

The VehicleDynamics Library — Overview and Applications

J. Andreasson and M. Gäfvert

MODELON AB

{johan.andreasson,magnus.gafvert}@modelon.se

Abstract

The VehicleDynamics Library provides a foundation for model-based vehicle dynamics analysis, in particular related to road-vehicle handling. This commercial new library is a major improvement and expansion of the previously available free vehicle dynamics library. This paper gives an introduction and overview of the new library and gives some example use-cases.

1 Introduction

The potential of Modelica as a mean to efficiently describe vehicle models has been recognised for quite a while [1, 2, 3, 4, 5] and lead to the initiative to develop a library for vehicle dynamics studies [6]. This resulted in the free VehicleDynamics library that was developed and maintained by Johan Andreasson during 2000–2004, on the initiative and with support from Hilding Elmqvist (Dynasim AB) and Martin Otter (DLR Research Center Oberpfaffenhofen).

The great interest in the free VehicleDynamics library and increased requirements and demands on new features, continuity in maintenance and support, and not the least complete documentation triggered the decision to develop the library into a commercial product. This initiative was taken by MODELON AB in 2004, and has resulted in the commercial VehicleDynamics Library. The current version 1.0 of the library is a substantial extension and improvement of the now deprecated free VehicleDynamics library.

The VehicleDynamics Library, or VDL, is intended for vehicle dynamics analysis related to mechanical design and control design of automotive chassis. Handling behaviour is the primary target, but the library is also well suited for studies of other vehicle properties. The sequel will enlighten the contents and use of the library, pinpoint some key issues in the development and outline some expected enhancements.

2 Model and Library Structure

A road vehicle contains a multitude of parts and sub-systems. Conceptually, these can be organized in a hierarchy. For example, a car contains a chassis, that contains front and rear suspensions, that contains left and right suspension linkages and steering, etc. In analogy with this, it is natural to structure a model of a vehicle similarly and that is also reflected in how models in the VehicleDynamics Library and the library itself are organized as illustrated in figure 1 which contents are discussed more in section 3.

There are also vital models that are not reflected in neither of these illustrations that are gathered in special sub-packages throughout the package hierarchy:

Interfaces The Interfaces sub-packages contain common connector and parameter definitions and are used as base classes to ensure compatibility. Advanced users that define their own component models should use these classes, when applicable.



Templates The Templates sub-packages contains template classes that can be regarded as "wizards" to create more complex component models, vehicles, and experiments. By extending a template model and filling the placeholders with actual component models, new models can be implemented in an efficient way that is easy to maintain.



Components The Components sub-packages contain models that are used to build component models in the corresponding package. These models can be used by advanced users that build their own component models.



Internal The Internal packages contains models of no normal interest for the user, and these models should not be directly used by the user.



Experiments Models in VDL that are complete and meaningful to simulate are called *Experiments* and are indicated by a specific icon. Templates for experiments are located throughout the



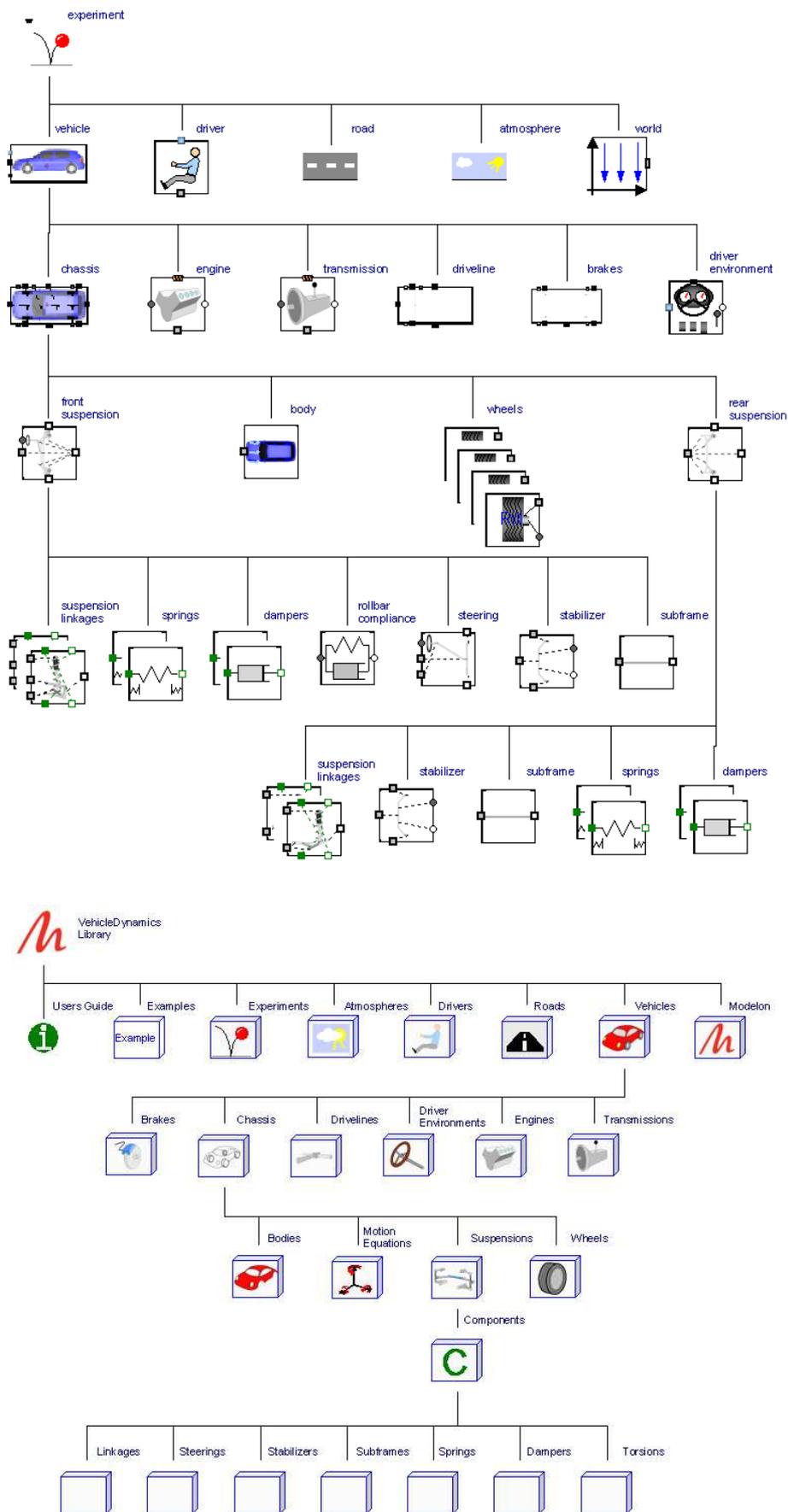


Figure 1: Hierarchy of an experiment model (top) and part of the hierarchical library structure (bottom).

library structure and include setups like full vehicle or chassis on a road, as well as test-rigs for analysing individual components or subsystems as discussed in section 5.

As for the sub-packages, there are also component types that are found throughout the library:

Base Base class for group of components in a sub-package. Located in Interfaces subpackage. Partial (incomplete) classes that must be extended before being instantiated.

Standard Base class for group of components in a subpackage. The Standard base classes differs from the Base base-classes in that they are less general. Standard base classes are designed to provide a standard connector interface that covers the most common variants for the component type. Standard base-classes extend from Base base-classes.

 **None** No (empty) implementation of a component that still simulates. Null components are mainly used in templates as a way to "leave out" components.

Typical Component that represents a typical variant.

Basic Basic model of a component.

Ideal Ideal model of a component.

3 Library Contents

3.1 Atmospheres

The Atmospheres package contains models of atmospheric conditions such as wind speed, humidity and density. This information is useful for analysis involving aerodynamic resistance or downforce, and responses to wind disturbances. It can also be used for other models that requires atmospheric information such as engine cooling.

3.2 Drivers

The Drivers package contains configurable open-loop, closed-loop, and event-driven driver models. The open-loop models has an extended set of sources to also be able to handle standard maneuvers such as instability triggering sine with dwell time or sine increasing amplitude. The closed-loop models use road information (see below) for positioning on and speed control. More complex sets of instructions are handled by the event driven driver model that allows changing between and combining open and closed loop tasks.

3.3 Roads and RoadBuilder

The Roads package contains road models and related components. The flat ground with optional inclination is useful for open-loop studies, while the versatile 3-dimensional table-based road model can be used with closed-loop driver models to analyze maneuver responses and driving on road-segments with user-defined geometry. The road models also contain reference trajectories for drivers without planning abilities. Obstacles such as cones and wind indicators can be used to make animations more illustrative.

The RoadBuilder sub-package contains utilities to create tables for the 3D-road model, either specialized for e.g. a double lane change or more generic, figure 2 illustrates typical input for the latter: Curvature is the inverse of the radius of the center line (1), road slope at the center line is the altitude increment along the road heading direction (2) and banking is the altitude increment perpendicular to the road center line (3). Road is the downwards-outwards increment from the center line which is present on roads to allow efficient water drainage (4). Left (5a) and right (5b) road edges are defined so that positive values defines the offset to the left of the center line. Note that the right edge always should have a greater value than the left edge. The start position (6) and heading angle (7) is the position and angle of the center line at zero distance, respectively. The track position (11) is defined as an offset from the road's center line that a driver model may follow by varying the steering input. If the offset is 0, the driver will follow the road center line and accordingly there is a velocity reference (11). The user can also specify the maximum error that is allowed due to linear approximation of a curve segment (8) and a maximum allowed distance between two consecutive grid points, this only occur for long straights (9). The resolution (10) defines the minimum distance between two grid points that RoadBuilder consider as separate when merging the input tables. Additionally there are cones that can be positioned along the road.

3.4 Vehicles

The Vehicles package contains models of whole vehicles and related components. Vehicle models are partitioned into the subsystems: chassis, driver, environment, engine, transmission, driveline, and brakes. The focus of VehicleDynamics Library is on chassis models, but there are also models of other subsystems such as driver, environments, engines, transmissions, drivelines, and brakes. The Vehicles package contains one sub-package for each subsystem.

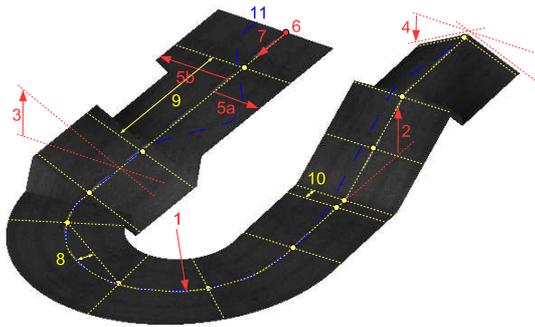


Figure 2: An example of possible RoadBuilder user inputs, note that the inputs in the example are exaggerated to make them clearer.

3.5 Chassis

The Chassis package is further partitioned into sub-packages for bodies, suspensions, wheels, etc. It contains a set of typical chassis configurations that represent standard cars like compacts, sedans, and SUV:s. In particular, there are models of different fidelities with the same interface and it is thus possible to use them with e.g. the same driveline and brake models. This minimizes modeling effort.

3.6 Suspensions

VDL includes a wide range of suspension models of both independent, semi-independent, and rigid axle types. The fidelity level spans from planar models with fixed roll and pitch centers, to table-based and geometric kinematic and elasto-kinematic models. As described in section 5, there are full-fledged kinematics and compliance (K&C) experiments of full suspensions as well as sub-components for analysis and model reduction.

3.7 Bodies

The Bodies package contains body models including inertial and aerodynamic properties as well as animation shape. Bodies can also conveniently be configured for different payloads.

3.8 Wheels and Tyres

The main content of this package is the tyre models. VDL includes a selection of well-known and commonly used tyre models for handling such as variants of Magic Formula, Pacejka models, Rill models, and semi-empirical models based on brush-model mechanics. An advantage is that the partitioning of the models

allow the same models throughout the chassis fidelity range which allow convenient configuration and comparison of results.

3.9 Sensors

Sensors can be freely located on the vehicle to be used as feedback signals for controllers or observers. VDL includes a set of sensors that are easy to configure and extend.

3.10 Drivelines and Brake Systems

The driveline and brake system models in VDL include 3D-position and orientation as described in section 4.1. Models of joints, shafts, differentials and more resolve driveline effects in a detail suitable for handling analysis. VDL also includes ideal and basic hydraulic brake systems and also allow inclusion of user-defined models based on any Mechanics interface from the Modelica Standard Library (MSL).

3.11 Engines and Transmissions

The engine package contains map-based and MVEM engine models and basic models of manual, automatic and CVT transmissions. Again, the adaption of a rotational interface with 3D capabilities, user defined models based on both the Rotational and the Multi-Body interfaces in MSL can be adapted.

3.12 The Modelon Package

The Modelon Library contains classes that complements the MSL. The structure of the library follows MSL whenever possible to facilitate navigation. Components that have a corresponding component in MSL are in general modified/improved versions but with the same basic functionality as the MSL version. The next section highlights some of the contents.

4 Development key issues

4.1 Rotational3D

There has been some development to describe the three-dimensional effects of components in the rotational library such as coriolis forces [7] but there was no satisfactory approach available for slightly more complex components such as shafts with different joint geometries. To overcome this, a package Rotational3D was introduced with a composite interface definition consisting of a 1D rotational flange resolved in a MultiBody frame. This allow the minor rotations of the axle mounts to be separated for the

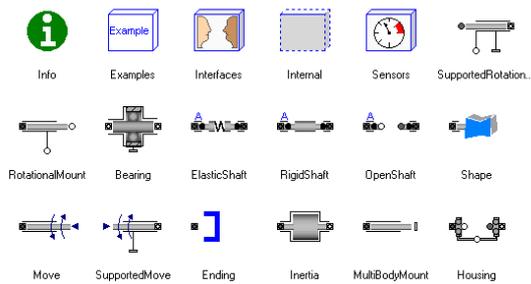


Figure 3: Rotational3D package contents.

axle spin which allows a consistent interface to both `MSL.Rotational` and `MSL.MultiBody` and a snapshot of the library contents is seen in figure 3.

4.2 State Selection in Suspensions

Unlike many other tools, Dymola use symbolic manipulation to resolve kinematic constraints which essentially mean that for a linkage model without included elasticities, the number of states correspond to the number of degrees of freedom that needs to be chosen with care. This is best illustrated with an example; consider the double wishbone linkage in figure 4. The leftmost version is completely rigid and the five links used constrain five of the original six degrees of freedom and there are thus two states that needs to be selected. If Dymola is able to select the states by its own, there are at least four combinations that could come up but only two of these turn out to be reasonable. Having the state in the unsuspended body is not reasonable for at least three reasons: First the numerical accuracy, if the vehicle is moved a considerable distance from its reference position, the resolution of the wheel travel would be low since it means subtracting two large numbers from each other. Secondly, since the motion of the upright would be defined relative to the world frame which would mean that if the vehicle would roll enough, the representation would be numerically tough and eventually singular. The third reason is common with having the state in the strut and allow multiple solutions according to figure 5. For some suspension linkages, multiple solutions could also occur when having the state selection in the joints but experience has shown that this is far easier to deal with simply by just supplying reasonable start values.

As seen from the discussion above, the ability to explicitly chose states is of significant important and it is not enough to be able to select all or no states in a joint as illustrated by the elasto-kinematic linkage in figure 4. The kinematic linkage needs two states to

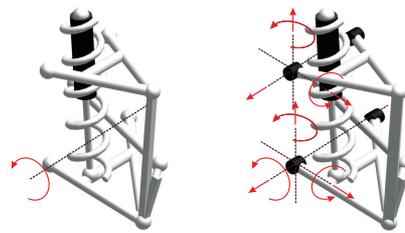
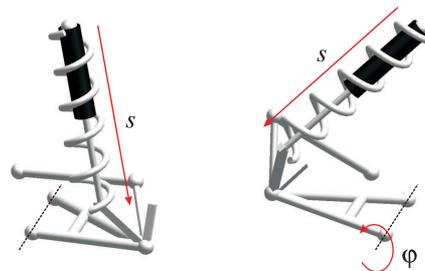


Figure 4: State selection for a kinematic (left) and an elasto-kinematic (right) double wishbone suspension linkage.

Figure 5: If s would be chosen as state, it is difficult to separate between wanted (left) and unwanted (right) configurations. Choosing φ instead makes the configurations unique.

specify the only degree of freedom which can be set in either of the joint pairs. Using bushings instead of inner joints gives ten additional degrees of freedom, i.e. eleven in total. However, since each of the pair of bushings have twelve potential states, 22 out of 24 have to be selected.

The standard `MultiBody` library has been designed with ease of use in mind [8] so the available models have the state selection lumped for all degrees of freedom which in the above case only would allow the user to specify 12 of the 22 states and leaving the rest for Dymola to figure out which works in some cases but not always. To overcome this issues, a set of joint models were developed with advanced users in mind, where each joint state candidate can be set independently of the others. Additionally, a body model without potential states was implemented, which enforces the possible states to be relative motion in joints. Also the quaternion representation was disabled since it require additional code and events for the dynamic state selection.

4.3 Signal Bus

Any complex modeling involving some kind of control sooner or later run into the need to structure informa-

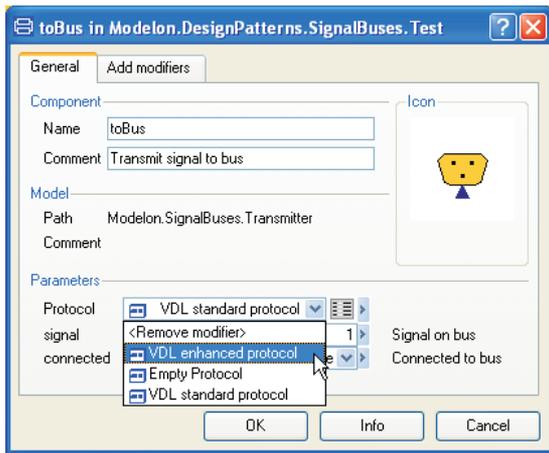


Figure 6: Choice of protocol in a transmitter.

tion exchange. More specifically, the following desires were identified: 1) Being able to predefine a set of possible signals on the bus. 2) Being able to replace this set. 3) Being able to have several sets at the same time. 4) Allowing efficient diagnosis for fault detection of connections. 5) Allowing GUI support from the tool. 6) Being able to model the dynamics of the actual bus (e.g. delay). 7) No need to draw extra connections to have components exchange signals with the bus.

The suggested implementation use the dynamic name lookup using the `inner` and `outer` constructs that allow coupling of models throughout the hierarchy. The actual bus definition is built around a base package definition containing a bus and a protocol and this package is declared as a `replaceable` type in the receiver and transmitter models.

By this construction, a user can define any own protocol by extending the base package definition and complement with the signal names that should be available. When connecting a component to the bus using a transmitter or a receiver component, the user can select what of the predefine protocols should be used and there is also enough information for a tool such as Dymola to actually display a list of the available signals in the selected protocol, see figure 6.

Despite any bus definition, the actual signal flow remain complex and being able to support the user in fault detection and debugging is crucial for efficient usage. Since a general Modelica model is free from causalities, it occurs as systems of equations and thus if a signal on the bus is not defined or defined more than once, the whole model will become singular and it is hard for a tool to give useable feedback on these types of errors.

The suggested construction allow multiple busses but require that there is only one set of signals per bus

which allow efficient diagnosis. By introducing a debug mode with a cardinality variable, the bus model can ensure that one and only one transmitter is connected to each signal and also replaced undefined signals with dummies. In this way warnings can be given if two or more transmitters try to send the same signal to the bus, if a receiver tries to access a variable that is not sent by a transmitter etc.

5 Library Usage

This section highlights some issues relating to the library usage.

5.1 VDWorkbench

VDWorkbench is a user-writable area that is installed together with the VehicleDynamics Library. It is opened from the **File|Libraries** menu. In this area users can work with examples and store their own models and experiments. The structure of VDWorkbench is laid out to mimic the one in VehicleDynamics and users that produce a lot of custom models and components are encouraged to continue this.

5.2 Summary Records

Many standard components contain a `Summary` record, indicated with an icon in the diagram layer. There are different summary records for different classes of components. The summary record contains variables that are of general interest for post-simulation analysis and plotting. For example, the summary record for a standard engine contains the variables for engine speed, torque, and power.

Summary records are convenient to use in the Variable Browser in the Simulation tab. By clicking the **Advanced** button and entering "summary" in the search filter the variable list is restricted to summary records as illustrated in 7.

5.3 Custom Vehicles

The library provides templates for building models of standard cars. Non-standard vehicles are built either directly from the component base classes. It is also possible to define custom templates for non-standard vehicles. The library contains a large number of components that can be used in the templates and it possible for users to define their own components to use in templates. For example, parts like A-arms, bushings, struts, links, gears etc. are available in the Suspensions package for users that defines their own suspensions. Figure 8 shows the the implementation of a



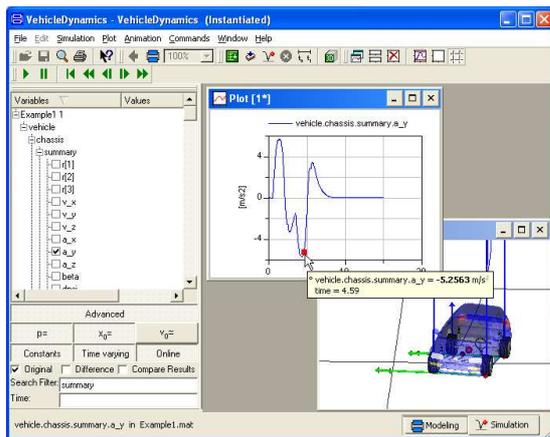


Figure 7: An example of using a summary record.

vehicle with *autonomous corner modules* (ACM), [9], where each wheel is steered, cambered, suspended and driven/braked individually.

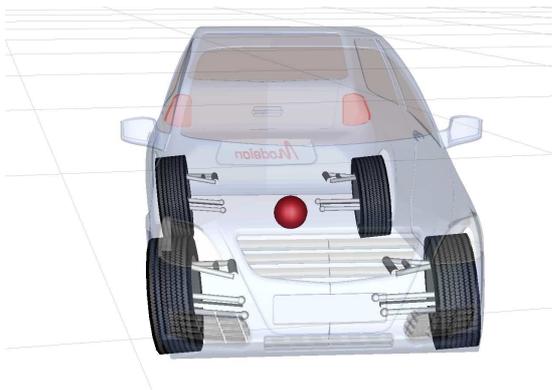


Figure 8: Custom vehicle design exemplified with the ACM concept.

5.4 Migration

The ParsfileConverter utility is used for migrating models from the Parsfile format used by CarSim into VehicleDynamics Library. The utility can handle hierarchical Parsfiles that are split into multiple files as well as flat single-file Parsfiles. The ParsfileConverter is a stand-alone executable that is supplied with VDL.

5.5 Vehicle Control

A major area of application for VDL is vehicle control. This example illustrates how active stabilizers that can be used to enhance stability are incorporated in a VehicleDynamics model. Figure 9 shows the response of an evasive lane change performed by a closed loop drivers for two full loaded vehicles, with and without roll control.

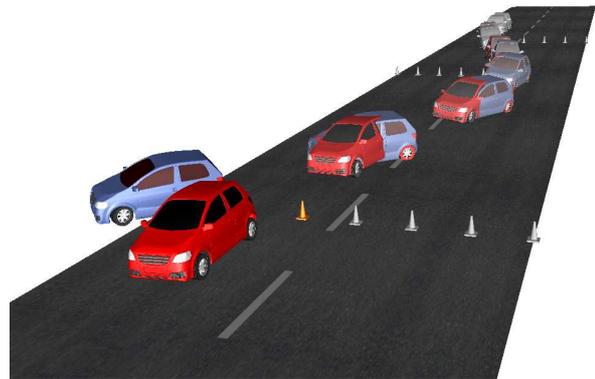


Figure 9: Active stabilizers can be used to prevent roll-induced instability.

5.6 Component Analysis

A vital part in any system design is a thorough understanding of the involved subsystems and an analysis of a full vehicle is often useless if the behavior of involved subsystems is not sufficiently understood. To isolate component behavior and to avoid unnecessary computational cost, test rig experiments are great aids. In VDL they are designed, parameterized and animated as real test rigs where applicable. To a test rig, various controller can be attached to govern the experiment and facilitate the test procedure. Figure 10 illustrates a typical K&C experiment with the corresponding user settings.

5.7 Wheels and Tyres

The tyres are critical components to determine the response of a road vehicle to driver inputs. Tyres are also famously difficult to model and calibrate with experimental data. VDL include several of the most well-known and established tyre models for handling studies. It is important to understand the differences between the different models, and to have full insight into the properties of different data-sets for the model parameterizations in order to correctly interpret experiment results on chassis. To support the users in this, VDL includes versatile test rigs where wheels and tyres can be studied in isolation. The test rig in figure 11 can be used to study quasi-static and dynamic properties under combinations of sweeps and constant levels of slip-angle, slip ratio, camber, load, and velocity.

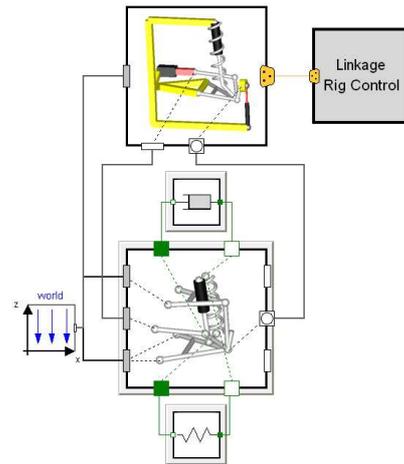
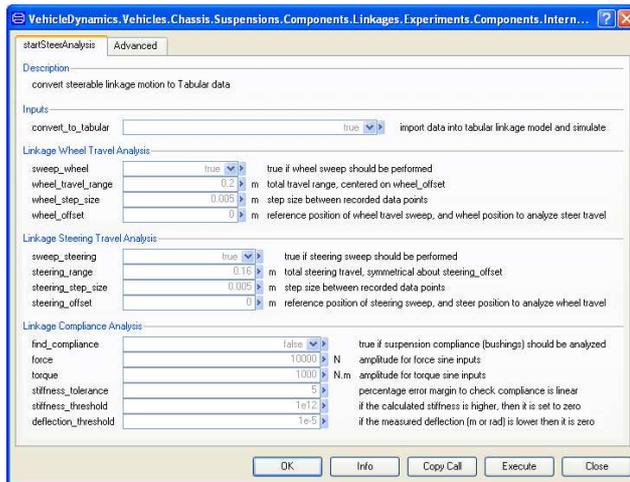


Figure 10: K&C experiment (right) with user settings (left).

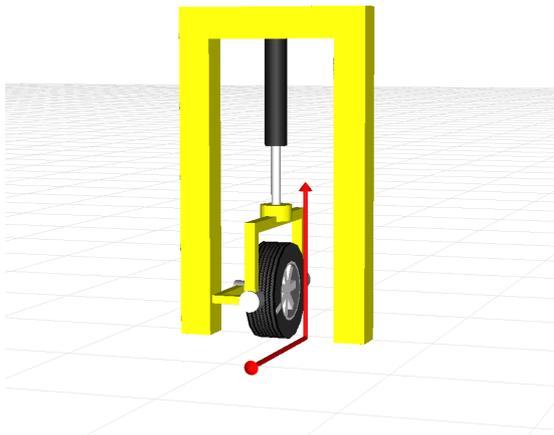


Figure 11: Tyre test rig.

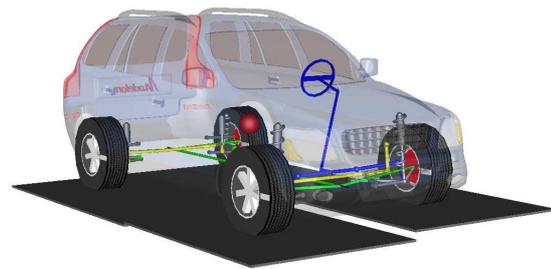


Figure 12: Vehicle mounted in a test rig.

5.8 Vehicle Analysis

Vehicle analysis can be performed both on road using a driver or robot to control the vehicle which allow simulations to mimic real test procedures to generate e.g. cornering characteristics diagram. Additionally, there are also ideal models of e.g. drivelines that allow precise wheel spin velocities or torques to be applied. Additionally, just as for a component, the vehicle can be constrained in different test rigs to generate various kinds of measures that requires extensive laboratory equipment to be performed in reality.

In figure 12 such an experiment is exemplified using a shaker rig. The model allow the level under each wheel to be adjusted individually and depending on the input used, this can be used to analyze chassis out-of-plane characteristics such as roll and pitch dynam-

ics, spring and damper settings and roll moment distributions. Additionally, it can for example be used to analyze driveline motion and study joint angles, shaft lengths, and other variables during various suspension-travel scenarios to verify or tune the geometric design.

6 Ongoing and Future Extensions

This section enlightens some of the extensions of the VDL that are ongoing and/or under consideration. Two other Modelon libraries, PneuLib and HyLib, opens up for incorporation of more detailed hydraulics and pneumatic models. Hydraulics models are relevant for especially brake systems but also other hydraulic systems found in e.g. power steering. Pneumatics is highly relevant for the ongoing enhancement to also consider heavy vehicles, figure 13, both for brake systems and air spring suspensions. For this extension, flexible elements are of increased impor-



Figure 13: Tractor with semitrailer negotiating a turn.

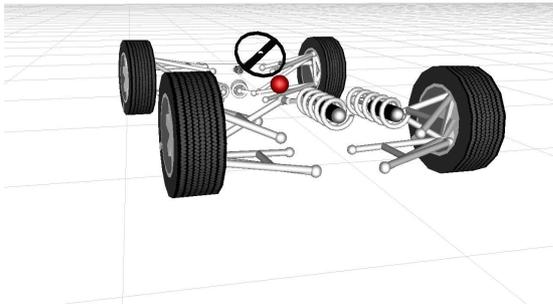


Figure 14: Cornering characteristics test of a Formula SAE chassis.

tance and this is discussed in more detail in [10].

A racing extension is also considered, requiring mainly different types of suspensions and analyzes, figure 14 shows a test run of a Formula SAE chassis.

Acknowledgements

The development of the VehicleDynamics library has included the generous support and encouragement of many people and organizations. Especially, the authors would like to thank:

- DLR, Oberpfaffenhofen, main author of the Modelica MultiBody library, in particular Prof. Martin Otter, for support and cooperation in realizing a Modelica vehicle dynamics library
- Royal Institute of Technology, Division of Vehicle Dynamics, Stockholm, Sweden, for generous support in the realization of the library

- Dynasim AB, for extensive support and enhancements of Dymola to form an optimal platform for vehicle dynamics modeling and simulation. Particular thanks to President Dr. Hilding Elmquist.

References

- [1] J. Andreasson, A. Möller, and M. Otter. Modeling of a racing car with Modelicas MultiBody library. In *Proc. of the 1st Int. Modelica Workshop*, Lund, October 2000. The Modelica Association and Lund University.
- [2] S. Drogies and M. Bauer. Modeling Road Vehicle Dynamics with Modelica. In *Proc. of the 1st Int. Modelica Workshop*, Lund, October 2000. The Modelica Association and Lund University.
- [3] B. Jacobson et al. Modelica usage in automotive problems at Chalmers. In *Proc. of the 1st Int. Modelica Workshop*, Lund, October 2000. The Modelica Association and Lund University.
- [4] M. Tiller et al. Detailed vehicle powertrain modeling in Modelica. In *Proc. of the 1st Int. International Modelica Workshop*, Lund, October 2000. The Modelica Association and Lund University.
- [5] M. Dempsey et al. Coordinated automotive libraries for vehicle system modelling. In *Proc. of the 5th Int. Modelica Conf.*, Wien, September 2006. The Modelica Association and arsenal research.
- [6] J. Andreasson. VehicleDynamics library. In *Proc. of the 3rd Int. Modelica Conf.*, Linköping, November 2003. The Modelica Association and Linköping University.
- [7] C. Schweiger and M. Otter. Modelling 3D Mechanical Effects of 1D Powertrains. In *Proc. of the 3rd Int. Modelica Conf.*, Linköping, November 2003. The Modelica Association and Linköping University.
- [8] M. Otter, H. Elmquist, and S.E. Mattson. The new Modelica MultiBody library. In *Proc. of the 3rd Int. Modelica Conf.*, Linköping, November 2003. The Modelica Association and Linköping University.
- [9] S. Zetterström. Electromechanical steering, suspension, drive and brake modules. In *Proc. of 56th Vehicular Technology Conference*, volume 3, pages 1856 – 1863. IEEE, 09 2002.
- [10] N. Philipson. Leaf spring modeling. In *Proc. of the 5th Int. Modelica Conf.*, Wien, September 2006. The Modelica Association and arsenal research.