Proceedings
of the 3rd International Modelica Conference,
Linköping, November 3-4, 2003,
Peter Fritzson (editor)

Christian Schweiger, Martin Otter
Institute of Robotics and Mechatronics, *DLR* :
Modeling 3D Mechanical Effects of 1D Powertrains
pp. 149-158

Modelling 3D Mechanical Effects of 1D Powertrains

Christian Schweiger[*]   Martin Otter[†]

German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Oberpfaffenhofen, 82234 Weßling, Germany
http://www.robotic.dlr.de/control/

Abstract

It is described how to extend one-dimensionally modelled rotational mechanical systems such that they can be mounted on three-dimensional multi-body system models without neglecting any dynamic effects. This is performed by adding support torques to existing drive train elements and by introducing new components that take care of the gyroscopic torques and the transformation of one-dimensional into three-dimensional support torques. It is demonstrated that this approach is convenient for the user and leads to efficient simulation code.

1   Introduction

Dependent on the point of view, there are traditionally two ways of modelling and simulating powertrains. It is possible to obtain the complete dynamics of a powertrain using multi-body systems simulation. One drawback is the high effort needed in describing the powertrain in its complete geometry. Another disadvantage is the comparative low efficiency of this method concerning simulation of friction elements as used in clutches and gearboxes. Since the latter are the essential part of powertrains with a major impact on the dynamics, multi-body systems simulation seems to be not the appropriate method for realtime simulation.

For realtime simulation of powertrains, hybrid discrete-continuous modelling techniques became quite common. They allow modelling not only by differential equations, but by additional boolean equations. This combination is very useful for modelling of variable structure systems, especially of friction elements. As friction elements are considered only one-dimensional (1D), it was acceptable to neglect three-

dimensional (3D) mechanical effects in the past. Their neglect allowed to define the considered powertrain in simply one dimension with high efficiency.

With the growing level of detail in the vehicle dynamics area, there is an upcoming interest on the influence introduced by the powertrain dynamics. For example, the vehicle dynamics is influenced by the *support torques*, which act on the vehicle body over the mounting of the gearbox. Also *gyroscopic torques* could be important during gear shifts of a yawing vehicle.

The objective of the work described in this paper is to merge together the advantages of both 1D and 3D modelling of powertrains by using appropriate components, which provide the resulting torques of the powertrain to the 3D vehicle dynamics model without neglecting any dynamic effects.

The basic idea is as follows: The bearings of a 1D powertrain modelled with the 1D Rotational library (Modelica.Mechanics.Rotational) and/or the 1D PowerTrain library [1] are fixed rigidly on a *carrier body* which moves in 3D space and which is modelled as 3D multi-body system with the new Modelica MultiBody library [2]. The 3D movement of the whole system is described correctly and without any neglections in the following way:

- It is assumed that all rotating bodies in the powertrain have rotational symmetry.

- The carrier body on which the powertrain is fixed has to be defined in such a way that it has the common mass, the common center-of-mass and the common inertia tensor of the body together with the powertrain under the assumption that the rotating bodies in the powertrain are fixed relative to the carrier body. It does not matter at which angle the rotating parts are fixed, since it is assumed that the bodies have rotational symmetry.

---
[*]Christian.Schweiger@dlr.de
[†]Martin.Otter@dlr.de

This approach is also convenient for the user: It suffices to measure the mass, the center of mass and the inertia tensor of, say, a complete automatic gearbox or get this data from a CAD system. Additionally, only the inertias of the rotating shafts around their axis of rotation are needed, as required for 1D modelling of powertrains. This approach is simpler and more practical as requiring to get mass, center of mass and complete inertia tensor of every single piece of a powertrain.

In the next section it is described how to extend the existing 1D components such that support torques are computed. Section 3 introduces hybrid 1D/3D components. Section 4 discusses the implementation of a gearbox modelled solely with multi-body components in order to be able to compare the different modelling philosophies. An extract of the tests performed on the components are presented in Sec. 5. The paper closes with an application example in Sec. 6 and subsequent conclusions.

2 Support Torque in 1D

As discussed in the previous section, the 3D mechanical effects of a powertrain comprise the support torques for components which interact with the powertrain housing, e.g., gears and brakes. In the past, these support torques have not been considered neither in the `Modelica.Mechanics.Rotational` nor in the `PowerTrain` library. This was disadvantageous even for simple 1D powertrains: It was, e.g., not possible to model the dynamics of a gearbox housing, mounted on the ground via spring-damper-systems.

In order to overcome this deficiency, it is necessary to introduce an additional 1D connector representing the bearing flange. This bearing connector can be used to fix components on the ground or on other rotating elements or to combine it with force elements. As a side effect, the support torque is computed explicitly.

For backward compatibility and convenience reasons, it is desired not to be forced to connect this connector in every case. With the Modelica operator **cardinality** it is possible to inquire the number of connections to a connector. This information is used to provide different equations in case the connector is not connected. Otherwise, the duplication of many models would have been necessary.

In the following, it is shown for the model `Modelica.Mechanics.Rotational.Ideal-Gear`, Fig. 1(a), how the respective components of the rotational library are adapted. In Lstg. 1 the flange

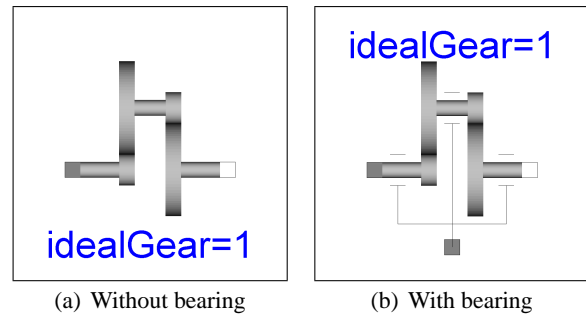(a) Without bearing   (b) With bearing

Figure 1: Ideal gearbox

connectors from the Rotational library are recalled.

Listing 1: Flange connectors from Rotational library

```
within Modelica.Mechanics.Rotational.Interfaces;
connector Flange
  import SI = Modelica.SIunits;
  SI.Angle  phi "absolute flange angle";
  SI.Torque tau "cut-torque in flange";
end Flange;

connector Flange_a = Flange;
connector Flange_b = Flange;
```

The previous implementation of the `IdealGear` model is shown in Lstg. 2.

Listing 2: Previous gearbox implementation

```
model IdealGear
  import Modelica.Mechanics.Rotational;

  Rotational.Interfaces.Flange_a flange_a;
  Rotational.Interfaces.Flange_b flange_b;

  parameter Real ratio=1;
equation
  flange_a.phi = ratio*flange_b.phi;
  0 = ratio*flange_a.tau + flange_b.tau;
end IdealGear;
```

Since this change has to be carried out for a lot of models, it is advisable to create a superclass including common components and equations, cf. Lstg. 3.

Common equations are the torque balance and the computation of the relative angles with respect to the bearing flange.

If the bearing flange is not connected, i.e. **cardinality**(bearing) == 0, the Modelica default connection rule defines bearing.tau = 0 and the additional equation supplied in the corresponding if-section sets the bearing angle to zero, i.e., the bearing does not move.

Otherwise, the support torque is identical to the torque of the bearing flange and the bearing angle

Listing 3: Superclass including bearing torque

```
partial model TwoFlangesAndBearing
  import Modelica.Mechanics.Rotational;

  Rotational.Interfaces.Flange_a flange_a;
  Rotational.Interfaces.Flange_b flange_b;
  Rotational.Interfaces.Flange_a bearing;

  Modelica.SIunits.Torque tau_support;
  Modelica.SIunits.Angle  phi_a;
  Modelica.SIunits.Angle  phi_b;
equation
  0 = flange_a.tau + flange_b.tau + tau_support;

  phi_a = flange_a.phi - bearing.phi;
  phi_b = flange_b.phi - bearing.phi;

  if cardinality(bearing) == 0 then
    bearing.phi = 0;
  else
    bearing.tau = tau_support;
  end if;
end TwoFlangesAndBearing;
```

`bearing.phi` is defined from the component connected to the bearing.

Using this superclass and inheriting from it, it is a straightforward procedure to adapt the components. The expression `flange_a.phi` has to be changed simply to `phi_a` and `flange_b.phi` to `phi_b`, which leads to the implementation given in Lstg. 4. The icon shown in Fig. 1(b) indicates the function of the bearing flange.

Listing 4: New gearbox implementation

```
model IdealGear
  extends TwoFlangesAndBearing;

  parameter Real ratio=1;
equation
  phi_a = ratio*phi_b;
  0 = ratio*flange_a.tau + flange_b.tau;
end IdealGear;
```

For components which had up to now only one flange (Torque, Move etc.), the procedure is similar.

The described changes have been performed directly to the components of the Modelica Standard Library and the PowerTrain Library, since these changes are backward compatible, i.e., existing user models are not affected by this addition.

3   Hybrid 1D/3D Components

The 3D mechanical effects induced by a powertrain on its carrier body are in fact additional torques, see Sec. 1. In order to produce these torques on the car-

rier body, some components are needed with both 1D (flanges) and 3D (frames) connectors. They are described in this section.

3.1   3D Connector "Frame"

The definition of the needed 3D connector of the MultiBody library is shortly sketched. For more details, see [2]. The variables of the Frame connector are displayed in Fig. 2.
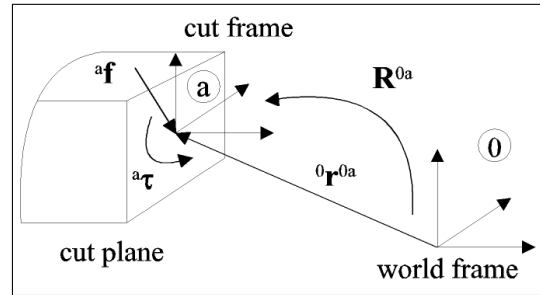

Figure 2: Variables of 3D connector `Frame`

A coordinate system frame a is rigidly fixed at an attachment point of a 3D mechanical part. This frame is described with respect to the world frame (i.e. an inertial coordinate system) by the

- position vector $^0\mathbf{r}^{0a}$ that is directed from the origin of the world frame to the origin of frame a and is resolved in the world frame and by the

- orientation object $\mathbf{R}^{0a}$ describing the relative orientation between the world frame and frame a.

To ease usage, the MultiBody library is designed such that knowledge about the actual description form of orientation is not necessary. This is achieved by providing a pre-defined type `MultiBody.Frames.Orientation` and utility functions in `MultiBody.Frames` operating on instances of this type. In the sequel, only the utility function `Frames.AngularVelocity2` is needed to compute the angular velocity of the frame, resolved in the local coordinate system attached to the part.

It is assumed that a cut is performed between mechanical parts that shall be connected together at frame a. In the cut plane a resultant cut force $^a\mathbf{f}$ and a resultant cut torque $^a\tau$ act on frame a. Both vectors are resolved in this frame.

The four previously defined variables are used in connector `Frame`, see Lstg. 5. The additional connectors `Frame_a` and `Frame_b` have the identical definition as connector Frame. The only difference is that

Listing 5: MultiBody connector Frame

```
connector Frame
  import SI = Modelica.SIunits;
  SI.Position[3]                r_0;  // = ⁰r⁰ᵃ
  MultiBody.Frames.Orientation R;     // = R⁰ᵃ
  flow SI.Force [3]             f;    // = ᵃf
  flow SI.Torque[3]            t;    // = ᵃτ
end Frame;

connector Frame_a = Frame;
connector Frame_b = Frame;
```

Listing 6: Implementation of Mounting1D

```
model Mounting1D
  import Modelica.Mechanics.Rotational;
  Rotational.Interfaces.Flange_b flange_b;

  parameter Modelica.SIunits.Angle phi0 = 0;
  parameter Real[3] n={1,0,0};
equation
  flange_b.phi = phi0;
  frame_a.f    = zeros(3);
  frame_a.t    = -n*flange_b.tau;
end Mounting1D;
```

`Frame_a` and `Frame_b` have different icons in order to be able to distinguish `Frame` connectors more easily in a composition diagram. The cut force and cut torque are flow variables in order that the force and torque balance at a point where several components are connected together is fulfilled. Note, that two connected frames (a and b) coincide, since `a.r_0 = b.r_0` and `a.R = b.R` due to the connection rules of Modelica.

3.2　New Component "Mounting1D"

In order to acquire support torques from the powertrain and to propagate them to the carrier body, a new component called `Mounting1D` is used, see Fig. 3. It has the equations of the `Rotational.Fixed` component and in addition a 3D frame connector for the mounting on a multi-body component, as well as a parameter vector `n` that defines the direction of the axis of rotation of the 1D flange connector. At the same time, `n` defines the direction of the support torque.
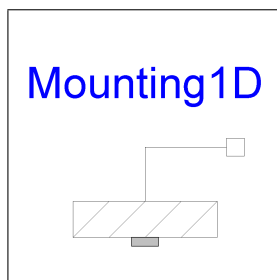
Mounting1D

Figure 3: Component `Mounting1D` for propagating support torques to carrier body

This component transforms the 1D bearing torque into 3D space, see Lstg. 6, and enables 3D movement of all 1D elements connected to it.

All components of a powertrain that are connected to a common `Mounting1D` element need to have the same axis of rotation along parameter vector `n`. This means that, e.g., bevel gears where the axis of rotation of `flange_a` and `flange_b` are different cannot be described properly by connecting to the `Mounting1D` component. It is discussed later, how to treat this case.

3.3　New Component "Rotor1D"

Powertrain parts rotating relative to their carrier body exert gyroscopic torques on this body, if the carrier is rotating. This effect can be mathematically described with a so-called gyrostat as illustrated in Fig. 4. It consists of a carrier and a body with rotational symmetry, called *rotor*, that is mounted on the carrier by rigid bearings.
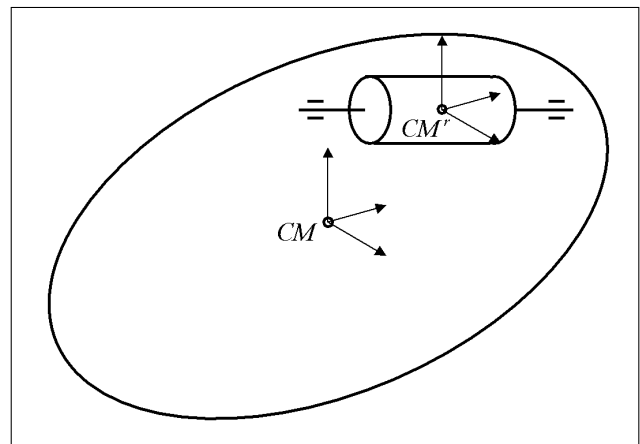
Figure 4: Gyrostat consisting of carrier and symmetric rotor

Two coordinate systems are present: The carrier frame is fixed in the carrier at the common center of mass *CM* of the total system. The rotor frame is also fixed in the carrier but at the center of mass *CMʳ* of the rotor. The rotor frame is parallel to the carrier frame.

According to, e.g., [3, 4], the equations of motion of this combined system are described by

$$\mathbf{J}\dot{\omega} + \mathbf{J}^r\dot{\omega}_{\text{rel}} + \omega \times (\mathbf{J}\omega + \mathbf{J}^r\omega_{\text{rel}}) = \tau \qquad (1)$$

with the absolute angular velocity of the carrier, ω, the angular velocity of the rotor relative to the carrier, ω_{rel}, the inertia tensor of the total system with respect to its common center of mass CM, $\mathbf{J}$, the inertia tensor of the rotor with respect to the center of mass of the rotor CM^r, $\mathbf{J}^r$, and the external torque with respect to CM, τ. All vectors and tensors are resolved in the carrier frame. Reordering of terms yields

$$\underbrace{\mathbf{J}\dot{\omega}+\omega \times \mathbf{J}\omega}_{\tau_{\text{body}}}+\underbrace{\mathbf{J}^r \dot{\omega}_{\text{rel}}+\omega \times \mathbf{J}^r \omega_{\text{rel}}}_{\tau_{\text{gyro}}} = \tau \ . \quad (2)$$

By comparison with the equations of motion for a rigid body, it can be seen, that the term τ_{body} represents the contribution of the total system with the rotor fixed on the carrier and τ_{gyro} represents the additional contribution caused by the rotation of the rotor relatively to the carrier. Since the properties of the total system with the non-moving rotor are modelled completely by the carrier body, see Sec. 1, τ_{gyro} has to be considered in the Rotor component for gyroscopic torques.

The property of the rotor axis of rotation coinciding with one of its principal axis of inertia yields the simplification

$$\mathbf{n}J^r \dot{\omega}_{\text{rel}}+\omega \times \mathbf{n}J^r \omega_{\text{rel}} = \tau_{\text{gyro}} \quad (3)$$

with $\mathbf{n}$ being a unit vector in direction of the axis of rotation of the rotor, J^r the moment of inertia around $\mathbf{n}$ and ω_{rel} the absolute value of the relative angular velocity of the rotor with respect to the carrier.

Whereas (3) considers solely the rotational degrees of freedom of the total system, the rotor has an additional degree of freedom of its own, as an external torque $\tau^r(t)$ is exerted on it. In [3, 4], the respective equation of motion is derived as

$$\mathbf{n}^T (\mathbf{J}^r \dot{\omega}+\mathbf{J}^r \dot{\omega}_{\text{rel}}) = \mathbf{n}^T \tau^r \ . \quad (4)$$

In a similar way as above, this equation can be simplified to

$$J^r \mathbf{n}\dot{\omega}+J^r \dot{\omega}_{\text{rel}} = \tau^r(t) \ . \quad (5)$$

The new component `Rotor1D` is constructed by using the `Modelica.Mechanics.Rotational.Inertia` model as a basis, attaching a 3D `Frame` connector and adding the equations from (3) and (5). The Modelica implementation is shown in Lstg. 7.

The parameter `NeglectCoupling` was introduced in order to optionally neglect the term `J*(n*z_a)`, which corresponds to $J^r \mathbf{n}\dot{\omega}$ in (5). This term is usually negligible if the powertrain accelerates much faster as the base body (this is, e.g., the case in

Listing 7: Implementation of Rotor1D

```
model Rotor1D
  import Modelica.Mechanics.Rotational;

  Rotational.Interfaces.Flange_a flange_a;
  Rotational.Interfaces.Flange_b flange_b;
  MultiBody.Interfaces.Frame_a   frame_a;

  parameter Modelica.SIunits.Inertia J = 1;
  parameter Real                  n[3] = {1,0,0};
  parameter Boolean NeglectCoupling = false;

  Modelica.SIunits.Angle              phi;
  Modelica.SIunits.AngularVelocity    w_a[3];
  Modelica.SIunits.AngularAcceleration z_a[3];
  Modelica.SIunits.AngularVelocity    w;
  Modelica.SIunits.AngularAcceleration a;
equation
  flange_a.phi = phi;
  flange_b.phi = phi;

  w = der(phi);
  a = der(w);
  J*a = flange_a.tau + flange_b.tau -
    (if NeglectCoupling then 0 else J*(n*z_a));

  w_a = MultiBody.Frames.angularVelocity2
        (frame_a.R, der(frame_a.R));
  z_a = der(w_a);

  frame_a.f = zeros(3);
  frame_a.t = n*(J*a) + cross(w_a, n*(J*w));
end Rotor1D;
```

vehicle powertrains). The essential advantage is that an algebraic loop is removed since then there is only an action on acceleration level from the powertrain to the base body but not vice versa.

3.4 New Component "BevelGear1D"

A new component is needed for 1D modelling of gearboxes with non-parallel axes, see Fig. 5. In general, the axes of rotation of `flange_a` and `flange_b` and the direction of the support torque vector are different in this case. Therefore, it is necessary to attach the 3D connector directly to this component.

In accordance to Fig. 6, the bevel gear is characterized by

$$i = \frac{\omega_{\text{in}}}{\omega_{\text{out}}} = -\frac{\tau_{\text{out}}}{\tau_{\text{in}}} \quad (6)$$

with the gear speed ratio i, shaft angular velocities ω and shaft torques τ [5]. As illustrated in Fig. 6, the indices refer to the input and output shaft respectively.

With $\mathbf{n}_{\text{in}}$, $\mathbf{n}_{\text{out}}$ vectors in direction of the input and output shaft, respectively, a 3D torque balance results in

$$\mathbf{0} = \tau_{\text{in}} \frac{\mathbf{n}_{\text{in}}}{|\mathbf{n}_{\text{in}}|} + \tau_{\text{out}} \frac{\mathbf{n}_{\text{out}}}{|\mathbf{n}_{\text{out}}|} + \tau_{\text{support}} \ . \quad (7)$$
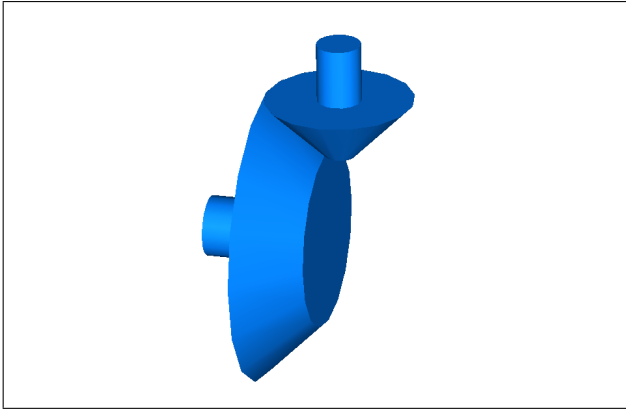
The implementation is shown in Lstg. 8.

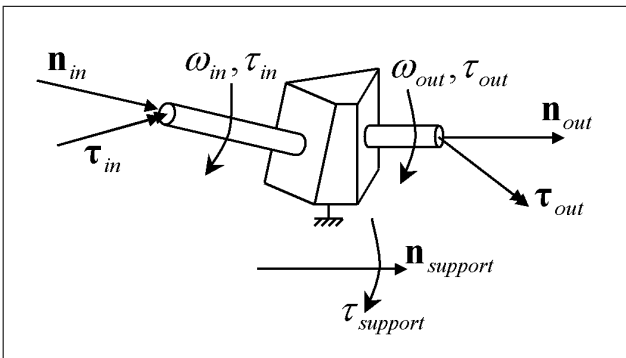Figure 5: Gearbox with non-parallal axes

Figure 6: Quantities of a bevel gear

3.5 Animation

Most components of the PowerTrain library are animated. All these components have been equipped with a 3D Frame connector to provide the information needed for deducing the 3D position and orientation of the animation shapes. The possibility to switch off animation and to completely remove the corresponding code was conserved.

4 New Component "GearConstraint"

In order to be able to model gearboxes completely in 3D, a component to provide the gear constraint (6) for multi-body systems was introduced. By default, the component allows to model gearboxes with nonparallel shafts as well.

In a first step, see Fig. 7, the 3D gear constraint was implemented without using 1D rotational components in order to be not forced to take care for support torques. Therefore, the constraint equations have been

Listing 8: Implementation of BevelGear1D

```
model BevelGear1D
  import Modelica.Mechanics.Rotational;

  parameter Real ratio=1;
  parameter Real n_a[3]={1,0,0};
  parameter Real n_b[3]={1,0,0};
protected
  parameter Real e_a[3]=n_a/sqrt(n_a*n_a);
  parameter Real e_b[3]=n_b/sqrt(n_b*n_b);
public
  Rotational.Interfaces.Flange_a flange_a;
  Rotational.Interfaces.Flange_b flange_b;
  MultiBody.Interfaces.Frame_a frame;
equation
  flange_a.phi = ratio*flange_b.phi;
  0 = ratio*flange_a.tau + flange_b.tau;

  frame.f = zeros(3);
  frame.t = -flange_a.tau*e_a - flange_b.tau*e_b;
end BevelGear1D;
```

given as Modelica text taking into account

$$\tau_{in} = \boldsymbol{\tau}_{in}\,\frac{\mathbf{n}_{in}}{|\mathbf{n}_{in}|} \;, \qquad \tau_{out} = \boldsymbol{\tau}_{out}\,\frac{\mathbf{n}_{out}}{|\mathbf{n}_{out}|} \qquad (8)$$

with the corresponding frame cut torque $\boldsymbol{\tau}$ and axis of rotation $\mathbf{n}$ as shown in Fig. 6. The constraints (6) implicitly define the torques to be applied at the two revolute joints.
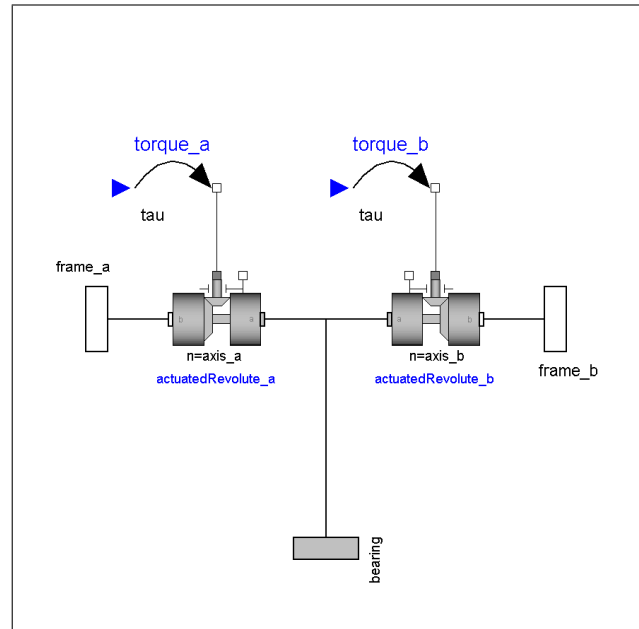

Figure 7: 3D gear constraint without 1D rotational components

Since equivalent equations are provided by Modelica.Mechanics.Rotational.IdealGear, in a second step the 3D gear constraint was imple-

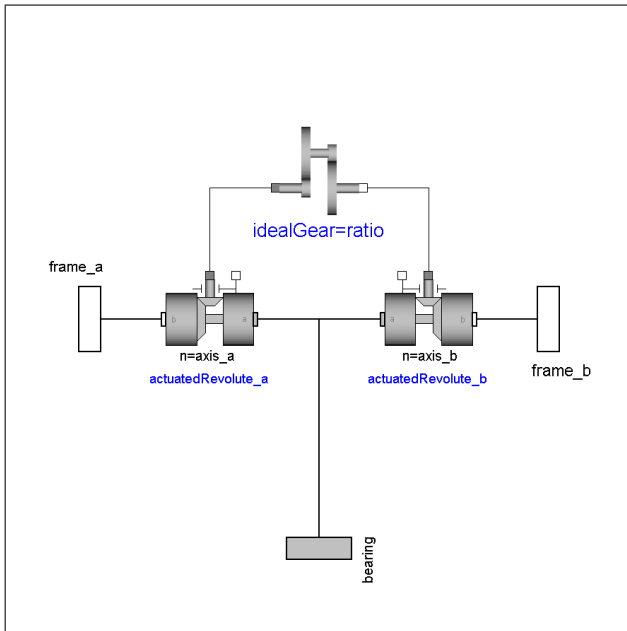mented as shown in Fig. 8. No additional equations are necessary.

Figure 8: 3D gear constraint using model IdealGear

5 Validation Examples

The following examples illustrate the usage of the introduced components and are used in order to validate the design by comparison of 1D with 3D realizations.

5.1 Gyrostat

As mentioned above, a so-called gyrostat consists of a carrier body and a rotor, which is mounted on the carrier body. There are two possibilities of modelling such a system, especially the rotor. One possibility is the usage of a rotating 3D body. Alternatively, the rotor could be modelled by a non-rotating 3D body in combination with the Rotor component described in Sec. 3.3. It was intended to show the equivalence of both possibilities. Several simulations have been carried out with different joint combinations, axis directions and driven or free rotor. In the following, a selected setup is presented.

In the example shown in Fig. 9, the two rotors to be compared are mounted in both cases on a carrier cylinder of their own which is able to move in three rotational degrees of freedom, since the latter is mounted to the ground by a spherical joint.

At the start of the simulation, the carrier cylinders are in an elevated position and start moving due to
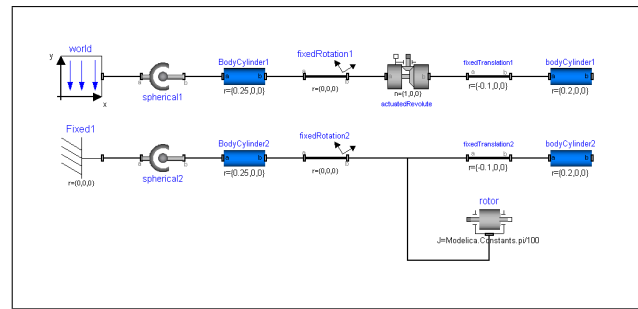
Figure 9: Modelica model of rotating body and non-rotating body with rotor

gravitation. The rotors are not actuated and have an initial angular velocity $\omega = 10$ rad/s. All relevant signals, especially the mounting forces (Fig. 11) and the body orientations, are identical.
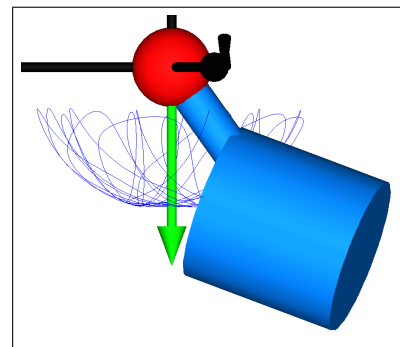

Figure 10: Animation of the two gyrostat systems

5.2 Body-mounted Actuator

Another example is shown in Fig. 12. It consists of a robot arm that is connected with a revolute joint to the base (i.e. world frame). On the (moving) robot arm a gearbox and a motor is present that drive the revolute joint. One wheel of the gearbox is rigidly attached to the axis of rotation of the revolute joint.

Again, two different Modelica models of this system are compared. One model was implemented using solely 3D components, cf. Fig. 13(a). The other model replaces some 3D components by 1D equivalences, cf. Fig. 13(b).

Both implementations yield identical results.

6 Application: Automatic Gearbox

The basic intention of modelling 3D mechanical effects of 1D powertrains was to be able to examine the
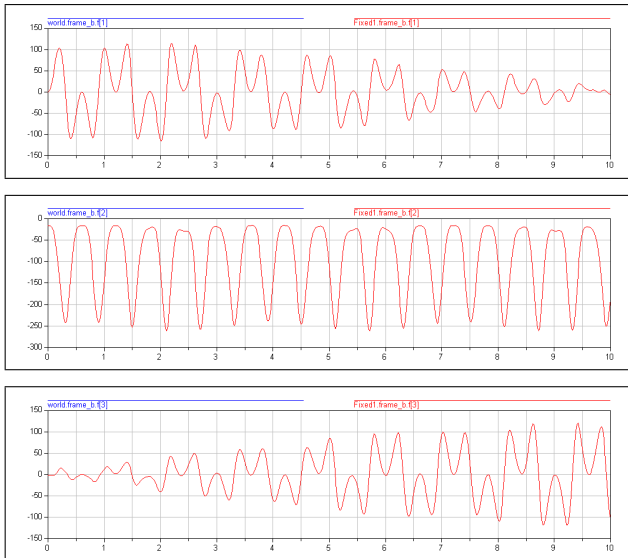
Figure 11: Mounting forces of the two gyrostat systems in x-, y-, z- direction in the world frame
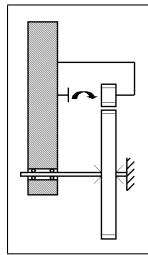
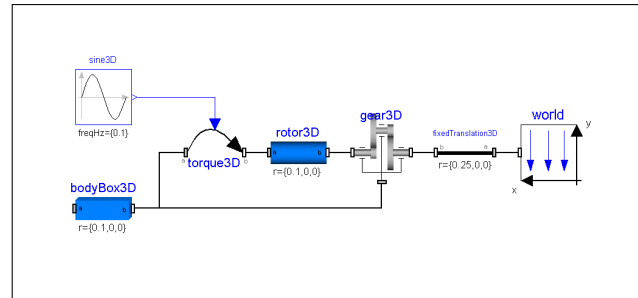

Figure 12: Sketch of body-mounted actuator

interaction of vehicle and powertrain dynamics. The example discussed in this section sketches the idea of merging both modelling areas.

A six-speed automatic gearbox based on a Lepelletier wheelset [6], Fig. 14, is considered. A corresponding model is available in the PowerTrain library.

This model was changed such, that the previous 1D rotational mechanical components have been replaced by these introduced above, cf. Fig. 15. As a consequence, the Lepelletier wheelset model was equipped with a MultiBody frame connector.

The obtained component is then used in the setup shown in Fig. 16. A carrier body is connected to the ground by a revolute joint and moved similar to a yawing vehicle. The yaw rate was increased up to 2 rad/s and then held constant. The automatic gearbox is mounted on the carrier body and connected with two rotor components at the input and output shafts, respectively. The drivetrain is accelerated by a torque of 10 Nm.

Figure 17(a) shows the angular velocities of the ro-

(a) Solely using 3D MultiBody components
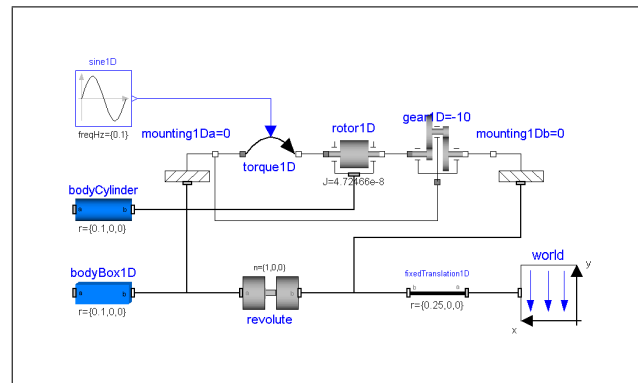
(b) Using both 3D and 1D components

Figure 13: Modelica models of body-mounted actuator

tors at the input and output shaft. At a simulation time of 4 s, a gear shift is initiated from the third to the fourth gear, reducing the system's ability to accelerate. The influence exerted on the carrier body can be seen in Fig. 17(b).

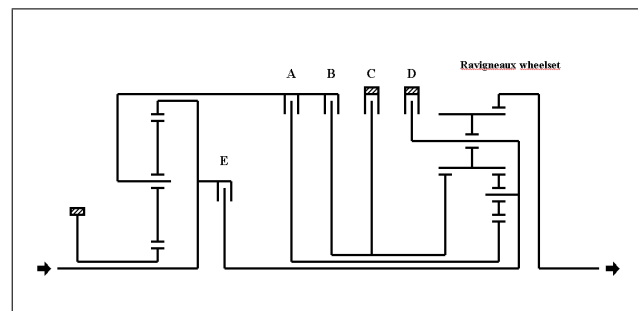In Fig. 18, an animation screenshot illustrates the assembly of the gearbox.

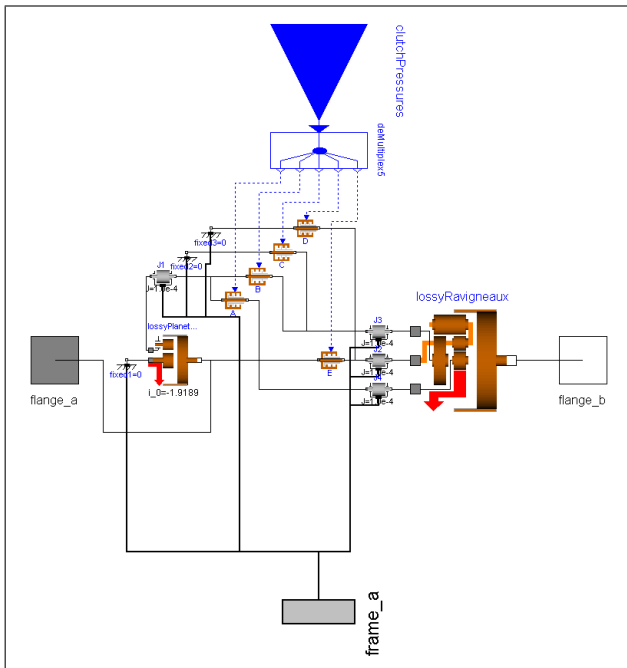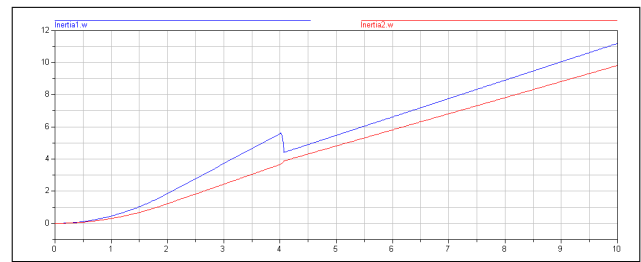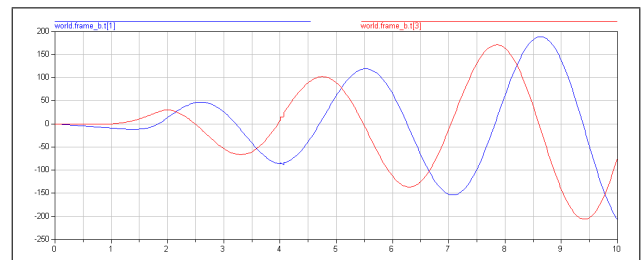
Figure 14: Schematic of Lepelletier wheelset

Figure 15: Modelica model of Lepelletier wheelset

(a) Angular velocities of rotors

(b) Support torques of carrier body in x- and z-direction of world frame

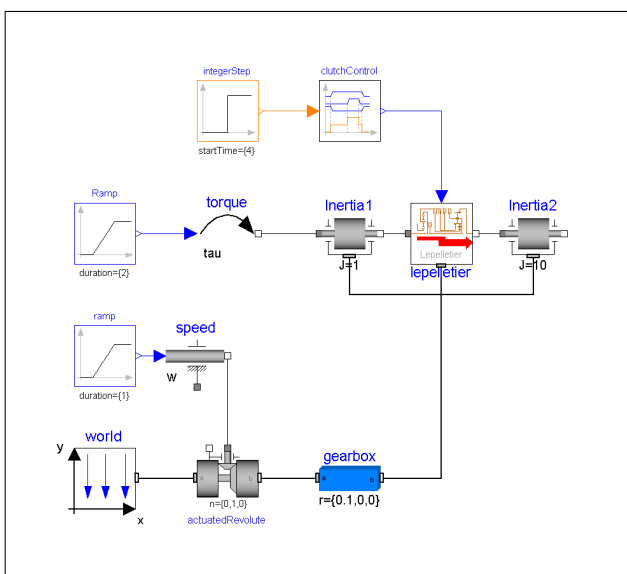Figure 17: Simulation results for automatic gearbox on yawing carrier
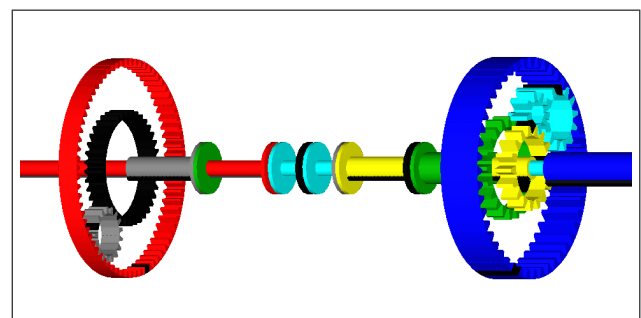

Figure 16: Test setup similar to yawing vehicle

Figure 18: Animation screenshot for Lepelletier wheelset

7   Conclusions and Outlook

All the details have been described how the one-dimensional rotational Modelica libraries have been extended in order that drive trains modelled with these libraries can be mounted on parts of the MultiBody library moving in three dimensions. All dynamic effects, such as support and gyroscopic torques, are taken into account.

This approach has several advantages:

- Only the mass, the center of mass and the inertia tensor of a complete powertrain, such as an automatic gearbox, has to be provided by the modeller together with the rotational inertias along the axes of rotations. For a convential modelling with multi-body components, data for mass, center of mass and inertia tensor is required from every single piece of a powertrain.

- The powertrain can be modelled and tested first as a pure one-dimensional system.

- Mounting a powertrain on a three-dimensionally moving system just requires to connect the three-dimensional Frame connectors of the powertrain components to appropriate mounting objects (Mounting1D) that are fixed on the multi-body parts.

- A hybrid 1D/3D model does not have problems with possible overconstraining that is a major problem for a 3D model of a powertrain.

Future work will include integration of powertrains into vehicle dynamics models [7] and examination of the interaction between vehicle and powertrain dynamics.

Acknowledgements

References

[1] German Aerospace Center (DLR), Oberpfaffenhofen, *PowerTrain Library 1.0 – Tutorial*, Dezember 2002. http://www.dynasim.se/www/PowerTrainTutorial.pdf.

[2] M. Otter, H. Elmqvist, and S. E. Mattsson, "The New Modelica MultiBody Library," in *Proceedings of the 3rd International Modelica Conference* (P. Fritzson, ed.), (Linköping), The Modelica Association and Linköping University, November 2003.

[3] J. Wittenburg, *Dynamics of Systems of Rigid Bodies*, vol. 33 of *Leitfäden der angewandten Mathematik und Mechanik*. B. G. Teubner Verlag, 1977.

[4] M. Otter, *Objektorientierte Modellierung mechatronischer Systeme am Beispiel geregelter Roboter*. Dissertation, Ruhr-Universität Bochum, 1994.

[5] J. Loomann, *Zahnradgetriebe*. Springer, 3rd ed., 1996.

[6] H. Dach, W.-D. Gruhle, and P. Köpf, *Pkw-Automatgetriebe*, vol. 88 of *Die Bibliothek der Technik*. Landsberg/Lech: Verlag Moderne Industrie, 2nd, revised ed., 2001.

[7] J. Andreasson, "VehicleDynamics Library," in *Proceedings of the 3rd International Modelica Conference* (P. Fritzson, ed.), (Linköping), The Modelica Association and Linköping University, November 2003.