Employing a Recurrent Linear Model to Guide Mobile Robots Adaptively

Ashraf Aboshosha
Computer Science Dept. (WSI), University of Tübingen,
Sand 1, 72076 Tübingen, Germany.
shosha@informatik.uni-tuebingen.de
http://www-ra.informatik.uni-tuebingen.de

Abstract

Controller adaptation is always a major concern. A controller that meets certain performance design objectives can not be satisfactory unless it can preserve them in the presence of system parameters fluctuation over a wide range of operation conditions. For autonomous robots, a controller that preserves stability and robustness dynamically is especially desirable. This paper presents, therefore, a design of an adaptive controller using recurrent linear model (RLM) to guide a robot driven by multi-references. Moreover, it shows how the artificial neural networks (ANN) span from their classical pattern recognition to the adaptive behaviour learning. The main objective of that is to maintain the stability and to reinforce the robustness of the overall control system by which the transition among different set points will be smoothed. The RLM is systematic, methodical and easy to design.

Keywords: *Adaptive control, RLM, autonomous mobile systems (AMS), robot guidance, sensor integration.*[1]

INTRODUCTION

Although the ANN control architecture was proposed in the middle of the seventies, quite a lot open questions have been left even for nowadays. Among them, the most important ones are its modelling and adaptation capabilities. Throughout this framework we will focus on investigating these capabilities and on employing them to steer autonomous mobile robots adaptively.

The traditional theory of neural controllers (neurocontrollers) relies mainly on learning a network to serve as a controller by which the performance of the system under control is improved. To control autonomous mobile robots (AMS), the neurocontroller generates a reference signal which leads the system to reach a pre-defined set point. The important point of using neurocontrollers is the strategy used to design and teach the neurocontroller how it can generate the control signal. This procedure comprises two phases: the first one is the *learning phase* and the second is the *access phase*. Several

---

[1] This study has been implemented on the B21-RWI robot platforms

ANN topologies can be used to implement control theories, e.g: multi-layer perceptrons (MLP), radial basis function (RBF) networks, Cerebellar Model Articulated Controller (CMAC) and the RLM.

Among ANNs topologies, MLPs are the most popular ones. Due to their universal approximation capability, they are widely used in various linear and non-linear applications. The most serious drawback of MLPs and MLP-based dynamic networks is their slow training procedure, which is considered as cumbersome to apply in real-time. The algorithm used for MLPs learning is called error back propagation training algorithm (BP). The basic idea of this algorithm is to learn an approximation of the system characteristics and then to use it to generate an appropriate control signal. The approximation of the system characteristics is understood as a gradual learning, which depends on the observation of the system input-output data in real-time, during different operating conditions.

The CMAC was first developed and described in a series of inspiring papers in the 1970's by J. Albus [5]. After that scientists worldwide developed CMAC-based controller combined with several algorithms such as fuzzy systems (Mamdani 1975, Takagi-Sugeno (TS) 1985, Tsukamoto, B-spline and adaptive network-based fuzzy inference system (ANFIS)), genetic algorithms GA, support vector machines (SVM) [14], adaptive Kalman filters, Bayesian rule and Karhunen-Loeve transforms. The use of CMAC in robotic control was further developed in a series of seminal papers on real-time control of robotic arms [11]. Essentially, a CMAC architecture maps the input space of the problem into a much larger virtual address space via what can be called coarse encoding. The number of entries in the virtual address space is usually quite large, clearly far too large to be used for direct storage of tunable parameters. In practice, this large virtual address space is drastically reduced in size by hashing the virtual address to a smaller working address. CMAC has many excellent properties, which make it a real alternative to MLPs. Perhaps the most important feature is its exceptional learning characteristics: the speed of learning may be much higher than that of a corresponding MLP using the BP algorithm. A further important feature is that the classical binary CMAC can be implemented without using multipliers, so it is especially suited for digital hardware implementation. Moreover, CMAC controllers have an ideal network architecture for embedded applications where relatively high operating speed, small size and reduced implementation cost are the most important requirements.

The main reason to avoid using CMAC controllers is the deficit of modelling and adaptation capabilities. Therefore, the main goals of our proposal is to develop a controller which owns a behaviour modelling and adaptation capabilities. Throughout this framework, a RLM has been employed to interpolate adaptive control techniques. Traditionally, Adaptive control techniques can be implemented in 4 standard phases, they are: **1)** modelling, **2)** identification, **3)** controller design and **4)** optimization. Because of the linearity of its activation function, the RLM structure is suitable for adaptive systems implementation. The RLM is associated conventionally with the least mean square (LMS) learning rule, called Widrow-Hoff learning rule. This paper demonstrates how the RLM can learn in on-line mode using the matrix inversion lemma (MIL). Throughout this study the RLM has been used to build a model of the robot dynamics and to design an adaptive controller that drives the robot using a multi-reference signal.

The remainder of this paper is organized as follows. Section (1) focuses on modelling of the robot dynamics using the RLM and discusses the MIL learning rule compared with LMS and maximum likelihood (ML) learning approaches. Section (2) presents the design of RLM adaptive controller used to guide the robot. Finally, section (3) introduces the conclusion and a comment on the presented results.

1 Modeling of Robot Dynamics

1.1 RLM paradigm

The RLM paradigm of system dynamics is derived from the Kalman filter [2, 1], see figure (1). This structure takes into account both the observed state y_k and the reference control signal u_k which is given by:

$$y_k = \sum_{i=1}^{na} a_i y_{k-i} + \sum_{i=1}^{nb} b_i u_{k-i} + \epsilon_k \qquad (1)$$

Where ϵ_k is a modelling residual, Gaussian white noise, n_a is the model order of the observed state (also called the number of poles), while n_b is the model order of the control signal (also called the number of zeros). The operator q^{-1} is the back shift operator or delay, where $q^{-1}\, y_k = y_{k-1}$, that follows:

$$\begin{aligned} A(q^{-1})y_k &= B(q^{-1})u_{k-1} + \epsilon_k \\ A(q^{-1}) &= 1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a} \\ B(q^{-1}) &= b_1 q^{-1} + \ldots + n_b q^{-n_b} \end{aligned} \qquad (2)$$

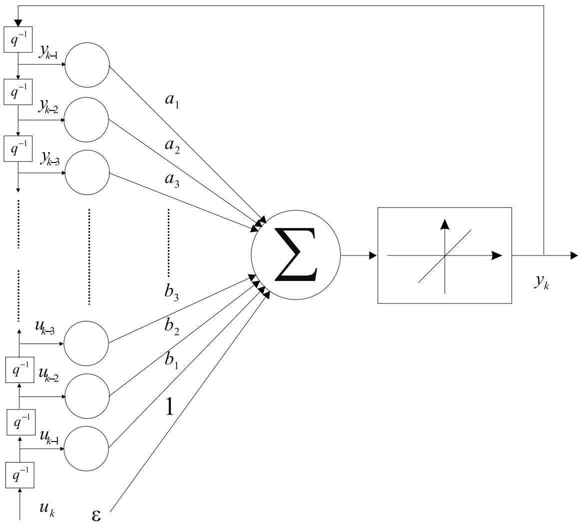


Figure 1. The structure of RLM network

In this case the observed state y_k is the robot's longitudinal velocity, heading and rotational velocity. The reference control signal u_k is the guide's range, acquired from distributed robot sensors (vision system, laser scanner and sonar). The Gaussian distributed noise, associated with the observed output, enables applying weights (coefficients) identification algorithms such as maximum likelihood ML, least mean square LMS and recursive least squares (RLS). The RLM paradigm is applicable within linear and quasi-linear systems. Therefore, we applied it to underlie the speed control, while it failed to cope with position control due to the presence of large non-linear odometric errors [4, 1].[2]

1.2 MIL based Learning of RLM Paradigm

In general, the weights estimation problem of RLM paradigm can be treated as a linear regression problem. The linear regression is the simplest type of parametric models. Its origin can be traced back to Gauss (1809), who used such a technique, equation (3), for calculating orbits of the planets.

Now let us explain, how to teach the RLM to deduce its weights Θ. The MIL is a stepwise learning algorithm, this means that the estimation of the RLM's weights yields a gradual convergence. Compared with LMS and ML algorithms, the MIL needs a lower computation and it is reliable to be applied on-line.

For weights estimation purposes, it is convenient to write the RLM paradigm in a form in which emphasizes the weights vector to be estimated and the data available. This is achieved by using the backward shift interpolation of (q^{-1}) to cast the RLM model in the form:

$$y_k = \Phi_k^T \Theta + \epsilon_k \qquad (3)$$

---

[2] The detailed derivation of the LMS and ML algorithms can be reviewed in [3]

Where Θ is the vector of unknown weights, defined by:

$$\widehat{\Theta}_k^T = \left[-\widehat{a}_1,\ \dots\ ,-\widehat{a}_{n_a},\ \widehat{b}_1,\ \dots\ ,\ \widehat{b}_{n_b}\right] = \left[\widehat{\theta}_1,\ \widehat{\theta}_2,\ \dots\ ,\ \widehat{\theta}_{n_a+n_b}\right] \tag{4}$$

and Φ_k is a regression vector partly consisting of measured input/ouput variables and defined by:

$$\Phi_k^T = [y_{k-1},\ \dots\ ,\ y_{k-n_a},\ u_{k-1},\ \dots\ ,\ u_{k-n_b}] = [\phi_1,\ \phi_2,\ ,\ \dots\ ,\ \phi_{n_a+n_b}] \tag{5}$$

LMS: The least squares estimator for the weights vector $\widehat{\Theta}$ is directly calculated by:

Batchwise:

$$\widehat{\Theta} = [\Phi^T\Phi]^{-1}[\Phi^T y] \tag{6}$$

The LMS algorithm depends on the direct matrix inversion, so applying this technique to some models seems critical.

Stepwise: The Widrow-Hoff learning rule (1967) is widely used for static pattern classification purposes of RLM nets. It depends mainly on a stepwise change of weights to learn the model. This rule has a limited precision and has a limited processing range. If y_i is the desired output and α is the learning speed constant then the adjustment of the weight $\theta_{i,j}$ linking input i with output j is accessed as follows:

$$\Delta\theta_{i,j} = \alpha(y_i - \Phi^T\Theta)\phi_j,\ \ for\ \ j = 1, 2, \dots, n \tag{7}$$

ML: The principle of maximum likelihood can be shown to give asymptotically (when $N \rightarrow \infty$, where N is the number of weights) unbiased and efficient (minimum variance) estimates. Let us consider the system described by equation (3). Then assume that ϵ is normally distributed and has N elements, furthermore its mean $E[\epsilon] = 0$ and its covariance $E\left[\epsilon\epsilon^T\right] = C_\epsilon$ is known. It is equally valid to use the probability density function of the noise since there is a one-to-one transformation between Y and ϵ. The log-likelihood function is expressed as:

$$\log L(\widehat{\Theta}) = -\frac{1}{2}\log\left[(2\pi)^N \det(C_\epsilon)\right] - \frac{1}{2}\left[Y - \Phi^T\widehat{\Theta}\right]^T C_\epsilon^{-1}\left[Y - \Phi^T\widehat{\Theta}\right] \tag{8}$$

At the maximum of the log-likelihood function we obtain the estimate $\widehat{\Theta} = \Theta$. In the case when the noise is a normally distributed white noise with a covariance matrix $C_\epsilon = \sigma^2 I$, it is easy to show that the maximization of the likelihood function equation (8) is equivalent to the minimization of the loss function equation (9)

$$V(\widehat{\Theta}) = \frac{1}{2}\left[y - \Theta^T\widehat{\Theta}\right]^T\left[y - \Phi^T\widehat{\Theta}\right] = -\sigma^2\log L(\widehat{\Theta}) + \text{constant} \tag{9}$$

MIL: Before going to introduce the MIL algorithm let us explore some common properties of LMS and ML techniques.

1. The LMS and ML estimate of weights approaches the optimum Wiener solution as the data length N approaches infinity, if the input and the desired response are jointly stationary ergodic processes.

2. The LMS estimate of the weights vector is unbiased if the error signal ϵ_k has zero mean for all k.

3. The computation of both LMS and ML algorithms is relatively high and that is cumbersome especially if we apply them to real-time processes.

4. Both LMS and ML are slow learning algorithms and that is not suitable for adaptive systems.

Let A and B be two positive definite, m by m matrices related by

$$A = B^{-1} + CD^{-1}C^T \tag{10}$$

where D is another positive definite, n by n matrix and C is an m by n matrix. According to the matrix inversion lemma, we may express the inverse of the matrix A as follows:

$$A^{-1} = B - BC\left[D + C^T BC\right]^{-1} C^T B \tag{11}$$

In the special case of $n = 1$ and $D = \lambda$, for simplicity let $\lambda = 1$, we get:

$$A^{-1} = B\left[I_m - \frac{CC^T B}{\lambda + C^T BC}\right] \tag{12}$$

The MIL is widely used as an estimator in different machine learning algorithms such as; Kalman filter, ANFIS fuzzy, recursive least squares (RLS) etc. We will employ this rule to estimate the weights of RLM network which serves as a model of mobile robot dynamics. The estimation process can be accomplished according to the following.

At time step k:

1. Form Φ_k using the new data (input-output patterns)

$$\Phi_k^T = [y_{k-1},\ \ldots\ ,\ y_{k-n_a},\ u_{k-1},\ \ldots\ ,\ u_{k-n_b}] = [\phi_1,\ \phi_2,\ \ldots\ ,\ \phi_{n_a+n_b}] \tag{13}$$

2. Calculate the estimation error ϵ_k using

$$\epsilon_k = y_k - \Phi_k^T \widehat{\Theta}_{k-1} \tag{14}$$

3. Apply the MIL rule as in equation (12) to calculate the covariance matrix Ψ_k, sometimes called Kalman gain based update matrix. So if $A^{-1} = \Psi_k$, $B = \Psi_{k-1}$, $D = \lambda = 1$ and $C = \Phi_k$, then the MIL will be

$$\Psi_k = \Psi_{k-1}\left[I_m - \frac{\Phi_k \Phi_k^T \Psi_{k-1}}{(1 + \Phi_k^T \Psi_{k-1} \Phi_k)}\right] \tag{15}$$

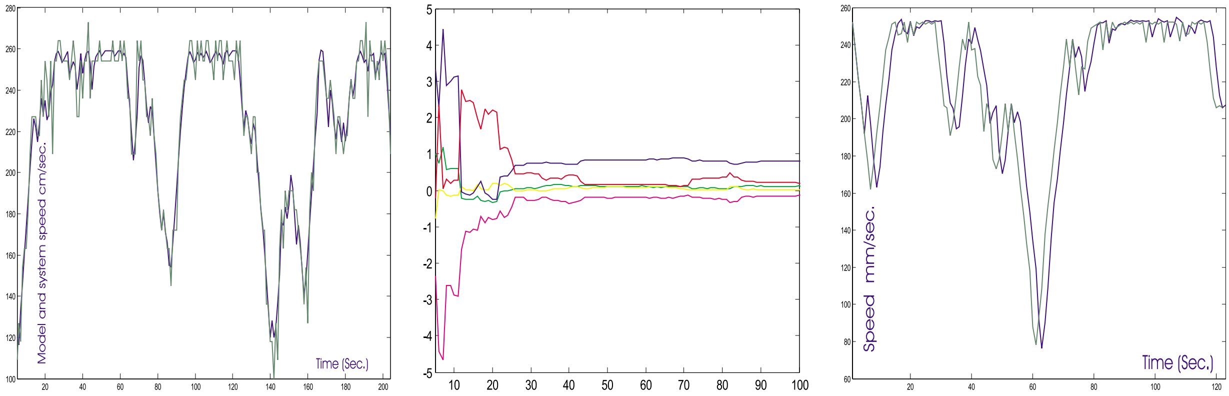
4. Update weights

$$\widehat{\Theta}_k = \widehat{\Theta}_{k-1} + \Psi_k \Phi_k \epsilon_{k-1} \tag{16}$$

where,

$$\widehat{\Theta}_k^T = \left[-\widehat{a}_1,\ \ldots\ ,-\widehat{a}_{n_a},\ \widehat{b}_1,\ \ldots\ ,\ \widehat{b}_{n_b}\right] = \left[\widehat{\theta}_1,\ \widehat{\theta}_2,\ \ldots\ ,\ \widehat{\theta}_{n_a+n_b}\right] \tag{17}$$

5. Wait for the next time step to elapse and loop back to step (1)



(a) RLM and real system outputs (b) RLM weights identification (c) RLM response in prediction mode

Figure 2. RLM weights and response in prediction mode

MIL parameters have been initialized using an empty vector. Figure (2.a) shows the output of RLM compared with the actual output, while figure (2.b) shows the convergence of RLM weights during learning. The output of the RLM seems smooth due to filtering of high frequencies. RLM Models have the capability to model mobile robot dynamics and predict its behaviour if the input (i.e. the obstacles histogram or the visual ranging entries) is known, see figure (2.a and 2.c).

2 Adaptive Neurocontroller

2.1 Sensor Integration

To resemble the human behaviour in control and adaptation, we have to close the loop between robot sensors and robot actuators. The main goal of that is to steer the robot adaptively, while chasing a dynamic object or to navigate safely in unknown terrain without collision, based on multi-sensor integration. Figure (3) shows a scheme of sensor integration system. The binocular vision system uses the projection of a coloured object on two CCD cameras to deduce its horizontal components of the centre of gravity x_1 and x_2 with the number of projection pixels N_1 and N_2. The calibration of the geometrical vision system is explained in details in [2]. The obstacles histogram, acquired from the laser scanner, is used to regulate the robot velocity, heading and rotational velocity. The multiplexing unit is a fuzzy membership function which interpolates the control signal from references (visual and laser ranging modes).

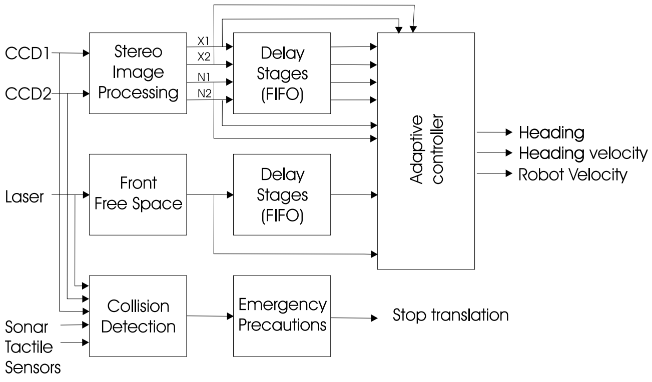
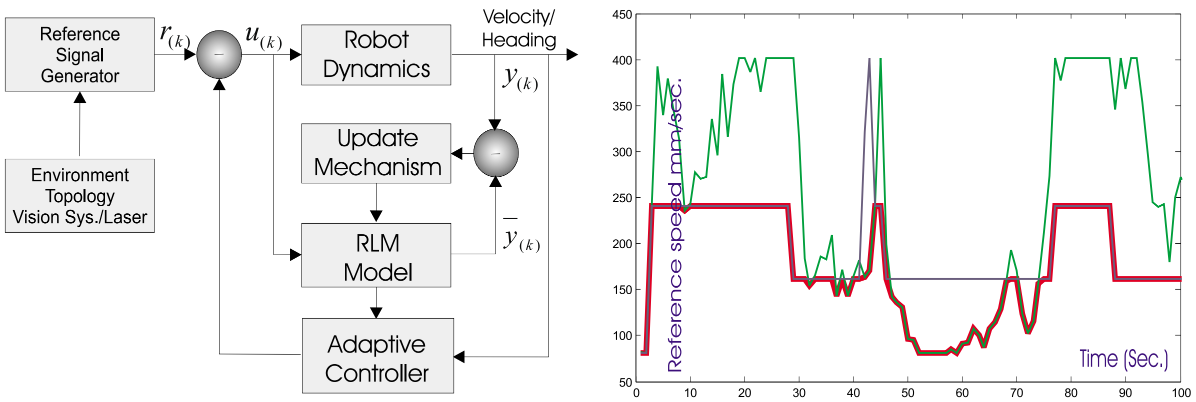


Figure 3. The structure of multi-sensor integration for adaptive steering of mobile robots

The strategy is to build an integrated system, which enables the robot to pursue a dynamic object using a geometrical vision system. Moreover, other distributed sensors collaborate to support the robustness of the overall system. The control of robots navigation (heading and translation) relies primarily upon outputs of both the laser range finder and the binocular vision system. In emergencies, all associated distributed sensors such as vision, laser, sonar and tactile sensors collaborate to hinder collisions, see figure (3).

2.2 Controller Design

The structure of RLM nets is a reliable technique to implement adaptive control systems. Figure (4.a) shows the structure of adaptive control system using RLM nets, while (4.b) shows the membership function used to generate a unique control signal interpolated from two references (binocular vision ranging and laser ranging) [1, 2].



(a) RLM adaptive controller (b) Membership function

Figure 4. RLM Control of autonomous mobile robots

To achieve the adaptability of the system under study, the weights of the RLM controller can be calculated according to the following rules.

$$\frac{y_k}{r_k} = q^{-d}.\frac{B(q^{-1}).H}{T(q^{-1})} \quad where \quad H = \lim_{q \to 1} \frac{T(q^{-1})}{B(q^{-1})}, \quad and \quad d = 1 \tag{18}$$

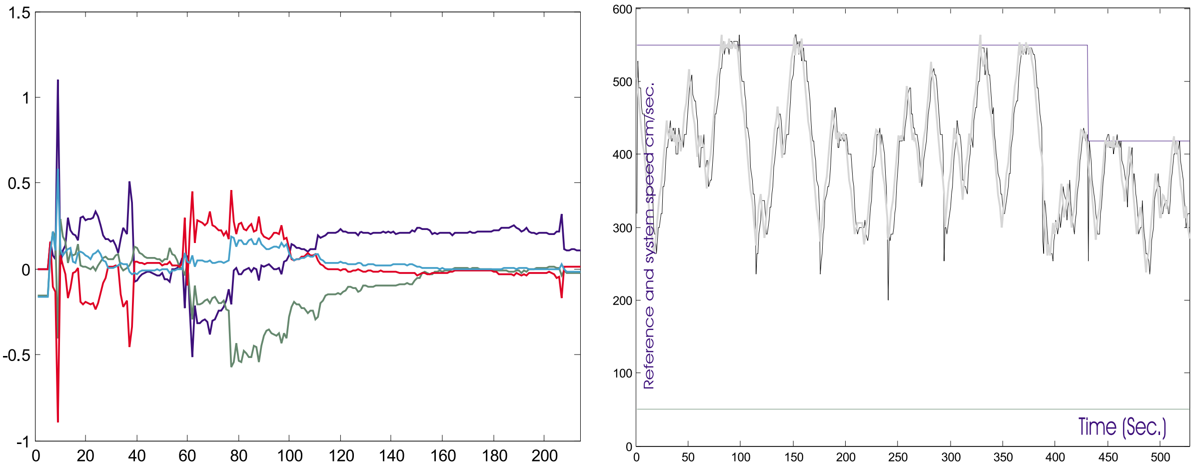
$$F(q^{-1})A(q^{-1}) + q^{-1}B(q^{-1})G(q^{-1}) = T(q^{-1}) \tag{19}$$

$$\begin{aligned} T =& \quad 1 + t_1 q^{-1} + \ldots + t_{n_t} q^{-n_t} \\ G =& \quad g_0 + g_1 q^{-1} + \ldots + g_{n_g} q^{-n_g} \\ F =& \quad 1 + f_1 q^{-1} + \ldots + f_{n_f} q^{-n_f} \end{aligned} \tag{20}$$

Where $n_f = n_b - 1, \;\; n_g = n_a - 1, \;\; and \;\; n_t \leq n_a + n_b - 1$. The union (AND) fuzzy logic operation, equation (21), has been applied to generate the control signal r_k as an interpolation between visual and laser reference signals. This operation selects the minimum value by which the system attains more safety, see figure (4.b).

$$r_k = \mu_{v \cap l} = min(\mu_v, \;\; \mu_l) \tag{21}$$

The controller comprises three parts; the first one is the compensator (H), feedback filter G and the preconditioning unit $1/F(q^{-1})$. The design of RLM controller is shown in figure (6). $T(q^{-1})$ is the selected poles function, the main job of this function is to preserve the overall stability of the system. The compensator H corrects the offset error of the output while $G(q^{-1})$ and $F(q^{-1})$ components are the controller poles and zeros that adapt the performance. The controller parameters F and G can be



(a) Controller parameters (b) Output w.r.t. reference signal

Figure 5. Controller parameters and system response

calculated from equation (19). If we consider $n_a = 2,\ n_b = 3$ then $n_f = 2$ and $n_g = 1$. Multiplying out and equating coefficients of q^{-i} for $i = 1, 2, \ldots, n$ leads to the following equation set:

$$\begin{bmatrix} 1 & 0 & b_1 & 0 \\ a_1 & 1 & b_2 & b_1 \\ a_2 & a_1 & b_3 & b_2 \\ 0 & a_2 & 0 & b_3 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ g_0 \\ g_1 \end{bmatrix} = \begin{bmatrix} t_1 - a_1 \\ -a_2 \\ 0 \\ 0 \end{bmatrix} \tag{22}$$

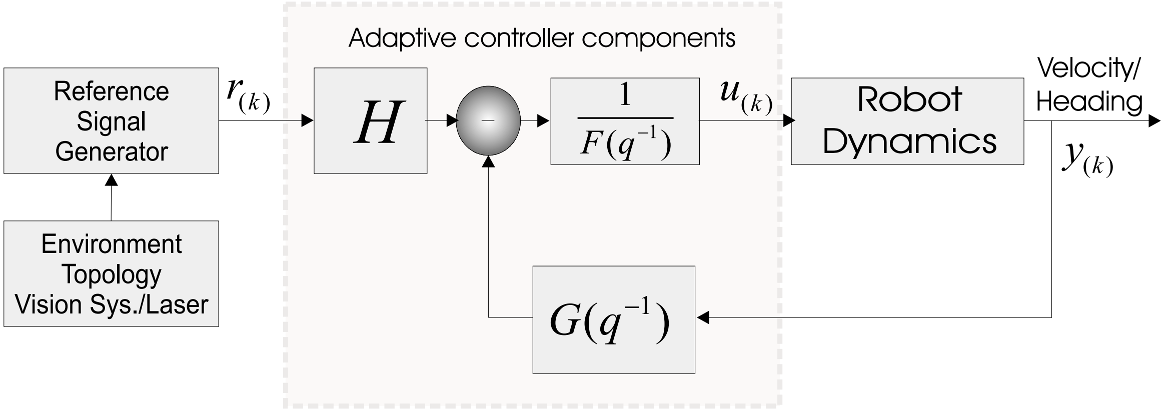
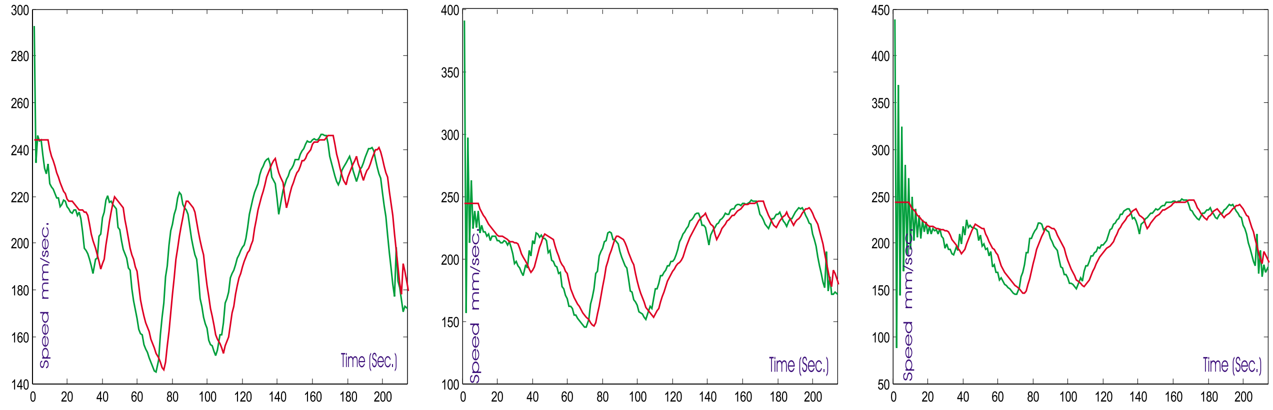


Figure 6. The structure of RLM adaptive controller

The estimated controller parameters f_1, f_2, g_0 and g_1 are shown in figure(5.a). The selection of the closed loop poles is the most important factor in quality measure of control rules. As the assigned poles tend to reach the zero of Z-domain or -∞ of Laplace S-domain, the cross correlation [1, 2] between the reference signal (Laser histogram, visual ranging or sonar ranging) and the response reaches unity. It is worth mentioning that it is not wise to consume the system resources and fuel to attain the high precision. The cost function of a control system plays an important role in definition of task priorities based on multi-criterian optimization. Figure (7) shows the system response under different assigned poles.



(a) Response at t_1=0.1 (b) Response at t_1=0.3 (c) Response at t_1=0.4

Figure 7. System response under different assigned poles (0.1, 0.3, 0.5)

3 Conclusion

The major contribution of this article is the formulation of a new learning approach of RLM nets, MIL learning rule, instead of LMS, the previously dominant learning rule of this type of networks. Furthermore, the use of this rule provided AMS with more flexibility and adaptability by which the ANNs spanned from their traditional static classification domain to the dynamically behaviour learning approach. By manipulating the manner in which feature information of both laser and visual data is incorporated into the model, it can be shown that significant improvements in the performance of the algorithm can be attained. Moreover, the simplicity and the efficiency of adaptive control systems in dynamic tracking techniques succeeded to reinforce the robustness of the overall system.

Acknowledgement

The author would like to acknowledge the financial support by the German Academic Exchange Service (DAAD) of his PhD scholarship at the University of Tübingen.

References

[1] A. Aboshosha. Adaptation of rescue robot behaviour in unknown terrain based on stochastic and fuzzy logic approaches. In *proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2003*, Las Vegas USA, October 27-31 2003.

[2] A. Aboshosha. Adaptive visual guidance of mobile robots based on takagi-sugeno model. In *proceedings of the Applied Machine Vision - From Image Processing to Vision Systems conference, GMA - VDI*, Stuttgart, Germany, October 21-22 2003.

[3] A. Aboshosha. An introduction to robot distributed sensors. Technical Report WSI-2003-13, Tbingen, Germany, 2003.

[4] A. Aboshosha and A. Zell. Robust mapping and path planning for indoor robots based on sensor integration of sonar and a 2d laser range finder. In *INES 2003*, Assiut-Luxor, Egypt, March 2003.

[5] J. S. Albus. A new approach to manipulator control: The cerebellar model articulation controller. *Transactions of the ASME*, 97(3):220–227, 1975.

[6] M. Brown, C. J. Harris, and P. Parks. The interpolation capability of the binary cmac. *Neural Networks*, 6:429–440, 1993.

[7] D. Burschka and G. D. Hager. Principles and practice of real time tracking on consumer hardware . In *Tututial 1 at IEEE VR2003: Recent Methods for Image-Based Modeling and Rendering*, pages 55–66, March 2003.

[8] J. T. Economon, A. Tsourdos, and B. A. White. Takagi-sugeno model synthesis of a quasi-linear parameter varying mobile robot. In *IROS*, Lausanne, Switzerland, 2002. EPFL.

[9] D. Ellison. On the convergence of the multidimensional albus perceptron. *The International Journal of Robotics Research*, 10:338–357, 1991.

[10] S. Feyrer and A. Zell. Tracking and pursuing persons with a mobile robot. In *International Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS)*, Corfu, Greece, 1999.

[11] T. W. M. F. H. Glanz and L. G. Kraft. Cmac: An associative neural network alternative to backpropagation. *Proceedings of the IEEE*, 78:1561–1567, 1990.

[12] D. Kukolj and E. Levi. Identification of complex systems based on neural and takagi-sugeno fuzzy model. *IEEE Transaction on Systems Man and cybernetics*, 34(1), February 2004.

[13] A. Mojaev and A. Zell. Tracking control and adaptive local navigation for nonholonomic mobile robots. In *The 8th Conference on Intelligent Autonomous System*, Amsterdam, The Netherlands, 10-13 March 2004.

[14] J. A. K. Suykens, L. Lukas, and J. Vandewalle. Sparse approximation using least squares support vector machines. *Proc. of the IEEE International Symposium on Circuits and Systems ISCAS*, II:757–760, 2000.

[15] O. P. Vlad and T. Fukuda. Design method for neuro-fuzzy motion controllers. In *Proc. of the IEEE International Conference on Industrial Technology ICIT*. IEEE, 2002.

[16] A. Zell, N. Mache, T. Sommer, and et. al. Stuttgart neural network simulator, users manual, version 3.2. Technical report, University of Stuttgart, Germany, June 1994.