The Vehicle Dynamics Library: New Concepts and New Fields of Application

Johan Andreasson
Modelon AB
Ideon Science Park
SE-22370 LUND
johan.andreasson@modelon.com

Abstract

The Vehicle Dynamics Library is a commercial Modelica library for vehicle dynamics applications. This paper highlights recent development with focus on extended usability. Key changes are improved interoperability with other tools, improved simulation performance, extended vehicle system simulation, and expanded analysis. Examples are given from efficient simulation of drivelines, development of active safety systems, and quasi-steady-state analysis, among others.

Keywords: Vehicle dynamics; mechanics, active systems, quasi-steady-state analysis

1 Introduction

The Vehicle Dynamics Library (VDL) [2] is a commercial Modelica library providing a foundation for model-based vehicle dynamics analysis. Since the introduction of the library in 2006, there have been significant extensions and improvements. In this paper, some of these and their fields of application are discussed.

The scope of VDL spans from classic vehicle mechanics analysis to full vehicle system simulation and evaluation. One of the fundamental guiding principles of the library is the ability to mix between behavioral and physical models to make it possible to conveniently change, not just between different configurations, but also between different levels of detail.

As such, VDL is designed with several aspects in mind and in this paper, the contents is focused on four main aspects; Section 2 focuses on the work on the mechanical models of the vehicle while Section 3 treats the system aspects of the vehicle, in this case meaning the part of the vehicle that is used to control the vehicle mechanics. Section 4 describes further options to interface VDL with other tools and Section 5 describes extensions to the scope of analysis.

As this paper presents the incremental work relative to what is presented in [2], it is strongly recommended referring to that work for a more thorough background to VDL and its architecture.

2 Vehicle Mechanics

Here, the focus is on the work driven by improvements in configurability and simulation speed. The two first sub-sections explain the improvements of the suspension models while the latter focus on mechanics related to driveline and brakes.

2.1 Extended Suspension Templates

Typically, a suspension, just like any part of the vehicle is configured from different templates. This allows to conveniently exchanging joints and parts while maintaining the topology. Main reasons for this are to improve working efficiency and reduce model maintenance [3].

To facilitate the configurability also of the suspension topology, new components have been introduced that allow also for topology changes to the connection structure to be made based on parameter settings. As illustrated in Figure 1, the stabilizer mount (a) can be connected to the upper A-arm (b), the upright (c) or the lower A-arm (d). In this example also the spring mounts connections can be reconfigured which in this case covers 18 different topology configurations. Further customization is straightforward for the user if necessary.

Unlike some other multi-body simulation software, the Modelica formulation is transformed so it can be integrated by standard index 1 solvers. This method has several advantages and is one of the keys to the inherent multi-engineering capabilities. One implication is that a closed mechanical loop generates implicit constraint equations that have to be solved, either symbolically or numerically.

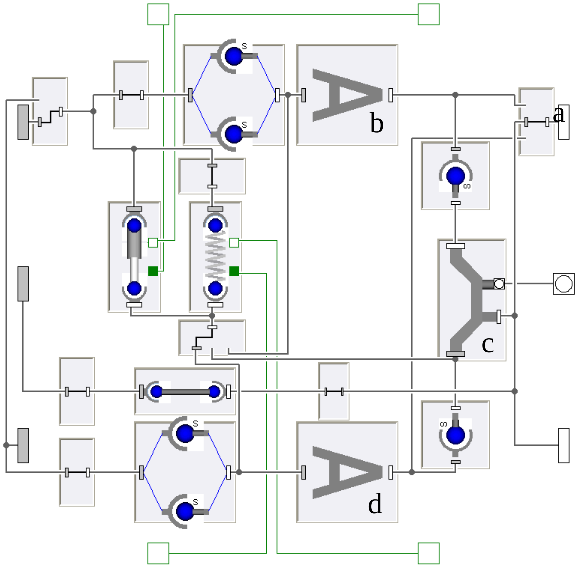

Figure 1 a reconfigurable suspension linkage template where each element can be replaced.

To support symbolic solutions, which normally is the much faster alternative, a dedicated formulation was introduced in Modelica.Mechanics.MultiBody [9,1] and is based on the idea that for a sequence of joints that in total have six degrees of freedom, no constraints are required if a composite model representing all the joints is created.

This type of formulation can significantly improve simulation speed as the lack of constraint equations also eliminates the corresponding nonlinear systems of equations in the resulting simulation code. This is important especially for real time simulation. In VDL, the concept has been modified slightly to allow for users to replace individual linkage components within the composite model.

The concept is illustrated in Figure 2, showing the same suspension topology as seen in Figure 1, with the difference that the upper control arm (a), the king-pin (b) and the tie rod (c) is represented as one composite joint without any constraints. This mechanism defines the wheel carrier motion (d) given the motion of the chassis (e), the lower control arm (g) and the steering rack (f).

From a user perspective, there is no difference between the models in Figure 1and Figure 2, except for the improved simulation speed; all other VDL features such as force visualization remain the same.

2.2 Behavioral Suspensions Models

Behavioral suspension models are a common way to represent suspension characteristics in a convenient way. The idea is to record how the wheel carrier moves depending on the degrees of freedom in the suspension, and to capture that in functional representations, e.g. by tables or polynomials.

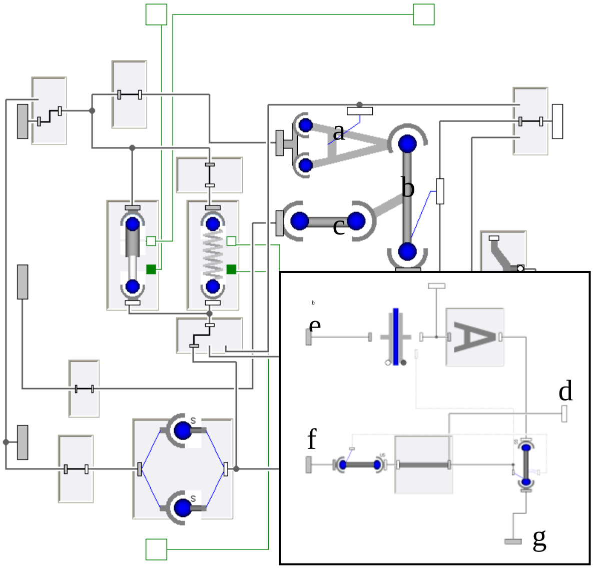

Figure 2 the same suspension topology as in Figure 1, but with an efficient model formulation to improve simulation performance

In VDL, the focus has been on supplying a complete functional representation so that the result should be identical with any kinematically well defined suspension. This includes not just the actual characteristics, but also the transmitted reaction forces and torques. This is essential especially for steering design.

The suspension kinematics is represented by a set of tables, where the number of dimension depends on the degrees of freedom of the linkage. For an independent front suspension linkage, the representation uses two dimensions, one for suspension travel, and one for steering, giving eight functions, three for hub position and orientation, and one for spring and damper compression, respectively.

The tabular representation has the same interface as a multi-body variant. This modular approach allows selecting parts of the chassis or suspension to be implemented using tabular characteristics, and other parts to be represented with traditional multibody implementations.

To address the compliance which typically is present in a suspension, due to elastic bushing elements or material compliance, the effects of the compliance is in the behavioral case super-positioned on the kinematic motion. This can be done, either separately for each linkage, or lumped for the whole suspension.

The suspension compliance, just as the kinematics, can be calculated from a more complex model, but a common scenario is to get compliance and/or kinematic information from measurements on real vehicles. Typically, the compliance information is given as compliance matrix $\mathbf{C}$ (inverted stiffness) rather than stiffness as typically is used in simulation software. For a model described as a spring-mass system

$$F = \mathbf{K}\Delta, F = \mathbf{M}\ddot{\Delta},$$

the compliance data renders two problems: First, $\mathbf{K} = \mathbf{C}^{-1}$ has to be generated but $\mathbf{C}$ is not necessarily full rank and probably ill-conditioned. The reasons may be many, for cost and time reasons for example, often only parts of $\mathbf{C}$ are measured. As a result, manual work and assumptions are often required to compute $\mathbf{K}$. Second, the relation between $\mathbf{M}$ and the resulting $\mathbf{K}$ are often such that the Eigen-frequency is much higher than the frequency range you are considering when using this type of models. Simulating this would lead to a significant performance loss.

The latter problem described above is in this context actually an opportunity to reformulate the model. Since the inherent frequency of the system is unwanted, the compliance data can be used directly in a model that is formulated as

$$\Delta = G(s)\mathbf{C}F,$$

with $G(s)$ being a second order system with a defined cut-off frequency. This frequency can be set by the user so that the model's static behavior is identical to the spring-mass system, and with a dynamic response that is fast enough for the performed analysis.

A further advantage with this approach is that the arbitrary values in $\mathbf{C}$ can be set to zero, corresponding to the removal of degrees-of-freedom, without the need to change the model topology. It is therefore easy to switch between a compliant and a rigid version of a suspension, by just modifying the parameters. With the spring-mass approach, such a change requires a recompilation of the model before it can be simulated. The approach has been used in e.g. [8] and is also suitable to use with kinematic models for real time simulation purposes.

2.3 Driveline and brake mechanics

In [2,3] Rotational3D was introduced, a concept that allows for the three-dimensional effects of 1D rotational mechanics to be captured with a straightforward representation. Compared to the MultiBody approach, simulation performance improvement is in a typical case around a factor 20, the details are explained in [4].

Since then, focus has been on reducing computational cost in drivelines and brakes further by improving the performance of the hybrid elements, especially friction. With the reached performance increase, one can conveniently model and simulate different active drivelines without any considerable simulation slowdown. The new friction components are provided with the Modelon.Mechanics library and these are the recommended choice for vehicle subsystems such as brakes and powertrains, especially in real-time critical applications.

The driveline topology has been reworked to facilitate the configuration of e.g. user defined differentials. The idea is to use Rotational3D to build housings for e.g. the differential, and then to use components based on the standard Rotational formulation. This is illustrated for two differentials in Figure 4; the open differential to the left consists of three Rotational3D components corresponding to the bearing of each axle (a), and two components for visualization (b). The other components; the cut component (c), the pinion-ring gear (d) and the differential (e) are purely one-dimensional components.

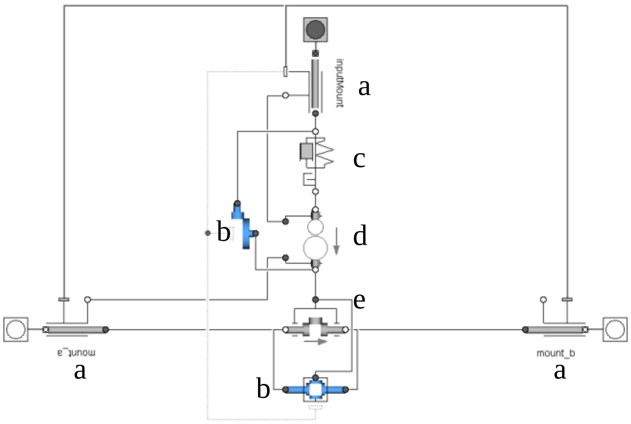

Figure 3. The layout of an open differential using standard Rotational components in a Rotational3D housing. Components with blue icons represent graphical information only.

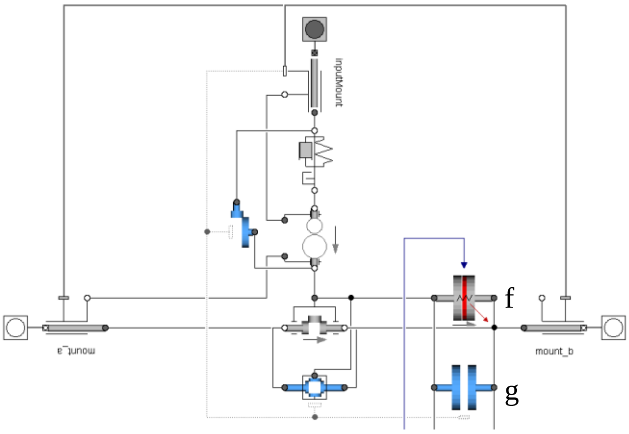

Figure 4. The differential from Figure 3, extended with a clutch between the differential case and the right axle, used for slip control.

One reason to separate the differential from the ring-pinion gear is seen in Figure 4; the clutch controlled differential is an extension with an additional 1D clutch (f) and a component for its visual properties (g). The clutch is connected to the differential case and the right axle, allowing the differential to lock.

To facilitate the analysis of the performance of drivelines, Rotational3D has been extended with built-in visualization of torque flow. The direction of the torque flow visualization does not visualize the sign of the torque but the direction of the power flow. This gives a quite intuitive interpretation of the visualization of the results. An example is given in Figure 5.

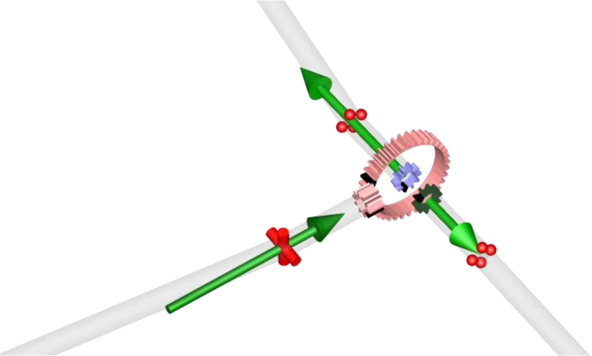

Figure 5 Visualization of a driveline where more torque is distributed to the right (top) shaft than to the left (bottom)

3 Vehicle Systems

The pure mechanical part of vehicle dynamics does and will continue to decrease as different active systems play an increasing roll for vehicle behavior. VDL was designed with this in mind from the start and recent improvements have been focused on extending this functionality. One significant part of this work is to include behavioral models that make it easy to quickly get to a minimal representation of the complete system, and then from there be able to select what details to focus on. There is a clear analogy to the tabular suspension models in Section 2.1 that are used to reduce complexity of the chassis. Here, it is illustrated for two active safety systems on one hand, and electrical or partly electrical propulsion on the other.

3.1 Electronic Stability Control

Functional representations on common safety systems have been implemented, including the required actuators and sensors, as well as sample architectures. A first example is an embedded brake system controller with anti-lock braking, yaw stabilization, brake force distribution and traction control, as seen in Figure 6. The signal bus is defined as an inner/outer expandable connector, which is similar to a globally available namespace. This allows for controllers, sensors and actuators to be anywhere in the model hierarchy.

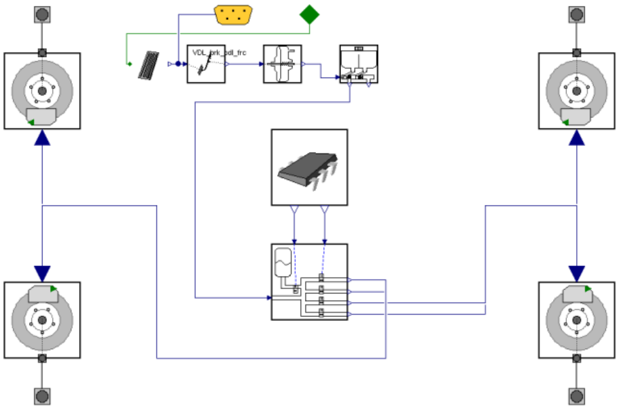

Figure 6 Brake system with modulator, and embedded brake system controller.

3.2 Brake Assist

The models can easily be copied and modified by the user to fit their specific needs. An example of a brake assist controller illustrates this. It is based on the previous example, and extended with a centralized vehicle controller that incorporates information from a distance sensor to add brake action when approaching the object ahead either too fast and/or too close.

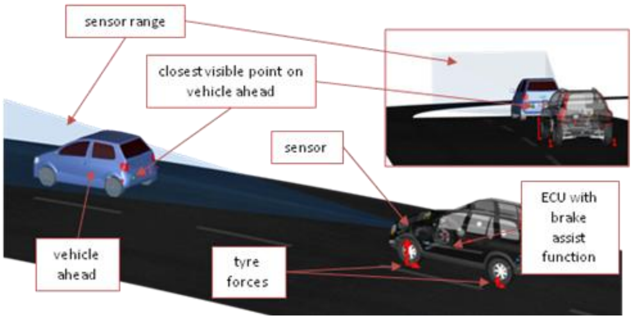
sensor range
closest visible point on vehicle ahead
sensor
ECU with brake assist function
vehicle ahead
tyre forces

Figure 7 Animation view of a test scenario for a vehicle equipped with brake assist.

The modular approach makes it easy to add, extend or modify the existing architecture or to build a new one from scratch, whatever fits the application better. In any case, a user can select what part of the safety system should be modified, a typical scenario is to use the ABS and ESC model when doing initial studies and switch to more detailed models for verification purposes.

3.3 Electrical actuation

Modelica is of course a natural platform for multidisciplinary investigations with VDL and libraries like Smart Electric Drives (SED) [11] or SPOT [12], detailed models of vehicles with electrical actuators can conveniently be modeled [7]. In many cases, however, the need to start on a more conceptual level is fundamental.

To facilitate conceptual analysis, behavioral models for electric systems are introduced. The idea is to parameterize these based on a minimal set of output-related properties such as power, torque and efficiency for machines and then use this information to calculate the electric power consumption.

The models are defined with the same interface, mechanical and electrical connectors, to allow them to be replaced with more detailed versions whenever needed.

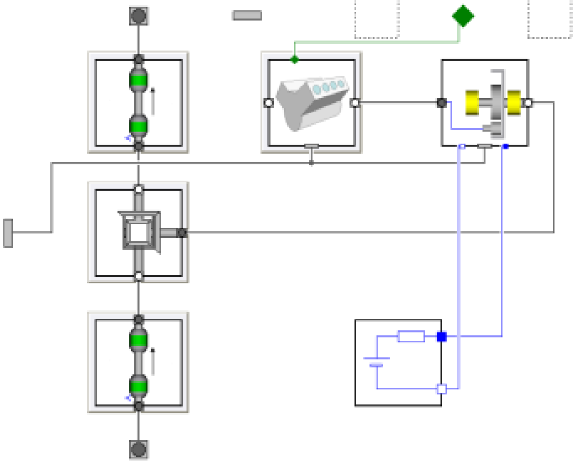

Figure 8. Hybrid electric driveline where components can be exchanged. This allows for convenient switching between detailed and behavioral models.

4 Interfacing

With the use of Dymolas source code and binary export capabilities, the use of VDL spreads to various applications such as in customized track-side tools and as vehicle models in driving simulators and other vehicle dynamics simulation software [13]. This section elaborates on some key improvements in VDL to facilitate such applications.

4.1 External Ground Representations

VDL has a ground representation that is based on a herring-bone representation, described as depending on two independent coordinates. This allows for an efficient representation with high resolution where needed. The interface allows for various implementations, both in Modelica and as external code. By supplying information about the position, heading direction and normal based on these coordinates, VDL is able to calculate contact points.

Recently, this has been extended to allow for user to supply own routines for contact point calculation, this in turn facilitates the use of external ground representations further and allow for more convenient plug-in of VDL models in other types of environments, such as driving or traffic simulators.

For full functionality, three different contact point calculation methods must be supplied as described below and illustrated in Figure 9 and Figure 10.

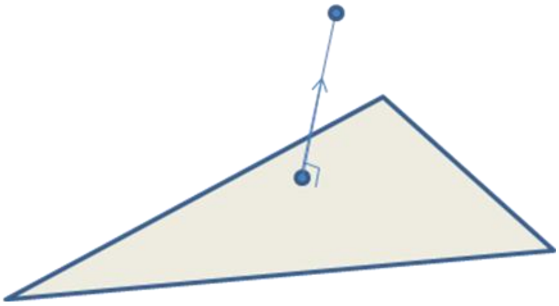

Figure 9 Closest point on surface.

The closest point on the surface, Figure 9, is the default contact point, defined so that the third ground coordinate corresponds to the height over ground along the road normal. This representation is used for e.g. ground impact models.

For a wheel, the first representation is however not a suitable representation if the wheel is inclined relative to the surface. This as the resulting point would diverge from the actual tyre contact. In this case the closest point on the surface that lies within the plane of the wheel should be returned, Figure 10, left.

The third method returns the intersection between the ground and a line defined from the point along a predefined vector a relevant representation, Figure 10, right.

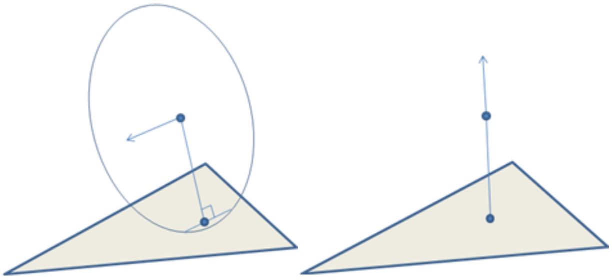

Figure 10 Closest point on surface that lays within a plane, left, and intersection point between line and surface, right.

4.2 Ground Generation

For the tabular ground representation, the RoadBuilder was introduced in [1]. It conveniently generates road data from input like curvature and banking. The functionality of the RoadBuilder has been extended to also handle trajectories defined by measured points, and closed loop circuits.

4.3 Ground Access

Information about the ground can always be accessed from within Modelica by direct function calls. To facilitate usage and to make sure that the most efficient calculations are used, a block set has been developed that allow for users to conveniently build own models that requires ground information.

In Figure 11 this approach is being used to allow for simple tire models to be used on non-smooth surfaces. It works as follows: The original contact point (a) is used as a reference to create a compression profile of the tyre (b), and by a weighting of this information, the resulting contact point (c) can be calculated. With this representation underlying surface model can be made to work with standard single contact point models.

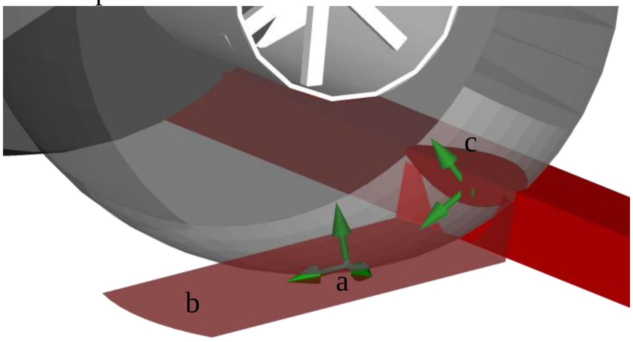

Figure 11 application of contact point filtering for a wheel traversing a cleat.

4.4 External Tire Models

The reconfigurability of VDL makes it easy to include external tire models. This can be done in several ways. The two most common ways are to either replace the VDL tire force calculation only while maintaining contact point calculation in VDL, or to replace the wheel including the contact point calculation. The first method is commonly used to incorporate in-house tire models while the latter is used to interface e.g. DelftTyre [6] and FTire [5].

5 Analysis

Since earlier, VDL provides a various experiments for dynamic simulation for full vehicles and subsystems. This section especially focuses on how also steady-state, and quasi steady-state analyses can be performed. Also, an expansion of the capabilities of the sensor suite is motivated.

5.1 Sensors

For the analysis of vehicle dynamics, sensors are required that can generate forces with respect to different coordinate systems. A typical example is the lateral acceleration of the car that with respect to the car body corresponds to what you would measure on a test drive, and with respect to the ground or the trajectory that corresponds to the cornering capabilities of the car.

To be able to conveniently change between different relevant coordinate systems, the standard set of the sensors has been extended with further options to resolve output. Especially, the vehicle frame projected onto the ground surface is added to support cases as described above. This is defined either according to the ground plane under the frame, or as a plane defined by averaging the wheel contact points. These planes coincide with the world horizon for flat roads.

5.2 Steady state and quasi steady state analysis

In many analyses, the dynamic response of the vehicle is not of interest; instead the focus is on the steady-state characteristics. Particularly in racing this type of analysis is a central part in balancing and tuning of vehicles. In Modelica, the complete experiment is defined as

$F(p,u,x,\dot{x}) = 0,$

with p, u, x, and $\dot{x}$ are parameters, input, states and state derivatives, respectively. For dynamic simulation, this is transformed into a formulation that a standard integrator can handle, i.e.

$\dot{x} = f(p,u,x).$

Finding the steady state solution in many cases means solving this equation for $\dot{x} = k$, where k often but not always are zeros. The unknowns are typically combinations of (u,x). When approaching the problem from a solver point-of-view, the result is a nonlinear system of equations that in many cases is difficult to handle. There is ongoing work with Homotopy methods to improve the capabilities of the tools in this respect [10].

Practically, there are some further aspects to consider, and the most important one is that you want an answer even if there is no mathematical solution. A typical application is to take the car to its limit in some sense. The purpose of the analysis is to figure out where that limit is so any approach will at some point shoot over the target.

VDL has been extended to handle this in a robust way. The idea is to modify the model so that the steady state formulation is presented to the solver as a dynamic simulation. The states that describes the principal motion of the car, x_p, is separated from the other states and u_p is introduced for the corresponding inputs. A set of residual flow variables, R_p, is introduced to match the principal states and then $u_p$is used to minimize R_p. To the tool, this is presented as

$$G\big(p, (u', x_p), (x', u_p), (\dot{x}', \dot{u}_p)\big) = 0,$$

where principal states now are inputs and principal inputs now are states. The structure of this formulation makes it straight-forward for the tool to transform it into

$$(\dot{x}', \dot{u}_p) = g\big(p, (u', x_p), (x', u_p)\big),$$

which can be solved with standard integrators. From this simulation one can gain knowledge of all the resulting vehicle states as well as the required driver input. At any point in time, R_p, will give a measure on the validity of the solution. Figure 12 shows the result from a quasi steady-state analysis. The car is set-up at a defined point along the race track, in this case a corner. The speed is increased while cornering curvature is maintained until the lateral acceleration capabilities are exceeded. The screen shots shows the car at different stages of this test.

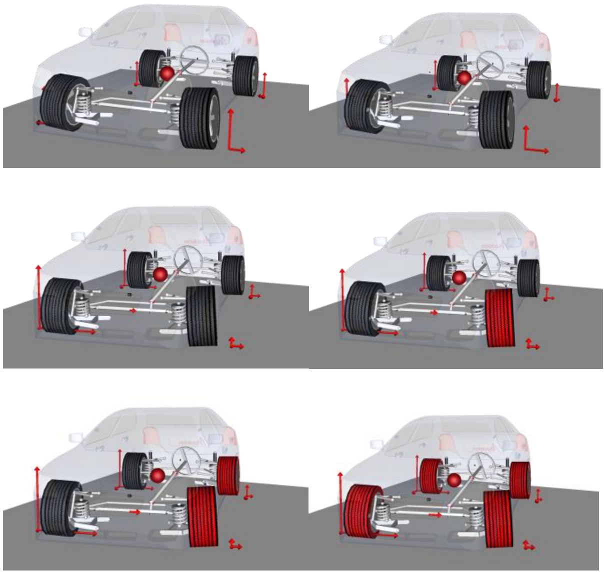

Figure 12 Steady state corner exit solutions for 10, 50, 90, 95, 98, and 100 percent of the car's capacity. Red wheel color indicates saturation. Arrows indicate tire forces. Note the change in steering wheel angle.

Summary

This paper highlights new features in VDL with emphasis on improved interoperability with other tools, improved simulation performance, and an expanded range of analysis.

References

[1] Andreasson J et al, Modeling of a racing car with Modelicas MultiBody library. In: Proceedings of the Modelica 2000 workshop, Lund, Sweden, October 2000

[2] Andreasson J et al, The VehicleDynamics Library - Overview and Applications. In: Proceedings of the 5th Modelica Conference, Vienna, Austria, 4-5 September 2006

[3] Andreasson J, On Generic Road Vehicle Modelling and Control, PhD thesis, ISBN 91-7178-527-2, 2006

[4] Andreasson J et al, Rotational3D - Efficient modelling of 3D effects in rotational mechanics. In: Proceedings of the 6th Modelica Conference, Bielefelt, Germany, 3-4 March 2008

[5] Beuter V, An Interface to the FTire Tire Model. In Proceedings of the 8th Modelica Conference, Dresden, Germany, 21-22 March 2011.

[6] Drenth E et al, Modelica Delft Tyre Interface. In Proceedings of the 8th Modelica Conference, Dresden, Germany, 21-22 March 2011.

[7] Gerl J et al. Multi-Domain Vehicle Dynamics Simulation. In Proceedings of the 8th Modelica Conference, Dresden, Germany, 21-22 March 2011.

[8] Jonasson M et al, Modelling and parameterisation of a vehicle for validity under limit handling, In: Proceedings of the 9th International Symposium on Advanced Vehicle Control (AVEC´08), Vol. 1, pp. 202–207, Kobe, Japan, 2008.

[9] Otter M et al, The New Modelica MultiBody Library, In: Proceedings of the 3rd International Modelica Conference, Linköping, Sweden, November 2003

[10] Sielemann M et al. Robust Initialization of Differential-Alegraic Equations Using Homotopy. In Proceedings of the 8th Modelica Conference, Dresden, Germany, 21-22 March 2011.

[11] Smart Electric Drives, https://www.modelica.org/libraries/SmartElectricDrives

[12] SPOT, https://www.modelica.org/libraries/spot

[13] Ziegler S et al, Extending the IPG CarMaker by FMI Compliant Units. In Proceedings of the 8th Modelica Conference, Dresden, Germany, 21-22 March 2011.