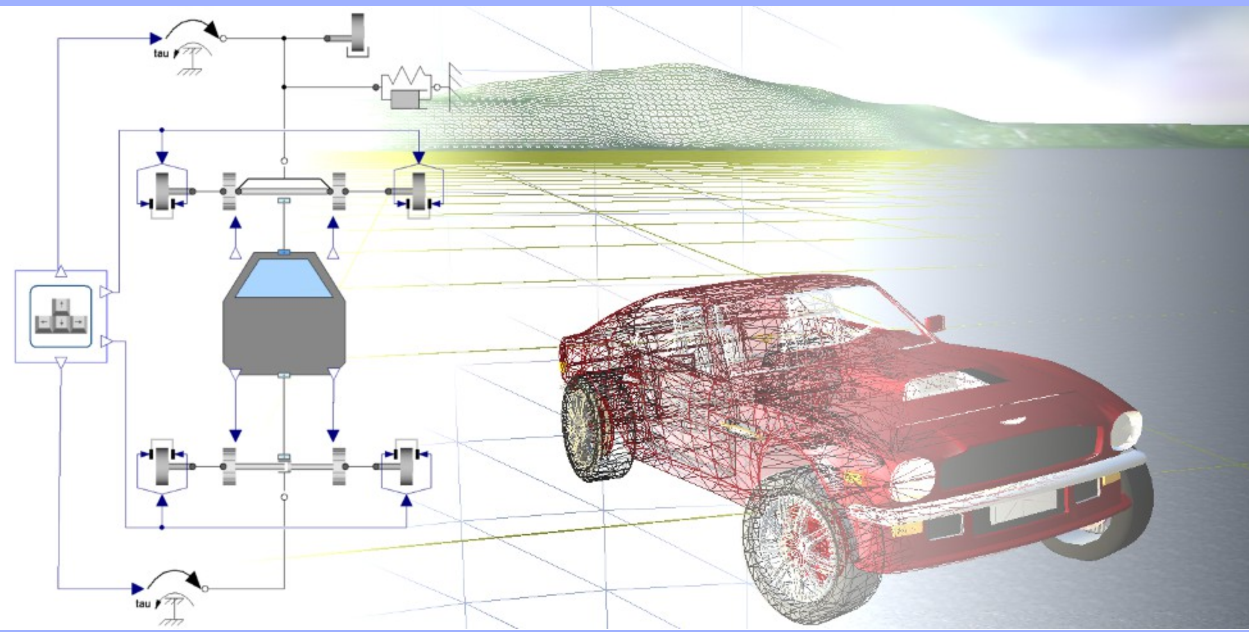
PlanarMechanics v1.2
A free Modelica Library

OpenModelica Annual Workshop,
February 3, 2014,
Linköping, Sweden

```
equation
  sx0 = cos(frame_a.phi)*sx_norm + …
  sy0 = -sin(frame_a.phi)*sx_norm + …
  vy = der(frame_a.y);
  w_roll = der(flange_a.phi);
  v_long = vx*sx0 + vy*sy0;
  v_lat = -vx*sy0 + vy*sx0;
  v_slip_lat = v_lat - 0;
  v_slip_long = v_long - R*w_roll;

  v_slip = sqrt(v_slip_long^2 + …
  -f_long*R = flange_a.tau;
  frame_a.t = 0;
  f = N*. S_Func(vAdhesion,vSlide,…
  f_long =f*v_slip_long/v_slip;
  f_lat  =f*v_slip_lat/v_slip;
  f_long = frame_a.fx*sx0 + …
  f_lat = -frame_a.fx*sy0 + …
```



Dr. Dirk Zimmer, Franciscus van der Linden, Zheng Qu

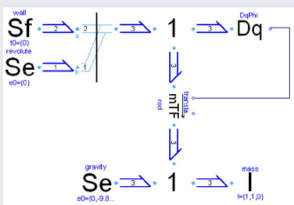
German Aerospace Center (DLR), Institute of System Dynamics and Control

Timeline

Institute of System Dynamics and Control

2006

Multi
BondLib

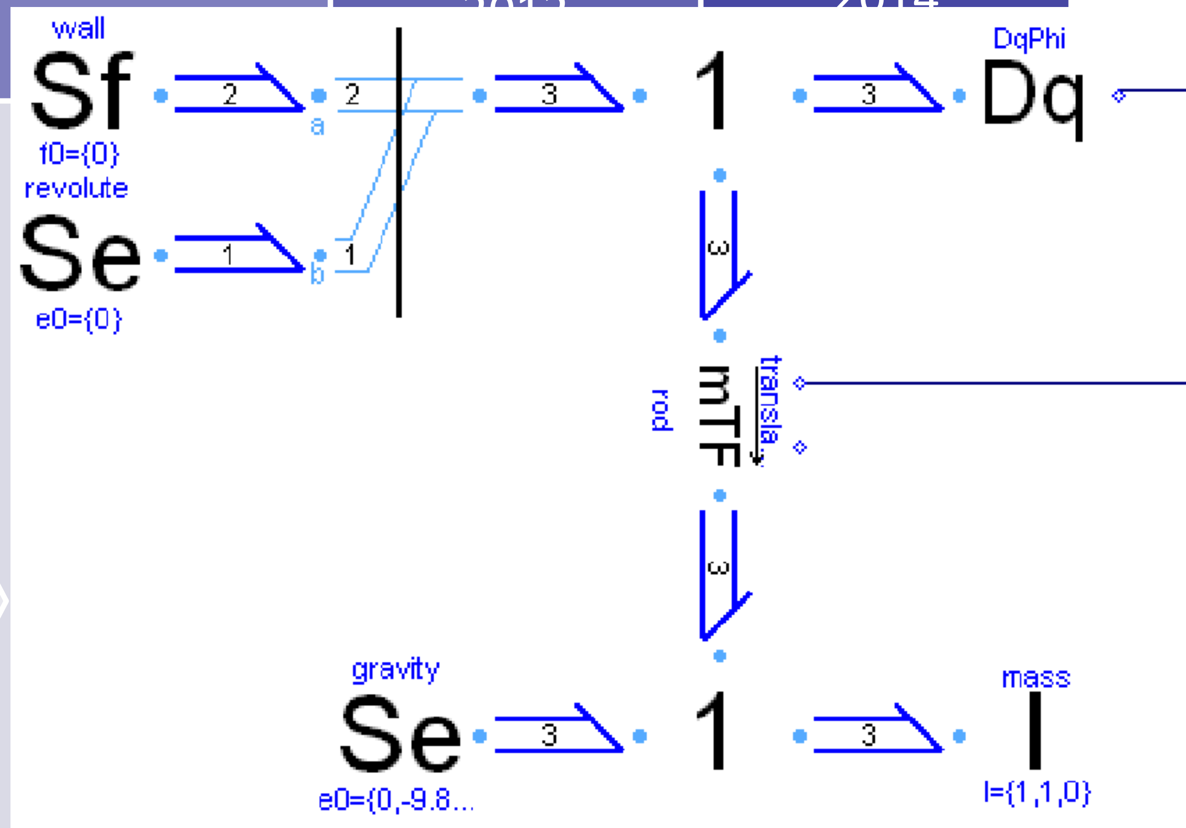


Part of Ms
Thesis

2014

Support!

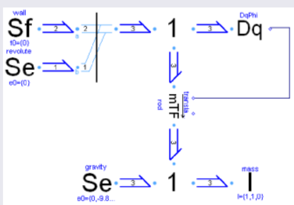
applications



Timeline

2006

Multi BondLib



Part of Ms Thesis

2010

Planar Mechanics for Teaching

Master-Level Course at TUM

2012

Standard Planar Mechanics

9th INTERNATIONAL MODELICA CONFERENCE

Presented at Modelica 2012

Wide Tool Support!

2014

Free Planar Mechanics Library

designed towards use in practical applications

Contributors

Dirk Zimmer (DLR)

- initial version, wheel models, etc.

Franciscus van der Linden (DLR)

- Gearbox components
- spring and dampers

Zhen Qu (DLR, TUM)

- Sensors,
- World model,
- improved animation, initialization, documentation

Outline

Institute of System Dynamics and Control

Part 1:

- Basics of Planar Mechanics (Excerpts from my lecture)

Part 2:

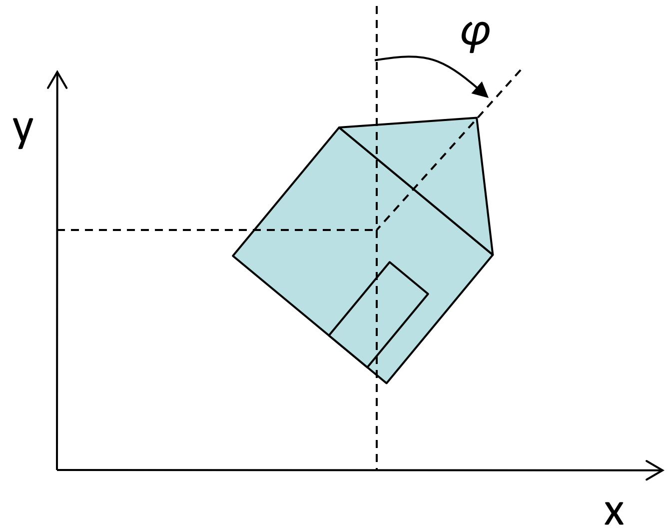
- New features / improvements of the free PlanarMechanics Library

Part 1:

- Basics of Planar Mechanics (Excerpts from my lecture)

Part 1: Planar Mechanics

- In planar mechanics, we describe the physics of a multi body system in a two-dimensional plane.



- In the planar world, all motions and positions can be described by two translational positions and an angular orientation
- By convention we denote the horizontal position with x, the vertical position with y and the orientation by the angle φ (phi).

Why Planar Mechanics?

- **Essentially**: **Sometimes 1D is too simple, 3D is too complex**.
- Tangible and visual systems
- The simulation results can be visualized and animated.
- Fundamental formulas (Newton's law, D'Alembert's principle) are taught already in high-school.
- Everyone has an intuitive understanding about the motion of mechanical systems and how to control it.
- Interactions to most other domains. (Electrical Engines, Hydraulics, Heat)

Connector Variables

Institute of System Dynamics and Control

- From 1D-mechanics, we know that the we should choose force and torque as flow-variables and position and angle as potential variables.
- Planar mechanics combine three 1D-subsytems. Hence the following connector design seems natural.

Potential variables

x (horizontal position)

y (vertical position)

φ (orientation angle)

Flow variables

f_x (horizontal force)

f_y (vertical force)

τ (torque)

Connector Variables: Modelica

- Here, the corresponding Modelica-Code:

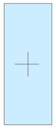
```
connector Frame
  "General Connector for planar mechanical components"

  SI.Position x      "x-position";
  SI.Position y      "y-position";
  SI.Angle phi       "angle (clockwise)";
  flow SI.Force fx   "force in x-direction";
  flow SI.Force fy   "force in y-direction";
  flow SI.Torque t   "torque (clockwise)";

end Frame;
```

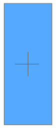
Connectors

Institute of System Dynamics and Control

- It is common style to extend two connectors with different icons from the general connector.
- Some components contain characteristics that are directed. Hence it is helpful to see, if your connecting to side A or side B.

```
connector Frame_a
  extends Frame;
end Frame_a;


connector Frame_b
  extends Frame;
end Frame_b;
```





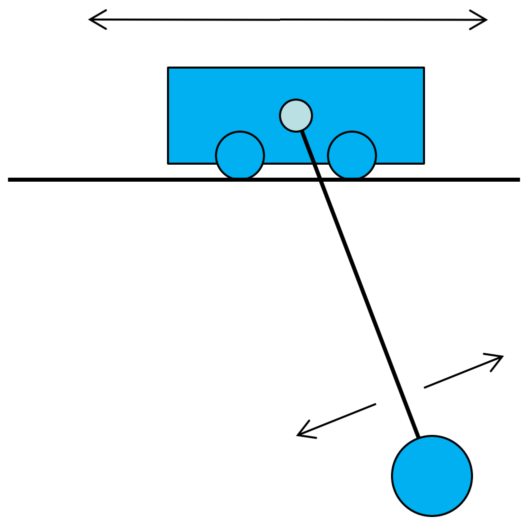
- All of these connectors are collected in an interface package.

Decomposition into components

Institute of System Dynamics and Control

- Using this connector, we can build all components for a crane crab

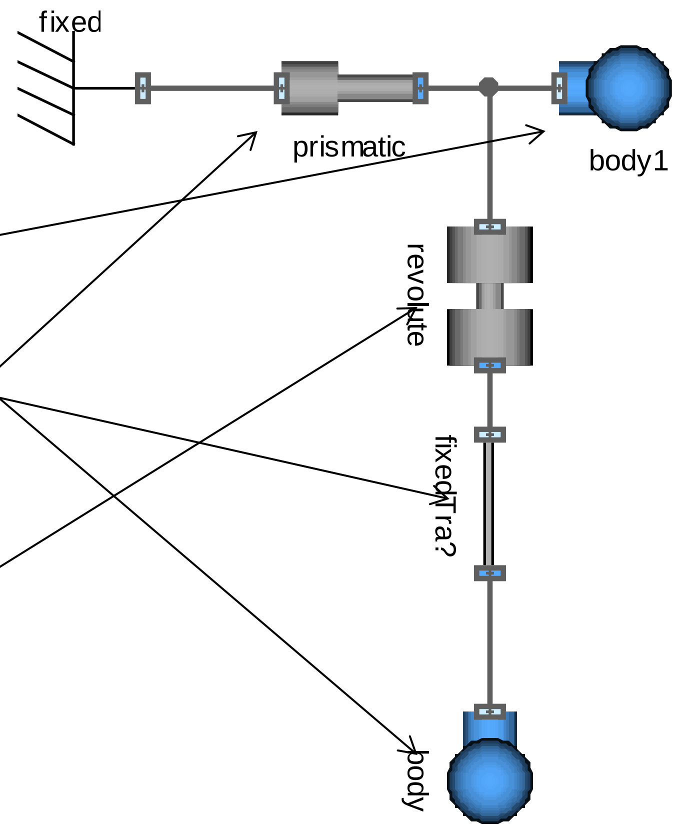
- Parts
  - Wall
  - Body (Mass and Inertia)
  - FixedTranslation
  - FixedRotation
- Joints
  - Prismatic Joint
  - Revolute Joint



Decomposition into components

DLR
Institute of System Dynamics and Control

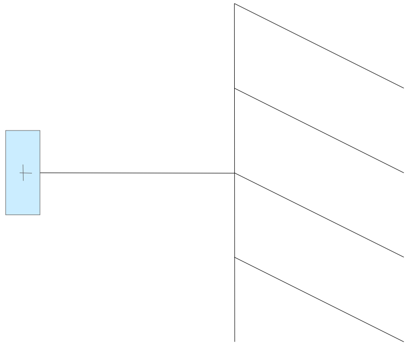
- Parts
  - Wall
  - Body (Mass and Inertia)
  - FixedTranslation
  - FixedRotation
- Joints
  - Prismatic Joint
  - Revolute Joint



Fixed Component

Institute of System Dynamics and Control

We can already model the first basic components. Let us start with the wall component that represents a fixation point.

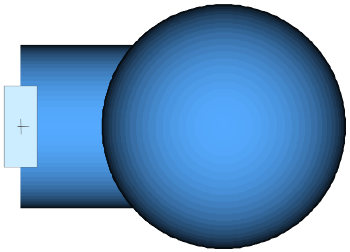


```
model Fixed "FixedPosition"

  Interfaces.Frame_a frame_a;
  parameter SI.Position r[2] = {0,0};
   "fixed x-position";
  parameter SI.Angle phi = 0
    "fixed angle";

equation
  {frame_a.x, frame_a.y} = r;
  frame_a.phi = phi;

end Fixed;
```

Body Component

Institute of System Dynamics and Control

- A little more elaborate is the body-component that represents a mass with inertia.



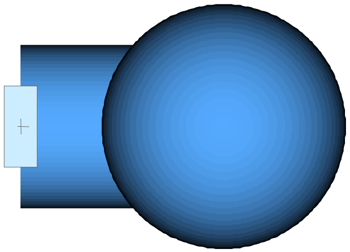
- Essentially, the model formulates Newton's law.

```
model Body
  Interfaces.Frame_a frame_a;

  parameter SI.Mass m;
  parameter SI.Inertia I;

  SI.Force f[2];
  SI.Position r[2];
  SI.Velocity v[2];
  SI.Acceleration a[2];
  SI.AngularVelocity w;
  SI.AngularAcceleration z;

equation
  r = {frame_a.x, frame_a.y}
  v = der(r);
  w = der(frame_a.phi);

  a = der(v);
  z = der(w);

  f = {frame_a.fx, frame_a.fy};
  f = m*a;
  frame_a.t = I*z;
end Body
```

Body Component

Institute of System Dynamics and Control

- Since the gravitational force is dependent on the mass (m*g), it makes sense to compute right in the body model.



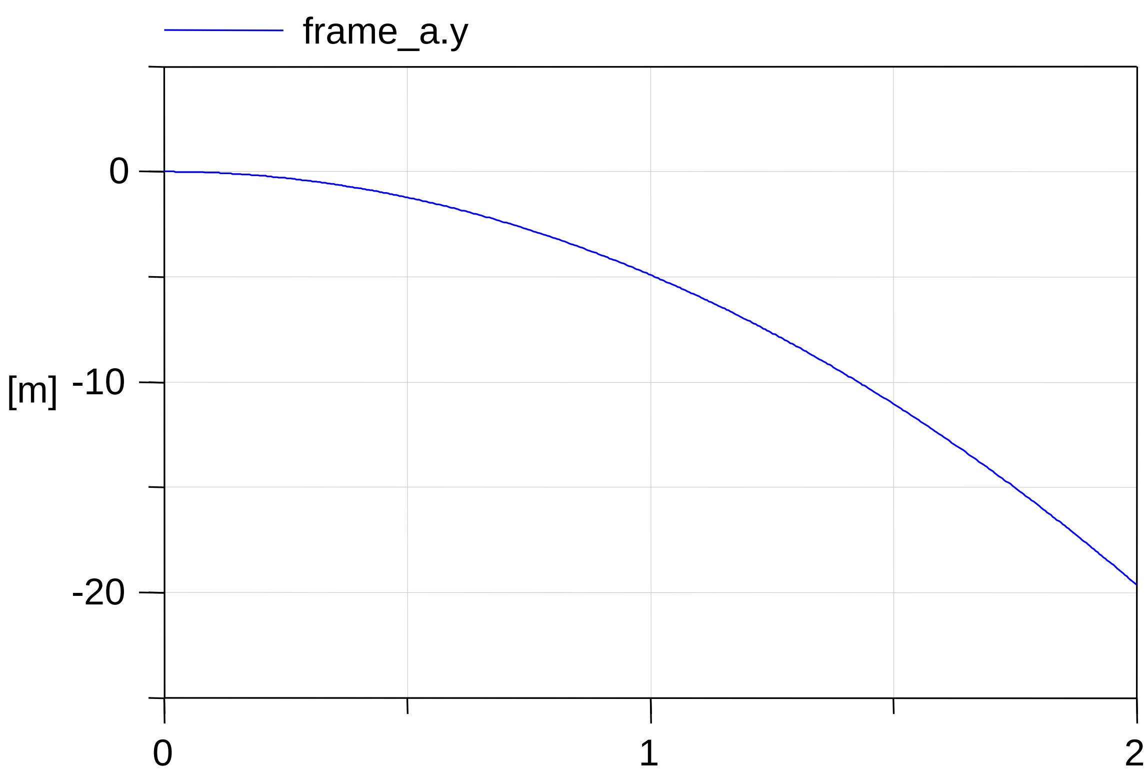
- A parameter for the gravitational acceleration is added and Newton's law is extended.

```
model Body
  Interfaces.Frame_a frame_a;

  parameter SI.Mass m;
  parameter SI.Inertia I;
  parameter SI.Acceleration[2] g={0,-9.81};
  SI.Force f[2];
  SI Position r[2];
  SI.Velocity v[2];
  SI.Acceleration a[2];
  SI.AngularVelocity w;
  SI.AngularAcceleration z;

equation
  r = {frame_a.x, frame_a.y}
  v = der(r);
  w = der(frame_a.phi);

  a = der(v);
  z = der(w);

  f = {frame_a.fx, frame_a.fy};
  f + m*g = m*a;
  frame_a.t = I*z;
end Body
```

Simulating the body model

Institute of System Dynamics and Control

- Here is the simulation result:



- It shows the parabolic descent of a body due to gravity acceleration.

Components with two Flanges

Institute of System Dynamics and Control

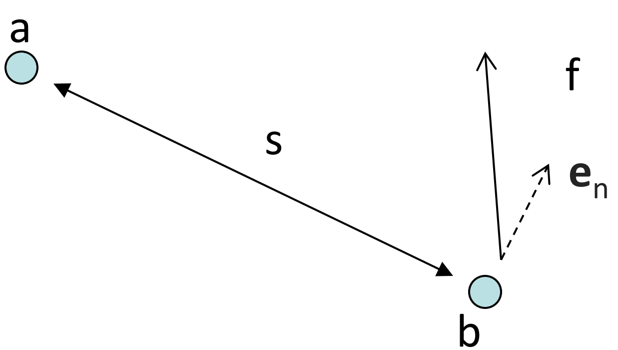
- Components that have two frames are little more difficult.
- Let us start by modeling a neutral component.
- The model itself is rather meaningless but it represents a good starting point for the design of any new component.

```
model Neutral
  Interfaces.Frame_a frame_a;
  Interfaces.Frame_a frame_a;



equation


  frame_a.fx = 0;
  frame_a.fy = 0;
  frame_a.t = 0;


  frame_a.fx + frame_b.fx = 0;
  frame_a.fy + frame_b.fy = 0;
  frame_a.t
  + frame_b.t
  - (frame_b.x - frame_a.x)*frame_b.fy
  + (frame_b.y – frame_a.y)*frame_b.fx
  = 0;
end Neutral
```

Components with two Flanges

Institute of System Dynamics and Control

- The model imposes no constraints on the positions.
- This component has two frames, but exhibits no effect.
- The balance equations for the forces contains the lever principle.



$\tau = \mathbf{f} \cdot \mathbf{e}_n \cdot s = \mathbf{f} \cdot (\mathbf{e}_n \cdot s)$

$\tau = (fx, fy) \cdot (sy, -sx)$

```
model Neutral
  Interfaces.Frame_a frame_a;
  Interfaces.Frame_a frame_a;


equation


  frame_a.fx = 0;
  frame_a.fy = 0;
  frame_a.t = 0;


  frame_a.fx + frame_b.fx = 0;
  frame_a.fy + frame_b.fy = 0;
  frame_a.t
  + frame_b.t
  - (frame_b.x - frame_a.x)*frame_b.fy
  + (frame_b.y – frame_a.y)*frame_b.fx
  = 0;
end Neutral
```

Components with two Flanges

Institute of System Dynamics and Control

Guidelines:

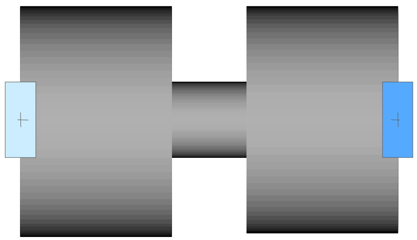
- For each positional constraint we add, we have to remove the corresponding force equation.
- For each variable that we add, we have to add an equation
- Finally, we may be able to simplify the balance equations.

```
model Revolute
  Interfaces.Frame_a frame_a;
  Interfaces.Frame_a frame_a;




equation


  frame_a.fx = 0;
  frame_a.fy = 0;
  frame_a.t = 0;


  frame_a.fx + frame_b.fx = 0;
  frame_a.fy + frame_b.fy = 0;
  frame_a.t
  + frame_b.t
  - (frame_b.x - frame_a.x)*frame_b.fy
  + (frame_b.y – frame_a.y)*frame_b.fx
  = 0;
end Revolute
```

Revolute Joint

Institute of System Dynamics and Control

Let us start with the revolute joint:

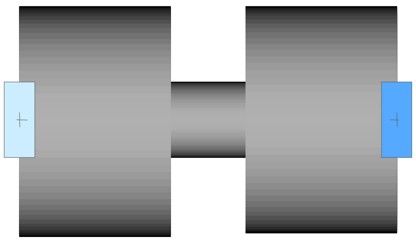


```
model Revolute
  Interfaces.Frame_a frame_a;
  Interfaces.Frame_a frame_b;







equation



  frame_a.fx = 0;
  frame_a.fy = 0;
  frame_a.t = 0;


  frame_a.fx + frame_b.fx = 0;
  frame_a.fy + frame_b.fy = 0;
  frame_a.t
  + frame_b.t
  - (frame_b.x - frame_a.x)*frame_b.fy
  + (frame_b.y – frame_a.y)*frame_b.fx
  = 0;
end Revolute
```

Revolute Joint

Institute of System Dynamics and Control

Let us start with the revolute joint:



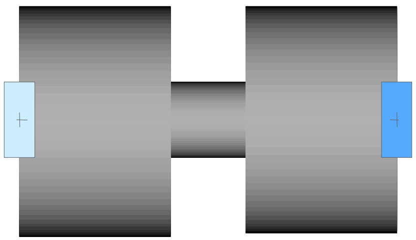
- The translational positions of a and b are equal. (2 constraints)
- No torque can act on the joint.

```
model Revolute
  Interfaces.Frame_a frame_a;
  Interfaces.Frame_a frame_b;




equation

  frame_a.fx = 0 replaced by
  frame_a.x = frame_b.x;
  frame_a.fy = 0 replaced by
  frame_a.y = frame_b.y;
  frame_a.t = 0;


  frame_a.fx + frame_b.fx = 0;
  frame_a.fy + frame_b.fy = 0;
  frame_a.t
  + frame_b.t
  - (frame_b.x - frame_a.x)*frame_b.fy
  + (frame_b.y – frame_a.y)*frame_b.fx
  = 0;

end Revolute
```

Revolute Joint

Institute of System Dynamics and Control

Let us start with the revolute joint:



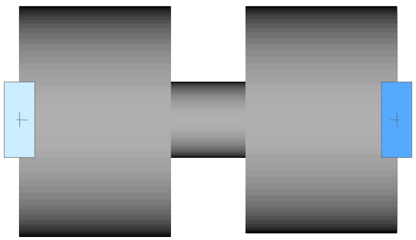
- The translational positions of a and b are equal. (2 constraints)
- No torque can act on the joint.
- The lever principle is redundant here…
- That's it! …actually

```
model Revolute
  Interfaces.Frame_a frame_a;
  Interfaces.Frame_a frame_b;

equation

  frame_a.fx = 0 replaced by
  frame_a.x = frame_b.x;
  frame_a.fy = 0 replaced by
  frame_a.y = frame_b.y;
  frame_a.t = 0;

  frame_a.fx + frame_b.fx = 0;
  frame_a.fy + frame_b.fy = 0;
  frame_a.t  + frame_b.t = 0;

end Revolute
```

Revolute Joint

Institute of System Dynamics and Control

Let us start with the revolute joint:



- For completeness, we'd like to add two differential equations for the angle, the angular velocity and its acceleration.
- After all, these variables are of interest.
- We can now use the joint in order to express motion.
- It also helps with initialization.

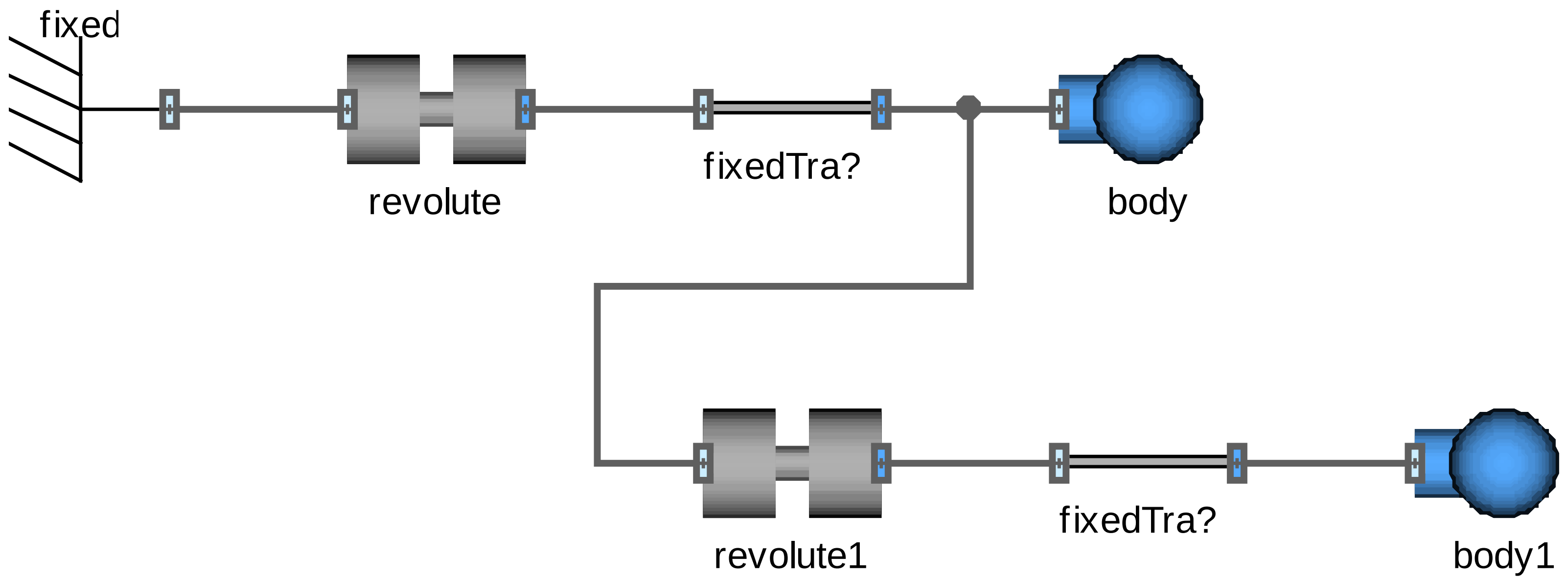
```
model Revolute
  Interfaces.Frame_a frame_a;
  Interfaces.Frame_a frame_b;

  SI.Angle phi
  SI.AngularVelocity w;
  SI.AngularAcceleration z;


equation
  frame_a.phi + phi = frame_b.phi;
  w = der(phi);
  z = der(w);

  frame_a.x = frame_b.x;
  frame_a.y = frame_b.y;
  frame_a.t = 0;


  frame_a.fx + frame_b.fx = 0;
  frame_a.fy + frame_b.fy = 0;
  frame_a.t  + frame_b.t = 0;


end Revolute
```

Further components

- We already have 3 components:
  - Fixation
  - Body
  - Revolute
- The lecture goes on explaining two more components:
  - Fixed Translation
  - Prismatic Joint
- Having just these five components available, we can already assemble many interesting systems.

Chaos: Double Pendulum

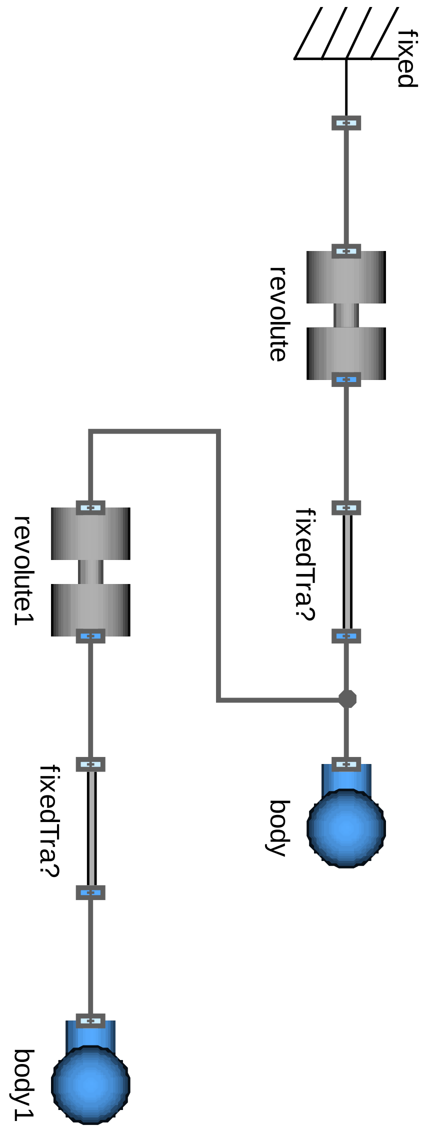
- A seemingly simple system is the double pendulum

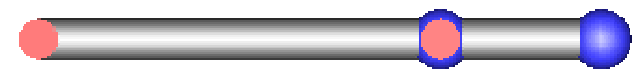


- Let us look at the motion of the peak of the second pendulum.

Double Pendulum

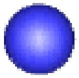
Institute of System Dynamics and Control





Double Pendulum: Peak Motion

Institute of System Dynamics and Control



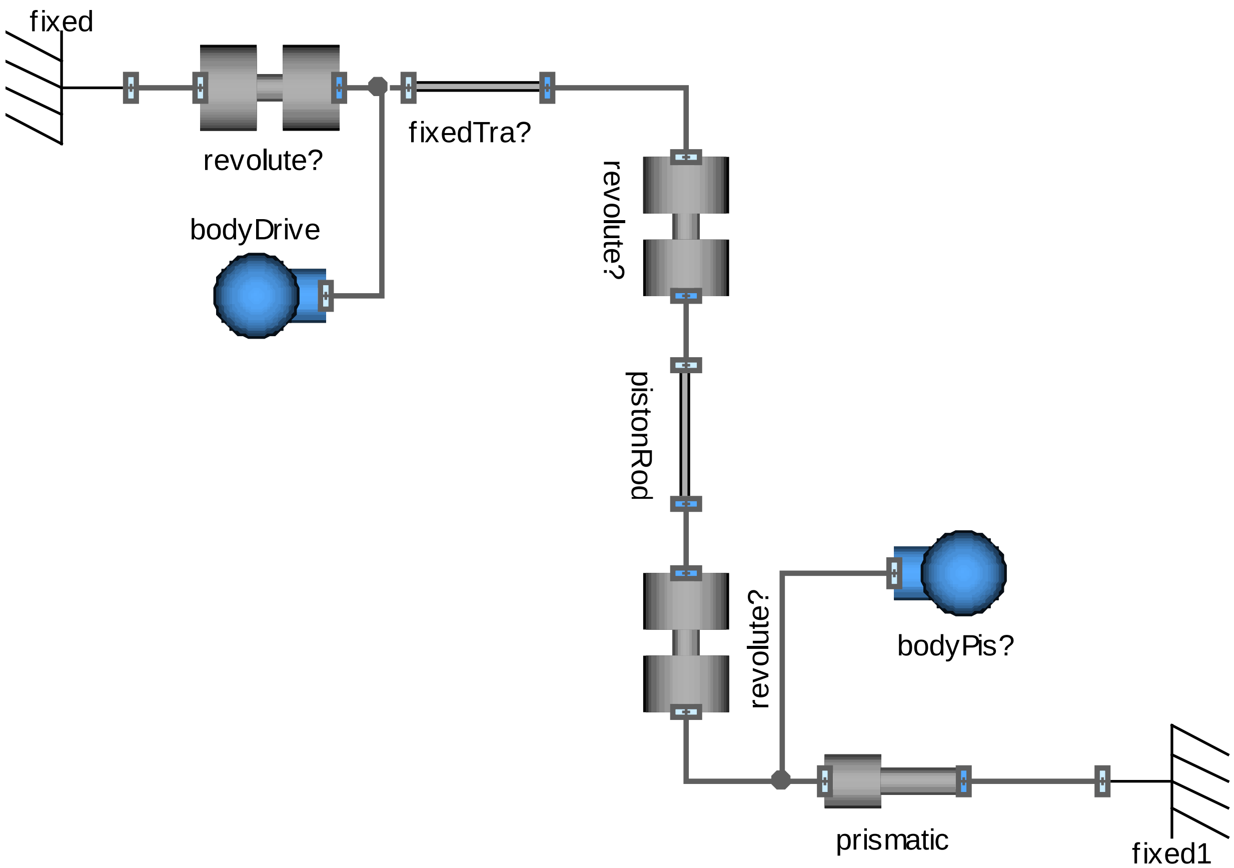
Double Pendulum

- The double pendulum is wonderful example of a chaotic system.

- Although the individual components are simple, the motion of the system cannot be predicted.

Kinematic Loops

Institute of System Dynamics and Control

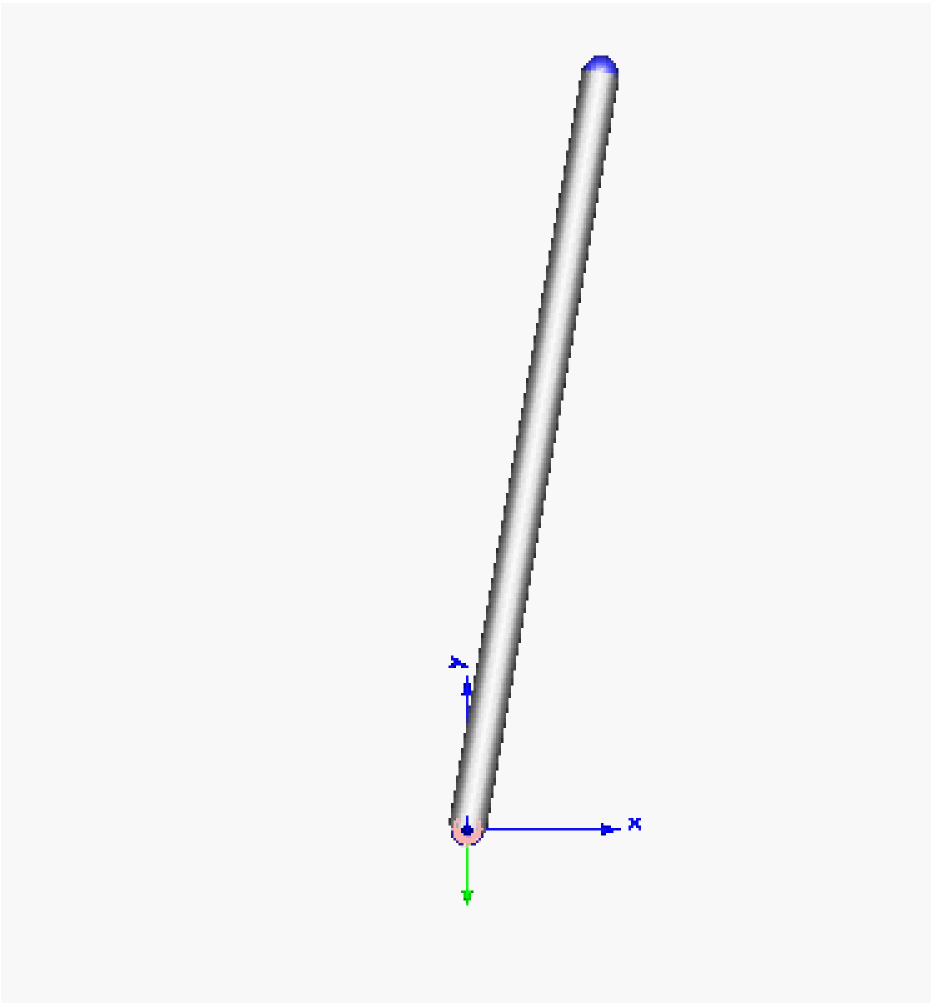


Kinematic Loops

- Discussing kinematic loops we can address the difficulties of initialization

- Also the matter of state-selection can be addressed and it is a good occasion to explain the Pantelides Algorithm and its limitations.

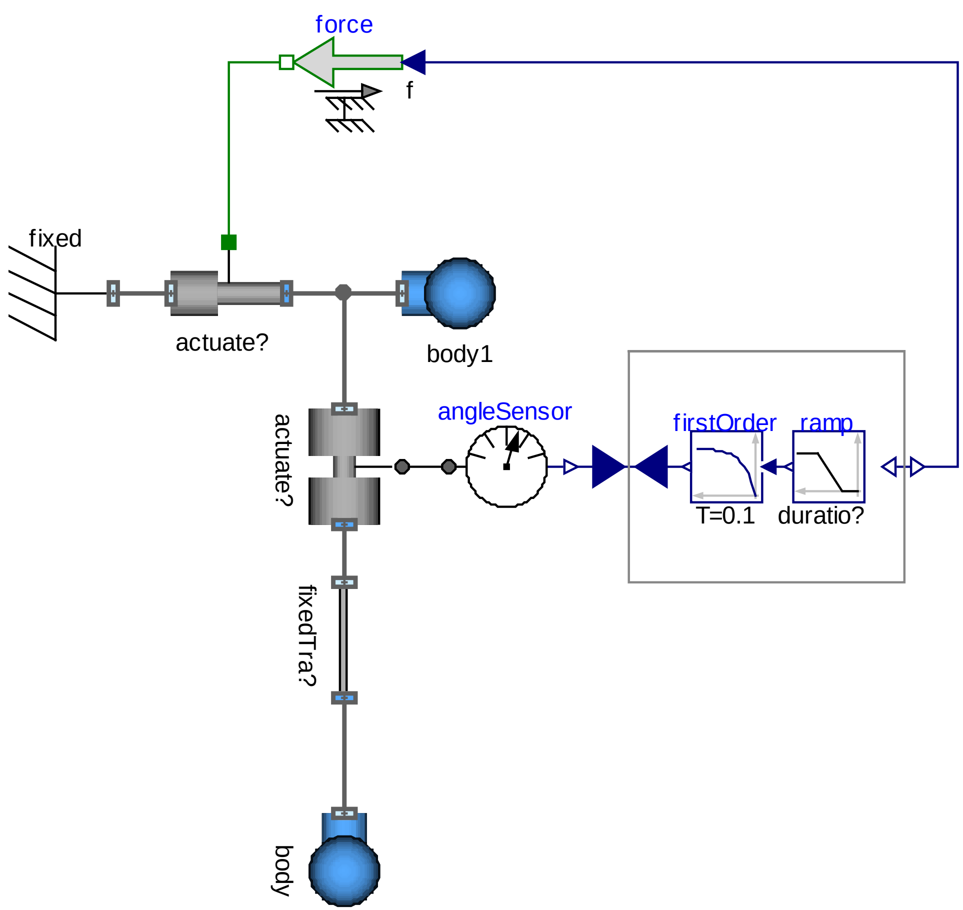
Inverse Pendulum



- The task is to balance a pendulum in upright position.
- On the left you see the simulation result using a PD controller for control.

Inverse Pendulum

Institute of System Dynamics and Control

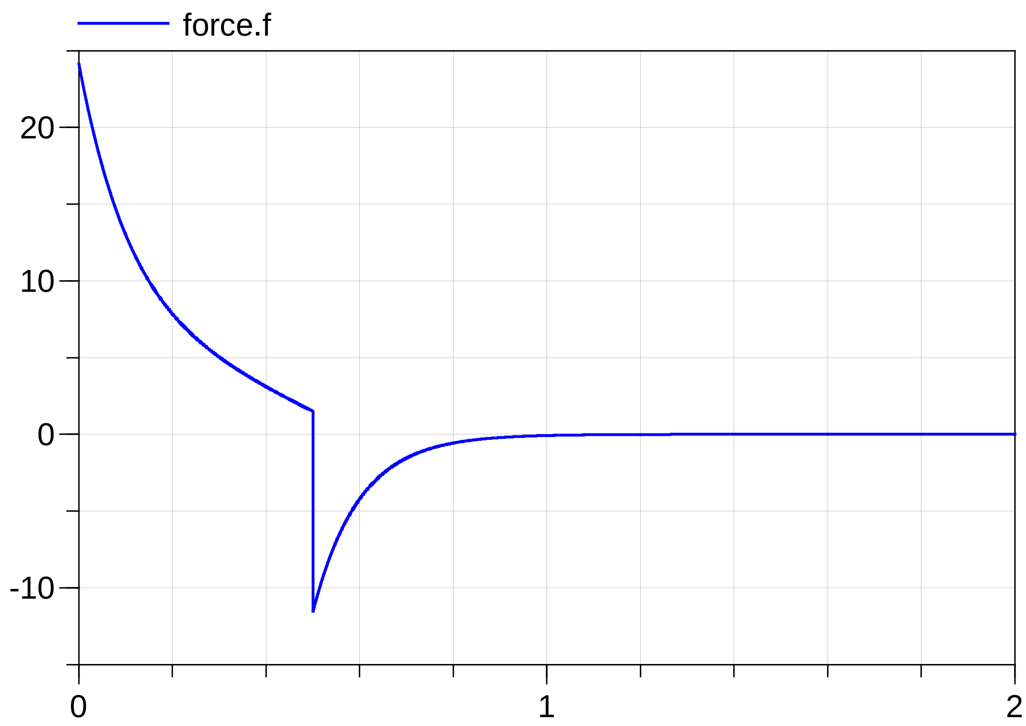


- This is the model of an inverted (upright) pendulum.
- Here, model inversion is applied.
- This means we prescribe the motion and compute the required force
- The motion must be differentiable since this is a higher index system

Inverse Pendulum

Institute of System Dynamics and Control

- The Inverse Pendulum is nice example for control tasks.
- Demonstrating model-inversion reveals to true generality of DAE-based modeling of physical systems.

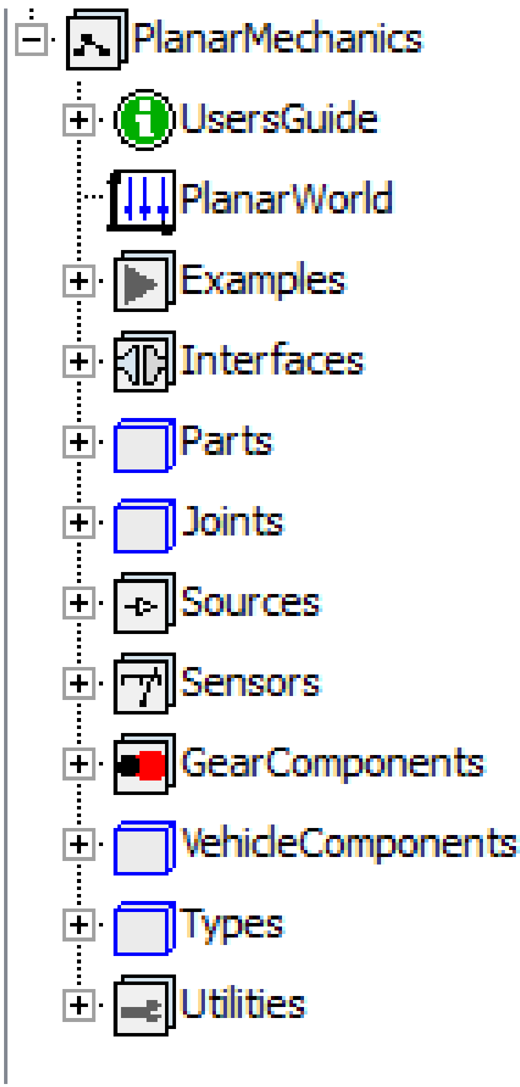


Part 2:

- New features / improvements of the free PlanarMechanics Library

Design of the Library

Institute of System Dynamics and Control

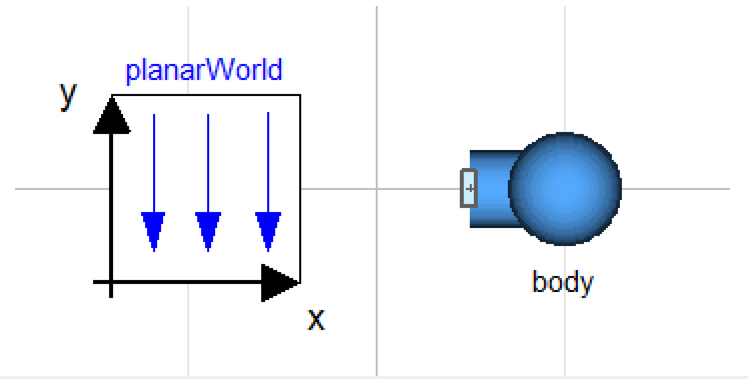


- The library is designed in resemblance to the Multibody library
- There are the classic elements such as
    - Rigid parts
    - Joint elements
    - Sensors
- In addition to these elements, there are simple wheel and tire models and gear components

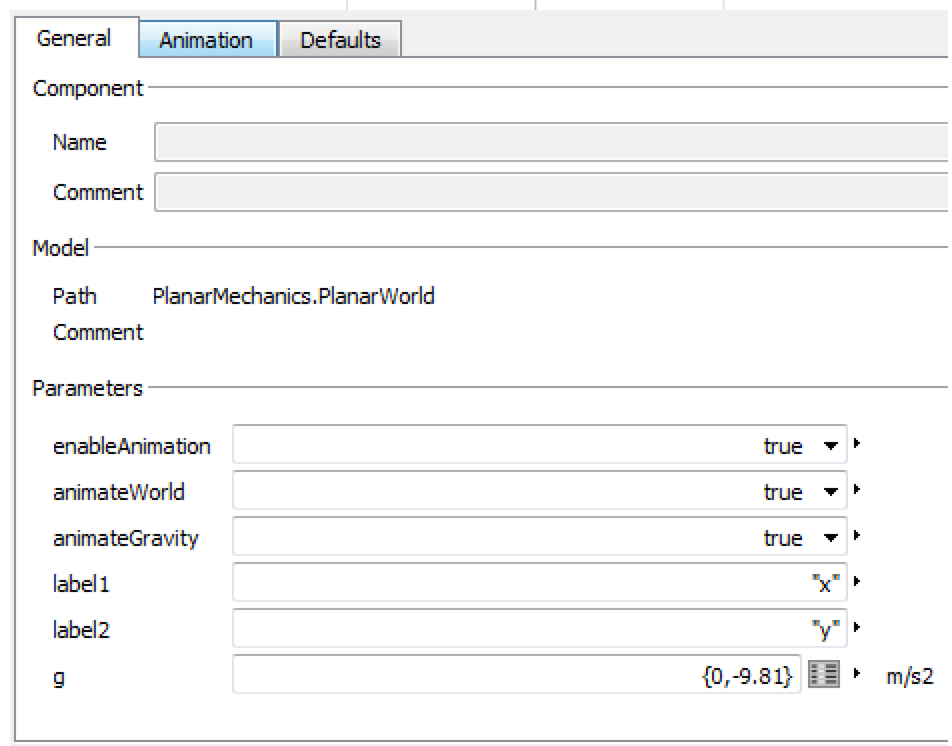
New features

Institute of System Dynamics and Control

+ Planar world model



- Gravity
- Coordinate System
- Defaults for animation

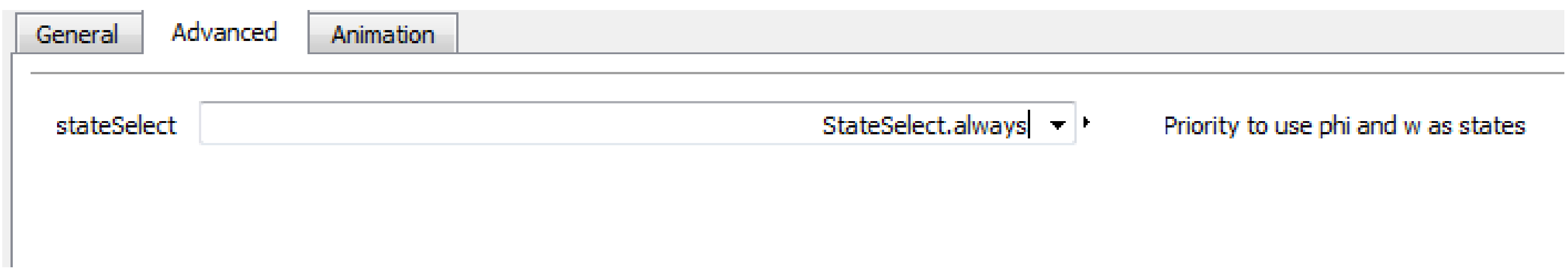


New features

Institute of System Dynamics and Control

+ Planar world model

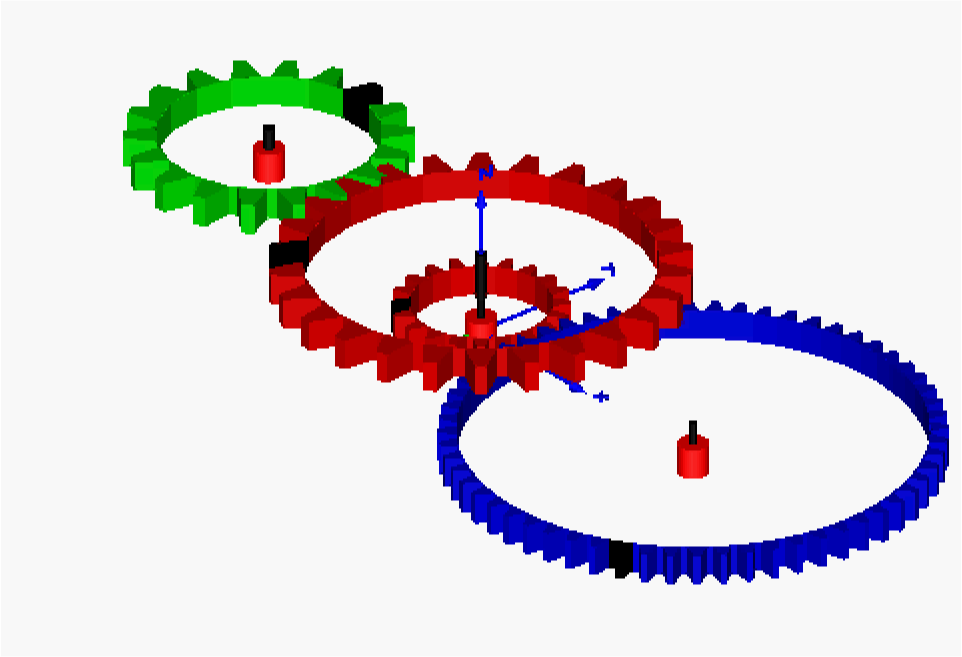
+ Improved initialization



New features

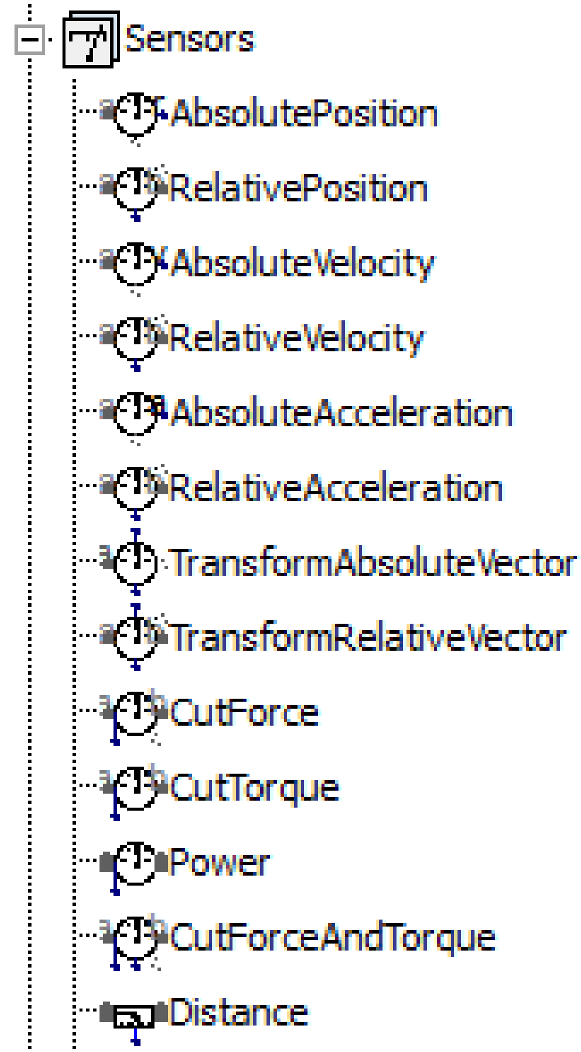
DLR
Institute of System Dynamics and Control

- Planar world model
- Improved initialization
- Improved animation with z-levels



New features

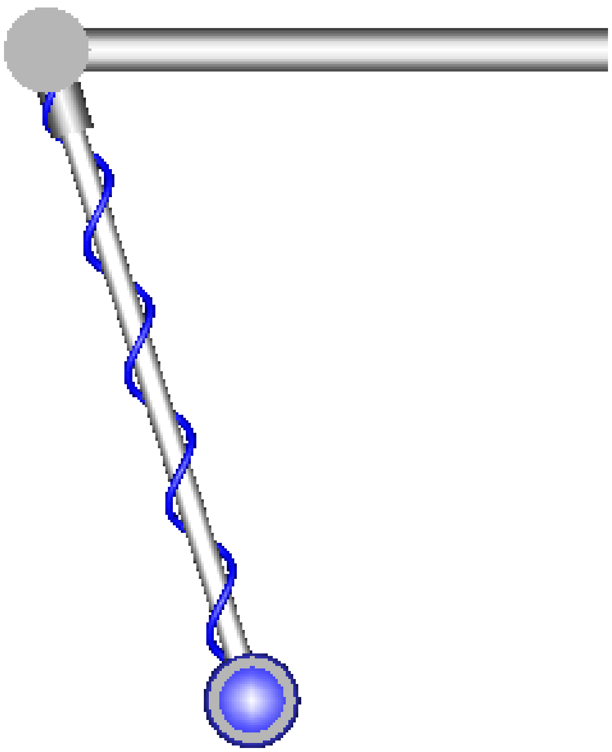
- **Planar world model**
- **Improved initialization**
- **Improved animation with z-levels**
- **Set of sensors**



New features

Institute of System Dynamics and Control

- Planar world model
- Improved initialization
- Improved animation with z-levels
- Set of sensors
- Spring and Damper components



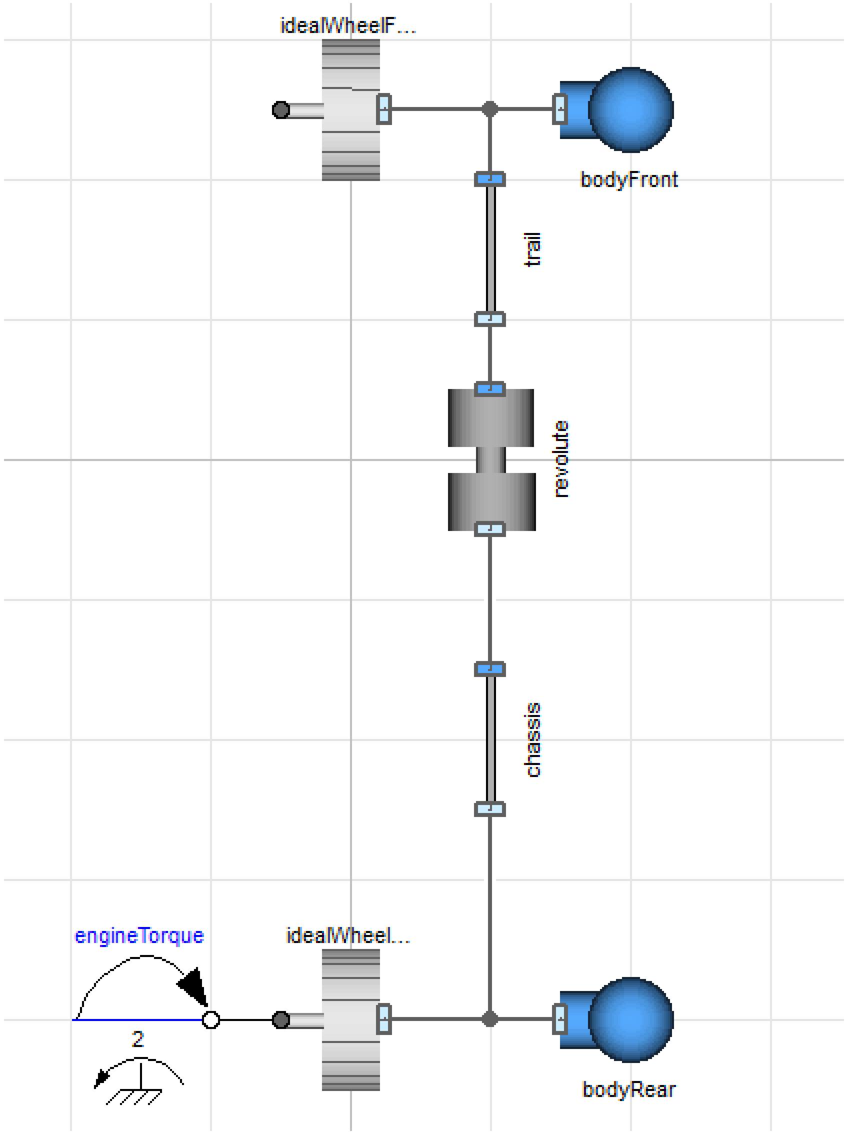
Vehicle Components

Wheel Joints for

- ideal rolling
- dry-friction based rolling
- slip based rolling

Examples:

- Single track model
- Two track model



Vehicle Components

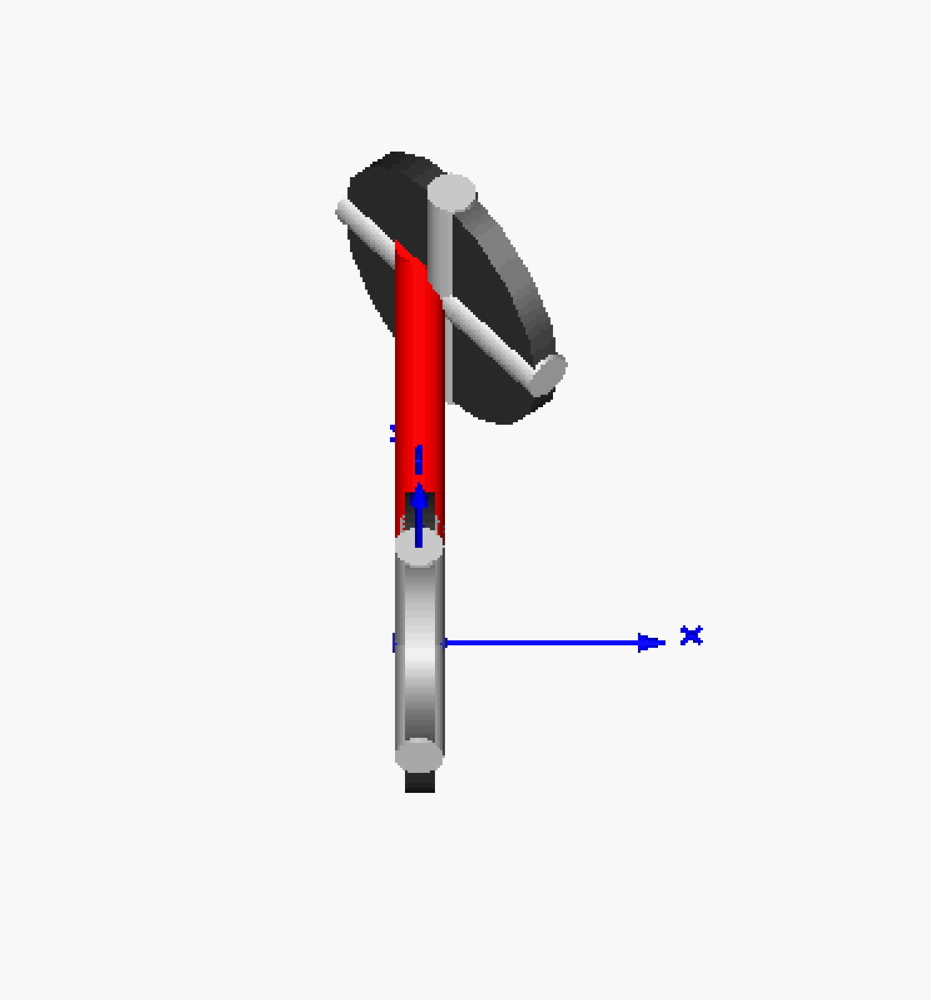
Institute of System Dynamics and Control

Wheel Joints for

- ideal rolling
- dry-friction based rolling
- slip based rolling

Examples:

- Single track model
- Two track model



Gear Components

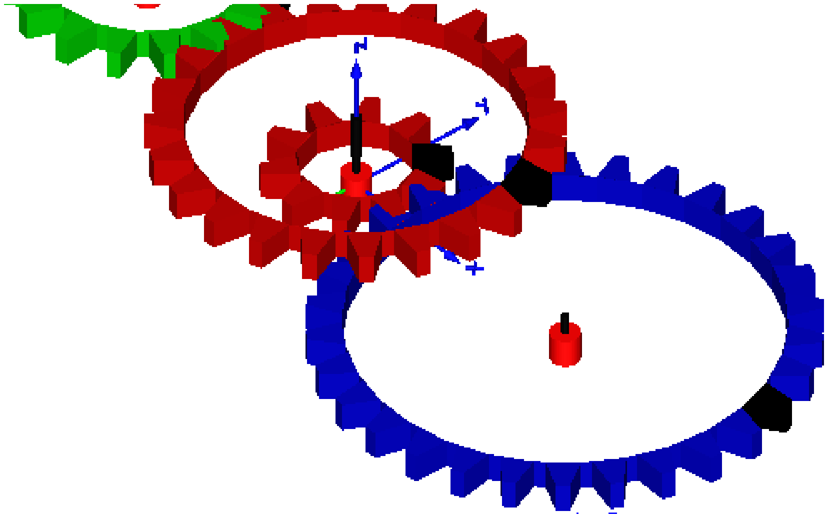
Institute of System Dynamics and Control

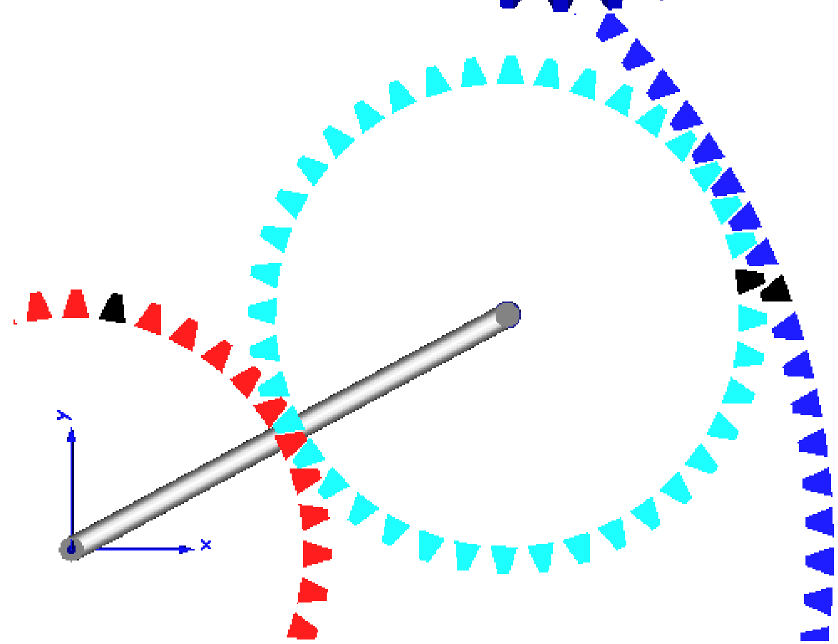
Components for Gears

- Ideal internal contact
- Ideal external contact

Examples:

- Spur Gear
- Planetary Gear

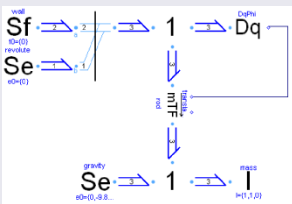




Future Work

2006

Multi BondLib



Part of Ms Thesis

2010

Planar Mechanics for Teaching

Master-Level Course at TUM

2012

Standard Planar Mechanics

9th International Modelica Conference

Presented at Modelica 2012

Wide Tool Support!

2014

Free Planar Mechanics Library

DLR

designed towards use in practical applications

Future

Improve Quality

Increase tool coverage

Goal:

MSL

Questions ?