A Modeling Approach for Mechatronic Systems - Modelin g and Simulation of an Elevator System

Peter Schneider, Erich Huck, Peter Schwarz
Fraunhofer Institute for Integrated Circuits, Design Automation Division Dresden, Germany

Abstract — Mechatronic systems as well as other technical systems (microsystems, distributed automation systems, ...) can be characterized as complex heterogeneous systems. Typically, they show some of the following features:
- mixed physical domains (mechanical, electrical, thermal, fluidic, ... phenomena),
- partially close coupling between these domains,
- distributed and lumped effects or elements, respectively,
- continuous and discrete signals and systems
(in electronics: analog and digital).

Often, the modeling of continuous systems leads to very large systems of stiff differential equations. In contrast, system simulation requires simpler and faster models to investigate the interaction between all components. That’s why a powerful methodology for the modeling and simulation of mechatronic systems is demanded which considers their special characteristics. This methodology must cover a unified modeling approach, standardized modeling languages, algorithms and tools for model generation and capabilities for system optimization. In the following an approach for modeling is presented which meets these requirements. A real-world example, the modeling of an elevator system, shows the application of the modeling approach.

Index Terms — mechatronics, modeling approach, differential algebraic equations (DAE), elevator system, neural networks, adaptive control strategy, automatic elevator setup

I. INTRODUCTION

Basically, depending on the level of abstraction, partial differential equations (PDE) and ordinary differential equations (ODE) are the mathematical models (system equations) of continuous systems. For the presented approach we will focus on ODE. There are a lot of algorithms and computer programs available for the numerical solution of these system equations. Some of them (e.g. MATLAB/SIMULINK, MATRIX/SystemsBuild) assume an explicit formulation:

\[ \dot{x} = f(x, u, \tau) \]

(1)

Other tools are able to handle the more general implicit form:

\[ 0 = \tilde{f}(\tilde{x}, x, u, \tau) \]

(2)

where \( x \) is the vector of state variables, \( \dot{x} \) is the vector of their time derivatives, \( u \) is the vector of inputs and \( \tau \) is time. Especially for systems of ordinary differential equations with algebraic constraints (differential algebraic equations - DAE) this kind of formulation - eq. (2) - is needed.

Due to the complexity of mechatronical systems [2] there is a strong demand for better assistance in formulating system equations. To analyze real-world problems a powerful interdisciplinary modeling methodology covering:
- a unified modeling approach,
- standardized modeling languages,
- algorithms and tools for model generation (order reduction, approximation etc.) and
- properties for system optimization

is necessary.

II. A MODELING APPROACH FOR MECHATRONIC SYSTEMS

Elevator systems are typical examples for complex mechatronic systems. High safety-relevant demands are special properties of these systems. Practical experiments for the function validation of developed components (subsystems) and their interactions are mandatory necessary. However, the opportunities for experimental system optimization are often limited by the system size and complexity. Experiments with system versions are too expensive or impossible because of safety requirements. Furthermore, the investigation of relevant operating conditions is very complicated. This difficult design situation can be improved by accompanying system simulations. A necessary precondition for simulation is the modeling of the system with all components.

Many physical effects in elevator systems can be formulated mathematically by ordinary or partial differential equations. For information-processing subsystems, e.g. the control system, both the analog and the digital descriptions are needed. Therefore, simulators for analog and mixed analog-digital simulation (e.g. SABER [3], ELDO [4], Spectre) are very important. These simulators are suitable for solution of nonlinear ordinary differential equations, often with algebraic constraints. In addition, they provide opportunities for simulation of time-discrete system components, e.g. digital electronic circuits. Distributed systems can be described by ordinary differential equations after discretisation concerning the spatial coordinates. Furthermore, the above-mentioned tools provide behavioral description languages (MAST, HDL-A, VHDL-AMS [5]) which are suitable for modeling of electrical as well as nonelectrical systems and the mixture of time-continuous and time-discrete systems.

A modeling methodology for complex systems can be described as follows:
The system is divided into subsystems which may be modeled much easier. This partitioning corresponds to the functionality of the system, to its physical structure, or to the design engineer’s experience about the processes in the system.

The subsystems exchange “signals” - i.e. energy flow, electrical or mechanical quantities, signal values in the control part etc. These signals are the only communication paths between the component models. They are provided at the terminals of subsystems, i.e. at the connection points of the models. Except for the signals at the terminals there is no further information about the entire system available to the model.

The entire system model results from connection of component models in the same manner, as the real system is assembled from its subsystems.

Component models may consist of the connection of elementary models (hierarchical procedure) or of equation-oriented descriptions of physical effects (behavioral descriptions). A mixture of structural and behavioral models is also possible.

This hierarchical structure-oriented modeling is similar to object-oriented approaches [6], [7], [8], [9]. It is very descriptive and can be supported easily by libraries of basic models.

In order to simplify the modeling, the definition of a uniform model structure and a uniform interface of all models is desirable [10], [11]. Figure 1 shows the model interface and a uniform mathematical description.

---

III. PROBLEM DEFINITION FOR ELEVATOR SIMULATION

An elevator system - a simplified structure is shown in Figure 2 - consists of the subsystems: elevator mechanics, elevator control, sensor system (process measurement instrumentation, car and floor panels) and motor control (speed specification, power electronics).

The elevator control implements receiving and management of transportation requests (calls) from passengers and the analysis of measured values for actual position and driving characteristics of the elevator.

---

Figure 1: Terminal signals of a multiport model - uniform mathematical description of model equations

There are two types of terminal signals:

- conservative signals v and i - e.g. voltages and currents in electrical models or force and velocity in mechanics - which are also related to the energy transport between subsystems,

- non-conservative signals a and d - signal values in the control part (analog as well as digital) which consider only the information exchange.

Signals of the same type are assembled into vectors, and thus they are vector-valued functions of time t. Furthermore, the terminal signals are separated into the following categories:

- $v_1, i_2, a_{in}, d_{in}$ independently chooseable difference, flow, and non-conservative signals (analog and digital)

- $v_2, i_1, a_{out}, d_{out}$ dependent difference, flow, and non-conservative signals (analog and digital)

Fig. 1: Terminal signals of a multiport model - uniform mathematical description of model equations

Fig. 2: Main components of an elevator system
The driving characteristics

- travel time,
- positioning precision,
- maximum car acceleration and
- the ripple of the car speed etc.

are crucial for travel comfort, elevator transportation performance, and power consumption.

These characteristics are influenced by the running curve which is the input specification for the angular speed at the motor shaft of the elevator [14]. The running curve is specified by parameters which are inputs for the motor control system (Figure 3). The closed loop control system has to track the elevator drive as precise as possible on the given curve.

Beside closed loop control, power electronics and the mechanical part of the elevator have a crucial impact on the driving characteristics. Nowadays, the parameters of the running curve are adjusted for the elevator setup by a technician. The „optimal“ values for the running curve parameters are determined by repeated tests. The technician checks during several travels in the elevator car, whether the elevator system offers a behavior which is subjectively pleasant to him. If not, the running curve parameters are changed, until the driving characteristics of the elevator correspond to the desired behavior. This desired behavior may vary from system to system according to requirements as travel comfort or transportation performance.

The design of a control strategy which determines running curve parameters automatically using neural networks (Figure 4) has to be supported by elevator simulations. This design task is important to elevator suppliers regarding two aspects:

- Automatic setup of the elevator:
The aim is to replace the technician’s adjustment by an automatic setup procedure integrated in the elevator control system.
- Adjustment under changing process conditions (ageing, wearout):
By wearout and ageing of the system a degradation of the elevator travel comfort occurs. In this case a maintenance is necessary. Using neural network the running curve parameters can be adapted during elevator operation in order to preserve optimal driving characteristics over a longer period of time.

![Fig. 3: Running curve (desired angular speed at the motor shaft)](image_url)

Parameters of running curve:
- acceleration \( a_{\text{pos}} \)
- deceleration \( a_{\text{neg}} \)
- maximum speed \( \omega_{\text{max}} \)
- curve roundings \( r_1, r_2, r_3 \) und \( r_4 \)

![Fig. 4: Elevator setup: a) setup by technician (actual situation); b) intelligent setup procedure with neural network (new approach)](image_url)

![Fig. 5: Elevator system - detailed structure of the entire system](image_url)
For the application of neural networks (NN) it is necessary to capture the actual driving characteristics of the elevator car by sensors. Speed of the car, car position and car acceleration have to be measured. From these data the driving characteristics must be extracted and passed on to the neural network as input data. The neural network determines a new vector of running curve parameters which takes the actual elevator behavior into account. For the computer-aided design and optimization of these control strategies, the examination of the interaction of the elevator mechanics, the control system (including NN) and the sensor system have to be carried out by simulations of the mechatronic system. Therefore, models of all subsystems of the elevator system are needed.

IV. MODELING OF THE ELEVATOR SYSTEM

The mechanical part of the elevator system may be described on different levels of abstraction. In general, all mechanical quantities are three-dimensional and have to be modelled in this way. A typical representative is the finite element method (FEM) [15], [16], [17] which is widely used in component design. Another way of three-dimensional modeling in mechanics are multi-body-systems (MBS) [18], [8] based on Lagrangian equations. Both approaches do not support the combined treatment of mechanics and electronics. That’s why we address system simulators like SABER or ELDO.

For system simulation the three-dimensional modeling is not a general problem if behavioral modeling languages like MAST, HDL-A or VHDL-AMS are used [19], [20], [21], [22]. Three-dimensional modeling with the build-in models of system simulators which are not focussed on mechatronics is very cumbersome. A typical example is the modeling of mechanical networks [23], [24], [25] by equivalent electrical circuits and simulating them with SPICE-like simulators. The extension from the one-dimensional approach to the three-dimensional one and the simultaneous consideration of translation and rotation requires a lot of ideal transformers. In contrast, behavioral modeling with equations is straight forward.

For the modeling of the elevator (Figure 5 shows a more detailed structure of the system), the procedure given in Section II was applied. A description of the elevator’s mechanical part as one-dimensional system using a reduced set of system equations is sufficient for the neural network design.

The first step is focussed on partitioning the system into subsystems as well as the definition of coupling quantities between these subsystems. The result of this partitioning is shown in Figure 6 as a schematic diagram of the elevator system. The interaction between the subsystem models is established by coupling quantities at the interfaces of the models (terminals) as depicted in Figure 6. As mentioned in Section II these coupling quantities are signals in the information-processing and control part (directed signals) as well as flow and difference quantities (e.g. torque and rotation speed) in the mechanical part of the model.

Subsystems can be modeled with a varying, problem-dependent accuracy, while maintaining the number and type of terminal quantities. This multi-level approach [26] offers the opportunity to fit system modeling to the special aim of a sim-
ulation-based investigation in a very flexible manner. For example, the effect of mechanical wearout can be considered as disturbance quantity in certain subsystems.

For the modeling of the elevator system different kinds of model description were used - block diagrams or generalized networks build up from basic elements (structured models) and behavioral descriptions. For subsystem modeling usually a combination of both methods is necessary, i.e. the structured models are assembled from standard elements of the simulator as well as from additional elements whose behavior is described by the user.

In Figure 7 the definition of the model interface a) and two gear models with different accuracy b) and c) are presented. The first model (GEAR_L1) considers only the transmission ratio of the gear. The second one (GEAR_L2) includes more complicated effects like bouncing, backlash and nonlinear friction modelled by the combination of standard elements and behavioral models.

The same principle is applied to the modeling of the block car/guide rail (Figure 8). The model consists partially of standard elements of the simulator - the force sources shown in the upper part of the block car/guide rail. For the nonlinear model of the guide rail friction the standard element for $f_r$ can be substituted by a behavioral model implementing the equations shown on the lower part of Figure 8. The nontrivial shape of $f_r(x_K)$ may be described approximately by the function shown in Figure 8 or alternatively by introducing an additional internal state.

An experimentation model of the system can be formulated maintaining the basic structure but using different subsystem models. This model meets different analysis requirements.

V. INTEGRATION OF NEURAL NETWORKS INTO THE ELEVATOR SIMULATION

The original objective of the investigations was to apply neural networks as adaptive control algorithms for automatic setup of the elevator control and for the compensation of effects caused by ageing and wearout of the elevator mechanics. Therefore, the model of the system has to be extended. Besides models for the feedback control system and the electromechanical system (including parasitic effects) models of the neural network are needed.

System simulators like SABER or ELDO do not support the simultaneous simulation of dynamic systems and neural networks. That's why for the inclusion of the neural networks into system simulation a design environment was implemented [27]. A simulator coupling between a system simulator (for mechatronic systems) and a neural network simulator [28] is the core of this environment. The algorithms of system simulators (SABER or KOSIM [26]) were coupled with the Stuttgart Neural Network Simulator (SNNS) [28] using interprocess communication routines.

The design environment allows a simulation of the mechatronic system (analog-digital, electrical-nonelectrical) with the system simulator and simultaneously the execution of the neural network algorithm. The coupled simulation is used during
the training phase of the NN as well as for the simulation runs with the trained NN (recall phase). Furthermore, it is possible to generate C-code of a trained network and to integrate this into model libraries of the system simulator. A more comprehensive description of this design environment and its capabilities is given in [27] and [29].

In order to improve the driving characteristics of the elevator by control strategies, system simulations of the elevator system were performed using this design environment.

For the training of the neural network a variety of simulation runs of the mechatronic system were carried out for different values of the running curve parameters. From these simulations the driving characteristics of the elevator (positioning time, maximum acceleration, ripple of the car speed etc.) were determined, and the training of the network was executed using this data. Actual curve parameters are used as input and driving characteristics as output data. The applied multilayer feed forward networks (MLP) [31] were able to represent the dependency between running curve parameters and the driving characteristics very well. Relyed on this trained network the calculation of the adjustment of the running curve parameters is carried out via gradual backward calculation of the error of the neurons. Therefore, based on desired driving characteristics (specification) an error is assigned to the output neurons of the trained network. The error of all neurons in the network is determined similar to the learning process (see [31] and [32] for details of the learning in MLP). From the error of the input neurons a favorable modification of the running curve parameters for the current elevator state is calculated [33]. This algorithm was integrated into the model of the elevator control. Figure 10 shows the structure of the elevator control model. The model frame is implemented in the behavioral language of the system simulator. The neural network for the parameter determination is embedded into this model frame via the simulator coupling.

VI. SYSTEM SIMULATION OF THE ELEVATOR SYSTEM

The investigation of the elevator behavior (Figure 6) is carried out with models of different accuracy. Saber was used as system simulator.

A first analysis was performed with a simplified elevator control model (without NN). The principal dynamic behavior of the elevator was examined. For these simulations the running curve parameters were set to default values. Figure 11 shows simulation results for a short travel between two floors for a typical elevator parameter set. The running curve and actual speed of the elevator car over an operating cycle are shown. The system dynamics is also influenced by the start
and brake behavior. The effect of different frictional forces at the guide rails are also shown. With unfavorable conditions and at low speed jerks occur at the start of the car (Figure 11c) which can be observed also in reality.

For the improvement of the driving characteristics by control strategies, various simulations were executed including the neural network. Using the procedure mentioned above the neural network was trained and integrated into the elevator control unit (Figure 10). Using this, an automatic elevator setup procedure was simulated. Starting with an initial set of running curve parameters a travel of the elevator car was calculated and the driving characteristics were derived. The neural network calculates a new parameter vector for the next travel which is then simulated, the driving characteristics are derived and again a new parameter set is calculated. This is performed until no further improvement of the elevator behavior is achieved. Figure 12 shows the result of the simulation for two criteria of travel comfort: a) minimal time for travel and b) decrease of maximum acceleration. Several travels of the elevator are depicted representing the automatic setup process. The improvement obtained by the neural network control is marked by arrows.

VII. CONCLUSIONS

An approach for modeling mechatronic systems was presented which is suitable for the combined analysis of electrical and mechanical subsystems together with the control systems. A simulator coupling between a system simulator and a neural network tool allows the integration of adaptive control strategies into the system simulation and optimization. The modeling approach and the simulator coupling were successfully applied to the analysis of an elevator system.

ACKNOWLEDGEMENT

We would like to thank our colleagues in the modeling and simulation department at Fraunhofer IIS / EAS Dresden as well as Karl-Heinz Diener for his assistance in preparing this paper.

The presented work was done in a project funded by the German Government: SIMKOS - Simulation komplexer Systeme unter Einbeziehung intelligenter Komponenten (Ref.-No.: 16SV429).

REFERENCES