Symbolic Model Reduction Applied to Realtime Simulation of a Construction Machine

Lars Mikelsons[1] Hongchao Ji[1] Thorsten Brandt[1] Oliver Lenord[2]
[1] Institute for Mechatronics and System Dynamics , University of Duisburg-Essen, Germany
[2] Bosch Rexroth AG, Germany
{mikelsons,ji,brandt}@imech.de

Abstract

The vehicle response of construction machines strongly depends on the tuning of the control system in interaction with the drive system. A compromise between performance and comfort needs to be found to fulfill the operators requirements on a high usability of the machine. In order to achieve an optimal behavior Hardware-in-the-Loop simulation techniques offer a suitable approach to determine the overall behavior in advance. Prerequisition is a realtime capable simulation model of the considered system. Therefore, in this paper the mathematical model of the system is automatically adapted by symbolic model reduction algorithms in order to match real-time requirements on a given hardware. Inputs to the automatic reduction algorithm are the complex mathematical system model, the desired realtime cycle and the number of floating operations per second (flops), which can be realized by the chosen target hardware. The outputs of the algorithm are the automatically reduced model, which is guaranteed to run in realtime on the target hardware and the maximal model error for the test scenario. In this paper, the reduction procedure is demonstrated for the complex hydromechanical model of a so-called skid steer loader. Summarizing, the proposed procedure of symbolic model reduction helps to reduce the developing phase of mechatronic prototypes dramatically as the adaptation of the system model with respect to the target hardware is completely automated.
Keywords: symbolic model reduction, realtime, construction maschines, object oriented modelling

1 Introduction

Nowadays many complex systems are modeled in object oriented simulation tools like for example Dymola [4] or SimulationX, which base on Modelica [5] and hence generate a symbolic representation of the emerging DAE system. Having a symbolic representation at hand, the equations can be manipulated, simplified or even reduced. While algorithms for simplification and index reduction are already implemented in those simulation tools, not much attention has been paid to symbolic reduction techniques [2, 13]. Though, they are a very powerful tool for automated generation of less complex models [9]. Symbolic reduction techniques were first used in analog circuit design [2] and based on the DC-analysis of nonlinear analog circuits. These techniques were extended to the reduction of arbitrary DAE-systems in [12, 13]. Hence, symbolic reduction techniques can be used for the modeling and design of mechatronic systems [10]. Examining a complex physical system like construction maschines in many cases only one model is not sufficient. Often a very accurate model is required in order to analyze certain physical effects, while at the same time a model for realtime simulation is required. Here symbolic reduction techniques come into play. Up to now symbolic reduction techniques lower the complexity (and therefore the level of detail) of the model until a user defined error bound is reached. In this contribution this approach is extended in order to obtain models which are usable for realtime simulation on a given realtime target in a given realtime cycle.
In section 2 symbolic reduction techniques are briefly introduced and extended for realtime reduction. After that the approach is applied to a construction machine called skid steer loader. In section 3 the MathModelica [6] model of the skid steer loader is presented, while in section 4 the reduction results are given. The paper closes with a conclusion and an outlook in section 5.

2 Symbolic Reduction Techniques

The basic idea of symbolic model reduction techniques is to identify those terms of a DAE (or ODE)

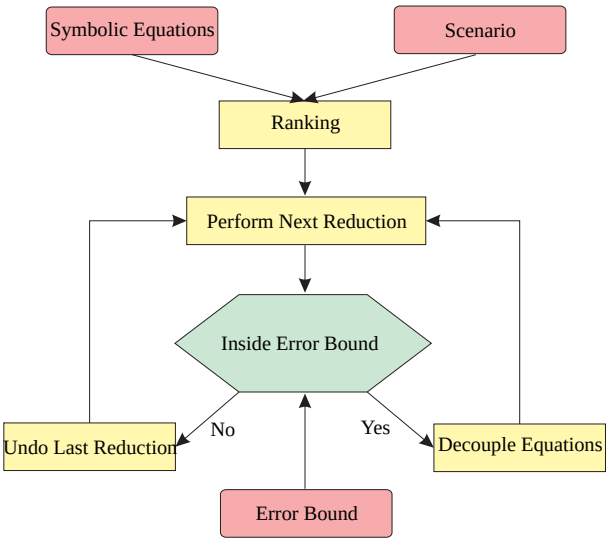


Figure 1: Scheme of the Reduction Algorithm

system, whose influence on the solution of the system is minor, and to perform a reduction on them (e.g. to neglect them). The algorithm consists of two steps, see for example [10] and [13]. First a specific reduction technique is chosen. Afterwards the relevance of each term for the solution of the DAE-System is estimated in the so called "ranking". Then the terms are sorted in increasing order with respect to their influence on the solution in order to perform the reductions as long as the solution of the reduced DAE-System remains within a user-defined error bound ε [13]. This basic idea is extended in section 2.4 in order to obtain reduced models, which can be simulated in realtime on a given realtime target. Possible reduction techniques are neglecting terms, setting terms to constants, linearization of terms or symmetry considerations. While the first three reductions are operations on terms of the DAE-System, the last one operates on variables and is explained later on. A scheme of the symbolic reduction algorithm is shown in Fig.1 for a chosen reduction technique. Given a scenario (system inputs,initial states and parameters) and an error bound, the algorithm starts with the ranking. Afterwards it is checked whether the reductions lead to an error inside the error bounds, beginning with the smallest. Finally, a less detailed model, performing within the prescribed error bounds results.
Let now

$$\mathbf{F} : \Omega \times I \mapsto \mathbb{R}^m \tag{1}$$

be differentiable, where $\Omega \subset \mathbb{R}^n \times \mathbb{R}^n$ is an open set. Then

$$\mathbf{F}(\mathbf{x}, \dot{\mathbf{x}}, t) = 0 \tag{2}$$

is called DAE-system if $\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{x}}}$ is singular. Furthermore, let $\mathbf{F}$ be given in expanded form

$$\mathbf{F}_i(\mathbf{x}, \dot{\mathbf{x}}, t) = \sum_{k=1}^{l_i^1} t_{k_i}^1(\mathbf{x}, \dot{\mathbf{x}}, t), \quad 1 \leq i \leq m, \tag{3}$$

where l_i^1 is the number of terms in $\mathbf{F}_i$ and $t_{k_i}^1$ denotes the k-th term in $\mathbf{F}_i$. Each term in the first level $t_{k_i}^1$ may consist of a function $f_{k_i}^1$, whose argument is a sum of $l_{k_i}^2$ second level subterms $t_{k_i}^2$ $(1 \leq i \leq l_{k_i}^2)$

$$t_{k_i}^1(\mathbf{x}, \dot{\mathbf{x}}, t) = f_{k_i}^1\Big(\sum_{k=1}^{l_{k_i}^2} t_{k_i}^2(\mathbf{x}, \dot{\mathbf{x}}, t)\Big), \tag{4}$$

and so on. Here level indicates the hierarchy of arguments nested into each other in each single summand. Then the set $\mathscr{T}^i$ is the set of all terms in the i-th level. The manipulation of a term is called reduction in the following. Consequently, for the set of all reductions $\mathscr{K}^i$ for one reduction technique in a level i, it holds

$$\left|\mathscr{T}^i\right| = \left|\mathscr{K}^i\right|. \tag{5}$$

For $\kappa \in \mathscr{K}$

$$\mathbf{F}^\kappa = 0 \tag{6}$$

is the DAE-system emerging from the reduction κ. Then for DAE-systems of the form of Eq. 2

$$\mathbf{F}(\mathbf{x}, \dot{\mathbf{x}}, t, \mathbf{u}) = 0 \tag{7}$$

with system inputs $\mathbf{u}$, a scenario is the set of a vector field defined on the interval I for the system inputs, the initial values and the parameters. Furthermore, $\mathscr{N}(F(\mathbf{x}, \dot{\mathbf{x}}, t), \mathbf{u})$ is the solution of Eq. 2 computed by a numerical integrator $\mathscr{N}$ at nodes $t_1, \ldots, t_N$. The solution

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_{out} \\ \bar{\mathbf{y}} \end{bmatrix} = \mathscr{N}(F(\mathbf{x}, \dot{\mathbf{x}}, t), \mathbf{u}) \tag{8}$$

consists of two components. In $\mathbf{y}_{out}$ the n_{out} output variables are contained, while $\bar{\mathbf{y}}$ consists of the remaining internal variables.

2.1 Reduction Techniques

As already mentioned above possible reduction techniques are the neglecting terms ($\mathscr{U}_{neg}$), setting terms to constants ($\mathscr{U}_{const}$), symmetry considerations ($\mathscr{U}_{sym}$) or the simplification of piecewise functions. In this contribution only $\mathscr{U}_{neg}$ is chosen. Certainly, the easiest manipulation of a term is neglecting it. Using $\mathscr{U}_{const}$ for each term a constant has to be chosen. Usually the mean value throughout the simulation is employed. Clearly, this mean value has to be determined before. At first sight, this looks like a drawback, but a reference simulation is essential for the ranking anyway as will be seen in the next section. However, other values than the mean value are thinkable. Choosing $\mathscr{U}_{sym}$ at first variables which have similar values throughout the simulation are sought. Alternatively variables, which are expected to be similar can be flagged. For two similar variables every occurrence of the first variable (or its derivative) is substituted by the second variable (or its derivative). Consequently, now one equation can be canceled. A reasonable choice is that equation which leads to smallest error.

2.2 Ranking

In [12] different ranking algorithms are proposed. In this contribution only the so called One-Step Ranking will be discussed. In general a ranking procedure estimates the influence of a reduction on the solution of a DAE (or ODE) system. A reasonable measure for the influence of a reduction is the error emerging from the reduction. In order to get a good estimate of that error a reference solution $\mathbf{y}^\star$ is required. The crux of the matter is that the quality of the estimate increases with the duration of the ranking procedure. Hence, a ranking procedure should be a good compromise between computation time and accuracy. Mathematically speaking a ranking procedure $\mathscr{R}$ maps two DAE-systems on a real value, estimating the error between their solutions. Apparently, perfect accuracy can be achieved by the use of simulations. Though, this would lead to very high computation costs.

One-Step Ranking Typically, computing the solution of a DAE-system, at each time step a non-linear system of equations is iteratively solved. Usually the solution of the preceding time step is used as the initial value for the solution of the system of non-linear equations at the next time-step. For the computation of the solution of Eq.6, the reference solution $\mathbf{y}^\star$ at the corresponding time steps can be used for the initial values. Now, additionally limiting the iterations to one, a estimate of the solution of Eq.6 $\widehat{\mathbf{y}}$ is obtained. Consequently

$$\mathscr{R}_{step}(\mathbf{F},\kappa) = \|\mathbf{y}^\star_{out} - \widehat{\mathbf{y}_{out}}\| \tag{9}$$

is computed. The one-step ranking usually delivers a good compromise between accuracy and runtime.

2.3 Term Cancellation

In the term cancellation procedure the ranking is used, to perform as many reductions as possible, while preserving the desired accuracy. Hence, reductions are performed as long as the error of the reduced model remains within the error bound ε. The error emerging from the reductions is measured only at the n_{out} output variables. Thus, ε has dimension n_{out}. To perform as many reductions as possible, it is beneficial to start with those reductions, which lead to a small error. Thus, first the set of reductions $\mathscr{K}$ is sorted in ascending order depending on the ranking, resulting in $\mathscr{K}_{sort}$. Now, one possibility is to check one reduction of $\mathscr{K}_{sort}$ after the other. This is done by checking the computed solution of the reduced DAE-system for staying within the error bound ε. However, this method can be accelerated by the use of clusters [11]. Using clusters, the set of reductions $\mathscr{K}_{sort}$ is divided into s disjunct subsets

$$\mathscr{K}_{sort} = \bigcup_{i=1}^{s} \mathscr{S}_i, \tag{10}$$

where

$$\mathscr{S} = [S_1, \ldots, S_s]. \tag{11}$$

Each cluster $\mathscr{S}_i$ contains reductions leading to a similar estimated error (for example up to a factor of 10). Now the clusters are checked one after another, beginning with S_1 containing the reductions leading to the smallest estimated error. Thus, multiple reductions can be verified by one simulation. If a cluster $\mathscr{S}_i$ can not be verified (the reductions of $\mathscr{S}_i$ lead to errors greater than the error bound ε), $\mathscr{S}_i$ is divided disjunct into two clusters $\mathscr{S}_i^1$ and $\mathscr{S}_i^2$. The term cancellation procedure then continues with $\mathscr{S}_i^k$ $(1 \leq k \leq 2)$. The whole reduction algorithm is shown in algorithm 1 for a reduction technique $\mathscr{U}$, a ranking procedure $\mathscr{R}$, a numerical integrator $\mathscr{N}$ and a certain level k. Here for a reduction $\kappa \in \mathscr{K}$, κ^{-1} undoes the reduction.

2.4 Symbolic Reduction for Realtime Purposes

In this contribution the algorithm described above is extended in order to obtain models, which can be used for realtime simulation on a given realtime target within a given realtime cycle. To simulate a model in realtime it must be guaranteed that one integration step can be computed within a realtime cycle, i.e., the worst case run time for one integration step has to be smaller than the realtime cycle. Hence, two quantities are important. First the maximal number of required floating point operations (FLOPs) for one integration step and second the number of FLOPs, which can be computed on the realtime target in one second (FLOP/s). The number of required FLOPs depends on $\mathbf{F}$ and the integration method used. Clearly, for realtime purposes a fixed step solver has to be chosen. Then the maximal number of required FLOPs σ_{req} (using a BDF method) can be expressed as

$$\sigma_{req} = n_{iter}^{BDF}\,(n_{eval}(\sigma_F + \sigma_J + \sigma_{dJ}) + \sigma_{LSE}) + \sigma_{event}. \tag{12}$$

Here, n_{iter}^{BDF} denotes the maximal number of Newton iterations during one integration step, n_{eval} denotes the number of required function and jacobian evaluations (depending on the order of the method), σ_F denotes the required number of FLOPs for one evaluation of $\mathbf{F}$, σ_J denotes the required number FLOPs for one evaluation of $\frac{\partial \mathbf{F}}{\partial \mathbf{x}}$, σ_{dJ} denotes the required number FLOPs for one evaluation of $\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{x}}}$ and σ_{LSE} is the number of required FLOPs for the solution of the emerging system of linear equations within every Newton iteration. Furthermore, σ_{Event} denotes the maximal number of required FLOPs for the event-handling (finding new consistent initial values), which has to be considered since an estimate for the worst case runtime is demanded. One common approach to calculate new consistent initial values is the "event iteration" [7]. Having a DAE system with n_{event} zero functions at hand, the maximal number of required FLOPs for σ_{event} then reads

$$\sigma_{event} = 2^{n_{event}} \cdot n_{iter}^{event} \cdot (\sigma_F + \sigma_{J_{event}}), \tag{13}$$

where n_{iter}^{event} denotes the maximal number of Newton iterations during the event-handling and $\sigma_{J_{event}}$ denotes the number of required FLOPs for one evaluation of the jacobian of $\mathbf{F}$ with respect to the unknowns during the event-handling. The required FLOPs for table lookup are included in σ_F, σ_J, σ_{dJ} and σ_{event}. With this knowledge the maximal number of FLOPs for one integration step can be computed, while the number of FLOP/s of the realtime target can be easily measured. Since no longer an error bound, but an upper bound for the number of FLOPs for one integration step is given, the term cancellation procedure has to be modified. In the modified term cancellation procedure no simulations are performed. After the ranking the reductions are performed as long as the maximal number of required FLOPs is greater than the upper bound for the FLOPs. Hence, this time no clustering is used, since no verification-simulations are performed and thus clustering would be quite inefficient. Clearly, here a very accurate ranking procedure is demanded, otherwise reductions with a small estimated error leading to a high error could be performed. In this contribution the one-step ranking is simply extended to a three-step ranking, which means that three Newton iterations are allowed. Moreover, the computed ranking value is divided by the number of required FLOPs for one evaluation of the term under consideration. Thus, among reductions with a similar ranking value, those which need many FLOPs are favored.

As can be seen in Eq.12 the dimension of $\mathbf{F}$ has big influence on the number of required FLOPs for one integration step, since the complexity for solving a system of linear equations of dimension k is of order $o(k^3)$. Hence, after each reduction it is checked whether $\mathbf{F}$ got decoupled. More precisely, it is checked whether the DAE system may be written as

$$\begin{bmatrix} \mathbf{F}_1(\mathbf{x}_1, \dot{\mathbf{x}}_1, t) \\ \mathbf{F}_2(\mathbf{x}_2, \dot{\mathbf{x}}_2, t) \end{bmatrix} = \mathbf{0}, \tag{14}$$

where $\mathbf{y}_{out}$ only depends on $\mathbf{x}_1$. In this case $\mathbf{F}_2$ and $\mathbf{x}_2$ can be canceled out of the DAE system.

3 Modeling of the Skid-Steer Loader

The skid-steer loader (Figure 2) is a small high maneuverable vehicle that is usually used in locations where maneuverability and turning space are severely restricted. The high degree of maneuverability is due to their method of steering which is so-called skid steering. They are typically four-wheel drive vehicles with the left-side drive wheels independent of the right-side drive wheels. By having each side independent of the other, wheel speed and direction of rotation of the wheels determine the direction the loader will turn. The drive system on the skid-steer loaders has no mechanical transmission. Instead it uses a combination of hydraulic pumps and motors, the hydrostatic drive system, to drive the wheels as well as the working hydraulic mechanisms. It generally comprises a

diesel engine having its output shaft coupled to a pair of variable displacement pumps. The output of each pump is connected to the respective hydraulic motor, which operates independent chain transmissions and drives on the vehicle. From the modeling point of

Figure 2: Skid-Steer Loader

view, the skid-steer loader comprises mainly the following parts: hydraulic control unit, diesel engine, hydrostatic drive system, working hydraulic mechanisms, tire-road contact and chassis. Due to simplicity there are some limitations concerning the modeling:

- the dynamical effects are only considered in the longitudinal direction.
- the working hydraulic mechanisms are simplified as a rigid body.

Based on these two limitations, the chassis together with the working hydraulics is modeled as a sliding mass. All the other parts are introduced in the following.

3.1 Hydrostatic Drive System

The hydrostatic drive system is constructed using a variable displacement pump to drive a constant displacement motor. In this closed circuit, a charge pump is needed to replenish fluids lost and to provide a minimum pressure in the return line. A low-pressure relief valve is used to control the discharged pressure. There are two more relief valves to limit the pressure in the high pressure line. Furthermore, a pair of check valves are used to restrict the flow direction. In order to model the hydrostatic drive system, a simple *hydraulic library* was built in MathModelica. After modeling all the necessary components, the hydrostatic drive system can be easily obtained. Due to space limitations details are neglected here.

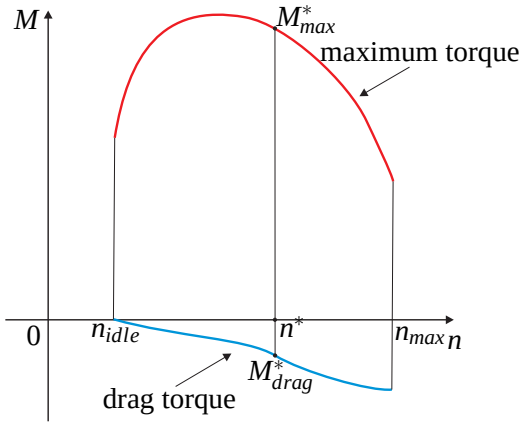


Figure 3: Characteristic Curves of the Engine

3.2 Hydraulic Control Unit

The skid-steer loader is controlled by two joysticks and one foot throttle. The left-hand joystick controls the speed and direction, and the right-hand joystick controls the loader arm. Furthermore, the input signal of the foot throttle can effect the transformation of the input of the two joysticks. The signal itself is only sampled by the trigger in the control unit. In the MathModelica model the left-hand joystick is modeled as two paralleled signal sources namely the driving and steering signal. The right-hand joystick is not necessary to model since the working hydraulic mechanisms are considered as a rigid body. Hence the input signals of the hydraulic control unit are driving and steering signals as well as throttle. The output signals of the control unit control the swivel angles of hydraulic variable pumps in the hydrostatic drive system and the sampled throttle signal to drive the diesel engine. The transformation behavior of the control unit can be described by three characteristic curves. Two curves characterize the relations between swivel angle and the steering and driving signal respectively. Another curve illustrates the effects of driving signals on the steering signals. All these three curves are identified by measurement. In addition, there are some limiters in the control unit to limit the output signals. For example, an amplitude limiter is used to restrict the swivel angle and a rate limiter is used to limit the driving maneuver.

3.3 Engine

The engine used in the skid-steer loader is a diesel engine. It serves to drive the two hydrostatic drive systems. The input of this diesel engine is the foot throttle, which can be normalized in the interval $[0, 1]$.

Since there are no different pedal levels, the foot throttle is proportional to the rotational speed of the engine. The idle rotational speed which is the speed when the throttle is 0 is 1000 rpm and the maximum rotational speed which is the speed when the throttle is 1 is 3200 rpm. The relationship between rotational speed and the generated driving torque can be described by two characteristic curves, namely, the maximum and drag torque with respect to the rotational speed. Figure 3 shows the two characteristic curves. The driving torque M^* when the rotational speed is n^* is calculated by

$$M^*(n^*) = M^*_{max} - M^*_{drag} \tag{15}$$

The dynamical behavior of the engine is also approximated by a PT_2 system. The speed control is realized by a PID-controller.

3.4 Tire

For the reason of skid steering which causes the high dynamical effects in the lateral direction, the normal tire model of a skid-steer loader can be very complex. As only the longitudinal dynamics is considered here, a simplified one dimensional tire model is sufficient to describe these effects. Figure 4 shows the free body

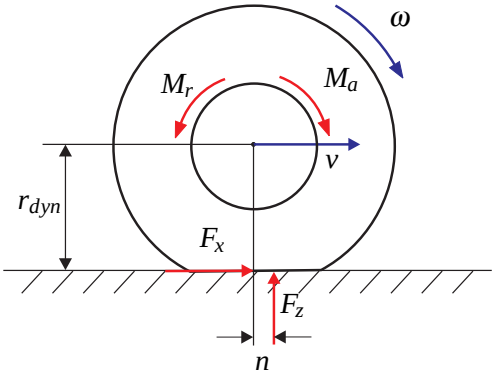


Figure 4: Forces and Moments on the Tire

diagram. All the necessary velocities, forces and moments are depicted. The circumferential velocity of the wheel is

$$v = \omega \cdot r_{dyn} \tag{16}$$

where ω is the angular velocity of the wheel and r_{dyn} is the effective rolling radius. The rotational motion the wheel can be described by

$$J\,\dot{\omega} = M_a - F_x\, r_{dyn} - F_z\, n \tag{17}$$

The wheels are driven by the driving torque M_a. The distribution of the tire load is normally not unit in the contact patch. Thus the supporting force F_z acted not in the middle and generated a rolling resistance torque M_r. The distance from the acting point to the middle is called pneumatic trail n.

$$M_r = F_z\, n \tag{18}$$

The longitudinal force F_x is calculated with the longitudinal slip s_x. The longitudinal slip is defined by

$$s_x = \frac{v_P}{\omega \cdot r} = \frac{\omega \cdot r - v}{\omega \cdot r} \tag{19}$$

There exist already some tire models describing the mathematical function between these two variables. For example, the magic formula tire model with a pure mathematical description based on the experiment results [1], and the physical HSRI tire model with lower computational efforts. In this work the static HSRI tire model was used. The longitudinal force F_x is described in the following equation.

$$F_x = \begin{cases} \dfrac{C_x\, s_x}{1 - s_x} & s_R \le 0.5 \\ \dfrac{C_x\, s_x}{1 - s_x} \cdot \dfrac{s_R - 0.5}{s_R^2} & s_R > 0.5 \end{cases} \tag{20}$$

The longitudinal stiffness C_x is a parameter depending on the properties of the tire. It is defined as the linearization of the force-slip relation at $s_x = 0$ and $\alpha = 0$.

$$C_x = \left.\frac{\partial F}{\partial s_x}\right|_{s_x \to 0} \tag{21}$$

The variable s_R is an indicator to identify the linear or non-linear tire behavior, which can be calculated by

$$s_R = \frac{\sqrt{(C_x\, s_x)^2 + (C_\alpha\, \alpha)^2}}{\mu\, F_z (1 - |s_x|)} \tag{22}$$

The friction factor μ is defined as

$$\mu = \mu_0 (1 - A_s\, v_x \sqrt{s_x^2 + (\tan\alpha)^2}) \tag{23}$$

where, A_s is the adhesion reduction factor, which gives $A_s = 0.011 s/m$ for adhesion coefficients $\mu_0 \in [0.53, 1.05]$. μ_0 can be estimated for different road surfaces. In this section, the introduction of the HSRI tire model enhanced on the mathematical equations. For a more detailed and physical description see [3].

3.5 Driver

The driver modeled here is simply a source of input signals. The output signals from the driver are exactly the same as the inputs of the control unit, namely, steering, driving and throttle signals. Some standard maneuvers were included in this model, such as, the ramp, step and start-stop driving maneuvers.

3.6 Air Resistance

Air resistance describes the influence of the environment. A drag force can be generated by the wind. The equation is as follow.

$$F_{drag} = c_w A \rho \left(v_x - v_{wind}\right)^2 \tag{24}$$

3.7 Overall System

The overall system of the skid-steer loader is obtained by coupling all these sub-systems: drive, hydraulic control unit, engine, hydrostatic drive system, air resistance and the mechanical parts. The acausality of the Modelica language enables the comfort connections between the sub-systems. Figure 5 shows the object diagram of the skid-steer loader in MathModelica.

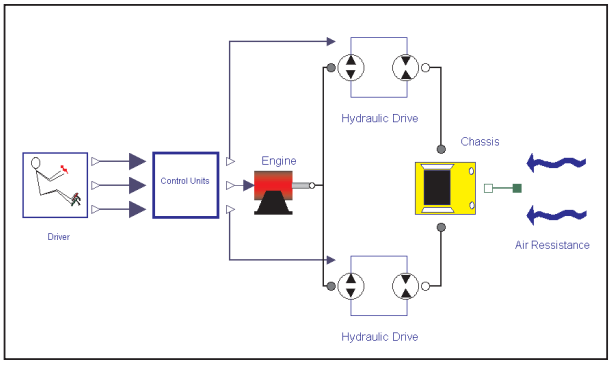


Figure 5: Object Diagram of the Skid-Steer Loader

4 Simulation Results

The symbolic reduction algorithms are implemented in Matlab using the the Maple Toolbox for Matlab. The DAE-system of the previous described model was imported via a MathML interface. Using the Mathematica interface of MathModelica the flat model can be exported to Mathematica and then translated into MathML. The emerging DAE systems are solved using a fixed step BDF method of second order.
In this section, the results for two reductions are given. First the model of the skid-steer loader is reduced using an error bound as stopping criteria. Second the same model is reduced by the extended algorithm for realtime purposes using a maximum number of FLOPs as stopping criteria. Both reductions are performed under a standard start-stop-start-stop straight driving maneuver. Moreover, the longitudinal acceleration is chosen as output variable.

4.1 Reduction with Error Bound

	Original	Error Bound
Number of Equations	69	48
Maximum FLOPs per step	3.42×10^6	1.19×10^6
Maximum absolute error	$\cdots$	0.2851
Maximum relative error	$\cdots$	7.58%

Table 1: Comparison of Original Model and Reduced Model with Error Bound

As presented in Section 2, an error bound for the output variable must be provided for the reduction. Here an error bound of $0.3\,m/s^2$ is set for the longitudinal acceleration a_x. The ranking is computed using the one-step ranking procedure. Negligence of terms is the only reduction technique chosen in this example. Figure 6 shows the nearly overlaying curves of the longitudinal acceleration a_x of the original and the reduced model from the reduced model. It can be seen that the maximum error of $0.2851\,m/s^2$ occurs at the acceleration peaks, where the longitudinal acceleration of the reduced model is slightly higher than the acceleration of the original model. The reduction of

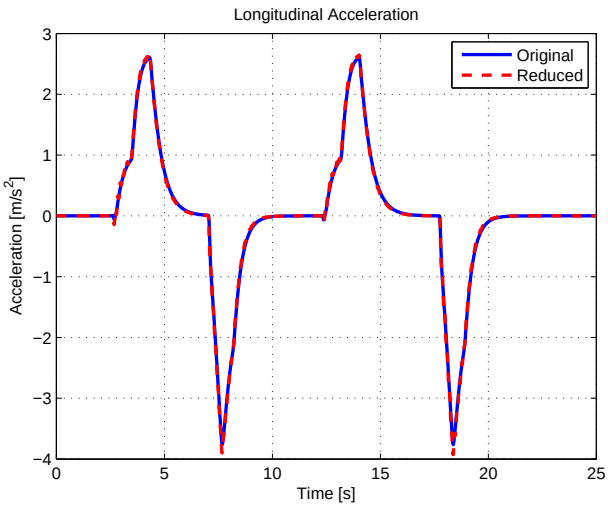
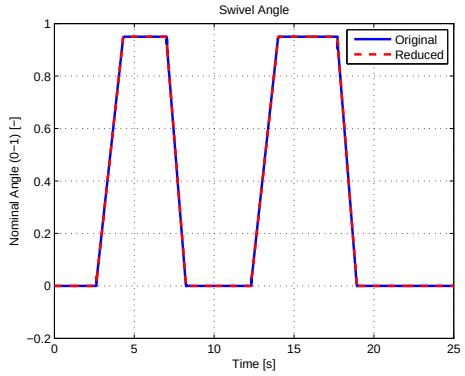
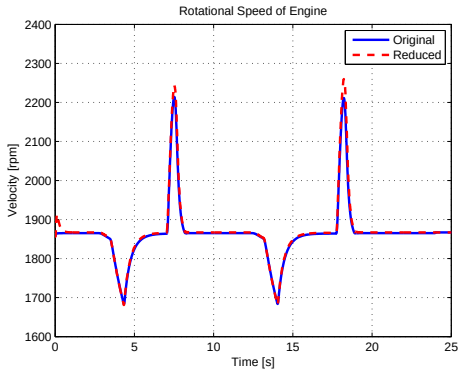


Figure 6: Simulation Results of Output Variable in Reduction with Error Bound

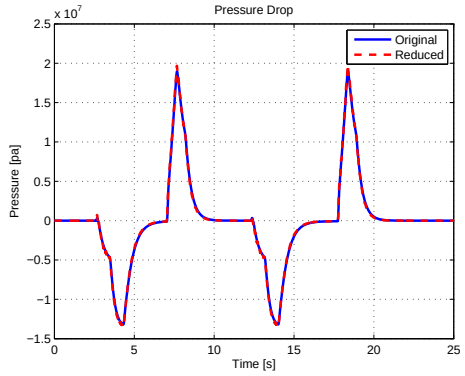
complexity of the DAE-systems can be seen in Table 4.1. The number of equations is reduced from 69 to 48, corresponding to a reduction of approximately 30%. Moreover, the maximum required FLOPs for one integration step is reduced by approximately 65%. Thus, the computation time is accelerated by a factor



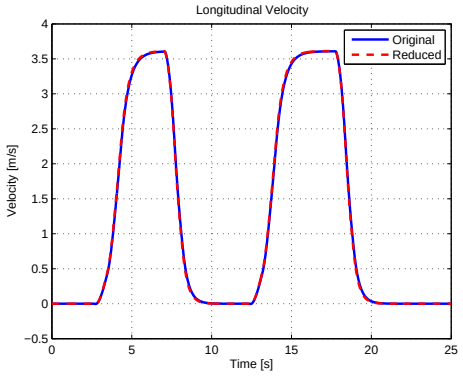
(a) Swivel Angle of Pump



(b) Rotational Speed of Engine



(c) Pressure Drop



(d) Longitudinal Velocity

Figure 7: Simulation Results of System Variables in Reduction with Error Bound

of three.

According to the reduction algorithms described before, only the error of the output variable is considered during the reduction. In Figure 7, some other system variables are plotted. The simulation results of the reduced model of those variables are also very close to the original model. That implies that not only the longitudinal acceleration but also another important dynamical effects are conserved during the reduction.

4.2 Realtime Reduction

In the previous section the original model was reduced until a given error bound was (nearly) reached. Thus, simulating the reduced model will require less time than simulating the original model. In practice models often have to run in realtime environments. For such applications the previously obtained model is more or less worthless, since no worst case runtime for one integration step is known. In this section the original model is reduced in order to obtain a model, which can be simulated in realtime on a given realtime target in a given realtime cycle. Hence, instead of providing an error bound, a realtime target as well as a realtime cycle is provided for the realtime reduction. The realtime target is assumed to be able to perform 1×10^9 FLOP/s, which corresponds roughly to Pentium III. The realtime cycle is chosen as $2\,ms$. Thus, 2×10^6 are the maximal available FLOPs for one integration step. The results from the reduced model for

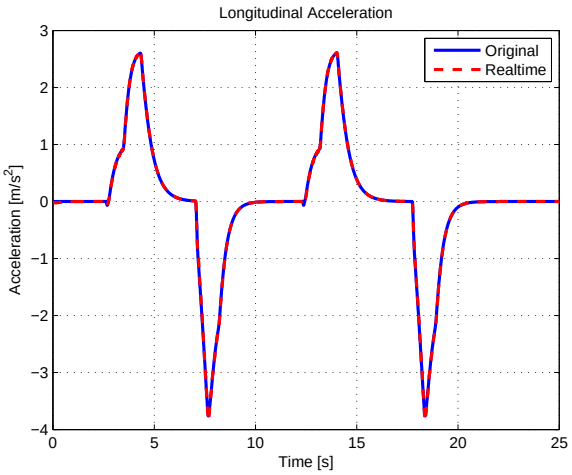
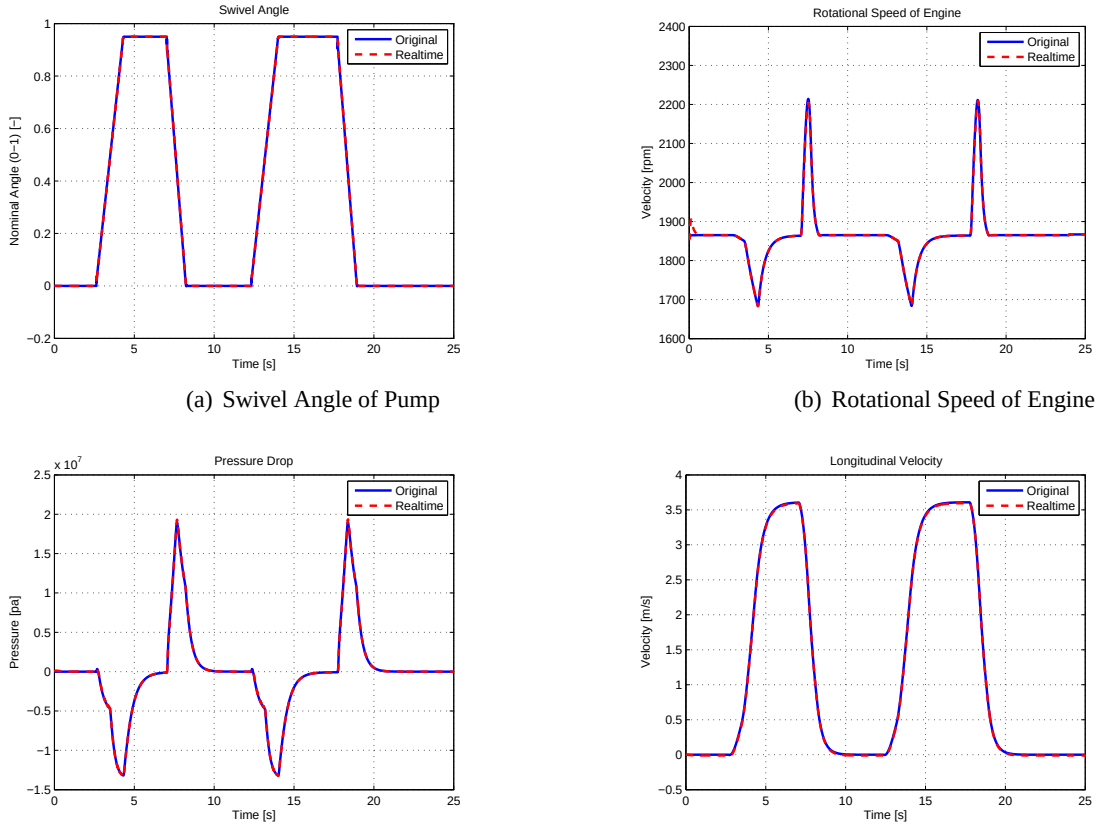


Figure 8: Comparison of Realtime Reduction

realtime purpose are shown shown by the almost iden-



(a) Swivel Angle of Pump

(b) Rotational Speed of Engine

(c) Pressure Drop

(d) Longitudinal Velocity

Figure 9: Simulation Results of System Variables in Reduction for Realtime

tical curves in 8. Nevertheless they are quite different from the previous reduced model. Again the maximal absolute error of $0.1111\,m/s^2$ occurs at the the acceleration peaks, but this time the longitudinal acceleration of the reduced model is slightly lower than the acceleration of the original model. The FLOPs for one integration step is reduced by a factor of 1.8 to 1.92×10^6. Noteworthy, reducing the original model by a factor of 3 is possible without significant loss of accuracy as can be seen in the previous section. Figure 9 shows other relevant system variables. It can be observed that the reduced model is in very good agreement with the original model for all shown system variables. Therefore, again the relevant physical effects are conserved.

5 Conclusion and Outlook

In this contribution a reduction algorithm is extended in order to generate models for realtime purposes. While up to now an error bound was used as stopping criteria, the extended algorithm uses a maximum number of flops for one integration step as stopping criteria. Furthermore, in this contribution the new approach is applied to the model of a construction machine. The generated model is in quite good agreement with the original model at a computational effort, which is considerably lower. In this contribution only the longitudinal dynamics is considered. In the near future the model will be extended by lateral and vertical motion. Moreover, the implementation of the generated models on a realtime target is part of current work.

The presented reduction method takes into account only one scenario. This strongly limits the guaranteed validity of the model. In [8] it has been tried to overcome this drawback by using interval arithmétics. Unfortunately, this approach works only for rather simple systems. Therefore, the scenario has to be chosen quite carefully and can thus be a worst case scenario for example. In future works it shall be investigated how one or multiple scenarios can be chosen systematically such that the desired effects remain.

Since a quite accurate ranking is required for the realtime reduction, here the one-step ranking was extended to a three step ranking. Currently, a reliable ranking procedure based on a sensivity analysis is developed. The new ranking procedure is expected to be

	Original	Realtime
Number of Equations	69	57
Maximum FLOPs per step	3.42×10^6	1.92×10^6
Maximum absolute error	$\cdots$	0.1111
Maximum relative error	$\cdots$	2.58%

Table 2: Comparison of Original Model and Reduced Model for Realtime Purpose

more time efficient, since modern solvers like DASPK offer a sensitivity analysis during the integration and hence the ranking can be computed together with the reference solution.

References

[1] E. Bakker, L. Nyborg, and H. Pacejka. Tyre modelling for use in vehicle dynamics studies. In *Society of Automotive Engineers international congress and expo*, volume 23, 1987.

[2] C. Borchers. Symbolic behavioral model generation of nonlinear analog circuits. *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on [see also Circuits and Systems II: Express Briefs, IEEE Transactions on]*, 45(10):1362–1371, 1998.

[3] H. Dugoff, P.S. Fancher, and L. Segel. Tire Perfomance Characteristics Affecting Vehicle Characteristics to Steering and Braking Control Inputs. Technical report, Highway Safety Research Institute, University of Michigan, 1969.

[4] H. Elmqvist, D. Brück, and M. Otter. Dymola-User's Manual. *Dynasim AB, Research Park Ideon, Lund, Sweden*, 1995.

[5] P. Fritzson and V. Engelson. Modelica-a unified object-oriented language for system modeling and simulation. *Lecture Notes in Computer Science*, 1445:67–90, 1998.

[6] P. Fritzson, J. Gunnarsson, and M. Jirstrand. MathModelica-an extensible modeling and simulation environment with integrated graphics and literate programming. *Proceedings of the 2nd International Modelica Conference*, pages 18–19, 2002.

[7] H. Lundvall, P. Fritzson, and B. Bachmann. Event Handling in the OpenModelica Compiler and Runtime System. *Linköping University Electronic Press*, 2006.

[8] L. Mikelsons and T. Brandt. Symbolic Model Reduction for Interval-Valued Scenarios. *To appear in Proceedings of the ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2009.

[9] L. Mikelsons, M. Unterreiner, and T. Brandt. Generation of Continuously Adjustable Vehicle Models using Symbolic Reduction Methods. *To appear in ECCOMAS Multibody Dynamics*, 2009.

[10] R. Sommer, T. Halfmann, and J. Broz. Automated behavioral modeling and analytical model-order reduction by application of symbolic circuit analysis for multi-physical systems. *Simulation Modelling Practice and Theory*, 2008.

[11] T. Wichmann. Computer aided generation of approximate DAE systems for symbolic analog circuit design. *Proc. Annual Meeting GAMM*, 2000.

[12] T. Wichmann. Transient Ranking Methods for the Simplification of Nonlinear DAE Systems in Analog Circuit Design. *PAMM*, 2(1):448–449, 2003.

[13] T. Wichmann. Symbolische Reduktionsverfahren für nichtlineare DAE-Systeme. *Berichte aus der Mathematik. Shaker Verlag, Aachen, Germany*, 2004.