MODELING, IDENTIFICATION AND CONTROL, 1999, VOL. 19, NO. 4, 185–206
doi:10.4173/mic.1998.4.2

A Software Environment for Gain Scheduled Controller Design

T. A. JOHANSEN*†, K. J. HUNT‡ and H. FRITZ§

Keywords: *Nonlinear control, gain scheduling, software*

Recent theoretical developments have improved the understanding of gain scheduled control and suggested new methods for design, analysis and implementation of such nonlinear control systems. An integrated software environment for gain scheduled local controller network design and analysis, including computer-aided modelling and system identification, is described. Some background theory is included, and a speed control design problem for an experimental vehicle illustrates the application of the approach.

1. Introduction

The main purpose of this paper is to describe a software environment for computer aided gain scheduled local controller network (LCN) design and analysis, including computer aided modelling and system identification. The software functionality is demonstrated with reference to the design of a high-performance longitudinal dynamics control system for an experimental vehicle. The design task involves first modelling the nonlinear dynamics of the system, and validation of the model. The nonlinear model is then used as the basis of controller design and verification. Finally, the controller is exported as a C-function for real-time implementation. All of these steps are carried out within our integrated software environment.

In addition to a description of the software, recent developments in the understanding, design and analysis of gain scheduled control (Hunt and Johansen 1997; Boyd *et al.* 1994; Driankov *et al.* 1996; Johansen *et al.* 1998) are summarized. The current state-of-the-art is compared to traditional gain scheduling design methods and tools (Shamma and Athans 1990; Rugh 1991).

A MATLAB-based integrated software environment for local model network (LMN) development, and gain scheduled LCN design, analysis and implementation is used in the application example. The LMN modelling tool is called ORBIT (Operating Regime Based modelling and Identification Toolkit), and the gain-scheduled LCN design tool is called ORBITcd (ORBIT Control Design toolkit).

An overview of the ORBIT software environment can be seen in Figure 1 and is completely described in Johansen (1997). ORBIT is implemented in MATLAB). The ORBIT core contains the graphical user interface (GUI), parameter and structure identification algorithms and model validation algorithms, model database manage-

Received 9 October 1997.

*Email: Tor.Arne.Johansen@itk.ntnu.no

†Current address: Department of Engineering Cybernetics, Norwegian University of Science and Technology, N-7034 Trondheim, Norway.

‡Daimler-Benz Research and Technology, Intelligent Systems Group (F3S/I), Alt-Moabit 96a, D-10559, Berlin, Germany. Email: hunt@DBresearch-berlin.de

§Daimler-Benz Research and Technology, Driver-assistance Systems Group (F1M/IA), D-70546 Stuttgart, Germany. Email: fritz@dbag.stg.DaimlerBenz.com

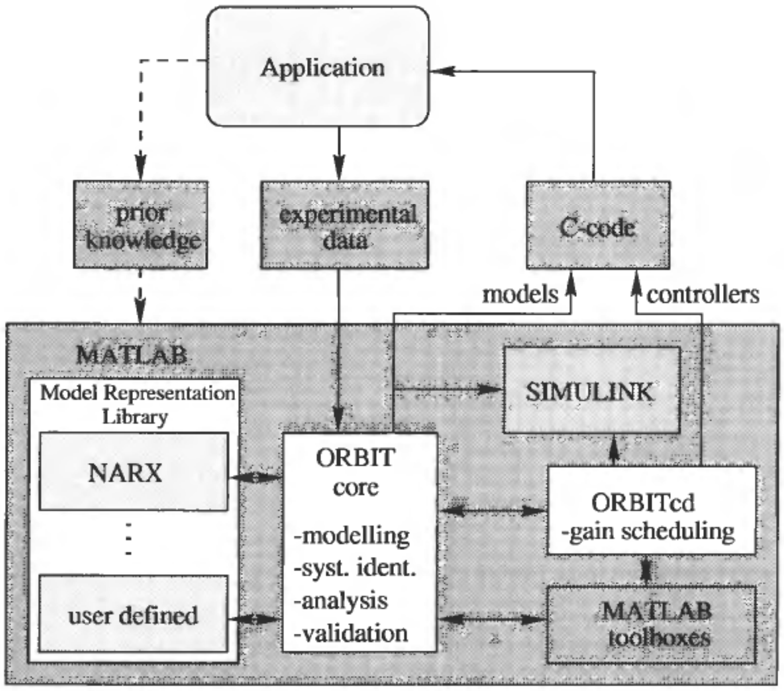


Figure 1. The ORBIT/ORBITcd software environment implemented in MATLAB.

ment, application programmers interface (API), as well as interfaces to various generic MATLAB tools and toolboxes. ORBIT can support a wide range of model representations, including NARX (Nonlinear Autoregressive with eXternal input), NARMAX (Nonlinear Autoregressive Moving Average with eXternal input) and nonlinear state-space models based on local models. Only the NARX representation (Johansen and Foss 1993) is currently implemented as part of the standard model representation library but the advanced user is free to include customized or generic model representations in this library by programming the required MATLAB functions. ORBIT models and ORBITcd controllers can be made available as SIMULINK S-functions and blocks for simulation. Using the built-in code generation facility, models and controllers can be exported as C-functions for real-time application or simulation. For real-time application of ORBIT models and controllers in languages other than C, one can develop MATLAB scripts that code the model parameters into the required programming language and format. Local model parameters can also be interchanged with other MATLAB tools, including the MATLAB Control Toolbox, Signal Processing Toolbox, and LMI Toolbox. An application programmers interface (API) allows other MATLAB programs to access the ORBIT model database. ORBIT is extendible, i.e. its core model representation and functions are documented.

The ORBIT Control Design toolkit supports design and analysis of gain-scheduled nonlinear controllers on the basis of ORBIT models. The theory behind these design and analysis methods will be outlined, and the tool itself will be described in more detail.

Experimental application data and prior knowledge form the basis of model development in ORBIT. These can be pre-processed and analyzed using generic MATLAB and SIMULINK functions before they are made use of in ORBIT.

Many aspects of the ORBIT/ORBITcd software functionality are illustrated using a speed control design example for an experimental vehicle [Hunt *et al.* 1997). Using measured data from the vehicle, nonlinear models are identified and validated. The models are used for nonlinear control design in ORBITcd, the design is verified by simulation, and finally the controller was exported as C-code, implemented, and tested in the vehicle. Typical experimental control results are shown.

2. Description of the Application

The ORBIT/ORBITcd software environment has been applied to a variety of applications. One such application is the problem of nonlinear dynamics modelling and controller design for an experimental vehicle. This problem concerns the longitudinal vehicle dynamics, and the goal is to design a high-performance speed controller. The software environment has been used in the complete design cycle for speed control: estimation of nonlinear models using measured data; extensive model validation; nonlinear controller design and verification; and, finally, automatic generation of the designed controller as a C-function which is directly embedded in the real-time control software in the vehicle. Experimental results are described fully in Hunt (1997).

The experimental vehicle is shown in Figure 2. It is a middle class 8-tonne Mercedes-Benz truck called OTTO (Optical Truck Transportation Optimization). More details of the vehicle and previous applications are given in (Franke *et al.* and Seeberger 1995; Hunt *et al.* 1996; Gehring and Fritz 1997). In this paper, it is used to exemplify the various aspects of the software functionality; a brief description of the problem is given in this section.

Figure 2. The experimental vehicle OTTO.

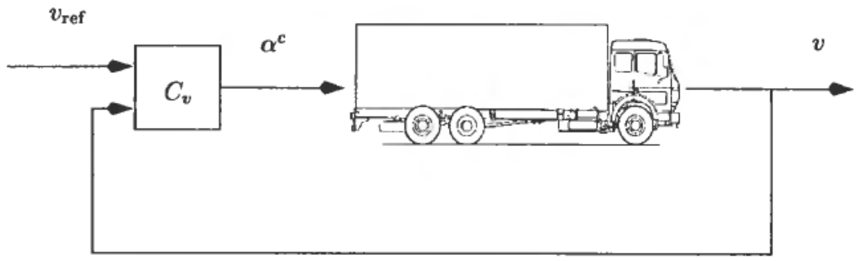


Figure 3. Structure of the speed-control loop. The vehicle speed is v, the desired speed is v_{ref}, and the throttle angle is α^c.

The structure of the speed controller is shown schematically in Figure 3. The vehicle's speed v is controlled by manipulation of the throttle angle α^c. A further possibility is to use the brakes as an additional control signal, but for simplicity the brakes are not considered in this presentation. The desired speed is denoted as v_{ref}. C_v is the nonlinear speed controller; it is designed on the basis of an empirically-derived model of the dynamic relationship between throttle angle α^c and speed v.

The plant's nonlinearities come from the various components in the drive-train. The engine produces a torque which is dependent upon the throttle angle and engine speed; this relationship is described by the nonlinear engine characteristics. The experimental vehicle is equipped with an automatic transmission. At low speeds a hydrodynamic torque converter is active. At higher speeds the torque converter is automatically bypassed by activation of a lock-up clutch. When the torque converter is not active, there is a rigid connection between the engine and the drive wheels. The torque transferred to the gearbox input depends nonlinearly upon the engine speed and the converter output speed. Finally, gear changes are dependent upon the throttle angle, vehicle speed and the currently engaged gear. The gearbox produces a hard-switching effect which results in step changes in the drive-train characteristics; it is very important to take account of this in modelling and in controller design.

These considerations lead to the conclusion that the key measurable variables which can be used to characterize the plant's nonlinearity are:

1. vehicle speed v;
2. throttle angle α^c;
3. gear, denoted as g.

It will be crucial to utilize this information in the construction of models and in the design of controllers.

The main design goals and engineering constraints for this problem can be summarized as follows:

1. the closed-loop system should give suitably fast disturbance rejection (road inline, rolling resistance, model error and wind resistance are in the main of low-frequency character);
2. the design should be insensitive to unmodelled dynamics and measurement inaccuracies. These uncertainties are primarily high-frequency, resulting from the throttle actuator (which has a dead-time and a fast underdamped response) and the limited-precision speed sensor;
3. the closed-loop should have a pre-specified command response;
4. the closed-loop properties should be consistent over a wide operational envelope (i.e. from low speed in first gear to high speed in top gear) despite the strong system nonlinearities.

In order to meet these demands, it is clear that a high-precision nonlinear model is required.

For identification of the plant dynamics in each gear, a number of test inputs $\alpha^c(t)$ were applied and the resulting speed was measured. Since the system is strongly nonlinear, a goal in the design of the test inputs was to 'cover' the operational range of the plant to as great an extent as possible. Thus, it is important to investigate both the large-signal and small-signal behaviour of the system. The main input signal types applied were:

1. PRBS-signals (Pseudo-random binary sequence) of relatively low magnitude (typically 10%) were applied around a number of equilibria in each gear;

2. in each gear, large steps of varying magnitude (both positive and negative) with a superimposed low-amplitude PRBS excitation were applied. Before application of these signals the system was driven manually to various equilibria.

3. Local Controller Networks

A local controller network (LCN) (Hunt and Johansen 1997) consists of a number of linear local controllers, each designed to achieve the desired performance in a particular operating regime of the nonlinear system. Figure 4 illustrates the operating regimes chosen for the vehicle model M42 and corresponding controller. These regimes are characterized by throttle, speed and gear.[1] In order to achieve stable and robust performance also during major transients, the LCN consists not only of linearizations about equilibria, but also linearizations about transient states that may be on the system trajectory. This particular aspect is discussed in detail in (Johansen *et al.* 1998). In Figure 4 the equilibrium manifold for each gear is represented by a solid line. The regimes which are associated with off-equilibrium operation of the vehicle can be easily observed.

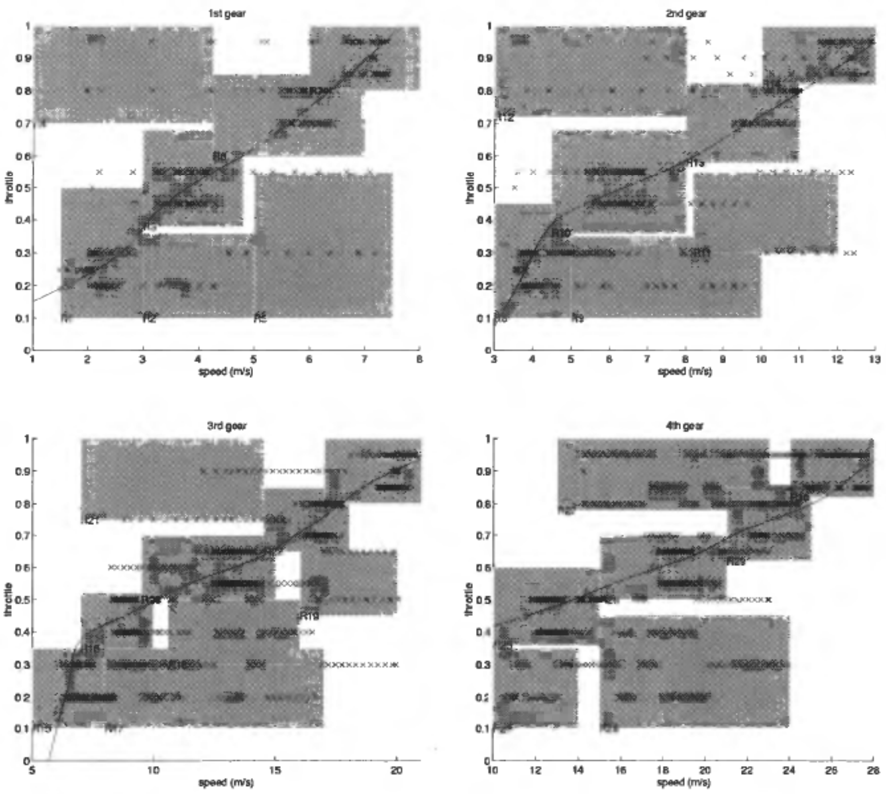


Figure 4. Operating regimes for the model M42 of the experimental vehicle. The regimes (shaded areas) are characterized by throttle, speed and gear. The marked points correspond to experimental data, and the solid lines are the estimated equilibrium manifolds for each gear.

[1]Note that for first-order systems described by $v(t+1) = f(v(t), \alpha^c(t))$, this plane completely describes the plant's operational range for a given gear. In the first-order case, therefore, this plane must be sufficiently covered by measurement points. In practice, both first and second-order models were used in system identification.

Notice that the operating regimes are typically not regions with hard boundaries, as illustrated in the figure, but rather regions with soft boundaries (or fuzzy sets). This means that there will be a smooth transition between local controllers when moving between regimes. This is viewed and implemented as interpolation in the parameter space of the controller structure.

This section outlines the essentials of LCN design: operating regime decomposition, development of linear local models to be used for the design, local control design algorithms, and analysis of the local and global stability properties of the closed loop.

3.1. *Local Model Networks*

Design of linear local controllers is based on linear local models valid for each operating regime. These can be individual linear models identified on the basis of specific experiments designed for each operating regime or experiments that cover several operating conditions simultaneously, or they can be linearizations about different operating points of a nonlinear model. The validity of each local model is typically restricted to its corresponding operating regime. A particular nonlinear model representation that is explicitly parameterized in terms of local linear models is the local model network (LMN). An LMN consists typically of a set of linear models that are combined by weighting functions to form a global nonlinear model. Such a model representation is especially useful for LCN design for two reasons. First, the local models are directly available. Second, and more important, the LMN's weighting functions, are applicable as weighting functions for gain scheduling in the LCN, since they are designed to capture the nonlinearities of the system, which is a key issue in gain-scheduled control (Shamma and Athans 1990; Rugh 1991). Development of LMNs and their properties are extensively studied in e.g. (Johansen and Foss 1993; Murray-Smith and Johansen 1997).

Here we restrict our attention to single-input single-output NARX models. This representation relates a control input $u(t)$ and an auxiliary input $\alpha(t)$ to an output $y(t)$ according to

$$y(t) = f(\psi(t), \alpha(t)) + e(t) \tag{1}$$

$$\psi(t) = (y(t-1), \ldots, y(t-n_y), u(t-1), \ldots, u(t-n_u))^T \tag{2}$$

where ψ is the information vector. The NARX model can be composed of a number of local models, see Johansen and Foss (1993):

$$y(t) = \sum_{i=1}^{N} f_i(\psi(t))w_i(z(t)) + e(t) \tag{3}$$

$$z(t) = H(y, u, \alpha)(t) \tag{4}$$

where the auxiliary variables $\alpha(t)$ are only used for scheduling, and $e(t)$ represents the unmodelled dynamics, disturbances and noise. A block diagram for this model is provided in Figure 5. The elements of this model representation are

- The functions $f_1, \ldots, f_N$ define a set of local models. These can in general be arbitrary, but in the LCN context linear functions are definitely most useful, corresponding to local ARX models:

$$f_i(\cdot) = d_i - a_{i,1}y(t-1) - \ldots - a_{i,n_y}y(t-n_y) + b_{i,1}u(t-1) + \ldots + b_{i,n_u}u(t-n_u). \tag{5}$$

- The integer parameters n_y and n_u define the order of the NARX model.
- The positive semi-definite weighting functions $w_1, \ldots, w_N$ will define the relative

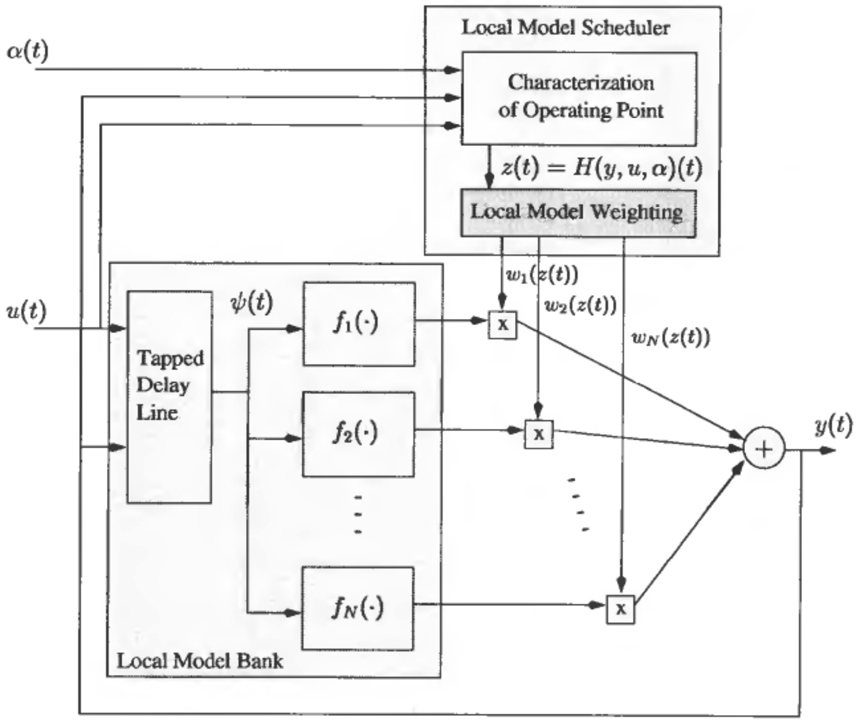


Figure 5. The LMN NARX model representation. It consists of a Local Model Bank and a Local Model Scheduler that assigns weights to each local model as a function of the operating point. The global model output is the result of weighting the local model outputs $f_i(\psi(t))$, where the information vector $\psi(t)$ contains delayed inputs and outputs.

weight of the local models at each operating point z. In practice, these functions will characterize the model's N operating regimes and satisfy

$$\sum_{i=1}^{N} w_i(z) = 1$$

for all z. The representation of these functions is described below.

- The variables that characterize the operating regimes, z, are defined by a general nonlinear operator H.

Axis-parallel hyper-rectangles with soft edges have proven to give a sufficiently flexible parameterization of the operating regimes and weighting functions. The flexibility is partially due to the user's freedom to select operator H that defines variables z that characterize the operating regimes. The operating regimes can be viewed as fuzzy sets which are characterized by their membership functions μ_i. For example, a d-dimensional axis-parallel rectangle with soft edges can be represented in terms of its projections $\mu_{i,j}$ onto the d axes as

$$\mu_i(z) = \prod_{j=1}^{d} \mu_{i,j}(z_j). \tag{6}$$

The weighting functions are now defined by Takagi and Sugeno (1985)

$$w_i(z) = \frac{\mu_i(z)}{\sum_{j=1}^{N} \mu_j(z)}.$$

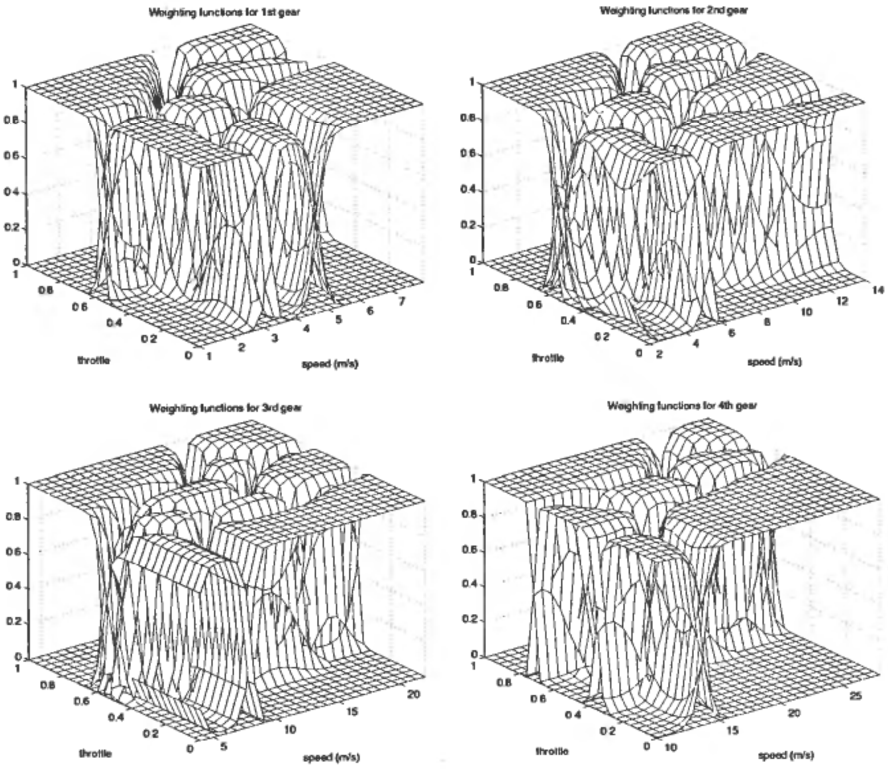


Figure 6. Weighting functions used in the local model and controller interpolation for the operating regimes in M42 of the vehicle, c.f. figure 4. The z-axes in these plots show the values of $w_i(z)$, where $z = (v, \alpha^c)^T$ in this application.

Figure 6 illustrates the weighting functions corresponding to the operating regimes of the experimental vehicle, c.f. Figure 4. We observe that they give a smooth blending of the local models and controllers across the regime boundaries.

3.2. *Local Control Design Algorithms*

The objective of the control system is to make the output $y(t)$ track a reference trajectory $r(t)$ while rejecting the disturbances $e(t)$. To this end, assume that the control signal u consists of a feed-forward component u_{ff} and a feedback component u_{fb},

$$u(t) = u_{ff}(t) + u_{fb}(t). \tag{7}$$

The purpose of the feed-forward is to generate in real time a trajectory $u_{ff}(t)$ such that $y(t)$ tracks $r(t)$, nominally. The feedback will compensate for uncertainties, such as modelling error and disturbances.

Consider for the moment a single linear model of the following form, c.f. (5):

$$A_i(q^{-1})y(t) = B_i(q^{-1})u(t) + d_i + e(t), \tag{8}$$

where A_i and B_i are polynomials in the delay operator

$$A_i(q^{-1}) = 1 + a_{i,1}q^{-1} + \ldots + a_{i,n_y}q^{-n_y}$$

$$B_i(q^{-1}) = b_{i,1}q^{-1} + \ldots + b_{i,n_u}q^{-n_u}.$$

In general, any linear control design method can be applied to design the local controllers in the gain scheduled LCN framework. This section describes the implemented local control design algorithms (they are described in more detail in Hunt and Johansen (1997)).

The local linear controller is a quite general two-degrees-of-freedom controller (Åström and Wittenmark 1997)

$$u(t) = \frac{1}{T_i(q^{-1})}(S_i(q^{-1})r(t) - R_i(q^{-1})y(t)). \tag{9}$$

Here, S_i, R_i and T_i are polynomials in the delay operator:

$$S_i(q^{-1}) = s_{i,0} + s_{i,1}q^{-1} + \ldots + s_{i,n_s}q^{-n_s}, \tag{10}$$

$$R_i(q^{-1}) = r_{i,0} + r_{i,1}q^{-1} + \ldots + r_{i,n_r}q^{-n_r}, \tag{11}$$

$$T_i(q^{-1}) = 1 + t_{i,1}q^{-1} + \ldots + t_{i,n_t}q^{-n_t}. \tag{12}$$

Combining equations (8) and (9), the closed-loop characteristic polynomial is easily found to be $A_iT_i + B_iR_i$. For a given desired closed-loop characteristic the following equation is therefore solved:

$$A_i(q^{-1})T_i(q^{-1}) + B_i(q^{-1})R_i(q^{-1}) = A_{i,0}(q^{-1})A_{i,m}(q^{-1}). \tag{13}$$

Here, $A_{i,0}$ corresponds to a desired observer polynomial and $A_{i,m}$ to the remaining desired closed-loop poles. The S_i polynomial in (9) is chosen to achieve a servo response: $S_i(q^{-1}) = \lambda_i A_{i,0}(q^{-1})$ where $\lambda_i = A_{i,m}(1)/B_i(1)$. These definitions result in local closed-loop transfer functions

$$y(t) = \lambda_i \frac{B_i(q^{-1})}{A_{i,m}(q^{-1})}r(t) + \frac{T_i(q^{-1})}{A_{i,o}(q^{-1})A_{i,m}(q^{-1})}e(t) + \frac{T_i(q^{-1})}{A_{i,o}(q^{-1})A_{i,m}(q^{-1})}d_i. \tag{14}$$

In the application example, a sample indirect method is used for selection of the design polynomials $A_{i,o}$ and $A_{i,m}$ (Åström and Wittenmark 1997). In each case, the rise-time t_r and damping factor ζ of a desired continuous second-order linear system

$$\frac{\omega_n^2}{s^2 + 2\omega_n\zeta s + \omega_n^2} \tag{15}$$

are specified. The natural frequency ω_n can be shown from a simple time-domain analysis of the 2nd order system (15) to be related to the rise-time t_r through $\omega_n \approx 3{\cdot}2/t_r$. The system (15) is discretized, and its poles are used to define $A_{i,o}$ or $A_{i,m}$. Typically, the rise-time for $A_{i,m}$ is about 4–5 times greater than the observer rise-time. In the vehicle application, the rise-times for $A_{i,m}$ lie in the range 4–10s, and those for $A_{i,o}$ in the range 0–2s. The purpose of the observer polynomial is to ensure robustness against unmodelled high-frequency dynamics by introducing sufficient damping in the loop transfer function at high frequencies. The feedback loop contains integral action, i.e. $T_i(q^{-1})$ contains a factor $1 - q^{-1}$. From (14) we observe that this eliminates steady-state error due to disturbances, modelling error and inaccurate knowledge of the offset d_i.

As an alternative to direct specification of rise-time and damping, consider the local LQG cost function

$$J_i = \varepsilon[y^2(t) + \mu_i u^2(t)] \tag{16}$$

where ε denotes the expectation operator and μ_i is a local control weighting. The optimal local controller polynomials T_i, R_i are solutions to the linear equation

$$A_i(q^{-1})T_i(q^{-1}) + B_i(q^{-1})R_i(q^{-1}) = F_i(q^{-1}). \tag{17}$$

Here, F_i is the stable solution to the spectral factorization

$$F_i(q^{-1})F_i^*(q^{-1}) = B_i(q^{-1})B_i^*(q^{-1}) + \mu_i A_i(q^{-1})A_i^*(q^{-1}), \tag{18}$$

with the adjoint of a polynomial defined through $X_i^*(q^{-1}) = X_i(q)$. Note that this formulation of the problem assumed that A_i and B_i have no common factors and this ensures the existence of stable spectral factors. For details of the more general solution and other technical conditions refer to the texts (Kucera 1989; Hunt 1989, 1993). An observer polynomial can also be included in the problem formulation, either by optimal design (where the measurement noise properties are specified) or using the informal method outlined above. The remaining controller polynomials S_i (which act on the reference signal r) are designed to achieve desirable command tracking properties, but in the simplest case are

$$S_i(q^{-1}) = \lambda_i = F_i(1)/B_i(1). \tag{19}$$

The LCN is a gain scheduled interpolation of a set of linear compensators (9):

$$\sum_{i=1}^{N} T_i(q^{-1})w_i(z(t))u(t) = \sum_{i=1}^{N} (S_i(q^{-1})w_i(z(t))r(t) - R_i(q^{-1})w_i(z(t))y(t)) \tag{20}$$

This control structure can be interpreted as a quasi-linear controller with operating-point-dependent parameters, see also Figure 7.

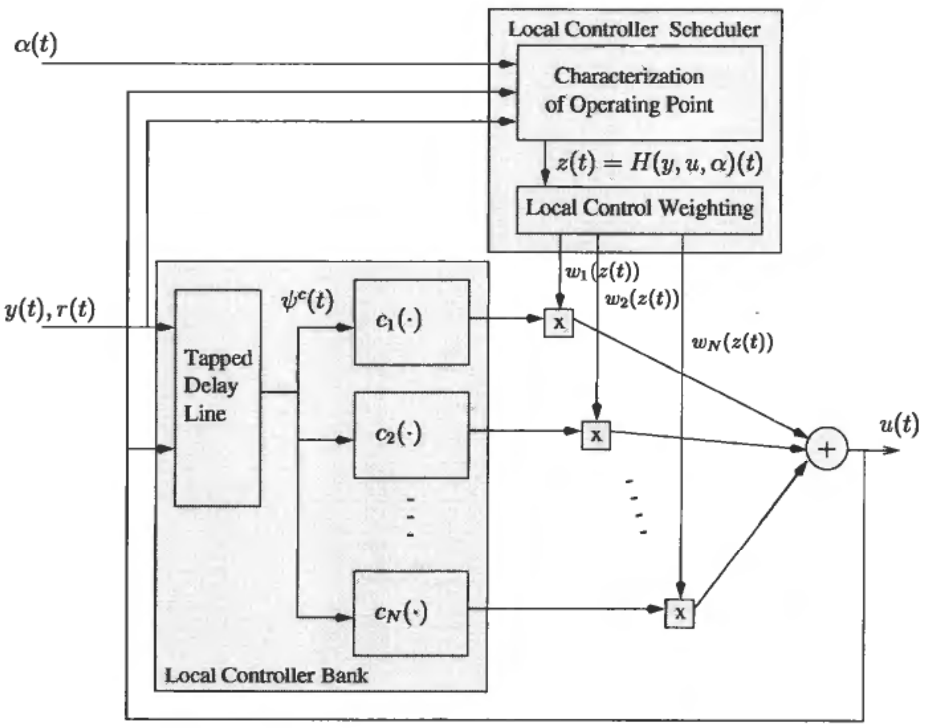


Figure 7. The LCN representation. It consists of a Local Controller Bank and a Local Controller Scheduler that assigns weights to each local controller. Here, the local controllers' outputs are denoted as $c_i(\psi^c(t))$, and are designed on the basis of the respective local models $f_i(\cdot)$. The controller information vector $\psi^c(t)$ contains delayed values of the reference signal and the plant's input and output signals.

3.3. *Local and Global Properties*

The stability, performance and robustness properties of each linear local controller are well understood and can be analyzed using standard tools such a Bode and Nyquist plot, for each *fixed* operating point. This is a highly useful property of the gain scheduling control design philosophy. However, it is also well known that these local properties does not necessarily lead to guaranteed global properties (Rugh 1991 and Hunt 1993). Global properties also cannot be guaranteed if proper modifications are not made when implementing the gain scheduled controller (Kaminer *et al.* 1995; Leith and Leithead 1996). For example, global stability does not in general follow from stability of the separate local linear closed loops when the controllers are scheduled on the state, input or output of the system. The traditional approach has been to schedule only on slowly time-varying variables or external variables such as the reference signal, which of course restricts the applicability of gain scheduling control (Shamma and Athans 1990; Hunt and Johansen 1997).

In the following we will briefly review some conditions for the existence of a quadratic Lyapunov function for the nonlinear gain scheduled closed loop system. For convenience, we approximate both the system and the LCN controller with continuous-time state-space systems, for which the NARX LMN is approximated by

$$\dot{x}_s = \sum_{i=1}^{N} (A_i^s x_s + B_i^s u + d_i^s) w_i(z) \tag{21}$$

$$y = D^s x_s \tag{22}$$

where the parameters of the matrices can be defined from the local ARX model parameters. Likewise, the LCN is approximated by

$$\dot{x}_c = \sum_{i=1}^{N} (A_i^c x_c + B_i^c y + C_i^c r + d_i^c) w_i(z) \tag{23}$$

$$u = \sum_{i=1}^{N} (D_i^c x_c + E_i^c y + F_i^c r + g_i^c) w_i(z). \tag{24}$$

When the system and controller state-space representations are combined, we get the following closed loop system

$$\begin{pmatrix} \dot{x}_s \\ \dot{x}_c \end{pmatrix} = \left(\sum_{i=1}^{N} \sum_{j=1}^{N} A_{ij} w_i(z) w_j(z) \right) \begin{pmatrix} x_s \\ x_c \end{pmatrix} + \dots$$

where

$$A_{ij} = \begin{pmatrix} A_i^s + B_i^s E_j^c D^s & B_i^s D_j^c \\ B_j^c D^s & A_j^c \end{pmatrix}.$$

The stability properties of the closed loop are now determined by the A_{ij} matrices. In particular, there exists a quadratic Lyapunov function $V(x) = x^T P x$, where $x^T = (x_s^T, x_c^T)$, provided there exists a positive definite matrix $Q = Q^T$ that satisfies the set of linear inequalities (LMIs)

$$A_{ij} Q + Q A_{ij}^T < 0 \tag{25}$$

for all i, j that satisfy $w_i w_j \neq 0$ and $Q = P^{-1}$ (Boyd *et al.* 1994). In other words, the inequalities (25) give sufficient conditions for asymptotic stability of the nonlinear closed loop, independent of the rate of change of the operating point. However, since only the class of quadratic Lyapunov functions is considered, the conditions (25) are sufficient but not necessary. This restricts the practical usefulness of this tool significantly and with the application example we have encountered situations when simulations or experiments "prove" stability of a nonlinear control design for the experimental vehicle, although it was not possible to find a quadratic Lyapunov function for the nominal closed loop.

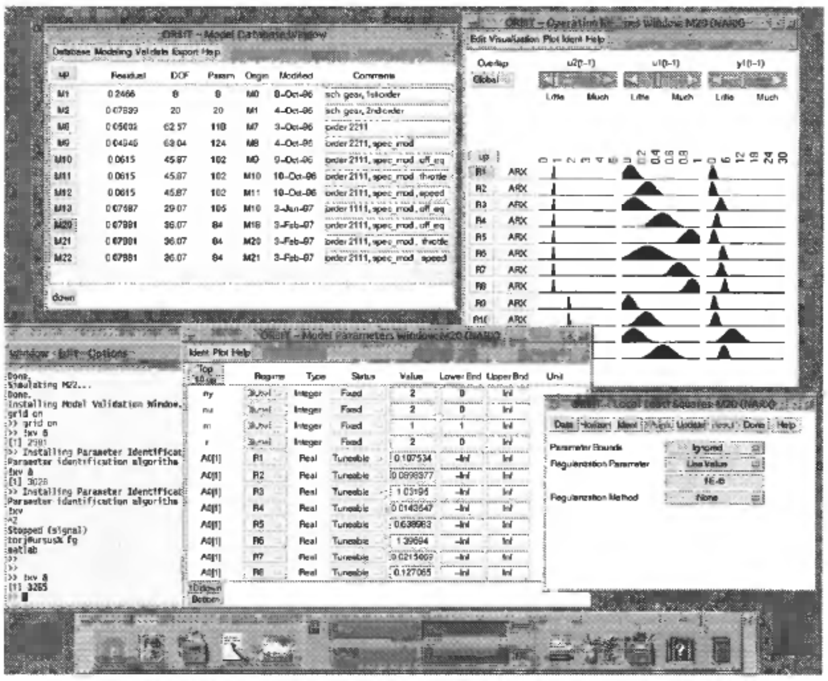
3.4. *Links to Fuzzy Control*

The LCN operating regimes can be viewed as fuzzy sets, and the weighting functions w_i interpolating the controller parameters are functionally equivalent to Takagi-Sugeno-Kang (TSK) fuzzy inference systems (Takagi and Sugeno 1985; Hunt, Haas and Murray-Smith May 1996; Hunt, Haas and Brown June 1995). It is evident that the LCN approach has a very close resemblance to some TSK fuzzy model-based control schemes such as those described in (Takagi and Sugeno 1985; Zhao, Gorez and Wertz 1997; Wang, Tanaka and Griffin 1996). A major difference compared with the above mentioned work, which is restricted to static state feedback and linearization only at equilibria, is that the present work applies dynamic output feedback and opens the possibility for off-equilibrium linearization and control design.

4. **ORBIT Functionality and User Interface**

The ORBIT environment for LMN development and identification consists of a number of different windows. Figure 8:

- The Model Database Window (upper left window of the screen-dump in Figure 8) contains a list of ORBIT models with summarized information about each model. Functions for database management, including storing and recalling model databases, and for generating external representations (such as C-code and SIMULINK S-functions) can be invoked here.
- The Operating Regimes Window (upper right window of the screen-dump in Figure 8) allows the user to inspect and manipulate operating regimes. The operating regimes can be visualized as 2D rectangles, or by their membership function $\mu_{i,j}$ along each characteristic variable axis, in this case gear, throttle and speed, respectively. The regimes' characteristic variables, their position, amount of overlap etc. can easily be manipulated, and regimes can be added and removed. The selected model in Figure 8 is M20 (which is a previous version of the currently best model M42).
- The Model Parameters Window (centre window of the screen-dump in Figure 8) contains a list of all model parameters and their properties. This includes structural and functional parameters, in addition to local model parameters. Parameter identification and visualization tools can be invoked from this window.
- There are various windows for parameter and structure identification. For example, the lower right window in the screen-dump in Figure 8 contains the parameters and functionality of the Locally Weighted Least Squares parameter identification algorithm. The functionality available in these windows will be described in detail in sections 4.1 and 4.2.
- There are also several windows for model validation and comparison, see section 4.3.
- The MATLAB Command Window (lower left window in the screen-dump in Figure 8) is applied to display messages and information. In addition, it allows the user to interact with ORBIT through the API.



MP	Residual	DOF	Param	Origin	Modified	Comments
M1	0.2466	8	8	M0	8-Oct-96	sch gear, 1storder
M2	0.07839	20	20	M1	4-Oct-96	sch gear, 2ndorder
M8	0.05003	62.57	110	M7	3-Oct-96	order 2211
M9	0.04946	63.04	124	M8	4-Oct-96	order 2211, spec_mod
M10	0.0615	45.87	102	M0	9-Oct-96	order 2111, spec_mod, off_eq
M11	0.0615	45.87	102	M10	10-Oct-96	order 2111, spec_mod, throttle
M12	0.0615	45.87	102	M11	10-Oct-96	order 2111, spec_mod, speed
M13	0.07687	29.07	105	M10	3-Jan-97	order 1111, spec_mod, off_eq
M20	0.07881	36.07	84	M18	3-Feb-97	order 2111, spec_mod, off_eq
M21	0.07881	36.07	84	M20	3-Feb-97	order 2111, spec_mod, throttle
M22	0.07881	36.07	84	M21	3-Feb-97	order 2111, spec_mod, speed

```
Done.
Simulating M22...
Done.
Installing Model Validation Window.
grid on
>> grid on
>> !xv &
[1] 2981
>> Installing Parameter Identificat
Parameter identification algorithm
!xv &
[1] 3028
>> Installing Parameter Identificat
Parameter identification algorithm
!xv
^Z
Stopped (signal)
torj@ursus% fg
matlab
>>
>>
>> !xv &
[1] 3265
>>
```

	Regime	Type	Status	Value	Lower Bnd	Upper Bnd	Unit
ny	Global	Integer	Fixed	2	0	Inf	
nu	Global	Integer	Fixed	2	0	Inf	
m	Global	Integer	Fixed	1	1	Inf	
r	Global	Integer	Fixed	2	0	Inf	
A0[1]	R1	Real	Tuneable	0.107534	-Inf	Inf	
A0[1]	R2	Real	Tuneable	0.0898377	-Inf	Inf	
A0[1]	R3	Real	Tuneable	1.03196	-Inf	Inf	
A0[1]	R4	Real	Tuneable	0.0143647	-Inf	Inf	
A0[1]	R5	Real	Tuneable	0.638963	-Inf	Inf	
A0[1]	R6	Real	Tuneable	1.39694	-Inf	Inf	
A0[1]	R7	Real	Tuneable	0.0215909	-Inf	Inf	
A0[1]	R8	Real	Tuneable	0.127065	-Inf	Inf	

Figure 8. Modelling with ORBIT. The Model Database Window (upper left) contains a database of ORBIT models together with some information about each model. The Operating Regimes Window (upper right) contains a graphical representation by the operating regimes of the current model (M20). The regimes are represented by the projection of their characterizing (fuzzy) hypercube onto each characteristic axis (shaded dumps). Furthermore, functions for regime manipulation and structure identification are available in this window. The Model Parameters Window (lower centre) contains a list of all the parameters of the current model and their properties. Parameter identification algorithms can be invoked from this window. The Parameter Identification Window (lower right) contains the parameters of a parameter identification algorithm and provides functionality for parameter identification. The MATLAB Command Window (lower left) displays status information and allow interaction with the full MATLAB environment.

4.1. *Parameter Identification Methods*

The parameter identification problem consists of determining the local model parameters. With local linear ARX models, the one-step-ahead predictors based on each local ARX model are linearly parameterized. Furthermore, since the weighting functions do not contain any parameters to identify, the global (interpolated) one-step-ahead predictor can also be easily seen to be linearly parameterized (Johansen and Foss 1993).

ORBIT contains three basic parameter identification algorithms:

- The **prediction error method** is the most general method where the PE criterion

$$V_M(\theta) = \sum_{t=1}^{M} l(\varepsilon(t;\theta)) \tag{26}$$

is minimized using an SQP (sequential quadratic programming) algorithm. The prediction error is defined as

$$\varepsilon(t;\theta) = y(t) - \hat{y}(t|t-h;\theta) \tag{27}$$

where $\hat{y}(t|t-h;\theta)$ is the h-step-ahead predictor based on the global interpolated model and θ are the model parameters. The PE criterion is defined in terms of a general user-specified penalty on the prediction error $l(\cdot)$ (not necessarily quadratic) and user-specified prediction horizon h. Initial values for the SQP algorithm can be specified.

- **Least squares method.** For the special case when the criterion function is quadratic and a one-step-ahead predictor is applied ($h=1$), the criterion (26) is quadratic and a least squares solution can be found.
- Rather than defining the identification problem in terms of a single global predictor $\hat{y}(t|t-h;\theta)$ that combines the local models, one can define a separate identification problem for each local model based on local predictors $\hat{y}_i(t|t-h;\theta_i)$ where θ_i are the local model parameters for the local model with index i. Of course, only the data that are relevant in the current operating regime should be applied to identify the corresponding local model. This can be implemented as a **locally weighted least squares method** when the criterion function is quadratic, i.e.

$$V_{M,i}(\theta_i)=\sum_{t=1}^{M}(y(t)-\hat{y}_i(t|t-1;\theta_i))^2 w_i(z(t)) \tag{28}$$

is minimized. This method has some interesting properties that are studied in detail in Murray-Smith and Johansen (1995, 1997). In contrast to the above global methods, this method ensures that the local linear models are approximations to the linearized system at the appropriate operating points. This is a crucial assumption in LCN and this approach is therefore particularly suited to identification of local models that are used for LCN design.

All identification methods allow the user to specify hard or soft constraints on each parameter separately. Furthermore, regularization methods are implemented to improve the solution of ill-conditioned parameter identification problems (Johansen and Foss 1997; Johansen 1997).

When examining the experimental data for the test vehicle, it was discovered that the local models in several of the transient operating regimes ae not easily identifiable. The lack of identifiability is caused by the fast transients in off-equilibrium regimes. Some parameters of these local models were therefore constrained during identification, using information from an analysis of step-responses. It was also observed that the dynamics in 1st gear are 2nd order, while in higher gears they are essentially 1st order. This was expected, since the automatic transmission contains a lockup clutch that is effective only at high speeds (switching occurs typically at some speed in the 2nd gear). This lockup clutch fixes the ratio between the wheel and engine speeds at a fixed gear. Consequently, these two states become a single state.

The nonlinearity of the model can be seen in Figure 9 where the impulse responses of each local ARX model are illustrated. There are significant differences between each gear, and also within each gear. We observe that there are significant changes in response-time and gain over the operating range, reflecting the nonlinear torque curve of the engine: the response is faster with higher high-frequency gain at intermediate throttle than at very high and low throttle. Also, due to "saturation" in the engine, the response is faster and with higher high-frequency gain near equilibria than far away from equilibria. Finally, one can observe the under-damped 2nd order dynamics at some operating conditions in 1st gear.

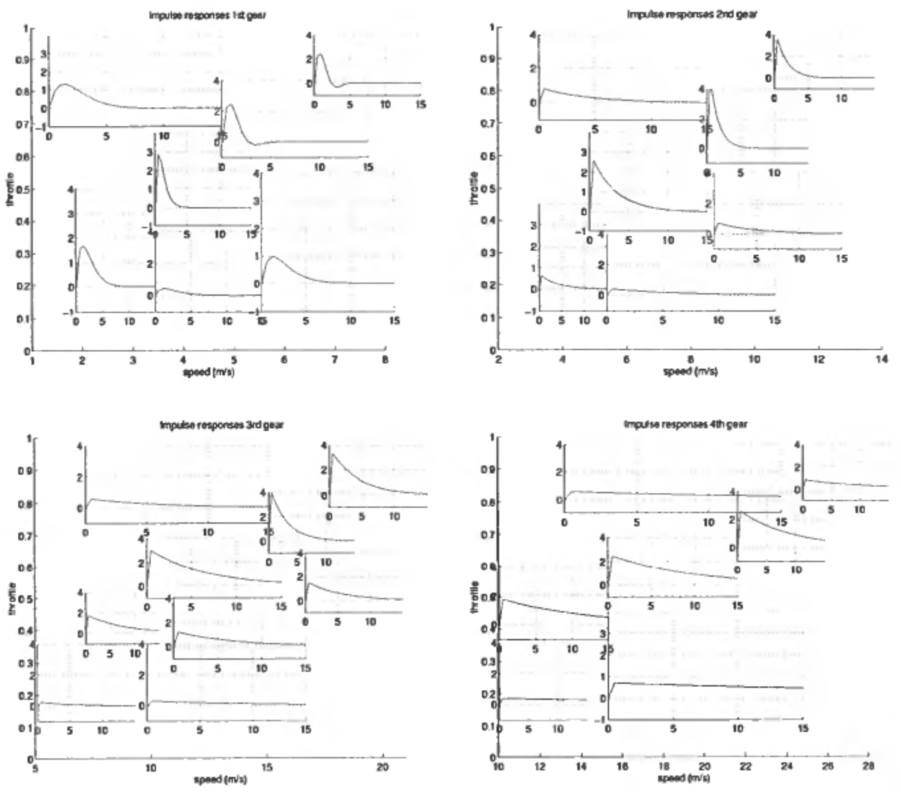


Figure 9. Impulse responses of the local linear ARX models with each operating regime for the model M42 of the experimental vehicle. The plot of each impulse response is placed in its corresponding operating regime, and the axes are time (s) and speed (m/s).

4.2. *Structure Identification Methods*

The structural parameters of ORBIT models are

- The number of operating regimes, and their location in the operating space.
- Any integer parameters in the local models, such as order (n_u and n_y in the NARX representation).

ORBIT supports structural identification of these parameters on the basis of optimization of statistical criteria based on separate validation data, Final Prediction Error (FPE) or Minimum Description Length (MDL) that all estimate the mean squared prediction error. The set of model structures defined by the possible operating regime decompositions is viewed as a tree, Figure 10. At each node in this tree there can in addition be integer parameters related to the local models. ORBIT allows the user to interactively explore the model structure tree in Figure 10. User-specified sub-trees can be searched to the user-specified depth. The heuristic search method is described in detail in Johansen and Foss (1995) and is based on Sugeno and Kang (1988). Working interactively, the user may keep promising models and validate and compare them using other methods, and as simulation and residual analysis. The user can also manipulate the operating regimes directly in the GUI.

Structure search were used in the early stages of the vehicle modelling, using a less informative experimental data set Hunt *et al.* 1996). The operating regimes of

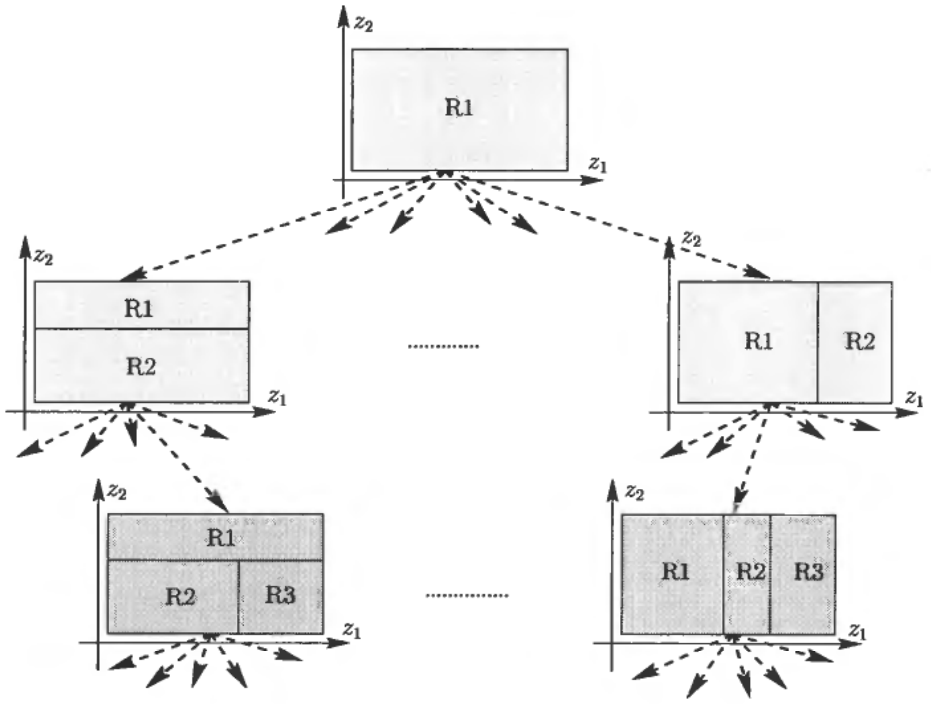


Figure 10. The model structure tree resulting from successive operating regime decomposition (in the picture the operating regimes are characterized by two variables z_1 and z_2). For example, the 2nd level in the tree corresponds to possible decompositions into two regimes, while the 3rd level corresponds to possible decompositions into three regimes etc.

Figure 4 were specified manually on the basis of these experiences and careful validation, analysis and simulation, taking into account the importance of having accurate models both at equilibrium and off-equilibrium regimes for the purpose of LCN design. The ORBIT API was exploited to write application specific MATLAB scripts that simplified iterative refinement of the model structure.

4.3. *Model Validation Methods*

Model validation is often viewed as a highly application-specific problem. This is recognised in ORBIT, and the local models and the global model can be made available to external analysis and simulation in the MATLAB/SIMULINK environment. However, there are some commonly used validation methods that are supported by ORBIT:

- The models can be compared by examining simulation results. The user can choose between simulation and prediction (arbitrary horizon), and define arbitrary external input signals for the simulation.
- Equilibrium data for the models can be computed by ORBIT, i.e. given inputs u^* and α^* the corresponding equilibrium output y^* can be computed according to

$$y^* = f(y^*, \ldots, y^*, u^*, \ldots, u^*, \alpha^*) \qquad (29)$$

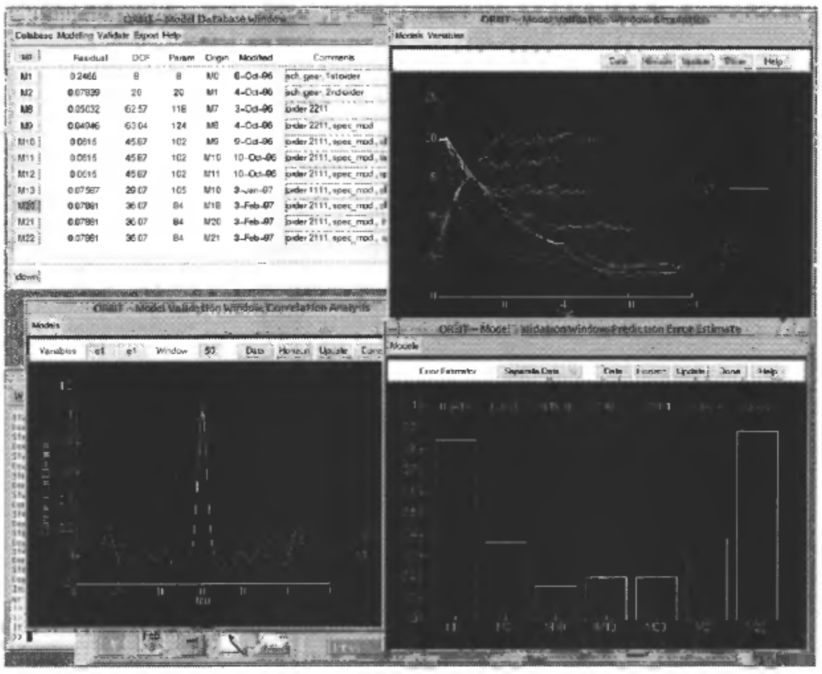
for a range $u^* \in U$ and $\alpha^* \in A$, cf. (1).

- Stastical measures of expected prediction performance such as FPE (Ljung 1987) which is based on the residuals

$$\mathrm{FPE} = \frac{1 + d/M}{1 - d/M} \sum_{t=1}^{M} \varepsilon^2(t) \tag{30}$$

  or the use of separate validation data can be visualized as error bar estimates. In (30) d is the model's degrees of freedom while the residual $\varepsilon(t)$ may correspond to an arbitrary prediction horizon h.
- Correlation testing can be applied to various model variables (such as inputs $u(t)$, residuals $\varepsilon(t)$, and outputs $y(t)$); the correlation test results can be visualized and analyzed.

Some validation results for various vehicle models can be seen in Figure 11. In the "Correlation Analysis Window" (lower left window of the screen-dump in Figure 11) the estimated autocorrelation function for the residuals $\varepsilon(t)$ with models M20, M21 and M22 are shown. M21 and M22 are simplified models with fewer operating regimes. While M20 is scheduled on throttle and speed, M21 is scheduled on speed and M22 on throttle only, in addition to gear. Comparative simulation results with the models M20 and M22 together with selected experimental data S are displayed in the "Simulation Window" (upper right window in the screen-dump in Figure 11). In the "Prediction Error Estimate Window" (lower right window of the screen-dump in Figure


ORBIT – Model Database window
Database Modeling Validate Export Help
ORBIT – Model Validation Window, Simulation
Models Variables
Data Simulate Update Zoom Help
ORBIT – Model Validation Window, Correlation Analysis
Models
Variables e1 e1 Window 50 Data Horizon Update Close
ORBIT – Model Validation Window, Prediction Error Estimate
Models
Error Estimator Separate Data Data Horizon Update Done Help
down

NR	Residual	DOF	Param	Origin	Modified	Comments
M1	0.2466	8	8	M0	6-Oct-96	sch. gear, 1st order
M2	0.07839	20	20	M1	4-Oct-96	sch. gear, 2nd order
M8	0.05032	62.57	118	M7	3-Oct-96	order 2211
M9	0.04046	63.04	124	M8	4-Oct-96	order 2211, spec_mod
M10	0.0615	45.67	102	M9	9-Oct-96	order 2111, spec_mod,
M11	0.0615	45.67	102	M10	10-Oct-96	order 2111, spec_mod,
M12	0.0615	45.67	102	M11	10-Oct-96	order 2111, spec_mod,
M13	0.07587	29.07	105	M10	3-Jan-97	order 1111, spec_mod,
M20	0.07861	36.07	84	M18	3-Feb-97	order 2111, spec_mod,
M21	0.07861	36.07	84	M20	3-Feb-97	order 2111, spec_mod,
M22	0.07861	36.07	84	M21	3-Feb-97	order 2111, spec_mod,

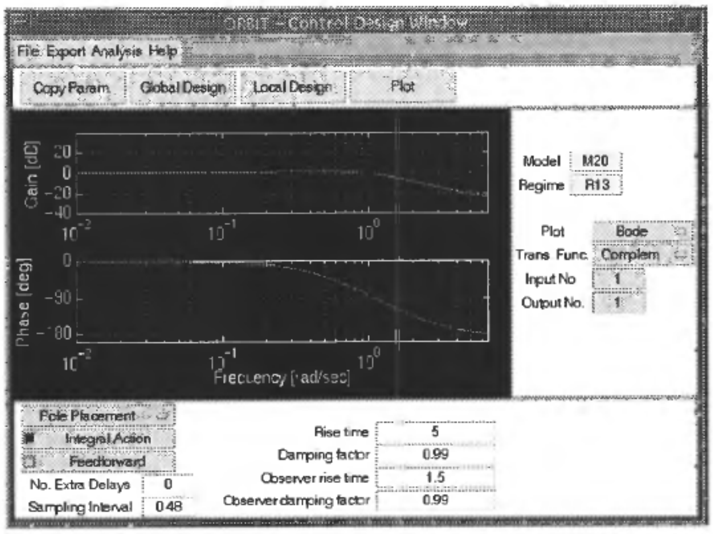
Figure 11. Model validation with ORBIT. The validation functions/windows are invoked from the Model Database Window (upper left). The Simulation window (upper right) displays simulation results for model analysis and comparison (models M20, M22 and measured data S). The Correlation Analysis Window (lower left) displays the autocorrelation function for the residuals (one-step-ahead predictors) of models M20, M21 and M22. The Prediction Error Estimate Window (lower right) contains a plot of the estimated expected predition error on the basis of separate validation data for a range of identified models.

11) a statistical measure of expected prediction performance is displayed as error bars for selected models. In this case the models' prediction performance are evaluated on a separate validation data sequence.

5. ORBITcd Functionality and User Interface

The ORBIT Control Design toolkit GUI (ORBITcd) is illustrated in Figure 12. The window has four areas:

- The menus and buttons on the top contain all the functions that can be executed from this interface. These include controller design, global controller analysis in terms of quadratic Lyapunov functions (c.f. section 3.3; this is implemented using the MATLAB LMI toolbox which contains useful numerical tools (Gahinet *et al.* 1995)), local controller analysis in terms of Bode or Nyquist plots, generating MATLAB or C-code SIMULINK S-functions, or C-code 'include' files for the real-time control system, and management of controller specification data files.
- The panel on the right contains selections of the current (active) model, operating regime, input/output channel, and graphical output.
- In the centre there is an area for displaying graphical output and analysis results, such as Bode or Nyquist plots of various local transfer functions: open loop, sensitivity and complementary sensitivity transfer functions.
- Finally, at the bottom of the local controller design parameter specifications are located. There are options for integral action and feed-forward. With pole placement, the design parameters are an arbitrary of rise-times/damping factors of the desired loop modes $A_{i,m}(q^{-1})$, including the modes of the observer polynomial $A_{o,i}(q^{-1})$. With LQG design, the design parameters are the criterion weights μ_i, and the same observer design parameters as with pole placement.



Model M20
Regime R13
Plot Bode
Trans. Func. Complem
Input No 1
Output No. 1
No. Extra Delays 0
Sampling Interval 0.48
Rise time 5
Damping factor 0.99
Observer rise time 1.5
Observer damping factor 0.99

Figure 12. ORBITcd Window. This window allows design and analysis of gain scheduled LCNs on the basis of ORBIT models. The window contains areas for entering design specifications, graphical display and buttons and means for invoking controller design and closed loop analysis functions.

The control design window shown in Figure 12 inherits the structural and local model parameters of the current model (in this case the model selected is M20). The design parameters shown in the window refer to the currently selected regime (regime R13 from M20). Note that if the model has a large number of regimes, it is possible to prepare a MATLAB script or application program which allows the user to more conveniently define the design parameters for all regimes using the API rather than the GUI. These parameters can be written to a file and loaded into the control design window using the "File" menu. The active regime (R13) in Figure 12 is one of the equilibrium regimes in second gear. For this regime the reference model and observer rise times have been chosen as 5s and 1·5s, respectively, and the damping in each case is close to unity. The active plot in the window shows a Bode plot of the complementary sensitivity function for this local design. It is clear to see that the bandwidth for this design is around 1rad/s, and that the complementary sensitivity function rolls off rapidly above this, thus ensuring that the loop will be insensitive to measurement noise and high-frequency unmodelled dynamics.

SIMULINK provides a flexible environment for simulation, validation and verification of the control system design. An example of a SIMULINK setup that includes a LCN generated by ORBITcd is illustrated in Figure 13. The model M42 is a refinement of the model M20 (the model has evolved as vehicle experiments have been carried out). A closed-loop control simulation in 3rd gear using this setup with model M42 to represent the plant (see section 3) is shown in Figure 14(a). As shown in section 3, this controller has 9 local controllers in 3rd gear. The closed-loop rise-times were chosen in each 3rd gear regime as 7s, and each local controller had a deadbeat observer polynomial. This controller was subsequently implemented in the experimental vehicle; the corresponding tracking response is shown in Figure 14(b). Note that this controller is scheduled on both speed and throttle (as well as gear). Despite the fact that the throttle changes rapidly, no destabilization of the loop occurs. A disturbance rejection test with

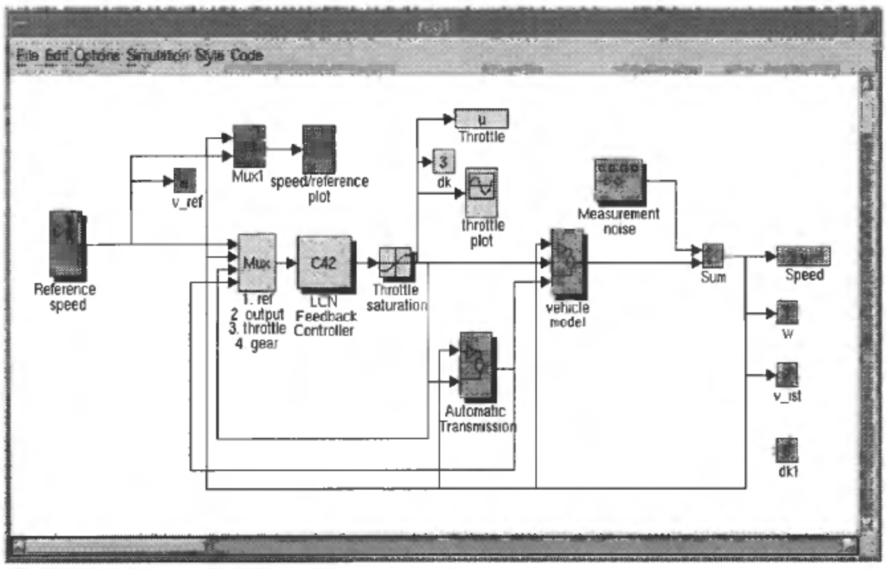
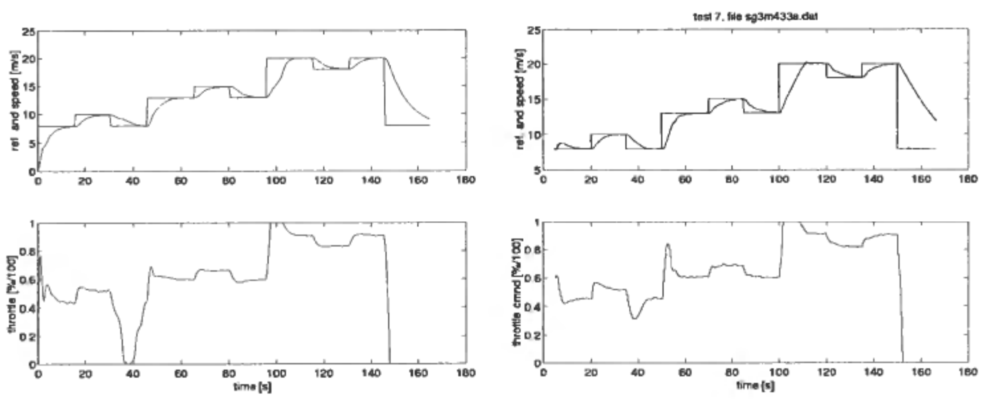


Figure 13. SIMULINK simulation of vehicle with nonlinear gain scheduled LCN (block C42). The remaining blocks define external signals, sensors and a model of the vehicle (including automatic transmission).



(a) Simulated tracking performance in 3rd gear.

(b) Tracking performance in the experimental vehicle in 3rd gear.

Figure 14. Closed-loop control: time-domain performance. In each part of the figure the top graph shows the desired and measured speed, while the lower graph is the control signal (throttle).

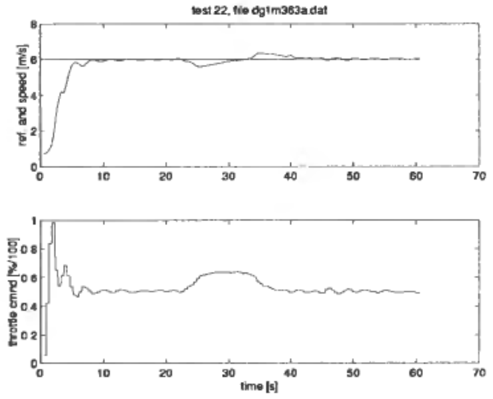


Figure 15. Disturbance rejection performance in the experimental vehicle in 2nd gear. The upward slope of 10% starts at $t = 22$s, and ends at $t = 33$s. The top graph shows the desired (6m/s) and measured speed, while the lower graph is the control signal (throttle).

this controller was carried out by driving the vehicle up a 10% slope in 2nd gear. The experimental response in shown in Figure 15. Satisfactory disturbance rejection is achieved. In second gear the nonlinear controller consists of 7 local controllers, each with a desired closed-loop rise-time of 6s and deadbeat observer.

6. Concluding Remarks

Gain scheduling control has several attractive properties from a practical nonlinear control design point of view. The most important is that linear control theory can be applied for design, and partially for analysis. Recent developments have extended the applicability and understanding of gain scheduled control; transient performance can be explicitly addressed by off-equilibrium linearization, there exist conditions for analyzing stability, robustness and performance when scheduling on rapidly time-varying variables such as the control input, and the close links to fuzzy control and feedback linearization can be exploited. This paper describes an integrated software environment for design, implementation and analysis of such advanced gain scheduled control design: local controller networks. The highly integrated tools for modelling, identification, and control design and analysis allow rapid prototyping of highly nonlinear control systems. An automotive application example illustrates the usefulness of the approach.

ACKNOWLEDGEMENTS

This work was supported by the European Commission under the ESPRIT Basic Research Project 8039 NACT.

REFERENCES

HUNT, K. J. and JOHANSEN, T. A. (1997). Design and analysis of gain-scheduled local controller networks. *Int. J. Control*, **66,** 619–651.

BOYD, S., EL GHAOUI, L., FERON, E. and BALAHRISHNAN, V. (1994). *Linear Matrix Inequalities in System and Control Theory*, SIAM, Philadelphia.

DRIANKOV, D., PALM, R. and REHFUSS, U. (1996). A Takagi-Sugeno fuzzy gain-scheduler, in *Proc. IEEE Conf. Fuzzy Systems, New Orleans*, pp. 1053–1059.

JOHANSEN, T. A., HUNT, K. J., GAWTHROP, P. J. AND FRITZ, H. (1998). Off-equilibrium linearization and design of gain scheduled control with application to vehicle speed control. *Control Engineering Practice*, 6.

SHAMMA, J. S. and ATHANS, M. (1990). Analysis of gain scheduled control for nonlinear plants. *IEEE Trans. Automatic Control*, **35,** 898–907.

RUGH, W. J. (1991). Analytical framework for gain scheduling. *IEEE Control Systems Magazine*, **11**(1), 79–84.

JOHANSEN, T. A. (1997). ORBIT control design toolbox V1.10—User's guide and reference, Tech. Rep. STF72 F97312, SINTEF, Trondheim, Norway.

JOHANSEN, T. A. and FOSS, B. A. (1993). Constructing NARMAX models using ARMAX models. *Int. J. Control*, **58,** 1125–1153.

HUNT, K. J., JOHANSEN, T. A., KALKKUHL, J. C., FRITZ, H. and GÖTTSCHE, TH. (1997). Nonlinear speed control design for an experimental vehicle using a generalised gain scheduling approach. *IEEE Trans. on Control Systems Technology*, Submitted for publication.

FRANKE, U., BÖTTIGER, F., ZOMOTER, Z. and SEEBERGER, D. (1995). Truck platooning in mixed traffic, in *Proc. Intelligent Vehicles Symposium, Detroit, USA*, pp. 1–6.

HUNT, K. J., KALKKUHL, J. C., FRITZ, H. and JOHANSEN, T. A. (1996). Constructive empirical modelling of longitudinal vehicle dynamics with local model networks. *Control Engineering Practice*, **4,** 167–178.

GEHRING, O. and FRITZ, H. (1997). Practical results of a longitudinal control concept for truck platooning with vehicle-to-vehicle communication, in *IEEE Conference on Intelligent Transportation Systems, Boston, USA*, Submitted for publication.

MURRAY-SMITH, R. and JOHANSEN, T. A., Eds. (1997). *Multiple Model Approaches to Modelling and Control,* Taylor and Francis, London.

TAKAGI, T. and SUGENO, M. (1985). Fuzzy identification of systems and its application to modeling and control. *IEEE Trans. Systems, Man and Cybernetics*, **15,** 116–132.

ÅSTRÖM, K. J. and WITTENMARK, B. (1997). *Computer Controlled Systems: theory and design*, Prentice-Hall, Third Edition.

KUCERA, V. (1979). *Discrete Linear Control: The Polynomial Equation Approach*, Wiley, Chichester.

HUNT, K. J. (1989). *Stochastic Optimal Control Theory with Application in Self-Tuning Control*, Springer-Verlag, Berlin.

HUNT, K. J., Ed. (1993). *Polynomial Methods in Optimal Control and Filtering*, P. Peregrinus.

SHAMMA, J. S. and ATHANS, M. (1992). Gain scheduling: Potential hazards and possible remedies. *IEEE Control Systems Magazine*, **12**(3), 101–107.

KAMINER, I., RASCOAL, A. M., KHARGONEKAR, P. P. and COLEMAN, E. E. (1995). A velocity algorithm for the implementation of gain-scheduled controllers. *Automatica*, **31,** 1185–1191.

LEITH, D. J. and LEITHEAD, W. E. (1996). Appropriate realization of gain-scheduled controllers with application to wind turbine regulation. *Int. J. Control*, **65,** 223–248.

HUNT, K. J., HAAS, R. and MURRAY-SMITH, R. (May 1996). Extending the functional equivalence of radial basis function and fuzzy inference systems. *Trans. IEEE on Neural Networks*, 7(3), 776–781.

HUNT, K. J., HAAS, R. and BROWN, M. (June 1995). On the functional equivalence of fuzzy inference systems and spline-based networks. *International Journal of Neural Systems*, **6**(2), 171–184.

ZHAO, J., GOREZ, R. and WERTZ, V. (1997). Fuzzy control based on linear models, in *Multiple Model Approaches to Modelling and Control*, R. Murray-Smith and T. A. Johansen, Eds. Taylor and Francis Ltd., London.

WANG, H. O., TANAKA, K. and GRIFFIN, M. F. (1996). An approach to fuzzy control of nonlinear systems: Stability and design issues. *IEEE Trans. Fuzzy Systems*, **4,** 14–23.

MURRAY-SMITH, R. and JOHANSEN, T. A. (1995). Local learning for local model networks, in *Proc. 4th IEE Int. Conference on Artificial Neural Networks*, Cambridge, UK, pp. 40–46.

JOHANSEN, T. A. and FOSS, B. A. (1997). ORBIT—operating regime based modeling and identification toolkit, in *Preprints IFAC Symposium on System Identification, Kitakyushu, Japan*, pp. 961–968.

JOHANSEN, T. A. (1997). Constrained and regularized system identification, in *Preprints IFAC Symposium on System Identification, Kitakyushu, Japan*, pp. 1467–1472.

JOHANSEN, T. A. and FOSS, B. A. (1995). Identification of non-linear system structure and parameters using regime decomposition. *Automatica*, **31,** 321–326.

SUGENO, M. and KANG, G. T. (1988). Structure identification of fuzzy model. *Fuzzy Sets and Systems*, **26,** 15–33.

LJUNG, L. (1987). *System Identification: Theory for the User*, Prentice-Hall, Inc., Englewood Cliffs, NJ.

GAHINET, P., NEMIROVSKI, A., LAUB, A. J. and CHILALI, M. (1995). *LMI Control Toolbox—For Use with MATLAB*, The MathWorks, Inc.