Slovak University of Technology in Bratislava
Institute of Information Engineering, Automation, and Mathematics

PROCEEDINGS

17th International Conference on Process Control 2009

Hotel Baník, Štrbské Pleso, Slovakia, June 9 – 12, 2009

ISBN 978-80-227-3081-5

http://www.kirp.chtf.stuba.sk/pc09

Editors: M. Fikar and M. Kvasnica

Masar, I., Gerke, M.: A Neuro-Fuzzy Controller for a Trajectory Following Mobile Robot, Editors: Fikar, M., Kvasnica, M., In *Proceedings of the 17th International Conference on Process Control '09*, Štrbské Pleso, Slovakia, 647–653, 2009.

Full paper online: http://www.kirp.chtf.stuba.sk/pc09/data/abstracts/112.html

A NEURO-FUZZY CONTROLLER FOR A TRAJECTORY FOLLOWING MOBILE ROBOT

I. Masar * and **M. Gerke** *

* *Control Systems Engineering Group, Department of Electrical Engineering, FernUniversitt in Hagen, D-58084 Hagen, Germany fax : +49 2331 987 354 and e-mail : ivan.masar@fernuni-hagen.de*

Abstract: The design of a motion controller for a mobile robot can be a very difficult and tedious task, especially for robots with a complex kinematic structure. Even though several types of motion controllers have been proposed in literature, they are not always applicable on car-like mobile robots equipped with conventional steering wheels. The reason is, that it is generally not possible to derive an inverse kinematic model for such robots. In this paper, a self-tuning intelligent controller for a quasi-omnidirectional mobile robot is presented. The controller is used to control the robot following a desired trajectory. It is implemented as a neuro-fuzzy controller, which can adapt its parameters by a self-learning process in such manner, that the mobile robot can follow a desired trajectory with required accuracy and speed. The process of tuning controller parameters tuning is demonstrated on experiments with a quasi-omnidirectional mobile robot F.A.A.K.

Keywords: mobile robot control, trajectory following, neuro-fuzzy controller, hybrid learning

1. INTRODUCTION

A precise motion control is a primary assumption for succesful application of every mobile robot. In the past years, various control design methods have been developed for this purpose. The situation is relatively easy for mobile robots with simple kinematic structure (e.g. with two differential driven wheels), or for robots without nonholonomic constraints, respectively. If their inverse kinematic model exits, it can be used to calculate the position and speed of every joint in the kinematic chain from the speed of the robot body. Therefore, a motion of the robot along a trajectory can be computed offline as a sequence of motions of robot joints and links. In an ideal case, after applying of these partial motions, the robot would move as required. However, there are external forces acting on the robot during its movement (driving, centrifugal, friction, Coriolis and gravitational forces), that must be compensated to follow the desired trajectory with high precision. For this purpose, the kinematic or dynamic model of the robot is used depending on requested precision of trajectory following. For simple applications, the kinematic model is usually preferred. The dynamic model of the robot is the base for a high precision motion control, but it is considerably more complicated than kinematic model. Various model-based control systems for mobile robots can be found in de Wit et al. (1996), Muir (1988).

However, the problem with implementation of a model-based control arises if the inverse kinematic model of the robot does not exist. Such situation occurs by every car-like mobile robot, i.e. if there are more wheels on a common axis or at least one steering wheel with the steering axis going through the contact point of the wheel with the

floor. An example of a conventional steering wheel is depicted in Fig. 1.

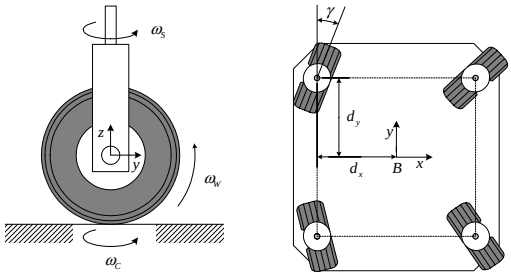


Fig. 1. Conventional steering wheel

Motion of the robot body consists of two translation (in X- and Y-axis) and one rotation (about Z-axis). It can be expressed in the robot *body* coordinate system B as

$$\dot{\mathbf{p}}_B = \begin{bmatrix} v_{B_x} \\ v_{B_y} \\ \omega_{B_z} \end{bmatrix} = \mathbf{J}.\mathbf{q} = \mathbf{J} \begin{bmatrix} \omega_W \\ \omega_C \\ \omega_S \end{bmatrix}, \qquad (1)$$

whereby ω_W is a wheel rotational speed about X-axis (driving speed of the wheel), ω_C is a rotational speed of the wheel about Z-axis in the contact point with the floor and ω_S is a rotational speed about the steering axis. The matrix $\mathbf{J}$ is called a wheel Jacobian and in the case, that the steering axis of the steered wheel goes through the contact point with the floor, it is defined as follows

$$\mathbf{J} = \begin{bmatrix} -R\sin(\gamma) & d_y & -d_y \\ -R\cos(\gamma) & -d_x & d_x \\ 0 & 1 & -1 \end{bmatrix}, \qquad (2)$$

whereby R is a wheel radius and d_x, d_y are distances of the wheel steering axis from the robot midpoint.

It is obvious, that the matrix $\mathbf{J}$ is singular and therefore it is not possible to directly derive the inverse kinematic model of the robot. A conventional steering wheel has less degrees of freedom than input variables and therefore it is redundant. Hence, the model-based control using the inverse kinematic model is not applicable on the mobile robots equiped with conventional steering wheels.

For this reason, we have developed a neuro-fuzzy controller which can be used for trajectory following for car-like mobile robots with non-holonomic constraints. Before describing the controller functionality, a mobile robot used for verification of proposed solution will be briefly introduced.

2. QUASI-OMNIDIRECTIONAL MOBILE ROBOT F.A.A.K.

The quasi-omnidirectional mobile robot F.A.A.K. (Fig. 2) was developed on our department as an experimental platform for testing of advanced control and navigation strategies Masr and Gerke (2004). Even though it is a small-size robot, it is equipped with external sensorics, visual system and enough processing power to implement advanced control strategies to able to operate fully autonomous in various environments.

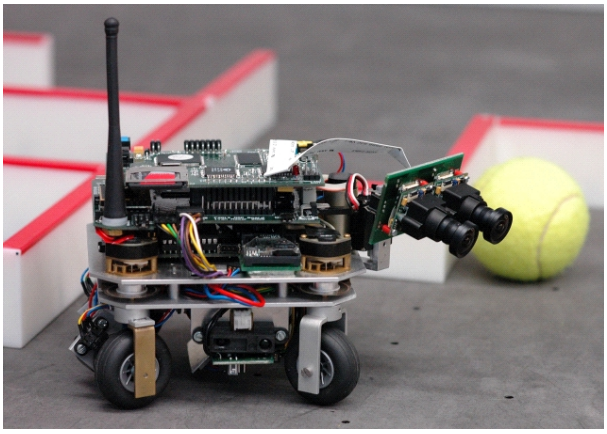


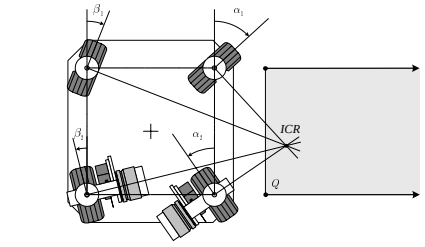
Fig. 2. Mobile robot F.A.A.K.

The robot is built on a non-conventional chassis with four steered wheels, whereas two of them are driven. Such kinematics allows performing of various types of motion, depending on performed task and actual situation in robot environment. Thereby it is possible to change not only the radius, but also the instantaneous centre of robot rotation ICR (Fig. 3(a), Fig. 3(b)) and thus to turn the curves with very small radius. Moreover, the robot can cruise (Fig. 3(c)), move laterally (Fig. 3(d)) or rotate about its vertical axis (Fig. 3(e)), respectively.

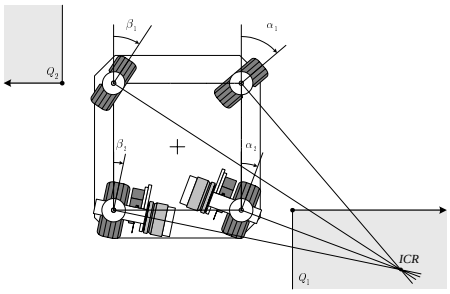
By using of these motion types, the robot can overcame the constraints caused by the non-holonomic couplings and thus it is able to move in any direction with an arbitrary orientation. Therefore, the robot can move quasi-omnidirectional. Such quasi-omnidirectional movement concept is very advantagenaous for robotic applications in artificial environments with many obstacles like households, offices, hospitals etc. Namely, thanks to posibility to re-configure its kinematic structure, the robot can manoeuvre on a very limited space.

3. TRAJECTORY FOLLOWING PROBLEM

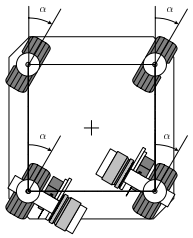
The problem of trajectory following by a mobile robot is showed in Fig. 4. In the figure, a simplified kinematic model of the F.A.A.K. robot is used. This simplification can be applied under assumption, that the steering wheels are controlled using Ackerman steering principle, i.e. that all perpendicular wheel axis intersect in an ICR at any time. If this condition is fulfilled, the steering



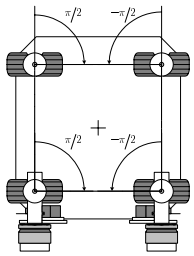
(a) Turning with $\mathrm{sgn}(\alpha_i) \neq \mathrm{sgn}(\beta_i)$



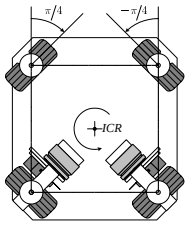
(b) Turning with $\mathrm{sgn}(\alpha_i) = \mathrm{sgn}(\beta_i)$



(c) Cruising



(d) Side ride



(e) Rotation on a spot

Fig. 3. Motion types of F.A.A.K. mobile robot

wheels on a common axle can be replaced by a virtual wheel in the middle of this particular (front or rear) axle. The real robot wheels are then controlled according to Ackermann principle mentioned above. By introducing of virtual steering wheels, the forward kinematic model of the robot can be reduced significantly. The control variables are then the steering angles γ_1, γ_2 of the virtual wheels and angular velocity of the rear steering wheel ω_2.

The position of the robot body is defined in global *floor* coordinate system F as

$$\mathbf{p}_F = \begin{bmatrix} p_x & p_y & \theta_z \end{bmatrix}^T \tag{3}$$

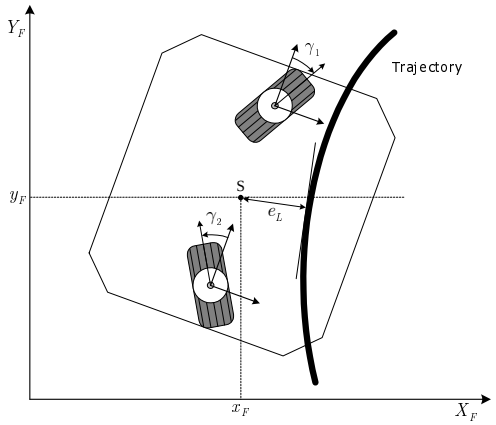


Fig. 4. Line following problem

The trajectory following error e_L is measured as the perpendicular distance of the robot midpoint to the trajectory, which is implemented using cubic splines. The control objective is to follow the trajectory with required forward speed and with minimal trajectory following error.

4. NEURO-FUZZY CONTROLLER

As mentioned above, it is not possible to derive an inverse kinematic model for this kind of mobile robots and therefore to compute the steering angles and driving speeds of the wheels directly from the desired trajectory. For this reason, we developed a neuro-fuzzy controller to solve the trajectory following problem.

The main idea of the presented neuro-fuzzy controller is based on a fact, that it is possible to control the mobile robot following a predefined trajectory by a simple PD-controller under certain conditions. Such PD-controller controls the steering angles of the steering wheels γ_1, γ_2 dependent on the actual distance from the trajectory. However, this controller works properly only for a specific forward speed of the robot and for a limited maximal distance from the trajectory.

To control the robot over its entire operating range, several PD-controller could be designed. Every controller would be optimized for specific working point defined by a forward speed of the robot. Moreover, an additional logic would have to switch among PD-controllers according to the actual operating point. Nevertheless, such type of controller switching causes well-known problems, like ejecting of control signal discontinuities and oscillations on the border between two working points, eventually instability of the control loop. Moreover, the design of controller parameters for various operating points is not an easy task and requires many trials.

Therefore, we fused several PD-controllers with switching logic to a fuzzy logic controller (FLC), which changes the parameters of the PD-controller continuously according to actual conditions. The rule base of the controller contains the decision rules in the form (here as an example for the control of the front steering wheel):

$$r_i : \text{Wenn } e_L \text{ ist } E_i \text{ und } \omega_2 \text{ ist } \Omega_\text{i} \text{ dann} \quad (4)$$

$$\gamma_{1i} \text{ ist } P_i\, e_L + D_i\, \dot{e}_L, \quad i = 1, \ldots, N, \quad (5)$$

where e_L, $\dot{e}_L$ and ω_2 are input variables; E_i and Ω_i are linguistic values of the input variables e_L and ω_2, respectively, used in the i-th rule. Output function implements a PD-controller with parameters P_i and D_i using trajectory following error e_L and its time change $\dot{e}_L$. Hence, the fuzzy controller is based on Takagi-Sugeno inference system.

The crisp output value of the control signal is computed by singleton method. For the steering angle γ_1 of the front wheel, the output is comupted from local outputs of the N rules described above using the formula:

$$\gamma_1 = \frac{\sum_{i=1}^{N} \gamma_{1i}(e_L, \dot{e}_L)\mu_i(e_L, \omega_2)}{\sum_{i=1}^{N} \mu_i(e_L, \omega_2)}, \quad (6)$$

where μ_i is the weight of the i-th rule. The control signal is thus computed as a weighted average from several PD-controllers and hence without discontinuities. The fuzzy rules for the steering angle of the rear wheel γ_2 and its angular speed ω_2 are implemented in an analogous manner.

Fuzzy controller with inference systems described above can be easily implemented as a structured neural network. Main advantage of such implementation is the possibility to optimize its parameters (i.e. the values of proportional and derivative gain of the PD-controllers and the parameters of the input/output fuzzy sets) by some adaptation procedure. Moreover, the parameters can be continuously adapted during robot movement by means of on-line training methods and available input-output data.

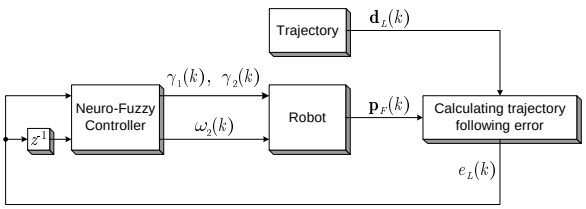


Fig. 5. Control-loop with fuzzy controller

The structure of the closed control loop with neuro-fuzzy controller for trajectory follwing is shown in Fig. 5. The proposed controller has two inputs - trajectory following error $e_L(k)$ and its one time-delayed sample $e_L(k-1)$ to compute the time change of the distance from the trajectory. Trajectory following error $e_L(k)$ is either computed from the actual robot position $\mathbf{p}_F(k)$ and predefined trajectory coordinates $\mathbf{d}_L(k)$ or directly measured by line camera, which is mounted underneath of the robot. The outputs are steering angles $\gamma_1(k)$ and $\gamma_2(k)$ for both front and rear steering axes and the maximum forward speed of the rear wheel $\omega_2(k)$.

The tuning of controller parameters uses backpropagation learning and takes place in two steps.

4.1 *Step I - Training of dynamical model of the system*

In the first phase, the dynamical model of the mobile robot following a predefined trajectory must be trained. This model is required for backpropagating of the trajectory error to the controller during an adaptation of its parameters. Because it is very difficult (if not impossible) to derive this model analytically in the form of differential equations and then backpropagate control error through it, we used multi-layer neural network instead. The inputs to this network are two steering angles $\gamma_1(k)$, $\gamma_2(k)$, angular velocity of the driving wheel $\omega_2(k)$ and last distance of the robot from the trajectory $e_L(k-1)$ plus one delayed sample of every input variable in order to build the model dynamic, i.e. $\gamma_1(k-1)$, $\gamma_2(k-1)$, $\omega_2(k-1)$ and $e_L(k-2)$. Output of the network is the estimated distance from the trajectory $\tilde{e}_L(k-1)$. The structure of proposed neural network is given in Fig. 6.

For learning of neural network based model of the robot, sufficient amount of training data (coordinates of the trajectory, robot position error, robot forward speed, etc.) must be recorded. Three methods of training data acquisition can be used:

- **Manual steering of the mobile robot along the desired path.** In this data acquisition mode, the line camera is used to measure the distance from the trajectory. The trajectory is represented by a contrast line draw on the floor (e.g. black line on the white floor). The mobile robot is teleoperated from MATLAB in this mode.
- **Manual steering of the simulation model of the robot.** In principle it is the same method of data acquisition as the previous one, but instead of the real robot, robot simulation in a 3D environment running in

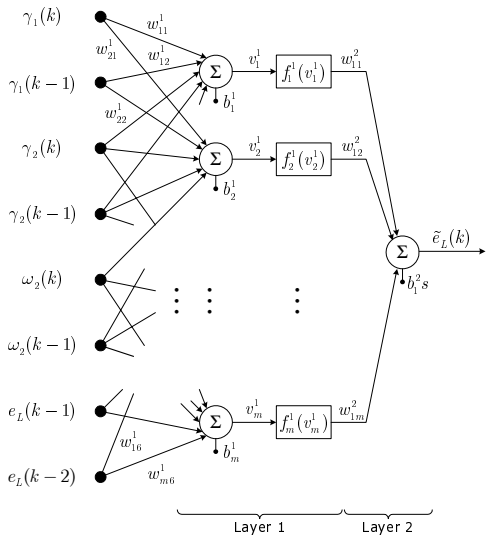


Fig. 6. Neural network for modelling of the trajectory following robot

MATLAB/Simulink is used. The main advantage of this method is the possibility to control the mobile robot also with a greater distance to the trajectory, because the distance is not limited by measuring range of the line camera. Moreover, the motion of the robot can be viewed by simulation from various viewpoints, including "through the robot camera lens" perspective, and the simulation can be stopped if the robot goes away from the trajectory or moves in wrong direction.

- **Autonomous ride of the robot controlled by a PD-controller.** As mentioned above, in certain circumstances it is possible to follow a trajectory by mobile robot using a simple PD-controller. The conditions for trajectory following are limited by forward speed of the robot and maximal distance from the trajectory. Therefore, the training data sets can be recorded also during real or simulated ride of the robot controlled by the PD-controller as long as it moves along the trajectory.

After collecting of sufficient amount of training data, the neural network implementing robot model can be trained. For this purpose, hybrid learning Jang et al. (1997) is very suitable because of its fast convergence and computational effort. This type of learning combines an adaptation of non-linear parameters of the system by the Levenberg-Marquardt algorithm according to equation

$$\mathbf{\Theta}_{N_{New}} = \mathbf{\Theta}_{N_{Old}} - [\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I}]^{-1}\mathbf{J}^T(\mathbf{e}_L - \tilde{\mathbf{e}}_L), \quad (7)$$

and the estimation of its linear parameters using LSE-method

$$\mathbf{\Theta}_{N_{Best}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{e}_L, \quad (8)$$

where $\mathbf{\Theta}_{N_{New}}$ and $\mathbf{\Theta}_{N_{Old}}$ are the vectors of the actual and the last adapted parameters, respectively. $\mathbf{J}$ is the Jacobian of the system, $\mathbf{A}$ is a matrix of activation functions of neurons from the Layer 1, $\mathbf{e}_L$ is a vector of output errors and $\mathbf{\Theta}_{N_{Best}}$ are new parameters of the activation functions of neurons from the Layer 1.

Because the learning of this neural network is a computational consumptive process, it is usually done off-line. Learning of the neural network is depicted in Fig. 7.

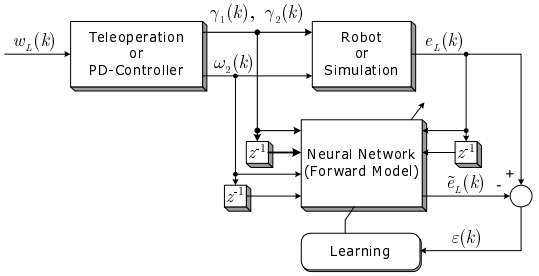


Fig. 7. Training of the feedforward neural network

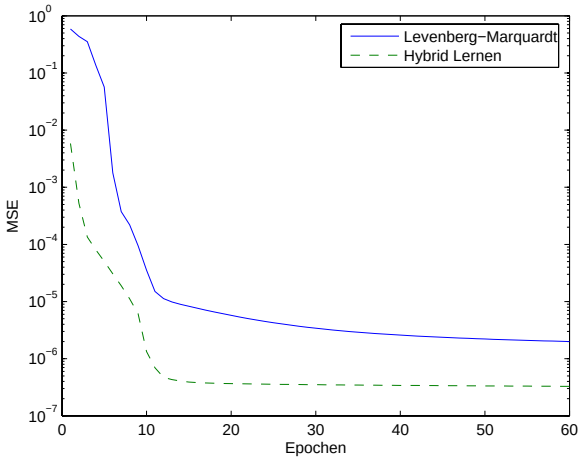


Fig. 8. Comparing of the Levenberg-Marquardt and hybrid learning methods used for training of the neural network based robot model

4.2 *Step II - Training of the neuro-fuzzy controller*

In the second step, the parameters of the neuro-fuzzy controller are adapted so that the mobile robot is able to follow the desired trajectory with specified tracking error and with arbitrary forward speed. For the purpose of learning, a neuro-fuzzy controller is implemented using a structured neural network. In this network, each layer represents a part of the Takagi-Sugeno fuzzy inference system.

The control loop with neuro-fuzzy controller during self-learning is depicted in Fig. 7.

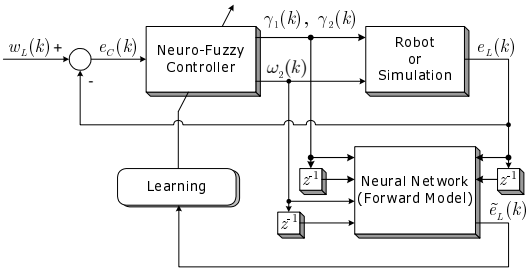


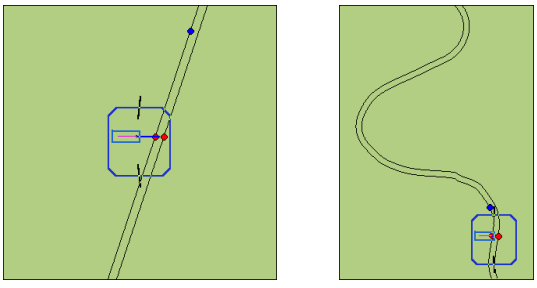
Fig. 9. Training of the neuro-fuzzy controller

The learning of the neuro-fuzzy controller can be realized again using simulations or by experiments with the real robot. However, the robot simulation is the preferred way, because it can accelerate the learning process, Moreover, the problems that can occur by experimentation with the real robot controlled by a not well-tuned neuro-fuzzy controller can be avoided. In such case, the robot must be always stopped to prevent possible collisions. Afterward the learning can be started again, but the training process is delayed. Good results can be achieved by combining of simulations and real experiments. In such case, the parameters of the controller are adapted using simulations first. Thereafter, they are fine-tuned using experiments with the real robot.

The adaptation algorithm is based as before on Levenberg-Marquardt concept. As the measure for parameter adaptation, the sum of squared distances from the trajectory is used. Experiments have shown that it is not necessary to adjust the parameters of input fuzzy sets of the neuro-fuzzy controller, but only parameters of output functions, i.e. the gain values of the PD-controllers.

5. EXPERIMENT REULTS

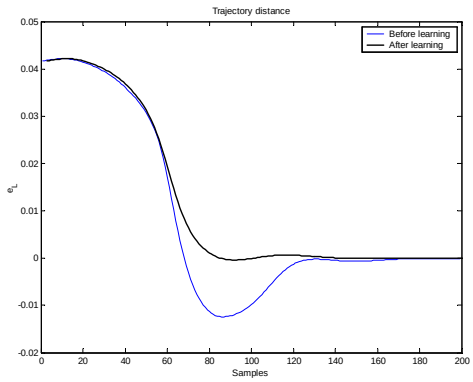
In the following graphs, the results of two experiments are presented. During experiment 1, the robot tries to find a straight line and follow it. During experiment 2, the robot follows a curved trajectory. The situations at the beginning of experiments are showed in Fig. 10.



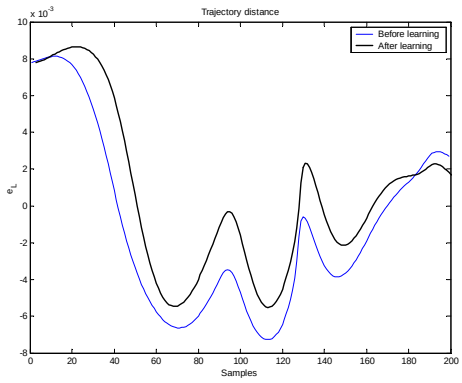
(a) Experiment 1 (b) Experiment 2

Fig. 10. Start positions

In Fig. 11, trajectory distances from both experiments before and after learning are compared. At the beginning, a fuzzy controller with trial-and-error designed output function is used.



(a) Experiment 1



(b) Experiment 2

Fig. 11. Trajectory following error

After several training steps, the trajecotry following error was reduced rapidly, particulary in the first experiment, where the robot has run on the trajectory very quickly and was able to follow it with very small tracking error. In the second experiment, the error was also reduced, but the robot was not able to follow the line with higher precision because of its dynamic constraints. However, the experiments have prooved the feasibility of the proposed neuro-fuzzy controller for the given application.

6. CONCLUSION

In this paper, the problem of motion control of non-holonomic mobile robots have been analyzed. Because of the absence of their inverse kinematic model, the classical model-based control methods are not applicable. As a solution we have developed an intelligent neuro-fuzzy controller for the trajectory following. The controller is able to adapt its parameters by self-learning, therefore

the inverse kinematic or dynamic model of the robot are not required by its design. Moreover, the parameters can be adapted also on-line during robot movement or off-line in some regular intervals, respectively. After learning phase, the mobile robot was capable to follow the desired trajectory with expected precision in any circumstance.

The proposed controller has been tested on the experimental quasi-omnidirectional mobile robot F.A.A.K. The simulation and experiment results have approved, that the designed neuro-fuzzy motion control concept is very suitable for the considered class of non-holonomic mobile robots because of its simple design and implementation.

References

Carlos Canudas de Wit, G. Bastin, and B. Siciliano. *Theory of robot control*. Springer-Verlag, Berlin, 1996.

J.-S.R. Jang, C.-T. Sun, and E. Mizutani. *Neuro-Fuzzy and Soft Computing. A Computational Approach to Learning and Machine Intelligence*. Prentice-Hall, Upper Saddle River, 1997.

Ivan Masr and Michael Gerke. Dsp-based control of mobile robots. In *Proceedings of The European DSP Education and Research Symposium EDERS-2004, Birmingham, UK*, 2004.

Patrick F. Muir. *Modeling and control of wheeled mobile robots*. PhD thesis, Department of Electrical and Computer Engineering and The Robotics Institute, Carnegie Mellon University Pittsburg, 1988.