NONLINEAR IDENTIFICATION AND ADAPTIVE CONTROL OF COMBUSTION ENGINES

Rolf Isermann and Norbert Müller

Darmstadt University of Technology
Institute of Automatic Control
Laboratory of Control Systems and Process Automation
Landgraf-Georg-Str. 4, D-64283 Darmstadt, Germany
Tel: +49/6151-162114, Fax: +49/6151-293445
{RIsermann,NMueller}@iat.tu-darmstadt.de

Abstract: Advanced engine control systems require accurate models of the thermodynamic-mechanical process, which are substantially nonlinear and often time-variant. After briefly introducing the identification of nonlinear processes with grid-based look-up tables and a special local linear Radial Basis Function network (LOLIMOT), a comparison is made with regard to computation effort, storage requirements and convergence speed. A new training algorithm for online adaptation of look-up tables is introduced which reduces the convergence time considerably. Application examples and experimental results are shown for a multi-dimensional nonlinear model of NO_X emissions of a Diesel engine, and for the adaptive feedforward control of the ignition angle of a SI engine.

Keywords: Engine control, Learning systems, Identification, Neural networks, Feedforward control

1. INTRODUCTION

The last two decades in the automotive industry have seen an ever increasing usage of electronics. In the middle 1970s car manufacturers introduced microprocessor-based engine control systems to meet the conflicting state regulations and customer demands of high fuel economy, low emissions, best possible engine performance and ride comfort. Especially the American regulations (clean air act 1963, on-board diagnosis (OBD I 1988, OBD II 1994)) and the corresponding European regulations EURO 1 (1992), EURO 2 (1996), EURO 3 (2000) had a considerable effect on the development of new engine control strategies. New sensors with electrical output and new actuators had to be developed. Also auxiliary units like electro-mechanical controlled carburetors and low pressure injection systems for SI engines and high pressure injection systems for Diesel engines have shown a development from pure mechanical to electro-mechanical devices with electronic control. New actuators were added like for exhaust gas recirculation, camshaft positioning and variable geometry turbochargers. Today's combustion engines are completely microcomputer controlled with many *actuators* (e.g. electrical, electro-mechanical, electro-hydraulic or electro-pneumatic actuators, influencing spark timing, fuel-injector pulse widths, exhaust gas recirculation valves), several *measured output variables* (e.g. pressures, temperatures, engine rotational speed, air mass flow, camshaft position, oxygen-concentration of the exhaust gas), taking into account different *operating phases* (e.g. start-up, warming-up, idling, normal operation, overrun, shut down). The microprocessor-based

control has grown up to a rather complicated control unit with 50-120 look-up tables, relating about 15 measured inputs and about 30 manipulated variables as outputs. Because many output variables like torque and emission concentrations are mostly not available as measurements (too costly or short life time) a majority of control functions is feedforward.

In the future new electronically controlled actuators and new sensors entail additional control functions for new engine technologies (variable turbine geometry (VTG) turbo chargers, swirl control, dynamic manifold pressure, variable valve timing (VVT) of inlet valves). Increasing computational capabilities using floating point processors will allow advanced estimation techniques for non-measurable quantities like engine torque or exhaust gas properties and precise feedforward and feedback control over large ranges and with small tolerances.

2. CONTROL STRUCTURE FOR INTERNAL COMBUSTION ENGINES: STATE OF THE ART

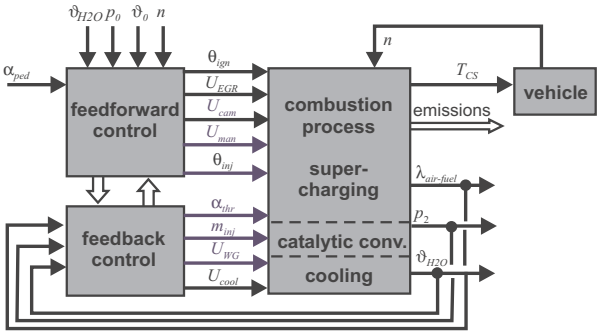

Fig. 1. Simplified control structure of a SI engine

Modern IC engines obtain increasingly more actuating elements. SI engines have - except the classical inputs like amount of injection m_{inj}, ignition angle θ_{ign}, injection angle θ_{inj} - additionally controlled air/fuel ratio, exhaust gas recirculation and variable valve timing, see **Fig. 1**. Diesel engines were until 1987 only manipulated by injection mass and injection angle. Now they have additional manipulated inputs like exhaust gas recirculation, variable geometry turbochargers, common rail pressure and modulated injection, see **Fig. 2**. The engine control system has therefore to be designed for 5-10 main manipulated variables and 5-8 main output variables, leading to a complex nonlinear multiple-input multiple-output system. Because some of the design requirements contradict each other suitable compromises have to be made. Since the majority of control functions are realized by feedforward structures, precise models are required. Feedback control is used in the case of SI engines for example for the λ-control (keeping a stoichiometric air/fuel ratio λ=1 for the catalyst), for

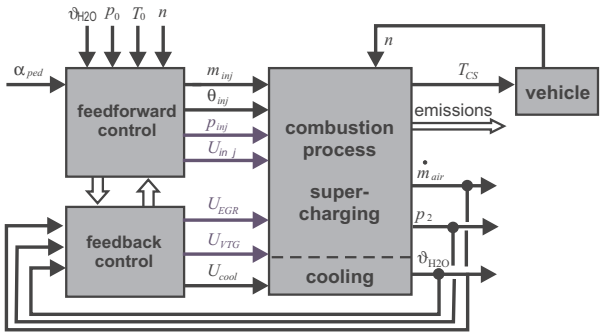

Fig. 2. Simplified control structure of a Diesel engine with turbocharger

the electronic throttle control and for the ignition angle in case of knock. Diesel engines with turbochargers possess a charging pressure control with waste gate or variable vanes and speed overrun break away. Both engines have idling speed control and coolant water temperature control. Additionally, there are several auxiliary closed loop controls like fuel pressure control and oil pressure control. All control functions have to be defined for all possible operating phases. Most feedforward control functions are implemented as grid-based two-dimensional (i.e. two input signals) look-up tables or as one dimensional characteristics. This is because of the strongly nonlinear static and dynamic behavior of the IC engines and the direct programming and fast processing in microprocessors with fixed point arithmetics. Some of the functions are based on physical models with correction factors, but many control functions can only be obtained by systematic experiments on dynamometers and with vehicles.

3. IDENTIFICATION OF NONLINEAR SYSTEMS: LOOK-UP TABLES

Due to the extremely nonlinear dynamics of internal combustion engines, and due to the fact that often measured data is the only available information on the connection between cause and effect, engine control units rely heavily upon the calibration of black-box models. In spite of their limitations to problems with one- and two-dimensional input spaces, grid-based look-up tables are by far the most common type of nonlinear static models used in practice. The reason for this lies in their simplicity and their extremely low computational evaluation demand. Modern engine control units of spark ignition engines contain more than 100 look-up tables (Adler, 2000).

In the following the basics of grid-based look-up tables are given. The off-line estimation of the heights of the interpolation nodes will be explained, followed by the presentation of an on-line adaptation algorithm using the

NLMS approach and a newly developed, modified RLS algorithm.

3.1 *Structure of two-dimensional look-up tables*

Grid-based look-up tables can also be interpreted as a 2nd order B-spline network (Brown and Harris, 1994). They consist of a set of data points or interpolation nodes positioned on a multi-dimensional grid. Each node comprises two components. The scalar data point heights are estimates of the approximated nonlinear function at their corresponding data point position. All nodes are stored e.g. in the ROM of the control unit. For model generation usually all data point positions are fixed a-priori.

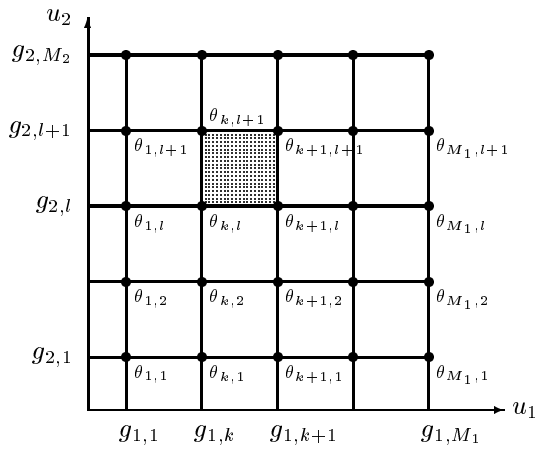

Fig. 3. grid-based placement of the interpolation nodes of a two-dimensional look-up table.

In the following we consider a two-dimensional look-up table of the size $M_1 \times M_2$, see **Fig. 3**. It consists of interpolation nodes, located on the grid lines $g_{1,1}, \ldots, g_{1,M_1}$ and $g_{2,1}, \ldots, g_{2,M_2}$. The model output $\hat{y}(u_1, u_2)$ for a given input (u_1, u_2) is calculated by considering the closest points to the bottom left, bottom right, top left, and top right of the model input, see **Fig. 4**. The heights of the interpolation nodes $\theta_{k,l}$, $\theta_{k+1,l}$, $\theta_{k,l+1}$, and $\theta_{k+1,l+1}$ are weighted with the corresponding opposite area:

$$\begin{aligned}\widehat{y} = \quad & \theta_{k,l} \cdot \frac{A_{k+1,l+1}}{A} + \theta_{k+1,l} \cdot \frac{A_{k,l+1}}{A} \\ & + \theta_{k,l+1} \cdot \frac{A_{k+1,l}}{A} + \theta_{k+1,l+1} \cdot \frac{A_{k,l}}{A}\end{aligned} \tag{1}$$

with the areas

$$\begin{aligned} A_{k+1,l+1} &= (g_{1,k+1} - u_1)\,(g_{2,l+1} - u_2) \\ A_{k,l+1} &= (u_1 - g_{1,k})\,(g_{2,l+1} - u_2) \\ A_{k+1,l} &= (g_{1,k+1} - u_1)\,(u_2 - g_{2,l}) \\ A_{k,l} &= (u_1 - g_{1,k})\,(u_2 - g_{2,l}) \end{aligned}$$

The total area A is calculated as

$$A = (g_{1,k+1} - g_{1,k})\,(g_{2,l+1} - g_{2,l}),$$

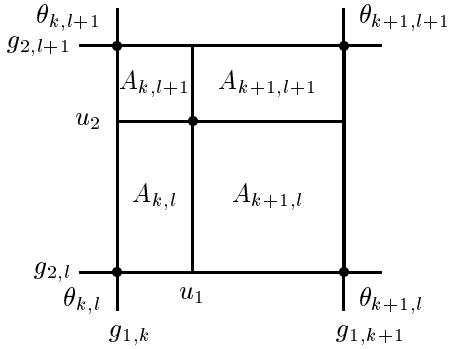

Fig. 4. Areas for interpolation within a two-dimensional look-up table.

with k and l denoting the closest interpolation node which is smaller than u_1 and u_2, respectively. (1) can be described in a basis function formulation

$$\widehat{y} = \sum_{i=1}^{M} w_i \cdot \Phi_i(\underline{u}, \boldsymbol{c}) \tag{2}$$

with the M-dimensional weighting vector

$$\begin{aligned}\underline{w} &= [\theta_{1,1} \ \ldots \ \theta_{M_1,M_2}]^T \\ &= [w_1 \ w_2 \ \ldots \ w_M]^T\end{aligned} \tag{3}$$

which represents the heights of the interpolation nodes. With regard to (1) for each input vector $\underline{u} = [u_1 \ u_2]^T$ in general only 4 basis functions Φ_i are non-zero. The basis functions depend on the positions $\boldsymbol{c}$ of the interpolation nodes and on the input vector $\underline{u}$.

3.2 *Off-line estimation of the look-up table heights*

For generating a look-up table, the most widely applied method to obtain the heights of the interpolation nodes is to position measurement data points directly on the grid points. Then, an additional approximation step can be avoided. However, if available measurement data points do not correspond to the desired positions of the grid, or if the locations of the grid have to be changed due to memory restrictions, estimation techniques have to be applied in order to derive the heights of the interpolation nodes. This allows an arbitrary positioning of the grid lines, irrespective of the location of the measurement data points. In the following an approach for off-line estimation of the heights of the interpolation nodes is explained.

The optimization of the grid location represents a nonlinear optimization problem and is not addressed in this paper, in principle, any nonlinear optimization technique can be applied, (Nelles, 2000).

The off-line estimation of the heights of the interpolation nodes can be performed by least squares method (Isermann, 1992), in order to fit to measurement data that does not lie on the pre-specified grid (Nelles and

Fink, 2000). The loss function $\sum_{i=1}^{N} e^2(i)$ is minimized, where $e(i) = y(i) - \hat{y}(i)$ represent the model errors for data sample $\{\underline{u}(i), y(i)\}$. For a given data set $\{\underline{u}(i), y(i)\}_{i=1}^{N}$, $N > M$, the regression matrix $\boldsymbol{X}$ is

$$\boldsymbol{X} = \begin{bmatrix} \Phi_1(\underline{u}(1), \boldsymbol{c}) & \Phi_2(\underline{u}(1), \boldsymbol{c}) & \cdots & \Phi_M(\underline{u}(1), \boldsymbol{c}) \\ \Phi_1(\underline{u}(2), \boldsymbol{c}) & \Phi_2(\underline{u}(2), \boldsymbol{c}) & \cdots & \Phi_M(\underline{u}(2), \boldsymbol{c}) \\ \vdots & \vdots & & \vdots \\ \Phi_1(\underline{u}(N), \boldsymbol{c}) & \Phi_2(\underline{u}(N), \boldsymbol{c}) & \cdots & \Phi_M(\underline{u}(N), \boldsymbol{c}) \end{bmatrix}$$

where N is the number of data samples of the input data $\{\underline{u}(i)\}_{i=1}^{N}$, and $\boldsymbol{c}$ contains the coordinates of the interpolation nodes.

The parameter vector $\underline{w}$, which contains the heights of the interpolation nodes, see (3), is given as

$$\underline{w} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \underline{y}, \tag{4}$$

with $\underline{y} = [y(1)\ y(2) \ldots y(N)]^T$.

The regression matrix $\boldsymbol{X}$ is typically sparse. In the two-dimensional case each row of $\boldsymbol{X}$ contains only 4 non-zero elements. If the measurement data does not cover the whole input space, some basis functions will not be activated at all, $\boldsymbol{X}$ will contain columns of zeros and may become singular. In order to make the least squares problem solvable the corresponding regressors and their associated heights have to be removed vom $\boldsymbol{X}$ and $\underline{w}$ (Nelles, 2000).

3.3 *On-line adaptation of look-up table heights using NLMS algorithm*

On-line -or instantaneous- learning means that information about the desired input/output behavior is given only by isolated samples $(\underline{u}(t), y(t))$. On-line adaptation is of special importance for time-variant systems, e.g. due to changing ambient conditions or aging. The adaptation method with the lowest computational requirements is the *normalized least mean square* algorithm (NLMS) (Brown and Harris, 1994), it can be applied as follows for the adaptation of the heights of the interpolation nodes:

$$w_i(t+1) = w_i(t) + \beta \cdot e(t) \cdot \frac{\Phi_i(\underline{u}(t), \boldsymbol{c})}{\sum_{j=1}^{M} \Phi_j^2(\underline{u}(t), \boldsymbol{c})},$$

$e(t)$ denotes the error between the correct value $y(t)$ and the old network output $\hat{y}(\underline{u}(t))$. The learning rate β must be within the range $0 < \mu < 2$, however, appropriate values vary between 1 for fast learning and $\mu << 1$ for robustness against measurement noise. In order to further improve robustness against noise and to avoid over-fitting, a dead zone with minimal error value for parameter adaptation can be set (Heiss, 1996).

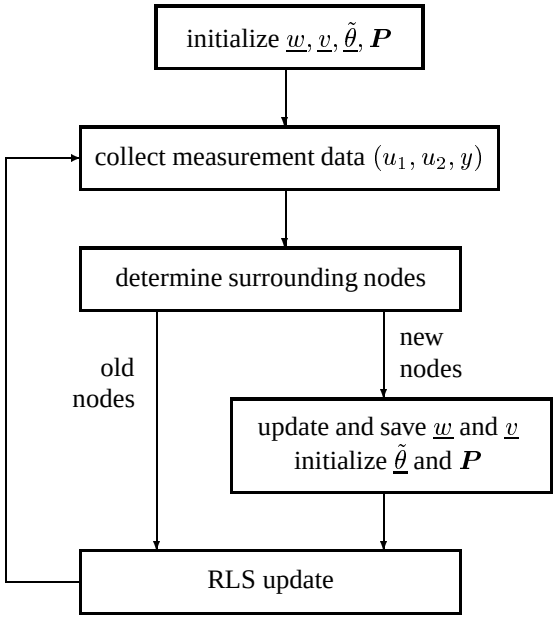

Fig. 5. flow chart of modified RLS algorithm

3.4 *On-line adaptation of look-up table heights using RLS algorithm*

In relation to the usually used NLMS algorithm, the convergence time during on-line adaptation of the heights of the interpolation nodes can be considerably reduced by applying recursive least squares (RLS) algorithms (Isermann, 1992). The look-up table consists of $M_1 \times M_2$ interpolation nodes, however numerical stability can only be obtained, if the problem is reduced to the estimation of fewer parameters. The algorithm proposed in the following estimates only the heights of the surrounding interpolation nodes of a query point, i.e. instead of the M parameters in $\underline{w}$ only 4 parameters $\underline{\tilde{\theta}}$ are estimated at each time instant. During adaptation, the algorithms requires for each height w_i the storage of a variance value σ_i^2. Without prior knowledge the variances $\underline{v} = [\sigma_1^2 \ldots \sigma_M^2]^T$ can be initialized as $\sigma_i^2 = 100 \ldots 1000$.

We consider the estimation of the parameter vector $\underline{\tilde{\theta}} = [\theta_{k,l}, \theta_{k+1,l}, \theta_{k,l+1}, \theta_{k+1,l+1}]^T = [w_j \ldots w_{j+3}]^T$, see **Fig. 4**. The matrix $\boldsymbol{P}$ is initialized as

$$\boldsymbol{P} = \begin{pmatrix} \sigma_j^2 & 0 & 0 & 0 \\ 0 & \sigma_{j+1}^2 & 0 & 0 \\ 0 & 0 & \sigma_{j+2}^2 & 0 \\ 0 & 0 & 0 & \sigma_{j+3}^2 \end{pmatrix}, \tag{5}$$

standard RLS estimation can then be applied:

$$\underline{\gamma}(t+1) = \frac{1}{\lambda + \underline{\psi}^T(t+1)\, \boldsymbol{P}(t)\, \underline{\psi}(t+1)} \boldsymbol{P}(t) \underline{\psi}(t+1)$$

$$\underline{\tilde{\theta}}(t+1) = \underline{\tilde{\theta}}(t) + \underline{\gamma}(t+1) \left(y(t+1) - \underline{\psi}^T(t+1) \underline{\tilde{\theta}}(t) \right)$$

$$\boldsymbol{P}(t+1) = \frac{1}{\lambda} \left(\boldsymbol{I} - \underline{\gamma}(t+1) \underline{\psi}^T(t+1) \right) \boldsymbol{P}(t)$$

with

$$\underline{\psi}(t) = [\Phi_j(\underline{u}(t), \boldsymbol{c}) \; ... \; \Phi_{j+3}(\underline{u}(t), \boldsymbol{c})].$$

λ is a forgetting factor that allows exponential forgetting of the old measurements. It allows a trade-off between tracking performance and noise suppression. Φ_j are the weighting functions as in (2).

Before each RLS update the algorithm verifies that the surrounding interpolation nodes did not change as a result of a changing operating condition. However, if the surrounding interpolation nodes changed since the last update, the diagonal elements of $\boldsymbol{P}$ are retrieved and saved in the variance vector $\underline{v}$. $\boldsymbol{P}$ is re-initialized as in (5), the variances of the new interpolation nodes are used as the new diagonal elements. **Fig. 5** shows the flow chart of the proposed RLS algorithm for updating look-up tables.

In order to avoid deterioration of the already estimated model parameters, a supervisory level freezes the identification algorithm if excitation is insufficient, e.g. if deviations between model output and desired output are incremental.

4. FAST LOCAL LINEAR NEURAL NETWORKS

Look-up tables are restricted to one- and two-dimensional applications since memory and computational requirements grow exponentially with the number of input signals. For high dimensional problems, neural networks have been successfully applied. Well known neural networks for the identification of nonlinear processes are multilayer perceptrons (MLP) and radial basis functions (RBF), (Isermann *et al.*, 1997). In the following local linear model networks are considered which have several advantages for engine measurements. Local linear model tree (LOLIMOT) is an extended radial basis function network. It is extended by replacing the output layer weights with a linear function of the network inputs (Nelles, 1999). Thus, each neuron represents a local linear model with its corresponding validity function. Furthermore, the radial basis function network is normalized, that is the sum of all validity functions for a specific input combination sums up to one. The Gaussian validity functions determine the regions of the input space where each neuron is active. The input space of the net is divided into M hyper-rectangles, each represented by a linear function.

The output y of a LOLIMOT network with n inputs $x_1, \ldots, x_n$ is calculated by summing up the contributions of all M local linear models

$$\hat{y} = \sum_{i=1}^{M} (w_{i0} + w_{i1}x_1 + ... + w_{in}x_n) \cdot \Phi_i(\underline{x}) \quad (6)$$

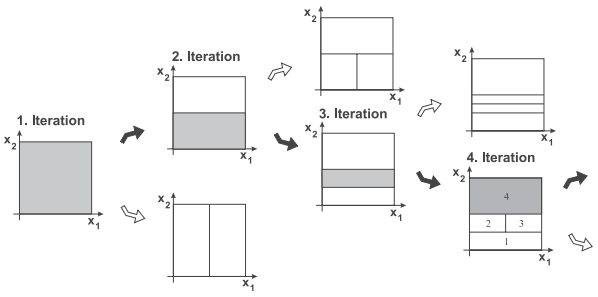

Fig. 6. First four iterations of LOLIMOT training algorithm

where w_{ij} are the parameters of the i-th linear regression model and x_i is the model input. The validity functions Φ_i are typically chosen as normalized Gaussian weighting functions.

$$\Phi_i(\underline{x}) = \frac{\mu_i}{\sum_{j=1}^{M} \mu_j} \quad (7)$$

with

$$\mu_i = \exp\left(-\frac{1}{2}\frac{(x_1 - c_{i1})^2}{\sigma_{i1}^2} - ... - \frac{1}{2}\frac{(x_n - c_{in})^2}{\sigma_{in}^2}\right)$$

with center coordinates $\underline{c}_{ij}$ and dimension individual standard deviations $\underline{\sigma}_{ij}$. LOLIMOT is trained as follows. In an outer loop the network structure is optimized. It is determined by the number of neurons and the partitioning of the input space, which is defined by the centers and standard deviations of the validity functions. An inner loop estimates the parameters and possibly the structure of the local linear models. The network structure is optimized by a tree construction algorithm that determines the centers and standard deviations of the validity functions (**Fig. 6**).

The LOLIMOT algorithm partitions the input space in hyper-rectangles. In the center of each hyper-rectangle, the validity function of the corresponding linear model is placed. The standard deviations are chosen proportional to the size of the hyper-rectangle. This makes the size of the validity region of a local linear model proportional to its hyper-rectangle extension. Thus, a model may be valid over a wide operating range of one input variable but only in a small area of another one. At each iteration of the outer loop, one additional neuron is placed in a new hyper-rectangle, which is derived from the local linear model with the worst local error measure

$$J_{local} = \sum_{j=1}^{N} \Phi_i(\underline{x}(j)) \cdot (y(j) - \hat{y}(j))^2.$$

In other words, the local error is the sum of the squared errors weighted with the corresponding validity function Φ_i over all data samples N. The new hyper-rectangle is

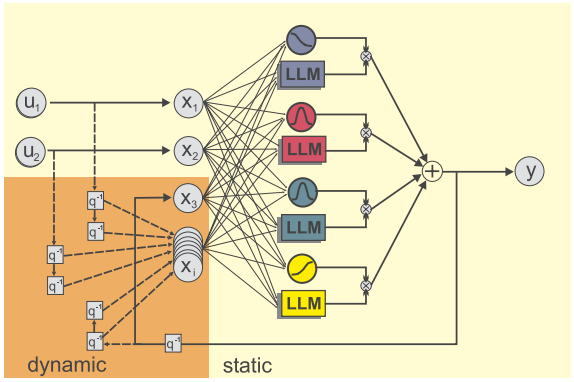

Fig. 7. LOLIMOT net with external dynamics, LLM: Local Linear Models The filters are chosen as simple time delays q^{-1}

then chosen by testing the possible cuts in all dimensions and taking the one with the highest performance improvement.

In an inner loop the parameters of the local linear models are estimated by a local weighted least-squares technique. The prediction errors are weighted with the corresponding validity function. Each local linear model is estimated separately, that is the overlap between neighbored local models is neglected. This approach is very fast and robust. It is especially important to note that due to the local estimation approach the computational demand increases only linearly with the number of local models.

In addition to approximating stationary relations of nonlinear processes, LOLIMOT is also capable of simulating the dynamic behavior of processes. In order to model multivariate, nonlinear, dynamic processes the following time-delay neural network approach can be taken.

$$\hat{y}(t) = f\left(\underline{x}(t)\right)$$
$$\underline{x}(t) = [u_1(t-1), ..., u_1(t-m), ..., u_p(t-1)..., u_p(t-m), \hat{y}(t-1), ..., \hat{y}(t-m)]^T$$

where m is the dynamic order of the model, $u_1(t), \ldots, u_p(t)$ are the p model inputs and $\hat{y}(t)$ is the model output at time t, see **Fig. 7**.

Some of the main advantages of the LOLIMOT approach is its fast training time of some 10..30 s, compared to many minutes to hours with other neural networks, its applicability to adaptive problems (Fischer *et al.*, 1998) and the interpretability of the net structure and parameters in a physical sense.

4.1 *On-line Adaptation of the LOLIMOT Parameters*

For on-line adaptation of the LOLIMOT network, the validity functions Φ_i are kept fixed and only the parameters of the local linear models are adapted.

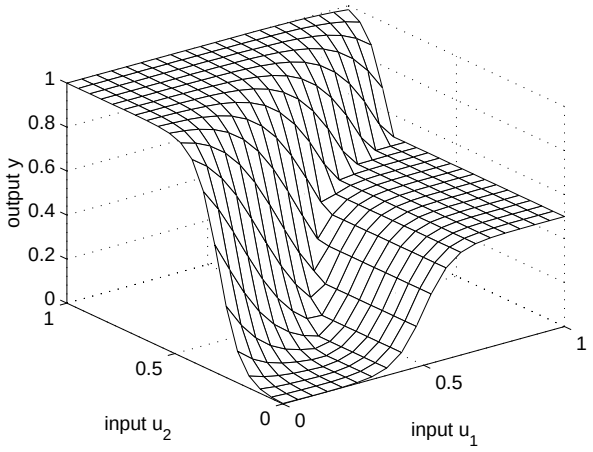

Fig. 8. Nonlinear test function with two input signals.

The following *recursive weighted least-squares* (RWLS) algorithm with exponential forgetting is utilized to estimate the parameters of each local linear model. For the j-th local linear model, it computes a new parameter estimate $w_j(t)$ at the time instant t as follows, see (Isermann *et al.*, 1992):

$$\underline{\gamma}(t+1) = \frac{1}{\frac{\lambda}{\Phi_j(\underline{x}(j))} + \underline{\psi}_j^T(t+1)\,\boldsymbol{P}_j(t)\,\underline{\psi}_j(t+1)}\,\boldsymbol{P}_j(t)\underline{\psi}_j(t+1)$$

$$\underline{w}_j(t+1) = \underline{w}_j(t) + \underline{\gamma}_j(t+1)\left(y(t+1) - \underline{\psi}_j^T(t+1)\underline{w}_j(t)\right)$$

$$\boldsymbol{P}_j(t+1) = \frac{1}{\lambda}\left(\boldsymbol{I} - \underline{\gamma}_j(t+1)\underline{\psi}_j^T(t+1)\right)\boldsymbol{P}_j(t)$$

λ is a forgetting factor that implements exponential forgetting of the old measurements and Φ_j is the weighting of the actual data with the validity function value. The role of the forgetting factor as well as supervisory tasks for the on-line adaptation are discussed in (Fink *et al.*, 1999).

5. COMPARISON OF LOOK-UP TABLE AND LOCAL LINEAR NEURAL NETWORK FOR THE TWO-DIMENSIONAL CASE

For the comparison of the on-line adaptation algorithms of look-up table and local linear neural network, a nonlinear test function with two input signals was considered, see **Fig. 8**. The testing structure shown in **Fig. 9** was used to train the look-up table using NLMS and RLS adaptation algorithms and the LOLIMOT neural network using RWLS adaptation algorithm. u_1 and u_2 have been random numbers within $[0, 1]$, before training the network parameters of look-up table and LOLIMOT neural network have been initialized to zero.

Fig. 10 shows the comparison of memory requirements and computation time, and of time of convergence for the on-line training. In the first and second diagrams,

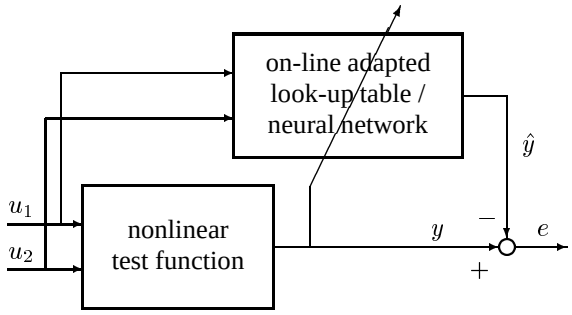

Fig. 9. Testing structure used for on-line adaptation of look-up tables and neural networks.

values in black refer to *generalization*, values in white refer to the case that also *adaptation* is activated. All data is normalized to look-up table values without adaptation. The first column shows values of the look-up table, using the NLMS adaptation algorithm. Memory and computational requirements are small, however time of convergence is large. Additional memory and computational requirements of the NLMS adaptation is fractional and not visible in the diagram. If the look-up table is adapted using the proposed RLS algorithm, memory expense doubles since in addition to the heights also their covariances have to be stored. However, time of convergence can be considerably reduced. Memory requirements are large for the LOLIMOT neural network, also the computational requirements, which is due to the evaluation of the exponential functions. However, due to the fast RWLS algorithm, time of convergence during adaptation is very fast.

For the two dimensional case the conventional look-up table outperforms the LOLIMOT neural network, if fast convergence is required the application of RLS adaptation is advantageous.

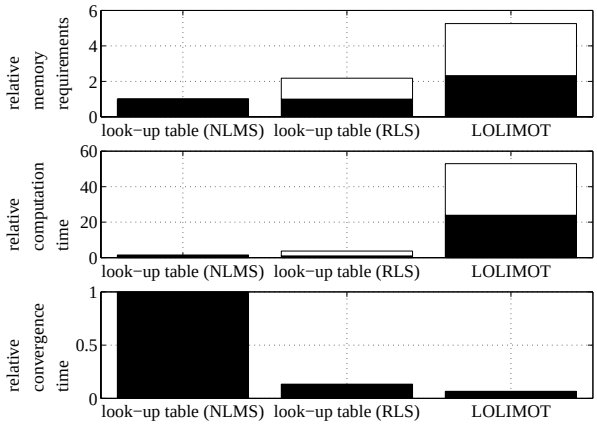

Fig. 10. Comparison of memory and computational requirements, as well as time of convergence for the on-line training of a two-dimensional look-up and a two-dimensional neural network.

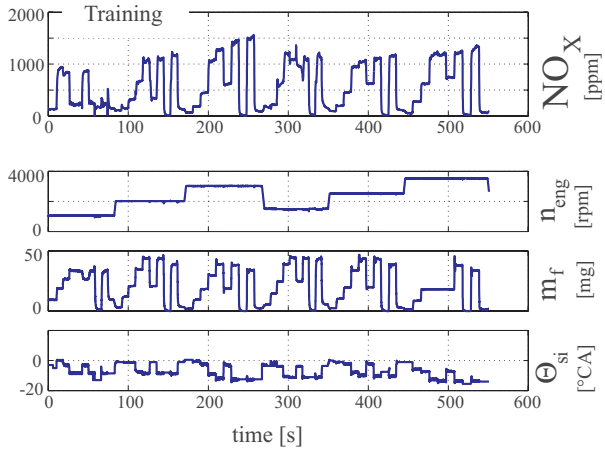

Fig. 11. Training data set for a dynamic NO_X model

Notice however, that for the high dimensional case memory and computational requirements of the look-up table grow exponentially with the number of input signals, which is not the case for the LOLIMOT algorithm (Nelles, 1999). Therefore in the high dimensional case the LOLIMOT algorithm would be the better choice.

6. NONLINEAR IDENTIFICATION OF EXHAUST GAS COMPONENTS

Theoretical models of exhaust gas concentrations are too complex and labor-intensive for control design. The reason is that complex thermodynamic and chemical equations, side-effects like swirl, tumble, quenching, local temperatures etc. have to be taken into account (Heywood, 1988). Therefore, exhaust gas models are mostly experimental.

In contrast to two-dimensional look-up tables, neural nets can easily be applied for high order problems in order to represent all relevant variables influencing the emissions. Depending on the complexity of the engine, more than 5-7 inputs might be necessary to consider the most relevant variables concerning the exhaust gas formation. Among these are e.g. the engine speed, load, injection angle and -pressure, EGR and the position of the guide blades of turbochargers with variable turbine geometry (VTG). Another advantage of dynamic neural nets is the ability to dynamically measure the engine's behavior and calculate its static maps by means of the dynamic neural network. Consequently, there is no need to measure the whole look-up table using measurement points on equidistant grids, which helps to save expensive test stand time. As an example for engine emission models, a dynamic NO_X model of a 1.9l direct injection diesel engine is described. When recording the training data for dynamic models, one should always keep in mind, that the measured data has to contain the whole range of amplitudes and dynamics of the process in

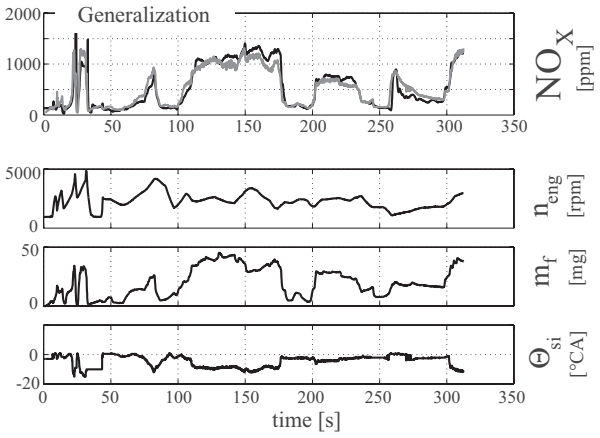

Fig. 12. Generalization of the NO_X model trained with the data in Fig. 11

order to get satisfying results. APRBS signals (amplitude modulated pseudo random binary signal) are often very suitable as process inputs because they excite the process at a wide range of amplitudes and frequencies. In this example, step responses of differing amplitudes have proven to be suitable for the training of the NO_X-models. **Fig. 11** shows the dynamically measured NO_X and the three net-inputs m_f, n_{eng} and Θ_{si} of the training data set (EGR and VTG were disabled during this test).

After building the neural model according to (8), the model can be applied to a new (unknown) data set and simulates the NO_X according to the respective input data (on-line in the vehicle, if necessary).

$$\begin{aligned} N\hat{O}_X(k) = f_{LOLIMOT}(&m_f(k-1), \\ &n_{eng}(k-1), \Theta_{si}(k-1), N\hat{O}_X(k-1) \end{aligned} \tag{8}$$

Fig. 12 illustrates the good generalization performance of a first order dynamic net with 15 neurons for the NO_X emissions. Despite the excitation of the net-inputs

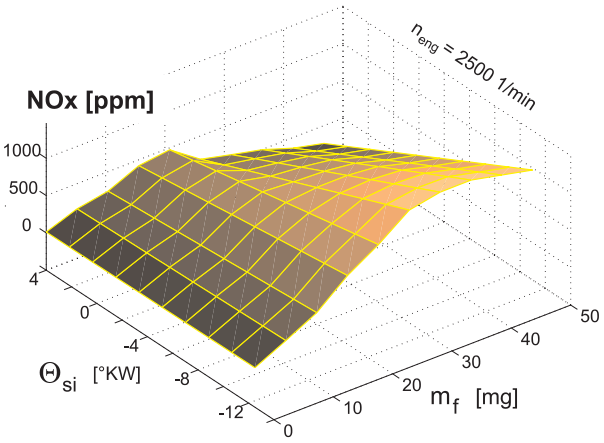

Fig. 13. NO_X-Map, calculated from dynamic neural net

in Figs. 11 and 12 were of quite different quality, the simulated NO_X is able to follow the measured values with an error of just a few percent, which was quite satisfying.

Another important feature of these dynamic neural models is the possibility of deriving stationary maps from dynamic models. The data in **Fig. 11** led to a dynamic model according to (8). This model was then used to calculate the stationary two-dimensional look-up table shown in **Fig. 13**. Thus, it is possible to get a good visual overview on the engine's characteristics by maps derived from dynamic measurements. Even more importantly, this feature allows a fast dynamic measurement of the engine's characteristics and a calculation of the static engine maps by means of the dynamic neural model.

The obtained dynamic model for the NO_X emissions can be used for online optimization tasks for modern engine control units (Hafner *et al.*, 2000). If sensors for the NO_X emissions should be available as they are used in latest direct injection gasoline engines (Glück *et al.*, 2000), the RLWS adaptation method, see Section 4.1 could be used for online adaptation. This could be used for compensation of varying fuel qualities or manufacturing tolerances.

7. ADAPTIVE FEEDFORWARD CONTROL OF IGNITION ANGLE

Cylinder pressure signals contain valuable information for closed loop engine control. For using this information low cost combustion pressure sensors with high long-term stability have been developed (Anastasia and Pestana, 1987; Herden and Küsell, 1994) and are already installed into certain production engines (Inoue *et al.*, 1993).

The objective of *ignition control* is to achieve optimum engine efficiency for each combustion event. Generally factors that influence the optimal ignition angle are engine specifications like configuration of the combustion chamber, operating conditions like engine speed, load, temperature, and exhaust gas recirculation flow rate, as well as ambient conditions such as air temperature, air pressure and humidity in the atmosphere.

Standard ignition control systems are based on feedforward control and therefore rely heavily upon calibration of look-up tables. The database values are initially calibrated from an analysis of a nominal engine under fixed environmental conditions. However, aging effects, manufacturing tolerances or a changing environment usually change the engine characteristics and lead to deteriorating performance.

Based on cylinder pressure sensors, Learning Feed-Forward Control (LFFC) can be used in order to opti-

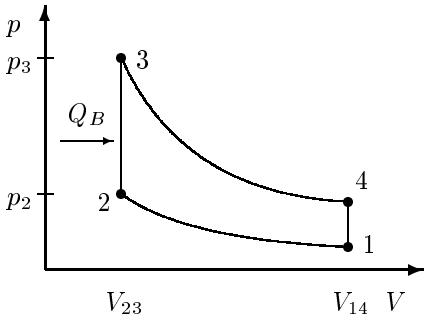

Fig. 14. Constant volume diagram of an ideal constant-volume combustion cycle

mize the point of ignition of each cylinder. The LFFC consists of a function approximator, the latter is represented by a two-dimensional look-up table. The adaptive look-up table is trained on-line using the RLS training algorithm. The learning process is performed during normal operation of the engine, without determining special excitation signals. The learning system is just collecting and processing input-output samples at the individual operating points.

7.1 *Cylinder Pressure Evaluation*

Not only thermodynamic analysis but also extensive experimental results suggest that the optimum efficiency of each combustion event is achieved if 50 % energy conversion occurs at 8° crankshaft angle (CA) after Top Dead Center (TDC), irrespective of the operating point, (Bargende, 1995; Matekunas, 1986). Therefore the crank angle location of 50% energy conversion is to be calculated and to be controlled by modifying the point of ignition accordingly.

The energy conversion during a combustion cycle can be described by the Mass Fraction Burned (MFB) $x_{MFB}(\theta)$ of the cylinder charge at a specific crank angle θ. The MFB, in turn, can be approximated by

$$x_{MFB}(\theta) = \frac{p(\theta)\,V(\theta)^{\kappa} - p_i\,V_i^{\kappa}}{p_{eoc}\,V_{eoc}^{\kappa} - p_i\,V_i^{\kappa}} \qquad (9)$$

where $p(\theta)$ and $V(\theta)$ denote the measured cylinder pressure and the cylinder volume at a certain crank angle, respectively. The indices eoc and i denote specific crank angle locations at the *e*nd *o*f *c*ombustion and before *i*gnition, respectively. κ stands for the polytropic index.

This approximation can be derived by considering the pressure-volume diagram of an ideal constant-volume combustion cycle, see **Fig. 14**. For the compression stroke ($1 \rightarrow 2$), as well as for the power stroke ($3 \rightarrow 4$) an polytropic behavior can be assumed, i.e.

$$p(\theta) \cdot V(\theta)^{\kappa} = C_1 \qquad (10)$$

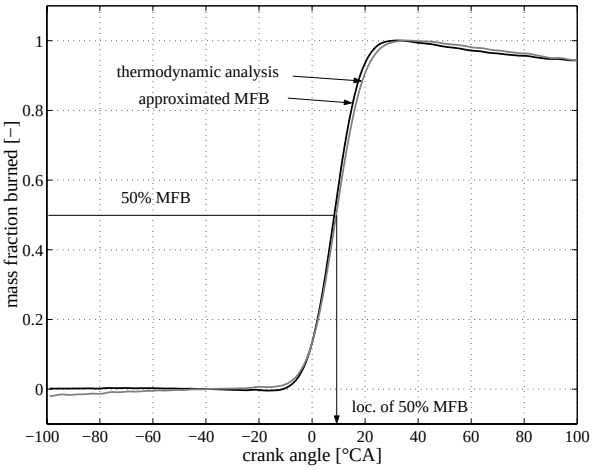

Fig. 15. Approximation of MFB in comparison to thermodynamic analysis. The location of 50% MFB is to be controlled by appropriate settings of the point of ignition.

for the compression stroke and

$$p(\theta) \cdot V(\theta)^{\kappa} = C_2 \qquad (11)$$

for the power stroke. Therefore the amount of energy Q_B, released between ignition (point 2) and the end of combustion (point 3) is

$$Q_B \sim C_2 - C_1 = p_{eoc} \cdot V_{eoc}^{\kappa} - p_i \cdot V_i^{\kappa}. \qquad (12)$$

During combustion the energy released is proportional to $p(\theta)\,V(\theta)$, normalized to the values between end of combustion and ignition, which leads to equation (9).

In **Fig. 15** for an arbitrary cycle the approximation of MFB according to (9) is compared to MFB calculated by thermodynamic analysis of the cylinder pressure. In the presented control system the crank angle location of $x_{MFB}(50\%)$ is calculated by means of the approximation mentioned above.

7.2 *Control Structure*

Applying closed-loop ignition control with for example standard PI-controllers, this cannot provide acceptable control performance under fast changing operating conditions since the controllers cannot be tuned for high control effort. This is due to the fact, that after ignition of the air-fuel mixture, the combustion cannot be influenced any further. Therefore the ignition angle can only be computed for the next cycle, based on measurements from the present engine cycles. Therefore a dead time of one cycle is inherent. Moreover as there exist significant cycle fluctuations even under steady operating conditions, the results of cylinder pressure evaluation

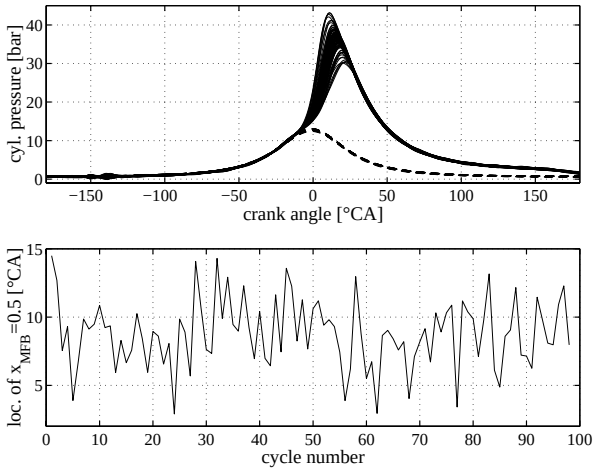

Fig. 16. Upper diagram: measured cylinder pressure signals and reconstructed polytrope; lower diagram: calculated crank angle locations of 50% MFB. $n_{mot} = 3000$ rpm, 50% load

of several engine cycles have to be averaged (e.g. a moving average over 10 cycles is used). Because the system error usually differs from one engine operating condition to another, during engine transients it takes a certain amount of time for the PI controller to ,,integrate" to a new ignition angle. The stochastic nature of the combustion events can be seen in **Fig. 16**. The upper diagram shows the measured cylinder pressure signals of the compression and the power strokes of 100 consecutive cycles of an arbitrary cylinder. The dotted lines represent the reconstructed polytrope, which is the component of the cylinder pressure which is due to the piston motion (towed or motored pressure) . The lower diagram depicts the corresponding crank angle locations of 50% MFB, which are to be controlled to 8° CA. The stochastic nature of the combustion events is clearly visible.

This motivates the use of Learning Feedforward Control (LFFC), whose general structure is shown in **Fig. 17**. The linear controller C is used to compensate random disturbances and to provide a reference signal for the learning feed-forward controller. It does not need to have a high performance and can be designed in such a way that it provides a robust stability. Since the learning feedforward controller acts instead of a controller's integral term, the linear controller is preferably a simple proportional gain which can be deactivated for noise suppression.

For the ignition control system the function approximator is divided into the conventional, fixed ignition look-up table and into the adaptive offset map, see **Fig. 18**. The operating condition is determined by the engine load (a normalized value calculated from the intake manifold pressure signal) and the engine's rotational speed.

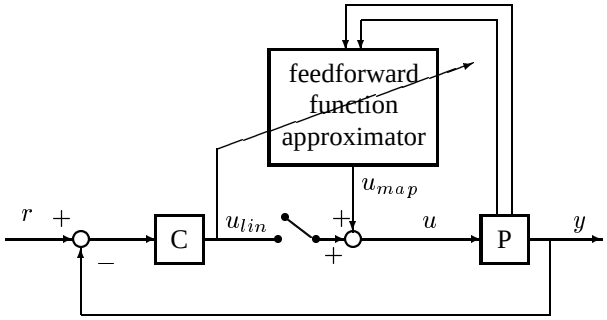

Fig. 17. General structure of a LFFC (Learning Feedforward Controller). The controller C is basically used for the adaptation of the feedforward map

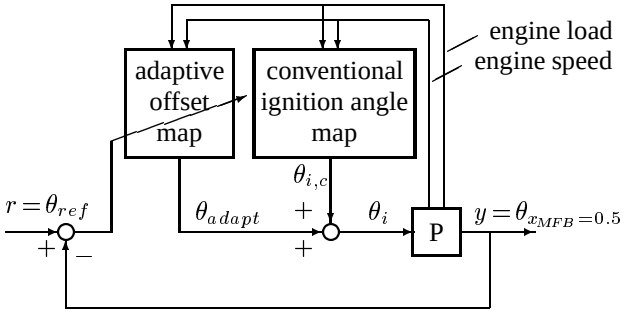

Fig. 18. Structure of the adaptive feedforward ignition control system

The conventional look-up table determines an approximate value of the ignition angle and is valid for all cylinders, it is depicted in **Fig. 20**. For each cylinder the location of 50% MFB is calculated and compared to the reference location of 8° CA. Correction values are calculated and memorized by the adaptive offset look-up table of each cylinder, at the corresponding operating condition. In order to reduce noise in the controller output signal θ_i, no linear proportional controller C is used in parallel to the function approximator, see **Fig. 17**.

7.3 *Measurement Results*

In **Fig. 19** at a constant engine speed of 2500 rpm a certain load change sequence is repeated for three times, see the third diagram. The first diagram shows the crank angle locations of 50% MFB for two cylinders, which are to be controlled to 8° CA after TDC. The second diagram depicts the correction values $\theta_{z,adapt}$ for the ignition points of the corresponding cylinders. For the first 50 sec only the conventional feedforward control is active. The considerable deviations of the crank angle locations of 50% MFB from the optimal value for best engine efficiency at 8° CA after TDC is visible, see the upper diagram. Between 50 and 168 sec the adaptive feedforward controller is active. Since the adaptive input-output map of each cylinder had been initialized to

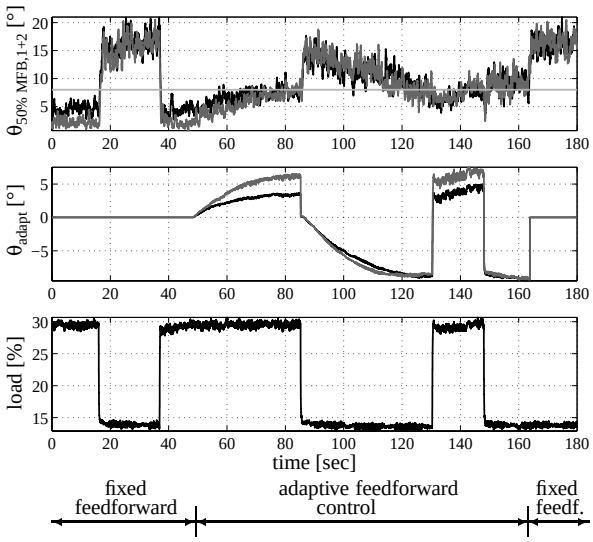

Fig. 19. Control performance of the ignition control system during three equivalent load changes. The adaptive look-up table is initialized with zero, after 50 sec the adaptive feedforward control is activated.

zero, it first has to be adapted for both operating conditions. The control performance for the already adapted feedforward control is shown between 130 and 160 sec. Then it is switched back to the conventional, fixed feedforward control. **Fig. 20** shows the conventional, fixed ignition angle look-up table (a.)) and the adaptive offset map of the second cylinder after a training sequence (b.)).

In spite of the considerable, stochastic nature of the combustion events, the adaptive feedforward control allows to maintain the mean crank angle location of 50% MFB around its optimal value at about 8° CA after TDC.

8. REAL-TIME COMPUTER SYSTEM AND EXPERIMENT CONTROL

As a matter of course the presented cylinder pressure evaluation and the adaptive control algorithms require a considerable calculation effort. State-of-the-art engine

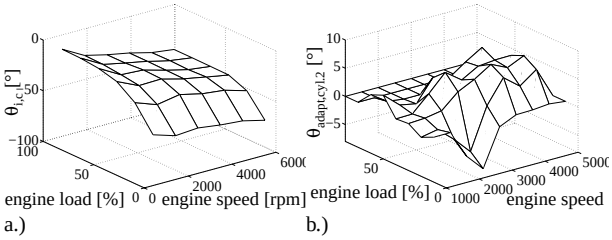

Fig. 20. Conventional ignition map (a.)) and adaptive offset map of the second cylinder after a training sequence (b.)).

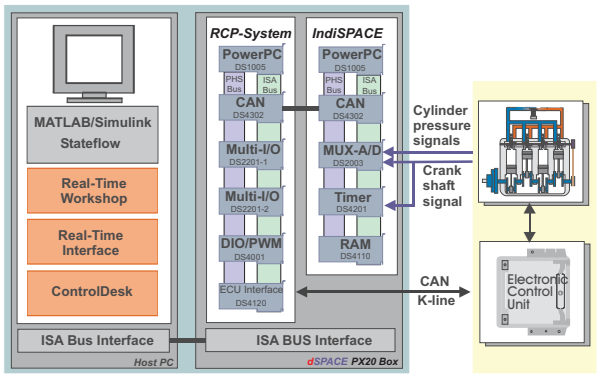

Fig. 21. Rapid control prototyping system configuration

control units (ECU) would not be able to calculate the algorithms in an appropriate time. Assuming that the growth in calculating power proceeds at the high speed of the past, future ECUs should make sufficient calculation time available within only a few years.

However, special real-time computer systems already allow an implementation and testing of the proposed algorithms in vehicle or engine test stands. In order to operate the control algorithms under realistic conditions, an indicating system was implemented on the basis of a PowerPC, type Motorola MPC 750, 480MHz, and coupled with a rapid control prototyping (RCP) system of the company dSPACE, which is also based on a PowerPC. Both systems operate in parallel to the production car's ECU. The goal of the RPC system is to enable a very fast end easy implementation and testing of new control concepts on real-time hardware. The user is enabled to code newly developed algorithms from block diagrams (e.g. MATLAB/Simulink) and download the code by means of an automatic code generation software to the real-time hardware. A complete design iteration can thus be accomplished within a few minutes (Hanselmann, 1996).

The indicating system allows to evaluate the cylinder pressure in real time with a resolution of 1° crankshaft angle for four cylinders. Thermodynamic and signal-based characteristic cylinder pressure features are transmitted to the RCP system. Based on this information, appropriate correction values for the injection duration, the ignition angle and the exhaust gas recirculation rate are calculated and sent to the production car's ECU via CAN bus interface, see **Fig. 21**. The described real-time hardware system enables very fast and easy implementation and testing of complex algorithms including extensive data preprocessing for the cylinder pressure, even under harsh real-time conditions of combustion engines. **Fig. 22** shows the dynamometer, consisting of an 160kW asynchronous motor coupled to the combustion engine.

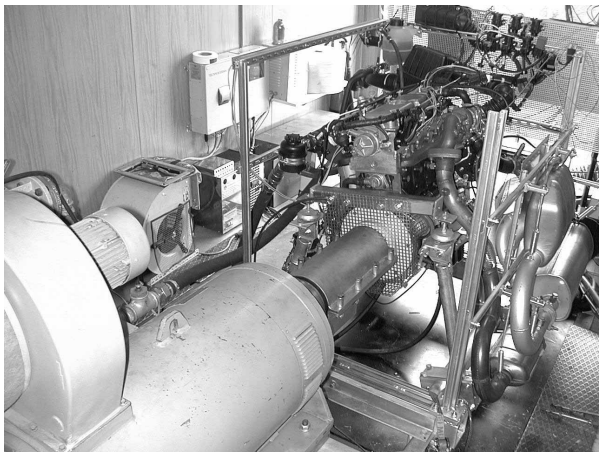

Fig. 22. Asynchronous motor coupled to the combustion engine

9. CONCLUSIONS

The demands for improved engine emissions, performance and efficiency require the application of advanced identification and control strategies. The proposed adaptation algorithms for grid-based look-up tables allow the online training of the heights of the interpolation nodes with improved convergence time. Fast neural networks with local linear models can approximate multi-dimensional problems within small training times of mostly less than a minute. The latter has been shown for the dynamic modelling of the NO_X emissions of a direct injection Diesel engine.

Recent computational capacities and sensor technologies allow the online evaluation of cylinder pressure signals for ignition control. Using cylinder-selective adaptive feedforward controllers, the 50% point of energy conversion can be kept near to the optimal point. The proposed control system is capable to compensate for manufacturing tolerances, fuel quality variations and long-term effects such as aging or wear of the engine. Moreover, the complexity and cost of engine calibration can thus be reduced.

10. REFERENCES

Adler, Ulrich (Ed.) (2000). *Automotive Handbook*. 5 ed.. Robert Bosch GmbH.

Anastasia, Ch.M. and G.W. Pestana (1987). A cylinder pressure sensor for closed loop engine control. In: *SAE Technical Paper Series*. number 870288.

Bargende, M. (1995). Most optimal location of 50 % mass fraction burned and automatic knock detection components for automatic optimization of SI-engine calibrations. *MTZ Worldwide*.

Brown, M. and C. Harris (1994). *Neurofuzzy Adaptive Modelling and Control*. Prentice Hall. New York.

Fink, F., M. Fischer and O. Nelles (1999). Supervision of nonlinear adaptive controllers based on fuzzy models. In: *14th IFAC World Congress*. Beijing, China.

Fischer, M., O. Nelles and R. Isermann (1998). Adaptive predictive control of a heat exchanger based on a fuzzy model. *Control Engineering Practice* **6**(2), 259–269.

Glück, K.-H., U. Göbel, H. Hahn, J. Höhne, R. Krebs, T. Kreuzer and E. Pott (2000). Die Abgasreinigung der FSI-Motoren von Volkswagen. *MTZ Motortechnische Zeitschrift* **61**(6), 402–412.

Hafner, M., M. Schüler and R. Isermann (2000). Model-based optimization of ic engines by means of fast neural networks - part 2: Static and dynamic optimization of fuel consumption versus emissions. *MTZ Worldwide*.

Hanselmann, H. (1996). Automotive control: From concept to experiment to product. In: *IEEE International Symposium on Computer-Aided Control System Design*.

Heiss, M. (1996). Error-minimizing dead-zone for basis function networks. *IEEE Transactions on Neural Networks* pp. 1503–1506.

Herden, W. and M. Küsell (1994). A new combustion pressure sensor for advanced engine management. In: *SAE Technical Paper Series*. number 940379.

Heywood, J. B. (1988). *Internal Combustion Engine Fundamentals*. McGraw-Hill.

Inoue, T., S. Matsushita, K. Nakanishi and H. Okano (1993). Toyota lean combustion system - the third generation system. In: *SAE Technical Paper Series*. number 930873.

Isermann, R. (1992). *Identifikation dynamischer Systeme*. Vol. 1. 2 ed.. Springer Verlag. Berlin.

Isermann, R., K.-H. Lachmann and D. Matko (1992). *Adaptive Control Systems*. Prentice Hall. Englewood Cliffs.

Isermann, R., S. Ernst and O. Nelles (1997). Identification with dynamic neural networks - architectures, comparisons, applications -. In: *IFAC Symposium on System Identification*. Fukuoka, Japan.

Matekunas, F. A. (1986). Engine combustion control with ignition timing by pressure ratio management. *U.S. Patent*.

Nelles, O. (1999). Nonlinear System Identification with Local Linear Neuro-Fuzzy Models. PhD thesis. Darmstadt University of Technology. Darmstadt, Germany.

Nelles, O. (2000). *Nonlinear System Identification*. Springer Verlag. Berlin, Heidelberg.

Nelles, O. and A. Fink (2000). Tool zur Optimierung von Rasterkennfeldern. *atp Automatisierungstechnische Praxis* **42**(5), 58–66.