J. Andreasson. A. Möller, M. Otter:
Modeling of a Racing Car with Modelica's Multi-Body Library.
Modelica Workshop 2000 Proceedings, pp.

Paper presented at the Modelica Workshop 2000, Oct. 23.-24., 2000, Lund, Sweden.

All papers of this workshop can be downloaded from
http://www.Modelica.org/modelica2000/proceedings.html

Modeling of a Racing Car with Modelica's Multi-Body Library

Johan Andreasson, DLR, Germany (T96_ANO@t.kth.se)
Andreas Möller, Dynasim AB, Sweden (mra@ffa.se)
Martin Otter, DLR, Germany (Martin.Otter@dlr.de)

Abstract

The modular modeling of the 3-dimensional mechanics of a racing car with Modelica is discussed. For example, one base model class of the wheel suspension is defined and used to model 4 occurences in the front and rear suspension. This model of the car is based on a new Modelica library to model 3-dimensional mechanical systems. This library is explained in some detail.

1 Introduction

In this article the modular modeling of the 3-dimensional mechanics of a racing car is discussed, see figure 1 for a typical representative. The model is described with a new Modelica library for modeling of multi-body systems. This approach has the advantage that it is straightforward to built up the system in a hierarchical way from basic building blocks, such as a suspension or a wheel, which are in turn defined with elements of the multi-body library. The overall model itself is a component which has 1D-connectors to drive the wheels and to steer the car. Using the standard 1D-mechanics libraries of Modelica it is then easy to add the powertrain of the car. This racing car model is mainly utilized to get experience with the handling of complex vehicle models in Modelica and Dymola [2] (component structuring, combining different libraries, symbolic transformation, efficiency of simulation).

2 Formula car characteristics

A formula car is a one seated race car with a minimal carossery. The car model used in this article is taken from [3] and represents a typical formula car very well [7]. Maximum performance on smooth tarmac race tracks is the dominating design criterion and results in the following properties of the different parts of the vehicle:

Mechanics: The driver sits in a low position and the fuel tank is placed around the centre of mass to not effect the weight distribution during a race. The engine and the integrated gearbox are placed between the rear axle and the driver, and work as a structural part of the chassis.

Solid mechanics: The chassis is built up like a carbon fibre shell, a so called monocoque and all connections are stiff, especially the suspension is built up with stiff rods connected by ball joints.

Vehicle dynamics: The geometry and angles are chosen in a way that conserves the wheel geometry during load, like cornering or braking, and compensates for deformation that occurs in wheels rims, hubs and suspension links - all for the sake to maximise the use of the tires full potential in every situation.

Aerodynamics: The vehicle body is slimmed to just house the driver and the engine. Extensions from the vehicle body, such as air intake and housings for the radiators, have as small projection area as possible. As a result, body mass is kept quite close to the cars longitudinal axis, i.e., around this axis the inertia is low.

3 Libraries used for the modelling

The Modelica libraries used as a basis for the modeling of the racing car are shortly discussed. These libraries can be downloaded from the Modelica Webside (`http://www.Modelica.org`). They are part of the standard distribution of a Modelica environment, such as Dymola.

Figure 1: A formula car in action, in this case, a Dallara 398 at Zandvort, NL.

3.1 Modeling of 1D mechanical components

The steering system and the wheel driving/braking system of the racing car are modeled with the 1-dimensional mechanical Modelica libraries `Modelica.Mechanics.Rotational` and `Modelica.Mechanics.Translational`. In figure 2 the Rotational library is displayed. This library contains basic elements, such as a rotational inertia, ideal gearboxes, shaft elasticity, bearing friction, clutches, brakes, and sensors. Additional components dedicated more specifically to power trains of vehicles, are available in the PowerTrain library, see [4] for details. These two libraries allow to build up quite detailed models of the mechanical part of the driving system of a vehicle, e.g., the detailed shifting behaviour of automatic gearboxes can be modeled conveniently. For the racing car, currently only a very simple model of the driving system is used. However, it is planned to add more details.

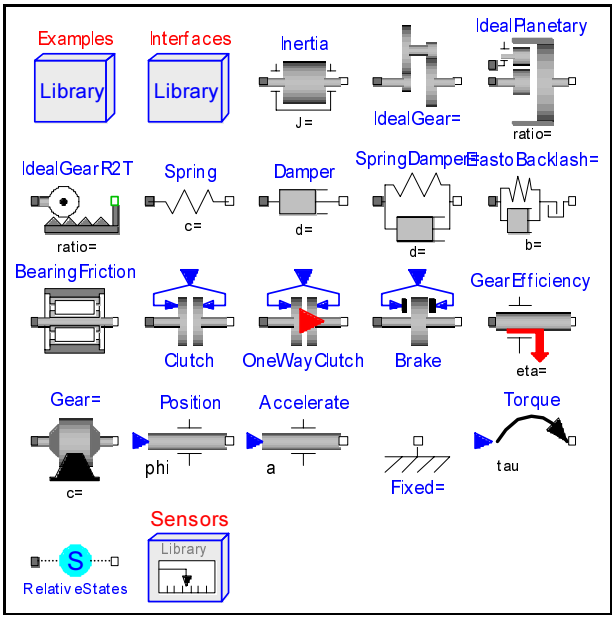

Figure 2: Library Modelica.Mechanics.Rotational.

3.2 Modeling of 3D mechanical components

The modeling of the vehicle chassis is performed with the new Modelica library `ModelicaAdditions.MultiBody` which is used to describe 3-dimensional mechanical systems, see figure 4. The components of the MultiBody library can be combined with the 1D-mechanical libraries mentioned in the previous subsection.

A simple example of the usage of this library can be seen in figure 3, where a simple pendulum with damping in the joint is modeled. Component **inertial** in figure 3 defines the inertial system of the model as well as the gravity acceleration. Component **revolute** describes a revolute joint which is connected to the inertial system. The axis of rotation is defined to be the vector $\{0,0,1\}$, i.e., a vector in z-direction resolved in the frame at the left side of the revolute joint. Since this frame is rigidly attached to the inertial frame, these two frames are identical. As a conse-

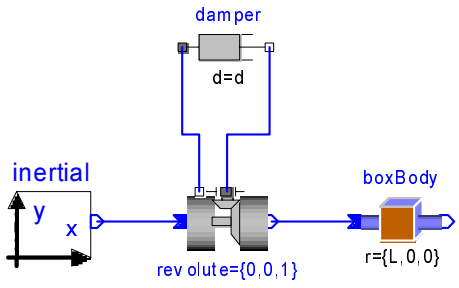

Figure 3: Example: Pendulum with damping in joint.

quence, the revolute joint rotates around the z-axis of the inertial system.

Component **boxBody** defines a box. The box is defined by lenght, width and height and some material properties, such as the density. The mass, center of mass and inertia tensor are calculated from this data. Additionally, the shape of the box is used for animation. Finally component **damper** is a 1D-element from the Rotational library and describes the velocity dependent damping in the joint axis. It is connected between the driving and the bearing flange of the joint. Both of these flanges are 1D mechanical connections.

The 3D-components are connected together at connectors **Frame_a** or **Frame_b**. Both connectors define a coordinate system (a frame) at a specific location of the component, which is fixed in the component, see figure 5. The frame, i.e., the corresponding connector, defines all the kinematic properties, such as the absolute rotation matrix from the frame to the inertial system ${}^0\mathbf{T}^a$ and the absolute position vector from the inertial system to the frame origin ${}^0\mathbf{r}^{0a}$, resolved in the inertial frame. Additionally, the connector contains the resultant cut-force ${}^a\mathbf{f}^a$ and cut-torque ${}^a\boldsymbol{\tau}^a$ at the origin of the frame, both resolved in Frame_a.

The Modelica MultiBody library has sublibraries for joints, cut-joints, parts, force elements, sensors and example systems, see upper part of figure 4. Ideally, all elements of the library can be connected together in any meaningful way. Unfortunately, this is not possible with the present version of this library[1]. The reason is that the orientation of a coordinate system is described with a redundant set of coordinates and this leads to an overdetermined, but consistent, set of equations if closed kinematic loops are present. Such systems cannot be handeled with the standard algorithms of object-oriented modelling techniques.

[1]The next release of this library is currently under development and has no longer the mentioned restrictions. Especially, the user does not have to define cut-joints and can connect the joints, parts and other elements in any meaningful combination.

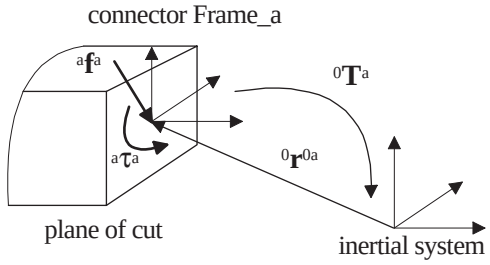

Figure 5: 3-dimensional mechanical connector.

Therefore, users have to remove joints, until the resulting connection of joints and parts forms a tree. Then the removed joints have to be added from the special sublibrary `CutJoints` (and not from sublibrary `Joints`) to arrive at the desired mechanical system. Cut-joints are primarily used to remove the redundant description of the orientation.

The one-degree-of-freedom joints (Revolute, Prismatic, Screw) may have a **variable** structure. That is, the joint can be **locked** and **unlocked** during the movement of a multibody system. This feature can be used to model brakes, clutches, stops or sticking friction. Locking is modelled with elements of the Rotational library, such as model classes `Clutch` or `BearingFriction`, which can be attached to the flange of such a joint.

4 The racing car model

The model of the racing car consists of the chassis, the driving and steering system and the road. In the following subsections, these components are discussed in more detail and it is explained how subcomponents, such as the suspension, are provided as parameterized model classes. The most important parts of the model are displayed in figure 6.

As already discussed in section 2, a formula car differs in several aspects from a conventional vehicle. For the modeling process, the most important differences are:

- In formula cars mostly ball joints are used to connect parts instead of bushings. This results in very stiff part connections. Therefore, more parts are kinematically constrained as it is usually the case, leading to bigger numbers of closed kinematic loops.

- The dominating suspension linkage in a formula car is a double wishbone linkage. This suspension type has 5 closely coupled kinematic loops. On the other hand, a McPherson suspension often used in conventional vehicles has only 2 kinematic loops.

Since every kinematic loop introduces nonlinear equations on position variables, both the simulation and the initialization is more difficult.

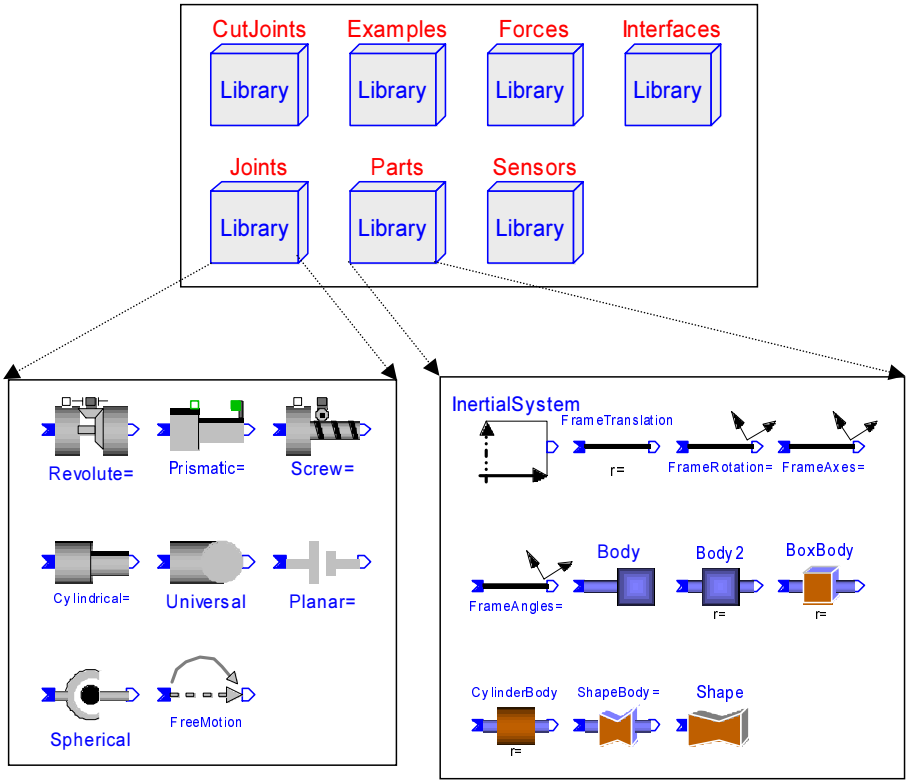

Figure 4: Library ModelicaAdditions.MultiBody.

4.1 Driving and steering system

In subfigure 1 of figure 6 the topmost level of the vehicle can be seen. It consists of the chassis model which has five rotational flanges (= 1D connectors visualized by filled squares at the border of the chassis model) to drive the four wheels and to steer the steering wheel of the driver. Elements of the 1D-rotational library can be used to built up quite detailled models of the power train to drive the wheels and of the steering system. In the figure, very simple models are used:

Via a differential gearbox, a driving torque is split and applied to the rear wheels. The driving torque is determined from a signal source. The front wheels are not driven and therefore the wheel flanges have no connection. Additionally, on all wheels a brake from the Rotational library could be attached. The steering angle is kinematically constrained by component `Position` and forced to follow the output of a signal source. Alternatively, a torque could be used together with a driver model which produces this torque.

4.2 Chassis model

Subfigures 2–5 of figure 6 display the various levels of the chassis model.

Subfigure 2 is the *chassis*, built up from the main car body (= rigid body + animation shape), a front and a rear suspension. The two suspension systems are rigidly attached to the car body. Via joint `FreeMotion`, the car body is connected to the inertial system, i.e., the 6 absolute translational and rotational coordinates, as well as the 6 absolute velocity and angular velocity coordinates of the car body are used to describe the movement of the car relatively to the inertial system.

The generic suspension systems have a steering connector. Since only the front wheels are steered, the steering angle of the rear wheels are set to a fixed angle of zero degrees via component `FixedSteering`.

In subfigure 3, the realization of the rear suspension can be seen. It consists of components for steering, dampers, drive train, suspension linkages, wheel body and tyre models. These elements can be combined to build up different kinds of suspension systems. The wheels at the upper and lower part of this

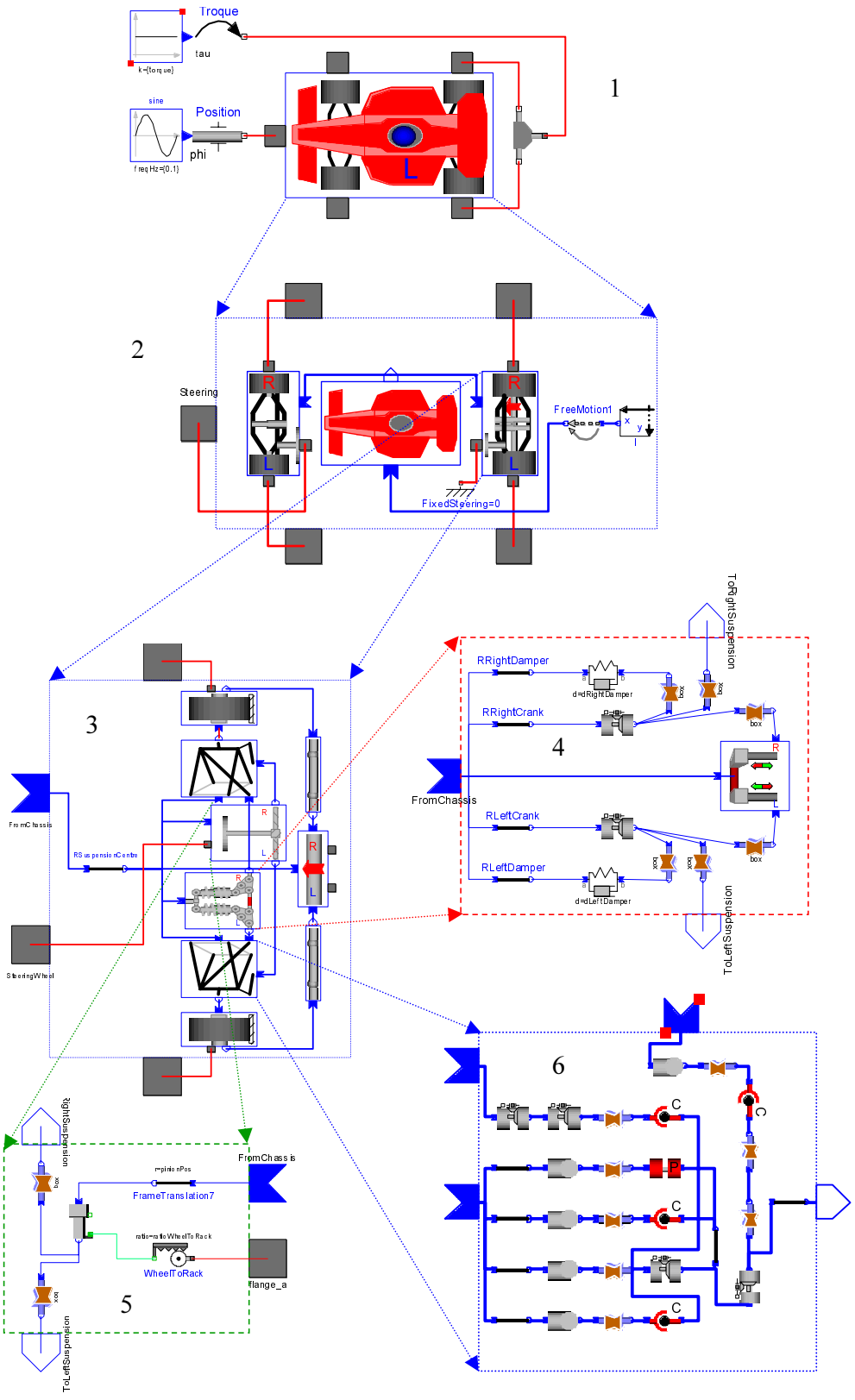

Figure 6: Composition diagrams of the different levels of the racing car model.

subfigure consist of a wheel body and a tyre model according to [5], chapter 2.6. There are three rotational flanges (= 1D-rotational connectors) on each suspension to regulate wheel rotation and steering wheel angle, as well as one Frame (= 3D connector) to attach the suspension system rigidly at a specific position and orientation on another part. The different suspension model classes have the same interfaces and are therefore completely interchangeable. However, the parameterization maybe different, e.g., a monodamper suspension has only parameters to define one damper while a twindamper has parameters for a left and a right damper.

In subfigure 4, the implementation of a twindamper suspension is shown and in subfigure 5 the steering system.

Finally, in subfigure 6 the model of the suspension linkages is displayed. This part contains 5 kinematic loops and therefore 5 cut-joints are utilized at appropriate places. The four joints at the lower left part of this subfigure are universal joints, i.e., they have 2 degrees of freedom. In reality, spherical joints are present at these places. The universal joints are used in order to remove the non-interesting self rotations of the rods of the suspension around the rod axes (since the inertia is small around these axes, the rotation can be neglected).

The 4 hierarchical levels of this model have been found appropriate. Of course, one could easily remove one level by implementing the car directly with steerings, dampers etc. and skip the level with front and rear suspensions. Since these units are fairly separated in a vehicle, e.g., they do not have any common closed loops, there are no advantages of using a flatter model.

Note, that with the subcomponents used to model the racing car, it is quite easy to describe completely different vehicles if they have a similiar structure. To illustrate this, a model `OffRoadTruck` is built up, see figure 7.

4.3 Parameterisation of subcomponents

In order that the subcomponents displayed in figure 6 are reusable, an appropriate parameterisation has to be utilized. Some aspects shall be discussed:

Parameter default values

In the car component library, the parameterisation is usually selected in such a way that the parameters do not influence each other, i.e., if one parameter is changed, there is usually no need to modify another parameter as well.

For all parameters, realistic default values are used. For example, a link arm is by default modelled as a bar with equally distributed mass, depending on the length of the rod: $\overline{r}_{CM} = \frac{\overline{r}}{2}$, $m = \rho|\overline{r}|$, $\mathbf{I}_{11} = \frac{m}{3}(\overline{r}_2^2 + \overline{r}_3^2)$ and the density ρ of, say, steel is taken. Therefore, whenever the length of the rod is changed, the position of the center of mass, the mass and the inertia tensor are changed appropriately. It is of course possible to overwrite at any time the default values by, e.g., supplying explicit numeric values for the mass and the inertia tensor. This approach has the advantage that by providing only a few parameters, a first rough model of a vehicle can be built and simulated. In further steps, the parameters can be better adapted to the actual vehicle data by overwriting the default values.

Parameterisation of closed kinematic loops

Usually, in every subcomponent the coordinate system of one of the 3D connectors is used as a reference frame for this component, and all other frames are selected to be parallel to this frame. As a consequence, all local vectors are expressed in the reference frame.

If a subcomponent is parameterized which does not have kinematic loops, such as the damper system in subfigure 4 of figure 6, it is most convenient to just directly use the parameterization necessary to describe the base components used. This means, e.g., that position vectors are defined as relative quantities between local frames.

On the other hand, if a subcomponent has kinematic loops (see, e.g., subfigure 6 of figure 6) such a parameterization leads to difficulties, since the generalized coordinates of the joints in the loops have to be determined at initial time by the solution of nonlinear algebraic equations where the start values may not be good. It is more robust to select as independent parameters the position vectors from the reference frame to the ”essential“ points, e.g., the connection points, at the initial configuration and compute the needed relative position vectors by simple vector addition rules.

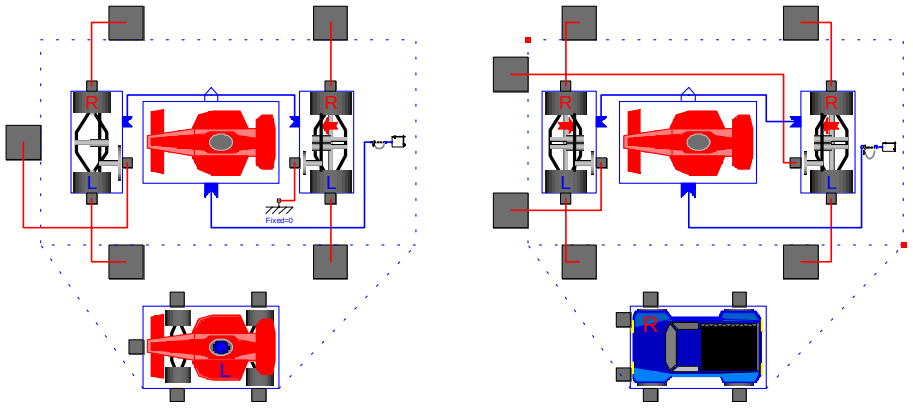

Figure 7: The `FormulaCar` compared to the `OffRoadTruck` model. Already defined components can be reused to built-up easily quite different cars.

As a consequence, at the initial position the kinematic loops are always consistent.

4.4 Road

The wheel and tyre models need the altitude and the friction coefficient at the point of contact of a tyre with the road. Changing the road description should not require to modify the car model itself. Since the road and the tyre exchange data, the standard way in Modelica would be to have a road component which is connected to all four wheels of a car. This would result in a close coupling of car and road model.

A better alternative is to see the road as a *physical field* and use the `inner/outer` Modelica language constructs designed for the description of such fields. In this case, the prototype of a function for the road properties is defined as:

```
partial function roadInterface
   input  Real x   "x-position";
   input  Real y   "y-position";
   input  Real v   "velocity";
   output Real z   "z-position";
   output Real mue "friction coeff.";
end road;
```

that is, given the x- and y-coordinates of a contact point, as well as the absolute value of the velocity vector at the contact point parallel to the road, the z-coordinate of the road and the coefficient of friction at this point are computed. Within the wheel/tyre model, a declaration of the form

```
outer function road = roadInterface;
```

is present, and function **road** is called when needed in the tyre model. Whenever this function is called, the actual function body is searched all the levels up in the model hierarchy, until a corresponding **inner** declaration of the following form is found:

```
inner function road = roadProfile1;
```

In this case, the actual function called in the tyre model instances is function **roadProfile1**. As a consequence, the road properties can be defined at the top-level of the model and can also be easily changed. Usually, a function is supplied which reads a table from a file defining the road profile. To summarize, inner/outer functions in Modelica can be seen as pointers to functions where the actual function body is defined at higher hierarchical levels.

5 Simulations results

The overall model of the racing car consists of 73 bodies with shape information for animation, 92 revolute joints, 24 spherical joints and 24 closed kinematic loops. Dymola needs a few minutes on a PC to transform this system to state space form with 30 states and about 6500 algebraic variables (reduced from a set of about 60000 unknown algebraic variables).

In figure 8 the animation view of the racing car is shown. It consists of a CAD-picture of the main body and of graphical objects, such as cylinders and boxes, to display other parts, such as the suspension systems.

In figure 9 a typical simulation result of a driving manoeuvre is shown, where the racing car drives through a curve and starts to skid.

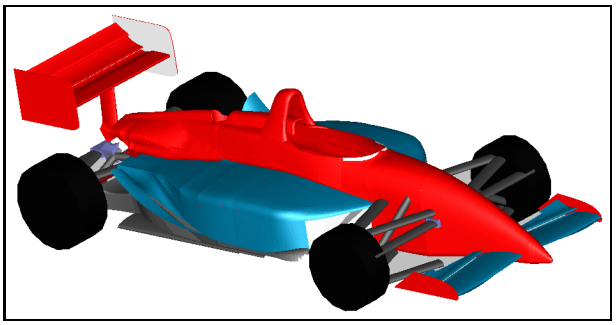

Figure 8: Animation view of racing car model.

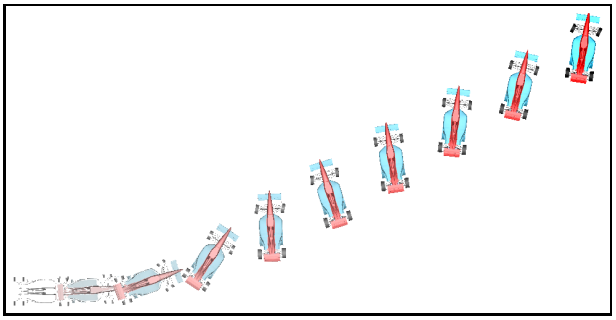

Figure 9: Driving through a curve with skidding.

6 Conclusions

The racing car model described in this article demonstrates that it is possible and suitable to model large 3-dimensional mechanical systems with Modelicas multi-body library. Since it is very easy to combine this library with other Modelica libraries, such as 1-dim. Rotational-, PowerTrain-, Hydraulics-, Blocks-library, it is straighforward to add detailled models of other components of the car, such as the power train.

In [6] the results of a feasibility study are presented, where a detailed model of a conventional vehicle has been realized, including the 3-dim. model of the chassis, engine, automatic gearbox and hydraulics to control the gearbox, as well as simulations performed in Dymola. For the 3D-mechanics part, an existing ADAMS [1] model of the vehicle chassis was used and transformed to Modelica with a newly developed translator.

The racing car model shows clearly that it is advantageous to model the vehicle mechanics directely in Modelica, because the hierarchical structering makes it possible to browse and understand all the details of the model and to re-use subcomponents for other vehicles.

7 Acknowledgments

The development of the ModelicaAdditions.MultiBody library was in parts supported by the European Commission under contract IST-199-11979 with DLR under the Information Societies Technology as the project entitled ”Real-time simulation for design of multi-physics systems”.

The authors would like to thank Hilding Elmqvist, Matthijs Langelaar, Sven Erik Mattsson, Hans Olsson and Andrea Burzoni for their support.

References

[1] *ADAMS*. Mechanical Dynamics Inc., Homepage: http://www.adams.com/

[2] *Dymola*. Homepage: http://www.dynasim.se/

[3] *Möller A:* Studies of Race Car Vehicle Dynamics. Master thesis, Lund Institute of Technology, Division of Mechanics, 1999.

[4] *Otter M., M. Dempsey* and *C. Schlegel:* Package PowerTrain: A Modelica library for modeling and simulation of vehicle power trains. In *Proceedings of Modelica'2000*, Lund, Oct. 2000.

[5] *Rill G.:* Simulation von Kraftfahrzeugen. Vieweg, 1994.

[6] *Tiller M., Bowles P., Elmqvist H., Brück D., Mattsson S.-E., Möller A., Olsson H.* and *Otter M.:* Detailed Vehicle Powertrain Modeling in Modelica. In *Proceedings of Modelica'2000*, Lund, Oct. 2000.

[7] *William F.* and *Milliken D.L.:* Race Car Vehicle Dynamics, 1995.