Multi-Level Vehicle Dynamics Modeling and Export for ADAS Prototyping in a 3D Driving Environment

Róbert Lajos Bücs, Juan Sebastián Reyes Aristizábal, Rainer Leupers, Gerd Ascheid
Institute for Communication Technologies and Embedded Systems, RWTH Aachen University, Aachen Germany
E-Mail: {*buecs,reyes,leupers,ascheid*}*@ice.rwth-aachen.de*

***Abstract*—Advanced Driver Assistance Systems (ADAS) have evolved into comprehensive hardware and software driven components with exponentially growing complexity. Model-Based Design (MBD) is traditionally employed to accelerate the design-validation-refinement cycle of such systems by reducing the occurrence of late nonconformities. However, MBD testing facilities can not provide a fully realistic verification environment compared to real test drives. Moreover, even road tests are limited, e.g., to reproduce driving situations and to test applications likely to cause property damage. Thus, this paper proposes to utilize driving simulators alongside MBD tools to achieve a fully virtual ADAS design and test environment. To put this idea in practice, the multi-level Vehicle Component Modeling and eXport (VCMX) framework is presented as the main contribution of this work. Finally, the combined tool support is evaluated by designing two ADAS from scratch so to highlight the advantages of utilizing VCMX especially in early prototyping stages of such applications.**

I. Introduction

Over the last decades, automotive control systems have evolved to sophisticated digital hardware (HW) and software (SW) dominated components. These systems consist of *(i)* a multitude of peripheral devices (e.g., sensors, cameras) capturing environmental conditions *(ii)* HW/SW control units fusing and processing the acquired information and *(iii)* mechatronic actuators applying regulatory actions. Such Advanced Driver Assistance Systems (ADAS) cover nowadays an increasing number of areas ranging from infotainment, over safety-critical applications, heading all together towards autonomous driving. To cope with the resulting immense design complexity, standards have been defined, e.g., the European standard for design methodology for mechatronic systems [1], and its extension for ADAS [2]. These guidelines usually define a V-model to capture requirements, set specifications, design and implement subsystems, and finally validate and integrate them into a target system, as shown in Fig. 1. However, this flow includes an iterative *design-validation-refinement cycle* resulting in an extended development process and thus increased cost.

Model-Based Design (MBD) tools are advised to be used to accelerate these iterations [3] by enabling early algorithmic prototyping, alongside in-tool testing and certified code generation facilities. Moreover, MBD tools further refine test steps via *X-In-the-Loop* (XIL) methods, as shown in Fig. 1. Particularly, *Model-*, *Software-* and *Hardware-In-the-Loop* (MIL, SIL, HIL) techniques assist to reduce late nonconformities. Yet, MBD tools have several limitations, e.g., unrealistic inputs and outputs (I/O) compared to ADAS evaluation via test drives. Moreover, even road tests have drawbacks, e.g., to reproduce situations, especially crucial if spurious failures occur.

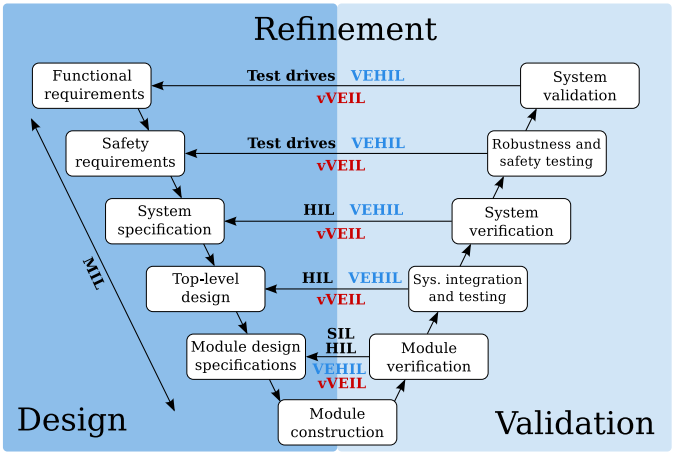


Fig. 1: V-Model for designing safety-critical automotive systems [2].

Driving Simulators (DSs) address these limitations via a realistic and deterministic simulation environment, ensuring exactly reproducible driving scenarios. Additionally, DSs allow to test and refine ADAS under various traffic and environmental conditions using even different vehicle types (e.g., cars, trucks). Finally, DSs can also interactively involve a *Driver In the Loop* (DIL) for ADAS evaluation, so to capture real human reactions instead of purely artificial I/O.

Multi-Domain Co-Simulation: To join such distinct simulation ecosystems, models need to be connected across tool and domain boundaries. For instance, simulation-driven ADAS design requires *(i)* fetching virtual sensor data from the environment of a DS *(ii)* processing data and executing the target algorithm prototyped in the MBD tool and *(iii)* applying regulatory actions on the actuator models of the virtual vehicle within the DS. This setup requires different heterogeneous vehicular subsystems to be interconnected. Tool-agnostic *multi-domain co-simulation* standards have been defined for such purposes to fully cover inter-domain interactions and to overcome the inflexibility of point-to-point coupling solutions.

Contributions: This paper proposes to combine the preceding tools and techniques to accelerate control algorithm and controlled system co-design and validation via virtualization. We refer to this collective approach as *virtual Vehicle Environment In the Loop* (vVEIL). To put vVEIL in practice, the *multi-level Vehicle Component Modeling and eXport* (VCMX) framework is presented as the main contribution of this work. Lastly, the vVEIL methodology is evaluated via VCMX by designing a multi-level vehicle dynamics library to support early refinement of two ADAS algorithms, presented later on.

978-1-5386-1526-3/17/$31.00 ©2017 IEEE

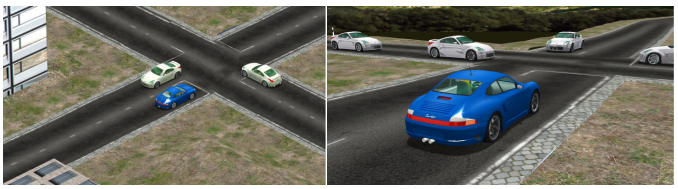


Fig. 2: Speed Dreams 2 driving simulator adapted for urban driving.

Tool Selection: MBD tools, driving simulators and co-simulation standards have been carefully compared to select the most suitable combination for this work. The requirements for MBD tools include the availability of advanced modeling semantics, a broad predefined block set and an advanced and expandable code generator. Both open-source (e.g., OpenModelica [4], JModelica [5], Scicos [6], Scilab [7], Scicoslab [8]) and commercial tools (e.g., Dymola [9], SCADE Suite [10], TargetLink [11], MATLAB/Simulink [12]) have been examined, and Simulink has been selected as it fulfills all posed requirements, and as its configurable code generation facility has been found to be the most powerful and versatile.

The requirements for driving simulators have been an accurate and adjustable physics engine, configurable connectivity and the possibility for ADAS virtual test drives. Similarly, open-source (e.g., VDrift [13], OpenDS [14], TORCS [15], Speed Dreams 2 [16], StuntRally [17], Rigs Of Rods [18]) and commercial tools (e.g., TRS [19], Pro-SiVIC [20], carSIM [21], VTD [22], CarMaker [23], ASM [24]) have been evaluated. From these, *Speed Dreams 2* (SD2) has been selected since a branch of this originally racing game (shown in Fig. 2) has been adapted for urban driving as well as multi-domain co-simulation in [25], solely for ADAS prototyping.

Finally, a multi-domain co-simulation standard has been selected among CosiMate [26], the High Level Architecture [27] and the *Functional Mock-up Interface* (FMI) [28]. Herein FMI was chosen, an open-source, tool-independent specification, as it is considered the de facto simulation interoperability standard by the automotive industry. The objective of FMI is interconnecting hierarchical systems, and supporting a fully simulation-based development and testing environment. FMI defines a master/slave concept, where the *FMI master* is a module responsible for synchronization and data exchange between the involved *FMI slaves*, that encapsulate a certain co-simulation model following the FMI programming interface.

II. Related Work

Early projects, e.g., FASIM_C++ [29], have paved the way for modeling vehicle dynamics. Later on, such systems have reached the maturity to assist ADAS development [30]. Finally, works like [31] have laid the foundations of vehicle dynamics used in driving environments for ADAS prototyping.

Several research efforts have been directed towards extending model-based system design. Gietelink et al. [2] proposed the previously shown V-model (Fig. 1) to improve this process. Beyond conventional refinement techniques, authors propose *VEhicle-Hardware-In-the-Loop* (VEHIL) validation, including an actual test bench vehicle in the evaluation. However, VEHIL is greatly expensive, especially when aiming only at early prototyping. In contrast, the vVEIL methodology, proposed in this paper, overcomes these limitations by full virtualization.

Significant scientific efforts have also been invested in the combined utilization of the previously mentioned approaches. Authors of [32] and [33] propose to assemble and generate a reconfigurable DS from an MBD tool tailored to evaluate specific ADAS applications. Yet, the interconnection of models and their interfaces requires significant manual effort which is error-prone, greatly demanding and inflexible for changes, considering the amount of models. Also, we strongly believe that a clear separation needs to be present between modeling vehicle dynamics and establishing the driving environment. Thus, the system designer is not unnecessarily burdened with creating the complete simulation environment from scratch.

In contrast to previous works, this paper proposes to fully virtualize the design-validation-refinement cycle of ADAS. The proposed VCMX framework incorporates model-based design alongside fully automated model export, interconnection and seamless integration facilities for co-simulation. The resulting setup ensures environmental consistency between the MBD and DS, and enables models to be integrated in a virtual vehicle to perform ADAS evaluation via virtual test drives. Furthermore, multi-level vehicle dynamics design is supported by VCMX, enabling the possibility to trade off simulation execution speed and modeling accuracy. Finally, as to the best knowledge of the authors, these aspects have never been considered together by any other research effort.

III. The VCMX Framework and Design Flow

A. Initial Co-Simulation Package Architecture Definition

As the initial step towards VCMX, the fundamental architecture of a co-simulation package has been laid down (later to be generated from Simulink). First, universal data structures have been defined to prevent ambiguities and type mismatches. Moreover, a dedicated file has been created to store car configurations via fixed settings of vehicle dynamics modules (e.g., material properties, dimensions). This mechanism foresees ADAS reuse by enabling experimentation with different component configurations and vehicle variants. Additionally, means for data exchange and safe encapsulation have been added to support inter-submodule communication. A further mechanism has been added to include multi-level implementations representing models on different accuracy levels. To plug the prototype seamlessly into a multi-domain co-simulation, the complete infrastructure has been encapsulated as an FMI slave, also referred to as a *Functional Mock-up Unit* (FMU). Lastly, to ensure data exchange with other co-simulation participants, the data encapsulation system has been extended to access selected module parameters externally.

B. Simulink-Based Custom Code Generation Facility

The next goal was to include various vehicle dynamics models and to generate the previously designed co-simulation package automatically from within Simulink. The code generation facility of Simulink, shown in Fig. 3, starts with invoking Simulink Coder, that compiles Simulink models (`*.slx`) into

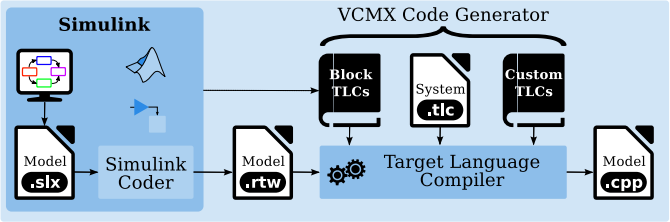


Fig. 3: Customizable code generation process of Simulink.

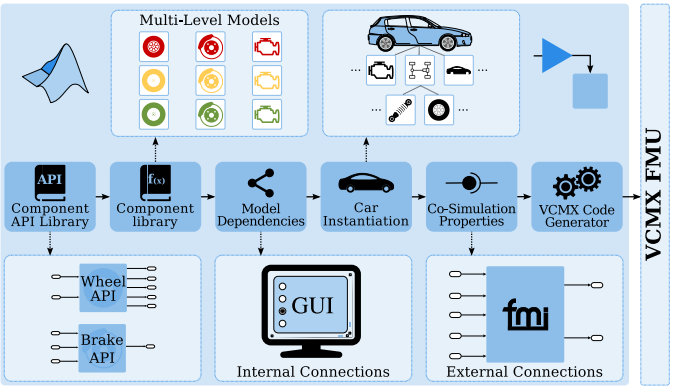


Fig. 4: VCMX multi-level component design flow.

an intermediate representation (`*.rtw`) that contains top-level and block-specific information. Afterwards, MATLAB's *Target Language Compiler* (TLC) converts this description into executable code (C/C++). However, this transformation can be externally controlled via (`*.tlc`) files. The System TLC is the initiator of the conversion process defining target-specifics and responsible of executing Block TLCs that specify how blocks are translated into code. Finally, to create libraries beyond the predefined set, Custom TLCs can be created.

C. VCMX Component Modeling Paradigm & Code Generator

The customizable code generation mechanism needs to be augmented with a comprehensive modeling flow to achieve the desired generation of the co-simulation infrastructure. Shown in Fig. 4, this flow first proposes a hierarchical separation of vehicle dynamics modules. This division also allows to *(i)* disconnect the mathematical representation of models from their external structure and *(ii)* to consistently and unambiguously interconnect submodels by their predefined I/O interfaces.

Depicted in Fig. 4, a Component API Library has thus been created via `*.slx` templates specifying I/O attributes of vehicle dynamics. These templates define different model types to be later identified (via meta-information in the `*.rtw`) and exploited by custom TLCs during code generation. The VCMX flow also grants API extensibility to incorporate additional I/O, create new models or restructure existing ones via MATLAB's model parameters configuration GUI, extended in this work.

Relying on these API templates, model functionality can be inserted. As shown in Fig. 4, support has been given for multiple implementations of individual component types, with the custom code generator adding each of these to the exported co-simulation package. The resulting Multi-Level Model library allows to interchange components to achieve an optimal trade-off between execution performance and simulation accuracy.

Illustrated in Fig. 4, Model Dependencies need to be resolved afterwards by connecting the I/O signals of vehicle dynamics components by their API definitions only, thus independently from their actual implementation. This way, Car Instantiation is automatically accomplished resulting in a multi-level virtual vehicle instance. Subsequently, the user can specify Co-Simulation Properties, e.g., selecting top-level I/O signals to be exposed towards the FMI-based co-simulation.

The VCMX Code Generator concludes all these steps by processing the produced meta- and model information through a set of custom TLCs created within this work. First, a custom system TLC (`VCMX.tlc`) is invoked, orchestrating the code generation of the configured simulation infrastructure. This module processes application-specific parameters, multi-level properties and predefined module I/O, among others, via the `*.rtw` file. Afterwards, the system TLC invokes target-specific TLCs to continue with the code generation process:

(i) TLC for behavioral code generation of individual model instances. This unique TLC is consistently defined to generate the step functions of arbitrary models. Furthermore, it also includes automated signal routing for inter-module data exchange and safe data encapsulation.
(ii) Glue code TLCs (e.g., headers, FMI wrapper)
(iii) Data handling TLCs (e.g., mappers, loggers, parsers).
(iv) Miscellaneous TLCs (e.g., FMI model description and component property descriptor files).

The resulting custom code generation facility enables "one-click" code export and integration of the VCMX FMU co-simulation package, ready to be instantiated by an FMI master.

IV. Multi-Level Vehicle Dynamics Library

A necessary prerequisite to improve ADAS refinement is to ensure a realistic simulation environment. The VCMX flow and the corresponding code generation facility have been explicitly tailored to guarantee this requirement and to accelerate as well as to facilitate the design process. To highlight this property, a multi-level vehicle dynamics library has been created via VCMX within this work. Herein the vehicle dynamics of SD2 have been partially re-modeled also to be able to verify modeling consistency. However, since this physics engine is relatively simplistic, to achieve a more accurate behavior, additional enhanced models have been created via the multi-level modeling capability of VCMX.

For model description, the mathematical symbol notation of vehicle dynamics has been organized into $modules$, each containing a set of $signals$ (denoting variables) and $properties$ (representing constants), all defined as tuples. The used module tuples for the notation of the multi-level models are: aerodynamics (A), brakes (B), brake system (BS), chassis (CH), road plane (R), user controls (U) and wheels (W). Furthermore, to specify the position of components with multiple instances (e.g., brake, wheel) unambiguously, an additional *location subindex* (i) has been defined. To clarify the mathematical notation, the symbol T_{1l}^{B} denotes the output temperature signal (T) of the brake module (B) located at the front axle (1) on the driver side, left (l) on the virtual vehicle.

TABLE I: Mathematical symbol notation of the brake model.

Symbol	Unit	Description
a_i^B	m^2	Operating piston cross section area
Φ_i^B	m	Disc outer diameter
cp^B	kJ/kg K	Disc material specific heat
m^B	kg	Disc mass
η^B	-	Dissipated heat share of the disc-shoe contact
ζ_i^B	-	Stopped friction coeff. between disc and pad (dry)
$\varkappa_i^B$	-	Rolling friction coeff. between disc and pad (dry)
ϱ_i^B	-	Stopped friction coeff. between disc and pad (wet)
ς_i^B	-	Rolling friction coeff. between disc and pad (wet)
μ_i^B	-	Instant disc-pad friction coefficient
κ^B	-	Decrease ratio of μ wrt. temperature variation
l_i^B	m	Brake pad cross section height
M_i^B	N m	Brake moment
P_i^{BS}	kPa	Output pressure of the brake system at location i
T_i^B	K	Brake disc surface temperature
ω_i^W	rad/s	Angular velocity of the wheel at location i
$\dot{Q}_{in}^B$	W/s	Disc heat up ratio
$\dot{Q}_{out}^B$	W/s	Disc heat dissipation ratio
$\dot{Q}_{cond}^B$	W/s	Conductive heat transfer
$\dot{Q}_{conv}^B$	W/s	Convective heat transfer
$\dot{Q}_{rad}^B$	W/s	Radiative heat transfer

A. Multi-Level Brake Modeling

Disc brakes have been first selected to be re-modeled from the SD2 vehicle dynamics. These components consist of a disc rotating with the wheel, while a stationary shoe applies normal force against it. This generates frictional forces between both surfaces resulting in brake torque. Table I shows the symbols for the mathematical notation of the brake model.

The SD2 reference model calculates brake torque as:

$$M_i^B = \frac{1}{2} * P_i^{BS} * \Phi_i^B * a_i^B * \varkappa_i^B \tag{1}$$

After examining the source code of the SD2 reference model, it has been determined that the disc temperature (T_i^B bound to $x \in \mathbb{R} \mid 0 \leq x \leq 1$) is only used for visual rendering representation, with no influence on braking capabilities.

The enhanced accuracy model, developed via VCMX, incorporates thus an instantaneous friction coefficient μ_i^B, affected by the temperature, as well as the intermediate medium and the relative velocity between contact surfaces [34]. The influence of precipitation (denoted by W^U $x \in \mathbb{R} \mid 0 \leq x \leq 1$, 0 meaning dry and 1 fully wet conditions) on the solid and fluid friction coefficients is represented using a linear interpolation between the boundary conditions $\zeta_i^B, \varkappa_i^B, \varrho_i^B, \varsigma_i^B$ that can be expressed (following the calculation and the notation in [35]):

$$\bar{\zeta}_i^B = \zeta_i^B + (\varrho_i^B - \zeta_i^B) * W^U \tag{2}$$

$$\bar{\varkappa}_i^B = \varkappa_i^B + (\varsigma_i^B - \varkappa_i^B) * W^U \tag{3}$$

The upper-score symbols refer to individual friction coefficients after interpolation. Moreover, static and fluid frictional conditions have also been modeled depending on the relative velocity of contact surfaces [34]. The final values are selected depending on the wheel's angular velocity ω_i^W (incorporating the relative velocity between disc and shoe, according to [35]):

$$\mu_i^B = \begin{cases} \omega_i^W \leq 3 \text{ [rad/s]} & \rightarrow \mu_i^B = \bar{\zeta}_i^B \\ \omega_i^W > 3 \text{ [rad/s]} & \rightarrow \mu_i^B = \bar{\varkappa}_i^B \end{cases} \tag{4}$$

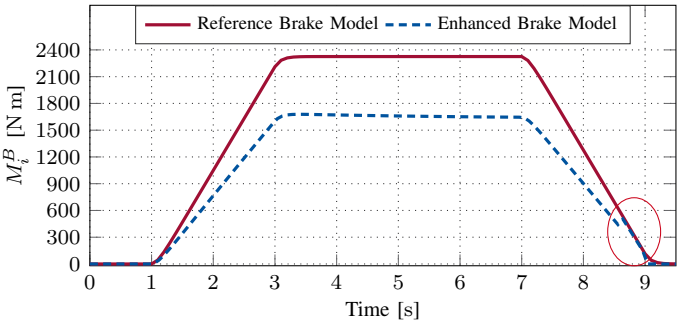


Fig. 5: Output torque: SD2 reference vs. enhanced disc brake.

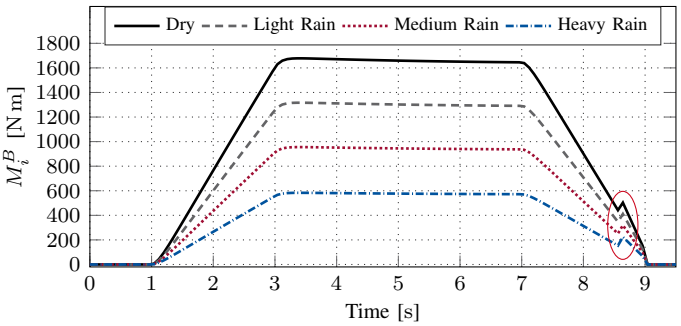


Fig. 6: Output brake torque vs. precipitation (enhanced model).

Afterwards, the effect of T_i^B on μ_i^B has been also modeled:

$$\mu_{i\,(T_i^B)}^B = \mu_i^B - \kappa^B * \Delta T_i^B \tag{5}$$

Herein the calculated instant disc-pad friction coefficient is corrected with regards to temperature variations. Finally, the brake torque can be calculated (based on [36]) as:

$$M_i^B = -\frac{(\Phi_i^B - l_i^B)}{2} * \frac{\omega_i^W}{\mid \omega_i^W \mid} * a_i^B * P_i^{BS} * [\mu_i^B - \kappa^B * \Delta T_i^B] \tag{6}$$

The brake torque is thus influenced by the area of contact surfaces, direction of rotation, applied brake system pressure and the corrected friction coefficient. Moreover, the brake temperature has been modeled defining a control volume around the disc. Herein the heat balance can be written as:

$$\dot{Q}_{in}^B = \omega_i^W * M_i^B * \eta^B \tag{7}$$

$$\dot{Q}_{out}^B = \dot{Q}_{cond}^B + \dot{Q}_{conv}^B + \dot{Q}_{rad}^B \tag{8}$$

$\dot{Q}_{in}^B$ corresponds to the heat flow transferred to the disc, produced by friction forces between the contact surfaces. Accordingly, $\dot{Q}_{out}$ denotes the heat flow from the disc to the surroundings by means of conduction, convection and radiation. Herein the wheel hub has been simplified as the unique conductive heat sink, with the ambient temperature T^U as its initial value. Finally, by neglecting the thermal radiation from the surroundings, the temperature can be expressed [37]:

$$T_i^B = \frac{\sum(\dot{Q}_{in}^B - \dot{Q}_{out}^B)}{cp^B * m^B} + T^U \tag{9}$$

These refinements drastically change the model's behavior. Fig. 5 presents the comparison of output torques of both brake models, using consistent inputs. Generally, the enhanced model generates less brake torque than the reference. This is explained by *(i)* the corrections of the friction coefficient on solid or fluid friction conditions and *(ii)* the improvement

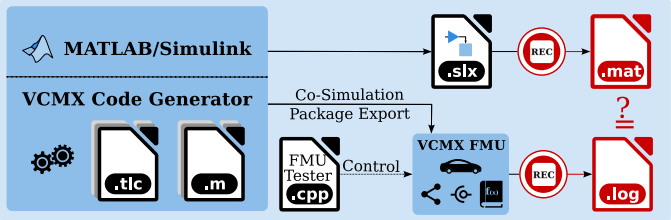


Fig. 7: Consistency test setup: Simulink vs. VCMX.

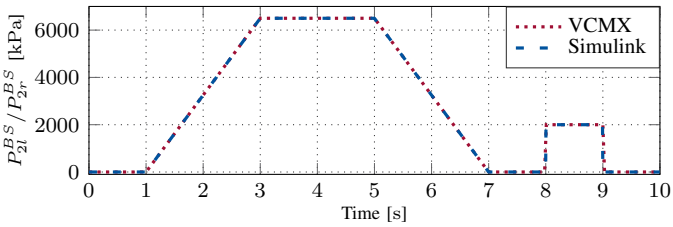


Fig. 8: Comparison of brake system pressure outputs (rear left/right).

of the brake pad geometry by realistic dimensions. Finally, the enhanced model shows a slight torque reduction over the braking process ($3-7\,\mathrm{s}$) that reflects the drop of the friction coefficient because of increasing disc temperature (Eq. 5).

Fig. 6 shows the test of the enhanced model under different precipitation. As expected, the wetter the contact surfaces, the less the friction coefficient, as well as the torque output becomes. Yet, a discontinuity can be observed around $8.6\,\mathrm{s}$ that is explained with the selection between solid $\bar{\zeta}_i^B$ and fluid $\bar{\varkappa}_i^B$ friction conditions (Eq. 4). This transition is softer in reality, but modeling such frictional conditions precisely (via specific *Stribeck* curves [34]) goes beyond the scope of this work.

The upcoming sections present further multi-level model enhancements performed via VCMX. The models are based on firm theoretical approaches, and have been described and formalized with the same mathematical thoroughness. Nevertheless, as the full description of all details would go beyond the extent of this paper, the characterizations are summarized.

B. Multi-Level Brake System Modeling

The brake system usually consists of a pedal, booster, master hydraulic cylinder and brake lines connected to the brakes. The pressure acting on the rear axle is traditionally reduced by a limiter valve to enhance maneuverability. The pressure ratio between axles is modeled via the *brake repartition coefficient*.

The SD2 reference model includes a parallel brake system with two independent hydraulic circuits acting on both wheels of an axle. In this model, the output pressure is scaled by the repartition coefficient. The model also includes a separate handbrake system simplified as hydraulic. Finally, the pressure acting on the rear brakes is the greater among the repartitioned brake system pressure and the handbrake pressure.

The enhanced accuracy model includes a step brake cylinder [38], where separate piston chambers supply hydraulic pressure to the brake circuits. A specific delay has also been added to model the build up, transmission and rise times until maximum pressure [39]. Ultimately, a diagonal system has been implemented [38], where each pressure line acts on one of the front wheels and the diagonally opposite rear wheel.

C. Multi-Level Aerodynamics Modeling

As the **SD2 reference model** mimics race cars via aerodynamic forces caused by rear and front spoilers, this model has not been replicated in VCMX. Instead, a theoretical approach has been followed in the **enhanced accuracy model** targeting urban vehicles. Herein drag, side and lift forces, and their effects on turning moments to the chassis have been modeled.

D. Multi-Level Chassis Modeling

The chassis is where all affecting forces act combined to control the motion of the car depending on their equilibrium.

The SD2 reference model uses these forces to model six degrees of freedom motion, i.e., translational (forward/backward, left/right, up/down) and rotational (yaw, pitch, roll) movement. Yet, several unrelated functions are present in this module, e.g., fuel consumption estimation and rendering tasks. Moreover, the model stabilization around boundary conditions does not rely on any physical background. Thus, the reference chassis model has not been re-modeled via VCXM either.

The enhanced accuracy model incorporates all acting forces and applies a transformation between the undercarriage (road plane) and the center of gravity (CoG - car chassis) coordinate systems. The total weight of the vehicle has also been added, acting at the CoG. As discussed earlier, aerodynamic forces (including lift) are also included, resulting in turning moments. Furthermore, advanced effects, such as the energy required to overcome the inertia of rotating elements along the power train (e.g., crankshaft, gearbox, shafts, differential, wheels), has also been added to this model following [40].

V. Framework Evaluation

This section evaluates the VCMX modeling flow, code generation infrastructure and the resulting multi-level co-simulation package. As stated earlier, one of the goals of this work is to facilitate early ADAS prototyping by means of virtualization. To achieve this, firstly, modeling consistency needs to be guaranteed between all simulation environments (Simulink, SD2 and VCMX). Subsequently, vehicle dynamics accuracy enhancements can be performed to be able to adapt and refine ADAS behavior. Finally, the complete co-simulation infrastructure can be employed to prototype ADAS applications in virtual field tests. All these properties are evaluated and corresponding experiments are presented in this section.

A. Model Consistency Verification: Simulink vs. VCMX

At first, it had to be shown that models created in Simulink are maintained coherently throughout the VCMX code generation process. To ensure this, all designed models have first been executed in Simulink. Subsequently, the VCMX FMU (including the same models) has been exported via the code generator and executed by a stand-alone tester program. The results have been recorded and the log files compared in both cases, as shown in Fig. 7. Similar test runs have been conducted for all designed models within this work. On a

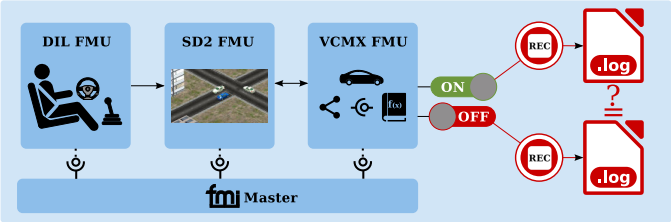


Fig. 9: Consistency test setup: VCMX vs. SD2 reference models.

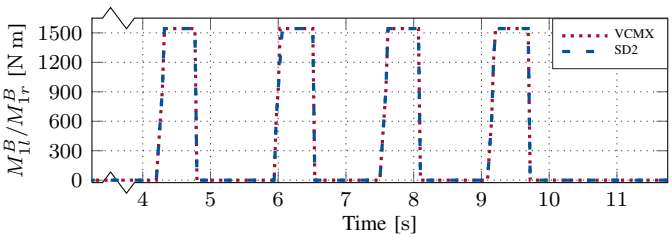


Fig. 10: Comparison of brake moment outputs (front axle left/right).

TABLE II: Co-simulation test scenario accuracy settings.

Test	Aerodynamics	Brake	Brake System	Chassis
A	reference	reference	reference	reference
B	reference	reference	enhanced	reference
C	reference	enhanced	enhanced	reference
D	enhanced	enhanced	enhanced	reference
E	enhanced	enhanced	enhanced	enhanced

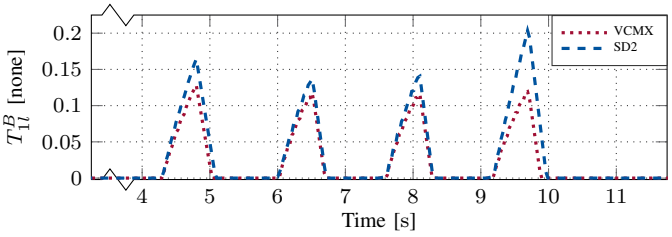


Fig. 11: Comparison of brake temperature outputs (front axle left).

final note, all test pairs have been executed using predefined, consistent input signals, so to fully isolate model behavior.

One of these comparisons can be seen in Fig. 8. Herein the rear brake system pressure outputs are shown first applying the user's brake and then the handbrake command. As seen in the figure, the results obtained from Simulink and the data acquired from the VCMX FMU show 100% match. Such comprehensive consistency tests have been conducted for every model implementation inside the VCMX package. Thereby, it has been verified that models developed via the VCMX code generation facility, fully match their MBD counterparts.

B. Model Consistency Verification: SD2 vs. VCMX

As discussed in Section IV, several SD2 vehicle dynamics reference models have been replicated following the VCMX design flow. The goal now was to verify that these models show the same behavior when utilized in a co-simulation alongside the driving simulator. As shown in Fig. 9, a dedicated co-simulation scenario has been assembled for this purpose. Herein specific signals of dedicated reference vehicle dynamics models have been recorded (into `*.log` files) once including and once excluding their counterpart within the exported VCMX co-simulation package. To ensure absolute consistency of the comparison, input commands have been recorded from a human driver and encapsulated in a DIL maneuver FMU also plugged into the co-simulation. The results obtained via co-simulation are only accounted after 2 s of simulation time, as the initialization of the driving simulator is performed prior. All signal records have been expected to be fully equal, e.g., the brake torque outputs shown in Fig. 10.

However, inconsistencies have been encountered in the brake temperature comparison (Fig. 11), although in the SD2 reference model brake temperature does not have any physical meaning (with its unit marked as *none*). By further examination, it has been found that these differences are the consequence of indeterminism within the SD2 road model. The reason is that road characteristics are assigned in SD2 randomly at simulation runtime. This behavior affects the wheel angular velocity ω_i^W that is an input for the calculation of T_i^B (as described in Section IV-A). This phenomenon in the driving simulator poses a limitation to be addressed in future work. Yet, model consistency has been achieved in VCMX where this external effect does not influence vehicle dynamics. Finally, a seamless integration of MBD models alongside third-party SW has been accomplished using a consistent and scalable approach without any behavioral affections.

C. Co-Simulation Results: Accuracy Enhancements

Vehicle dynamics accuracy enhancements have been presented in Section IV as a way to improve ADAS tuning capabilities. The ultimate goal is that results obtained during simulation are as close to real test track based evaluation as possible. Herein further co-simulation scenarios are presented to highlight how model accuracy enhancements affect the virtual vehicle's behavior. These setups incorporate different accuracy level settings for the models within VCMX, shown in Table II. The chassis velocity v^{CH} has been recorded in all these tests, as this signal directly reflects the combined effects of all enhanced vehicle dynamics models. To address the previously noted indeterminism of SD2, experiments have been executed multiple times and the results have been averaged.

Fig. 12 depicts the car response for each of the tests setups shown in Table II. Herein `Test A` corresponds to the SD2 reference vehicle dynamics. The enhanced model of the brake system has been enabled in `Test B`, achieving a more intense braking effect. This is explained by the repartition coefficient of the enhanced model as a limiting factor (as detailed in Section IV-B). This causes a higher pressure at the brakes resulting in higher torque. Moreover, the pressure build up behavior can also be observed from the different slopes of v^{CH} in `Test A` and `B`. The enhanced model of the brake has been activated in `Test C`, causing a reduction in brake torque. This can be explained with the corrections applied to the friction coefficient (detailed in Section IV-A). Herein the rolling conditions and the increasing temperature reduce the friction coefficient and thus the brake torque. The

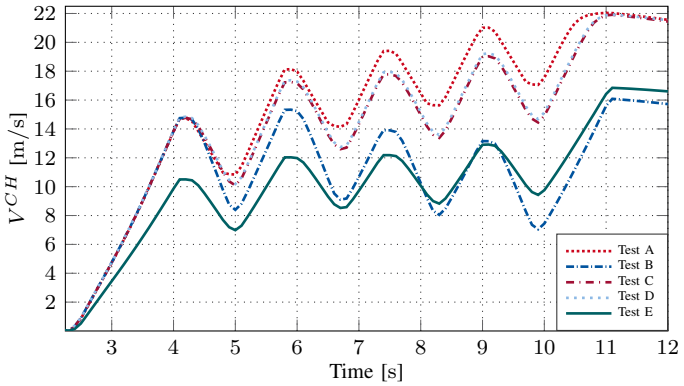


Fig. 12: Co-simulation results: car chassis velocity recordings.

enhanced aerodynamics model has been additionally enabled in `Test D`. Aerodynamic forces have a lower impact on V^{CH} due to relatively low speeds reached in the driving scenario and due to the absence of wind. Finally, the enhanced chassis model has been added in `Test E`, where a more realistic vehicle acceleration can be observed due to improvements on the rotational inertia behavior (Section IV-D). Compared to the reference run, the car accelerated from standstill in 1.5 s to 39 km/h instead of 54 km/h, reflecting a more realistic behavior. Finally, it can be concluded that VCMX adds the possibility to model components in depths required at a given development stage. Hence, the multi-level approach represents an advantage during the prototyping phase of ADAS.

D. Multi-Level Simulation Performance

Multi-level modeling enables to trade off simulation accuracy for performance. To estimate the impact of multi-level modeling on these aspects, both VCMX and SD2 have been instrumented with simulation time measurement facilities. Afterwards, the previous co-simulation scenarios have been repeated for recording performance results. In these tests[1], the execution time of one simulation step (simulated 2 ms), from now on *step time*, has been measured and averaged for both cases. The step times of the complete packages have first been obtained: 2.1096 ms for the SD2 reference, 2.0857 ms for the VCMX low and 2.0977 ms for the high accuracy package. During testing it was of utmost importance to be able to include a human driver in the ADAS evaluation. This requires near real-time simulation execution, meaning simulated 2 ms steps to be executed in around 2 ms real time. VCMX guarantees this property even with enhanced models, although the complete package does not include a full vehicle model as in the case of SD2. Thus, a fair per-module comparison is presented in Table III for models that could be re-created in VCMX, as detailed in Section IV-C and IV-D.

Herein an additional execution time overhead can be noticed between the functionally equivalent SD2 reference and the VCMX low accuracy models. First, it has to be considered that this overhead may arise due to measurement uncertainties at the µs level. Moreover, the step function code is created by

[1]Simulation host for all test runs: x86_64 AMD Phenom II 1055T 6x, 64K L1D and L1I caches, 512K L2 cache, 6144K L3 cache, 8 GB RAM, using Scientific Linux 6.8.

TABLE III: Per-module step time measurements.

Model	SD2 ref.	VCMX Low Acc.	VCMX High Acc.
Aerodynamics	0.4108 µs	-	1.0630 µs
Brake	0.1388 µs	0.9191 µs	5.8151 µs
Brake System	0.1560 µs	0.6483 µs	0.6815 µs
Chassis	2.5076 µs	-	4.3893 µs

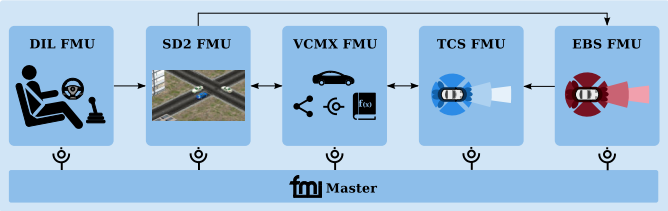


Fig. 13: ADAS prototyping co-simulation scenario.

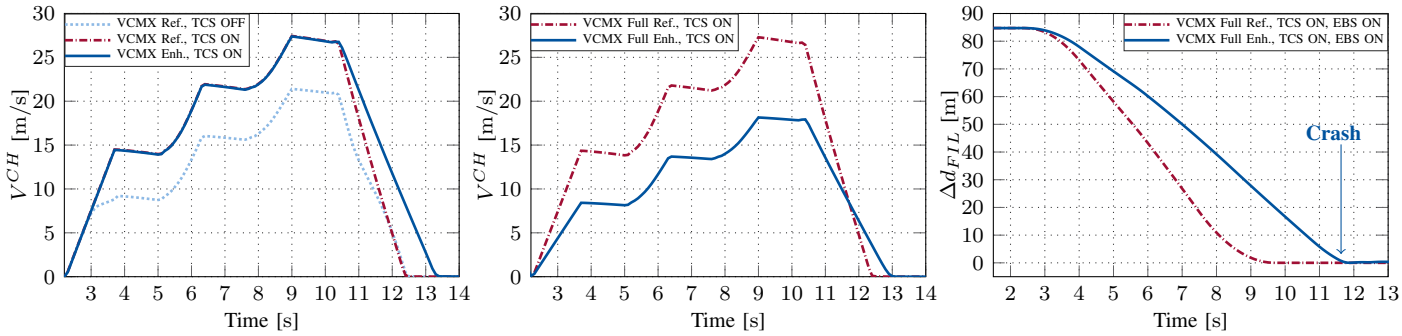
the generic code generation facility within Simulink, which is not expected to be as efficient as hand-written code but much more productive in terms of development time. Furthermore, the VCMX code generator exports additional simulation infrastructure around the behavioral code that might be more demanding in terms of execution performance. Thus, future work needs to be invested in a deeper profiling to determine optimization potential of the exported co-simulation package.

E. VCMX for ADAS Prototyping

To fully utilize the framework, two ADAS prototypes have been designed, refined and tested in a co-simulation aside VCMX. First a *Traction Control System* (TCS) has been created consisting of an *Anti-lock Brake System* and an *Anti Slip Regulation* subsystem. The former regulates per-wheel brake torque to reduce slip when braking, the latter adjusts the engine torque arriving at the driving wheels to reduce slip at acceleration. An *Emergency Brake System* (EBS) has also been created that estimates a safe stopping distance via a front radar (acquiring position and speed difference to obstacles), and engages the brakes automatically, if necessary. These ADAS are especially suited for vVEIL, as a TCS requires testing on various road surfaces and weather conditions, and an EBS is difficult to test on a real track without causing any damage.

Fig. 13 depicts the assembled scenario that contains the TCS and EBS additionally plugged into the co-simulation. Moreover, it was necessary to add additional inputs to the VCMX package, as the TCS requires to act on the brakes of the vehicle. This has been carried out easily via the model parameters configuration GUI, as described in Section III-C.

Fig. 14 depicts the obtained results for three test cases with the driving simulator set to rainy conditions. First, VCMX has been configured to include the brake and brake system models only (Fig. 14a). As the TCS is initially disabled, the vehicle is not able to accelerate up to the desired speed, and then to brake efficiently until standstill due to wheel slip on the wet road surface. Next, the TCS has been enabled and VCMX set first to low then high accuracy. As expected, the TCS assists accelerating and braking the vehicle more efficiently even with reduced braking capabilities of the enhanced model (discussed in Section V-C). Afterwards, the test has been repeated with the full VCMX package employed (Fig. 14b).



(a) VCMX brakes only, TCS OFF/ON (b) VCMX full, TCS OFF/ON (c) VCMX full, TCS ON, EBS OFF/ON

Fig. 14: Co-Simulation results of prototyping the TCS and EBS applications.

Herein acceleration and braking reflect a more reasonable dynamic behavior with enhanced models, for reasons detailed in Section V-C. However, these effects have severe consequences only in the experiment shown in Fig. 14c. Herein the EBS is additionally activated and a second car is placed in front of the test vehicle ($\Delta d_{FIL} = 85\,\mathrm{m}$). As the enhanced models are enabled, the previously discussed effects result in a crash of the test vehicles, due to physical properties the EBS application did not account for. After further refinement of the EBS, the vehicle could stop safely again. These test scenarios perfectly highlight how multi-level modeling helps improving ADAS and assists to improve and accelerate the design-validation-refinement cycle leading towards more mature ADAS designs.

VI. Summary

This paper proposed the vVEIL methodology to combine the advantages of utilizing MBD tools alongside driving simulators to achieve a fully virtual ADAS design and test environment. To put this approach in practice, the VCMX framework has been presented as the main contribution of this work. Herein the "one-click" model export and integration capability of VCMX has been highlighted alongside the possibility to design multi-level vehicle dynamics for early ADAS refinement. Lastly, vVEIL and VCMX have been evaluated by prototyping two ADAS, emphasizing on the impact of virtualization and multi-level modeling on the design.

References

[1] *Design methodology for mechatronic systems*, VDI Std. 2206, June 2004.
[2] O. Gietelink, J. Ploeg, B. De Schutter and M. Verhaegen, "Development of advanced driver assistance systems with vehicle hardware-in-the-loop simulations," *Vehicle Sys. Dynamics*, vol. 44, no. 7, pp. 569–590, 2006.
[3] M. Broy, S. Kirstan, H. Krcmar, and B. Schätz, "What is the benefit of a model-based design of embedded software systems in the car industry?" in *Emerging Technologies for the Evolution and Maintenance of Software Models*, 2012, pp. 6–7, doi:10.4018/978-1-61350-438-3.ch013.
[4] P. Fritzson, P. Aronsson, A. Pop, H. Lundvall, K. Nyström, L. Saldamli, D. Broman, and A. Sandholm, "OpenModelica - A Free Open-Source Environment for System Modeling, Simulation, and Teaching," in *IEEE Computer Aided Control System Design*, Oct 2006, pp. 1588–1595.
[5] JModelica.org Official Website, http://jmodelica.org/.
[6] Scicos Official Website, http://www.scicos.org/.
[7] Scilab Official Website, http://www.scilab.org/.
[8] Scicoslab Official Website, http://www.scicoslab.org/.
[9] Dymola, http://www.3ds.com/products-services/catia/products/dymola.
[10] SCADE Suite, http://www.esterel-technologies.com/products/scade-suite/.
[11] TargetLink, www.dspace.com/en/pub/home/products/sw/pcgs/targetli.cfm.
[12] Simulink Online Documentation, http://mathworks.com/help/simulink/.
[13] Vdrift Official Website, http://vdrift.net/.
[14] OpenDS Official Website, https://www.opends.eu/software/features.
[15] TORCS Official Website, http://torcs.sourceforge.net/index.php.
[16] Speed Dreams 2 Official Website, http://www.speed-dreams.org/.
[17] StuntRally Official Website, http://stuntrally.tuxfamily.org/.
[18] Rigs Of Rods Official Website, http://www.rigsofrods.org/.
[19] TRS by NISYS Official Website, http://www.nisys.de/en/products/trs/.
[20] Pro-SiVIC by CIVITEC, http://www.civitec.com/applications/.
[21] carSIM Official Website, http://www.carsim.com/.
[22] Virtual Test Drive by VIRES Official Website, http://www.vires.com/.
[23] IPG - CarMaker Virtual Test Driving Solution Official Product Website, ipg-automotive.com/products-services/simulation-software/carmaker/.
[24] dSpace - Automotive Simulation Models Official Product Website, dspace.com/en/inc/home/products/sw/automotive_simulation_models.cfm.
[25] R. L. Bücs, P. Lakshman, J. H. Weinstock, F. Walbroel, R. Leupers and G. Ascheid, "Near real-time ADAS prototyping via the virtual vehicle and hardware in the loop methodology," *Submitted to the 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2018.
[26] CosiMate Official Website, http://site.cosimate.com/.
[27] J. Dahmann, R. Fujimoto, and R. Weatherly, "The Department Of Defense High Level Architecture," in *Winter Sim. Conf.*, Dec. 1997, pp. 142–149.
[28] FMI Standard Official Website. https://www.fmi-standard.org/.
[29] M. Hiller, C. Schuster, and D. Adamski, "FASIM_C++ - A versatile developing environment for vehicle dynamics simulation," *International Journal of Crashworthiness*, vol. 2, no. 1, pp. 91–106, 1996.
[30] D. Ward, T. Bertram, and M. Hiller, "Vehicle dynamics simulation for the development of an extended adaptive cruise control," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (Cat. No.99TH8399)*, 1999, pp. 730–735.
[31] K. V. Neumann-Cosel, M. Dupuis, and C. Weiss, "Virtual Test Drive - Provision of a Consistent Tool-set for [D,H,S,V]-in-the-Loop," *Proceedings of the Driving Simulation Conference Monaco*, 2009.
[32] K. Abdelgawad, M. Abdelkarim, B. Hassan, M. Grafe, and I. Grassler, "A modular architecture of a PC-based driving simulator for advanced driver assistance systems development," in *15th International Workshop on Research and Education in Mechatronics (REM)*, Sept 2014, pp. 1–8.
[33] B. Hassan and J. Gausemeier, "A design framework for developing a reconfigurable driving simulator," *International Journal on Advances in Systems and Measurements*, vol. 8, no. 1 & 2, pp. 1–17, Sep. 2015.
[34] G. Jacobs, *"Interaction between Base and Contact Bodies" in Tribology*, 5th ed. Aachen, Germany: Druck & Verlaghaus Mainz, Aachen, 2014, ch. 3, sec. 3.2, pp. 42-55.
[35] *Schaltbare fremdbetätigte Reibkupplungen und Bremsen*, VDI Std. 2241, 1984, p. 5, available in German language.
[36] K. Kabus, *"Wellenkupplungen und -Bremsen" in Decker Maschinenelemente - Formeln*, 7th ed., F. Rieg, Ed. Regensburg, Germany: Carl Hanser München, 2014, ch. 20, sec. Reibungskupplungen, pp. 124-128, available in German language.
[37] R. Marek and K. Nitsche, *"Konvektion" und "Wärmestrahlung" in Praxis der Wärmeübertragung*, 2nd ed. Leipzig, Germany: Carl Hanser München, 2010, ch. 6, 8, sec. 6.1, 8.1 , pp. 185-200, 238-263, available in German language.
[38] L. Eckstein, *Longitudinal Vehicles Dynamics*, 7th ed. Aachen, Germany: fka - Forschungsgesellschaft Kraftfahrwesen mbH Aachen, 2014, ch. 2, sec. 2.2, pp. 103-104.
[39] M. Meywerk, *"Driving Limits" in Vehicle Dynamics*, 1st ed., ser. Automotive, T. Kurfess, Ed. Chichester, England: John Wiley & Sons Ltd, 2015, ch. 6, sec. 6.2. [Online]. Available: http://rwth-aachen.ciando.com/book/index.cfm?bok_ID=1914831
[40] S. Breuer and A. Rohrbach-Kerl, *Fahrzeugdynamik: Mechanik des bewegten Fahrzeugs*, 1st ed., ser. ATZ/MTZ-Fachbuch. Wiesbaden, Germany: Springer Vieweg Wiesbaden, 2015, ch. 8, sec. 8.2, pp. 303-304, available in German language.