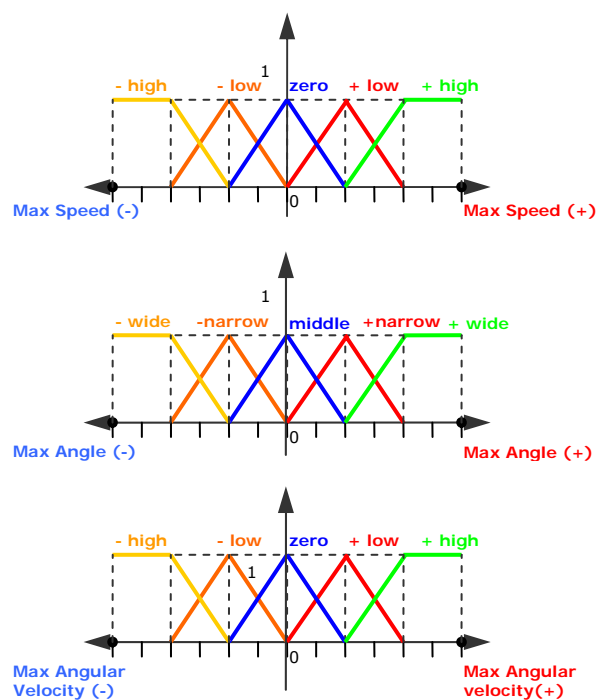


Figure 2.26 Fuzzification

- ii. **Rule evaluation:** Fuzzy logic rule evaluation from graphics. Second step is to define the fuzzy rules. The fuzzy rules are a series of if-then statements as mentioned before. These statements are usually derived by an expert to achieve optimum results. Some examples of these rules are: If angle is zero and angular velocity is zero then speed is also zero. If angle is zero and angular velocity is low then the speed shall be low. And so on. The full set of rules can be seen in the table below

Table 2.6 Control rules

		<i>Velocity</i>				
		NL	NM	ZR	PM	PL
<i>Angle</i>	NL	NL	NL	NL	NM	NS
	NM	NL	NL	NM	NS	PS
	NS	NL	NM	NS	PS	PM
	PS	NM	NS	PS	PM	PL
	PM	NS	PS	PM	PL	PL
	PL	PS	PM	PL	PL	PL

* ZR: Zero; NL: Negative Large; NM: Negative Medium; NS: Negative Small; PL: Positive Large; PM: Positive Medium; PS: Positive Small.

After completing table, the next step is to describing regions with due to desired output of the system (see Figure 2.27). Let us say the actual value of the angle is the value at the point A in the angle graph and angular velocity has the value of the point B in angular velocity graph. As for the rule, ‘the strength of the output is as strong as the weakest component’, using AND operator, the minimum value is taken for this area, so the graph of the speed for this area can be seen also in Figure 2.27. If there is more than one activated output for the same region, then the strongest one prevails for the region. These operations are applied to all regions due to desired behavior of the system. If first input is x_1 in A_{1ij} ; second input is x_2

in A_{2ki} and output is y in O_m^i , rules for obtaining the region values are equations (2.39) and (2.40).

$$\mu O_m^i = \min [\mu A_{1ji}(x_1), \mu A_{2ki}(x_2)] \tag{2.39}$$

$$\mu O_m^{p\&q}(y) = \max [\min [\mu A_{1jp}(x_1), \mu A_{2kp}(x_2)], \min [\mu A_{1jq}(x_1), \mu A_{2kq}(x_2)]] \tag{2.40}$$

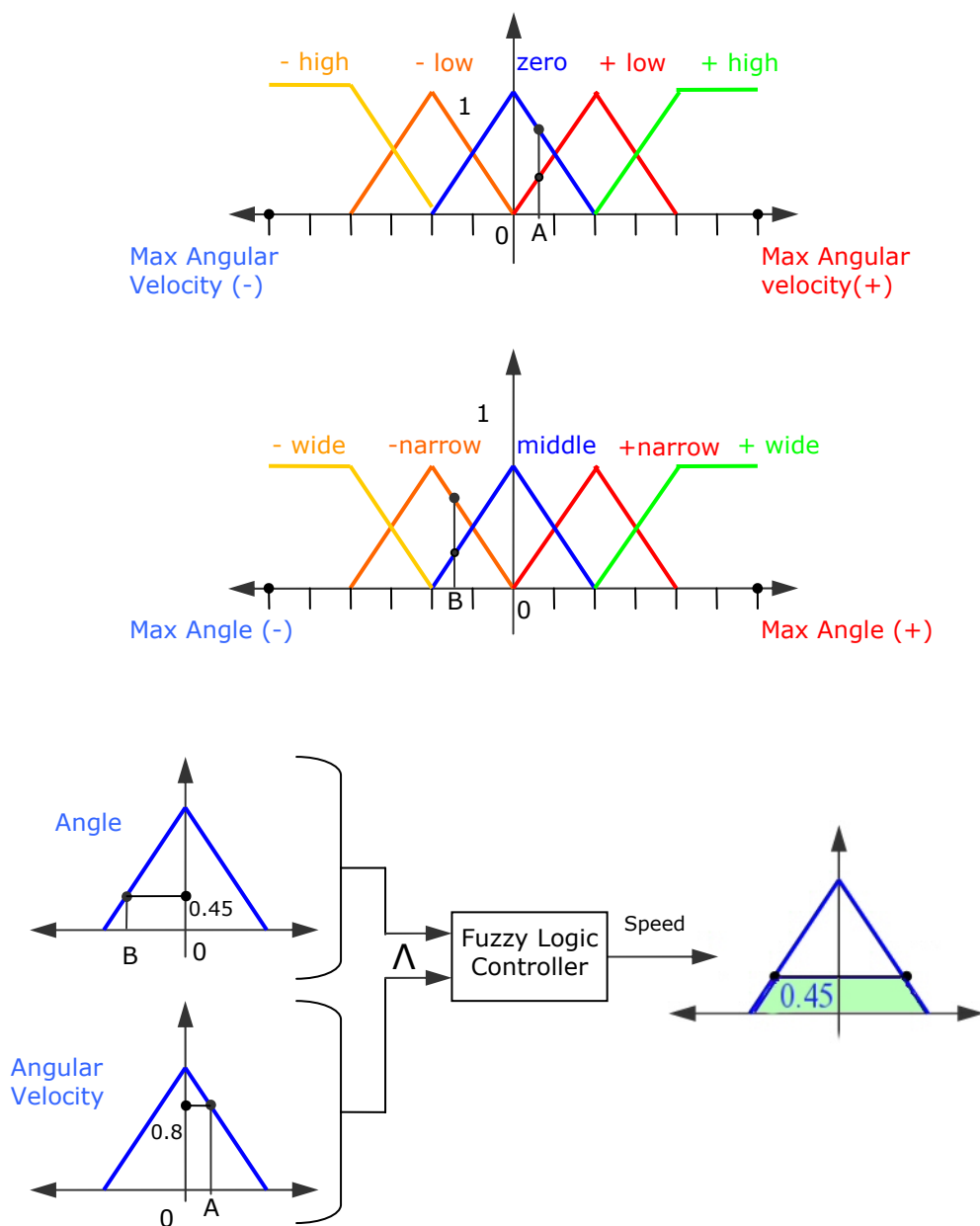


Figure 2.27 Describing regions for desired output

- iii. **Defuzzification:** Obtaining crisp results from rules. The values of the speed due to regions are combined in a graph. An example graph can be seen in Figure 2.28.

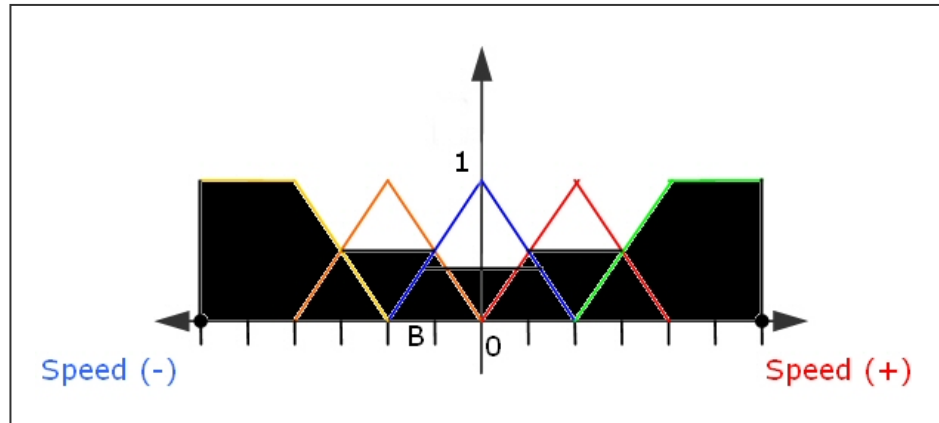


Figure 2.28 Defuzzification stage in FLC design (output of the Fuzzy Logic Controller).

CHAPTER THREE
AUTONOMOUS NON-HOLONOMIC MOBILE ROBOTS

3.1 Non-Holonomic Vehicles

Nonholonomic conditions are assumed to be expressible as non-integrable differential relations. The terminology was introduced by Hertz.

Non-holonomic constraint is defined in very different ways. One of them is: “A non-holonomic constraint is expressed as a non-integrable equation involving derivatives of the configuration parameters and it can not be reduced to equality constraint on the position parameters.” (Dimirovski, 2000)

Non-holonomic means to have fewer controllable degrees of freedom than total degrees of freedom. In other words, the outcome of a non-holonomic system is path dependent and after its motion, it is not sure that the system will return to its original position. For example, ordinary automobiles are nonholonomic systems. As a result of this, they can not move to every direction in any moment. Ideal holonomic system is a space craft in space (or a submarine in sea) which can move to any desired direction. But, the mobile robots which can move to every desired direction on the ground also called as holonomic mobile robots even they can not fly.



Figure 3.1 Some holonomic wheels and systems. Holonomic wheels are wheels with two degrees of freedom. They are also known as omni-directional wheels or omni-wheels.

Non-holonomic constraint of the model of this research is formed by non-steering front wheels. Mobile robot rotates with the difference between left and right front wheel velocities (see Figure 4.2). But, it is good to mention once more that this constraint is for the motions in directions of X and Y axes; in other words, the robot has no actuator for motion in the direction of Z axis.

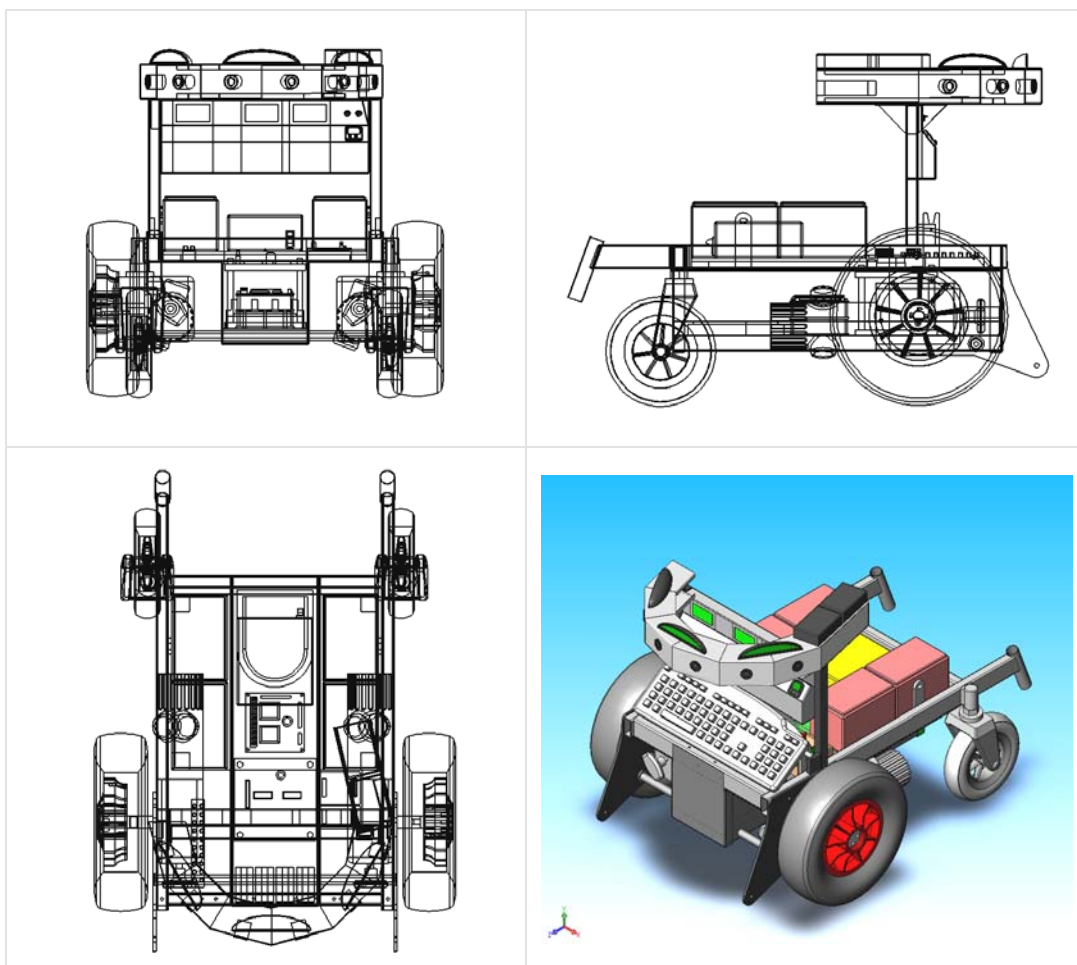


Figure 3.2 Autonomous wheeled nonholonomic mobile robot model in this research has four wheels.

3.2 Autonomous Wheeled Mobile Robots

Autonomous mobile robot is the robot which can change its location and move sensible interacting with environment including objects and living things. Autonomous motion has three stages: Sensing, planning and acting.

Kinematic modeling of a wheeled mobile robot is different than a stationary manipulator. In kinematic modeling of stationary manipulators, the mechanisms are low-pair whereas in wheeled mobile robots they are high-pair. Since Denavit-Hartenberg Convention has some problems with multiple closed chains, Sheth – Uicker Convention is used. The angular velocities of the wheels can be converted directly into translational velocities along the surface which the wheels are on (Muir, 1986).

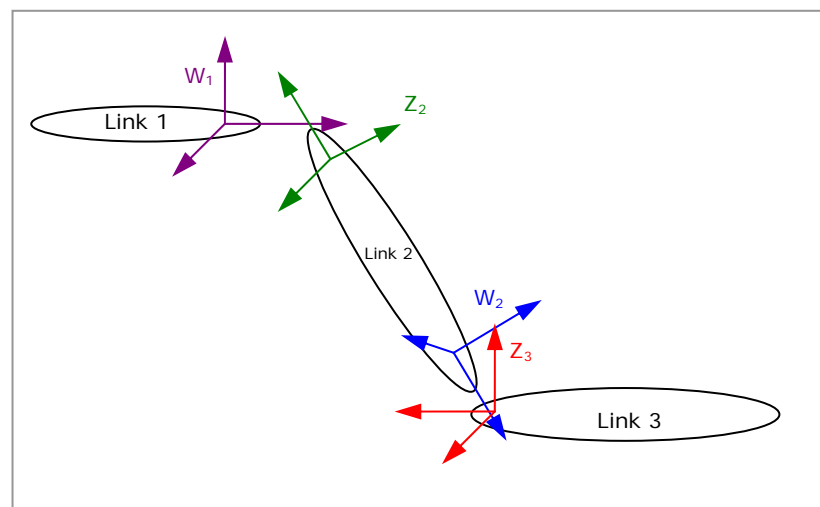


Figure 3.3 In Sheth – Uicker notation, each joint has two coordinate systems because of high kinematic pairs.

Wheeled mobile robots (WMR) have very huge amount of different types as in number of wheels / actuators, dimensions or as tasks control types.

In kinematic analysis of mobile robots, there are four main differences for kinematic analysis of robot manipulators (Muir, 1986).

- i. Stationary manipulators only form closed chains when they are contact with fixed objects whereas; wheeled mobile robots form many closed chains at the same time.
- ii. The contact between a wheel and plane forms a higher-pair, but stationary manipulators contain only lower-pair joints.

- iii. In wheeled mobile robots, only some degrees of freedom of a wheel are actuated. However, all DOF's of each joint of a stationary manipulator has at least one actuator.
- iv. Each joint in stationary manipulator has position and velocity sensors. In wheeled mobile robots, only some degrees of freedom of a wheel have position or velocity sensors. Sheth-Uicker convention allows modeling the higher-pair relationship between each wheel on the WMR and the floor. Thus, the Sheth-Uicker convention is more suitable for finding transformation matrix in analyzing kinematics of wheeled mobile robots.

Left and right front wheel velocity equations of a four wheeled mobile robot whose two wheels are actuated can be seen in figure below. This is called differentially driven mobile robot. The reference point is denoted with M in Figure 3.4. M is the middle point of the two front wheels. As mentioned before, only the front wheels have motors.

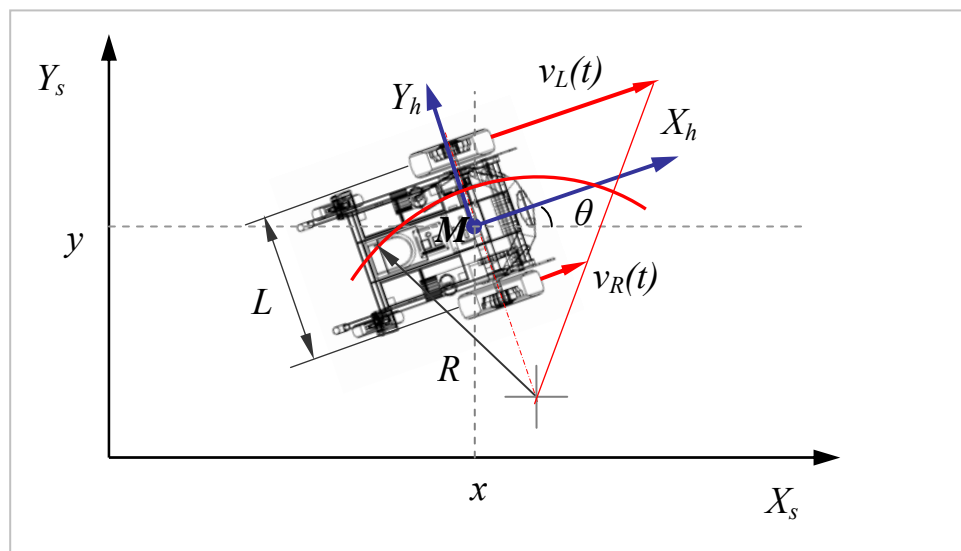


Figure 3.4 Velocities of left and right wheels of the mobile robot model in this research.

r : Dynamic radius of wheel [m],

R : Radius of curvature, instantaneous rotation radius or steer radius [m],

L : Distance between the middles of two front wheels [m],

v : Linear velocity of the mobile robot, [m/s],

$v_R(t)$: Linear velocity of the right front wheel, [m/s],

$v_L(t)$: Linear velocity of the left front wheel, [m/s],

θ : Heading angle, [rad],

$w(t)$: Angular velocity in z coordinate of the mobile robot, [rad/s],

$\{X_h, Y_h\}$: Moving coordinate axes,

$\{X_s, Y_s\}$: Stationary coordinate axes.,

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos \theta(t) & 0 \\ \sin \theta(t) & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v(t) \\ w(t) \end{bmatrix} \quad (3.1)$$

$$\dot{y} \cos \theta - \dot{x} \sin \theta = 0 \quad (3.2)$$

Equation (3.2) shows the non-holonomic constraint of the model. Equation 3.3 denotes velocities of left and right side front wheels. $r(s)$, is the radius of curvature at s and $\kappa(s)$ is the curvature at s . θ is the angle between the stationary coordinate axis and the tangent of the curve at point P.

$$\begin{bmatrix} V_x(t) \\ V_y(t) \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \left(\frac{r}{2}\right) & \left(\frac{r}{2}\right) \\ 0 & 0 \\ -\left(\frac{r}{L}\right) & \left(\frac{r}{L}\right) \end{bmatrix} \cdot \begin{bmatrix} w_L \\ w_R \end{bmatrix} \quad (3.3)$$

$$\kappa(s) = r(s)^{-1} \quad (3.4)$$

$$v_L = v \left(1 - \frac{L \cdot \kappa(s)}{2} \right) \quad (3.5)$$

$$v_R = v \left(1 + \frac{L \cdot \kappa(s)}{2} \right) \quad (3.6)$$

$$v_R = v \frac{r(s) + \frac{L}{2}}{r(s)} \quad (3.7)$$

If the WMR model steering is car like, which may be seen in Figure 3.5, this time the instantaneous rotation radius (R) is equal to L_m over $\tan(\sigma)$. The main difference between the car like mobile robots and differentially driven mobile robot is that the rotation. In differentially driven mobile robots, rotation motion is achieved with having different velocities of the wheels whereas in car like mobile robots, rotation is achieved with steering of one (or some) wheel(s).

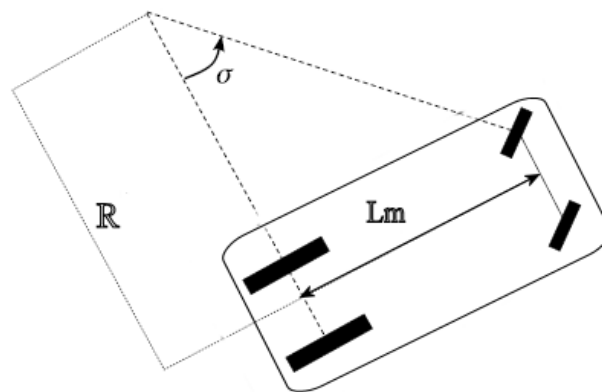


Figure 3.5 Car like wheeled mobile robot.

In analyzing the mobile robots, inverse kinematics can be used easier than direct kinematics because of the similarities of the mobile robots with the parallel manipulators (Tsai, 1999). In inverse kinematics, the end effector position is known and the aim is to find the joint values to make the effector in next position. Assumptions for mobile robot analysis are: Pure rolling, non slipping, constant wheel base and diameter.

3.3 Dead Reckoning

Dead reckoning is the real-time calculation of the wheeled mobile robot position in floor coordinates usually from wheel sensor measurements. This is also called forward position kinematics. Uncertainty of the WMR model, disturbance during motion of the mobile robot, localization errors (caused by sensors or difference of radius of wheel during motion...) and should be cumulated resulting in the position failure (Zu, 2004; Muir, 1986).

Discrete time kinematics is applied for dead reckoning to the model. If mobile robot travels with $v(k)$ and rotates with $\omega(k)$ in a uniform sampling k^{th} interval t , approximate discrete kinematics is given as:

$$\begin{bmatrix} x(k+1) \\ y(k+1) \\ \theta(k+1) \end{bmatrix} = \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} + \begin{bmatrix} t.v(k).\cos\theta(k) \\ t.v(k).\sin\theta(k) \\ t.\omega(k) \end{bmatrix} \quad (3.8)$$

However, dead reckoning is an ideal approach for localization. Wheels slippage, accuracy of encoders, changing radius of wheels and similar reasons cause dead reckoning has some problems. So, linear and angular velocities found using equation 3.7 said to be estimated values. Adding the uncertainty value η_{uc} which is a white Gaussian noise with covariance matrix $E[\eta_{uc} \eta_{uc}^T]$, the considered discrete kinematics model is (Vale, 2005):

$$x(k+1) = f(x(k), \tilde{v}(k), \tilde{\omega}(k)) + \eta_{uc} \quad (3.9)$$

3.4 Architecture of Autonomous Wheeled Mobile Robots

The purpose of building the architecture of a robot is pointing out the interactive parts of the robot to build the hardware and software framework. Mobile robot models have sensors, control parts, man machine interfaces, security systems, handling systems or mechanical parts, actuators and so on. So, it is needed to have main control and sub control systems. For example, different sensors have their sub control systems to be evaluated by the main control system. Architecture of a robot also makes the control of the robot easier to understand in means of priority levels of these control systems. The framework of software and hardware to control of the AWMR model in this research, in other words the architecture of the model in this research, can be seen in Figure 3.6.

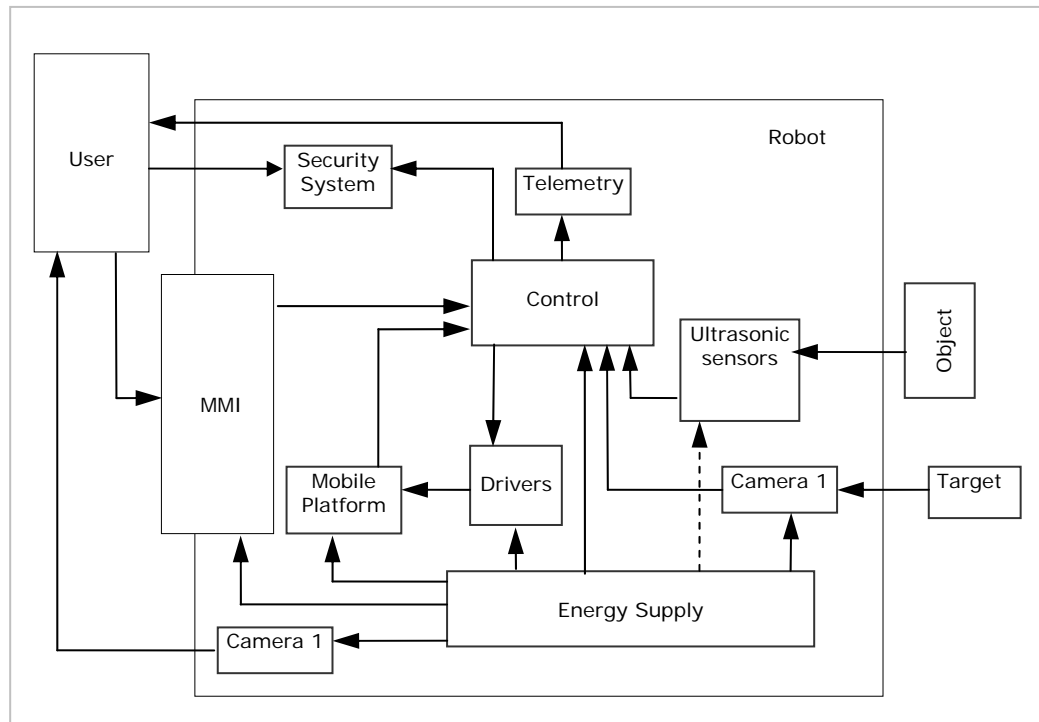


Figure 3.6 Architecture of the research model which is an example of autonomous wheel mobile robot

3.4.1 Mobile Robot Navigation

Robot navigation covers different type of navigation in meanings. Navigation is used for;

- Going from a totally different environment to another,
- Mapping environment for discovering a place,
- Finding path and reaching target.

The environment of the mobile robot must be considered if it will be used indoor or outdoor. For outdoor use, a very huge number of researches in recent years cover GPS receiver or obtaining précised GPS data (Gören, 2001). In addition to global positioning system receivers, mobile robots for outdoor use have encoders, laser sensors, ultrasonic sensors, maybe optical sensors, cameras, radars or sonar, and etc. The model in this thesis is an indoor mobile robot. For indoor mobile robots, however, it is not so easy to teach the robot to know the place that the robot itself

stands. Encoder, ultrasonic sensor, camera, RF transceiver, odometer, radar, gyroscope, sonar, laser sensors are some types which are used for indoor use of mobile robots. In this research, target is a visual unique landmark.

3.5 Braitenberg Vehicles

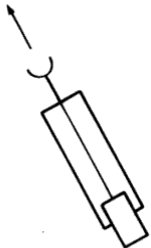
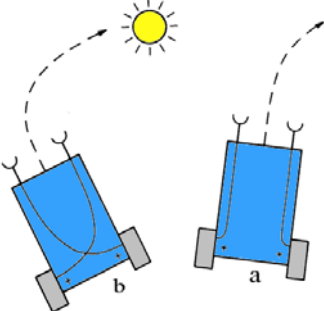
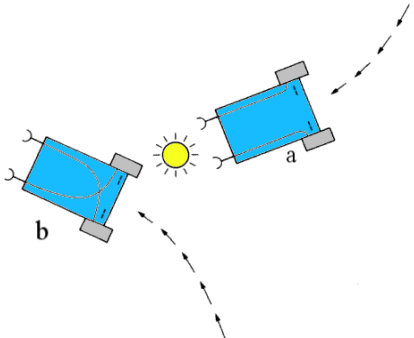
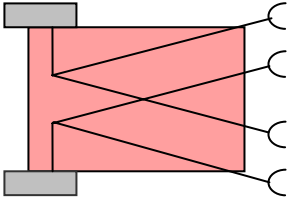
Braitenberg (1984), who is a neuro-scientist, described basic behaviors of wheeled mobile robots in his book 'Vehicles: Experiments in Synthetic Psychology' in 1984. He divided the behaviors in four thought experiments as for structure of the mobile robot. First type of the WMR has one wheel with one sensor. Sensor directly affects the motor activity.

The second type of Braitenberg vehicle consists of two sensors and two motors. Sensors are either directly or cross connected to motors and motor activity is directly proportional to sensor activity. Depending on the connection type between sensors and motors, vehicle accelerates to the source or accelerates to escape from source, so behaves like having either "fear" or "aggression" (see Table 3.1).

The third type of Braitenberg vehicles are the WMR's has two sensors and two motors similar to vehicle 2. The difference is the inhibitory effect of sensors to the motors. If the sensors cross connected to motors, WMR escapes from the source whereas it goes directly to the source when the sensors are directly connected to motors.

The last type of the Braitenberg vehicles which is also called as 'vehicle 4' is more near to the present modern WMR's. It has many different types of sensors (gyroscope, GPS, temperature, encoders, cameras, laser, sonar, ultrasonic, odometers, etc.) and via more than one controller; the WMR has different logics embedded on it. Combining the sensor data, it makes its decision for motion. Both senses and the motions of the robot are more complex than the other type of Braitenberg vehicles. In designing a mobile robot, even the task(s) of the robot is (are) complex, better type is the simpler type of Braitenberg vehicles that completes its task.

Table 3.1 Braitenberg vehicles.

<p>Vehicle 1: Getting around. Simplest vehicle, always forward. One motor with one sensor.</p>	<p>Vehicle 2: Fear and aggression. Two sensors and two motors. (a) is directly, (b) is cross connected. Excitatory.</p>
	
<p>Vehicle 3: Love. Inhibitory.</p>	<p>Vehicle 4: Different types of sensors mounted. Complex behavior.</p>
	 <ul style="list-style-type: none"> - temperature, - ultrasonic, - camera, - ... <p>different type of sensors</p>

CHAPTER FOUR

SYSTEM AND EXPERIMENTS

4.1 Kinematics of the Experimental Mobile Robot

In Chapter Three, kinematics, architecture and behavior of autonomous wheeled mobile robots are studied. The first part of this chapter is related with the experimental WMR model that was originally a wheel chair which has four wheels and whose two wheels are actuated with two motors (See Figure 4.3 and Figure 4.4). Experimental mobile robot position with respect to reference coordinates is:

$$q = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad (4.1)$$

Rotation matrix of the reference coordinates with respect to moving coordinate system is:

$$R(\theta) = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4.2)$$

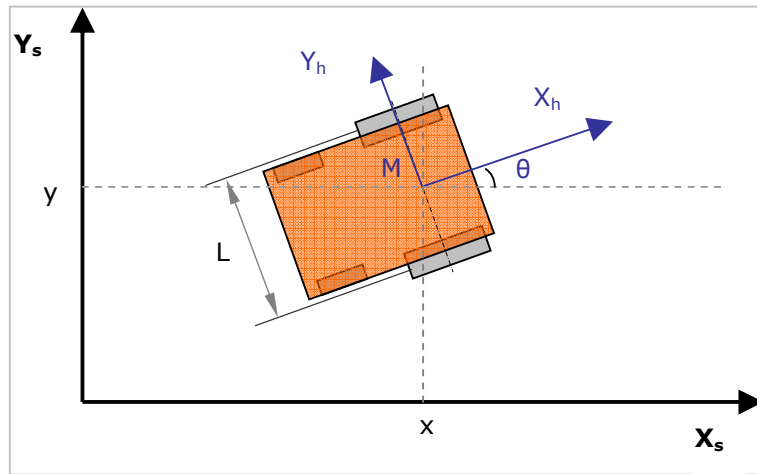


Figure 4.1 Coordinate System Locations of the mobile robot.

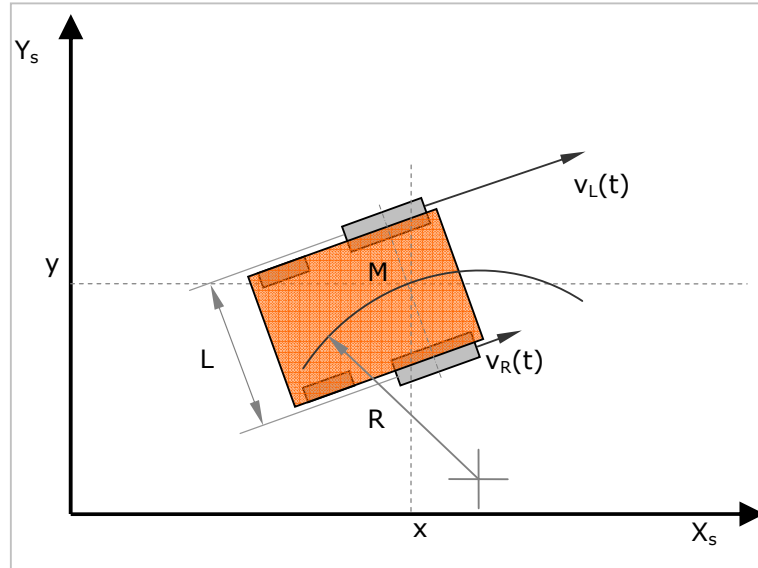


Figure 4.2 Left and right motor velocities of the mobile robot

In Figure 4.1, reference coordinate system and moving coordinate system; in Figure 4.2, velocity variations in respect to instantaneous center of rotation of Autonomous Mobile Robot (AMR) can be seen.

Velocities of the right and left wheels can be defined as:

$$\begin{bmatrix} V_x(t) \\ V_y(t) \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \left(\frac{r}{2}\right) & \left(\frac{r}{2}\right) \\ 0 & 0 \\ -\left(\frac{r}{L}\right) & \left(\frac{r}{L}\right) \end{bmatrix} \cdot \begin{bmatrix} w_L \\ w_R \end{bmatrix} \quad (4.3)$$

or

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos \theta(t) & 0 \\ \sin \theta(t) & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v(t) \\ w(t) \end{bmatrix} \quad (4.4)$$

In condition of going straight forward (if the velocity of the right and left wheel is equal);

$$w(t)=0 \rightarrow \theta:\text{constant} \quad (4.5)$$

$$\text{and } v(t)=v_L(t)=v_R(t) \quad (4.6)$$

If the mobile robot turns around on a point,

$$v(t)=0 \text{ and} \quad (4.7)$$

$$w(t) = \frac{2}{L} v_R(t) \quad (4.8)$$

This motion is one of the most known advantages of the dual drive vehicles. In this case, instantaneous center of rotation of the mobile robot model is on the middle of the front axis. And the model can turn around without going forward or backward.

4.2 Experimental System Model

Our laboratory built autonomous mobile robot (AMR) model in this research, which was originally a wheel chair having four wheels those all independent from each other. Front wheels are connected to two separated motors so that the autonomous wheeled robot can have more advantages in motion. Kinematic model of the robot is studied for a vehicle model that has four wheels and whose two wheels are driven.



Figure 4.3 Real model of experimental mobile robot

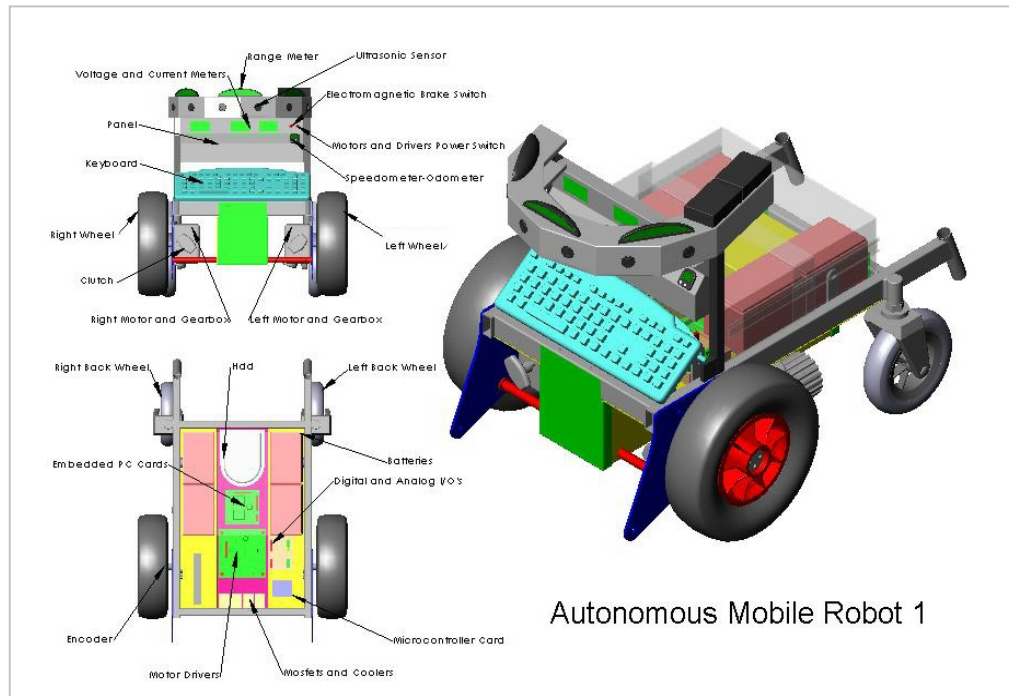


Figure 4.4 CAD model of mobile robot.

4.2.1 Ultrasonic Sensors for Navigation

A mobile robot has many sensors on it to know environment, to interact with human and to complete its tasks. For its purposes reliable sensor data is as important as accurate sensor data (Kopacek, 2006). But this is not the whole part, more is necessary for self localization of a mobile robot (Iyengar, 1991). After getting reliable data from sensors, method of evaluation in decision mechanism makes the robot movements either more quick and reasonable or redundant and clumsy. Frequently, more time is spent by robot to know environment than to move towards target. Most suitable method has to be selected rather than more complicated in means of motion and tasks of the mobile robot. But sometimes, it is inevitable to use more complicated evaluations and algorithms, especially when it is needed to map environment in details. In that case, generally priority level one task is to distinguish meaningful data in data mass.

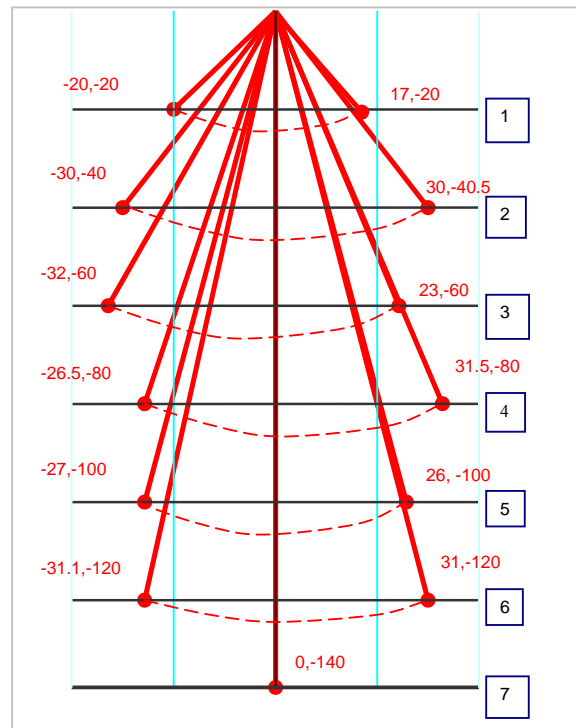


Figure 4.5 Tested characteristics of the ultrasonic sensors those are used (location units are in cm).

Six ultrasonic sensors with cone angle of more than 30° at a distance of 100cm are mounted on the model to scan 180 degrees. Two important parameters are effective for this choice. These are: Fast processing to control and reliable measurement data. For more than 30° cone angle, some regions intersect to each other. Data from the sensors are taken via microprocessor 1 in Figure 4.7 whereas second processor is used to get information from self-built encoders on wheels. Dempster – Shafer evidence theory helps to have reliable data from ultrasonic sensors, especially in intersecting regions which are roughly illustrated in Figure 4.7. Characteristics of the ultrasonic sensors and indicator table can be seen in Figure 4.5 and Table 4.1, respectively.

4.2.1.1 Ultrasonic Sensors Installation.

Input pins of Microcontroller 1 (PIC16F877) are connected to 2nd, 4th, 6th, and 7th output indicator pins of the ultrasonic sensor circuit which indicate closer than

0.2m, 0.6m, 1.0m and 1.4m respectively. This helps to determine 5 levels of closeness regions in every 30 degrees covering 180 degrees. These senses are needed to be very accurate since these mentioned values will be the inputs of the neural network in further research.

Table 4.1 Indicators of the Ultrasonic Sensors

	0m-0.4m	0.4m-0.6m	0.6m-0.8m	0.8m-1.0m	1.0m-1.2m	1.2m-1.4m	1.4m-2.0m
7	F						
6	F						
5							
4							
3							
2							
1							

Carrier frequency and signal shape from ultrasonic sensors can be seen in Figure 4.6. First signal in figure is the information of any object existence with its distance to related sensor in one region of six. Second signal on the figure is the synchronization signal. The period of the first signal is divided into four. Changing the form of the signal due to the distance sensed by the ultrasonic sensors, the information is sent to controller unit.



Figure 4.6 Oscilloscope output of the ultrasonic sensors

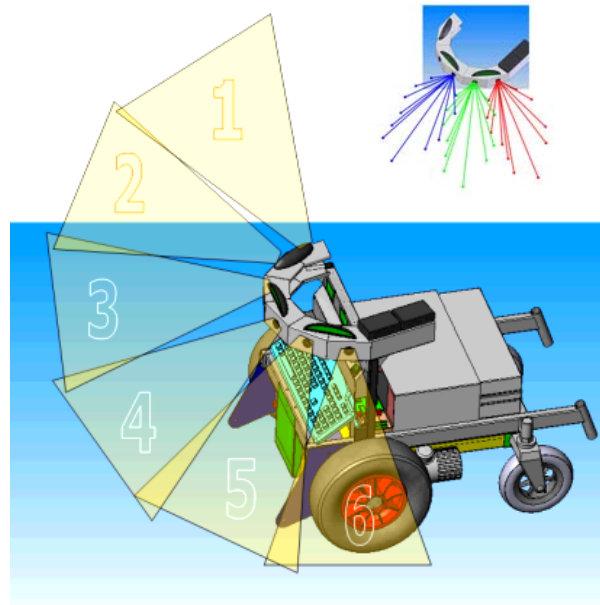


Figure 4.7 Ultrasonic sensor regions.



Figure 4.8 Installation of the ultrasonic sensors on real model.

4.2.2 Wireless Communications on Model

4.2.2.1 Radio Frequency Modems

Using serial port connected RF Modems; mobile robot is connected bidirectional to computer. Some part of the Quick Basic code for previous research on an open loop controlled mobile robot can be seen in Table 4.2 (Gören, 2001). In that research, mobile robot is equipped also with an embedded computer which has an 80386 CPU and very limited flash memory. The goal was to build a mobile robot to get information from a hazardous area for human. Mobile robot sends the data using RF modems and PC104 stable in a distance of 5km without effecting from disturbances. The main advantage of RF modem in that project was not to loose control of the mobile robot. The disadvantages of RF Modems are that they are heavy and relevantly slower in data transmission. Fault detection is also worse than othr solutions.

Table 4.2 RS232 connected RF-Modem QB sample program.

```

...
OPEN "com1:9600,n,8,1,rs,cs,ds" FOR OUTPUT AS #1
...
CASE "RIGHT"
CTR = CTR + RGTB
PRINT #1, CTR
SLEEP (1)
CTR = CTR - RGTB
PRINT #1, CTR
GOTO 110
...

```

In this research, however, the task of RF Modems is telemetry which means to send some information to monitor some data of the mobile robot on a stationary computer.

4.2.2.2 *Bluetooth*

An embedded computer which means a tiny main board and daughter boards in standards is mounted on the mobile robot. Using USB (Universal Serial Bus) port of this embedded computer, a Bluetooth dongle can be installed on the mobile robot. Bluetooth is a cable replacement technology offering point to point links without native support for IP. Main advantage is that the Bluetooth technology covers a very huge area, but it is not good for LANs.

Bluetooth offers the possibility to create an RfComm between a master and up to 6 slaves, with the SDP protocol to connect those pipes to specific applications or driver. TCP/IP is only one profile, implemented through PPP. Using Bluetooth software, serial communication ports of computer or mobile robot can be seen as ports of immobile computer. In addition to this, like using FTP server program in Wireless Ethernet 802.11g, Bluetooth file transfer program is tested to send meaningful text file to record path to mobile robot's computer.

4.2.2.3 Wireless Network

The 802.11g specification is a standard for wireless local area networks that offers transmission over distances at up to 54 Mbps. Compared with the 11 Mbps theoretical maximum of the earlier 802.11b standard, it is relatively short. 802.11g and 802.11b networks operate at radio frequencies between 2.400 GHz and 2.4835 GHz. The 802.11g uses orthogonal frequency division multiplexing (OFDM). 802.11g can fall back to speeds of 11 Mbps, so 802.11b and 802.11g devices are compatible. If the access point device uses 802.11b, modification of it to 802.11g compliance usually involves only a firmware upgrade.

In applications of wireless network in this research, VNC was used to monitor and teleoperate the robot whereas FTP was used to control the robot. VNC is Virtual Network Computing. It is remote control software which allows you to view and interact with another computer anywhere on the internet. It is operating system independent.

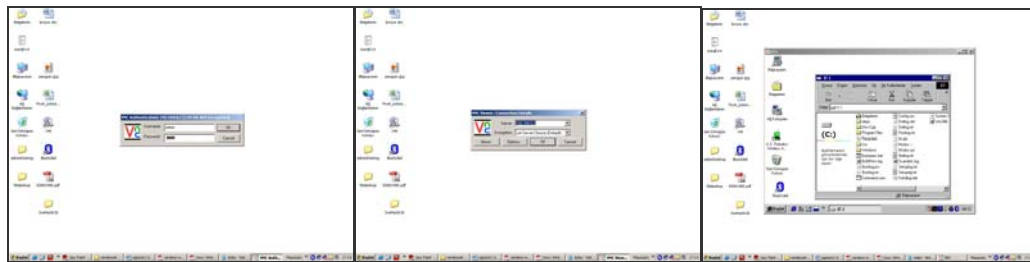


Figure 4.9 Screenshots of stationary computer during connecting to PC104 via VNC.

Using C Programming Language, desired path is transferred to the robotic vehicle. In application, this is a file transferred from the immobile computer, mainly just a text or character file. FTP Server program is installed in PC104 to get the relevant file from the desired port. This is just similar in Bluetooth. But, in Bluetooth, file transfer program is generally included in software package of Bluetooth device. Use of WLAN on mobile robot can be seen in Figure 4.10 and Figure 4.11.

In order to achieve security in wireless Ethernet, in addition to WPA-PSK security in wireless Ethernet, IP checking and user with a password is used for FTP server. Partly, SSH is also tested.

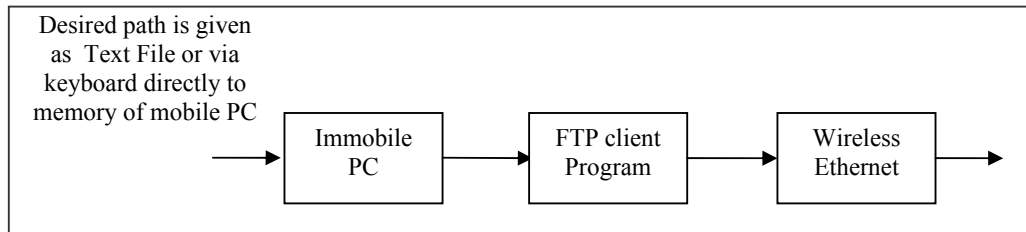


Figure 4.10 Immobile Computer (Server side)

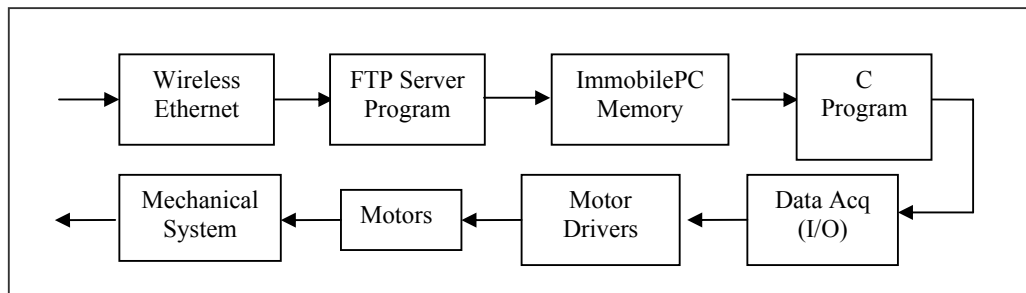


Figure 4.11 Mobile robot (Client Side)

4.2.3 Motors and Drivers

4.2.3.1 Motors of the Research Model

The AMR has two brush DC12V motors whose specifications are indicated in Figure 4.12.

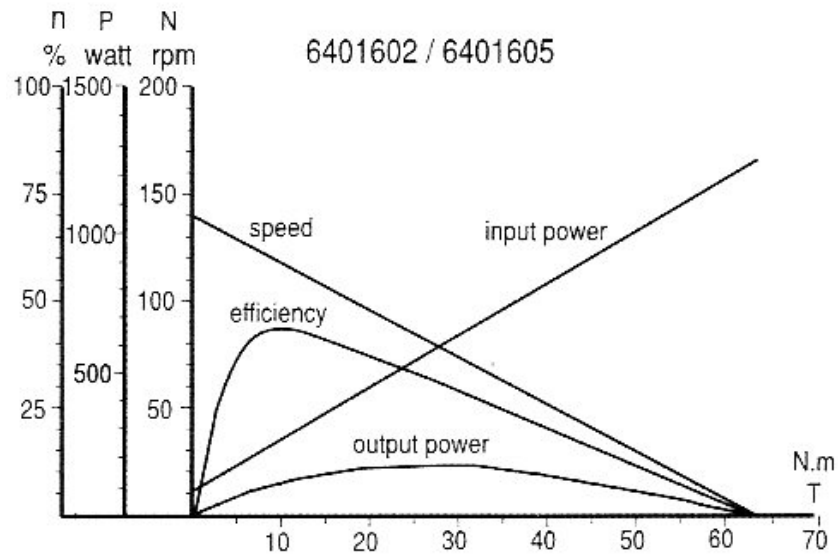
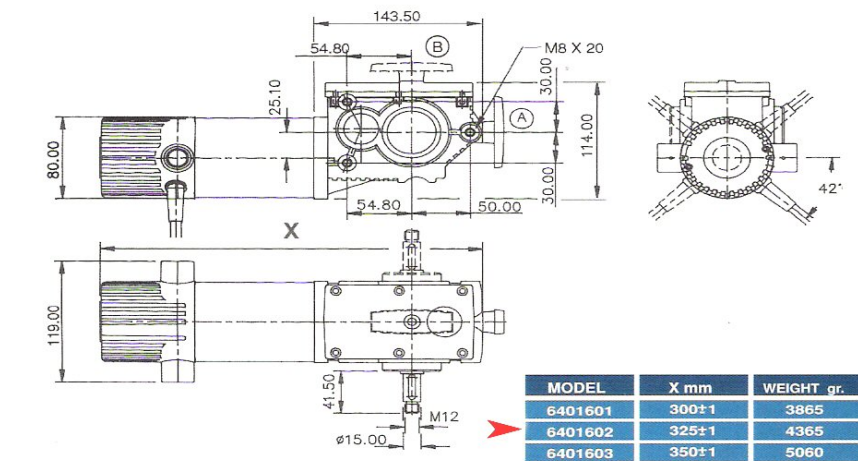


Figure 4.12 Dimensions and performance of the motors on WMR model.

4.2.3.2 Motor Drivers

Motor drivers of the model are also our lab made circuits which can be seen in Figure 4.13. Power voltage of the circuits is DC12V. It is an H Bridge whose power is 500 watt. Velocity of the motors can be changed as the velocity control input is changed between 0V - 5V. UC3525 in the circuit generates a width adjustable 20 kHz square signal. S5 transistor is used as a step-down-converter and changes the

output voltage, so the speed of the motor. Motor can be rotated backwards with making the input enable and input forward/backward high.

Effective period and frequency of the circuit can be set with the potentiometers on the circuit if needed.

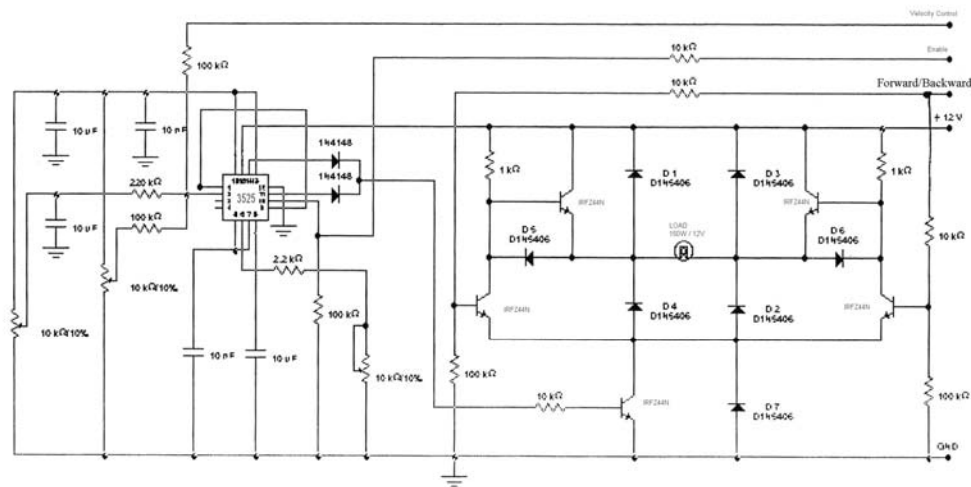


Figure 4.13 Motor driver card circuit (a larger version of this image can be found in Appendix 4, Schematic 2).

4.2.4 Controllers and Input – Output Cards

4.2.4.1 PC104: Embedded Computer on the Mobile Robot Model

As a master controller of mobile robot, a PC104 single board computer is selected. SBC is called PC104 because of the 104 pins that are used as a bus between the cards connected to each other. Using these pins, PC104 standard cards are connected physical one to another and different function cards use direct access to others. For the controller, an AMD Geode 300 MHz CPU PC104 with 256MB SD compact flash, 64MB SD Ram is selected. On the CPU module, an Ethernet chip and a VGA chip exists. USB, Parallel & Serial Port can be used as for the interfaces of the card. PCM 3350 type PC104 and its block diagram can be seen in Figure 4.14 and Figure 4.15 (PCM3350, 2006).

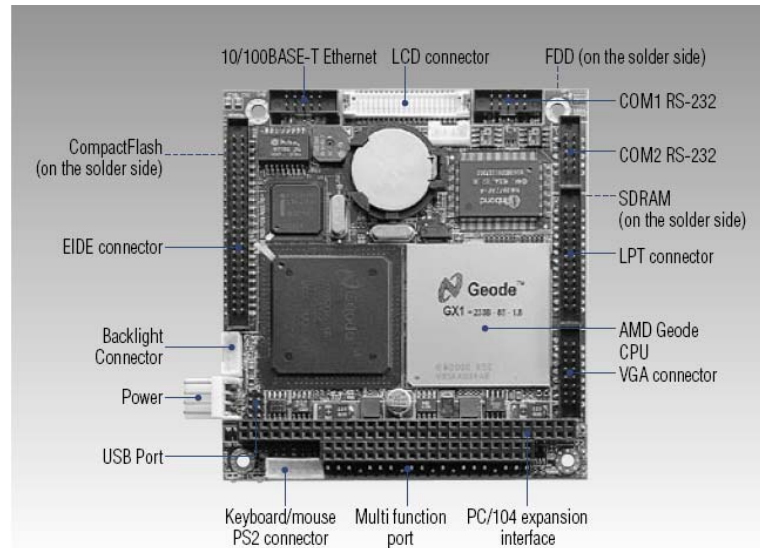


Figure 4.14 PCM 3350 type PC104 embedded PC.

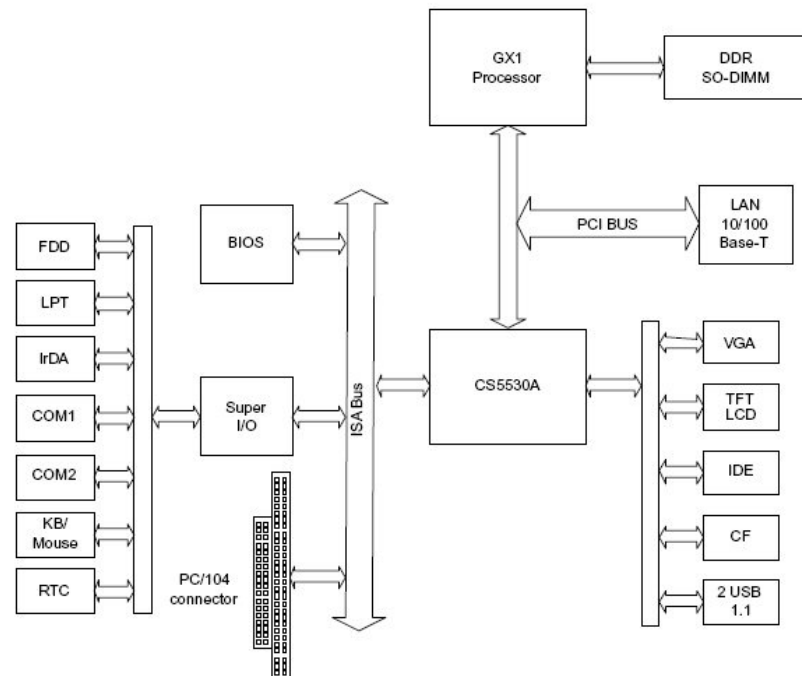


Figure 4.15. Block diagram of PCM 3350.

In addition to PCM-3350, an analog output card (PCM-3712-A) and a digital I/O and analog input card (PCM-3718H-B) is mounted on the PCM-3350 PC104 module. For features of the cards, please see Table 4.3.

Table 4.3 Features of PCM 3712 and PCM 3718 daughterboards (PCM-3712, PCM-3718H/HG, 2007).

Features of PCM 3712	
<ul style="list-style-type: none"> • 2 channel analog output • 0 to 5 V, 0 to 10 V, ± 2.5 V, ± 5 V, ± 10 V and 4 to 20 mA output range • 12-bit resolution • High speed • Single power (+5 V) operation • Output cut off at power on 	
Features of PCM 3718H	
<p>Analog Input:</p> <ul style="list-style-type: none"> • Channels: 16 single-ended or 8 differential, jumper selectable • Resolution: 12 bits • Input range: (software programmable, V_{DC}) <p>PCM-3718H</p> <p>Bipolar: ± 10, ± 5, ± 2.5, ± 1.25, ± 0.625</p> <p>Unipolar: $0 \sim 10$, $0 \sim 5$, $0 \sim 2.5$, $0 \sim 1.25$</p> <p>PCM-3718HG</p> <p>Bipolar: ± 10, ± 5, ± 1, ± 0.5, ± 0.1, ± 0.05, ± 0.01, ± 0.005</p> <p>Unipolar: $0 \sim 10$, $0 \sim 1$, $0 \sim 0.1$, $0 \sim 0.01$</p>	<p>Digital Input/ Output:</p> <ul style="list-style-type: none"> • Channels: two 8-bit • Level: TTL compatible • Input voltage: <p>Logic 0: 0.8 V max.</p> <p>Logic 1: 2.0 V min.</p> <ul style="list-style-type: none"> • Output voltage: <p>Logic 0: 0.33 V max. @ 6 mA (sink)</p> <p>Logic 1: 3.84 V min. @ 6 mA (source)</p>

4.2.4.2 Microcontrollers on the Mobile Robot Model

Peripheral Interface Controller, shortly PIC, is the name of the microcontroller made by Microchip. It was developed to control peripheral devices as can be understood from its name. It is alleviating the load from the main programming unit. Four PICs (PIC16F877 and PIC16F84) are installed on the experimental mobile robot model. The tasks of these microcontrollers may be seen in Figure 4.16. The features of these two types of PICs may be found in Appendix 4, Schematic 7, Schematic 8 and Schematic 9.

None of two motors have the same exactly same characteristics of each other. So, always a motion and velocity feedback is needed to correct errors to each other. Two encoders are constructed and mounted on the wheels of the AMR in initial stage of the project. These encoders are connected to a microcontroller (initial stages of the project a PIC16F84, then a PIC16F877) in order to have similar characteristics for left and right motion. This microcontroller is named as ‘Microcontroller 3’ in this project. Microcontroller 3 also sends the counted pulses via RS-232 to either ‘Microcontroller 1’ or PC104 for calibration, control and telemetry.

4.3 Block Diagrams

4.3.1 First Models of the Robot

In developing stages of the project, many changes are made to the control model; sometimes to improve the model, sometimes to adapt the model in use of daily life and sometimes because of limited project budget. Some changes in developing stages of the project can be seen in Appendix 4 Schematic 9.

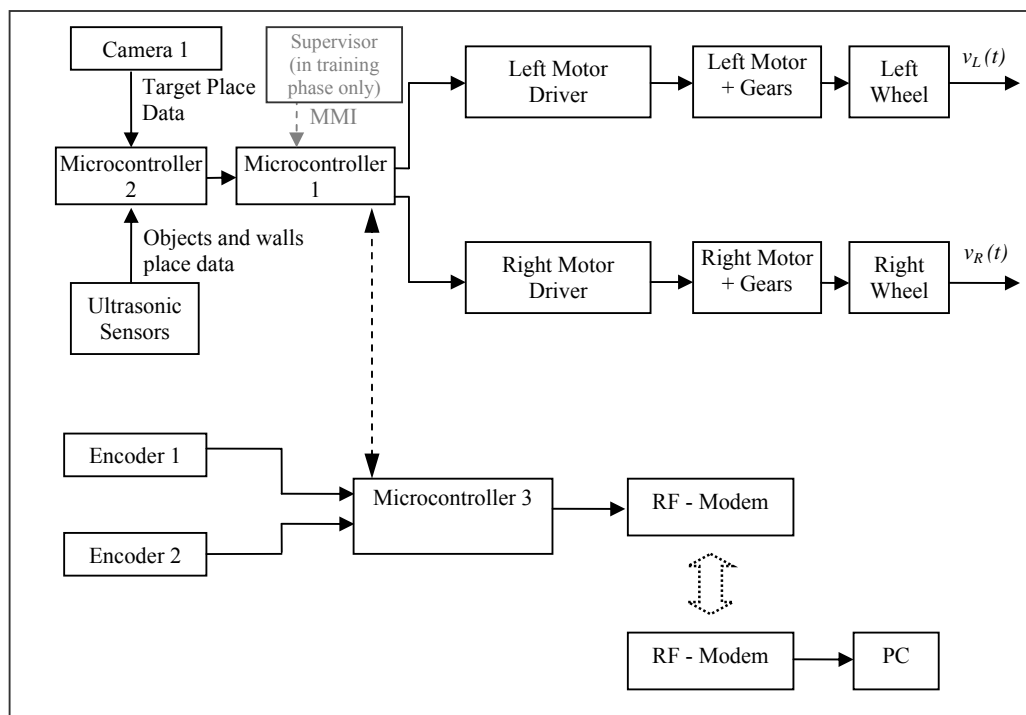


Figure 4.16 Block diagram of the model.

4.3.2 Control of the System

In many researches, mapping algorithms are tested for having knowledge of the environment (Beckerman, 1990; Awad, 2004). However, in this research focus is on more reliable data from sensors aiming the movement of the mobile robot quick and reasonable.

Two different control algorithms and performances were tested on the mobile robot. As mentioned before, an embedded PC (PC104) with digital and analog I/O's is mounted on the model for neural networks algorithm whereas two microcontrollers are mounted to test fuzzy logic algorithm. Control block in system block diagram in Figure 4.16 is Microcontroller 1 block that also demonstrates embedded PC in neural network algorithm experiments. Sensor fusion algorithm is loaded on this microcontroller also. Microcontroller 3 block is a modular unit which interfaces for the self-built encoders with RS232 standard to communicate either with embedded PC or the Microcontroller 1. Later, another PIC is used for ANN algorithm.

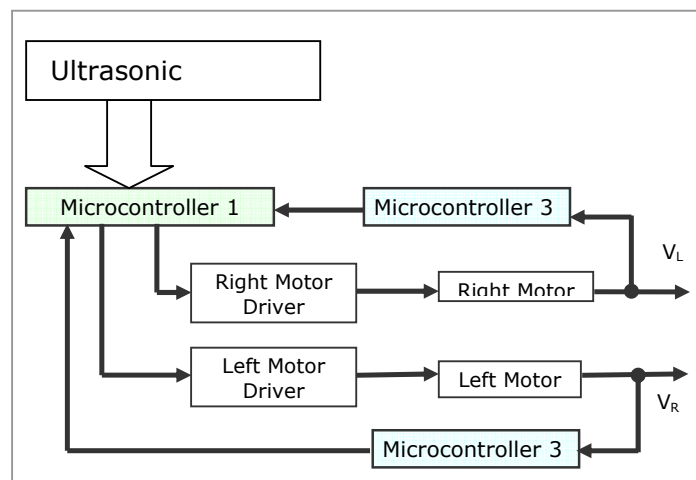


Figure 4.17 Control blocks of the mobile robot in DSET experiments.

For continuous control of the right and left side motors, equation (4.9) and (4.10) are used in algorithm. These equations are also used in forming fuzzy logic algorithm.

$$v_R = v_{BR} - (S_{VL} \times R_{VL}) \times K_R \quad (4.9)$$

$$v_L = v_{BL} - (S_{VR} \times R_{VR}) \times K_L \quad (4.10)$$

v_R : Right motor velocity

v_L : Left motor velocity

v_{BR} : Output Value for Right Motor Max Velocity (for Calibration)

v_{BL} : Output Value for Left Motor Max Velocity (for Calibration)

- S_{VL} : Sensor Value for Left (3 for S3 and S4, 2 for S2 and S5, 1 for S1 and S6)
 S_{VR} : Sensor Value for Right (3 for S3 and S4, 2 for S2 and S5, 1 for S1 and S6)
 R_{VL} : Region Value for left (Closeness factor)
 R_{VR} : Region Value for right (Closeness factor)
 K_R : Right Motor Velocity Factor
 K_L : Left Motor Velocity Factor
 X_1 : Closeness Factor for Region 1 (It is close.)
 X_2 : Closeness Factor for Region 2 (It is far.)
 $q_{3 \times 1}$: Location matrix for mobile robot.

4.4 Dempster – Shafer Evidence Theory Experiments

In forming control algorithms of continuous or fuzzy logic control, front region of the mobile robot is divided into 12 regions with respect to their closeness probability to unexpected objects and motion direction. With this division, a compact, simple algorithm, so quick motion for AMR is aimed. Dempster – Shafer evidence theory was tested in limitations of quick motion with reliable sensor data. In Table 4.4 sensor data from ultrasonic sensors and processed data can be seen.

Table 4.4 Sensor Data from Ultrasonic Sensors

	Unprocessed data	Processed data (Bayesian - Occupied)	Processed Data (Dempster – Shafer / Occupied)
...
T_n	6,4,6,0,4,6	0.29, 0.14, 0.79, 0.00, 0.14, 0.29	0.250, 0.339, 0.429, 0.232, 0.143, 0.250
T_{n+1}	6,4,6,0,4,6	0.29, 0.14, 0.79, 0.00, 0.14, 0.29	0.250, 0.339, 0.429, 0.232, 0.143, 0.250
T_{n+2}	6,4,6,0,4,6	0.29, 0.14, 0.79, 0.00, 0.14, 0.29	0.250, 0.339, 0.429, 0.232, 0.143, 0.250
T_{n+3}	4,4,6,0,4,6	0.46, 0.14, 0.79, 0.00, 0.14, 0.29	0.384, 0.384, 0.429, 0.232, 0.143, 0.250
T_{n+4}	7,4,6,0,4,6	0.46, 0.14, 0.79, 0.00, 0.14, 0.29	0.384, 0.384, 0.429, 0.232, 0.143, 0.250
T_{n+5}	7,4,6,0,4,6	0.46, 0.14, 0.79, 0.00, 0.14, 0.29	0.384, 0.384, 0.429, 0.232, 0.143, 0.250
T_{n+6}	7,4,6,0,4,6	0.46, 0.14, 0.79, 0.00, 0.14, 0.29	0.384, 0.384, 0.429, 0.232, 0.143, 0.250
T_{n+7}	4,4,6,0,4,6	0.46, 0.14, 0.79, 0.00, 0.14, 0.29	0.384, 0.384, 0.429, 0.232, 0.143, 0.250
T_{n+8}	4,4,6,0,4,6	0.46, 0.14, 0.79, 0.00, 0.14, 0.29	0.384, 0.384, 0.429, 0.232, 0.143, 0.250
...

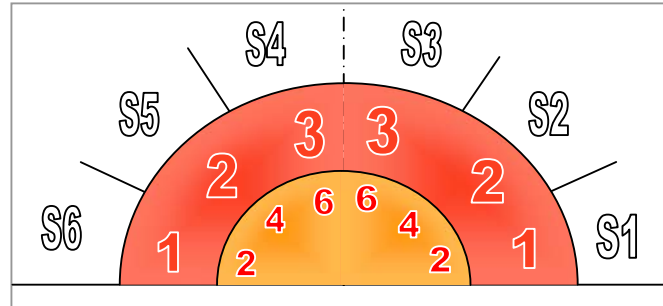


Figure 4.18 Regions for algorithm.

Translating sensor readings into probabilities and combining probabilities using Bayes Rule is one of the most known methods. For localization and map making, other two most popular methods are DS Theory and HIMM (Murphy, 2000). In Bayes Rule of evidence there are two parts evaluating the space empty or occupied. However, if there is spaces that can not be classified as occupied or empty although having signals from sensors, Dempster – Shafer Theory has to be used. If we simplify the rules, in Bayesian Theory, the sum of probability of a space to be occupied and the probability of a space to be empty is equal to one whereas in Dempster – Shafer Theory, the sum of the probability of a space to be occupied, to be empty and not to be known is equal to one. The main purpose is to combine different sensor data as being different sensors or as for the same sensor in different times. This is used in this research for the mobile robot to find a path itself as quickly and as reasonable as possible.

In Figure 4.19 and in Figure 4.20, unprocessed and processed signals for algorithm during the robot movement are shown respectively. When the robot comes a point that most of the signals are similar, it is easy for it to decide the direction since the processed signal values includes different sensors and different times. This also improves the quality of the path which can be easily seen in Figure 4.20. In this figure, the edges or the important places have greater values after using DS-Theory. In color scale of the processed sensor data values, they can be seen darker in color in this figure.



Figure 4.19. Unprocessed signals from ultrasonic sensors which are taken during movement of the mobile robot in a corridor.



Figure 4.20. Processed signals using DS Theory from ultrasonic sensors which are taken during movement of the mobile robot in a corridor.

To simplify the optimal path problem, smoothing the path curve is the first step. Let a discontinuous curve denoted as Type 0. A continuous curve whose tangent direction is discontinuous is called Type 1. The set of continuous curves whose tangent direction is also continuous is Type 2. A curve has curvature continuity as well as tangent direction continuity is called Type 3. Third type curves are considered “smoother” than the previous classes. See Figure 4.21. This type curve is said to be optimal motion for a mobile robot. Performance of the control algorithms and improvement with Dempster – Shafer evidence theory were also tested on this matter.

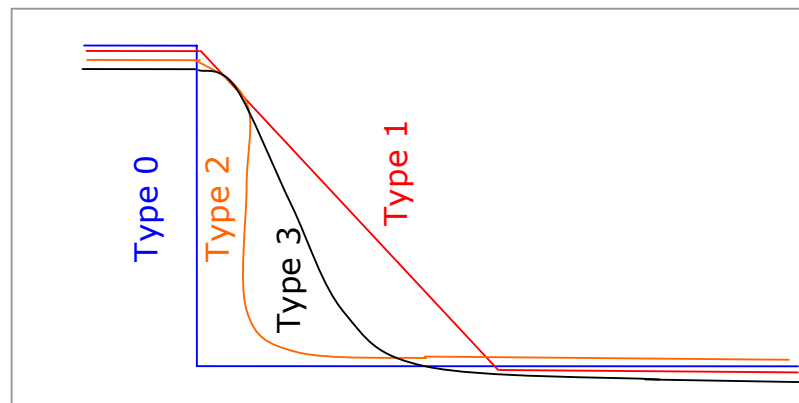


Figure 4.21 Smoothness of curves of different types

4.5 Fuzzy Logic Experiments (Find Target)

4.5.1 Fuzzy Logic Controller Experiment 1

Two basic well known problems of the autonomous mobile robots are finding empty space and heading target problems. These problems are the subjects of many researches. In developing solutions to these problems, two main research areas are formed. These are mapping environment as well as the location of the robot and sensor processing for finding target and heading (Janglova, 2004; Kopacek, 2006; Yi, 2000; Montaner, 1998).

The first part of fuzzy logic controller experiments is based on finding target and optimal behavior of movement of robot towards the target. Environment in heading

to the goal experiments can be seen in Figure 4.22. For simplifying the heading problem, x axis value of target on image is taken as input and left and right side motor velocities are taken as outputs first.



Figure 4.22 Environment in heading to the goal experiments and modeling environment using camera images.

The distance between the starting point and the goal is 20 meters. Camera is mounted on the central axis of the mobile robot. Microcontroller 2 receives the x_p value which means the location of the target on x axis on image from the mobile robot camera 1. Using this information robot determines the left and right motor velocities due to the fuzzy rules (see Table 4.5).

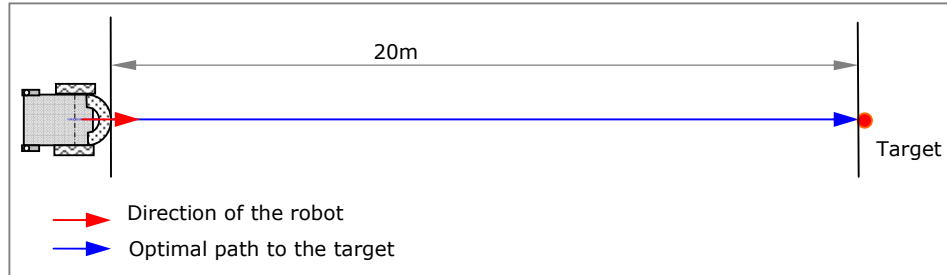


Figure 4.23 Experiment 1 scenario: Mobile robot finds the target and heads towards the target.

After getting the location data of the target, the next stage is Fuzzification process (see Figure 4.24).

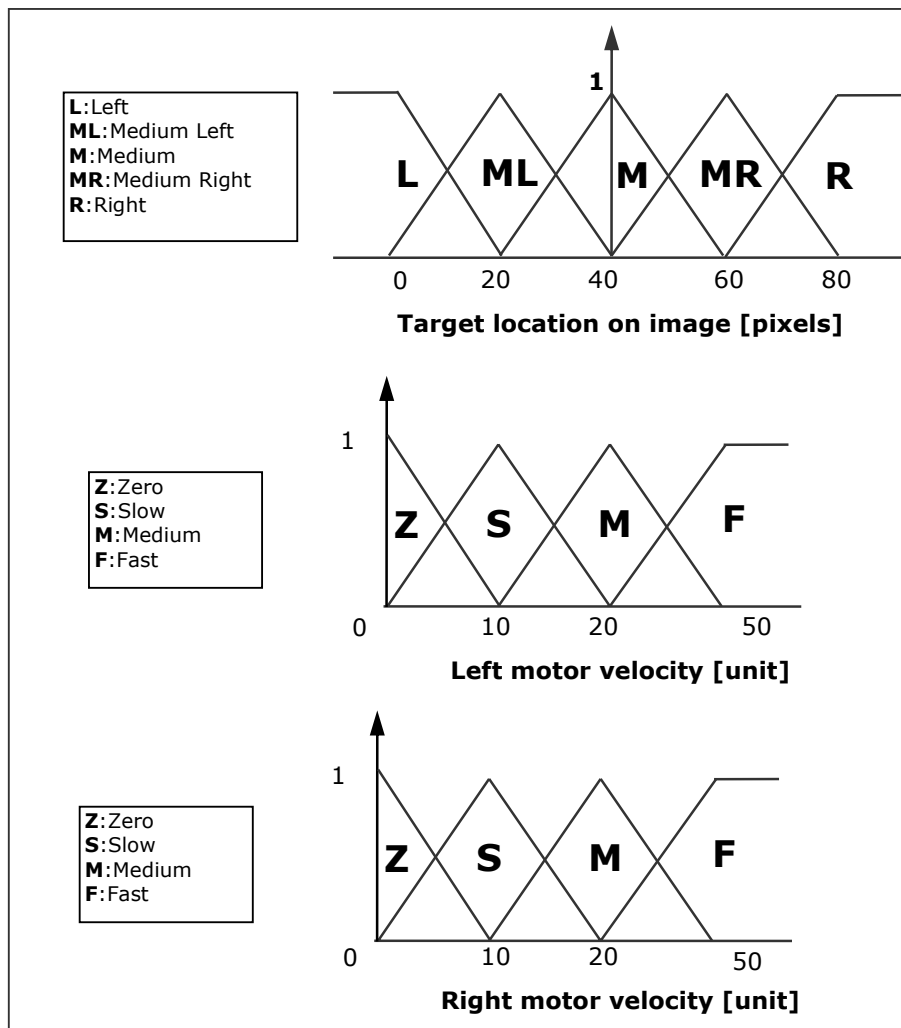


Figure 4.24 Fuzzy sets for the linguistic variables target on image left and right side motor velocities (Fuzzification).

Creating rules due to the graphics in Fuzzification process is the next stage. Rules in code on microcontroller 2 during the experiment 1 can be seen in Table 4.5. Outputs of the system can be seen in Figure 4.24. In each experiment, mobile robot reached the target successfully. But, in some experiments, the followed path on the way to reach the goal was not the best solution as can be seen in Figure 4.25. The reasons will be discussed in conclusion chapter of this thesis.

Table 4.5 Fuzzy rules on microcontroller program.

```

...
rref = 0; lref = 0

If cmu_mx <= 40 Then ' Fuzzy Table-Left
l_cmu = ((40 - cmu_mx) * 100) / 40
rref = rref + (20 * l_cmu) / 100
End If

If cmu_mx >= 40 Then 'Fuzzy Table-Right
r_cmu = ((cmu_mx - 40) * 100) / 40
lref = lref + (20 * r_cmu) / 100
End If
...
'Fuzzy Table Middle
If cmu_mx >= 20 And cmu_mx <= 60 Then
If cmu_mx < 40 Then
m_cmu = ((cmu_mx - 20) * 100) / 20
Else
m_cmu = ((60 - cmu_mx) * 100) / 20
End If
rref = rref + (50 * m_cmu) / 100
lref = lref + (40 * m_cmu) / 100
End If
...

```

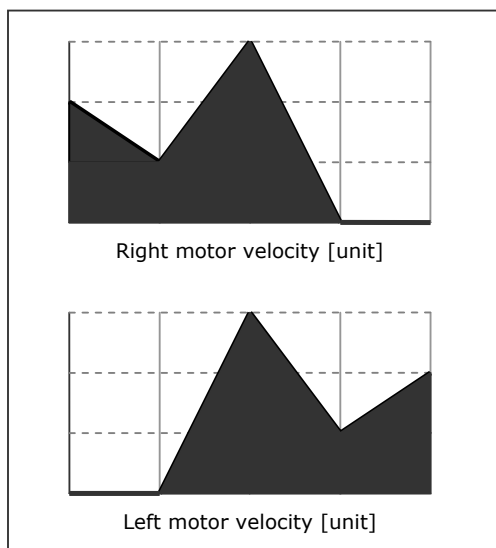


Figure 4.25 Outputs of the FLC experiment 1.

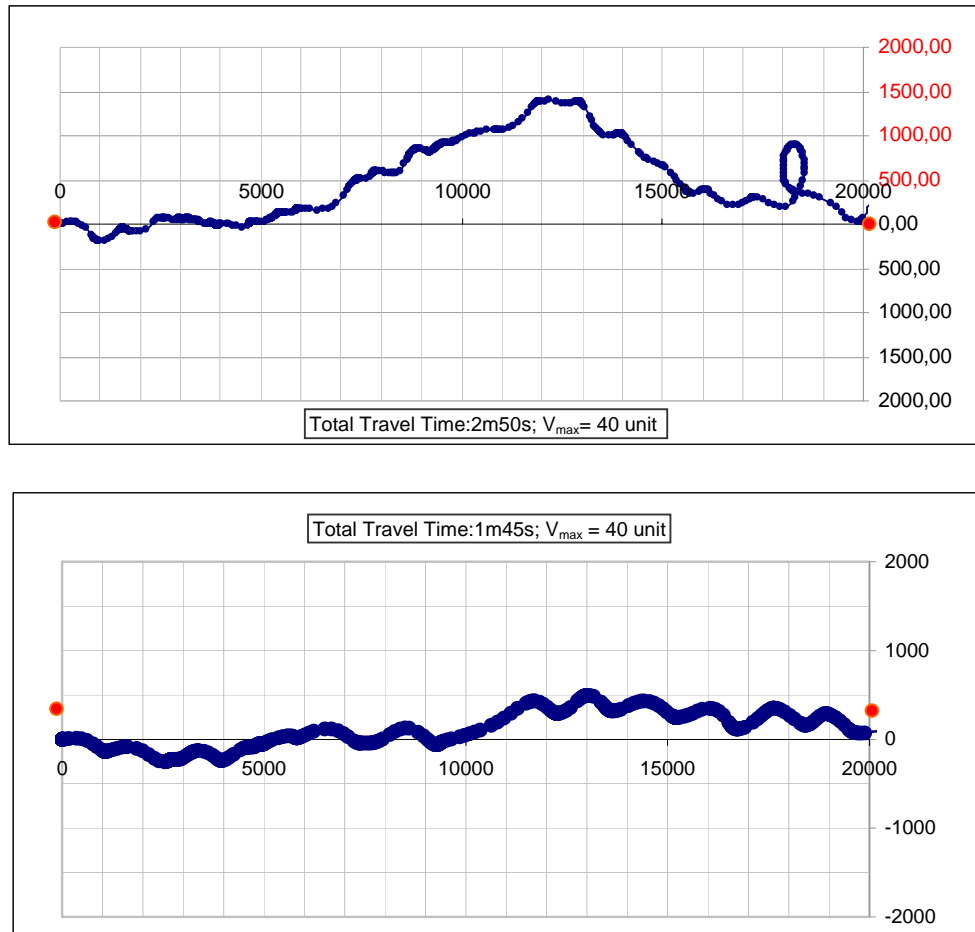


Figure 4.26 Fuzzy logic controller experiment 1 (Graphs are the followed real paths of the AMR)

4.5.2 Fuzzy Logic Controller Experiment 2

The goal of second FLC experiment is to find the efficiency of the seeking mode code and motion of the autonomous mobile robot. AMR has to find its path to the target starting without knowing the place of the target and without having chance to see the target. The program code is the same of the code in experiment 1 (which can be examined in Appendix 5).

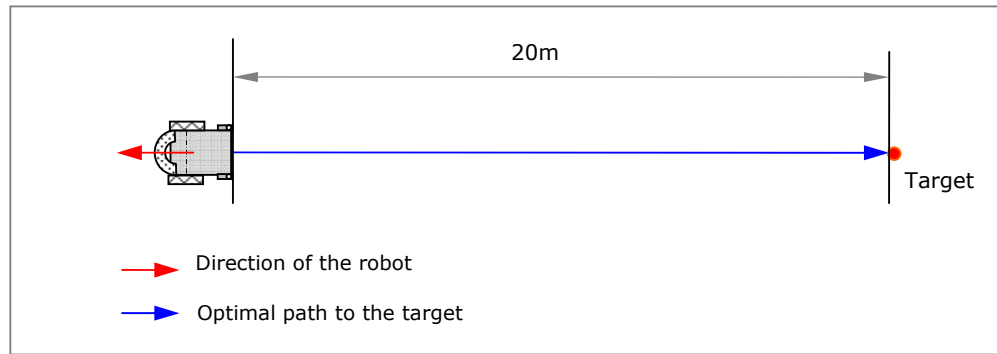


Figure 4.27 FLC experiment 2 scenario: Robot seeks the target, finds it and heads towards the goal.

Seeking mode and heading of the AMR is really effective. It also adapts itself due to the light intensity without losing the location information of the target during adaptation. AMR can be programmed to track not the color but the exact color of the target.

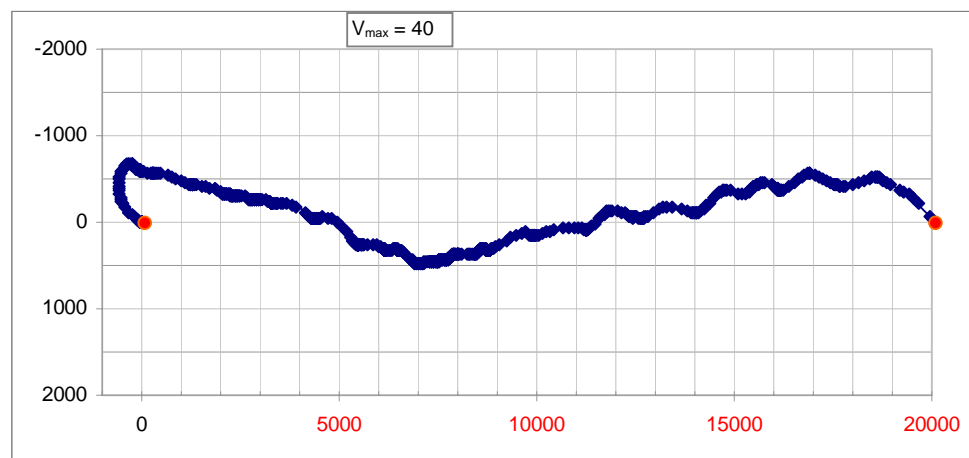


Figure 4.28 FLC Experiment 2 results (real navigation trajectory). Data are taken via telemetry.

4.5.3 Fuzzy Logic Controller Experiment 3

This experiment environment is similar with the experiment 1. However, the rules are changed to make the control region boundaries sharper. This change made the behavior of the robot less optimal. Because, big sudden changes in boundaries of regions causes robot to have worse behavior. The other problem in this experiment is to design the FLC behavior for the left motor and right motor velocities of the

mobile robot when turning left or right while going forward. In this condition, the error is small, so the response is not enough. But, the desired response is to have the control of interference even in very small angles in order to have a very stable behavior (see Figure 4.29). These two disadvantages made the behavior of the robot worse which can be seen also in Figure 4.31.

Fuzzy rules: Let us define the location of the target on x axis as x_p , maximum velocity of the robot is V_{max} ,

1. If $x_p < 20 \rightarrow v_L = 0 ; v_R = v_{max} \cdot \frac{x_p}{80}$
2. If x_p is between 20 and 30 $\rightarrow v_L = \frac{2 \cdot V_{max}}{3} ; v_R = \frac{2 \cdot V_{max}}{3} - \frac{2x_p}{80} \cdot V_{max}$
3. If x_p is between 30 and 50 $\rightarrow v_L = v_R = V_{max}$
4. If x_p is between 50 and 60 $\rightarrow v_L = \frac{2 \cdot V_{max}}{3} - \frac{2x_p}{80} \cdot V_{max} ; v_R = \frac{2 \cdot V_{max}}{3}$
5. If $x_p > 60 \rightarrow v_L = V_{max} \cdot \frac{x_p}{80} ; v_R = 0$

Rules above are found after Fuzzification that can be seen in Figure 4.30.

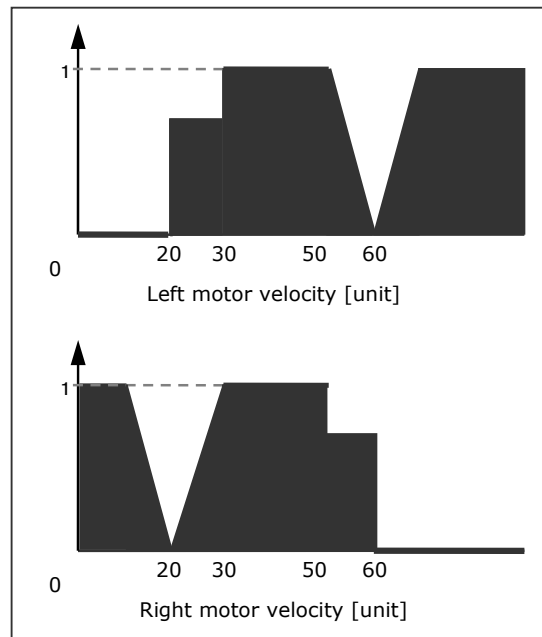


Figure 4.29. Defuzzification and outputs of the system.

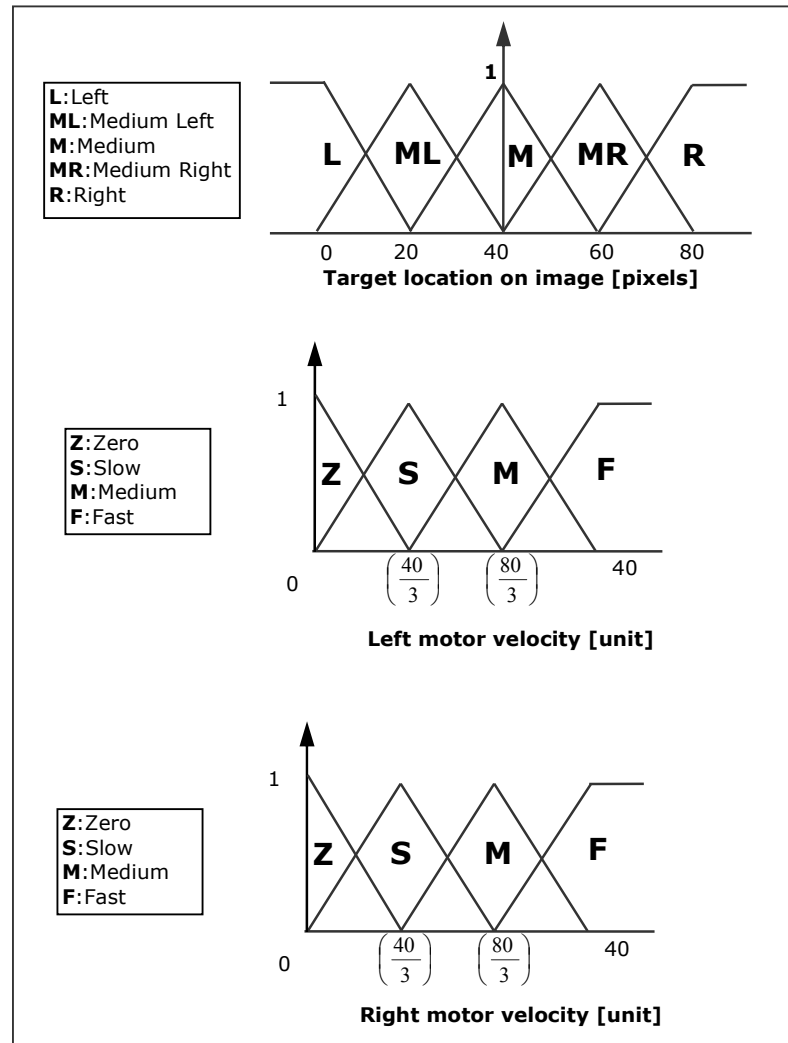


Figure 4.30 Fuzzy sets for the linguistic variables target on image left and right side motor velocities (Fuzzification).

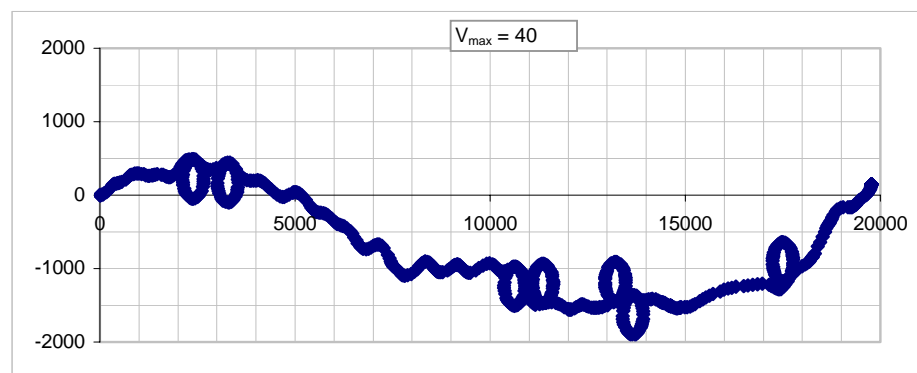


Figure 4.31 Movement of the AMR during one of first type of experiment using fuzzy logic.

4.5.4 Fuzzy Logic Controller Experiment 4

This time, the outputs for the left and right motors in program are set not to give zero output at any time, in other words, continue running even sharp rotating conditions. This reduces the effects of starting impacts to model, in other words makes motion smoother. Another difference is dividing the regions of outputs of fuzzy control into less number of regions (see Figure 4.32). This improves the continuous motion behavior, so the robot follows the optimal path in a narrow tolerance band (see Figure 4.33). The error at the end of the path in this figure comes from the lens effect of the light source. When the robot comes very near to the target, lens changes the sensed color of the target into a very bright color via increasing the intensity of the light, which is seen white by the camera. So, in this condition, robot seeks for the target. This is not accepted as a fault of the robot, but fault of selecting the light source of the target. However, it is obviously seen from this figure that even in this condition, robot finds the target successfully.

Simplified fuzzy rules in code for finding target are:

1. If $0 < x_p \leq 20 \rightarrow v_L = 5; v_R = V_{\max}$
2. If $20 < x_p \leq 30 \rightarrow v_L = 3(x_p - 20) ; v_R = V_{\max}$
3. If $30 < x_p \leq 50 \rightarrow v_L = v_R = V_{\max}$
4. If $50 < x_p \leq 60 \rightarrow v_L = V_{\max} ; v_R = 3(60 - x_p)$
5. If $x_p > 60 \rightarrow v_L = V_{\max} ; v_R = 5$

In evaluating the experimental results with figures in this chapter, it should be remembered that the distances in figures are in millimeters and the width of the robot is approximately 65 cm. Another note is that the velocity unit is not m/s or km/h but a unit that is defined in the program. This unit is the same unit in all experiments. The reason is to make the programming easier.

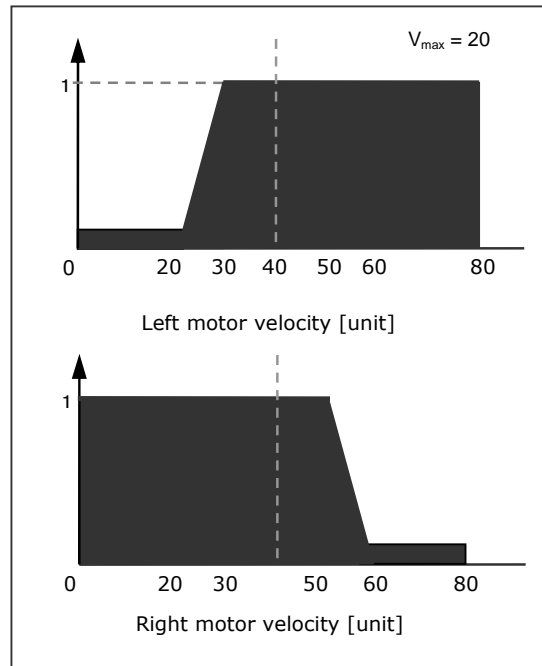


Figure 4.32 Velocity outputs of the FL model.

The figure above shows the outputs of the code whereas the figure below is the path in fuzzy logic experiment 4 (finding target problem, optimizing regions and robot motion on reaching the target.)

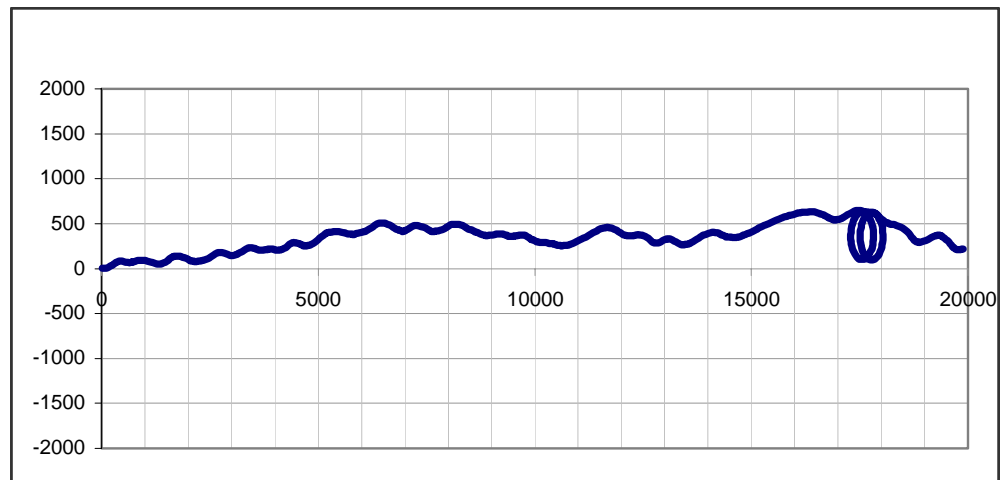


Figure 4.33 Motion of the AMR in one of first type of experiments using fuzzy logic (find and head to target problem).

4.5.5 Fuzzy Logic Controller Experiment 5

Fuzzy logic experiment group 5 is related with the corridor finding and obstacle avoidance with fuzzy logic controller code. In the first stage of this experiment group, robot is tested for finding the exit of different corridors. One of the experiments is shown in Figure 4.34 (a). In most of the experiments in this group robot failed to avoid the obstacles or could not find the exit. Changing the combined fuzzy rules of finding the target and obstacle avoidance some time later made the program very long and so the robot motion very slow. Relatively successful second part of this group experiments (Figure 4.34-b) can be seen in Figure 4.35. This results, however, very far away from neuro-fuzzy experiments results success, which will be told in following parts of this chapter.



(a)



(b)

Figure 4.34 Initial and final stages of the experiments on obstacle avoidance.

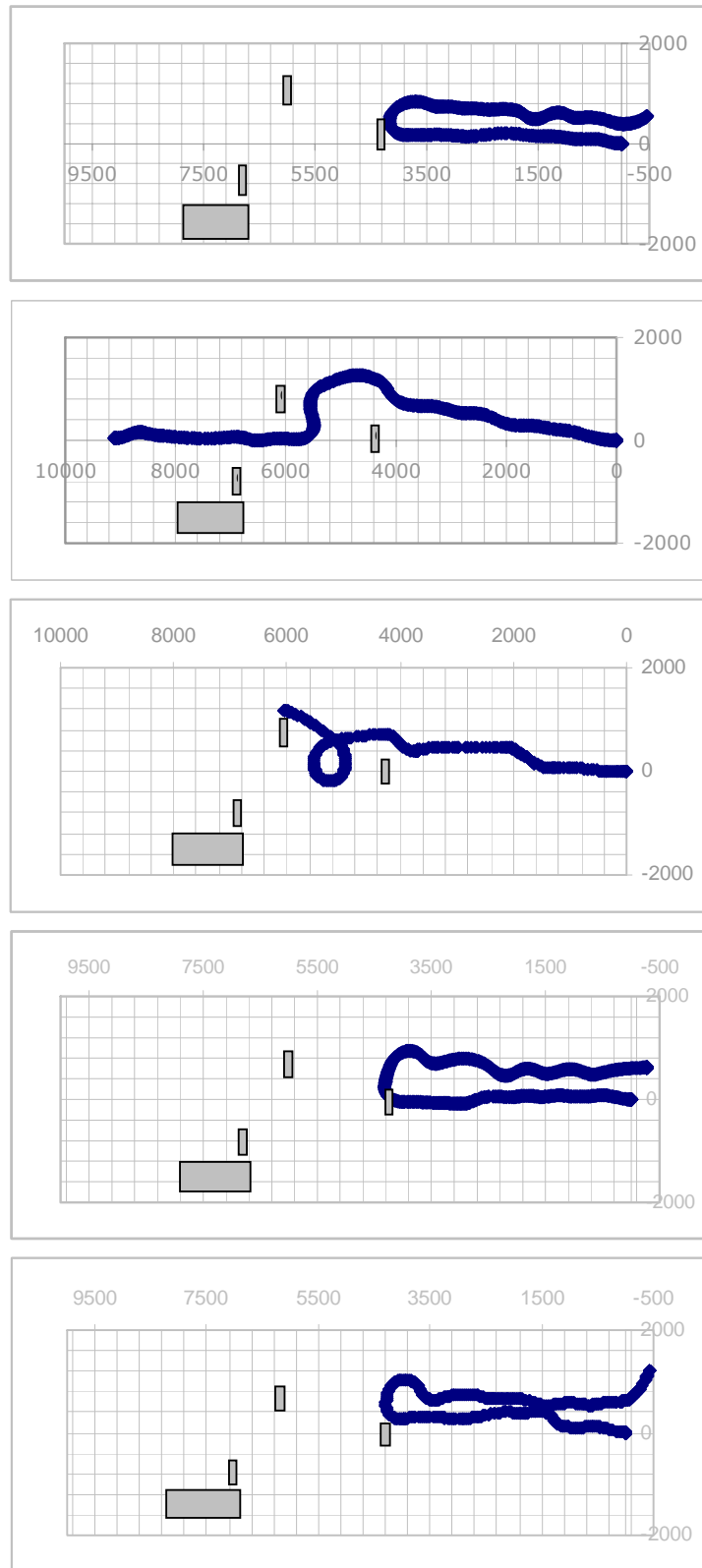


Figure 4.35 Fuzzy logic experiments, actual followed path.

4.6 Artificial Neural Networks Experiments (Find Empty Space)

The perceptron can be seen as the simplest kind of feedforward, supervised neural network. The perceptron is a kind of linear binary classifier that maps its inputs x to an output value $f(x)$. x is a binary vector whereas $f(x)$ is a single binary value. It is calculated as:

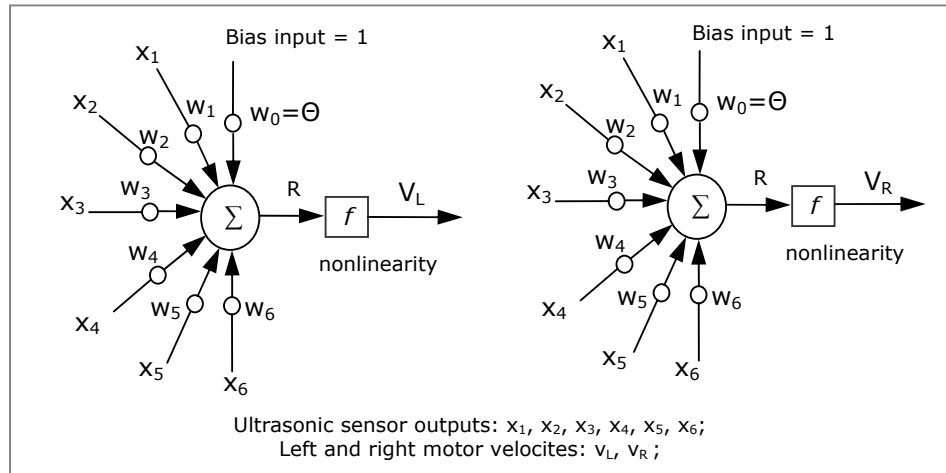


Figure 4.36 Perceptron model of the AMR.

$$f(\text{net}) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{else} \end{cases} \quad (4.11)$$

However, the outputs of the nets are binary, but the optimal velocities of the autonomous wheeled robot model should not be. For this reason, ADALINE NN model is thought to be more suitable for experimental AMR model. As mentioned in Chapter 2, ADALINE model is supervised and feedforward. It is a single neuron whose weights are updated according to LMS (Least Mean Square) algorithm. The LMS algorithm is an adaptive algorithm that computes adjustments of the neuron synaptic weights. ADALINE, as mentioned, is a single neuron. Barely, the outputs of the AMR model are not just one single output. Left and right motor velocities, which are the outputs of the system, have to be evaluated as one vector. Each input (ultrasonic sensors outputs) should have different weights in order to decide the heading precisely. So, the MADALINE model may be more appropriate for the model (see Figure 4.37 and Figure 4.38).

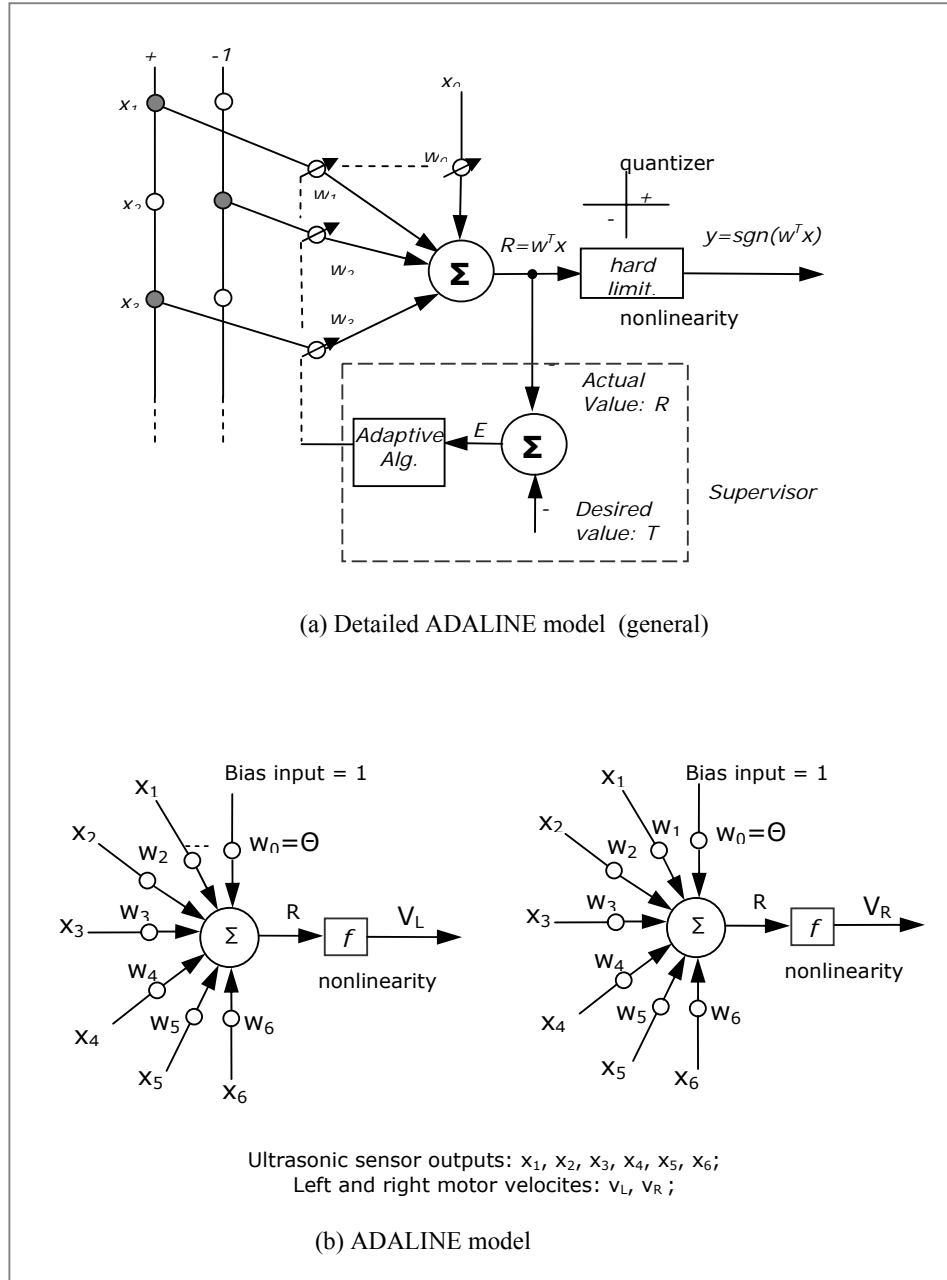


Figure 4.37 ADALINE model of the AMR.

In training phase of the experiments, initial weights are predicted. Supervisor decision is for refusing the motion of the AMR. If y is positive, which means left and right side systems are running, and d is also positive, which means the motion is approved, then the weights do not change (see Figure 4.38). The reason of getting supervisor decision only for refusing is to make the weights optimal in reasonable

time. After training, the followed paths can be seen as in Figure 4.39 and following figures.

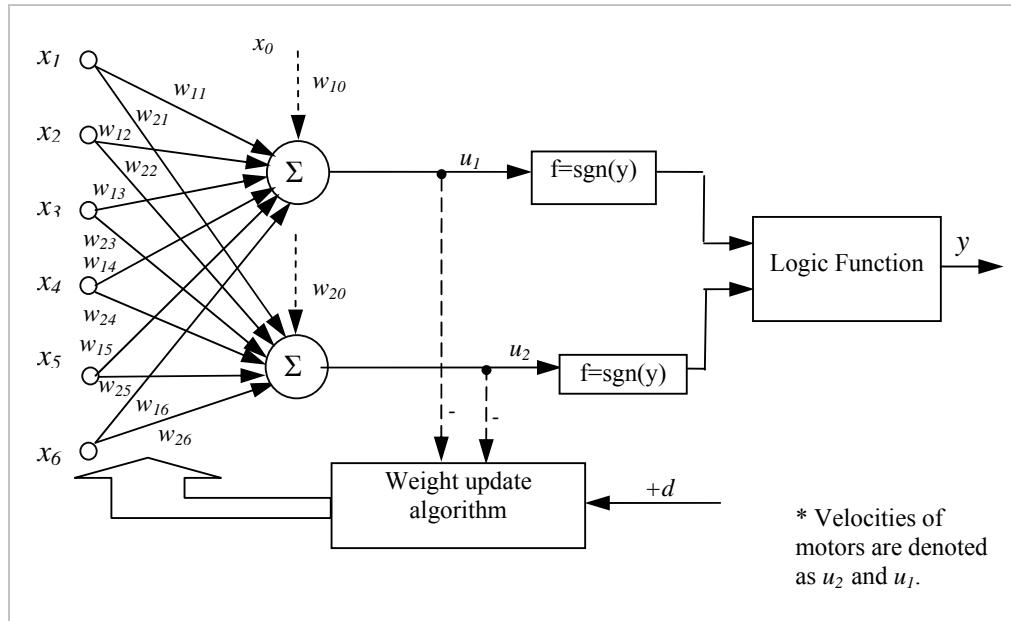


Figure 4.38 MADALINE model of the AMR.

4.6.1 ANN and Fuzzy Logic Combined Experiments

Experiment scenario for the model is decided as finding the target and avoiding the obstacles on the way during the motion. Target has to be recognized with any characteristic(s) of it. In this scenario, target is a led which is red in color. But, the difficulty of the task for finding the target is to discover the red led in various light intensities and contrast it from other red colored objects. Despite these difficulties, this task is completed very successfully. Programming the camera at RGB format in each color scale of 255, the target can be attracted successfully.

The second task is to make the AMR head to target autonomously. This is controlled via fuzzy logic algorithm. The main idea for this was: Since the target could be recognized successfully, fuzzy logic algorithm will make the control faster. The result was as expected, naturally after many optimization experiments which are explained in fuzzy logic experiments part of this chapter.

The third task, and the most difficult one, is to avoid the obstacles on the path. The AMR model has to notice the obstacles during motion and pass them with a smooth curve and then return back to its optimal path. Ultrasonic sensors are mounted on the front 180 degrees part of AMR model for this purpose. Although the ultrasonic sensors are physically independent from each other as for transceiver parts, control units, and as for scanned regions, signal of them can be reflected from walls and obstacles or effect each other using the body of the model. Isolating the ultrasonic transceiver signals as transferring through the model body is the simplest problem. It can be isolated using materials with appropriate softness that the signals would not go through. Finding solution for the reflections from other objects and walls problem, however, is more difficult. Dempster- Shafer Theory is studied in order to discover the important places on the path. This research, which had also successful results, was told in related part of this chapter. On the other hand, adding DS theory code with fuzzy logic control and NN control algorithms would make the code slower and the AMR model not to response in reasonable time intervals. And NN has also changeable weights, which can take the task of DS theory for this research model. NN model selected after studying several NN architectures and models, which are mentioned in Chapter 2.

First experiments with neural networks is using ANN for object avoidance and heading via fuzzy logic. The following figures show that control of the robot with neural networks even with bias values is more successful than the previous experiments which only fuzzy logic used in this research. After first group of experiments (see Figure 4.40), in second group of experiments, the NN tried to be trained. In this group, path following performance of robot is investigated due to the changing weights (please see Table 4.6 and Figure 4.41).

After study on the model, control algorithms and application of the theoretical control solutions on the model, the optimal simplified solution for the model is found as in Figure 4.39 below.

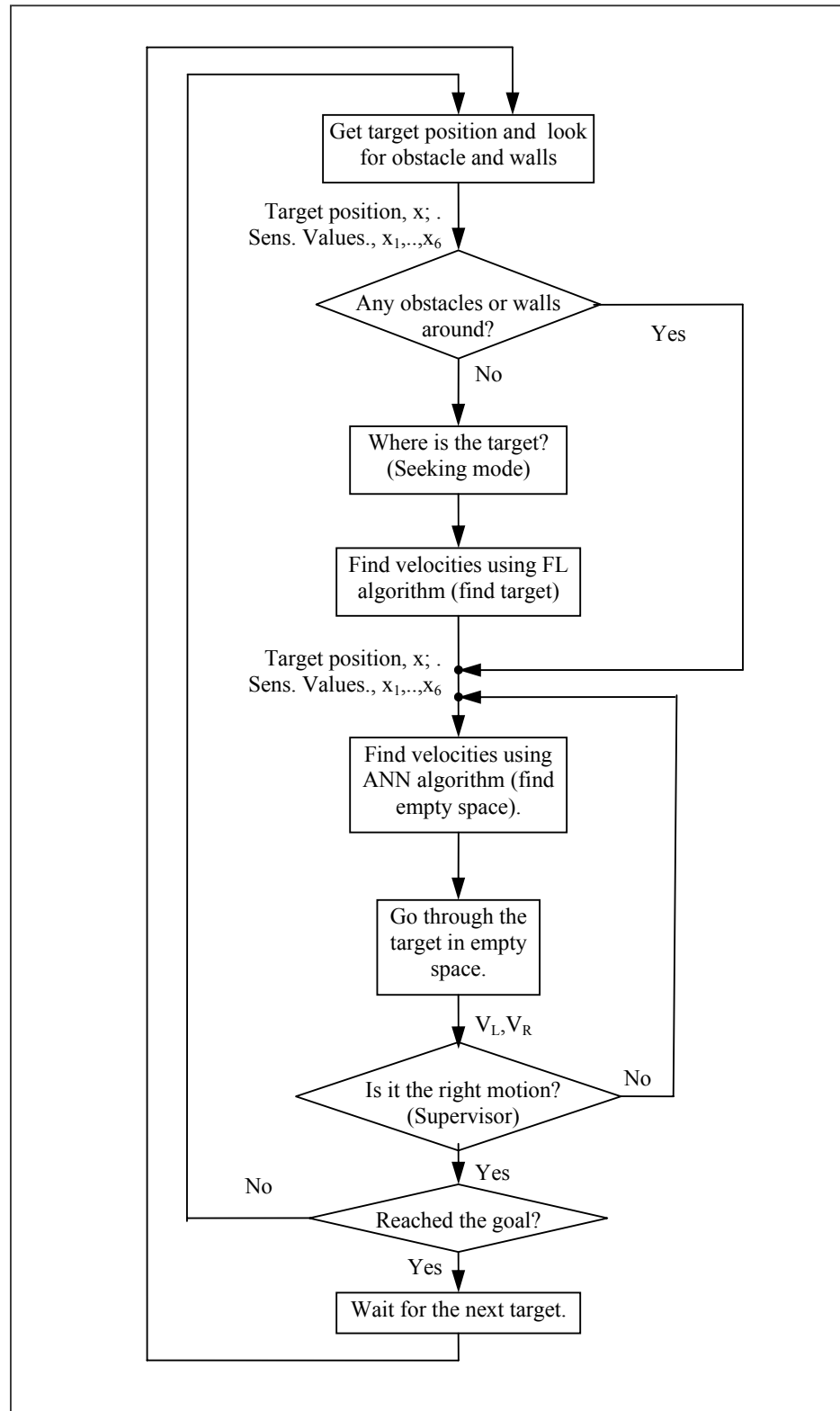


Figure 4.39 Simplified control algorithm of the model.

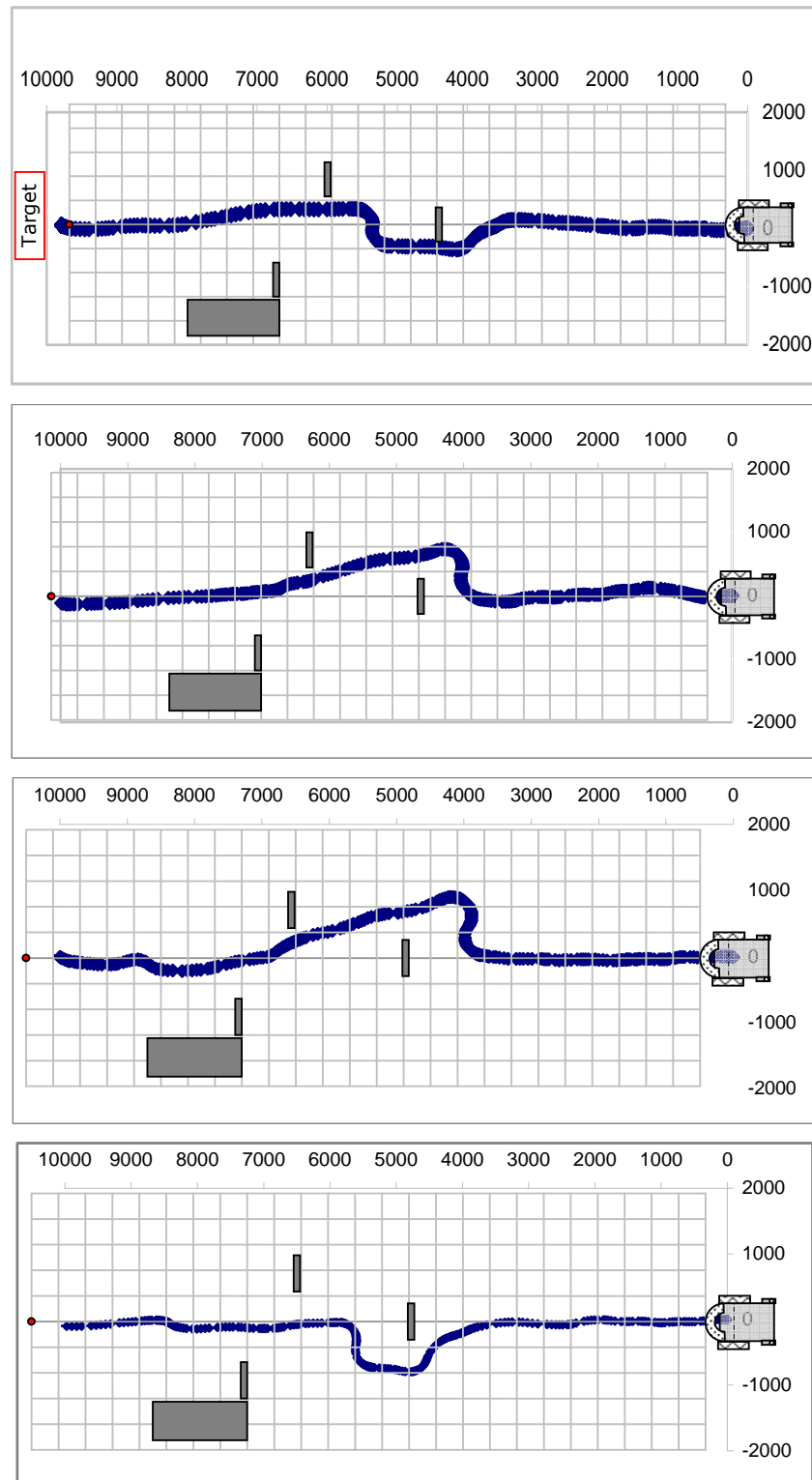
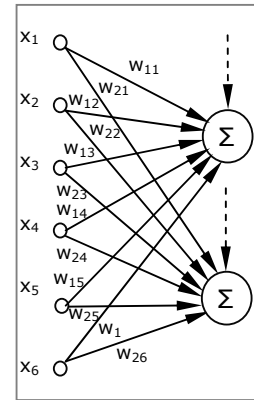


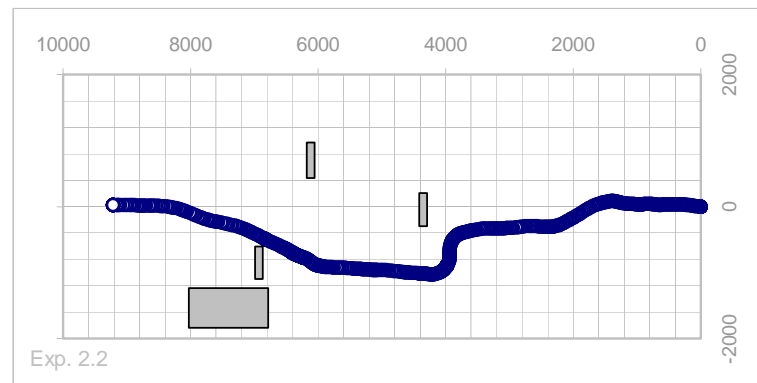
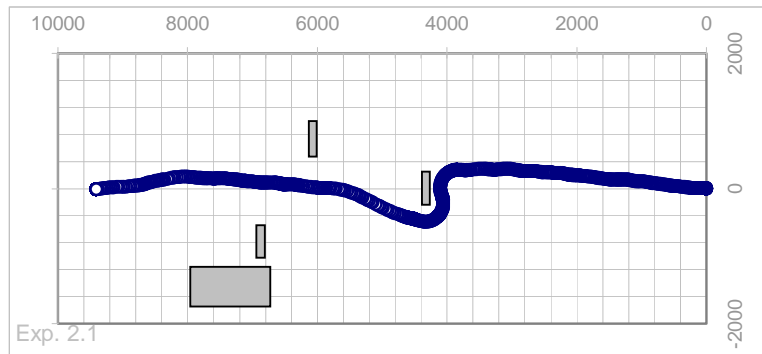
Figure 4.40 Target heading with object avoidance (I^2C protocol made the sampling time faster and so the path is more optimal). Dimensions of the AMR in figures are bigger than real dimensions. First groups of experiments with NN.

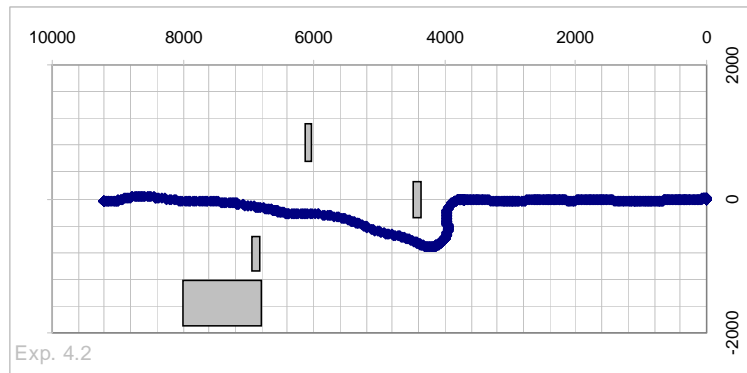
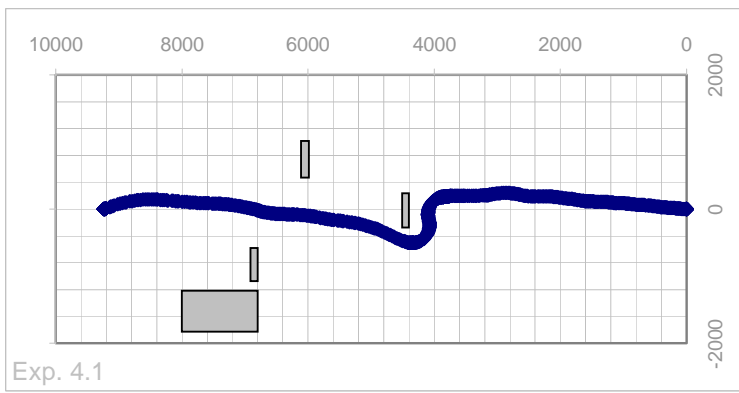
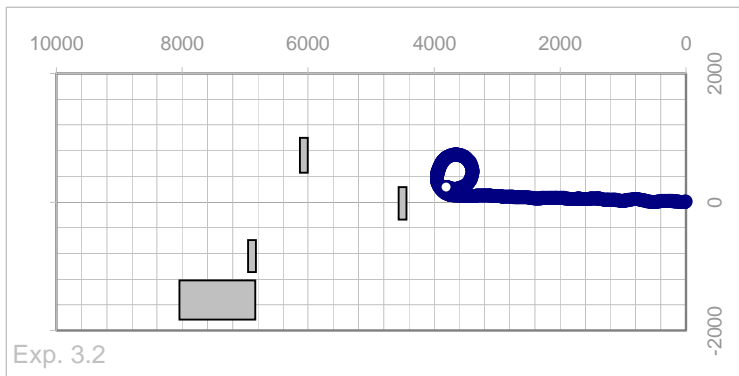
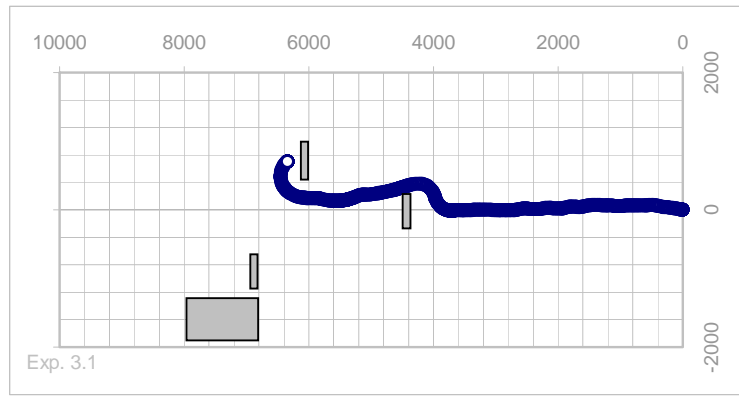
Table 4.6 Weights in experiments.

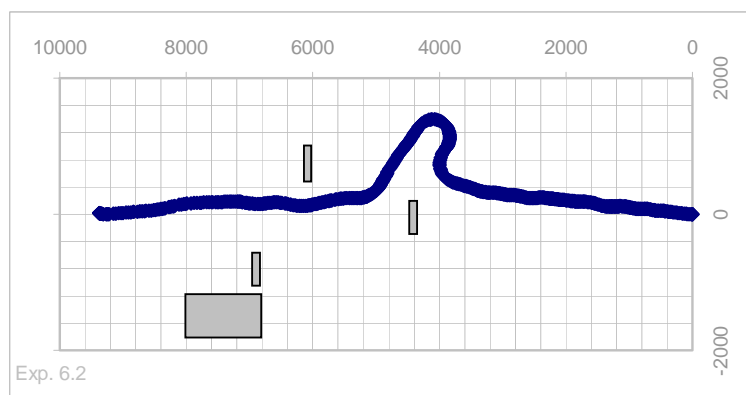
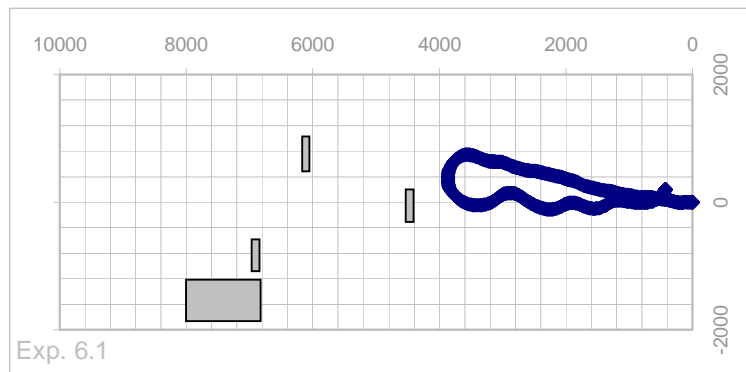
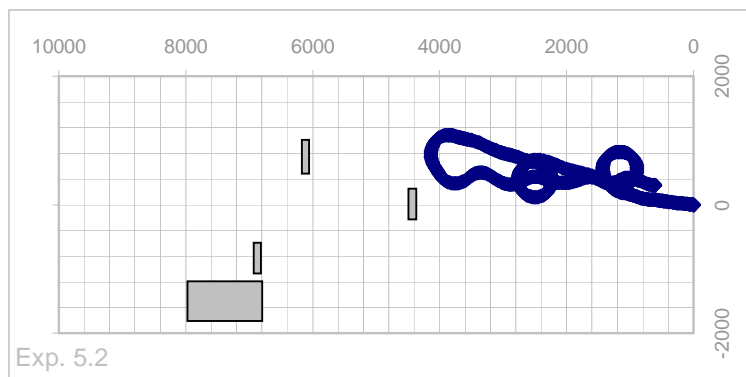
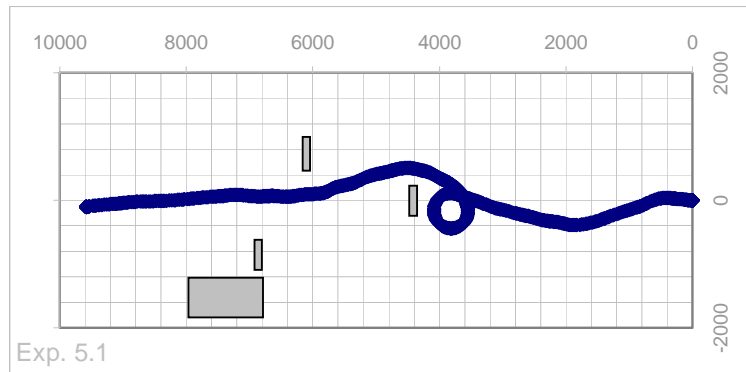
	Exp. 1	Exp. 2	Exp. 3	Exp. 4	Exp. 5	Exp. 6	Exp. 7	Exp. 8
θ	K	K	K	K	K	K	K	K
W11								
W12								
W13								
W14								
W15								
W16								
W21								
W22								
W23								
W24								
W25								
W26								
η	1	1	1	1	1	1	1	1



In table above, darker colors represent stronger weights. Weights are grouped if the data are from left or right side and if the weight is connected to neuron 1 or neuron 2. As an example, w_{11} , w_{12} , w_{13} are colored according to values of each other. Similarly, the following weight values are grouped each having three components.







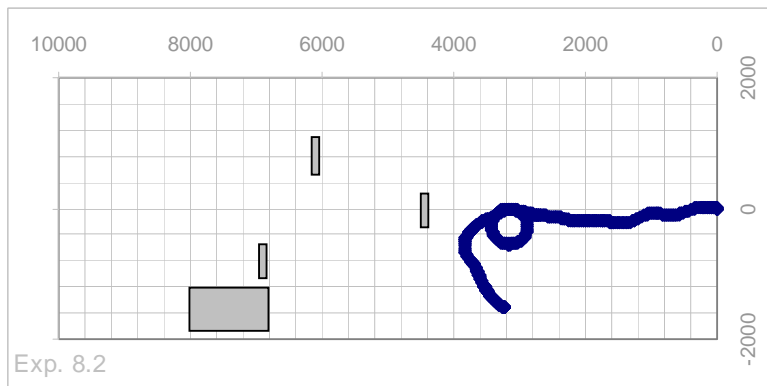
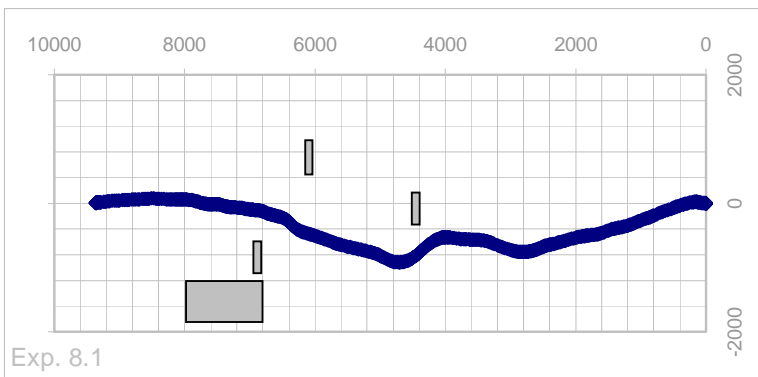
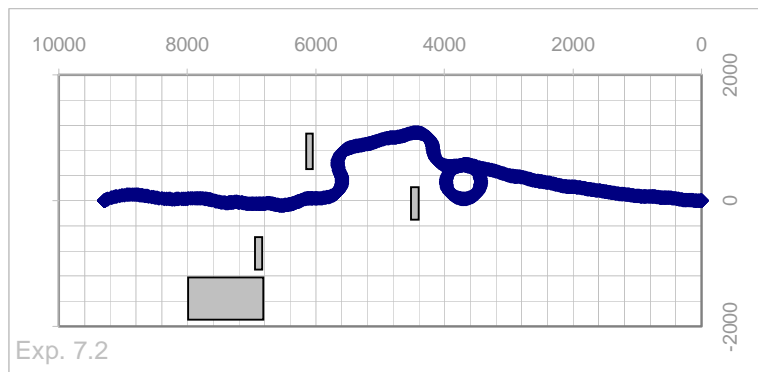
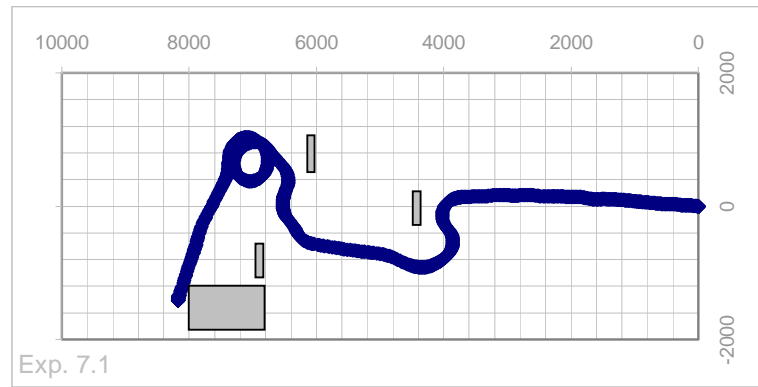


Figure 4.41 Followed paths due to different weights.

For the third group of experiments, the number of obstacles is increased to 8 and the distance to goal increased to 14 meters. In Figure 4.44, Figure 4.45 and Figure 4.46, left and right wheel responses due to the related outputs, six ultrasonic sensor outputs, target x coordinate and followed path during motion of AMR in three ‘4 obstacle and a target scenario’ experiments can be seen. Conclusion will be discussed in following chapter.



Figure 4.42 Third group of experiments with ANN.

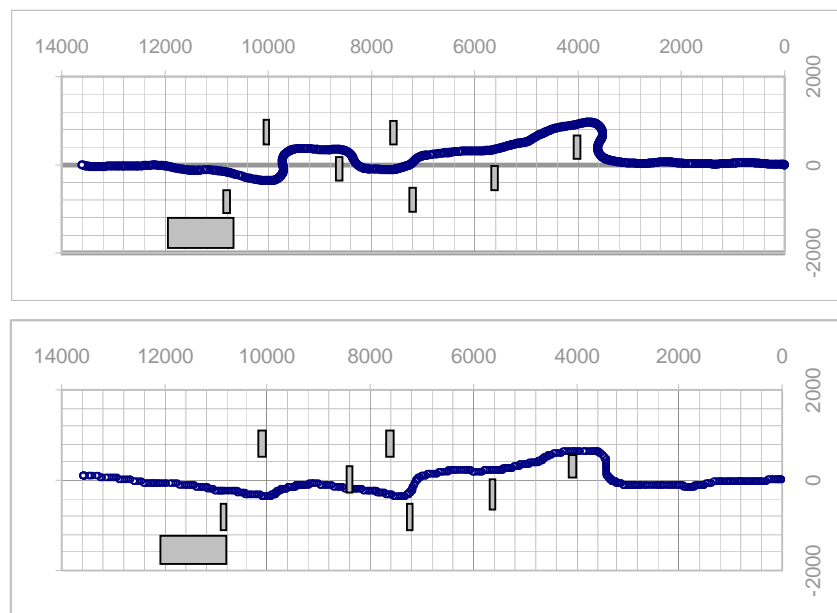


Figure 4.43 Obstacle avoidance and heading to target performance of the AMR.

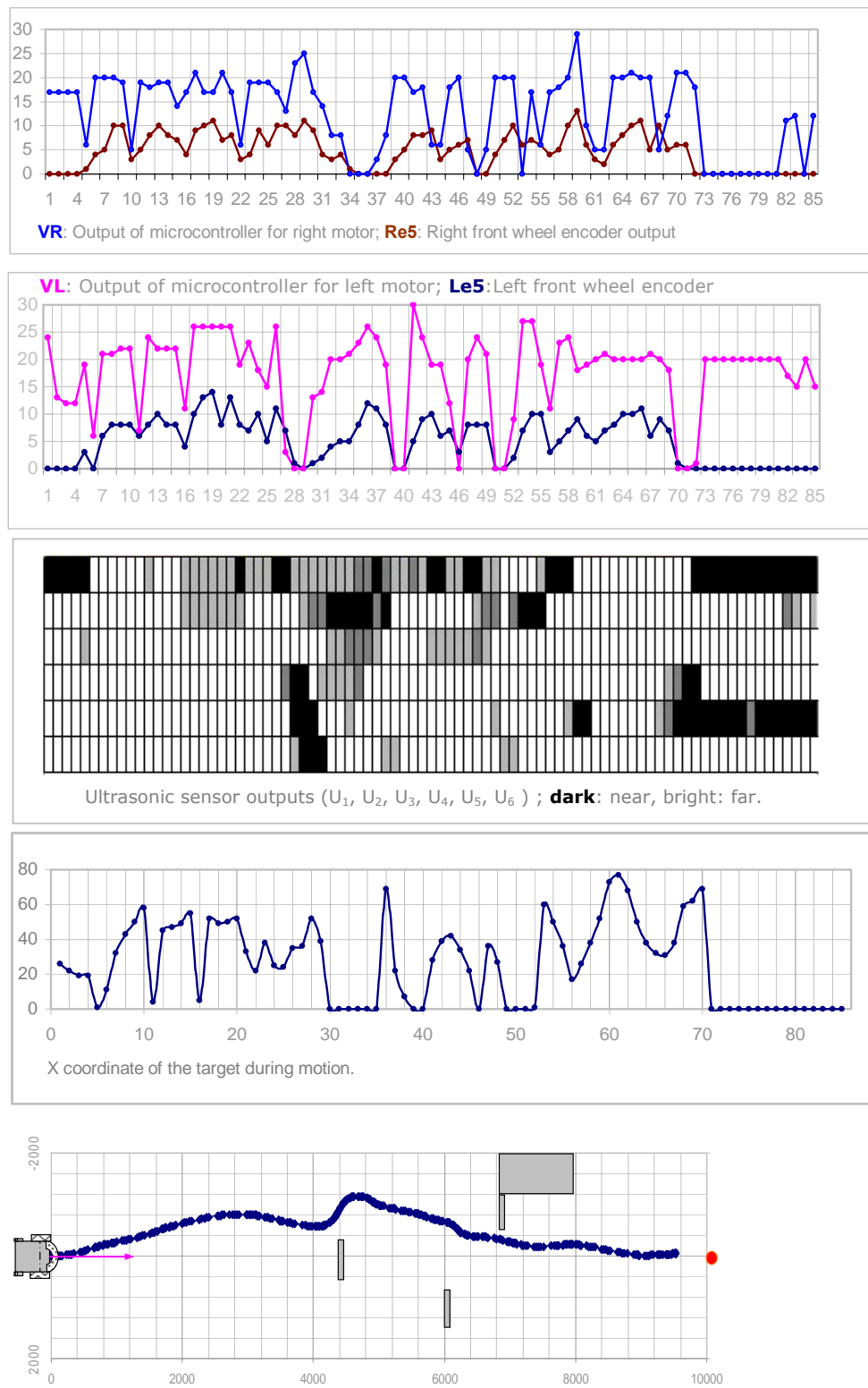


Figure 4.44 Right and left wheel velocities, microcontroller related outputs, sensor outputs during motion.

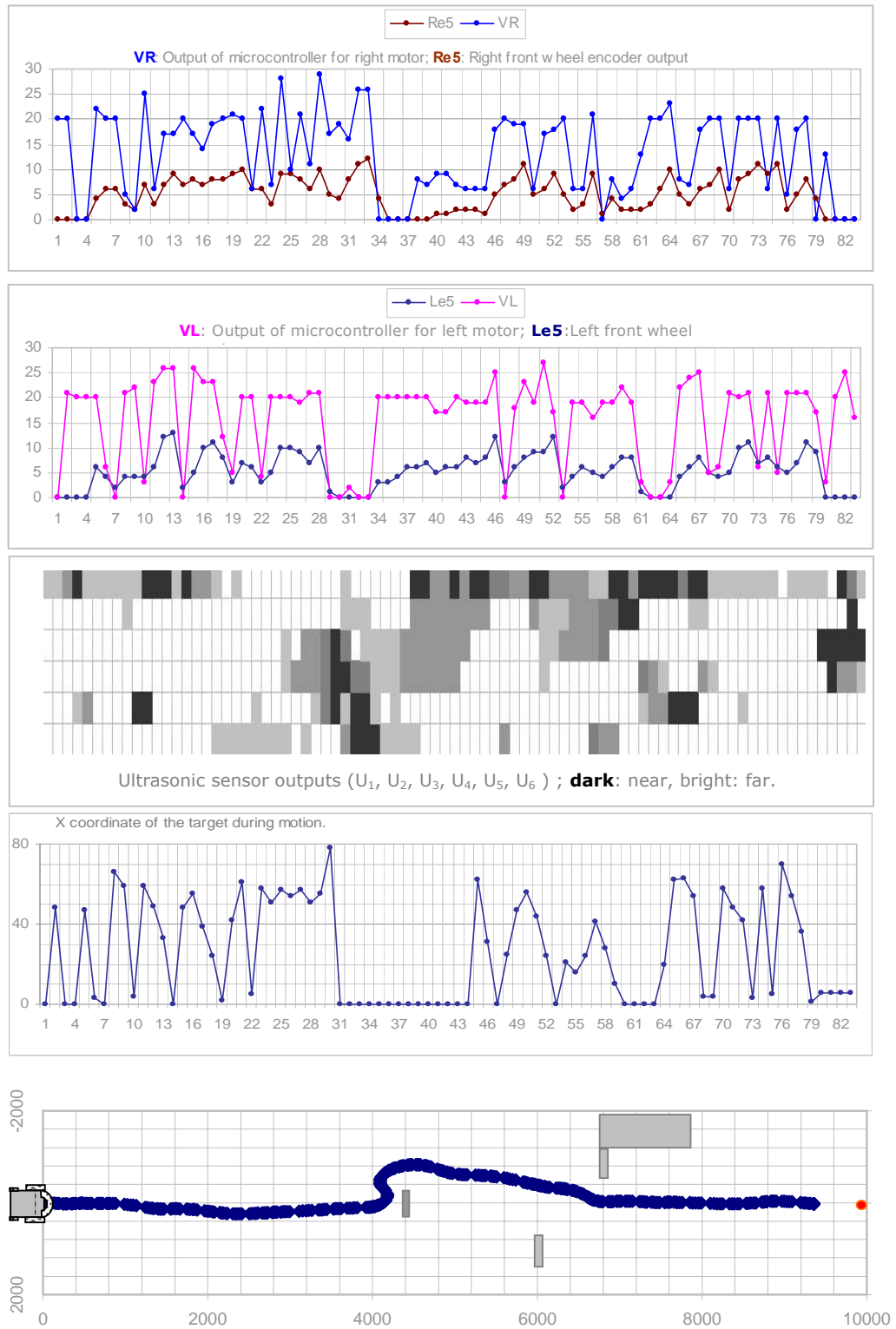


Figure 4.45 Right and left wheel velocities, microcontroller related outputs, sensor outputs during motion.

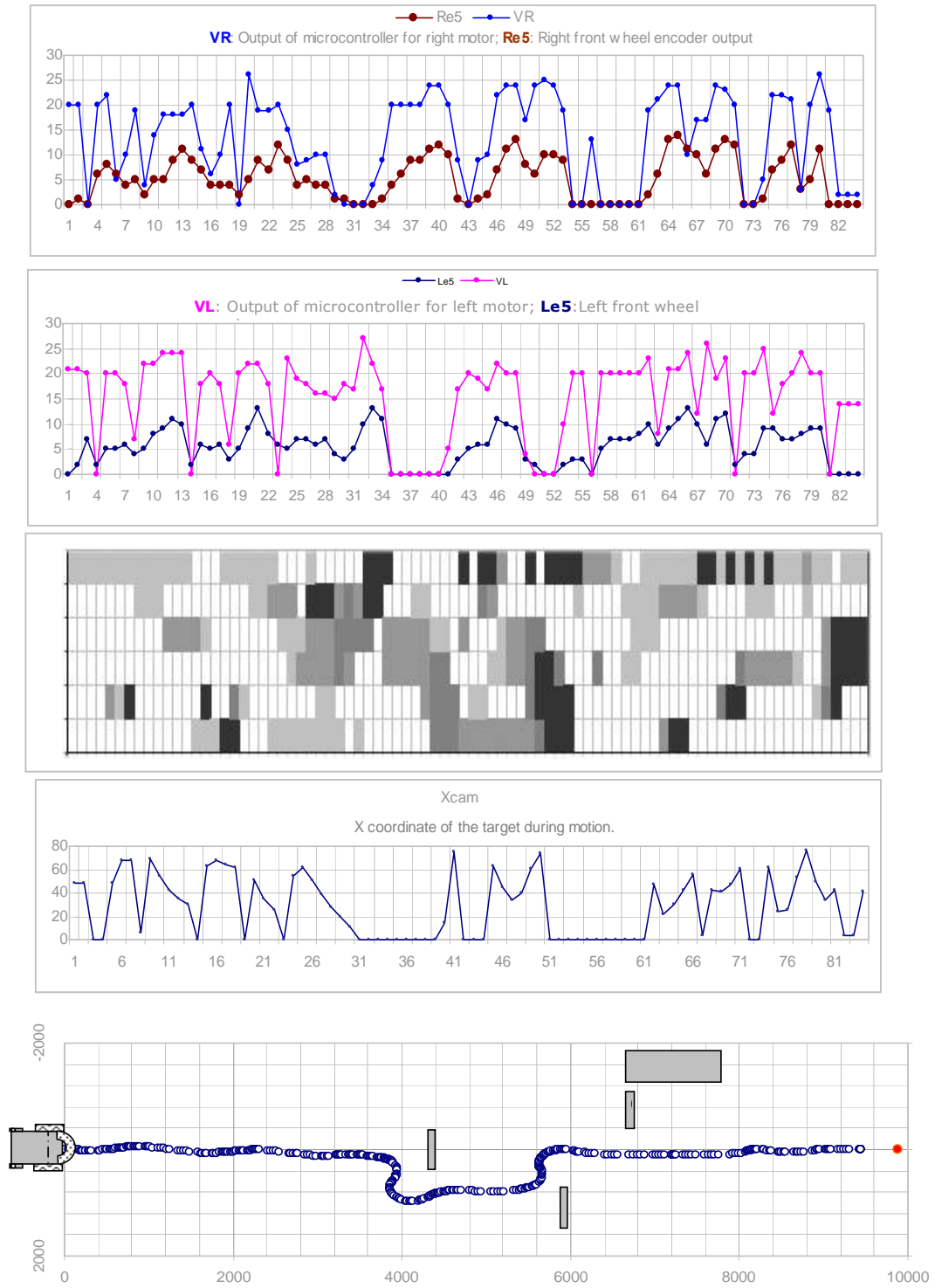


Figure 4.46 Right and left wheel velocities, microcontroller related outputs, sensor outputs during motion.

CHAPTER FIVE

CONCLUSIONS

5.1 Overview

In this research, model of a non-holonomic autonomous wheeled mobile robot is designed and constructed which is equipped with an embedded PC, micro controllers, cameras, encoders and ultrasonic sensors in order to study on some control and sensor fusion algorithms. The WMR model in this research was originally a wheel chair which has four wheels and whose two wheels are actuated with two motors. But, the systems are mounted as its back as its front. The reason was to decrease the disadvantages of its motional constraints. This can be understood easily if the differences between a car motion and a forklift motion are thought.

Nonholonomic conditions are assumed to be expressible as non-integrable differential relations. In analyzing the non holonomic mobile robots, inverse kinematics can be used easier than direct kinematics because of the similarities of the mobile robots with the parallel manipulators. However, forward position kinematics, in other words dead reckoning, is used in many applications or researches.

5.2 Conclusions About Control Techniques and Future Work

A mobile robot has many sensors on it to know environment, to interact with human and to complete its tasks. Uncertainties in ultrasonic sensors caused by the specular reflection from environments make them less reliable. In this thesis, applying Dempster-Shafer evidence theory to data from the sensors, it was aimed to have more reliable sensor data. DS evidence theory was selected because of its advantages over Bayesian theory. DS evidence theory was tested in limitations of quick motion with reliable sensor data. In scale of the processed sensor data values in the experiments, the edges or the important places have greater values after using DS-Theory. So, the sensor fusion part of this research is successful in getting the reliable data from sensors.

Two basic well known problems of the autonomous mobile robots are finding empty space and heading target problems. In means of main control structure, a camera feedback is used on the model for main feedback of target. The first part of fuzzy logic controller experiments is based on finding target and optimal behavior of movement of robot towards the target. Four different groups of experiments were completed in testing the target heading performance and optimizing the fuzzy regions for fast and précised heading. With these experiments, it is obviously seen that if the sensors have reliable and fast data, and motion due to parameters is analyzed well, fuzzy logic is very suitable for heading the mobile robot towards the target. It is not only fast but also robust in different environments. However, in avoiding the obstacles experiments, fuzzy logic controller was not that well working. Rule table, so the code gets very long if all conditions are need to be considered. As a result of long code and conditions, response and so the motion gets slower. If the code does not include all conditions, right decisions can not be made during the motion of avoiding obstacles and heading to target. This situation can be easily seen in Figure 4.34. Another great disadvantage of fuzzy logic controller for a mobile robot motion in object avoidance is that, even the fuzzy logic controller is well designed for a certain environment, if the environment changes, it is luck if the mobile robot makes the right decision.

The perceptron can be seen as the simplest kind of feedforward, supervised neural network. So, starting from idea of using perceptron as a NN controller, MADALINE was decided to be tested as a controller. As predicted, the training phase is the harder part to be applied. In training phase of the experiments, initial weights were predicted. For speeding up the algorithm, supervisor decision was decided to be just for refusing the motion of the AMR. The first groups of experiments with neural networks is using ANN for object avoidance and heading via fuzzy logic. The second groups is to find out the performance when changing weights and the third group of experiments, the number of obstacles is increased to 8 and the distance to goal increased to 14 meters. As for the conclusion of these three groups of experiments, the first result may be said with no hesitation that the performance of NN after

training for object avoidance, even in different conditions, is much more successful than the FLC for the same task. Nevertheless, for heading the robot towards the target, the FLC is fast and very robust. So, combining the system as FLC for heading and NN controller for object avoidance is the optimal solution for this research model. If Figure 4.43, Figure 4.44 and Figure 4.45, which are the graphics obtained during motion, are studied, it can be also seen that the response of the system is fast enough to find the approximate optimal path in real time. Learning the environment is a very great advantage of ANN, but unsupervised learning is more convenient for an autonomous robot. The reason why the supervised learning took part in this research was that the model is designed for a kind of wheel chair and safety motion is more important.

In designing a mobile robot, even the task(s) of the robot is (are) complex, better type is the simpler type of Braitenberg vehicles that completes its task. It is similar for the other criteria of mobile robot design.

The starting point of this thesis was to design an autonomous wheel chair. In completing stage of this research, it can be said the task is completed in means of the first stage of designing and producing a modular autonomous wheel chair which can be very useful for disabled people and old people. For the future work, developing the model for swarm robots of the same model might be very efficient for rest homes and hospitals.

REFERENCES

- Abdul Aziz, S.B. (1996). *Everything you've always wanted to know about designing fuzzy logic machines but were afraid to ask*. Retrieved September 7, 2007, from http://www-dse.doc.ic.ac.uk/~nd/surprise_96/journal/vol2/sbaa/article2.html
- Awad,H.A. & Mohamed A. Al-Zorkany (2004), Mobile robot navigation using local model networks, *International Conference on Computational Intelligence*, Istanbul, TR, 326-331.
- Becerikli, Y. & Çelik, B.K. (2007). Fuzzy control of inverted pendulum and concept of stability using java application. *Mathematical and Computer Modelling*, 46, 24–37.
- Beckerman M. & E. M. Oblow.(1990). Treatment of systematic errors in the processing of wide-angle sonar sensor data for robotic navigation, *IEEE Transactions on Robotics and Automation*, 6(2), 137- 145.
- Braitenberg, V. (1984). *Vehicles: Experiments in Synthetic Psychology*. Massachusetts: The MIT Press.
- Destri, G. (n.d.). *Cellular neural networks*. Retrieved August 18, 2007, from <http://www.ce.unipr.it/pardis/CNN/cnn.html>.
- Dimirovski, G.M., Gacovski, Z. M., Schlacher K. and Kaynak, O. (2000). Mobile non-holonomic robots: On trajectory generation and obstacle avoidance. *Proceedings of the 6th IFAC Symposium Robot Control 2000 - Syroco '00*, 549-554.
- Dohnal, V., Kuca, K. & Jun, D. (2005). What are artificial neural networks and what they can do?, *Biomed Pap Med Fac Univ Palacky Olomouc Czech Repub.*, 149(2), 221–4.

- Gharieb, W. & Nagib, G. (2000). Fuzzy guidance control for a mobile robot. *International MDP Conference*, 67-74.
- Gören, A. (2001). *Computer supported remote controlled manipulator vehicle and data transfer system*. Izmir: Dokuz Eylul University.
- Gören, A., Uyar, E. & Dicle, Z., (2007), Application of Dempster – Shafer evidence theory to sensors of an autonomous mobile robot. *Proceedings of IFAC DECOM-TT'07*, 39-44.
- Hamdi, A. A. & Al-Zorkany, M.A. (2004). Mobile Robot Navigation Using Local Model Networks. *Transactions on Engineering, Computing and Technology*, V1, 326-331.
- Haykin, S. (1994). *Neural Networks*. Macmillan College Publishing Company Inc.
- Hebb, D.O. (1949). *Organization of Behavior*. New York: John Wiley & Sons.
- History of Neural Networks*, (n.d.), Retrieved August 15, 2007, from <http://www.psych.utoronto.ca/~reingold/courses/ai/cache/neural4.html>.
- Iyengar, S.S., & Elfes, A. (1991). *Autonomous Mobile Robots: Control, Planning, and Architecture*. IEEE Computer Society Press.
- Janglova, D. (2004). Neural networks in mobile robot motion. *International Journal of Advanced Robotic Systems*, 1(1), 15-22.
- Kartalopoulos, S.V. (1996), *Understanding neural networks and fuzzy logic : Basic concepts and applications*, NJ: IEEE Press.
- Kopacek, P. (2006). *Autonome, Mobile Roboter*. Lecture notes, Technical University of Vienna.
- Lennox, B., Montague, G. A., Frith, A. M., Gent, C. & Bevan, V., (2001), Industrial application of neural networks – an investigation. *Journal of Process Control*, 11 (2001), 497-507.

- Montaner, M.B. & Ramirez-Serrano, A. (1998). Fuzzy knowledge-based controller design for autonomous robot navigation. *Expert Systems with Applications*, 14(1998), 179-186.
- Mori, H., Tamaru, Y. & Tsuzuki, S. (1992). An artificial neural-net based technique for power system dynamic stability with the Kohonen Model. *Transactions on Power Systems*, 7 (2), 856-864.
- Moser, S. & Pfaffhauser, E. (1998). *Introduction to cellular neural networks*. Retrieved August 20, 2007 from http://www.isi.ee.ethz.ch/~haenggi/CNN_web/introduction.html.
- Muir, P. F. & Neuman, C.P (1986). Kinematic modeling of wheeled mobile robots. *tech. report CMU-RI-TR-86-12*.Carnegie Mellon University, PA, US.
- Murphy, R.R. (2000). *Introduction to AI Robotics*. USA: MIT Press.
- Nissen, L. (2007). Fast Neural Networks Library, 13.06.2007, <http://leenissen.dk>.
- Noguchi, N., Ishii, K. & Tereo, H. (1997). Development of an agriculture mobile robot using a geomagnetic direction sensor and image sensors. *Journal of Agricultural Engineering Research*, 67(1), 1-15.
- PCM-3712, PCM-3718H/HG. (2007). *Advantech PCM-3712PCM-3718H/HG Datasheet*. Advantech Co.,Ltd.
- PCM-3350. (2006). *Advantech PCM3350 Datasheet*. Advantech Co.,Ltd.
- Schneider, E. (2007). Fuzzy control of a gantry crane. *Proceedings of IFAC DECOM-TT'07*, 297-300.
- Simulated Annealing*, (August 27, 2007). Retrieved September 4, 2007, from http://en.wikipedia.org/wiki/Simulated_Annealing.
- Senol, Y. (2002). *Artificial neural networks*. Lecture notes, Dokuz Eylül University.

The Major Structures of the Neuron, (n.d.). Retrieved June 20, 2007 from www.biocrawler.com/encyclopedia/Neurons.

Tsai, L.W. (1999). *Robot Analysis*. Canada: Wiley-Interscience Publication.

Uyar, E., Gören, A. & Zibil, A. (2002). Çift Tekerlekten Ayrik Tahrikli Bir Aracin Bilgisayar Destekli Iz Takip Kontrolu ve Denetimi. *Turkish Conference on Automatic Control* . 487-495.

Van Laerhoven, K. (1999). *On-line adaptive context awareness starting from low level sensors, Thesis*. Brussels: Free University of Brussels.

Yi, Z., Khing, H.Y., Seng, C.C. & Zhou Xiao Wei (2000). Multi-ultrasonic Sensor Fusion for Mobile Robots. *Proceedings of the IEEE Intelligent Vehicles Symposium*. Dearborn (MI), USA.

Zhang, W. (1992). *Two stage inverted pendulum*. Retrieved September 7, 2007, from <http://www.aptronix.com/fuzzynet/applnote/twostage.htm>.

Zu, L., Wang, H.K. and Yue, F. (2004). Artificial neural networks for mobile robot acquiring heading angle. *Proceedings of Third International Conference on Machine Learning and Cybernetics*. 3248-3253.

APPENDICES

APPENDIX 1

NOMENCLATURE

b	bias value []
$bel(A)$	belief function []
c	center []
d_i	desired value []
E	Global energy of the system []
e_i	error of the output []
f_{HL}	Hard limiter []
K	amount of conflict between the two mass sets []
K_R	right motor velocity factor []
K_L	left motor velocity factor []
L	distance between the middles of two front wheels [m]
$m(A)$	mass of set A []
m_i, m_c	codebook vector []
p	probability []
$P(X)$	power set, set of all possible subsets of X []
$pl(A)$	plausibility function []
$q_{3 \times 1}$	location matrix for mobile robot []
R	radius of curvature [m]
r	dynamic radius of wheel [m]
R_{VL}	region value for left (closeness factor) []
R_{VR}	region value for right (closeness factor) []
s_i, s_j	state of firing []
S_{VL}	sensor value for left (3 for S3 and S4, 2 for S2 and S5, 1 for S1 and S6) []
S_{VR}	sensor value for right (3 for S3 and S4, 2 for S2 and S5, 1 for S1 and S6) []
T	similarity ratio []
Te	synthetic temperature of the system []
v	linear velocity of the mobile robot [m/s]

v_{BR}	output value for right motor max velocity (for calibration) [m/s]
v_{BL}	output value for left motor max velocity (for calibration) [m/s]
v_{ij}	feedback weight of i^{th} neuron in layer to j^{th} neuron in previous layer []
$v_L(t)$	linear velocity of the left front wheel [m/s]
V_{max}	maximum linear velocity of the right and left front wheel [m/s]
$v_R(t)$	linear velocity of the right front wheel [m/s]
$w(t)$	angular velocity in z coordinate of the mobile robot [rad/s]
w_n	weights in the artificial neural networks []
X	universal set []
X_1	closeness factor for region 1 (it is close) []
X_2	closeness factor for region 2 (it is far) []
$\{X_h, Y_h\}$	moving coordinate axes []
$\{X_s, Y_s\}$	stationary coordinate axes []
x_i	input(s) of the system []
x_p	target location on x axis [pixel]
y_i	output(s) of the system []
δ_i	local slope []
\emptyset	empty set []
θ	constant in ADALINE model []
Θ	heading angle [rad]
$\kappa(s)$	curvature at point s [1/m]
$\mu_A(x)$	membership degree of x in set \tilde{A}
η	learning rate []
τ_i	threshold value []

APPENDIX 2

ABBREVIATIONS

ADALINE	Adaptive Linear Element, Adaptive Linear Neuron
AI	Artificial Intelligence
AMR	Autonomous Mobile Robot
ANFIS	Adaptive Network Based Fuzzy Inference System
ANNs	Artificial Neural Networks
ART	Adaptive Resonance Theory
BAM	Bidirectional Associative Memory
BP	Backpropagation
CC	Cascade Correlation
CMAC	Cerebellum Model Articulation Controller
CNN	Cellular Neural Networks
dLVQ	Dynamic Learning Vector Quantization
DOF	Degrees of Freedom
FAM	Fuzzy Associative Memory
FCM	Fuzzy Cognitive Map
FL	Fuzzy Logic
FLC	Fuzzy Logic Control
FS	Fuzzy Sets
GUI	Graphical User Interface
HAM	Hamming Net
HOP	Hopfield Network
LMS	Least Mean Square
LVQ	Learning Vector Quantization (or linear vector quantization)
MADALINE	Multiple ADALINE
M1	MADALINE I
M2	MADALINE II
M3	MADALINE III
MFT	Mean Field Theory
MR	Mobile Robot
OLVQ	Optimized LVQ
OLAM	Optimal Linear Associative Memory
PCA	Principal component analysis

PIC	Peripheral Interface Controller
PNN	Probabilistic Neural Network
QP	Quick Propagation
RBF	Radial Basis Function
RBP	Recurrent Backpropagation
RCE	Restricted Coulomb Energy
RF	Radio Frequency
RfComm	Radio Frequency Communication
SBC	Single Board Computer
SDP	Session Description Protocol
SOM	Self-Organizing Map
TAM	Trilateral Associative Memory
TDNN	Time-Delay Neural Net
USB	Universal Serial Bus
WMR	Wheeled Mobile Robot
WTA	Winner Takes All

APPENDIX 3

DEFINITIONS IN COMPUTER NETWORKS

Bandwidth : Commonly, it is the size of the channel used by the radio (the amount of frequency available to the system). By extension, it can also sometimes refer to the speed of the system (the bit rate).

Bit-rate : Speed at which bits are transmitted over the physical layer, also called signaling rate. Quite different from throughput.

Carrier: The base frequency used by the system. The modulation process will generate a signal centered on the carrier, of width equal to the bandwidth.

Carrier Sense: Checking the transmission medium to get permission if it is free or if there is a transmission going on. Usually measure of the received power. See CSMA.

CDMA (Code Division Multiple Access) : Technique used to share the same bandwidth between different channels using codes. The code is a signature multiplexed with the signal and used to recover it.

CSMA (Carrier Sense Multiple Access) : Using carrier sense to access the medium. One of the main MAC methods.

Cell : Radio neighborhood, area where all nodes can communicate with each other. As the range over radio is limited, the network is split into independent cells and a cell to cell communication is provided (via access point or internal routing).

Channel : On the radio, this is usually synonym of a specific frequency, and by extension the communication medium. It can also mean a stream of data between two nodes (a point to point link in connection oriented systems).

dB (decibel) : Logarithmic way to express a value. Usually the signal strength (transmitted and received power) is expressed in dBm (the reference is 1 mW - 0 dBm). A difference between two values in dBm is without unit, in dB (in fact, this is a factor between the two values).

Ethernet : Standard wired LAN protocol. Includes physical and link layers.

Fading : Variation in channel performance due to the dynamicity of the environment, make the receive signal strength change.

FEC (Forward Error Correction) : Technique used to overcome some type of errors created by transmission on noisy channels, by adding redundancy bits to the main data transmission.

Frequency band : Portion of the radio spectrum delimited for a particular use. For example, most wireless LANs use the 2.4 to 2.48 GHz band. A frequency band is usually divided in channels.

FTP: File Transfer Protocol

Header : Information added by the protocol in front of the payload in the packet for its own use (addresses, packet type, sequence number, CRC...). Each protocol adds a different header, so in a typical TCP/IP packet as transmitted, we have a MAC header, an IP header and a TCP header, followed by the payload.

IP : see TCP/IP.

IPX : Network protocol used in Netware, usually with SPX.

LAN (Local Area Network) : Network on a short distance, as opposed to WAN (typically inside a building).

Latency: Measure of the performance of a network for short requests and multimedia traffic. There is no real standard measurement, it might be the time to send and transmit a packet, or the time spent in the transmit queue, or the time for an answer to come back, or a number of requests per second...

Layer: This terminology comes from the OSI specification. It divides any communicating system into 7 layers, each having a different functionality. Layer 1 is the physical layer, and layer 2 is the link layer. IP could be assimilated as layer 3 (network layer), and TCP as layer 4 (transport layer).

Link layer: This is the part of the protocol managing the direct delivery between two devices on a specific physical layer (coaxial bus, point to point link, radio...). This includes packaging and addressing. Most of this is implemented in the MAC.

MAC (Medium Access Control) : This is the part of the radio device managing the protocol and the usage of the link. The MAC decides when to transmit and when to receive, creates the packets headers and filters the received packets.

Medium: Name to describe the mean used to transfer information. This could be a wire (twisted pairs, coax...), an optic fiber, the radio waves (the air), infrared light...

Mbps: Megabits per second

Modem (modulator/demodulator) : In a radio device, this is the part converting the bits to transmit into a modulation of the radio waves and the reverse at the reception. It does the analog to digital conversion, the generation of the frequency, the modulation and the amplification.

Modulation: Specific way of coding information on a radio frequency. Basically, there is amplitude modulation (AM - change waveform strength) and frequency / phase modulation (FM - change waveform timing), but there exist many variations and combinations each designated by a specific acronym.

NetBeui : Network protocol used in Lan Manager.

Node: A device part of the network, source or destination of the data. For us, a computer with a radio card in it.

Noise: Any unwanted signal. Background noise, interferences, and transmissions from nodes; not belonging to the network.

Packet: Unit of transmission over the network. The data to be transmitted is split into packets, which are sent individually over the network.

Protocol: Specification of the interactions between systems and the data manipulated. This describes what to do and when (the rules), and the format of the data exchanged on the lower communication layer.

Physical layer: This is the part of the device interacting with the medium. For a radio LAN, the physical layer is also called modem.

Roaming: Ability to move between cells of the same network.

SNR (Signal to Noise Ratio): Difference in strength between the signal we want to receive and the background noise (or any unwanted signal).

TCP/IP: Network protocol used by firstly UNIX and Internet. Better in some respects than NetBeui and IPX (allows routing, for example).

TDMA (Time Division Multiple Access): Technique used to share the same bandwidth between different channels using periodic time slots.

Throughput: Measure of the performance of a network for large data transfer (such as FTP, NFS, HTTP 1.1). This speed is expressed in bits per seconds or a multiple.

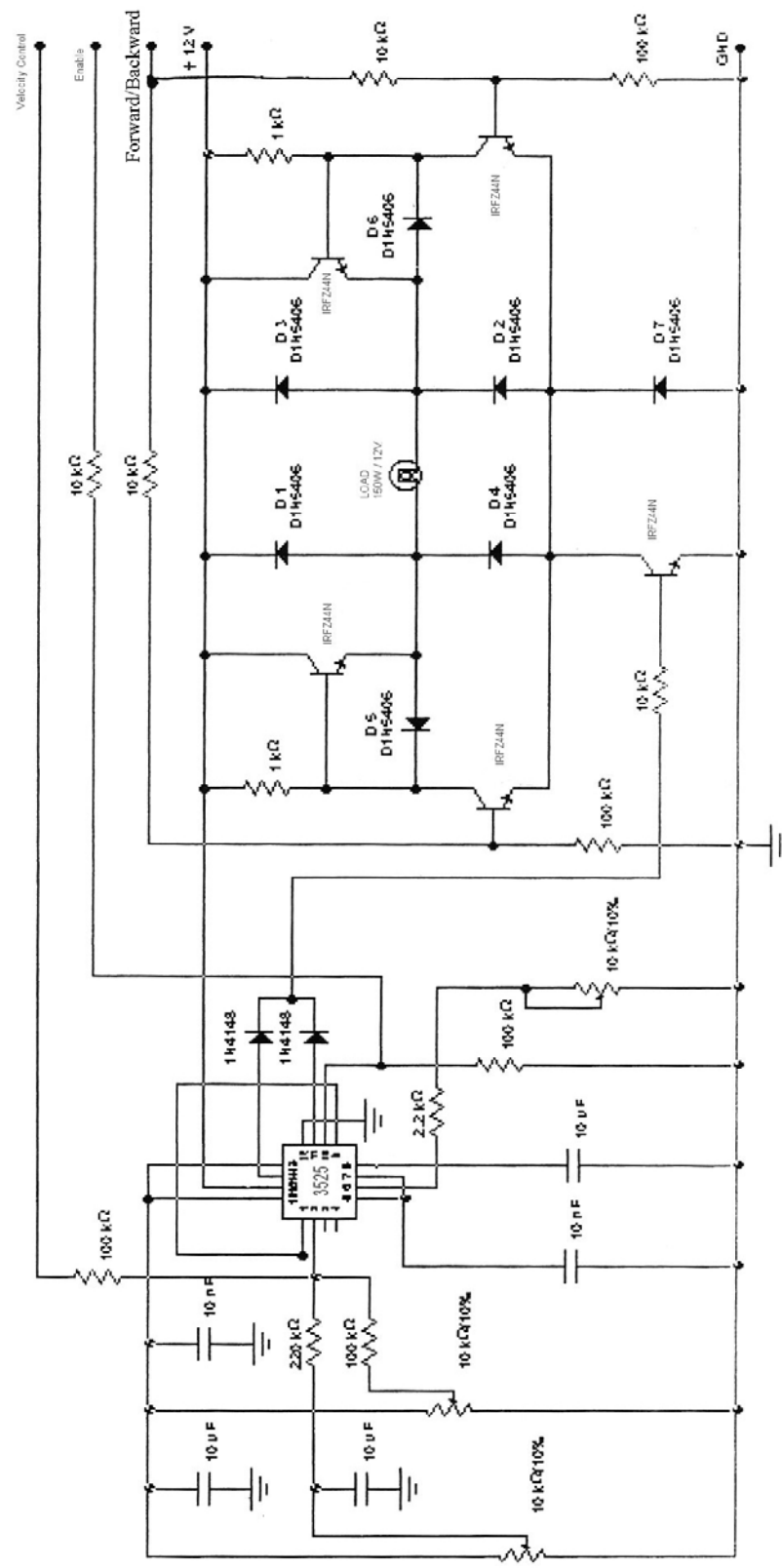
VNC stands for Virtual Network Computing. It is remote control software which allows you to view and interact with one computer (the "server") using a simple program (the "viewer") on another computer anywhere on the Internet. The two computers don't even have to be the same type, so for example you can use VNC to view an office Linux machine on your Windows PC at home.

WAN (Wide Area Network): Network on a large scale: a town, a country or the world.

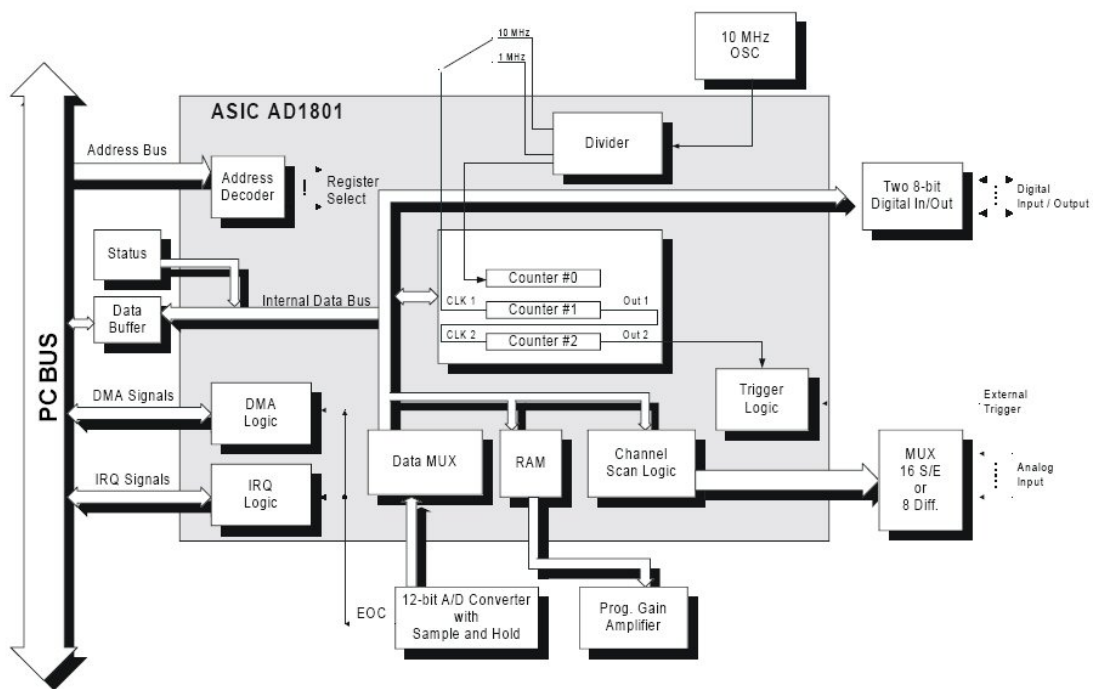
Wired: Using a wire.

Wireless: Not using a wire. For networks, it might be radio or infrared.

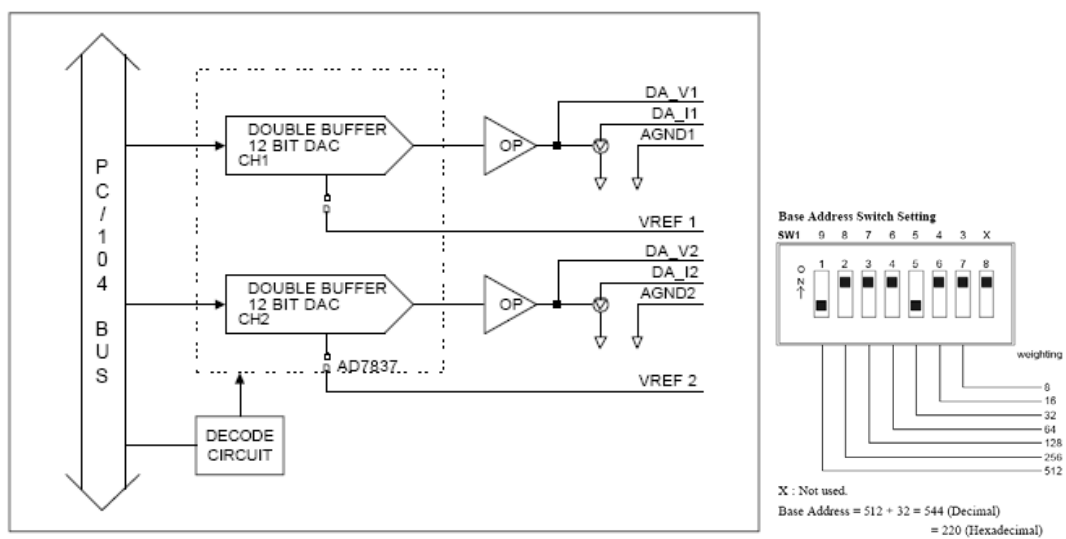
WLANS: wireless local area networks



Schematic 2 Experimental motor driver circuit

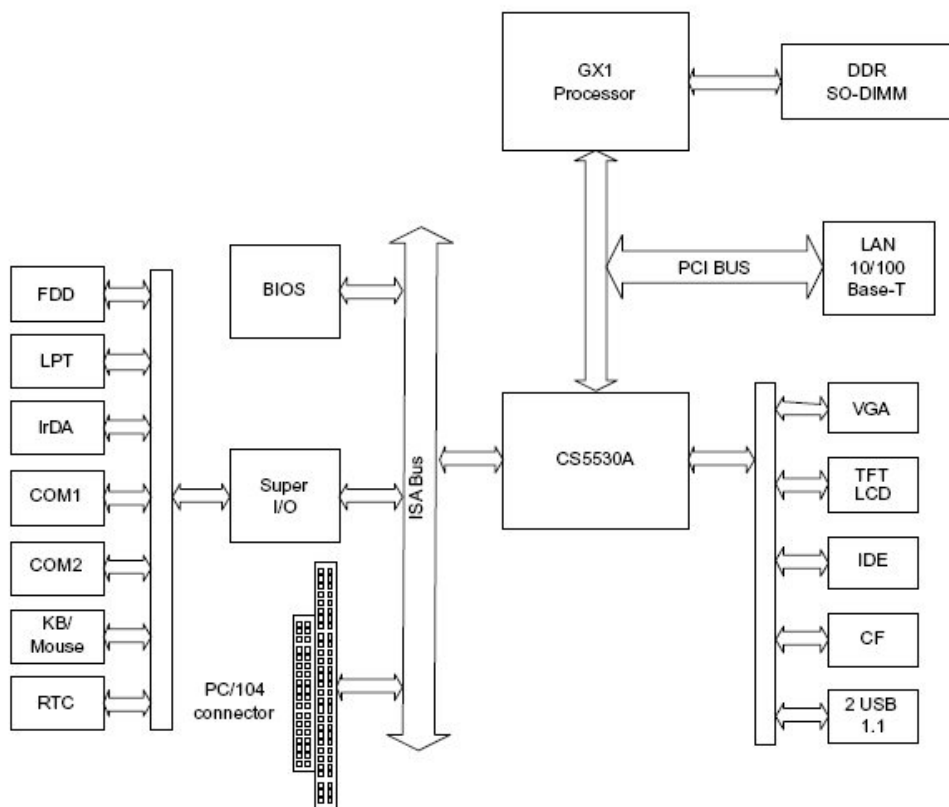


Schematic 3. PCM 3718 block diagram

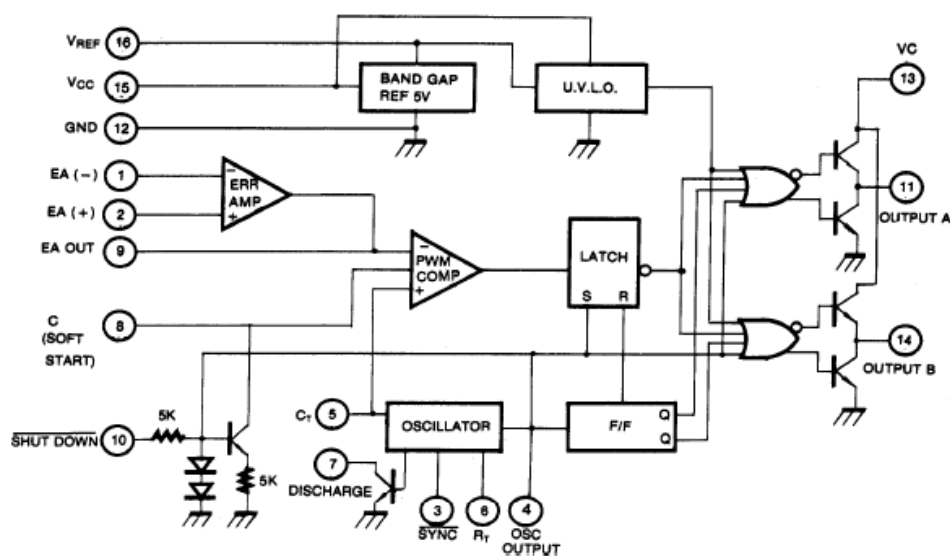


PCM 3712 base address selecting.

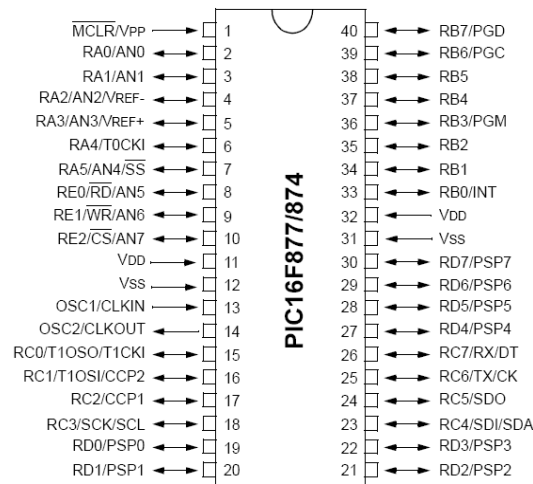
Schematic 4. PCM 3712 block diagram and base address selecting.



Schematic 5. Block diagram of PC104 main board (PCM 3350).



Schematic 6. Internal block diagram of UC3525A.



Schematic 7. 16F877 28/40-pin 8-bit CMOS FLASH microcontroller pin diagram

Pin Name	DIP Pin#	PLCC Pin#	QFP Pin#	I/O/P Type	Buffer Type	Description
OSC1/CLKIN	13	14	30	I	ST/CMOS ⁽⁴⁾	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	14	15	31	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate.
MCLR/VPP	1	2	18	I/P	ST	Master Clear (Reset) input or programming voltage input. This pin is an active low RESET to the device.
RA0/AN0	2	3	19	I/O	TTL	PORTA is a bi-directional I/O port. RA0 can also be analog input0. RA1 can also be analog input1. RA2 can also be analog input2 or negative analog reference voltage. RA3 can also be analog input3 or positive analog reference voltage. RA4 can also be the clock input to the Timer0 timer/counter. Output is open drain type. RA5 can also be analog input4 or the slave select for the synchronous serial port.
RA1/AN1	3	4	20	I/O	TTL	
RA2/AN2/VREF-	4	5	21	I/O	TTL	
RA3/AN3/VREF+	5	6	22	I/O	TTL	
RA4/T0CKI	6	7	23	I/O	ST	
RA5/SS/AN4	7	8	24	I/O	TTL	
RB0/INT	33	36	8	I/O	TTL/ST ⁽¹⁾	PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs. RB0 can also be the external interrupt pin. RB3 can also be the low voltage programming input. Interrupt-on-change pin. Interrupt-on-change pin. Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming clock. Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming data.
RB1	34	37	9	I/O	TTL	
RB2	35	38	10	I/O	TTL	
RB3/PGM	36	39	11	I/O	TTL	
RB4	37	41	14	I/O	TTL	
RB5	38	42	15	I/O	TTL	
RB6/PGC	39	43	16	I/O	TTL/ST ⁽²⁾	
RB7/PGD	40	44	17	I/O	TTL/ST ⁽²⁾	

Legend: I = input O = output I/O = input/output P = power
 — = Not used TTL = TTL input ST = Schmitt Trigger input

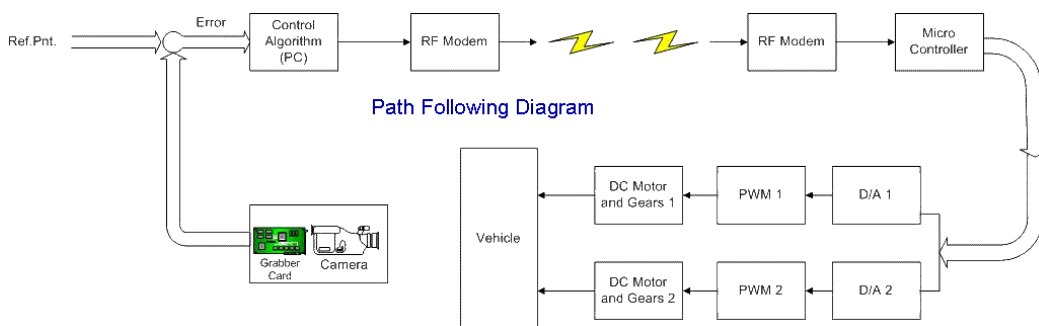
- Note 1:** This buffer is a Schmitt Trigger input when configured as an external interrupt.
Note 2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
Note 3: This buffer is a Schmitt Trigger input when configured as general purpose I/O and a TTL input when used in the Parallel Slave Port mode (for interfacing to a microprocessor bus).
Note 4: This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

Pin Name	DIP Pin#	PLCC Pin#	QFP Pin#	I/O/P Type	Buffer Type	Description
RC0/T1OSO/T1CKI	15	16	32	I/O	ST	PORTC is a bi-directional I/O port. RC0 can also be the Timer1 oscillator output or a Timer1 clock input.
RC1/T1OSI/CCP2	16	18	35	I/O	ST	RC1 can also be the Timer1 oscillator input or Capture2 input/Compare2 output/PWM2 output.
RC2/CCP1	17	19	36	I/O	ST	RC2 can also be the Capture1 input/Compare1 output/PWM1 output.
RC3/SCK/SCL	18	20	37	I/O	ST	RC3 can also be the synchronous serial clock input/output for both SPI and I ² C modes.
RC4/SDI/SDA	23	25	42	I/O	ST	RC4 can also be the SPI Data In (SPI mode) or data I/O (I ² C mode).
RC5/SDO	24	26	43	I/O	ST	RC5 can also be the SPI Data Out (SPI mode).
RC6/TX/CK	25	27	44	I/O	ST	RC6 can also be the USART Asynchronous Transmit or Synchronous Clock.
RC7/RX/DT	26	29	1	I/O	ST	RC7 can also be the USART Asynchronous Receive or Synchronous Data.
RD0/PSP0	19	21	38	I/O	ST/TTL ⁽³⁾	PORTD is a bi-directional I/O port or parallel slave port when interfacing to a microprocessor bus.
RD1/PSP1	20	22	39	I/O	ST/TTL ⁽³⁾	
RD2/PSP2	21	23	40	I/O	ST/TTL ⁽³⁾	
RD3/PSP3	22	24	41	I/O	ST/TTL ⁽³⁾	
RD4/PSP4	27	30	2	I/O	ST/TTL ⁽³⁾	
RD5/PSP5	28	31	3	I/O	ST/TTL ⁽³⁾	
RD6/PSP6	29	32	4	I/O	ST/TTL ⁽³⁾	
RD7/PSP7	30	33	5	I/O	ST/TTL ⁽³⁾	
RE0/ \overline{RD} /AN5	8	9	25	I/O	ST/TTL ⁽³⁾	PORTE is a bi-directional I/O port. RE0 can also be read control for the parallel slave port, or analog input5.
RE1/ \overline{WR} /AN6	9	10	26	I/O	ST/TTL ⁽³⁾	RE1 can also be write control for the parallel slave port, or analog input6.
RE2/ \overline{CS} /AN7	10	11	27	I/O	ST/TTL ⁽³⁾	RE2 can also be select control for the parallel slave port, or analog input7.
V _{ss}	12,31	13,34	6,29	P	—	Ground reference for logic and I/O pins.
V _{DD}	11,32	12,35	7,28	P	—	Positive supply for logic and I/O pins.
NC	—	1,17,28,40	12,13,33,34		—	These pins are not internally connected. These pins should be left unconnected.

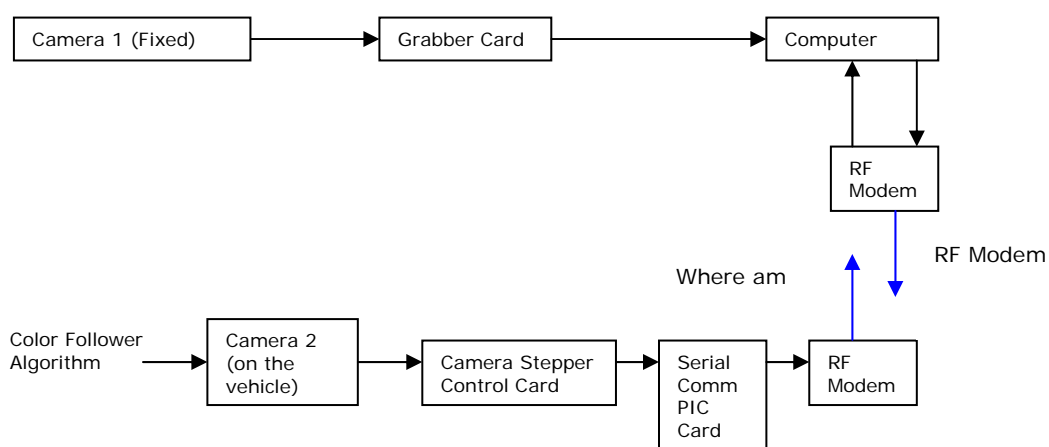
Legend: I = input O = output I/O = input/output P = power
 — = Not used TTL = TTL input ST = Schmitt Trigger input

- Note 1:** This buffer is a Schmitt Trigger input when configured as an external interrupt.
Note 2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
Note 3: This buffer is a Schmitt Trigger input when configured as general purpose I/O and a TTL input when used in the Parallel Slave Port mode (for interfacing to a microprocessor bus).
Note 4: This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

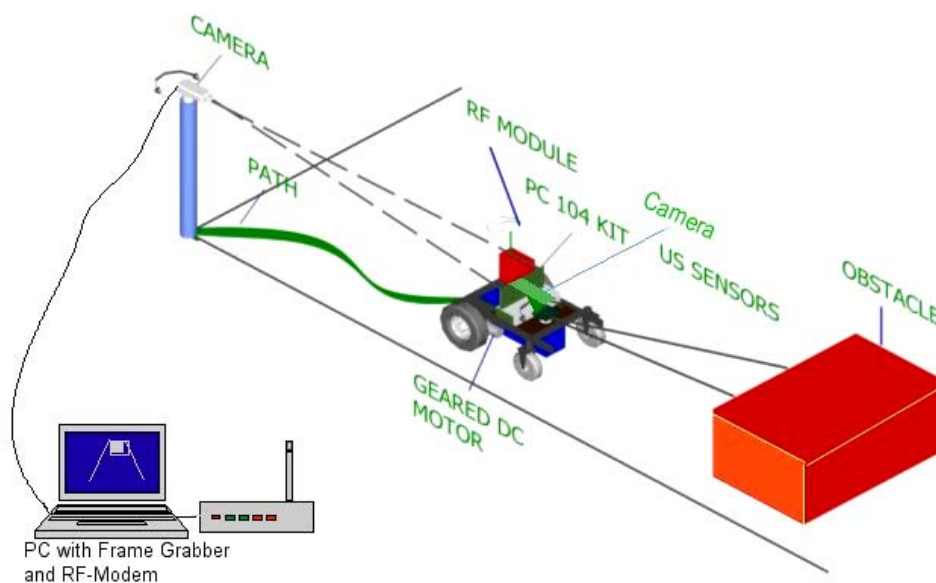
Schematic 8. 16F877 pinout description



(First Model: Path Following Diagram of the Vehicle)

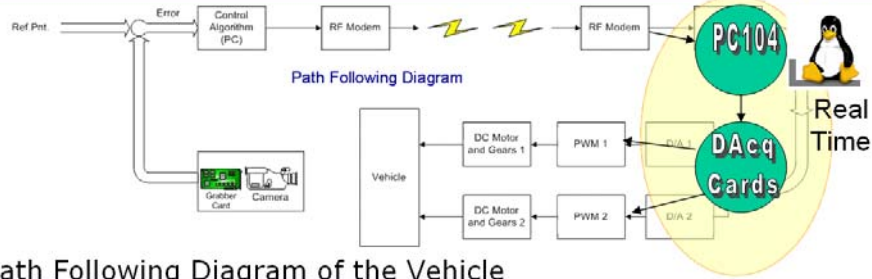


(First Model: Navigation Diagram of the Vehicle)

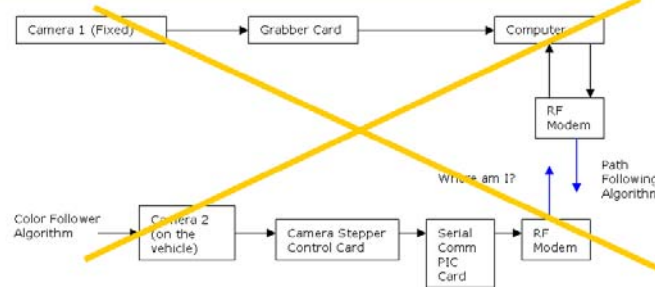


(First model: General view.)

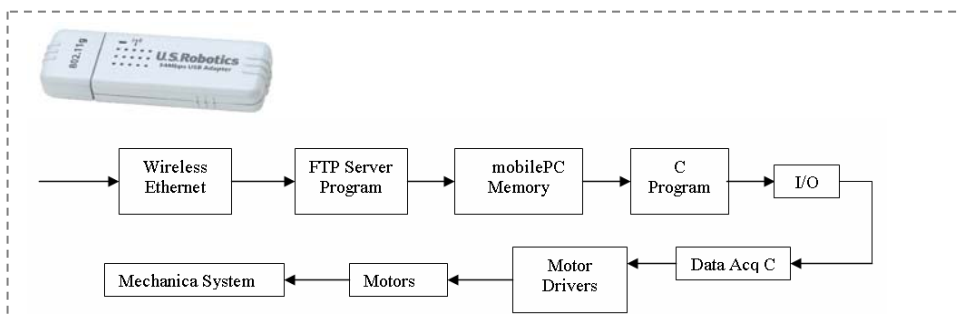
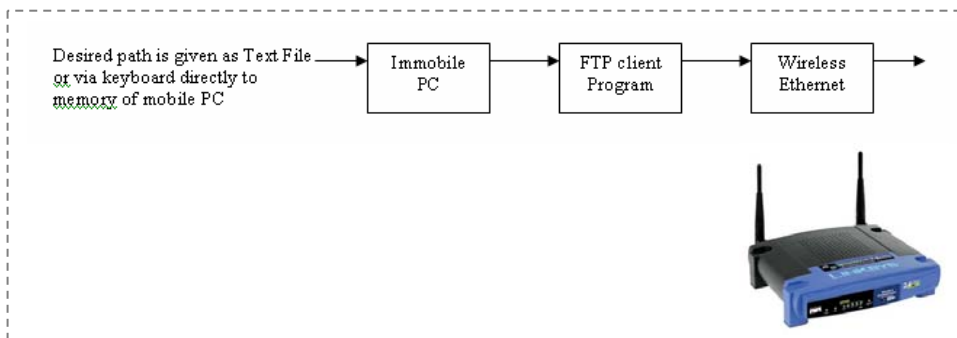
System



Path Following Diagram of the Vehicle

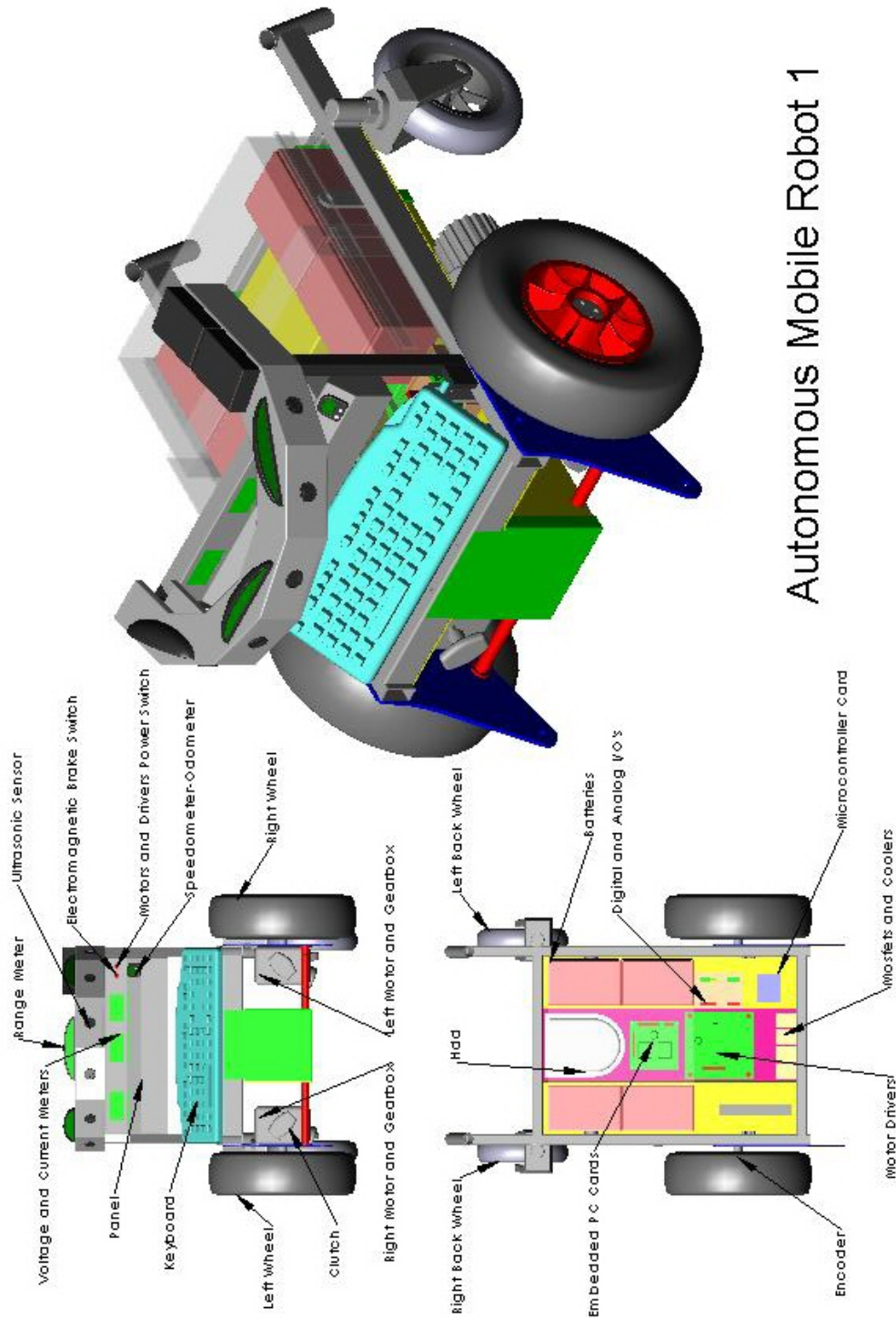


(Second model of the AMR)



(Third model of the AMR)

Schematic 10. Model developing stages.



Autonomous Mobile Robot 1

Schematic 11. 3D Model of the AMR:

APPENDIX 5

PROGRAMS

1. Ultrasonic Sensor Data Sending Pic Basic Pro Program

```

*****
'* Name      : SENSOR.BAS                               *
'* Author    : OK & AG                                   *
'* Notice    : Copyright (c) 2006                       *
'*           : All Rights Reserved                       *
'* Date      : 08.06.2006                               *
'* Version   : 1.0                                       *
*****
c1 var byte
c2 var byte
c3 var byte
c4 var byte
c5 var byte
c6 var byte
c7 var byte
*****
d0 var byte
d1 var byte
d2 var byte
d3 var byte
d4 var byte
d5 var byte
d6 var byte
d7 var byte
*****
sonuc1 var byte
sonuc2 var byte
sonuc3 var byte
sonuc4 var byte
sonuc5 var byte
sonuc6 var byte

lref var byte
rref var byte

TRISA=%11111111
TRISB=%10111111
TRISC=%11111111
TRISD=%11111111
TRISE=%00000000

PORTA = 0
PORTB = 0
portc = 0
portd = 0
PORTE = 0

pause 500

include "modedefs.bas"
serout PORTB.6,N9600,["Başlıyor..",13,10]

c1 = 0
c2 = 0
c3 = 0
c4 = 0
c5 = 0
c6 = 0
c7 = 0

lref = 0
rref = 0

start:

```

```

startc:
while portc.0=1
Wend
while portc.0=0
Wend
pause 2
c1.3=portc.1
c2.3=portc.2
c3.3=portc.3
c4.3=portc.4

pause 4
c1.2=portc.1
c2.2=portc.2
c3.2=portc.3
c4.2=portc.4
pause 4
c1.1=portc.1
c2.1=portc.2
c3.1=portc.3
c4.1=portc.4
pause 4
c1.0=portc.1
c2.0=portc.2
c3.0=portc.3
c4.0=portc.4
'sol algıla

sonuc5 = 0
if c4.3=0 then sonuc5=2
if c3.3=0 then sonuc5=4
if c2.3=0 then sonuc5=6
if c1.3=0 then sonuc5=7
sonuc6 = 0
if c4.2=0 then sonuc6=2
if c3.2=0 then sonuc6=4
if c2.2=0 then sonuc6=6
if c1.2=0 then sonuc6=7

startd:

while portd.3=1
Wend
while portd.3=0
Wend
pause 2
d4.3=portd.4
d5.3=portd.5
d6.3=portd.6
d7.3=portd.7
pause 4
d4.2=portd.4
d5.2=portd.5
d6.2=portd.6
d7.2=portd.7
pause 4
d4.1=portd.4
d5.1=portd.5
d6.1=portd.6
d7.1=portd.7
pause 4
d4.0=portd.4
d5.0=portd.5
d6.0=portd.6
d7.0=portd.7

'sol algıla
sonuc3 = 0
if d7.3=0 then sonuc3=2
if d6.3=0 then sonuc3=4
if d5.3=0 then sonuc3=6
if d4.3=0 then sonuc3=7
sonuc4 = 0
if d7.2=0 then sonuc4=2

```

```

if d6.2=0 then sonuc4=4
if d5.2=0 then sonuc4=6
if d4.2=0 then sonuc4=7

startb:

while portc.5=1
Wend
while portc.5=0
Wend
pause 2
c6.3=portc.6
c7.3=portc.7
d0.3=portd.0
d1.3=portd.1

pause 4
c6.2=portc.6
c7.2=portc.7
d0.2=portd.0
d1.2=portd.1
pause 4
c6.1=portc.6
c7.1=portc.7
d0.1=portd.0
d1.1=portd.1
pause 4
c6.0=portc.6
c7.0=portc.7
d0.0=portd.0
d1.0=portd.1
'sol algıla

sonuc1 = 0
if d1.3=0 then sonuc1=2
if d0.3=0 then sonuc1=4
if c7.3=0 then sonuc1=6
if c6.3=0 then sonuc1=7
sonuc2 = 0
if d1.2=0 then sonuc2=2
if d0.2=0 then sonuc2=4
if c7.2=0 then sonuc2=6
if c6.2=0 then sonuc2=7

'serout
PORTB.6,N9600, [#sonuc1,"",#sonuc2,"",#sonuc3,"",#sonuc4,"",#sonuc5,"",#sonuc6,13
,10]

rref = 30
lref = 30
If sonuc2 >= 4 Then rref = 30: lref = 10
If sonuc5 >= 4 Then rref = 10: lref = 30

serout PORTB.6,N9600,["L",lref,"R",rref]

GoTo start

End

```

2. Encoder Data Sending Pic Basic Pro Program

```

*****
'* Name      : ENCODER.BAS          *
'* Author    : OK & AG              *
'* Notice    : Copyright (c) 2006   *
'*           : All Rights Reserved   *
'* Date      : 08.06.2006           *
'* Version   : 1.0                  *
*****

TRISA=%11111100
TRISB=%11111111

PORTa = 0
portb = 0
left var word
left_durum var byte
right var word
right_durum var byte

i var word
include "modedefs.bas"
pause 500
Left = 0
left_durum = 0
Right = 0
right_durum = 0

mainloop:

For i = 0 To 10000
if portb.1=1 then
  If left_durum = 0 Then
    Left = Left + 1
    left_durum = 1
  End If
Else
  left_durum = 0
End If
if portb.0=1 then
  If right_durum = 0 Then
    Right = Right + 1
    right_durum = 1
  End If
Else
  right_durum = 0
End If
Next
cikti:
serout PORTa.1,N9600,["1",left.BYTE0,"2",left.BYTE1,"3",right.BYTE0,"4",right.BYTE1]
Left = 0
Right = 0
GoTo mainloop

```

3. Motor Control due to the Encoder Feedback Pic Basic Pro Program

```

*****
'* Name      : MAIN.BAS                               *
'* Author    : OK & AG                               *
'* Notice    : Copyright (c) 2006                    *
'*           : All Rights Reserved                   *
'* Date      : 08.06.2006                             *
'* Version   : 1.0                                    *
*****

TRISA=%11111111
TRISB=%10111111
TRISC=%11111111
TRISD=%11111111
TRISE=%00000000

PORTA = 0
PORTB = 0
PORTC = 0
PORTD = 0
PORTE = 0

lhiz var word
rhiz var word
lmax var word
rmax var word
lref var byte
rref var byte
lpwm var word
rpwm var word

i var word

lfeed_back var word
rfeed_back var word

pause 500
lref = 0
rref = 0
lmax = 70
rmax = 70
lfeed_back = 0
rfeed_back = 0
lpwm = 0
rpwm = 0

include "modedefs.bas"
serout PORTB.6,N9600,["Başlıyor..",13,10]
HPWM 1, lpwm, 20000
HPWM 2, rpwm, 20000

start:
'encoder verisi
SERIN portb.5,N9600,["1"],lhiz.BYTE0'motor verilerini alıyoruz.
SERIN portb.5,N9600,["2"],lhiz.BYTE1
SERIN portb.5,N9600,["3"],rhiz.BYTE0
SERIN portb.5,N9600,["4"],rhiz.BYTE1

'sensor Karari
SERIN portb.4,N9600,["L"],lref
SERIN portb.4,N9600,["R"],rref

serout PORTB.6,N9600,["L=",#lhiz,"R=",#rhiz,13,10]

lfeed_back = 0
rfeed_back = 0
lpwm = 0
rpwm = 0
If lref <= lhiz Then
lfeed_back = (lhiz - lref)

```

```
lpwm = ((lref - lfeed_back) * 255) / lmax
End If

If rref <= rhiz Then
rfeed_back = (rhiz - rref)
rpwm = ((rref - rfeed_back) * 255) / rmax
End If

If lref > lhiz Then
lfeed_back = (lref - lhiz)
lpwm = ((lref + lfeed_back) * 255) / lmax
End If

If rref > rhiz Then
rfeed_back = (rref - rhiz)
rpwm = ((rref + rfeed_back) * 255) / rmax
End If

serout PORTb.6,N9600,["lref=",#lref,"rref=",#rref,13,10]
serout PORTb.6,N9600,["Lpwm=",#lpwm,"Rpwm=",#rpwm,13,10]

If lpwm > 255 Then lpwm = 255
If rpwm > 255 Then rpwm = 255

HPWM 1, lpwm, 20000
HPWM 2, rpwm, 20000

i = i + 1

'if i=5 then lref=0:rref=45
'if i=10 then lref=30:rref=30
'if i=15 then lref=45:rref=0
'if i=20 then lref=30:rref=30:i=0

GoTo start

End
```

4. Fuzzy Mobile Robot Motion Control with Camera and Ultrasonic Feedbacks Pic Basic Pro Program (see Figure 4.15 & Figure 4.16)

```

*****
!* Name      : sensor(deney1).bas          *
!* Author    : OK & AG                    *
!* Notice    : Copyright (c) 2006         *
!*           : All Rights Reserved        *
!* Date      : 08.06.2006                 *
!* Version   : 2.0                         *
*****
c1 var byte
c2 var byte
c3 var byte
c4 var byte
c5 var byte
c6 var byte
c7 var byte

*****
d0 var byte
d1 var byte
d2 var byte
d3 var byte
d4 var byte
d5 var byte
d6 var byte
d7 var byte

sonuc var byte[7]
fuzzy_ultra var byte [3]
hareketli_mi var byte
enson_gordugu var byte

l_cmu var byte
r_cmu var byte
m_cmu var byte
lref var byte
rref var byte

cmu_x1 var byte
cmu_x2 var byte
cmu_y1 var byte
cmu_y2 var byte
cmu_mx var byte
cmu_my var byte
cmu_pix var byte
cmu_yog var byte

i var word

TRISA=%11111111
TRISB=%00101111
TRISC=%11111111
TRISD=%11111111
TRISE=%00000000

PORTA = 0
PORTb = 0
portc = 0
portd = 0
PORTE = 0

cmu_x1 = 0
cmu_x2 = 0
cmu_y1 = 0
cmu_y2 = 0
cmu_mx = 0
cmu_my = 0
cmu_pix = 0
cmu_yog = 0

pause 500

```



```

camera:

include "modedefs.bas"
serout PORTB.7,N9600,["Basliyor..",13,10]
serout portb.4,N9600,["TC 200 255 0 70 0 70",13]

c1 = 0
c2 = 0
c3 = 0
c4 = 0
c5 = 0
c6 = 0
c7 = 0

lref = 0
rref = 0
hareketli_mi = 0
start:

startc:

while portc.0=1
Wend
while portc.0=0
Wend
pause 2
c1.3=portc.1
c2.3=portc.2
c3.3=portc.3
c4.3=portc.4

pause 4
c1.2=portc.1
c2.2=portc.2
c3.2=portc.3
c4.2=portc.4
pause 4
c1.1=portc.1
c2.1=portc.2
c3.1=portc.3
c4.1=portc.4
pause 4
c1.0=portc.1
c2.0=portc.2
c3.0=portc.3
c4.0=portc.4
'sol algıla

sonuc [5] = 0
if c4.3=0 then sonuc[5]=2
if c3.3=0 then sonuc[5]=4
if c2.3=0 then sonuc[5]=6
if c1.3=0 then sonuc[5]=7
sonuc [6] = 0
if c4.2=0 then sonuc[6]=2
if c3.2=0 then sonuc[6]=4
if c2.2=0 then sonuc[6]=6
if c1.2=0 then sonuc[6]=7

startd:

while portd.3=1
Wend
while portd.3=0
Wend
pause 2
d4.3=portd.4
d5.3=portd.5
d6.3=portd.6
d7.3=portd.7
pause 4
d4.2=portd.4

```

```

d5.2=portd.5
d6.2=portd.6
d7.2=portd.7
pause 4
d4.1=portd.4
d5.1=portd.5
d6.1=portd.6
d7.1=portd.7
pause 4
d4.0=portd.4
d5.0=portd.5
d6.0=portd.6
d7.0=portd.7

'sol algıla
sonuc [3] = 0
if d7.3=0 then sonuc[3]=2
if d6.3=0 then sonuc[3]=4
if d5.3=0 then sonuc[3]=6
if d4.3=0 then sonuc[3]=7
sonuc [4] = 0
if d7.2=0 then sonuc[4]=2
if d6.2=0 then sonuc[4]=4
if d5.2=0 then sonuc[4]=6
if d4.2=0 then sonuc[4]=7

startb:

while portc.5=1
Wend
while portc.5=0
Wend
pause 2
c6.3=portc.6
c7.3=portc.7
d0.3=portd.0
d1.3=portd.1

pause 4
c6.2=portc.6
c7.2=portc.7
d0.2=portd.0
d1.2=portd.1
pause 4
c6.1=portc.6
c7.1=portc.7
d0.1=portd.0
d1.1=portd.1
pause 4
c6.0=portc.6
c7.0=portc.7
d0.0=portd.0
d1.0=portd.1
'sol algıla

sonuc [1] = 0
if d1.3=0 then sonuc[1]=2
if d0.3=0 then sonuc[1]=4
if c7.3=0 then sonuc[1]=6
if c6.3=0 then sonuc[1]=7
sonuc [2] = 0
if d1.2=0 then sonuc[2]=2
if d0.2=0 then sonuc[2]=4
if c7.2=0 then sonuc[2]=6
if c6.2=0 then sonuc[2]=7

serin2 portb.5,16468,[wait("M "),dec cmu_mx,wait(" "),dec cmu_my,wait(" "),dec
cmu_x1,wait(" "),dec cmu_y1,wait(" "),dec cmu_x2,wait(" "),dec cmu_y2,wait(" "),dec
cmu_pix,wait(" "),dec cmu_yog]

rref = 0
lref = 0

If cmu_mx <= 40 Then ' Fuzzy tablosu Sol

```

```

l_cmu = ((40 - cmu_mx) * 100) / 40
rref = rref + (20 * l_cmu) / 100
End If

If cmu_mx >= 40 Then ' Fuzzy tablosu Sag
r_cmu = ((cmu_mx - 40) * 100) / 40
lref = lref + (20 * r_cmu) / 100
End If

If cmu_mx = 0 Then GoTo arama
enson_gordugu = cmu_mx
If hareketli_mi = 0 Then hareketli_mi = 1: rref = 0: lref = 0: GoTo send

'CMU Fuzzy tablosu Orta
If cmu_mx >= 20 And cmu_mx <= 60 Then
If cmu_mx < 40 Then
m_cmu = ((cmu_mx - 20) * 100) / 20
Else
m_cmu = ((60 - cmu_mx) * 100) / 20
End If
rref = rref + (50 * m_cmu) / 100
lref = lref + (40 * m_cmu) / 100
End If

'ultrasonik Fuzzy
if sonuc[1]>sonuc[2] then
fuzzy_ultra[0]=sonuc[1]
Else
fuzzy_ultra[0]=sonuc[2]
End If

if sonuc[3]>sonuc[4] then
fuzzy_ultra[1]=sonuc[3]
Else
fuzzy_ultra[1]=sonuc[4]
End If

if sonuc[5]>sonuc[6] then
fuzzy_ultra[2]=sonuc[5]
Else
fuzzy_ultra[2]=sonuc[6]
End If

For i = 0 To 2
select case fuzzy_ultra[i]
Case 2
fuzzy_ultra [i] = 29
Case 4
fuzzy_ultra [i] = 57
Case 6
fuzzy_ultra [i] = 71
Case 7
fuzzy_ultra [i] = 100
End Select
Next

'Ultrasonic karar tablosu
if sonuc[1]+sonuc[2]>0 then rref = rref+(20*fuzzy_ultra[0])/100
if sonuc[3]>2 and sonuc[4]>2 then rref=0:lref=0:goto send
if sonuc[5]+sonuc[6]>0 then lref = lref+(20*fuzzy_ultra[2])/100

If rref > 50 Or lref > 50 Then 'hızların 50 yi gecmemesi gerekli
If lref > rref Then
rref = rref - (lref - 50)
lref = 50
Else
lref = lref - (rref - 50)
rref = 50
End If
End If

send:

```

```
serout PORTB.6,N9600,["L",lref,"R",rref]
serout2 PORTb.3,16780,["$RF
ULtra[",#sonuc[1],",",#sonuc[2],",",#sonuc[3],",",#sonuc[4],",",#sonuc[5],",",#sonuc[
6],"] CMU[",#cmu_mx,"] L"#lref," R"#rref," END",13,10]

GoTo start

arama:
hareketli_mi = 0
if sonuc[3]>2 and sonuc[4]>2 then rref=0:lref=0:goto send
rref = 20: lref = 0
If enson_gordugu > 40 Then: lref = 20: rref = 0

GoTo send

End
```