Modelling and Simulation of Rigid and Flexible Multibody Systems in Modelica

Tutorial at the SIMS2007 , Göteborg (Särö), October 30th, 2007

Andreas Heckmann, German Aerospace Center (DLR)
Institute of Robotics and Mechatronics

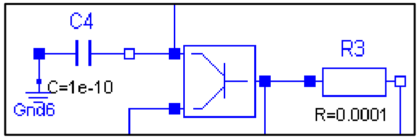
Deutsches Zentrum
DLR für Luft- und Raumfahrt e.V.
in der Helmholtz-Gemeinschaft

Contents

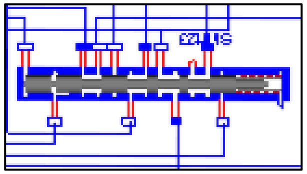
Modelica, very briefly (see Tutorial by Peter Fritzon for more)

- General goal:
  - modelling and simulation of **complex physical systems**, consisting of components from **different engineering domains**
- Focus:
  - **System Dynamics**: global behavior of complex systems
  - differential, algebraic, discrete equations (low-order PDE only if at all, i.e. no FEM or CFD)
- Platform for collaboration between engineering disciplines
  - all-in-one approach (contrary to e.g. co-simulation)
  - specialists contribute **libraries** (open/commercial) for general use
  - supplier provides executable specification of components for OEMs

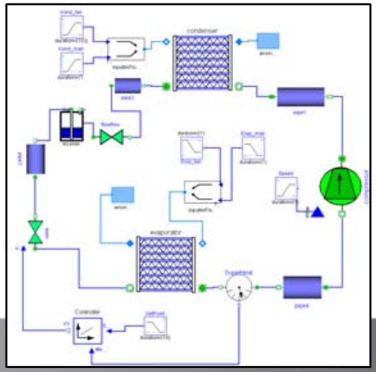
Modelica.Electrical.Analog



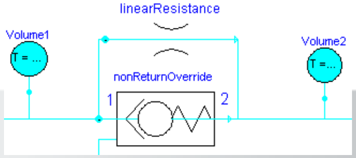
Modelica.Hydraulics



AirConditioning

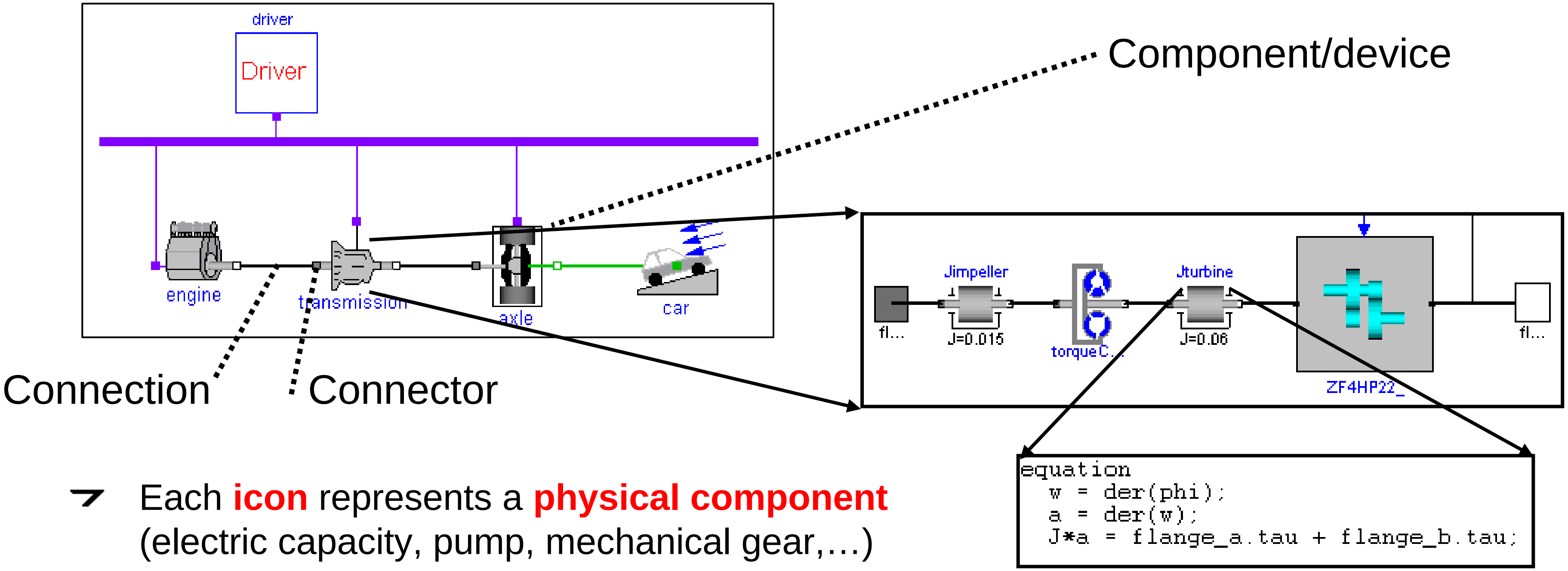


PneuLib



Modelica, very briefly (see Tutorial by Peter Fritzon for more)

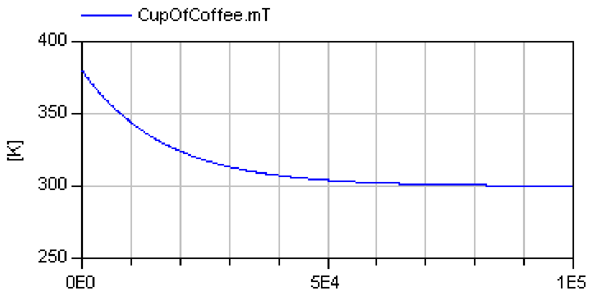
- Object orientation for complex model structuring
  - component-connections, hierarchies, inheritance

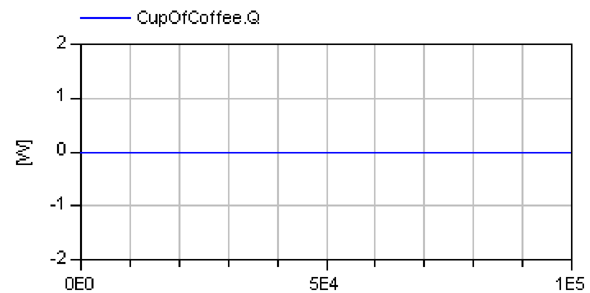


- Each **icon** represents a **physical component** (electric capacity, pump, mechanical gear,…)
- a **connection** line represents the actual physical **coupling** A component consists of (wire, fluid flow, heat flow,…)
- A **component** consist of **connected sub-components** (hierarchical structure) and/or is described by **equations**

Modelica, very briefly (see Tutorial by Peter Fritzon for more)

- Deklarative, acausal modelling
  - equations no assignments
  - mathematical description no algorithms

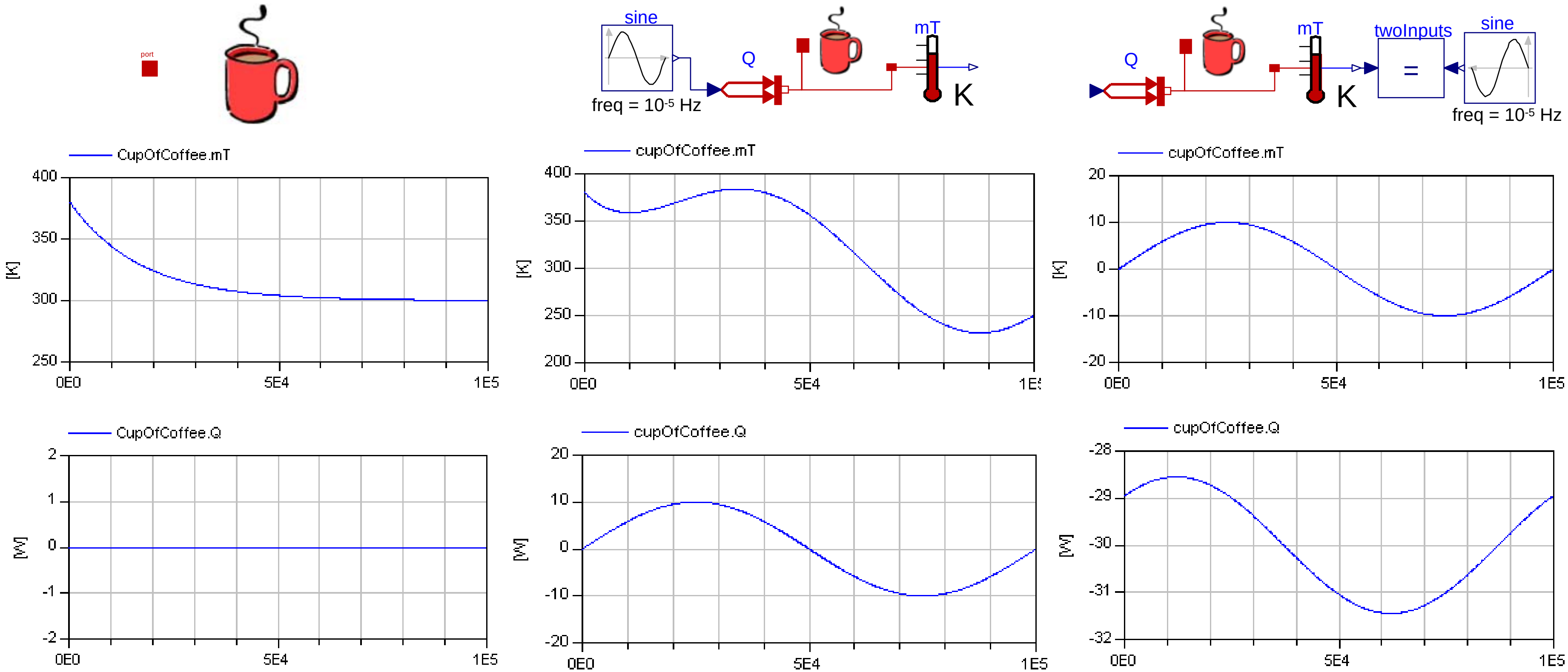





```
model CupOfCoffee
  import Modelica.SIunits.*;
  import Modelica.Thermal.HeatTransfer.Interfaces.*;

  parameter Temperature T0=380 "Initial temp.";
  parameter Temperature Tamb=300 "Ambient temperature";
  parameter Density rho=1000 "Coffee density";
  parameter SpecificHeatCapacity cv=4179 "Coffee specific heat";
  parameter CoefficientOfHeatTransfer h=25 "Convection coefficient";
  parameter Volume V=4e-4 "Volume of coffee";
  parameter Area A=4e-3 "Area of coffee";

  Temperature T(start=T0) "Coffee Temperature";
  HeatPort_a port "interface to external components" a;
  a;
equation
 0 = rho*V*cv*der(T) + h*A*(T - Tamb) - port.Q_flow "First law of thermodynamics";
 port.T = T;
end CupOfCoffee;
```

Modelica, very briefly (see Tutorial by Peter Fritzon for more)

- Deklarative, acausal modelling
  - equations no assignments
  - mathematical description no algorithms



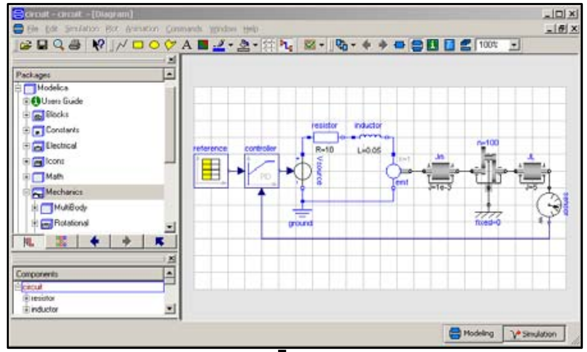
Modelica, very briefly (see Tutorial by Peter Fritzon for more)

- Deklarative, acausal modelling
  - equations no assignments
  - mathematical description no algorithms
    - strictly speaking: restricted use of algorithms
  - relies on symbolic maniplations
    - on system level
    - determine signal path at compile time
    - results in state space form and explicit ode

$$\dot{x} = f(x, t, \ldots)$$

Modelica and Simulation Environments

Graphical editor
for Modelica models



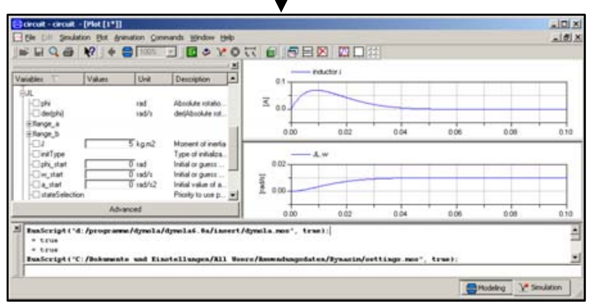
Modelica simulation environment (Dymola, MathModelica, SimulationX, Mosilab, ...)

Textual description
on file (equations, "schematic", animation)

```
model circuit
  a;
  Modelica.Electrical.Analog.Basic.Resistor resistor(R=10) a;
  Modelica.Electrical.Analog.Basic.Inductor inductor(L=0.05) a;
  Modelica.Electrical.Analog.Basic.EMF emf(k=1) a;
  Modelica.Electrical.Analog.Sources.SignalVoltage Vsource a;
  Modelica.Electrical.Analog.Basic.Ground ground a;
  Modelica.Blocks.Continuous.LimPID controller a;
  Modelica.Mechanics.Rotational.Inertia Jm(J=1e-3) a;
  Modelica.Mechanics.Rotational.IdealGear n(ratio=100) a;
  Modelica.Mechanics.Rotational.Fixed fixed a;
  Modelica.Mechanics.Rotational.Inertia JL(J=5) a;
  Modelica.Mechanics.Rotational.Sensors.SpeedSensor sensor a;
  Modelica.Blocks.Sources.CombiTimeTable reference a;
```

Free Modelica language

Translation of Modelica models in C-Code,
Simulation, and
interactive **scripting**
(plot, freq. resp., ...)

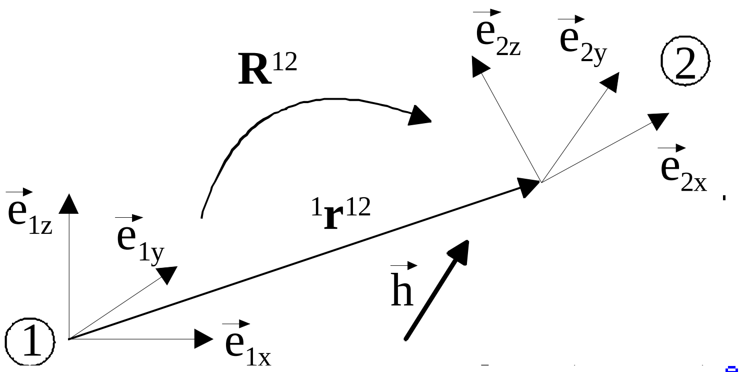


Modelica Simulation-environment (Dymola, MathModelica, SimulationX, Mosilab,…)

Contents

Modelica Multibody Basics: Orientation

- Coordinate systems and their orientation

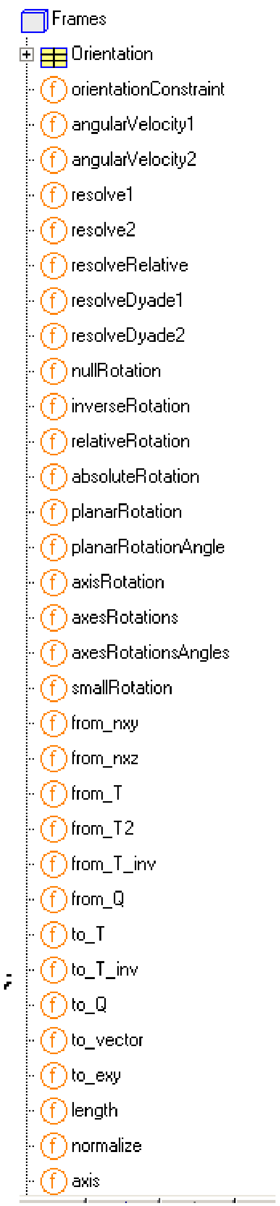



```
import MultiBody.Frames;
 Frames.Orientation R12;
 Real h1[3] "h resolved in frame 1";
 Real h2[3] "h resolved in frame 2";
equation
 h2 = Frames.resolve2(R12, h1); //or
 h1 = Frames.resolve1(R12, h2);
```

- Orientation object R^{12}
  - describes orientation of coordinate system 2 wrt.1
  - holds

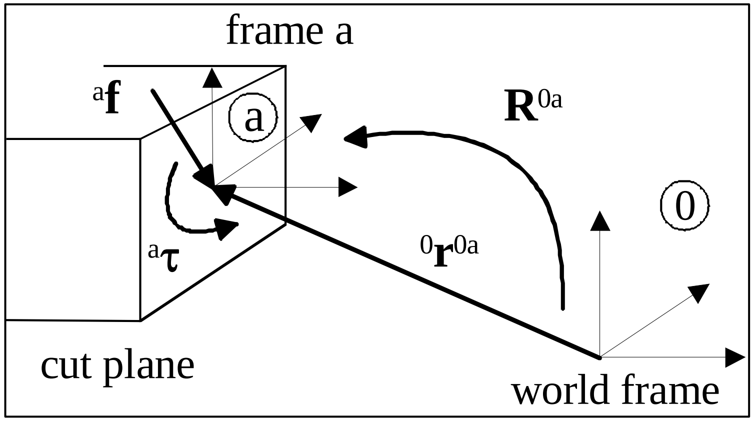
```
Real T[3, 3] "Transformation matrix from world frame to local frame";
SI.AngularVelocity w[3]
  "Absolute angular velocity of local frame, resolved in local frame";
```

  - may be computed using rotation angles or quaternions
- Multibody Lib. contains over 30 functions to operate on orientation objects



Modelica Multibody Basics: Connectors I

- Connectors: the interface to connect components
  - Position is resolved in world frame
  - Forces and torques are resolved in local frame



non-flow !

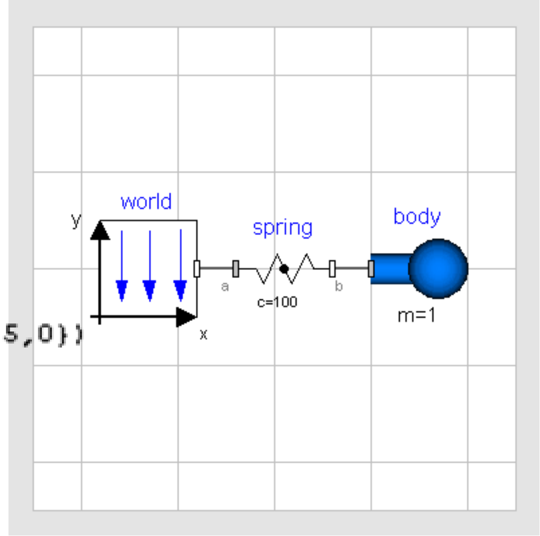
```
connector Frame
  "Coordinate system fixed to the component with one cut-force and cut-torque (no icon)"
  import SI = Modelica.SIunits;
  SI.Position r_0[3]
    "Position vector from world frame to the connector frame origin, resolved in world frame";
  Frames.Orientation R
    "Orientation object to rotate the world frame into the connector frame";
  flow SI.Force f[3] "Cut-force resolved in connector frame" a;
  flow SI.Torque t[3] "Cut-torque resolved in connector frame";
  a;
end Frame;
```

flow !

Modelica Multibody Basics: Connectors II

- Connectors: how they work

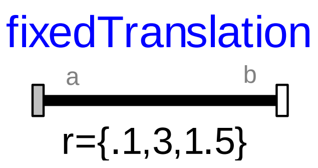
```
model SpringMass
  inner Modelica.Mechanics.MultiBody.World world
    a;
  Modelica.Mechanics.MultiBody.Forces.Spring spring(c=100)
    a;
  a;
  Modelica.Mechanics.MultiBody.Parts.Body body(r_0_start={0,.5,0})
    a;
equation
  connect(world.frame_b, spring.frame_a) a;
  connect(spring.frame_b, body.frame_a) a;
end SpringMass;
```



- Modelica‘s general connections rules
  - non-flow variables are set to be equal, i.e. frames coincide
    - since they represent „some kind of potential“
  - flow variables sum to zero (Kirchhoff‘s current law)
    - since they represent time derivatives of preserved quantities
    - are consequently set to zero if connector is not connected to anything
- see Modelica.UsersGuide.Connectors for a comparison of connectors in various domains

Modelica Multibody Basics: Components I

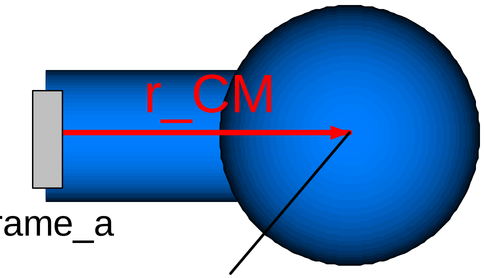
- Kinematics:
  - Component equations provide relations between connector variables on position level
  - MultiBody.Parts.FixedTranslation
    i.e. fixed translation of frame_b with respect to frame_a
  - Tool (e.g. Dymola) differentiates these equations twice for dynamics




```
frame_b.r_0 = frame_a.r_0 + Frames.resolve1(frame_a.R, r);
frame_b.R = frame_a.R;

/* Force and torque balance */
zeros(3) = frame_a.f + frame_b.f;
zeros(3) = frame_a.t + frame_b.t + cross(r, frame_b.f);
```

Modelica Multibody Basics: Components II

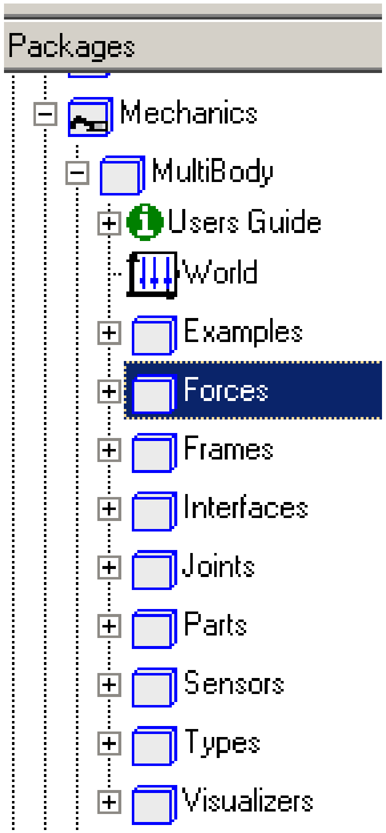
- Dynamics
  - Newton-Euler equations
  - MultiBody.Parts.Body



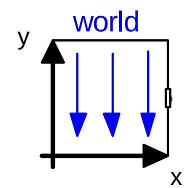

```
// import Modelica.Mechanics.MultiBody.Frames;
// translational kinematic differential equations resolved in local frame_a
v_a = Frames.resolve2(frame_a.R, der(frame_a.r_0));
a_a = der(v_a);

// rotational kinematic differential equations
w_a = Modelica.Mechanics.MultiBody.Frames.angularVelocity2(frame_a.R);
z_a = der(w_a);

// Newton/Euler equations with respect to center of mass
          a_CM = a_a + cross(z_a, r_CM) + cross(w_a, cross(w_a, r_CM));
          f_CM = m*a_CM;
          t_CM = I*z_a + cross(w_a, I*w_a);
     frame_a.f = f_CM;
     frame_a.t = t_CM + cross(r_CM, f_CM);
```

state selection

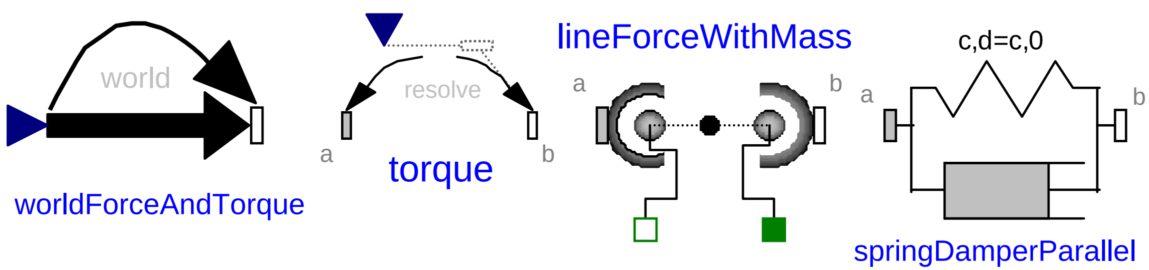
Modelica Multibody Basics: Elementary Components I



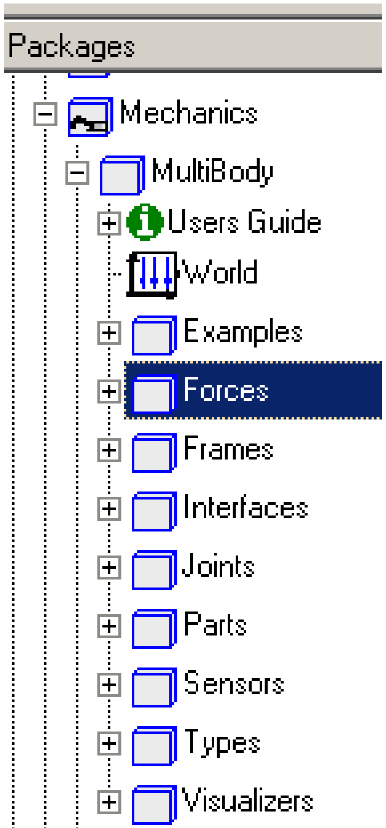
- Modelica.Mechanics.MultiBody.World
  - defines inertial frame, gravity, animation defaults



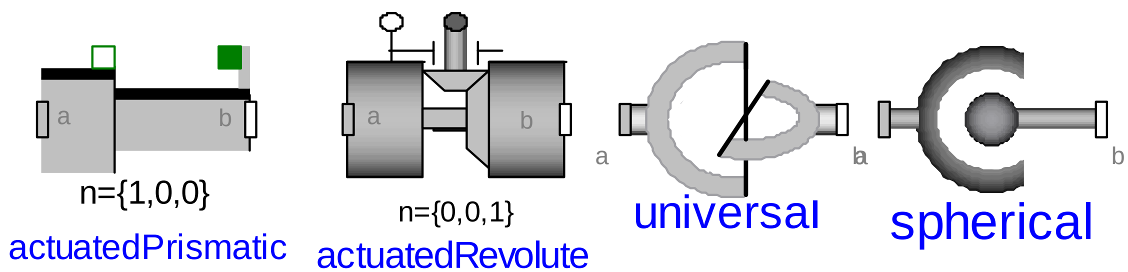
- Modelica.Mechanics.MultiBody.Forces
  - different resolution properties
  - interface to Real input functions and 1D mechanics
  - several spring/damper configurations



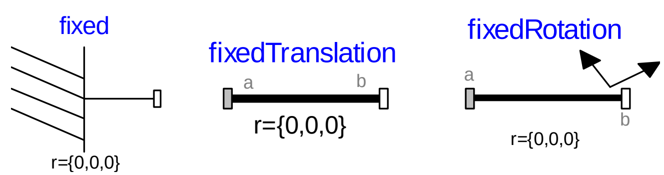
Modelica Multibody Basics: Elementary Components II



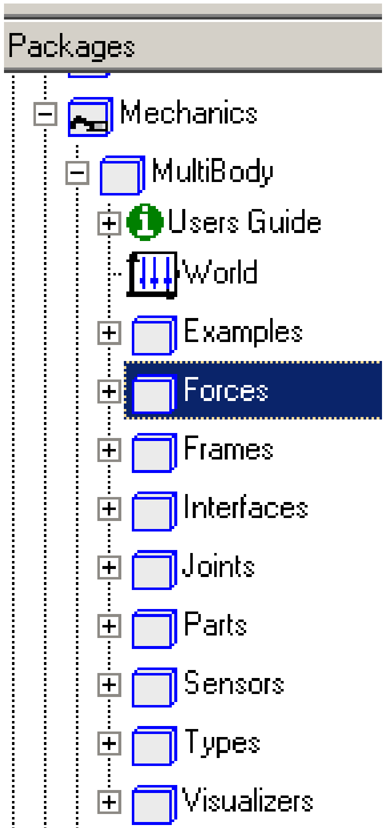
- Modelica.Mechanics.MultiBody.Joints
  - define specific degree of freedom
  - capability to set-up initial configuration
  - interface to/for 1D mechanics and rheonom motion
  - e.g.:



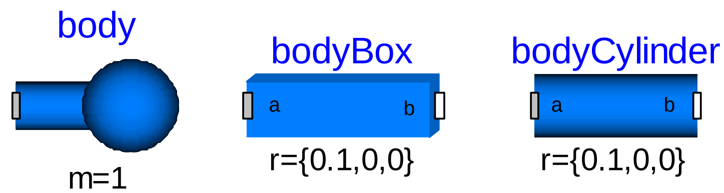
- Modelica.Mechanics.MultiBody.Parts
  - Fixed, FixedTranslation and FixedRotation



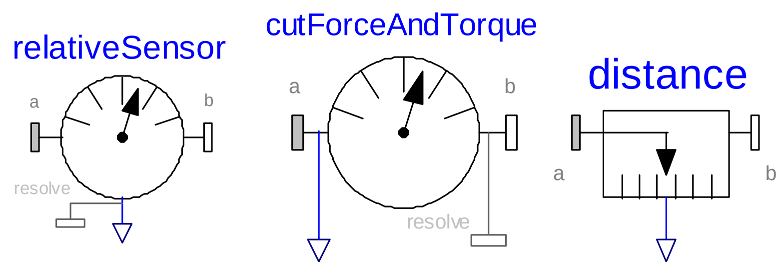
Modelica Multibody Basics: Elementary Components III



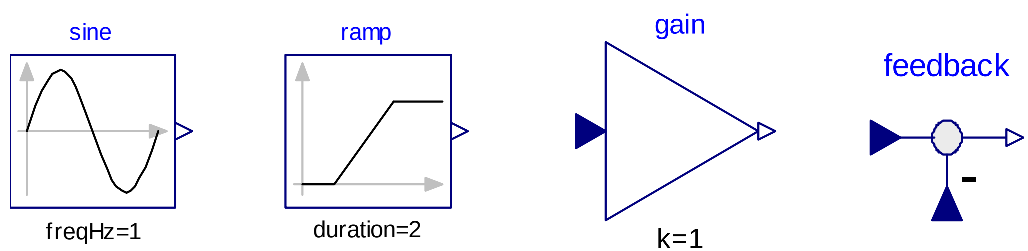
- Modelica.Mechanics.MultiBody.Parts
  - Rigid bodies with predefined geometric shapes



  - Modelica.Mechanics.Multibody.Sensors
    - for control and validation purposes

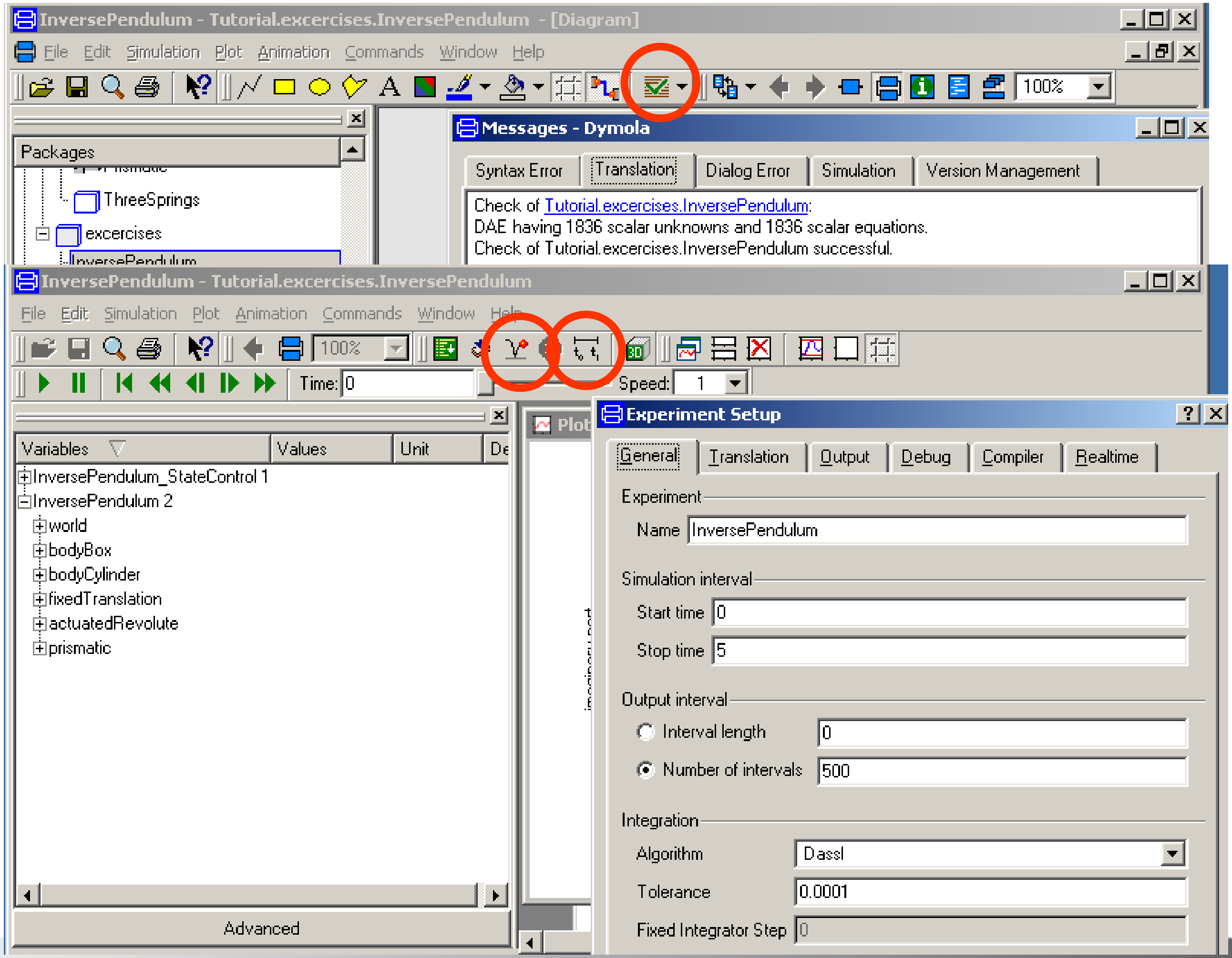


- Modelica.Blocks.Sources + Modelica.Blocks.Math



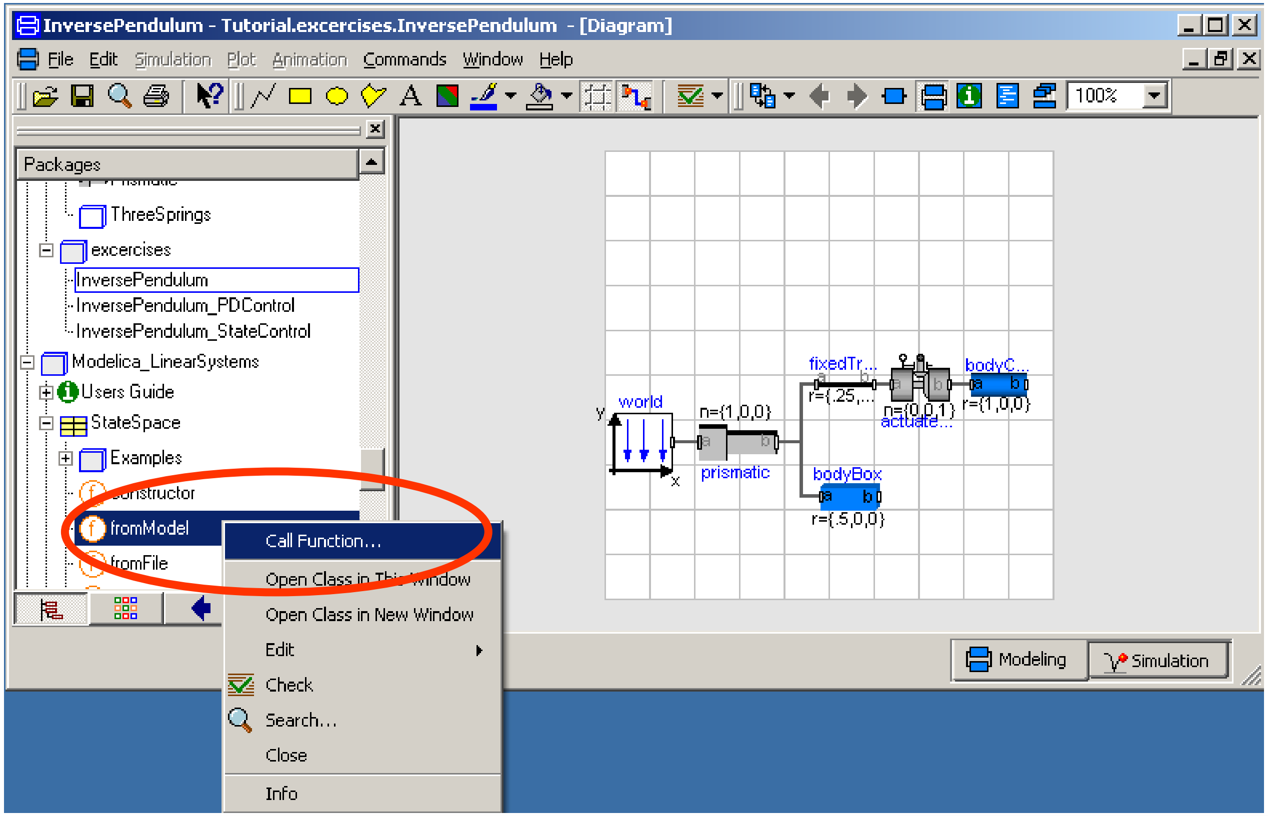
Modelica Multibody Basics: Analysis Methods

- Model check
- Experiment setup, translation and time simulation



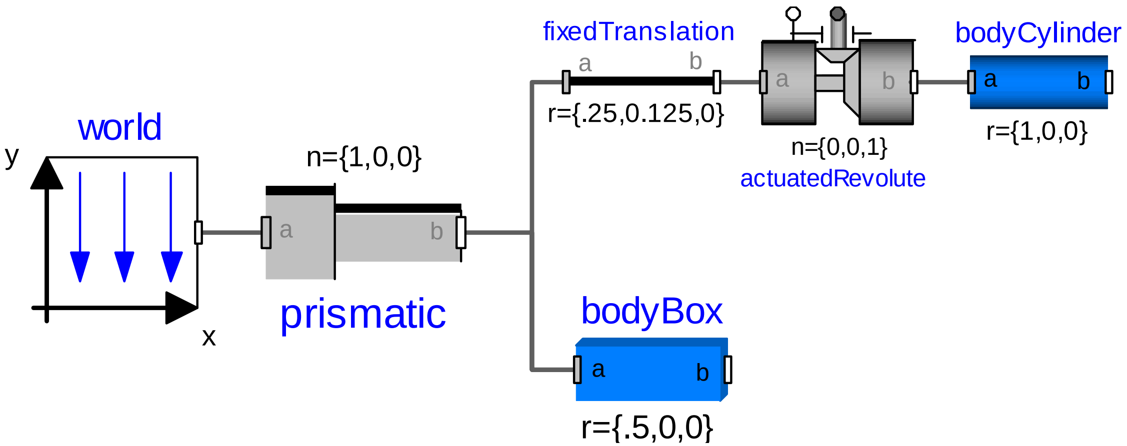
Modelica Multibody Basics: Analysis Methods

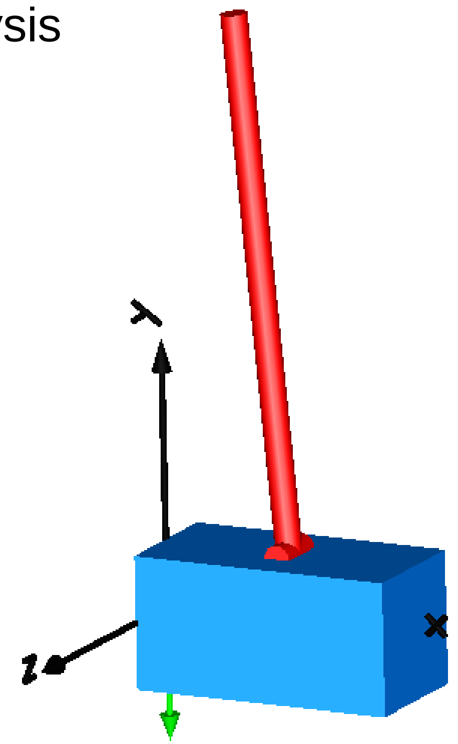
- Model check
- Experiment setup, translation and time simulation
- Eigenvalue analysis
  - Menu: File→Libraries→LinearSystems



Example 1: Control of an inverse pendulum I

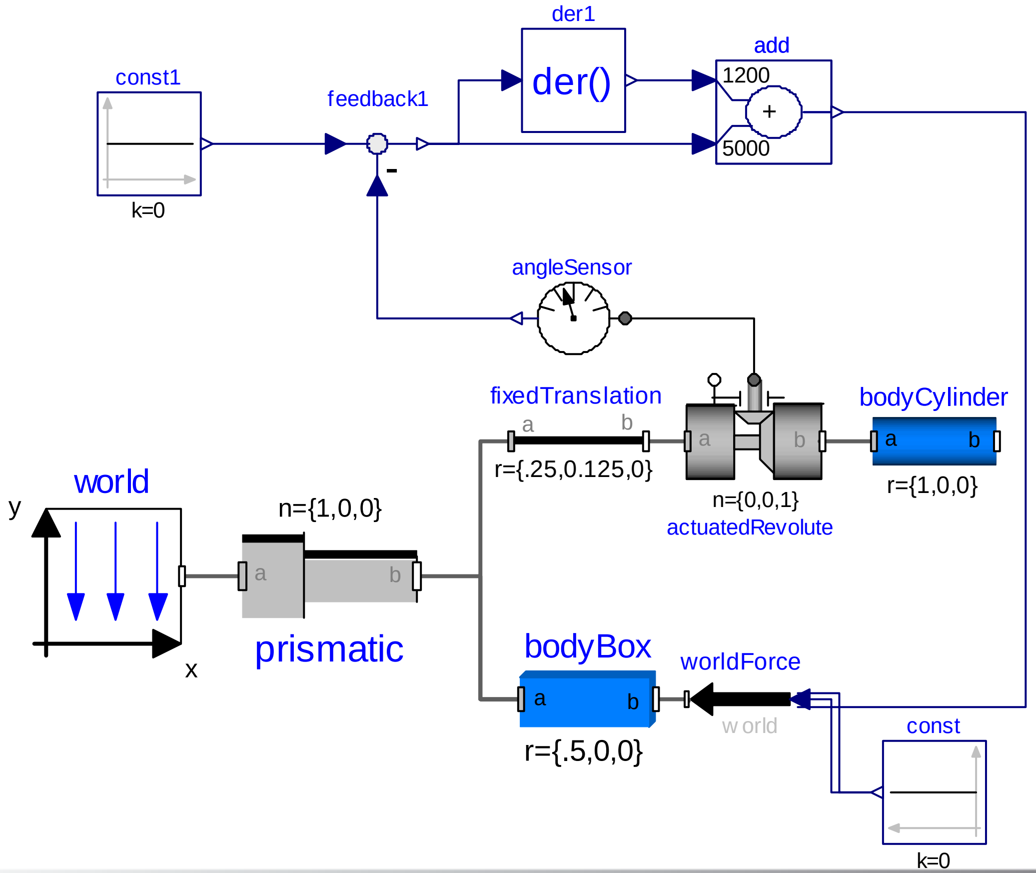
- Initial model
  - Box: 0.5 x 0.25 x 0.25 m
  - actuatedRevolute: 90° phi_offset, 5° phi_start
  - perform time simulation and eigenvalue analysis





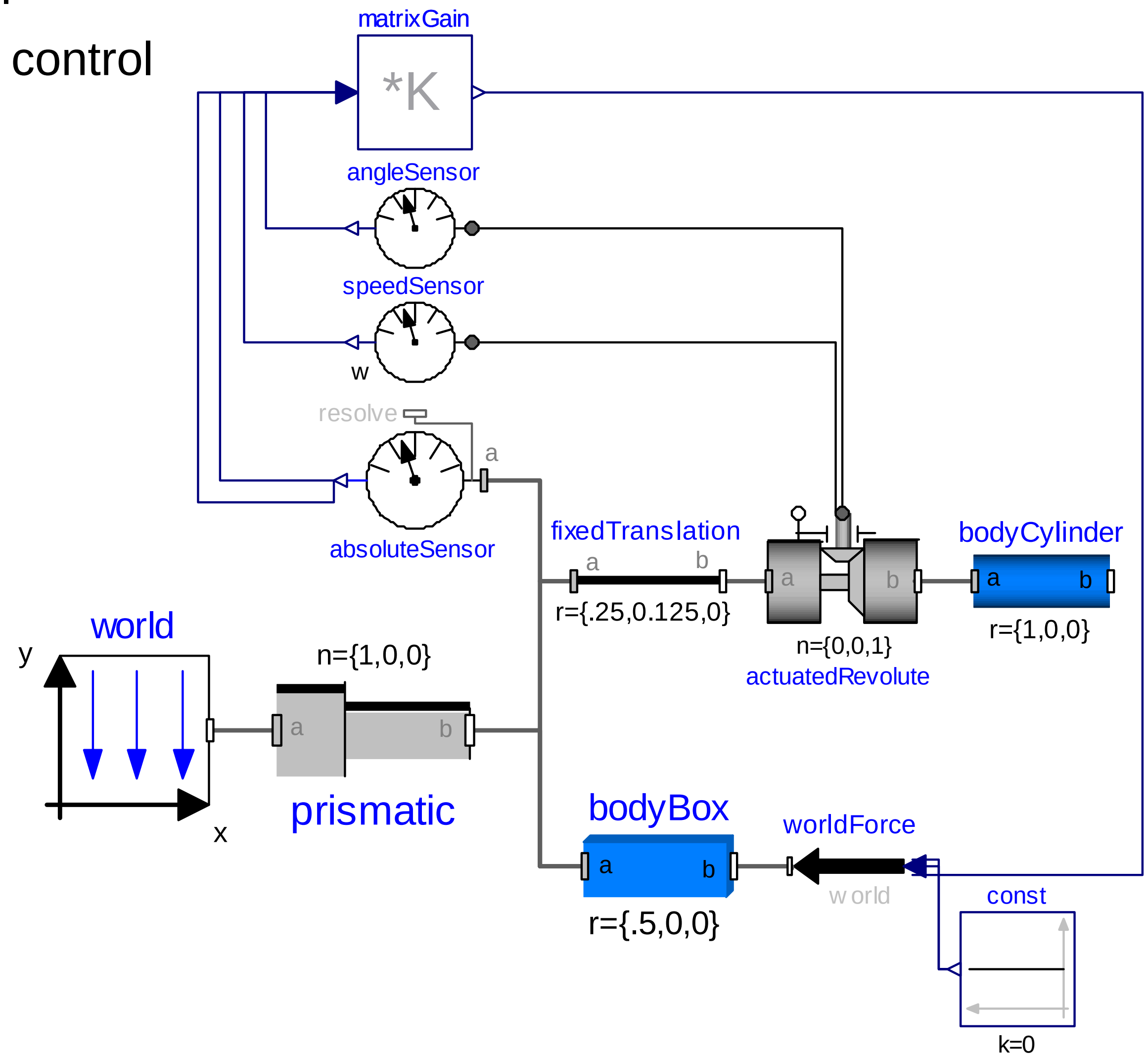
Example 1: Control of an inverse pendulum II

- PD-Control of angle only
  - use extend to inherit initial model
  - use transient response and pole placement to set-up contoller gains



Example 1: Control of an inverse pendulum III

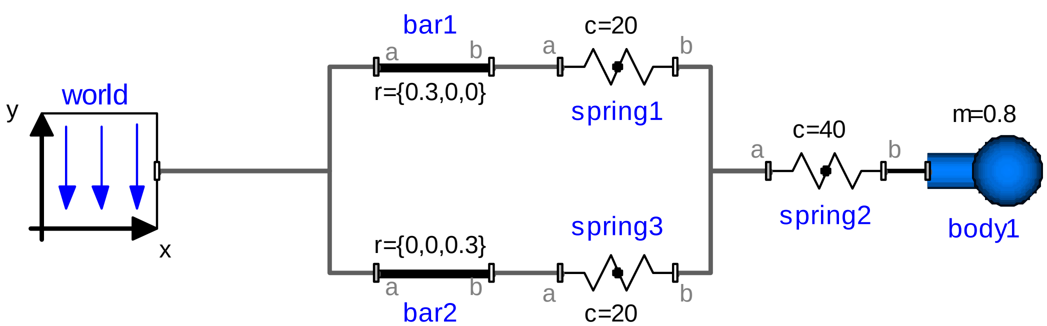
- Improve control
  - e.g. state control

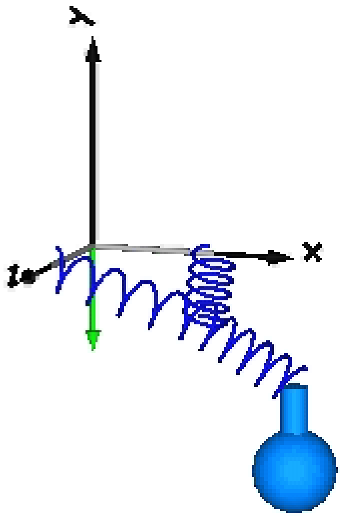


Contents

Modelica Multibody Advanced: State selection I

- Joints AND bodies have potential states
  - number of joints is independent from number of bodies
  - an assignment of joints to bodies is not mandatory
  - e.g.:



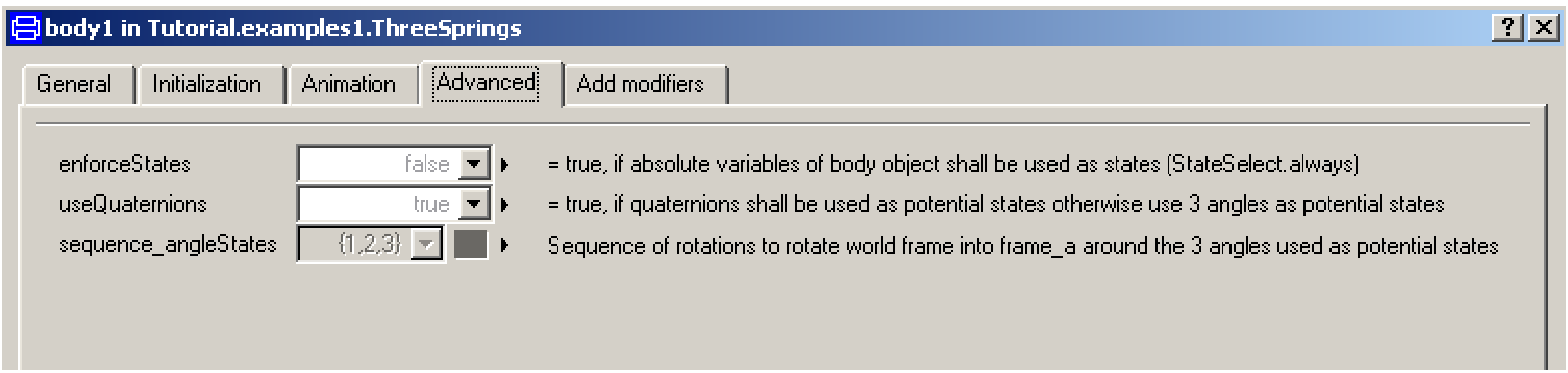


  - here: body coordinates: position, quaternions and their derivatives are used as states

Modelica Multibody Advanced: State selection II

- relative joint coordinates are used as states if possible
  - default: stateSelect = StateSelect.prefer
  - e.g. Multibody.Joints.Prismatic

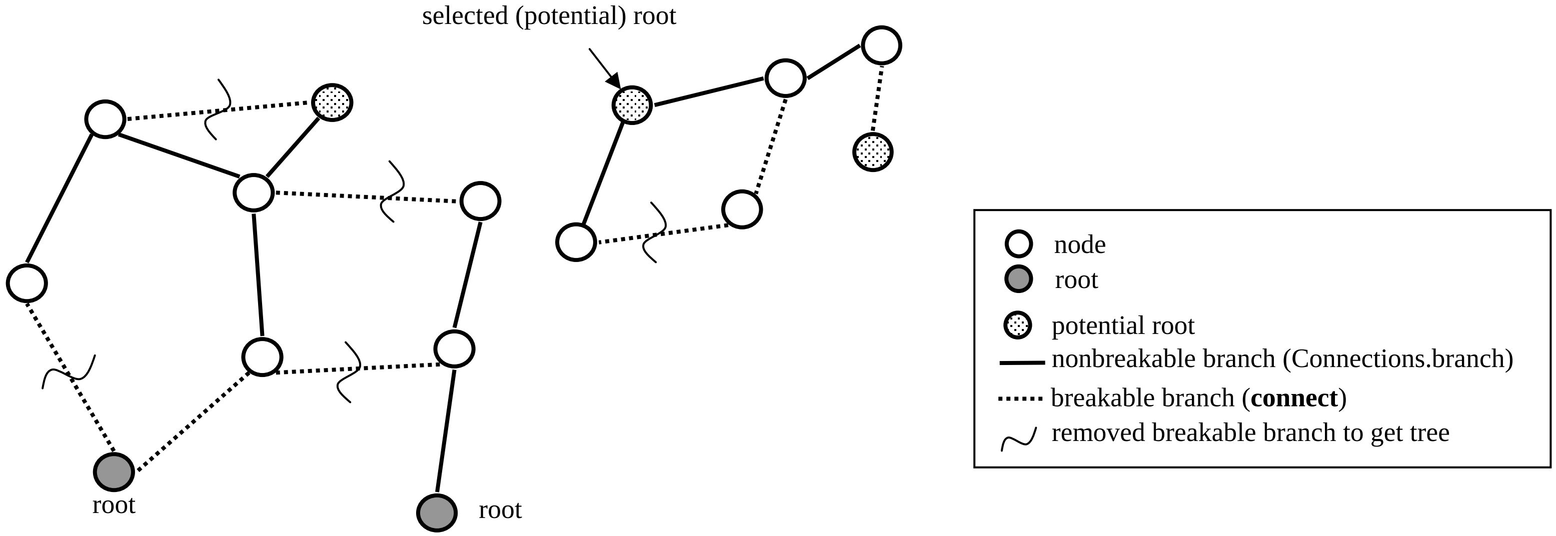

```
final parameter Real e[3]=Modelica.Mechanics.MultiBody.Frames.normalize(n)
  "Unit vector in direction of prismatic axis n";
SI.Position s(stateSelect=if enforceStates then
      StateSelect.always else StateSelect.prefer)
  "Relative distance between frame_a and frame_b";
SI.Velocity v(stateSelect=if enforceStates then StateSelect.always else
      StateSelect.prefer) "First derivative of s (relative velocity)";
```



  - Advanced user may influence state selection directly

Modelica Multibody Advanced: Loops I

- Standard case
  - no specific action by the user is required
  - every connector is one node in the virtual connection graph
  - roots of the virtual connection graph are found, e.g. world.frame_b
  - loops are virtually broken



Modelica Multibody Advanced: Loops I

- Standard case
  - no specific action by the user is required
  - every connector is one node in the virtual connection graph
  - roots of the virtual connection graph are found, e.g. world.frame_b
  - loops are virtually broken
  - the related constraint equations are provided
    ⇒ DAE

$$0 = \boldsymbol{f}(\dot{\boldsymbol{x}}, \boldsymbol{x}, \boldsymbol{y}, t, \ldots) \quad \dim(\boldsymbol{f}) = \dim(\boldsymbol{x}) + \dim(\boldsymbol{y})$$

  - Equations are rearranged to get a sequence for model evaluation
    (**B**lock **L**ower **T**riangle-partitioning)

$$\begin{array}{c} \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{array} \begin{array}{c} \begin{array}{ccccc} z_1 & z_2 & z_3 & z_4 & z_5 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix} \end{array} \Rightarrow \begin{array}{c} \\ f_2 \\ f_4 \\ f_3 \\ f_5 \\ f_1 \end{array} \begin{array}{c} \begin{array}{ccccc} z_1 & z_2 & z_3 & z_4 & z_5 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

Modelica Multibody Advanced: Loops I

- Standard case
  - no specific action by the user is required
  - every connector is one node in the virtual connection graph
  - roots of the virtual connection graph are found, e.g. world.frame_b
  - loops are virtually broken
  - the related constraint equations are provided
    $\Rightarrow$ DAE

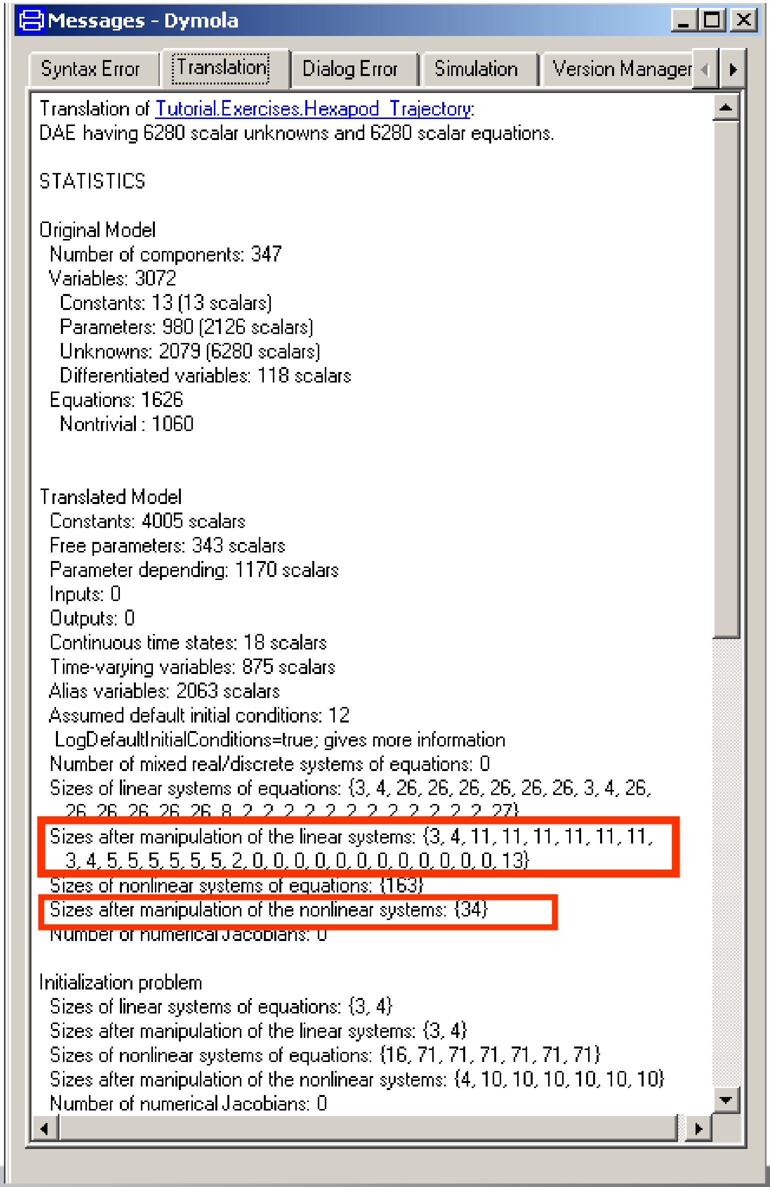
$$0 = \boldsymbol{f}(\dot{\boldsymbol{x}}, \boldsymbol{x}, \boldsymbol{y}, t, \ldots) \quad \mathrm{dim}(\boldsymbol{f}) = \mathrm{dim}(\boldsymbol{x}) + \mathrm{dim}(\boldsymbol{y})$$

  - Equations are rearranged to get a sequence for model evaluation (**B**lock **L**ower **T**riangle-partitioning)
  - Equations to be differentiated are determined (Pantelides algorithm)
  - superflous potential states are deselected dynamically (dummy derivative method) $\Rightarrow$ ODE:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, t, \ldots)$$

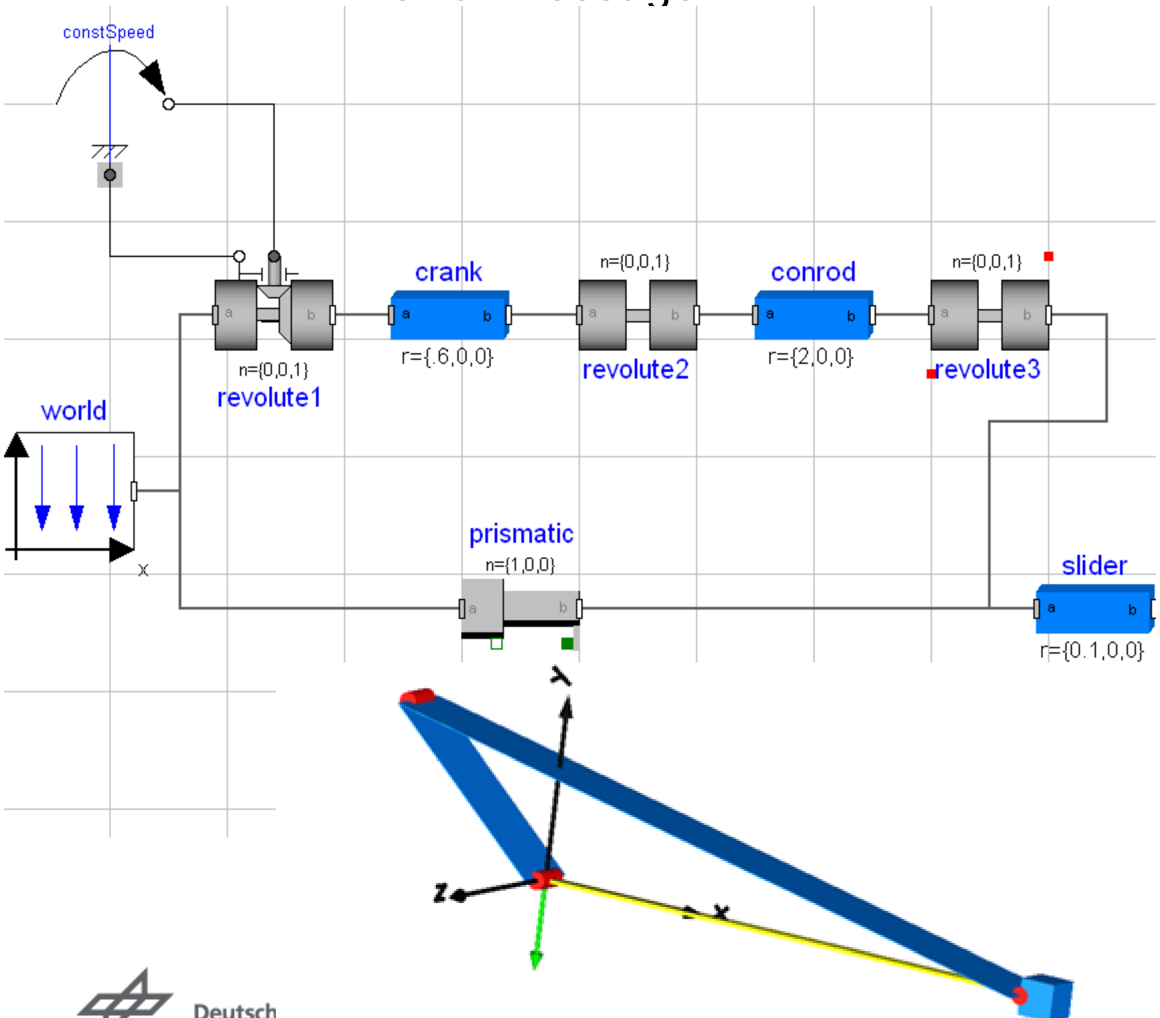
Modelica Multibody Advanced: Loops II

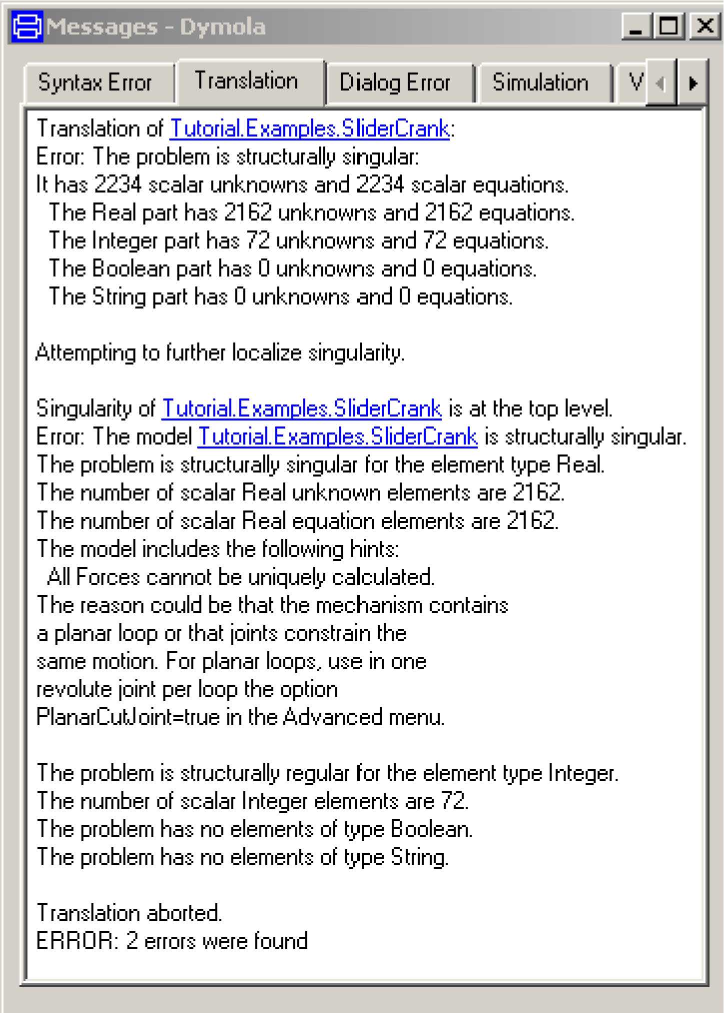
- review Translation Log in order to streamline simulation performance with model adjustments



Modelica Multibody Advanced: Loops III

- Planar loops
  - error message




Messages - Dymola

Syntax Error | Translation | Dialog Error | Simulation

Translation of Tutorial.Examples.SliderCrank:
Error: The problem is structurally singular.
It has 2234 scalar unknowns and 2234 scalar equations.
The Real part has 2162 unknowns and 2162 equations.
The Integer part has 72 unknowns and 72 equations.
The Boolean part has 0 unknowns and 0 equations.
The String part has 0 unknowns and 0 equations.

Attempting to further localize singularity.

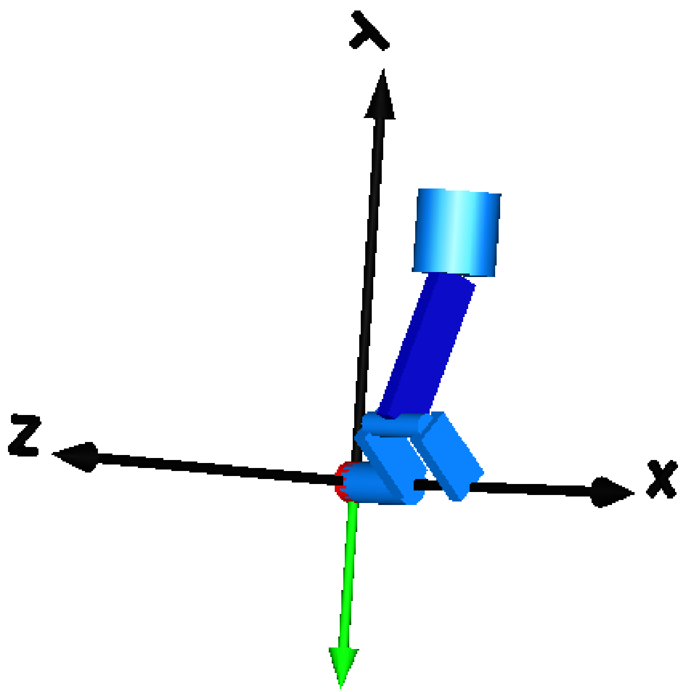
Singularity of Tutorial.Examples.SliderCrank is at the top level.
Error: The model Tutorial.Examples.SliderCrank is structurally singular.
The problem is structurally singular for the element type Real.
The number of scalar Real unknown elements are 2162.
The number of scalar Real equation elements are 2162.
The model includes the following hints:
All Forces cannot be uniquely calculated.
The reason could be that the mechanism contains
a planar loop or that joints constrain the
same motion. For planar loops, use in one
revolute joint per loop the option
PlanarCutJoint=true in the Advanced menu.

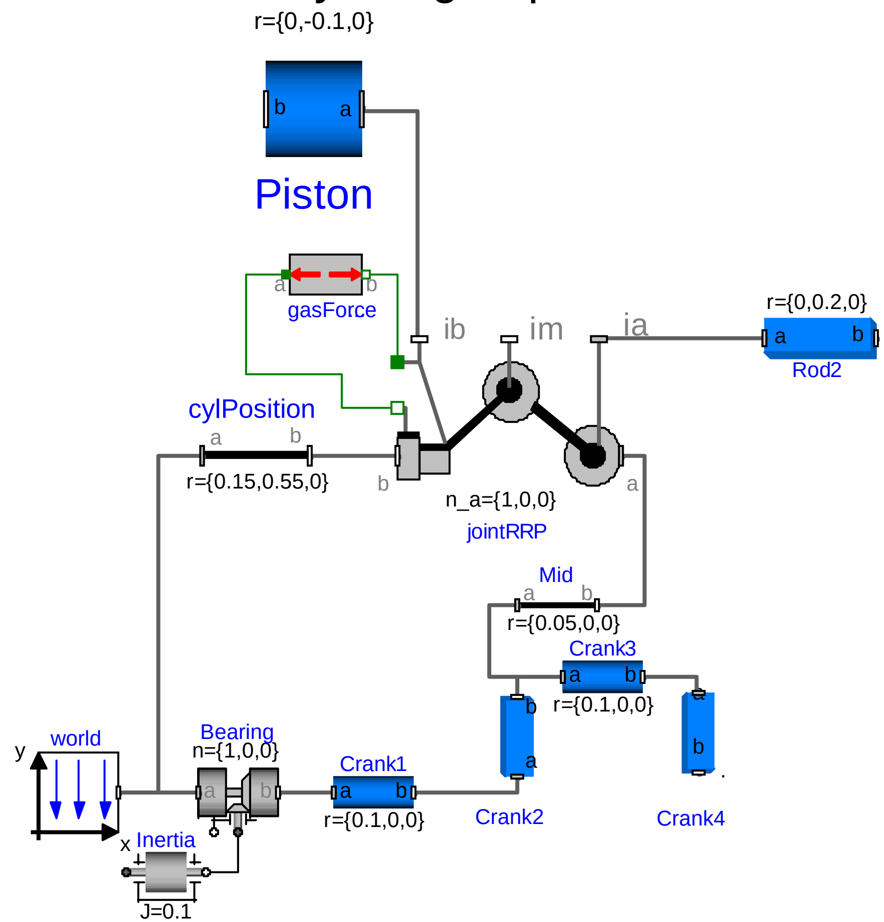
The problem is structurally regular for the element type Integer.
The number of scalar Integer elements are 72.
The problem has no elements of type Boolean.
The problem has no elements of type String.

Translation aborted.
ERROR: 2 errors were found

Modelica Multibody Advanced: Loops IV

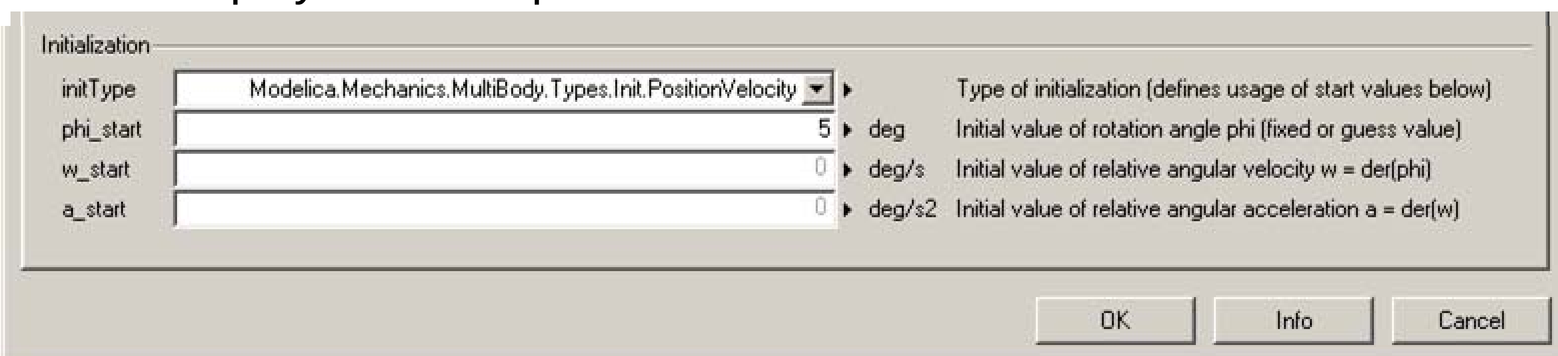
- Use of aggregrated joint objects
  - to profit from analytical loop handling according to the „characteristic pair of joints“ method by the group of Prof. Hiller





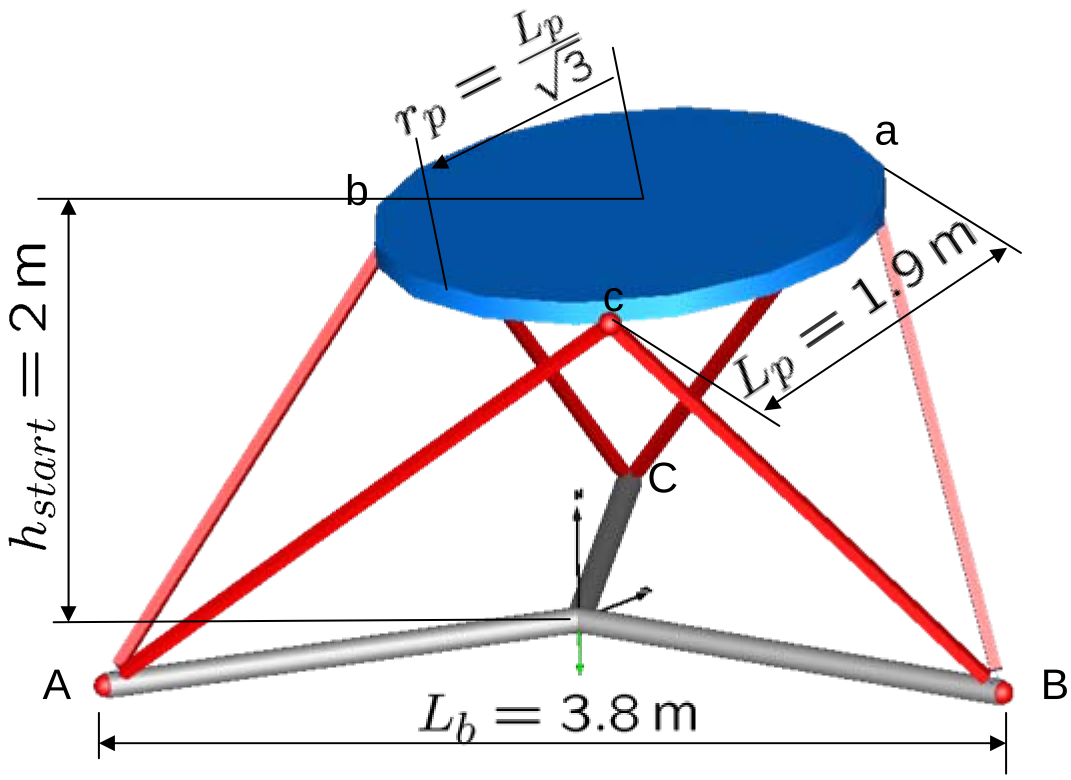
Modelica Multibody Advanced: Initialisation

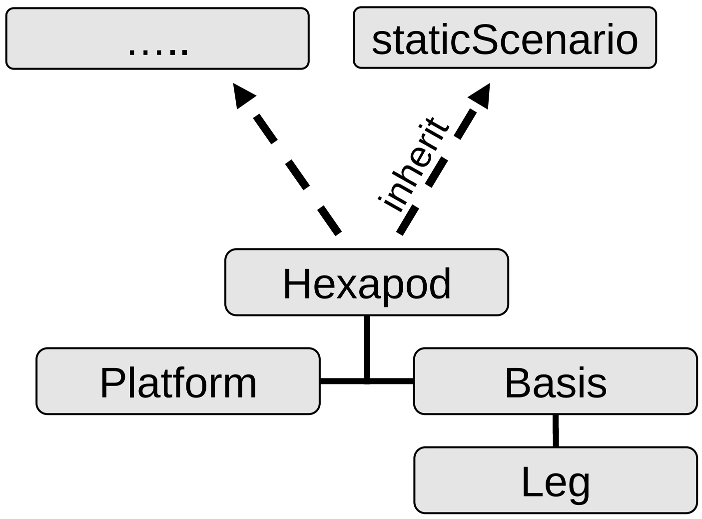
- Initialisation default:
  - every state is assumed to be arbitrary unless otherwise provided
  - Newton solver starts with guess value zero in order to find consistent initial states unless otherwise provided
- If initialisation fails
  - determine, i.e. fix, characteristic variables/states in order to influence the system of equations to solve
  - provide „good“ guesses for initial states
  - be aware of singular positions, e.g. piston at bottom dead center
  - keep system of equations consistent



Exercise 2: Hexapod I

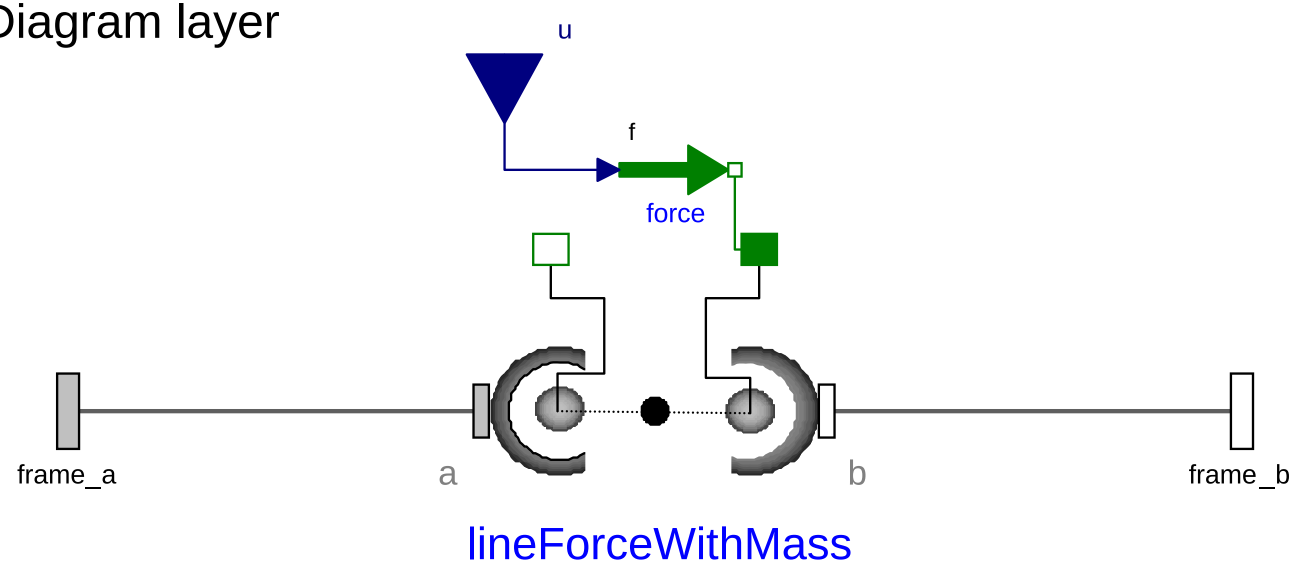
- The modelling concept



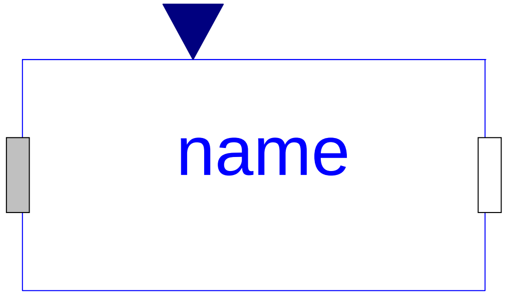


Exercise 2: Hexapod II

- Model Leg
  - Diagram layer



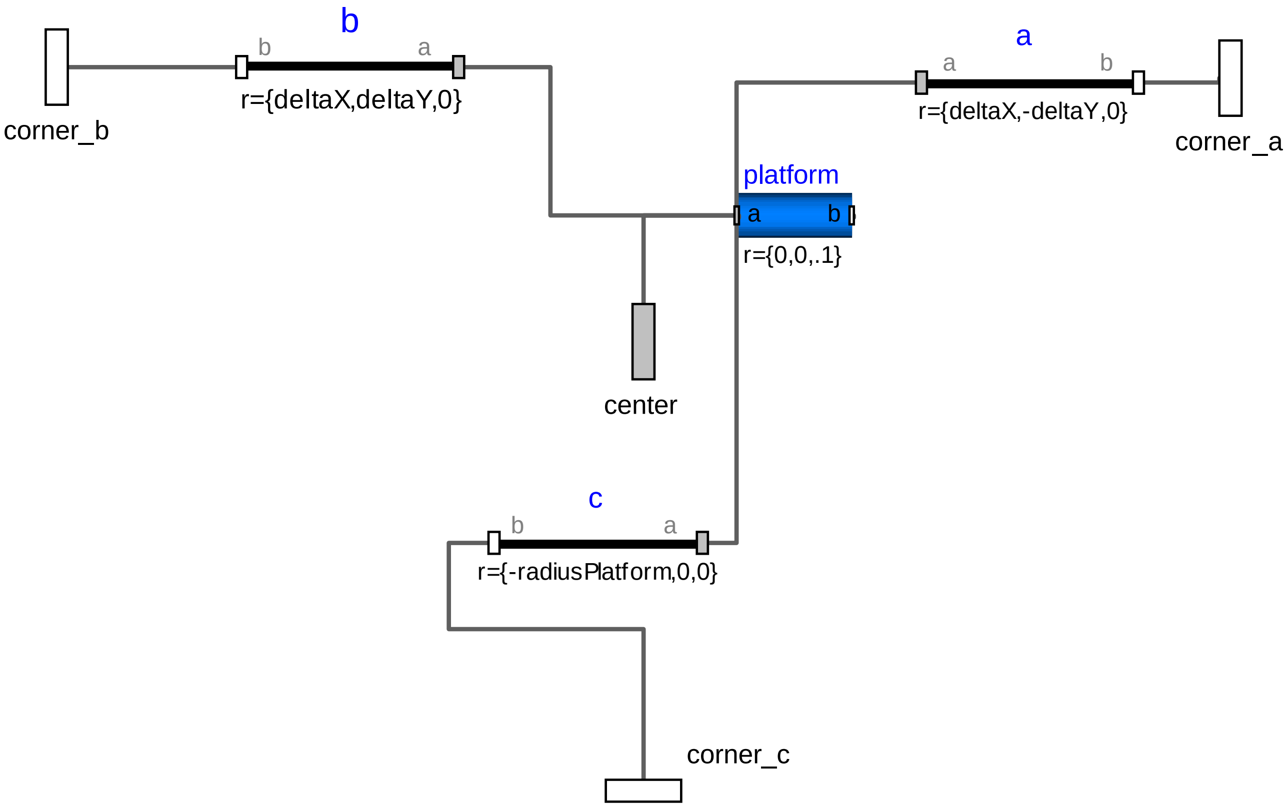
  - Icon layer



Exercise 2: Hexapod III

- Model Platform

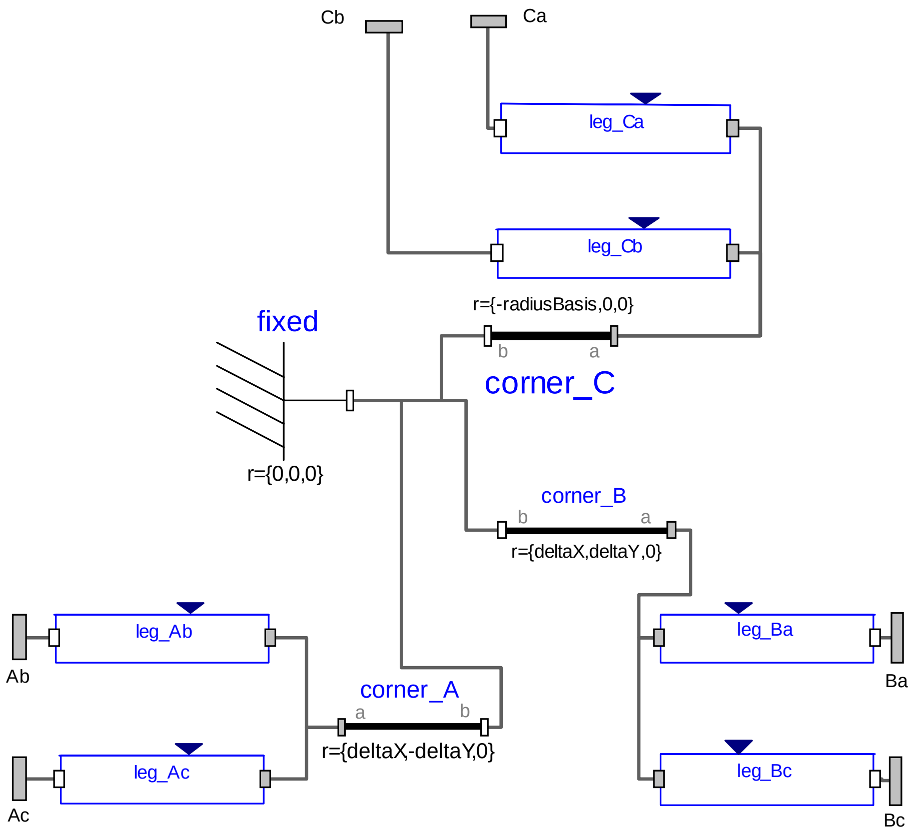
```
model Platform
  import SI = Modelica.SIunits;
  parameter SI.Length Lp=1.9 " length of a side of the platform";
  final parameter SI.Length radiusPlatform=Lp/sqrt(3);
  final parameter SI.Length deltaX=radiusPlatform/2;
  final parameter SI.Length deltaY=Lp/2;
```



Exercise 2: Hexapod IV

- Model Basis

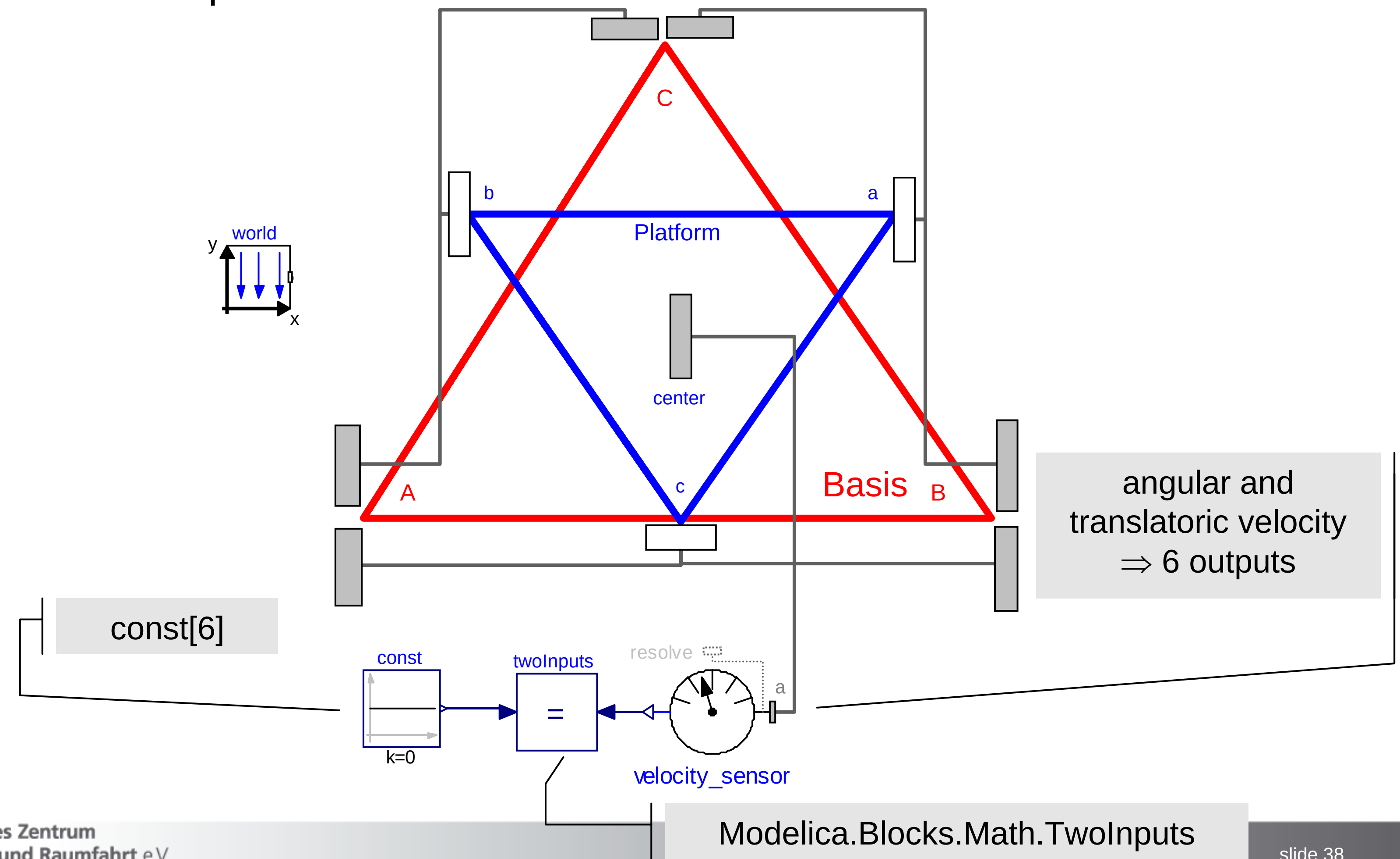
```
model Basis
  import SI = Modelica.SIunits;
parameter SI.Length Lb=3.8 " length of a side of the basis";
final parameter SI.Length radiusBasis=Lb/sqrt(3);
final parameter SI.Length deltaX=radiusBasis/2;
final parameter SI.Length deltaY=Lb/2;
```



Exercise 2: Hexapod V

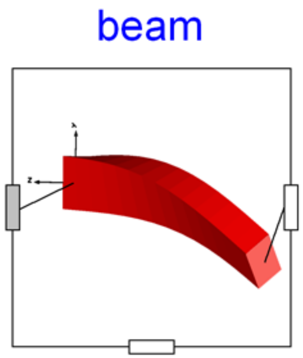
- Hexapod_StaticScenario
  - actuator force in initial position ?

```
partial model Hexapod
  import SI = Modelica.SIunits;
  parameter SI.Length h_start=2;
```



Contents

FlexibleBodies Library: Beams versus ModalBody



Common Issues

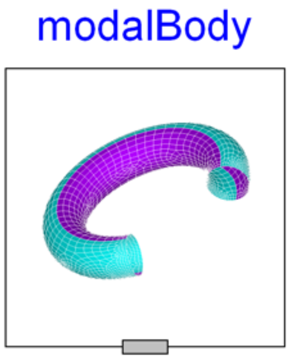
Floating frame of reference

Equations of motion

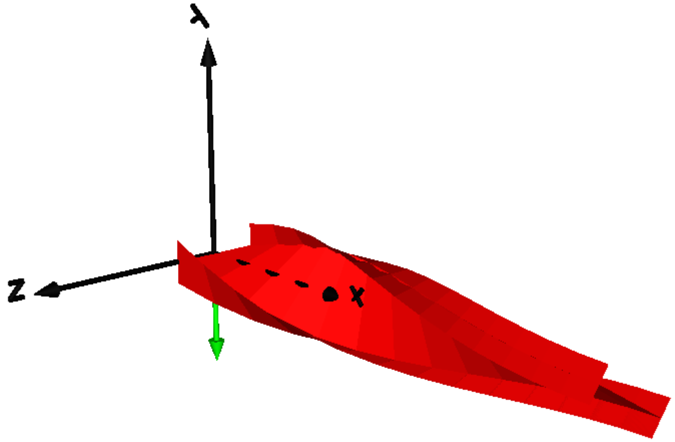
SID-data-structure

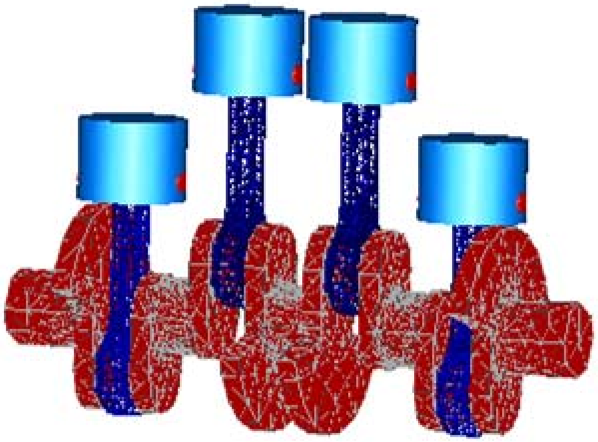
Standard-Input-Data: Wallrapp '94

Disjunctive Issues



Beam	ModalBody
Analytical beam description	**FEM-based body description**
Modelica generated SID	**External generated SID**
Animation uses beam description	**Animation based on external data**

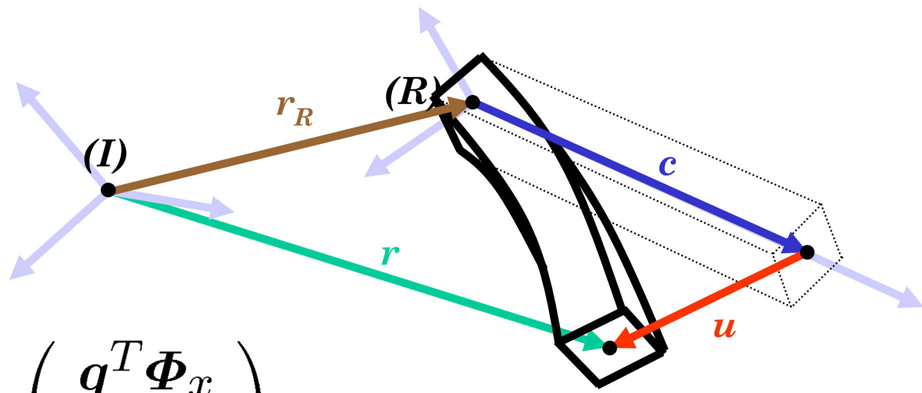




FlexibleBodies Library: Equations of motion

- Floating frame of reference

$$\boldsymbol{r} = \boldsymbol{r}_R + \boldsymbol{c} + \boldsymbol{u}$$



- Modal approach

$$\boldsymbol{u}(\boldsymbol{c},t) = \boldsymbol{\Phi}(\boldsymbol{c})\,\boldsymbol{q}(t) + \frac{1}{2}\begin{pmatrix} \boldsymbol{q}^T\boldsymbol{\Phi}_x \\ \boldsymbol{q}^T\boldsymbol{\Phi}_y \\ \boldsymbol{q}^T\boldsymbol{\Phi}_z \end{pmatrix}\boldsymbol{q}$$

- Equations of motion

$$\begin{pmatrix} m\boldsymbol{I}_3 & & \text{sym.} \\ m\tilde{\boldsymbol{d}}_{CM} & \boldsymbol{J} & \\ \boldsymbol{C}_t & \boldsymbol{C}_r & \boldsymbol{M}_e \end{pmatrix}\begin{pmatrix} \boldsymbol{a}_R \\ \boldsymbol{\alpha}_R \\ \ddot{\boldsymbol{q}} \end{pmatrix} = \boldsymbol{h}_\omega - \begin{pmatrix} 0 \\ 0 \\ \boldsymbol{K}_e\,\boldsymbol{q} + \boldsymbol{D}_e\,\dot{\boldsymbol{q}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{f}_a \\ \boldsymbol{f}_\alpha \\ \boldsymbol{f}_q \end{pmatrix}$$

Bremer/Pfeiffer ’92, Schwertassek/Wallrapp ’99

FlexibleBodies Library: Beam theory

- 2^{nd} order displacement field
  - bending in xy- und xz- plane, torsion and lengthening

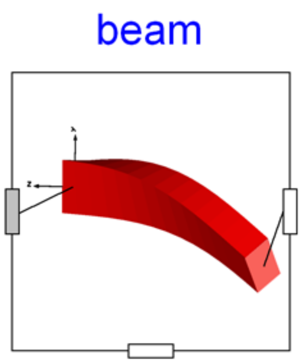
$$\boldsymbol{u}(x,t) = \begin{pmatrix} u \\ v \\ w \end{pmatrix} + \begin{pmatrix} -\frac{1}{2}\int_0^x v'^2 + w'^2 \,\mathrm{d}x \\ -\int_0^x\int_0^{\bar{x}} \theta\, w'' \,\mathrm{d}\bar{\bar{x}}\,\mathrm{d}\bar{x} + \int_0^x u'v' \,\mathrm{d}\bar{x} \\ \int_0^x\int_0^{\bar{x}} \theta\, v'' \,\mathrm{d}\bar{\bar{x}}\,\mathrm{d}\bar{x} + \int_0^x u'w' \,\mathrm{d}\bar{x} \end{pmatrix}$$

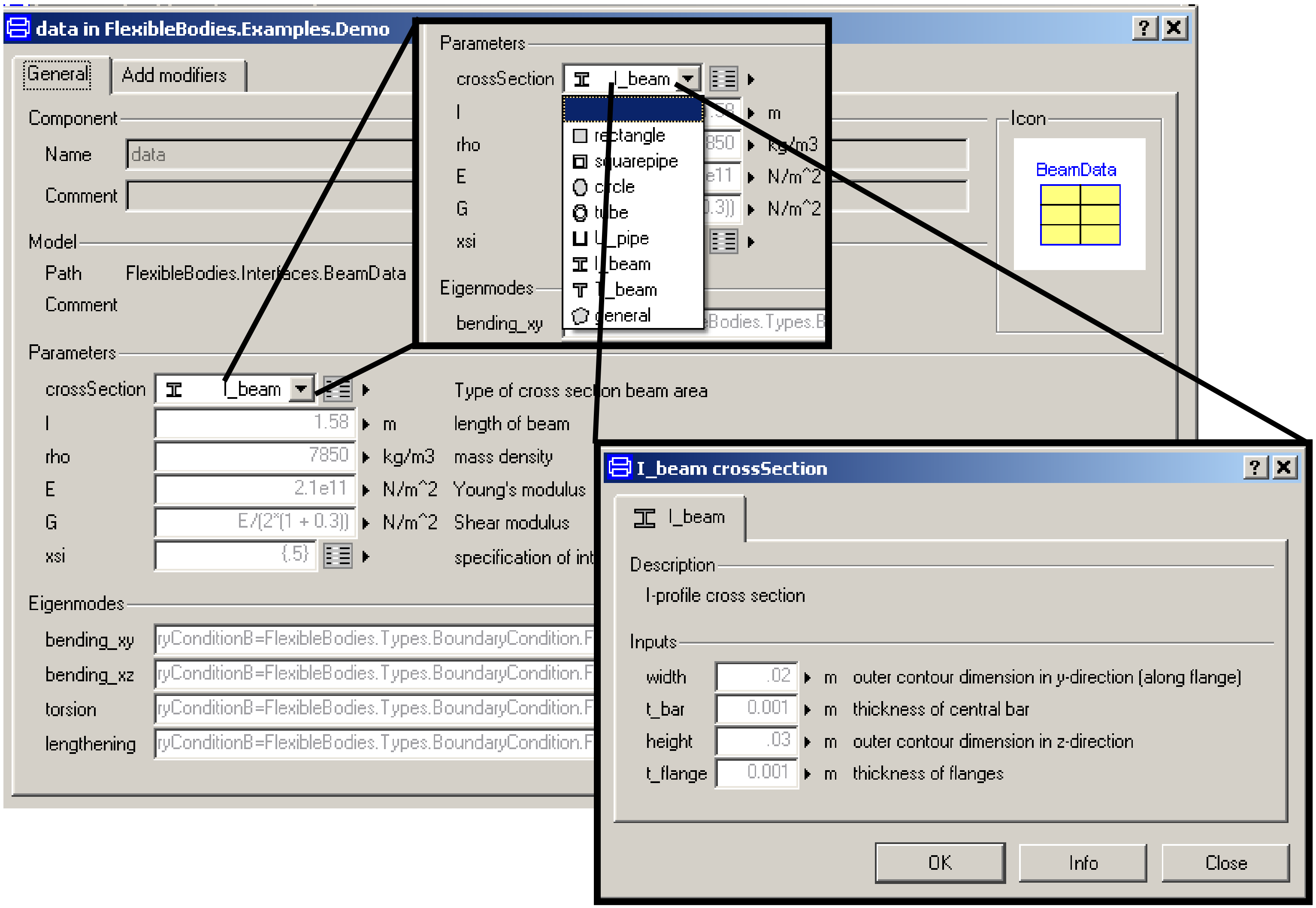
- Raleigh-Ritz-approach for a straight, homogenous, isotropic beam with constant cross section
  - expansion with analytic eigenvalue-solutions of the Euler-Bernoulli beam

$$v(x,t) = \boldsymbol{\Phi}_v(x)\boldsymbol{q}_v(t) \qquad \Phi_i = \begin{pmatrix} \cosh(\tau_i x) \\ \sinh(\tau_i x) \\ \cos(\tau_i x) \\ \sin(\tau_i x) \end{pmatrix}^T \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix}_i$$

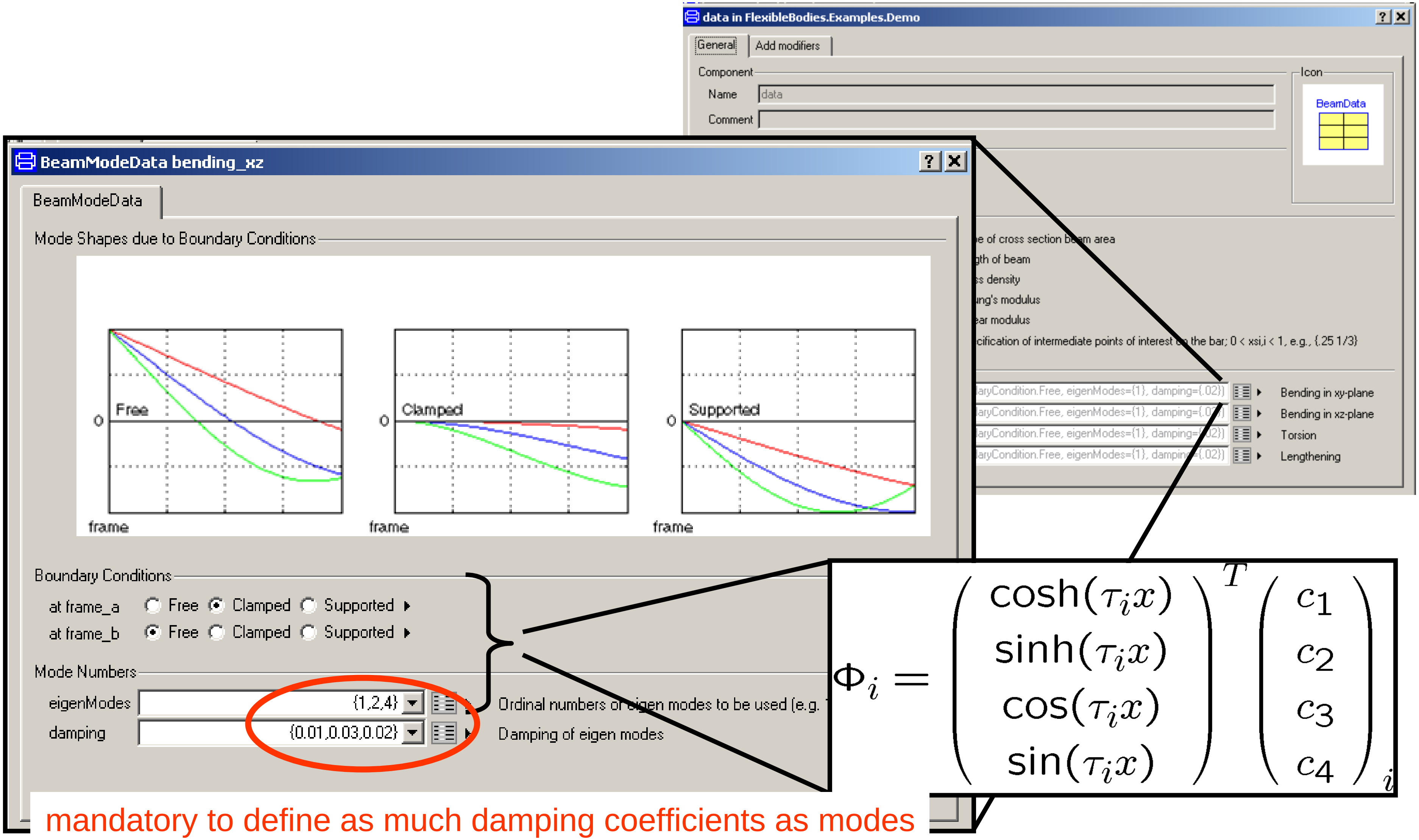
Bremer/Pfeiffer '92, Timoshenko '55

FlexibleBodies Library: Beam menu set-up I





FlexibleBodies Library: Beam menu set-up II



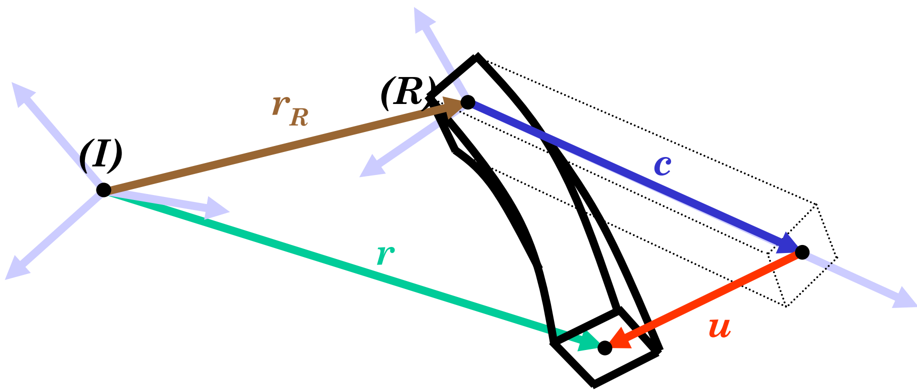
$$\Phi_i = \begin{pmatrix} \cosh(\tau_i x) \\ \sinh(\tau_i x) \\ \cos(\tau_i x) \\ \sin(\tau_i x) \end{pmatrix}^T \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix}_i$$

mandatory to define as much damping coefficients as modes

FlexibleBodies Library: Boundary Conditions I

- Mechanical interpretation

$$\boldsymbol{r}(\boldsymbol{c},t) = \boldsymbol{r}_R(t) + \boldsymbol{c} + \boldsymbol{u}(\boldsymbol{c},t)$$

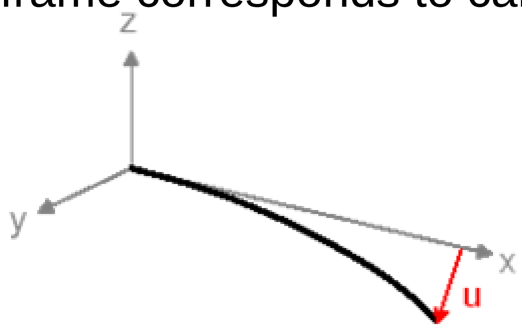


- let's say: frame of reference is pinned at frame_a with $\boldsymbol{c} = 0$

  $\Rightarrow$ motion of frame_a is completely described by $\boldsymbol{r}_R(t)$
  (and related orientation)

  $\Rightarrow$ $\boldsymbol{u}(\boldsymbol{c}=0,t) = 0 \quad \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{c}}(\boldsymbol{c}=0,t) = 0$

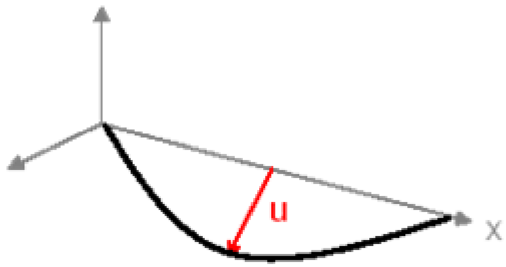
- clamped-free: tangent frame corresponds to cantilever beam boundary conditions



FlexibleBodies Library: Boundary Conditions II

- supported-supported: chord frame

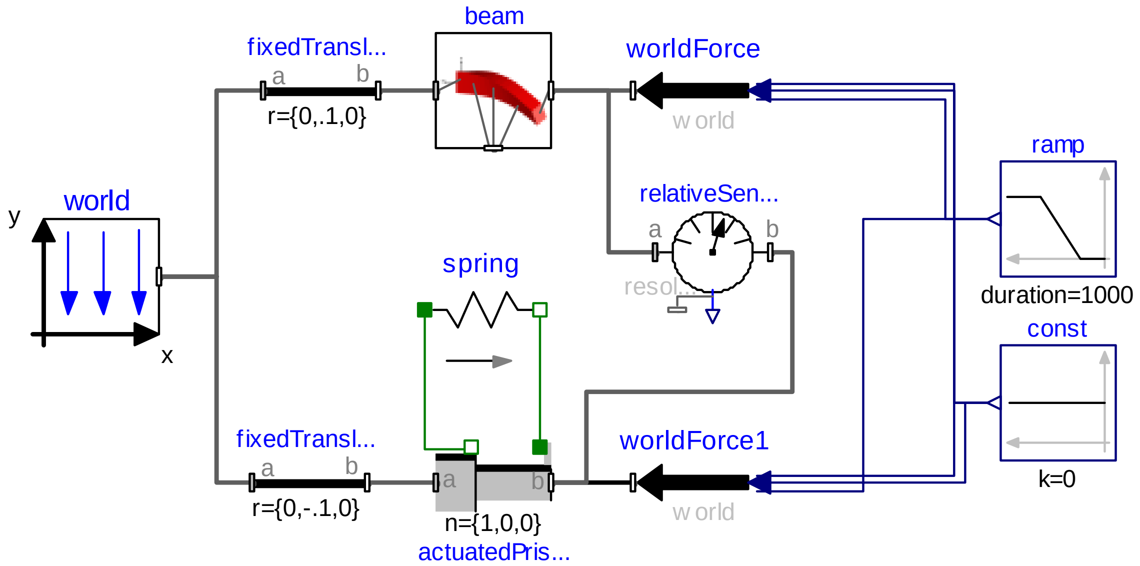
$$u(c = 0, t) = 0 \quad u(c = (l, 0, 0), t) = 0$$

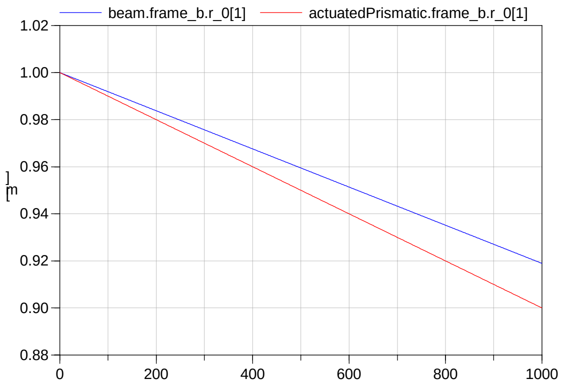


- free-free: Buckens frame
  - linear and angular momentum due to body deformation are zero
- every combination of tangent, chord and Buckens frames in different spatial directions is possible
- General rule
  - align the boundary conditions with the degree of freedom of the joints to which the beam is attached
    - boundary conditions are related to constraint forces
    - a joint cannot transmit a constraint force in the direction of its motion
  - BUT: Boundary conditions are validation issues

FlexibleBodies Library: A classic pitfall I

- static deflection: thrust force shortens beam and equivalent spring

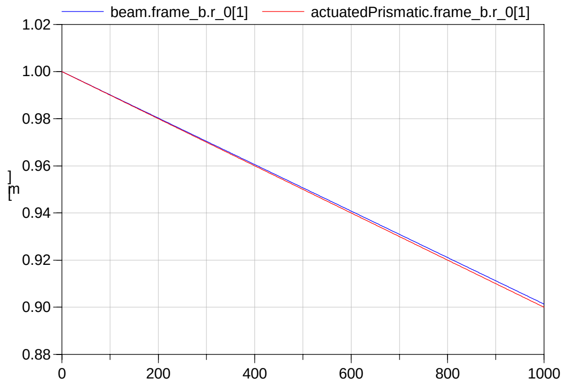




1 eigenmode

	spring	beam	error
1 eigenmode	-10 cm	-8.1 cm	19 %
5 eigenmodes	-10 cm	-9.6 cm	4 %
10 eigenmodes	-10 cm	-9.8 cm	2 %
15 eigenmodes	-10 cm	-9.9 cm	1 %

comparison: deflections at the end



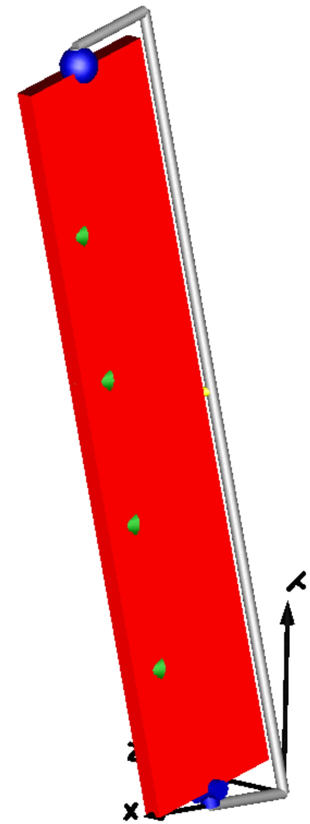
15 eigenmodes

FlexibleBodies Library: A classic pitfall II

- Mechanical background
  - static deflections rely on elastic properties only
  - eigenmodes consider elastic and interia properties
    - that‘s why they are well suited for dynamic problems
- Geometrical background
  - analytically: $u = c \cdot x$
  - expansion with eigenmodes: $u = \sin(\frac{2x}{\pi l}) + \sin(\frac{2x}{3\pi l}) + \ldots$
- It is proven that Raleigh-Ritz approach converges against true value
  - but how fast ?
  - this is an extreme example, e.g. bending is less sensitive
- Check whether a higher number of modes changes results !

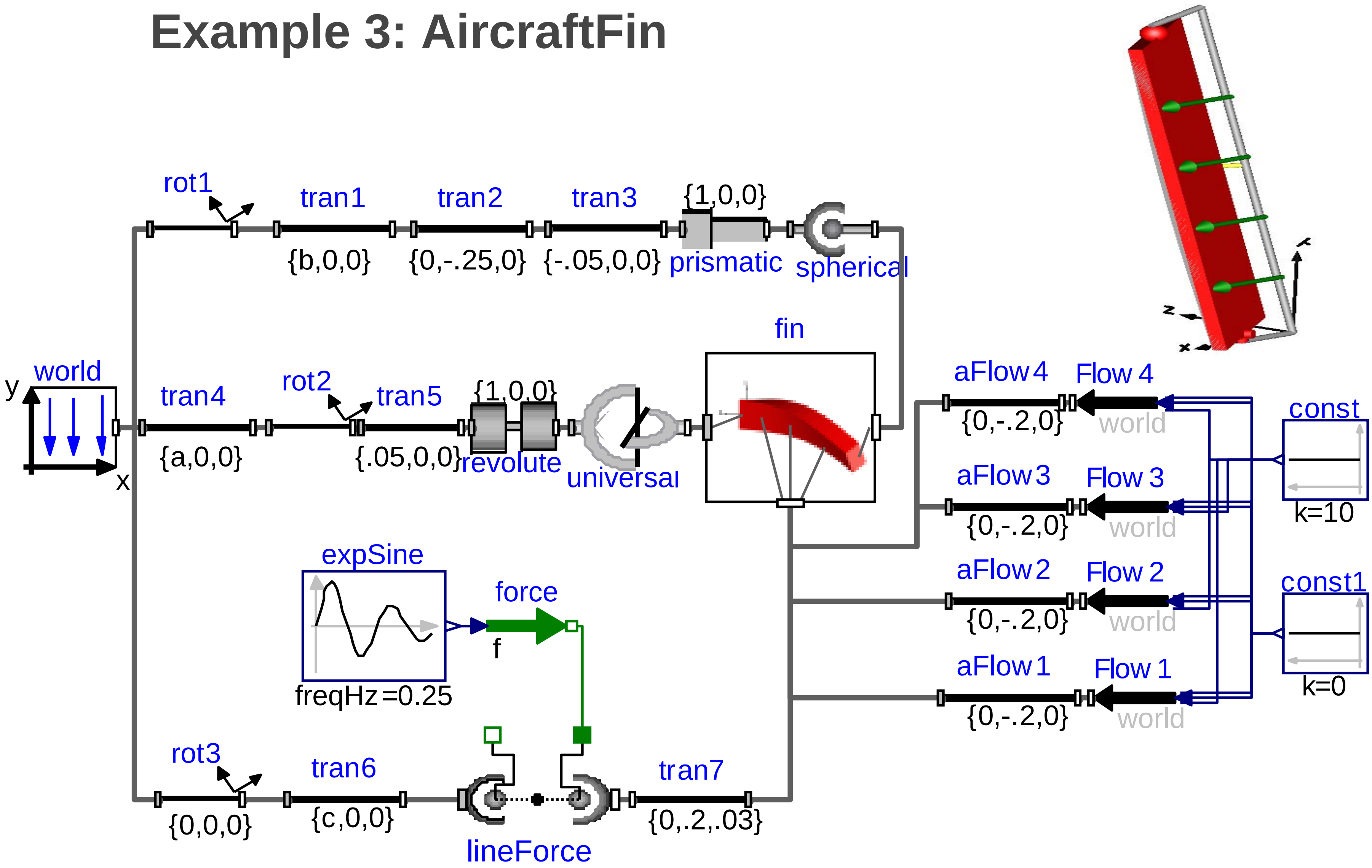
Example 3: AircraftFin

- fin
  - mounted in a frame with 15° inclination
  - profil: squarepipe 0.4 x 0.05 x 0.01 m
  - 2 m long
  - 2730 kg/m^3, $7 \cdot 10^{10}$ N/m^2 (aluminium)
  - 1 xz-bending and 1 torsion mode
  - 5 mid-nodes ξ = {0.2, 0.4, 0.5, 0.6, 0.8}
- actuator to turn the fin
  (force as cos-function of t, 1 Hz, 10 N amplitude)
- 4 constant forces, 10 N in world-x-direction,
  to represent flow forces



```
model AircraftFin

  import Modelica.Constants.pi;
  parameter Real a= 0.25/cos(pi/12);
  final parameter Real b=a*sin(pi/12)+2.1;
  final parameter Real c=a*sin(pi/12)+1.05;
```

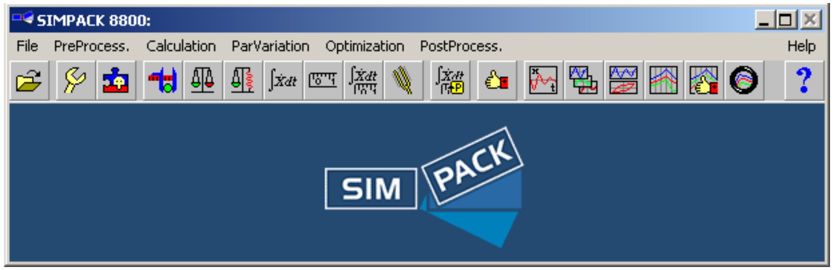
Example 3: AircraftFin



Contents

FlexibleBodies Library: FEM-preprocessing

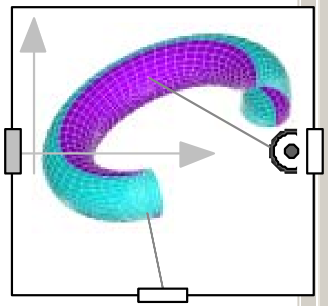
- Simpack-FEMBS: FEM to multibody system preprocessor
- Maintained and distributed by INTEC GmbH, Oberpfaffenhofen

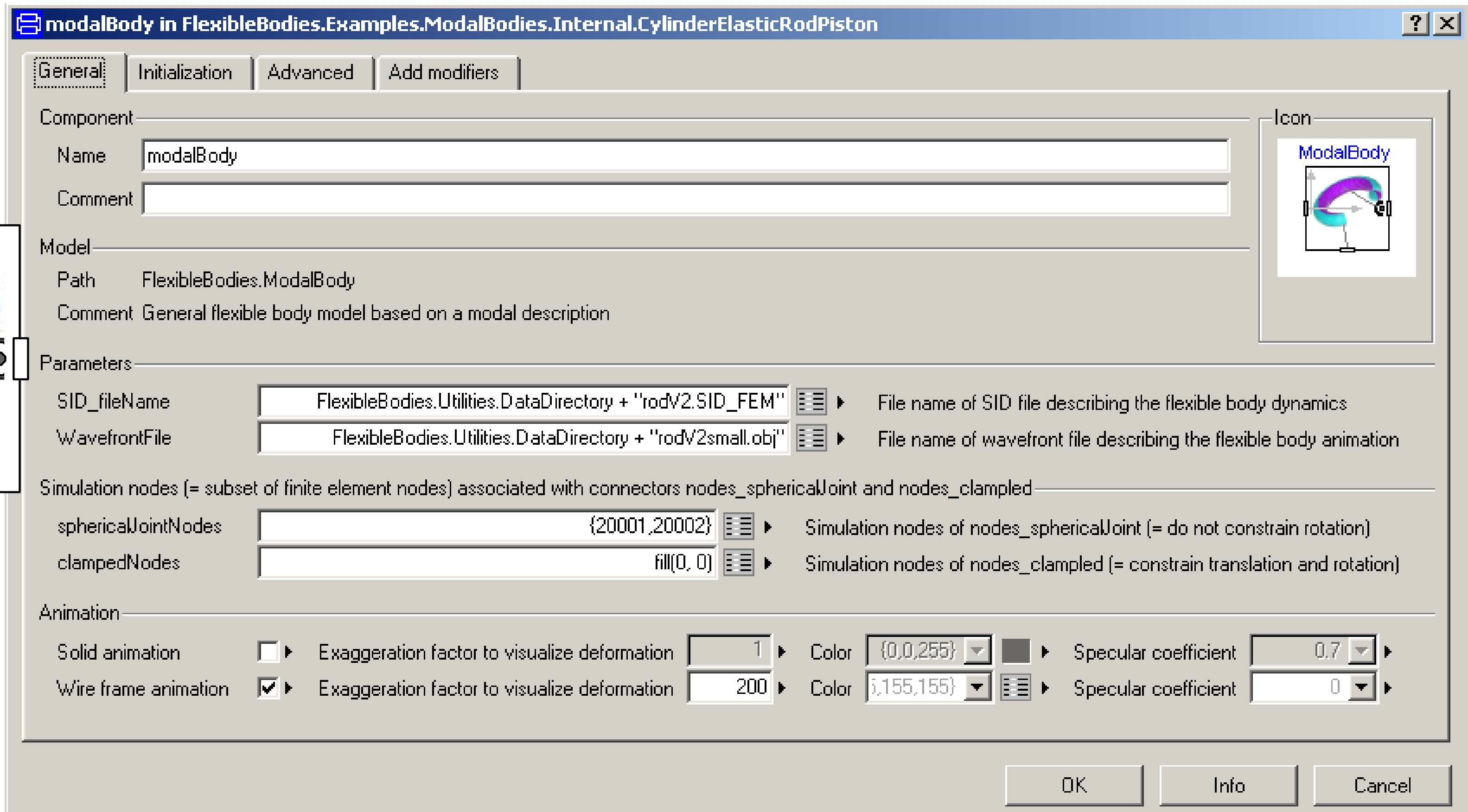


- Supports ABAQUS, ANSYS, MSC.Nastran, NX Nastran, I-DEAS, PERMAS
- Reduction of the FE-model in 2 steps
  - in FE-tool, e.g. Guyan- or Craig-Bampton-method
    $\Rightarrow$ system matrices, nodes to retain, eigenmodes, mesh information
  - in Simpack-FEMBS
    $\Rightarrow$ modes selection (and generation), multibody description, animation data
- SID- and wavefront-file as results

FlexibleBodies Library: ModalBody menu set-up I

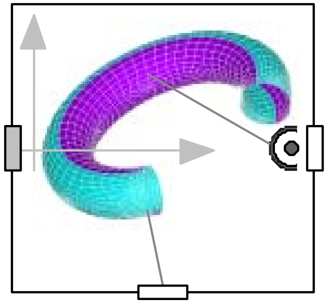
modalBody

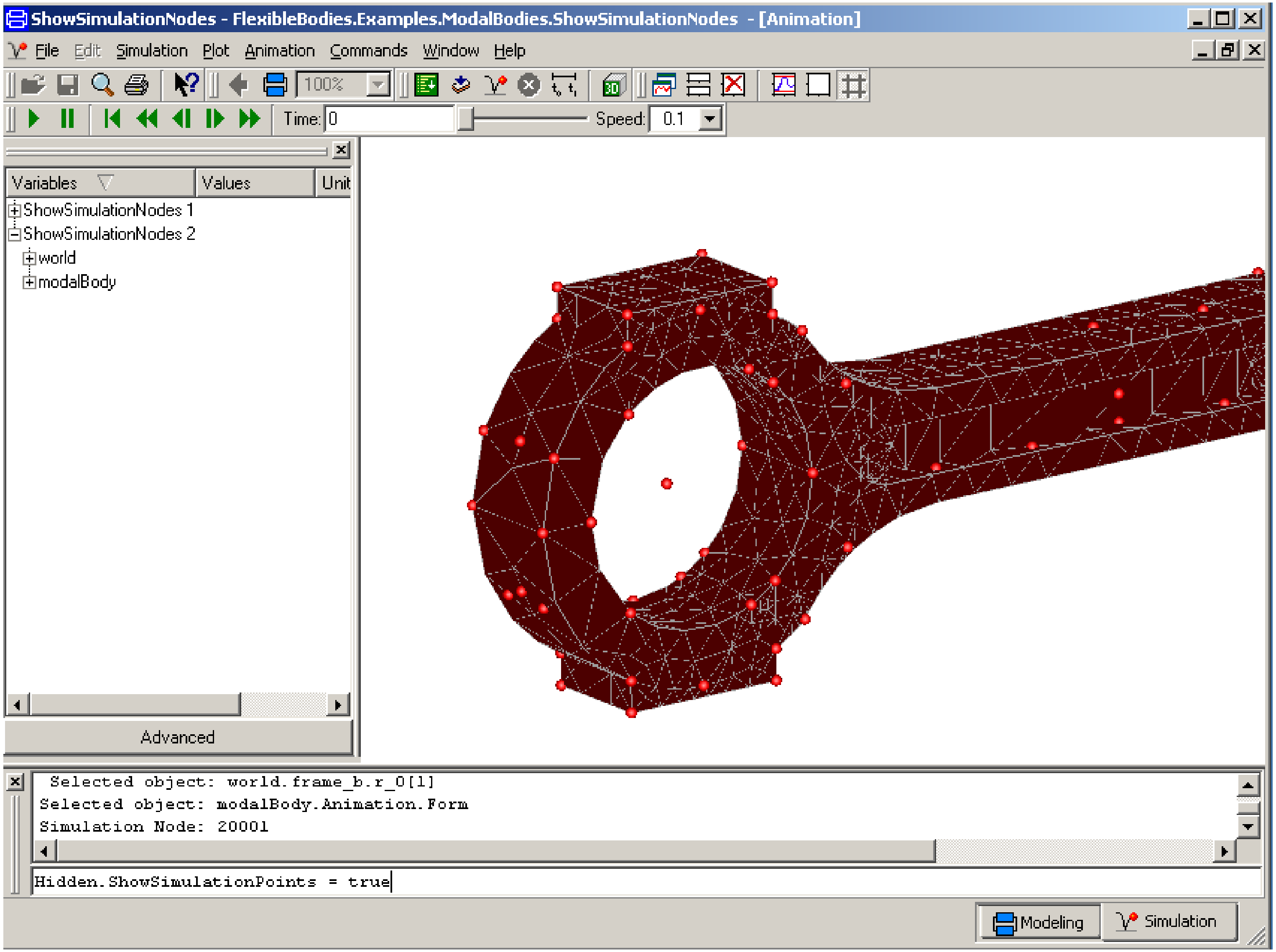




FlexibleBodies Library: ModalBody menu set-up II

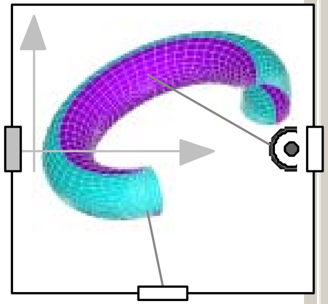
modalBody

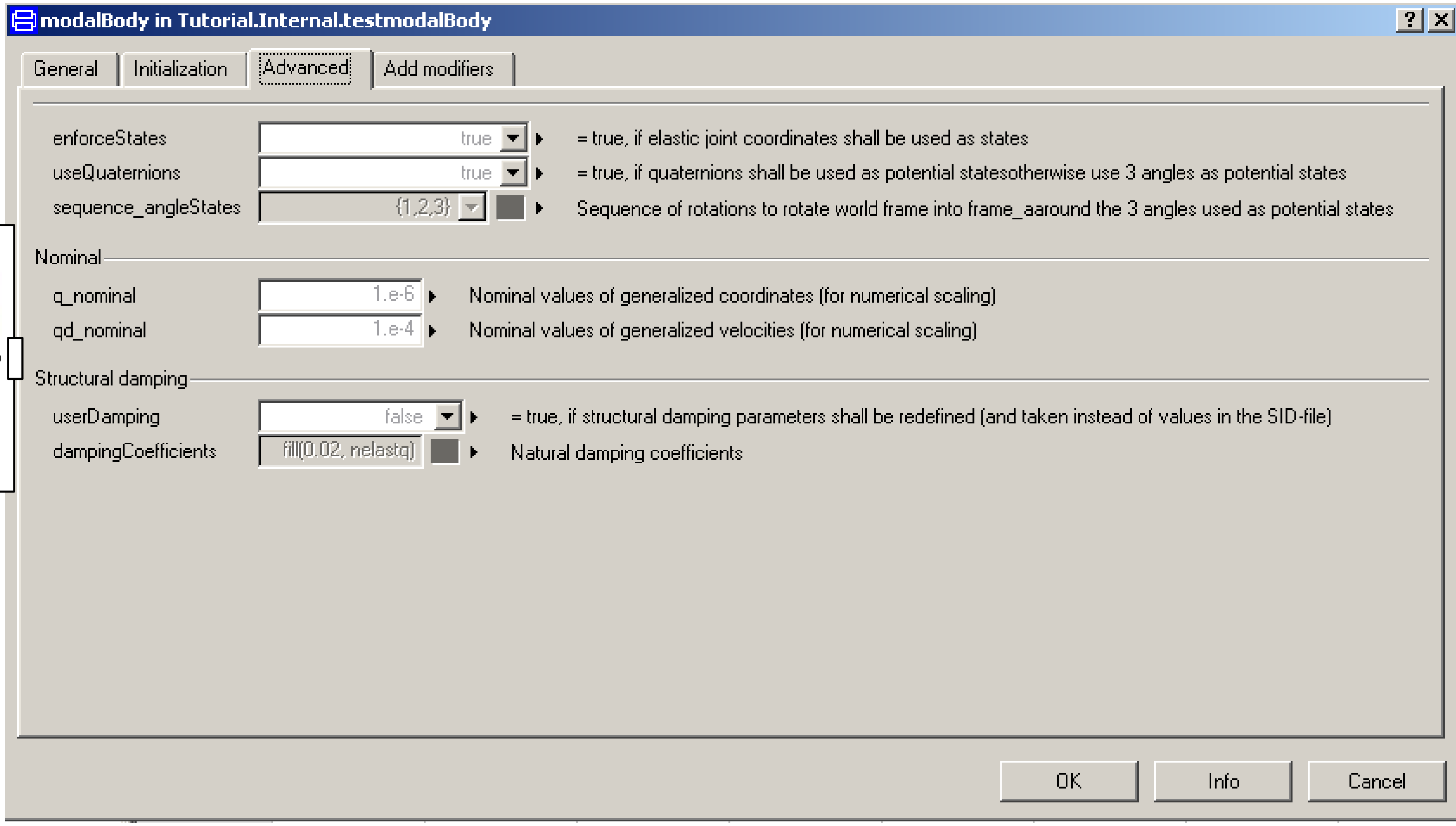




FlexibleBodies Library: ModalBody menu set-up II

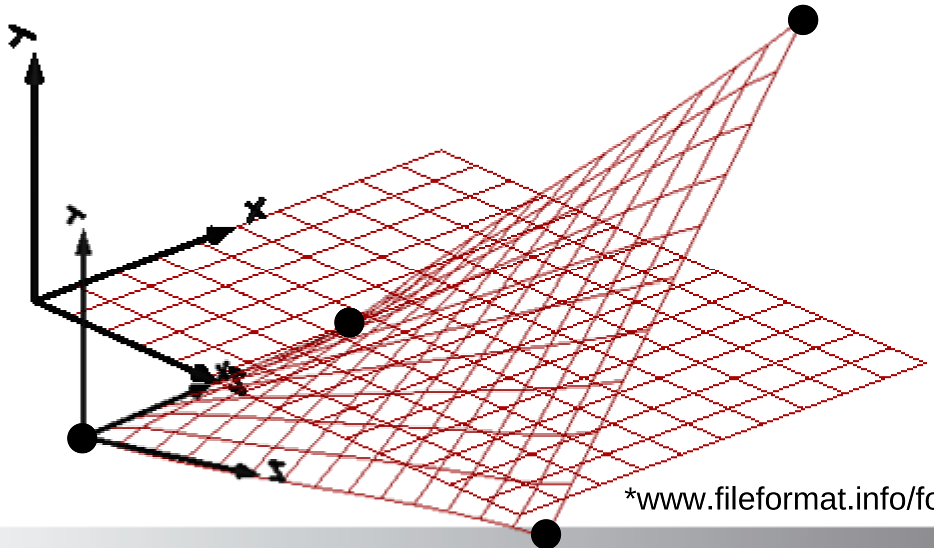
modalBody





FlexibleBodies Library: Animation

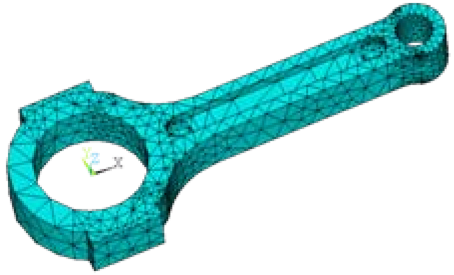
- Advantage of the modal approach:
  - only few geometrical information is needed ⇒ efficiency !
- Disadvantage of the modal approach:
  - only few geometrical information is needed ⇒ animation ?
- Simulation points ● versus animation points
  - new animation feature in Dymola for wavefront data*

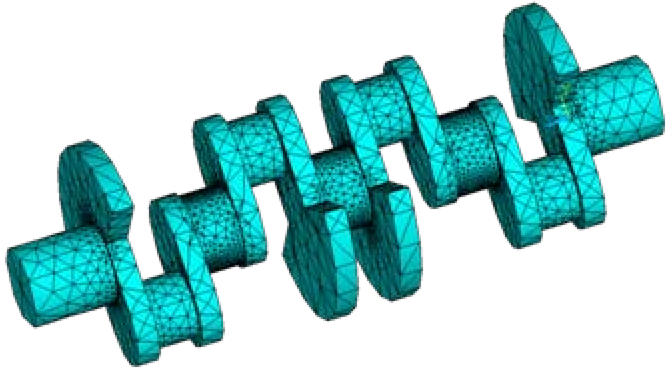


*www.fileformat.info/format/wavefrontobj/egff.htm

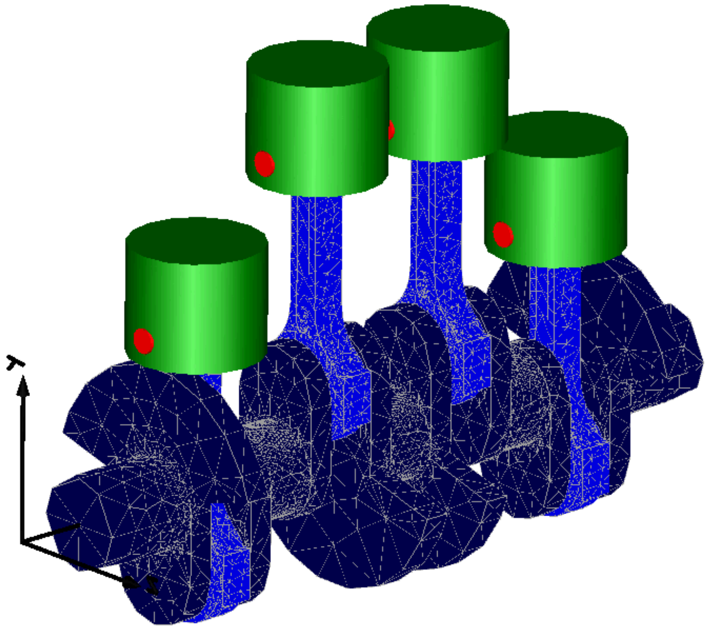
FlexibleBodies Library: 4 Cylinder Engine

- FEM
  - Crankshaft 85 342 nodes
  - Piston rod 12531 nodes
- Multibody representation
  - $<$ 1900 Hz
  - Crankshaft
    - 2 torsion eigenmodes
    - 273 simulation nodes
  - Piston rod
    - 4 eigenmodes each
    - 120 simulation nodes
  - Time integration with gas force, 38 states, $\approx$6 cpu-s per s





FlexibleBodies Library: 4 Cylinder Engine II



FlexibleBodies Library: Quo vadis?

- ModalBody is an issue of ongoing discussion
  - relies on an appopriate FE-preprocessor
    - a 3rd third party product
    - in principle agreements were made, but no contracts are yet signed
  - the 1st implementation uses additional c-code
    - there is no distribution process to support the application on real-time platforms today
  - the 2nd implemenation is pure Modelica code
    - might be good enough for moderate large models ?
    - release by the end of this year ?
  - an additional license key called ExtFlexibleBodies
  - graphical nodes-picking requires improvements
- Besides beams: implementation of other generic structures „on demand“
  - brake discs: annular Kirchhoff plate
  - framework structures

Thank you very much for your attention