15

Transputer Neuro-Fuzzy Controlled Behaviour-Based Mobile Robotics System

E. Al-Gallaf
Department of Electrical and Electronics Engineering, College of Engineering, University of Bahrain, P. O. Box 13184 Kingdom of Bahrain

1. Introduction

1.1 A Transputer Mobile Robotics System

Mobile robots have received a considerable attention from early research community, from (A. Benmounah, 1991), (Maamri, 1991), (Meystel, 1991) up to this instant (Hegazy, et. al, 2004), and (Pennacchio, et. al., 2005). A *fuzzy or neural* control Transputer based control mobile robots has received, rather, little attention. A number of, are (Welgarz,1994), (Probert, et. al, 1989), (Iida and Yuta, 1991), and (Brady, et. al., 1993). Recently, neurofuzzy logic controllers are found well suited for controlling mobile robots, (Rusu et. al. 2003). This is because, they are talented of building inferences even under certain uncertainty and unclear conditions, (Kim, and Trivedi, 1998). Having a hierarchical architecture that divides the neurofuzzy into *several smaller* subsystems will rather condense the negative effect that a large rule-base may have on real-time performance. Problems of *insufficient knowledge* for designing a rule base can be solved by using a neurofuzzy controller. Learning allows autonomous robots to acquire knowledge by interacting with the environment and subsequently adapting their behaviour. Behaviour learning methods are used to solve complex control problems that autonomous robots encounter in an unfamiliar real-world environment.

Neural networks, fuzzy logic, and reinforcement- and evolutionary-learning techniques can be utilized to achieve basic behavioural functions necessary to mobile robotics system. There are large number of recent research in mobile robot fuzzy behavior navigation. For instant, (Willgoss and Iqbal, 1999), have reported the use of neurofuzzy learning for teaching mobile robot behaviors, selecting exemplar cases from a potential continuum of behaviors. Proximate active sensing was successfully achieved with infrared in contrast to the usual ultrasonic and viewed the front area of robot movement. (Pennacchio, et. al., 2005), have presented, the FU.LO.RO., a new controller for mobile robot, that moves itself autonomously in an unknown environments. The system has been developed following two different approaches: first one enables a fuzzy controller to determine the robot's behavior using fuzzy basic rules; the second uses a neurofuzzy controller.

(Tsoukalas, et. al., 1997), have presented a neurofuzzy methodology is for motion planning in semi-autonomous mobile robots. The robotic automata considered are devices whose main feature is incremental learning from a human instructor. Fuzzy descriptions are used for the robot to acquire a repertoire of behaviors from an instructor which it may subsequently refine and recall using neural adaptive techniques. The robot is endowed with sensors providing local environmental input and a neurofuzzy internal state processing

Source: Mobile Robots, Moving Intelligence, ISBN: 3-86611-284-X, Edited by Jonas Buchli, pp. 576, ARS/plV, Germany, December 2006

predictable aspects of its environment. Although it has no prior knowledge of the presence or the position of any obstructing objects, its motion planner allows it to make decisions in an unknown terrain. (Hegazy, et. al, 2004) have shown controlling mobile robot navigation system that operates in an unknown and uncertain environment is a difficult operation. Much of this difficulty is due to environmental inconsistencies and sensor inadequacies. Training data was accumulated from robots sensors to generate a set of fuzzy rules that govern the robot navigation system on-line. A Transputer-based (T-805) locomotion module provides all of motor feedback and control of a robot (Iida and Yuta, 1991). Locomotion module was designed to follow a given trajectory, using feedback information from the robot's wheel encoders. The locomotion module operates as a digital PID controller to govern motion of the robot.

1.2 Research Outline

In this respect, this chapter discusses a neurofuzzy controller strategy for sensor-based mobile robotics system navigation for an indoor environment applications. A Transputer computation power is used to carry out complicated needed computation (reading sensors data, deciding actions, outputting wheels data, … system monitoring).

Robot control mythology was run on a parallel computing environment known as Transputers. The Transputer embedded real-time controller was used on board the robot to meet various intelligence requirements for the free navigation and obstacle avoidance. The control system consists of a hierarchy of robot behaviours. The mobile behavior control system was based on the use of a number of Transputers processors. Behavior methodology was based on the utilization of the structure of a five layers neuro-fuzzy system that learns, trains, and adapts itself to the environment within which it operates for the purpose of robot body maneuvering.

The autonomous mobile robot uses ultra-sonic sensors for detecting targets and avoiding collisions. The control system is organized in a top-bottom hierarchy of various tasks, commands, and behaviours. When multiple low-level behaviours are required, command fusion is used to combine the output of several neuro-fuzzy sub-systems. A switching coordination technique selects a suitable behaviour from the set of possible higher level behaviours. A parallel (Transputers based) fuzzy control is implemented for the robot guidance and obstacle avoidance. The mobile robot used in this work has been designed and constructed by the author at the University of Bahrain. The key issue of this research frame work is the utilization of a neurofuzzy system that runs over a parallel Trasputers. This has shown the ability to reduce the computational time needed for the movement.

2. The Mobil Robot

2.1 (Experimental Testbed) Physical Parameters

The mobile robot can be seen in Fig. 1. It is a small mobile autonomous robotic-Testbed (AL-Gallaf, 2006), is utilized to achieve defined robot behaviors. It has a Transputer system allowing high level control consisting of C++ , Matlab, and Occam routines to provide a multitude of functions. Its drive wheels are driven with a 10 : 1 gear ratio to reach motor torque of 10 N/m. The maximum speed it can reach is 0.7 m/s. The mobile robot weighs 2Kg in a rectangular shape of width 30 cm and length of 40 cm. It has two moving wheels of diameter 10cm located at the center of the robot used for motion and

steering. Shaft encoders are attached to each motor for wheel sensing. Free-wheeling castors are located one in the front and one at the back of the robot to balance the robot. Corners of the robot circumference are flat and with 45° angle from the adjacent flat wall. These flat corners are used to mount the corner ultrasonic sensors in order to enable the robot to see in that direction. Photograph of the mobile robot is shown in Fig. 1. The mobile robot must be capable to follow an (x-y) path in any direction (θ) over the plane. It should have one degree of translation and one degree of rotation. All steering axes are perpendicular to the surface. The mobile robot can follow a path in any direction, first it must rotate around its axis to face that direction. It has two diametrically opposed drive wheels. Because of the simplicity of its mechanical design it has simple kinematics.

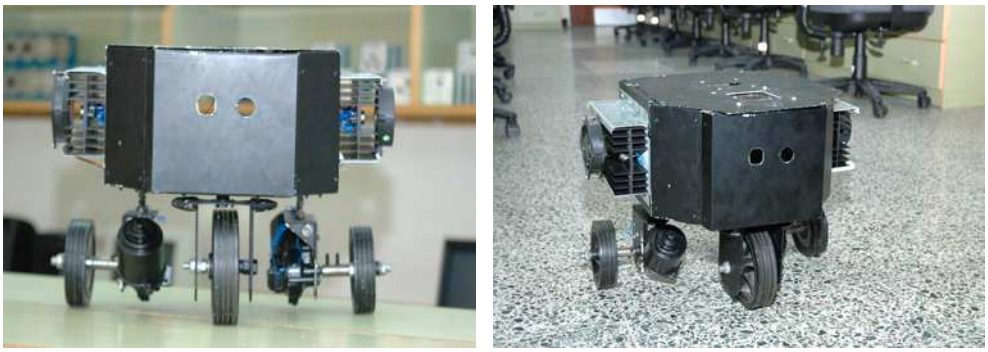

Fig. 1. (A Testbed) Mobile Robot Construction.

2.2 Wheel Model Type

Three types of wheels are used in the wheeled mobile robot design: Conventional, unidirectional, and ball wheels. The robot conventional wheels are used on a fixed axis therefore, there is no steering joint. The conventional wheels are modeled by a planner pair at the point of contact (McKerrow, 1991). With this representation, the multiple degrees of freedom of wheel motion can be modeled without ambiguities in the transformation matrices. A conventional wheel has only two degrees of freedom, because the third degree of freedom in the planar pair model is eliminated when the x component of the wheel velocity is set to zero to eliminate sideways slip. The y component of the wheel velocity is equal to the angular velocity times the radius of the wheel, ($y_x = \omega_z \times r$). The y component of wheel velocity allows travel along a surface in the direction of the wheel orientation. The conventional wheel is by far the most widely used wheel.

2.3 Robot Sensors

Robot sensors are mounted around the circumference of the robot as shown in Fig. 2. They are at 30 centimeters high from the ground floor. Sensors are slightly tilted upward to prevent ultrasonic echo reflection from the ground floor. The mobile robot is powered by four rechargeable 12V batteries, $2\times12A_h$ and $2\times6A_h$. These batteries are configured to supply +12V, –12V, +24V and –24V. A voltage converter is supplied to provide +5V.

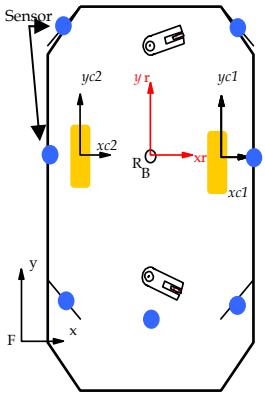

Fig. 2. The Mobile Robot Sensing.

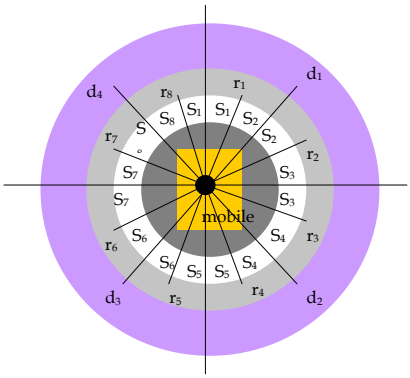

Fig. 3. Sensors grouped in four region.

2.4 The Sensor Fusion

The sensor fusion is the integration of all of the eight ultrasonic sensor to form the inputs to the neuro-fuzzy controller. Robot sensors are grouped in to four regions as shown in Fig. 3. Region-1 consist of sensors (s_1, s_2, and s_3). Region-2 consist of sensors (s_3, s_4, and s_5). Region-3 consist of sensors (s_5, s_6, and s_7). Region-4 consist of sensors (s_7, s_8, and s_1). In this arrangement some sensors are part of more than one region and this grouping is consistent with the neuro-fuzzy group classification where the borders between the sets (regions) is not crisp, (Fig. 3.). In the process of selecting a region out of the four, the following steps are implemented :

- *Read all the sensors (s_1, s_2, to s_8).*
- *Arrange sensor outputs value in ascending order.*
- *Identifying the three sensors with the lowest value.*
- *If two of the three are in one region, that region will be selected.*
- *If no, go to first step.*

3. Mobile Robots Kinematics Modeling

To provide a framework within which to develop the robot kinematics models, (Muir and Neuman, 1986) defined a wheeled mobile robot as : " A robot capable of locomotion on a surface solely through the action of wheel assemblies mounted on the robot and in contact with the surface. A motion of a robot is determined from geometry of the constraint imposed by the wheels motion. Kinematic analysis is based on the assignment of coordinate axes within the robot and its environment, and the application of (4×4) matrices to transform between coordinate systems. Kinematic model is derived with respect to coordinate axes assigned to each robot joint as will be shown in the following sections.

3.1 Robot Coordinate Frames Assignment

Coordinate frame is assigned at each link of the mobile robot. The mobile robot links are the floor and the robot body, Fig. 4. These links are connected by two joints: the wheel contact point with the floor and the mid-point of the robot. The mid-point of the robot is not a physical joint, but the relationship between the body of the robot and the floor is

modeled as a planar pair located at the center of the robot. The wheel is also modeled as a planner pair located at the point of contact between the wheel and the floor. The z axis of all these frame is vertical, and is neglected in two-dimensional analysis. The instantaneously coincident coordinate systems (frames C_F and R_F) are stationary coordinate frames located at the same point as the moving coordinate frame at the instant of observation. Position transform between the two frames is zero. These frames are instantaneously fixed with respect to floor and not to the robot. At the instant these frames are considered, they are coincident with the frames attached to the robot. Frame R_F coincides with frame R_B and frame C_F coincides with frame C_L. The driving wheels are fixed to the body, the steering frames do not move with respect to either the robot body frame or wheel contact frame. Floor coordinate frame F is stationary and serves as a reference frame for robot motion. Robot frame R_B is located at the center of the robot, and serves to define the location of the robot with respect to the floor frame for the kinematics. Fig. 5. shows the coordinate frames assignment for the mobile robot seen from top view. Fig. 6. is the side view of the same system.

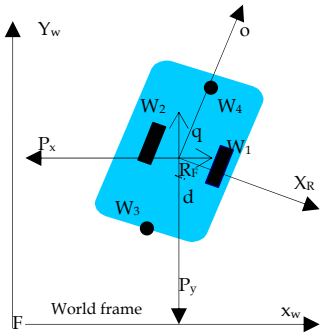

Fig. 5. Robot and World frames Coordinates.

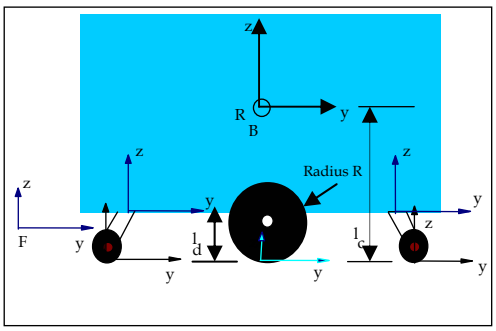

Fig. 6. The coordinate frames assignment for the mobile robot (side view -y axes point out of page).

4. Transformation Matrices

Modeling uses homogeneous transforms to describe the transformation. Homogeneous transformation matrices ($\in \Re^{4\times4}$) express the relative positions and orientations of coordinate systems (Beom & Cho, 1995). The homogeneous transformation matrix ${}^{A}\Pi_{B}$ transforms the coordinates of the point ${}^{B}r$ in coordinate frame B to its corresponding coordinates ${}^{A}r$ in the coordinate frame A:

$$ {}^{A}r = {}^{A}\Pi_{B}\,{}^{B}r \tag{1} $$

Vectors ${}^{A}r$ and ${}^{B}r$ denote points in space consist of three Cartesian coordinates and a scale factor as the fourth element, the scale factor is always unity:

$$ A_r = \begin{pmatrix} A_{r_x} \\ A_{r_y} \\ A_{r_z} \\ 1 \end{pmatrix} \tag{2} $$

The transformation matrices contain the $\in \Re^{3\times3}$ rotational matrix (n o a), and $\in \Re^{3\times1}$ translational vector p :

$$ {}^{A}\boldsymbol{\Pi}_{B} = \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3} $$

The three vector components n, o and a of the rotational matrix in (3) express the orientation of the x, y and z, respectively, of B coordinate systems are relative to the A coordinate system. The three components p_x , p_y, and p_z axes of the translational vector p express the displacement of the origin of the B coordinate system relative to the origin of the A coordinate system along the (x, y, and z) axes of the A coordinate system, respectively. All the coordinate systems are assigned to the robot with z axes perpendicular to the travel surface. All rotations between coordinate system are about the z axis. Because of the robot three-dimensional shape, there are translations in all three directions. Thus, a general transformation matrix for the mobile robot between the robot body frame (R) and floor frame (F) is given by (Muir and Neuman, 1987) :

$$ {}^{R}T_{F} = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & p_x \\ \sin\theta & \cos\theta_3 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4} $$

where θ is the rotational angle between robot frame and floor frame (Fig. 5.) and equal to $\omega \times r$. For zero rotational and translation displacements, the coordinate transformation matrix in Equ (1) reduces to an identity matrix. The velocity transformation matrix is calculated by differentiation of Equ (4) matrix component wise, to give :

$$ {}^{R}V_{F} = \begin{bmatrix} -\omega\sin\theta & -\omega\cos\theta & 0 & v_x \\ \omega\cos\theta & -\omega\sin\theta & 0 & v_y \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5} $$

where: ω = The angular velocity of the wheel, v_x = robot component of the linear velocity, v_y = robot y component of the linear velocity. From above matrix transformation, the robot linear velocity can be derived as follows (Reister & Pin, 1994) :

$$ \theta = \frac{r}{d}(\omega_R + \omega_L) \tag{6} $$

$$ v_x = (\omega_R + \omega_L) \times \cos(\theta/2) $$
$$ v_y = (\omega_R + \omega_L) \times \sin(\theta/2) $$

in which (r is the radius of each wheel) and (d is the distance between the two wheels). In Equ (6), ω_R and ω_L are wheel right and left angular velocities, respectively in radians per second.

5. The Transputer Based Parallel Controller

The mobile robot is controlled by a neuro-fuzzy system which runs on a trasputer (a parallel processing computing). Within this section we shall introduce the Occam.

5.1 The Trasputer

The architecture of the Transputer **T414,** Fig. 7., is rather simple and borrows architectural ideas from Texas Instrument's TMS 9000 microcomputer and the old Hewlett-Packart calculators. It has 32 bit 10 MIPS processor, 4 Gigabyte linear address space, 32 bit wide 25 MByte/sec memory interface, configurable on-chip memory controller, 2Kbytes high speed on chip RAM, 4 Inter-trasputer links, each with full duplex DMA transfer capability up to 20 Mbits/sec., advanced 1.5 micro CMOS technology, and low power dissipation (less than 500 mw).

The T414 is a 32 bit Trasputer capable of executing up to 10 MIPS at a speed of 20 MHZ, Fig. 7. From the Occam model, Inmos developed a hardware chip to carry out their concurrency model. This hardware is in the form of a very large scale integration (VLSI) integrated chip (IC) called the Transputer. The Transputer (Inmos part number T800) was a 32-bit microprocessor (20 MHz clock) that provides 10 MIPS (million instructions per second) and 2.0 MFLOPS (million floating point operations per second) processing power with 4K bytes of fast static RAM (Random Access Memory) and concurrent communication capability all on a single chip. Communication among processes is done by means of channels, Fig. 8. A channel between processes executing on the same Transputer is a soft channel, while a channel between processes executing on different processors is a hard channel.

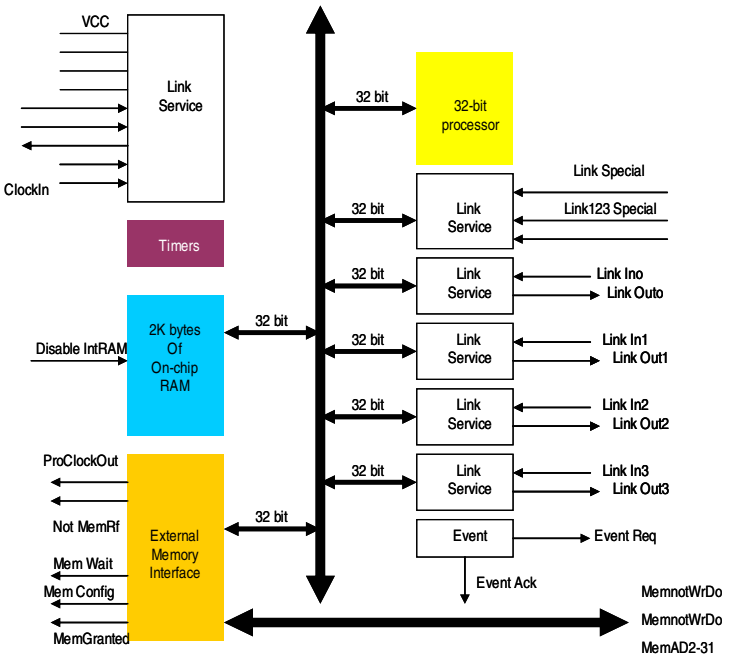

Fig. 7. A Transputer System.

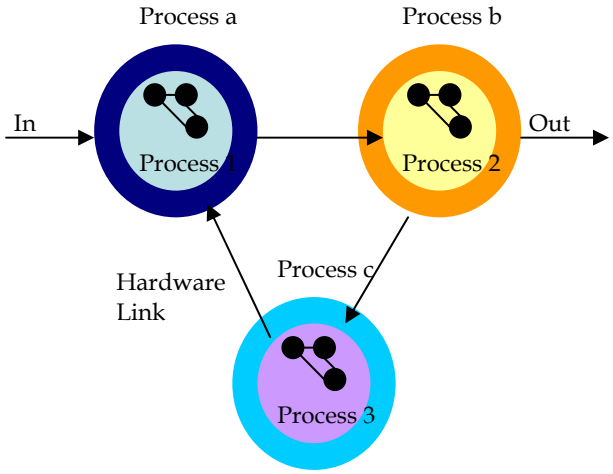

Fig. 8. Occam Processes on Separate Machines.

5.2 The Occam

Occam language enables programs to be written as a collection of self-contained programs (tasks) which may be executed simultaneously (parallel or concurrent) or simply one after another with a built-in inter-process communication mechanism, depending whether these processes are executed on a single machine or on separate ones. Occam is a high-level language, it can be viewed as the assembly language for the Transputer. Unlike most microprocessors, e. g., the M68000, the definition of the operations of the Transputer is in terms of the Occam model and not machine language. Besides being a high performance microprocessor (*half the speed of a VAX 8600*), the Transputer has on its chip four (4) serial bi-directional links (*each 20 Megabits per second*) to provide concurrent message passing to other Transputers. The "channels" in the Occam language are mapped to these hardware links which connect by way of twisted pairs of wires to other Transputers. Transputer hardware supports concurrency by scheduling (*time-slicing*), in round-robin fashion, an arbitrary number of Occam concurrent processes. The language and the hardware are so designed that an Occam program consisting of a collection of concurrent processes may execute on one Transputer (via time slicing between the different concurrent processes) or be spread over many Transputers with little or no change in the Occam code. Consequently, the designer can build up his Occam program on one Transputer, and if higher performance is required, can spread the Occam processes over a network of interconnected Transputers. The original Transputer (T414), having no floating point unit and only 2 Kbytes of RAM. Inmos has developed a new, faster version of the Transputer called the (T9000). The T9000 is a 150 MIPS microprocessor with a 20 MFLOPS floating point unit.

6. Neuro-Fuzzy Control System

Rule-based structure of fuzzy models allows for integrating heuristic knowledge with information obtained from process measurements. Fuzzy sets are used to define the process operating conditions such that fuzzy dynamic model of a nonlinear process can be described in the following way :

$$\begin{aligned} R_i: \quad & If\ operating condition i \\ Then \quad & \hat{y}_i(k) = \sum_j^o a_{ij} s(k-j) + \sum_j^i b_{ij} u(k-j) \qquad (i = 1,2,\ldots,r) \end{aligned} \tag{7}$$

Final model output is obtained by the center of gravity defuzzification as follows:

$$\hat{y}(k) = \frac{\sum_{i=1}^{r} \mu_i \hat{y}_i(k)}{\sum_{i=1}^{r} \mu_i} \tag{8}$$

In Equ (7) and Equ (8), y is the system output, u is the system input, $\hat{y}_i$ is the prediction of process output in the i^{th} operating region, (*r*) is the number of fuzzy operating regions, (*i*) and (*o*) are the time lags in the input and the output, respectively, μ_i is the membership function for the i^{th} model, and finally, a_{ij} and b_{ij} are the ARMAX model parameters. The membership function for an operating region is constructed in a number of ways. One approach to calculate its membership function as follows :

$$\mu_i = \min\left(\mu_h(\mathbf{x}), \mu_m(\mathbf{y})\right) \tag{9}$$

$$\mu_i = \mu_h(\mathbf{x}), \mu_m(\mathbf{y}) \tag{10}$$

In Equ (9), μ_i is a membership function in i^{th} operating region, $\mu_h(\mathbf{x})$ is membership function of *x* being (*high*), and $\mu_m(\mathbf{y})$ is the membership function of *y* being (*medium*). In Equ (10), we calculate gradient required for gradient based network training.

6.1 Neural Network Representation of Fuzzy Systems

Fuzzy Systems can be represented by a special type of network topology which is termed here a neuro-fuzzy. Fuzzy reasoning is capable of handling uncertain and imprecise information while a neural network is capable of learning from examples. Neuro-fuzzy intend to combine the advantages of both fuzzy reasoning and neural networks.

6.2 Neuro-fuzzy Architecture

For simplicity, it is assumed, the fuzzy inference system under consideration has large number of inputs, e.g. (d_1, d_2, d_n, sensors data, s_1, s_2, … s_n) and two outputs y_1 , y_2. If a rule base contains two fuzzy if-then rules of Takagi and Sugeno's type. A rule as:

rule 1: If d_1 is A_1 *and* d_2 is B_1, *then* $y_1 = p_1 s(1) + q_1 s(2) + r_1$

rule 2: If d_3 is A_2 *and* d_n is B_2, *then* $y_2 = p_2 s(1) + q_2 s(2) + r_2$

where *p, q,* and *r* are constants and called parameter set. That is, the if parts of the rules are same as in the ordinary fuzzy if-then rules, then parts are linear combinations of the input variables. The employed neuro-fuzzy architecture is shown in Fig. 8., where node functions in the layers are described below :

Within first layer: Each i^{th} node in this layer is a square node with a node function :

$$O_i^1 = \mu A_{\mathbf{x}} s(1) \tag{11}$$

where s_1 is the input to i^{th} node, and A_i is the linguistic label (small , large, .. etc.) associated with this node function. In other words, O_i^1 is the membership function of A_i and it specifies the degree to which the given s satisfies the quantifier A_i. Usually we choose $\mu A_i(s(1))$ to be bell-shaped with maximum equal to 1 and minimum equal to 0, such as :

$$\mu A_i s((1)) = \frac{1}{1+\left[\left(\frac{s(1)-c_i}{a_i}\right)^2\right]b} \tag{12}$$

or

$$\mu A_i(s(1)) = \exp\left\{-\left(\frac{s(1)-c_i}{a_i}\right)^2\right\} \tag{13}$$

where $\{a_i, b_i, c_i\}$ is the parameter set. As values of these parameters change, membership shaped functions vary accordingly, thus exhibiting various forms of membership functions on the linguistic label A_i. In second layer, every node in this layer is a circle node which multiplies incoming signals and sends their product out. For instance,

$$\mathbf{y}_i = \mu A_i s(1) \times \mu B_i s(2) \qquad i = 1,2 \tag{14}$$

Each node output represents the firing strength of a rule. In third layer, every node in this layer is a circle node. The i^{th} node calculates the ratio of the i^{th} rule's firing strength to the sum of all rules' firing strengths :

$$\bar{y}_i = \frac{y_i}{y_1 + y_2 + \cdots + y_i} \qquad i = 1,2 \tag{15}$$

For fourth layer, every node i in this layer is a square node with a node function

$$O_1^4 = \bar{y}_i f_i = \bar{y}_i (p_i s(1) + q_i s(2) + r_i) \tag{16}$$

where $\bar{y}_i$ is the output of third layer, and $\{p_i, q_i, r_i\}$ is the parameter set. Parameters in this layer will be referred to as consequent parameters. Finally, the fifth layer, the node in this layer is a circle node. It computes the overall output as the summation of all incoming signals, i.e. :

$$O_1^5 = overall output = \sum_i \bar{y}_i f_i = \frac{\sum_i y_i f_i}{\sum_i y_i} \tag{17}$$

Thus we have constructed an adaptive network which is functionally equivalent to a fuzzy inference system, achieved by neural system alone. This is shown in Fig. 9.

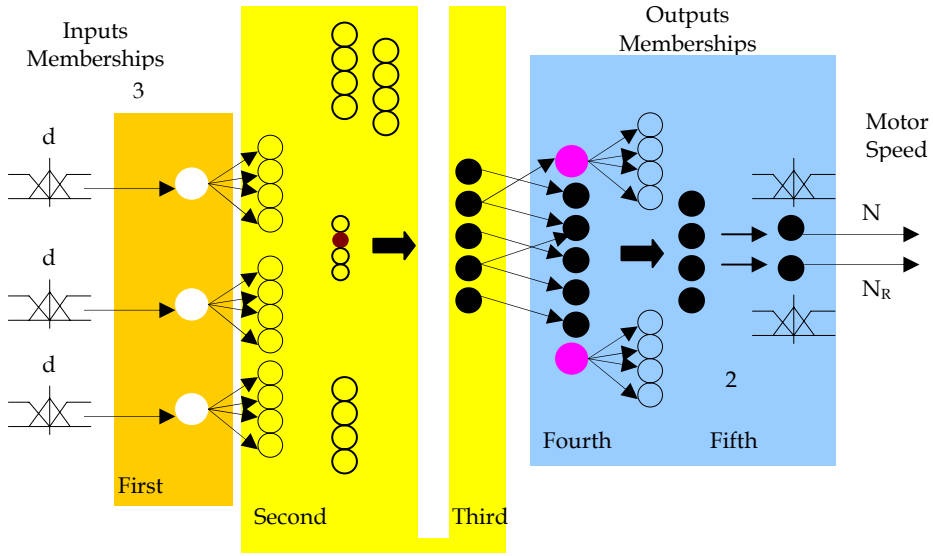

Fig. 9. Employed Neuro-fuzzy system.

6.3 Neuro-fuzzy Training

From the shown Neuro-fuzzy architecture shown in Fig. 9., it is observed that, given values of premise parameters, the entire output is expressed as a linear combinations of the consequent parameters. More precisely, output $\hat{\mathbf{y}}$ in Fig. 9. is rewritten as :

$$\begin{aligned}
\hat{y}_m &= \frac{y_1}{y_1+y_2}\hat{y}_1 + \frac{y_2}{y_1+y_2}\hat{y}_2 + \cdots + \frac{y_{m-1}}{y_m+y_{m-1}}\hat{y}_{m-1} \\
\hat{y}_m &= \bar{y}_1\hat{y}_1 + \bar{y}_2\hat{y}_2 + \ldots + \bar{y}_{m-1}\hat{y}_{m-1} \\
\hat{y}_m &= (\bar{y}_1 s(1))p_1 + (\bar{y}_1 s(2))q_1 + (\bar{y}_1)r_1 \\
&\quad + (\bar{y}_2 s(1))p_2 + (\bar{y}_2 s(2))q_2 + (\bar{y}_2)r_2 + \cdots
\end{aligned} \tag{18}$$

which is linear in the consequent parameters $(p_1, q_1, r_1, p_2, q_2 and\, r_2)$. The consequent parameters thus identified are optimal (*in the consequent parameter space*) under the condition that the premise parameters are fixed. A model's weights are conventionally identified by performing maximum likelihood estimation. Given a training data set $\mathbf{Z}^N = \{\mathbf{y}(\boldsymbol{k}), \mathrm{s}(\boldsymbol{k})\}_{k=1}^N$ the task is to find a weight vector which minimizes the following cost function of Equ (19) :

$$J_N(\mathbf{w}) = \frac{1}{N}\sum_{k=1}^{N}[\mathbf{y}(k) - \hat{\mathbf{y}}(s(k), \mathbf{w})]^2 \tag{19}$$

As the model, $\hat{\mathbf{y}}(\boldsymbol{s}(\boldsymbol{k}), \mathbf{w})$, is nonlinear with respect to the weights, linear optimization techniques cannot be applied. Instead the popular Truncated Newton nonlinear optimization algorithm is employed. All the five operations are computed via the Trasputer system, where is capable of computing fuzzy inputs in a concurrent way.

7. The Mobile Robot Controller

7.1 Over ALL Control

Eight ultrasonic sensor range measurement data (s_1, s_2, …s_n) are input to the controller (*Neuro-fuzzy*) as (d_1, d_2, …d_n). Outputs are the left and the right wheel speeds, (N_L and N_R). This controller is integrated with the sensor fusion and the low level PID controller to form the complete controller for the mobile robot as shown in Fig. 10.

The overall controller starts with a user interface where the user can enter the desired control parameters. In this interface the statues of the mobile robot will be displayed before starting the controller.

The robot checking routine will perform self testing on the various parts of the hardware of the robot and report any difficulty before starting. This section forms the very high level of the controller servo.

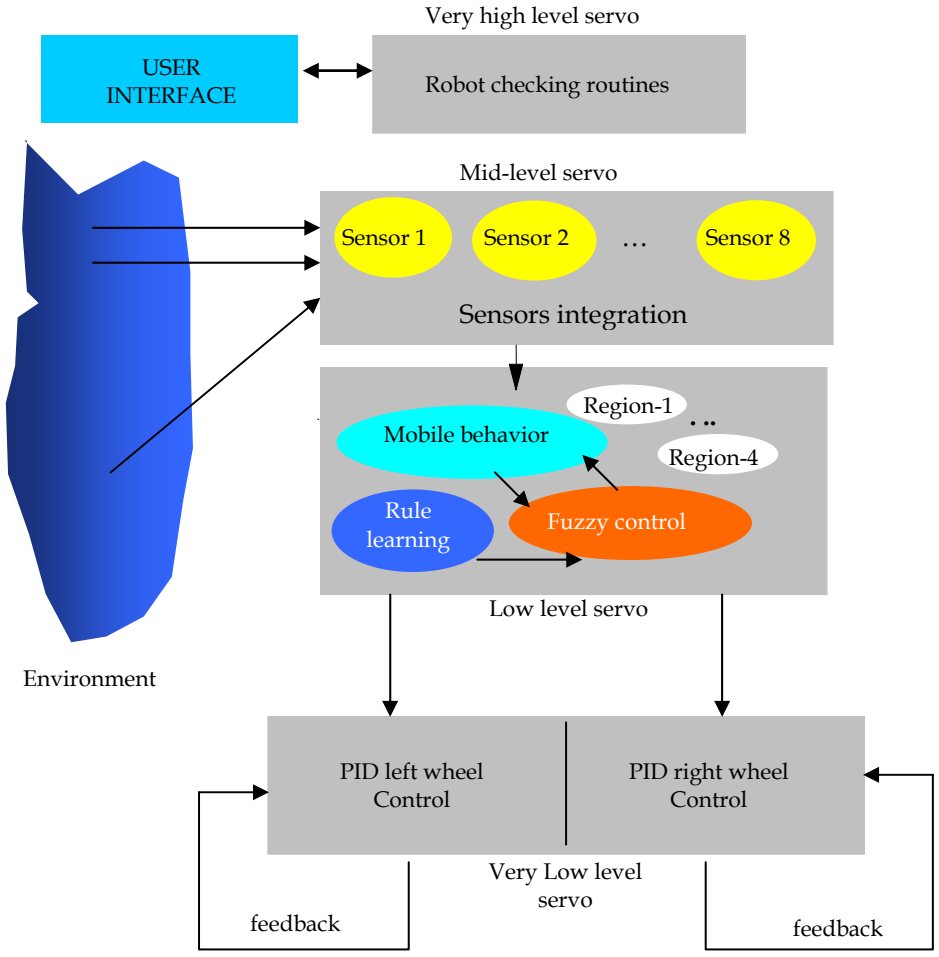

Fig. 10. Overall robot controller.

7.2 Fuzzy Control Information Processing

The processing of sensors information in the neuro-fuzzy process will first include the fuzzification of the sensors input variables (s_1, s_2, … s_n), then the defuzzification of the output variables as follows: First, the fuzzification of the senor (s_1 and s_2) to input variables (d_{min1} and d_{min2}). Once the region is selected the information supplied by the sensor in that region will indicate the nearest obstacle to the robot and the next nearest obstacle (d_{min1} and d_{min2}) facing that region. The mobile control can be realized by the control of the robot wheels rotation to keep the robot within a pre-set distance from the obstacle facing that region of the robot. The neuro-fuzzy operator converts the crisp senor input data, (say{d}), into the linguistic values $\hat{(d)}$ determined as labels of fuzzy sets given by Equ (20) :

$$fuzzifier(d_1, d_2) \qquad (\hat{d_1}, \hat{d_2}) \tag{20}$$

where fuzzifier denotes a Fuzzification operation. From now on, the sign (^) representing the fuzzy set will be omitted for simplicity. The sensors input linguistic variables d_1 and d_2 are expressed by linguistic terms as illustrated by the membership function in Equ (21) and Equ (22), respectively. Meaning of the linguistic term is given in Table. 1.

$$\{es\ vvs\ vs\ s\ m\ b\ vb\ vvb\ eeb\} \in \mathrm{m}\ (d_1) \tag{21}$$

$$\{es\ vvs\ vs\ s\ m\ b\ vb\ vvb\ eeb\} \in \mathrm{m}\ (d_2) \tag{22}$$

symbols	**Linguistic variable**
es	*extremely small*
vvs	*very very small*
vs	*very small*
s	*small*
m	*medium*
b	*big*
vb	*very big*
vvb	*very very big*
eeb	*extremely big*

Table 1. Fuzzy linguistic terms of Inputs (d_1, d_2, ..d_8).

The controller outputs are right and the left wheels speed (N_L and N_R) in rad/sec. They are expressed by linguistic values with membership functions having bill shape functions. Halfway of this membership function is determined by zero initial value. The linguistic provisions for output variables are given in Equ (23) and in Table. 2. :

$$\{bn, n, mn, z, mp, p, bp\} \subseteq \mathrm{m}(N_L) \tag{23}$$

symbols	Linguistic variable
bn	*big negative*
n	*negative*
mn	*medium negative*
z	*zero*
mp	*medium big*
p	*positive big*
bp	*big positive*

Table 2. Output variables N_L and N_R.

In Equ (23), the eleven states membership function are expressed in terms of degree of membership function (MF). Fuzzy subsets embody elements with degree of membership. On the other hand, fuzzy FM $\mu_z(\square)$ of the fuzzy set{□}, assigns a real number (between 0 to 1) to every element in the universe of discourse.

7.3 Construction of the Rule base Fuzzy System

The rule base for realizing each behavior can be constructed based on operator knowledge. For the fuzzy controller, the partial mapping is to be translated in to linguistic rules. A fuzzy rule has an IF-THEN format as follows :

$$\text{IF } (d_1 \textit{ is } mb \text{ AND } d_2 \textit{ is } vb) \text{ then } (N_L \textit{ is } vvb \text{ AND } N_R \textit{ is } vb)$$

where (d_1, d_2) , (N_L, and N_R) are the fuzzy variables and (*mb, vb, vvb and vb*) are the fuzzy subsets in the universe of discourses *X, Y* and *Z*. A fuzzy rule base consists of several rules of the form given above. The experience of the operator play a big role in the shape and form of the rules and in the number of the rules, in the rule base fuzzy controller. In this fuzzy controller, membership functions (μ_1, μ_2, μ_3, μ_4) with association with the (d_1 and d_2) are computed as follows (Baxter, 1995) :

$$x = int((d_1+1)/10) \tag{24}$$

$$y = int((d_2+1)/10) \tag{25}$$

where *int* is the integer part of an expression :

$$xx = (10x + 5 - d_1)$$

$$yy = (10y + 5 - d_2)$$

and the complement of these fuzzy sets is given by :

$$C_x = 1 - xx \text{ and } C_y = 1 - yy \quad \mu_i = \mu_i(d_1) \oplus \mu_i(d_2) \tag{26}$$

$$\mu_1 = min\ (xx,yy),\ \mu_2 = min\ (xx,C_x),\ \mu_3 = min\ (C_x,C_y),\ \mu_4 = min\ (C_x,C_y) \tag{27}$$

Equ (26) and Equ (27) is the fuzzy OR operator for the fuzzy set that select a minimum of the two variables. Membership functions of fuzzy variables (as in Equ (23)) are used in the defuzzification process (Zinger and Elbuluk, 1994).

8. Performance Validation

8.1. Behavior Learning From Samples of Robot Movements

Following the five layers neuro-fuzzy system was assembled, inputs to this system are sensory information coming from the robot. They are (d_1, d_2, and d_3). This rather represents associated distances and location the robot is with respect to obstacles. This neuro-fuzzy system was run by a net of Transputers through an Occam programming environment. Now the mobile robot is ready to be trained. First, the mobile robot was moved arbitrarily within the space. Sensors data were collected. This rather represents large number of data (since eight sensors were monitored simultaneously). Fig. 11-a shows some robot movement. Typical wheels speeds were also set, hence defining some preset values of the robot movement. Fig. 11-b shows such low level real-time step response of one wheel (recorded via one Transputer node). The wheel was controlled by a digital PID controller.

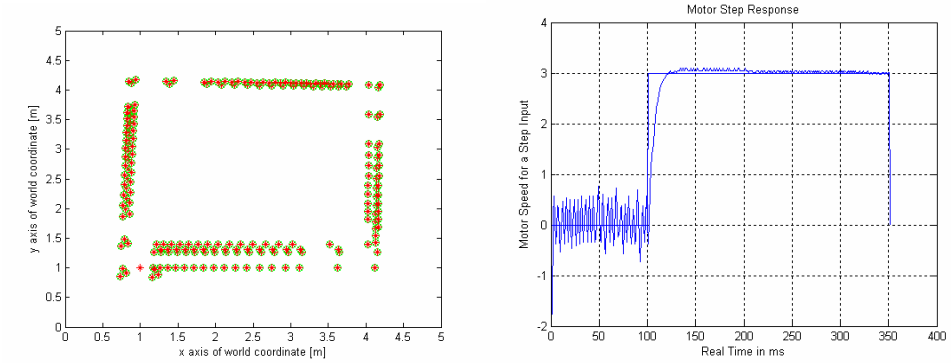

Fig. 11. Overall robot controller.
(a: Randomly moved robot), (b: Typical low level wheel control)

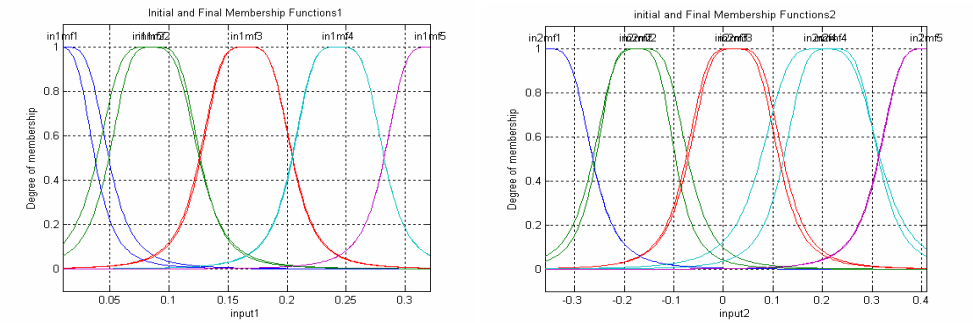

Fig. 12. Initial and final tuned memberships.
(a: Sensor data recording, as input to NF, e.g. senor data d_1), (b: Wheel speeds N_L)

Accordingly, Fig. 12 shows the initial and finally tuned memberships of fuzzy variables (d_1, d_2, and d_3). The initial memberships are set by the user himself, whereas, the final ones are as results of the neuro-fuzzy tuning action. By now, the robot is trained. For the purpose of

making sure the mobile robot has learned the surrounding environment, Fig. 13-a. shows a computed errors between the trained neuro-fuzzy system and the robot actual wheel speeds. The computed error is small enough to tell that the mobile robot has learned the fuzzy if then rules via the five layers.
On the other hand, Fig. 13-b. shows the relation between two different inputs of learned neuro-fuuzy system. This 3-D plot represents in fact what will be the associated mobile wheel speeds (N_L and N_R outputs) for any two combinations of inputs (d_1, d_2, and d_3). The plot can further shows the learned expert if-then rules. This will be useful batch of information once the trained mobile intelligence is compared with only fuzzy (*if-then*) rules.

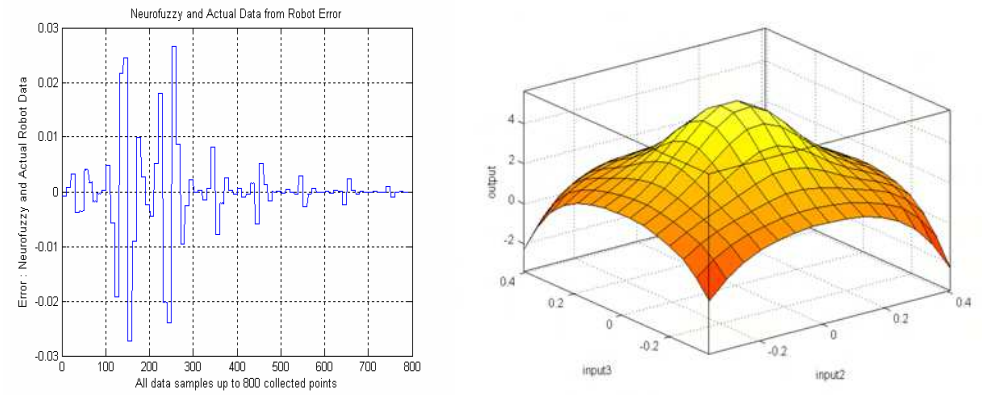

Fig. 13. Learned five layers Neuro-fuzzy system.
(a: Trained Neuro-fuzzy and actual robot error), (b: Learned fuzzy if-then 3-D plot)

8.2 Robot Posture Seeking

An obstacle avoidance and posture seeking behaviors have been simulated and run experimentally. Using neuro-fuzzy controller, the robot was run over an experimental of ($10m \times 10m$), with some obstacles. The code (Occam) begins by inputting an initial *x-y* locality, an orientation angle with respect to an *x-y* world coordinate, and the needed final posture. Results have shown that, robot moves around the plane seeking the final destination controlled by fuzzy-base rules with no colliding with surrounding boundaries. This is shown in Fig. 14. Posture seeking with an ability of speed alteration is shown in Fig. 14-a., whereas, in Fig. 14-b. illustrates robot seeking even with robot orientation change (robot was orientated initially by 90° degree up). This shows that neuro-fuzzy is capable of controlling the robot. The goal seeking algorithm is based on first rotating the robot until it does face the goal if it is not already facing it, than the algorithm try to reduce the distance between the robot and the goal to zero as shown in Fig. 11., where the distance to goal d_g (Baxter and Bumby, 1995) :

$$d_g = \sqrt{(x_g - x_v)^2 + (y_g - y_v)^2} \quad \text{and, heading error } \theta_{he} = (\theta_v + \theta_g)$$

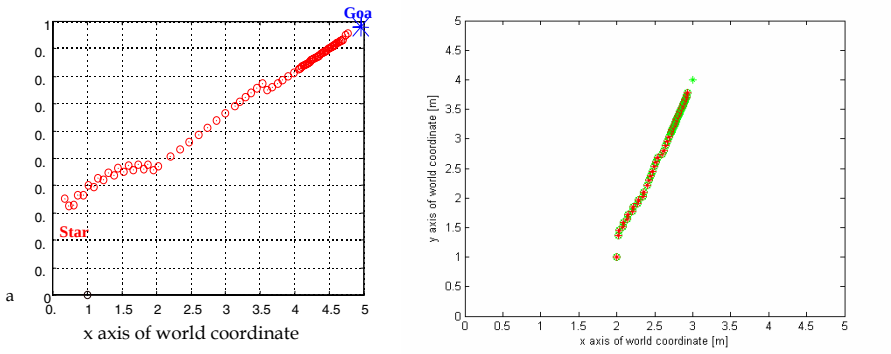

Fig. 14. Robot posture seeking.
(a: Slow in speed close to target),
(b: Posture seeking even; orientation change)

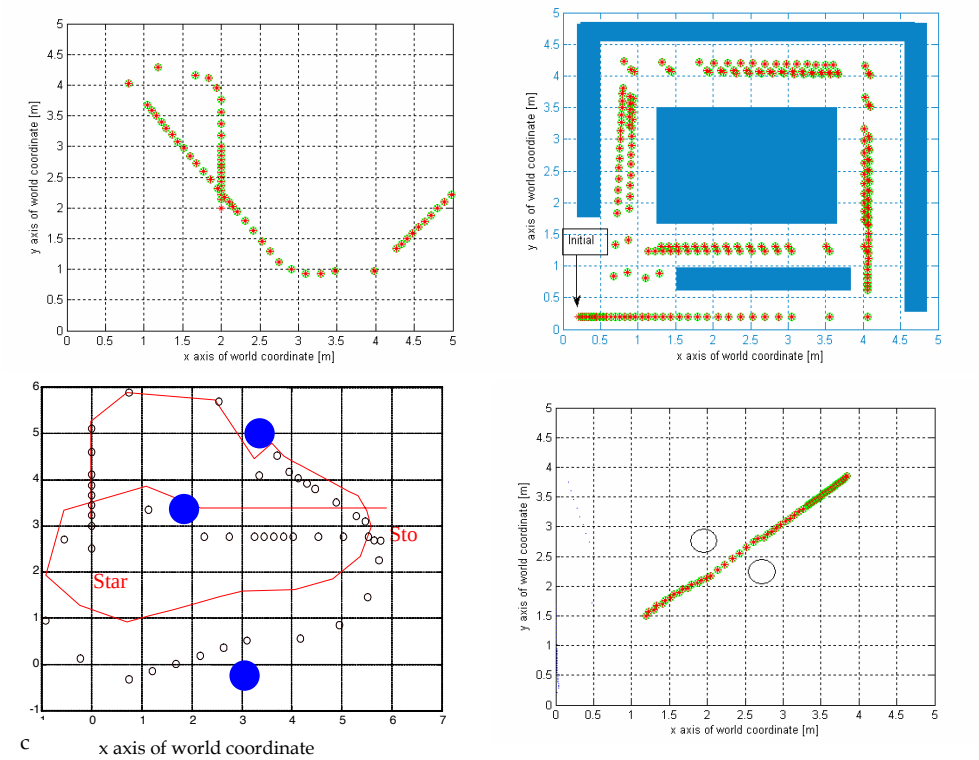

Fig. 15. Robot collision avoidance.
(a: Easy movement with no collision),
(b: Posture seeking, even orientation change)
(c: Easy movement),
(d: Posture seeking)

8.3 Robot Collision Avoidance

The above neuro-fuzzy controller has been divided in two parts: the obstacle avoidance routine and a goal seeking routine. Obstacle avoidance routine has been tested by simulation and by experiment. The goal seeking simulation was shown in Fig. 14., where the robot starts from known position and orientation in the world coordinate and stopped at the location of the desired goal position. The control algorithm enables the robot to start from a known position and stop at the specified goal position. If the robot encounters any obstacle in its way, the obstacle avoidance controller will take control of the mobile robot until the path of the robot is clear from any obstacle within the seeing range of the robot. This is exposed in Fig. 15. After the path is clear of any obstacle the goal seeking controller will be active again. The designed neuro-fuzzy controller was capable of controlling the mobile robot system with smooth translation from goal seeking to obstacle avoidance routines. Obstacle avoidance routine is much complicated due to needed decisions to turn left, turn right, move slowly, even to move backward. Goal seeking is simpler in tern of implementation. This due to the need to minimize the distance between an initial and final robot posture.

9. Conclusions

In this chapter a neuro-fuzzy -based embedded controller has been successfully used to drive a mobile robot run on a Transputer system. The Transputer ability, as an embedded controller, to perform a demanding parallel fuzzy controller, has been demonstrated in the real-time. Experimental testing of the mobile robot for obstacle avoidance behavior has been reported. Sensor fusion (of ultrasonic data) in the mobile robot navigation have been accomplished successfully. The neuro-fuzzy controller and how it was accomplished have been discussed in details. Two results of behaviors have been demonstrated. Obstacle avoidance, and posture seeking. Results have shown that; a Transputer computation system was a very suited environment to achieve an implementation of a neuro-fuzzy system for both obstacle avoidance and posture seeking. That was due to the parallel nature of computation. Sensory data were processed simultaneously. This rather gives the ability to have a quick decision regarding the mobile robot behavior.

10. References

Welgarz E., (1994), A General Robot Controller Using Transputer Based Hardware-Software System, *European Robotics and Intelligent Systems Conference* (EURISCON-94), August 22-26, Malaga, Spain, pp. 1055-1064.

Hu H.; Probert P. J.; & Rao B. Y. S. (1989), A Transputer Architecture for Sensor-Based Autonomous Mobile Robots. *Proceedings of the IEEE Int. Workshop on Intelligent Robots and Systems*.

Iida S. & Yuta, S. (1991), Vehicle Command System and Trajectory Control forAutonomous Mobile Robots. *In Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, Osaka, Japan, IROS'91, pp. 212–217.

AL-Gallaf E. (2006), A Mobile Robotics : An Intelligent Testbet Control System, *AnInternal Technical Report*, College of Engineering, University of Bahrain.

Brady M.; Hu M. H.; & Probert P. (1993), Transputer Architecture for Sensor-guided Control of Mobile Robots, *Proceedings of World Transputer Congress'93*, Aachen, September 1993, Germany, pp. 118-133.

Rusu P.; Petriu E.; Whalen T.; Cornell A.; & Spoelder H., (2003), Behavior-Based Neuro-Fuzzy Controller for Mobile Robot Navigation, *IEEE Transactions On Instrumentations and Measurement*, Vol. 52, No. 4.

Benmounah, A. (1991), Transputer Control of an AGV, Design, Construction, and Testing of a Mobile Platform, *Ph.D. Thesis*, University of Reading.

Maamri, M. (1991). Control of Mobile Platform in a Visually Monitored Environment, *Ph.D . Thesis*, University of Reading.

Meystel, A. (1991). Autonomous Mobile Robots, Vehicles with Cognitive Control, *World Scientific Publishing Company* Pte. Ltd.

Kim C. Ng & Trivedi Mohan M. (1998), A Neuro-Fuzzy Controller for Mobile Robot Navigation and Multi-Robot Convoying, IEEE Transactions On Systems, Man, And Cybernetics—PART B: Cybernetics, Vol. 28, No. 6, pp. 829-840.

Willgoss; & Iqbal R. J. (1999) Neurofuzzy Learning of Mobile Robot Behaviours: *Advanced Topics in Artificial Intelligence*. 12th Australian Joint Conference on Artificial Intelligence, AI'99. *Proceedings Lecture Notes in Artificial Intelligence*, Vol. 1747, pp. 278-90.

Pennacchio, S.; Abissi, F. ; Petralia, S. & Stendardo G. (2005), New Intelligent Controller for Mobile Robot Navigation in Unknown Environments: *WSEAS Transactions on Systems*, Vol. 4, No. 4, pp. 285-288.

Tsoukalas H.; Houstis, N., & Jones, V. (1997), Neurofuzzy Motion Planners for Intelligent Robots, *Journal of Intelligent and Robotic Systems: Theory and Applications*, Vol. 19, No. 3, pp. 339-56.

Hegazy, F., Fahmy, A., and El Refaie, M. (2004), An Intelligent Robot Navigation System Based on Neuro-fuzzy Control: Conference: *PRICAI 2004: Trends in Artificial Intelligence. 8th Pacific Rim International Conference on Artificial Intelligence*, Auckland, New Zealand, Proceedings, 9-13.

Shgiley J. E. (1969), Kinematics Analysis of Mechanisms, *McGraw-Hill, New York*, 1969. Muir P. F. & Neuman C. P. (1987), Kinematics Modeling of Wheeled Mobile Robots, *Journal of Robotics Systems* , Vol. 4, No. 2.

Reister D.B. and Pin F.G. (1994), Time-Optimal Trajectories for Mobile Robots With Two Independently Driven Wheels, *The International Journal of Robotics Research*, Vol. 13, No. 1, pp. 38-54.

Sarkar N.; Yun X. & Kumar V. (1994), Control of Mechanical Systems With Rolling Constraints: Application to Dynamic Control of Mobile Robots, *The International Journal of Robotics Research*, Vol. 13, No. 1, pp. 55-69.

McKerrow J., (1991), Introduction to Robotics, *Addison-Wesley Publishing Company*.

Pedrycz W. (1995), Fuzzy Control and Fuzzy Systems, *John Wiley & Sons International*.

Beom R. & Cho H. (1995), A Sensor-Based Navigation for a Mobile Robot Using Fuzzy Logic and Reinforcement Learning, *IEEE Transaction on Systems, Man, and Cybernetics*, Vol. 25, No. 3.

Baxter J. W. & Bumby J. R. (1995), Fuzzy Control of a Mobile Robotic Vehicle, *Proceeding of Institute of Mechanical Engineering*, IMechE.

Klir G.J. & Yuan B. (1995), Fuzzy Sets And Fuzzy Logic, Theory And Applications, *Prentice-Hall International*.

Klafter R. D.; Chmielewski T. A. & Negin M. (1989), Robotic Engineering an Integrated Approach, *Prentice-Hall International*.

Mir S. A.; Zinger D. S.; & Elbuluk M. E. (1994), Fuzzy Controller for Inverter Fed Induction Machines, *IEEE Transaction on Industry Applications*, Vol. 30, No. 1.

Mobile Robotics, Moving Intelligence

Edited by Jonas Buchli

ISBN 3-86611-284-X

Hard cover, 586 pages

Publisher Pro Literatur Verlag, Germany / ARS, Austria

Published online 01, December, 2006

Published in print edition December, 2006

This book covers many aspects of the exciting research in mobile robotics. It deals with different aspects of the control problem, especially also under uncertainty and faults. Mechanical design issues are discussed along with new sensor and actuator concepts. Games like soccer are a good example which comprise many of the aforementioned challenges in a single comprehensive and in the same time entertaining framework. Thus, the book comprises contributions dealing with aspects of the Robotcup competition. The reader will get a feel how the problems cover virtually all engineering disciplines ranging from theoretical research to very application specific work. In addition interesting problems for physics and mathematics arises out of such research. We hope this book will be an inspiring source of knowledge and ideas, stimulating further research in this exciting field. The promises and possible benefits of such efforts are manifold, they range from new transportation systems, intelligent cars to flexible assistants in factories and construction sites, over service robot which assist and support us in daily live, all the way to the possibility for efficient help for impaired and advances in prosthetics.