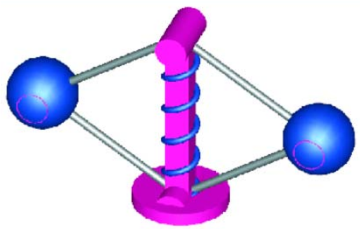
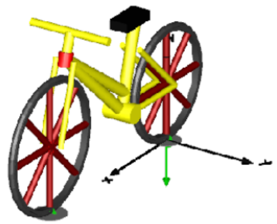
Virtual Physics
Equation-Based Modeling

TUM, November 25, 2014

3D Mechanics, Part II





Dr. Dirk Zimmer

German Aerospace Center (DLR), Robotics and Mechatronics Centre

3D Mechanics

Robotics and Mechatronics Centre

In this lecture, we look at the modeling of 3D mechanical systems.

- 3D mechanical models look superficially just like planar mechanical models. There are additional types of joints, but other than that, there seem to be few surprises.
- Yet, the seemingly similar appearance is deceiving. There are a substantial number of complications that the modeler has to cope with when dealing with 3D mechanics. These are the subject of this lecture.

3D Mechanics

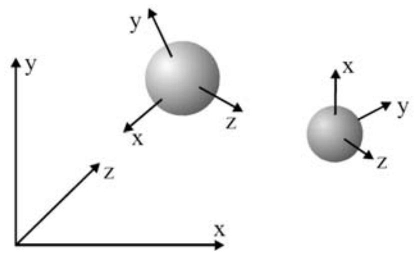
Robotics and Mechatronics Centre

Essentially, there are 3 major difficulties we have to cope with:

1. There are multiple ways to express the orientation of a body in three dimensional space.
2. In planar mechanics, all potential variables could be expressed in one common coordinate system: The inertial system. In 3D-mechanics, such an approach is unfeasible.
3. The set of connector variables contains a redundant set of variables. This causes severe problems for the formulation of kinematic loops.

Fundamental set of equations

Robotics and Mechatronics Centre



- In planar mechanics, all connector variables are resolved w.r.t. the inertial coordinate system.
- In 3D-mechanics, we will refer also to the body system. A coordinate system that is attached to each body.
- Notation The index *0* indicates that a vector is resolved w.r.t. to the ineratial system. The index *body* indicates that is resolved w.r.t. to its body system.

Rotation Matrix

Robotics and Mechatronics Centre

- The rotational matrix can be used to transform between these coordinate systems. For instance

$$\mathbf{R}\boldsymbol{\omega}_0 = \boldsymbol{\omega}_{body}$$

$$\boldsymbol{\omega}_0 = \mathbf{R}^T\boldsymbol{\omega}_{body}$$

- Repetition: The rotational matrix is the one to integrate:

$$\tilde{\omega}_0\mathbf{R} = \mathbf{R}\tilde{\omega}_{body} = \dot{\mathbf{R}}$$

Fundamental set of equations

Robotics and Mechatronics Centre

- The fundamental set of equations can be formulated in the inertial system:

$$\mathbf{f}_0 = m \cdot \mathbf{a}_0$$

$$\mathbf{t}_0 = \mathbf{J}_0\dot{\omega}_0$$

- In planar mechanics, the rotational inertia was represented by a simple scalar I. In 3D mechanics, it is represented by a 3D matrix **J**: the inertia tensor.
- However, $\mathbf{J_0}$ is not a constant during motion since it depends on the orientation of the body.

Fundamental set of equations

Robotics and Mechatronics Centre

- In the body-system, the inertia tensor $\mathbf{J}_{body}$ is constant. Hence we can transform the law into the body system:

$$\mathbf{t}_0 = \frac{d}{dt}\left(\mathbf{R}^T\mathbf{J}_{body}\boldsymbol{\omega}_{body}\right)$$

Fundamental set of equations

Robotics and Mechatronics Centre

- In the body-system, the inertia tensor $\mathbf{J}_{body}$ is constant. Hence we can transform the law into the body system:

$$\mathbf{t}_0 = \frac{d}{dt}\left(\mathbf{R}^T\mathbf{J}_{body}\boldsymbol{\omega}_{body}\right)$$

$$\mathbf{t}_0 = \dot{\mathbf{R}}^T\mathbf{J}_{body}\boldsymbol{\omega}_{body} + \mathbf{R}^T\mathbf{J}_{body}\dot{\boldsymbol{\omega}}_{body}$$

Fundamental set of equations

- In the body-system, the inertia tensor $\mathbf{J}_{body}$ is constant. Hence we can transform the law into the body system:

$$\mathbf{t}_0 = \frac{d}{dt}\left(\mathbf{R}^T \mathbf{J}_{body}\boldsymbol{\omega}_{body}\right)$$

$$\mathbf{t}_0 = \dot{\mathbf{R}}^T \mathbf{J}_{body}\boldsymbol{\omega}_{body} + \mathbf{R}^T \mathbf{J}_{body}\dot{\boldsymbol{\omega}}_{body}$$

$$\mathbf{R}^T \mathbf{t}_{body} = \mathbf{R}^T \tilde{\omega}_{body}\mathbf{J}_{body}\boldsymbol{\omega}_{body} + \mathbf{R}^T \mathbf{J}_{body}\mathbf{z}_{body}$$

Fundamental set of equations

- In the body-system, the inertia tensor $\mathbf{J}_{body}$ is constant. Hence we can transform the law into the body system:

$$\mathbf{t}_0 = \frac{d}{dt}\left(\mathbf{R}^T \mathbf{J}_{body}\boldsymbol{\omega}_{body}\right)$$

$$\mathbf{t}_0 = \dot{\mathbf{R}}^T \mathbf{J}_{body}\boldsymbol{\omega}_{body} + \mathbf{R}^T \mathbf{J}_{body}\dot{\boldsymbol{\omega}}_{body}$$

$$\mathbf{R}^T \mathbf{t}_{body} = \mathbf{R}^T \tilde{\omega}_{body}\mathbf{J}_{body}\boldsymbol{\omega}_{body} + \mathbf{R}^T \mathbf{J}_{body}\mathbf{z}_{body}$$

$$\mathbf{t}_{body} = \boldsymbol{\omega}_{body} \times \mathbf{J}_{body}\boldsymbol{\omega}_{body} + \mathbf{J}_{body}\mathbf{z}_{body}$$

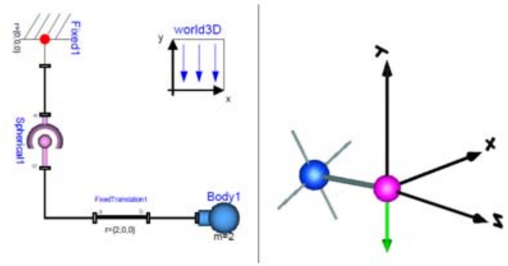
Fundamental set of equations

- In the body-system, the inertia tensor $\mathbf{J}_{body}$ is constant. Hence we can transform the law into the body system:
- An additional term for the torque occurs: The gyroscopic torque.
- This torque is a pseudo-torque that resulted out of the transformation into the body system.

$$\mathbf{t}_{body} = \boxed{\boldsymbol{\omega}_{body} \times \mathbf{J}_{body}\boldsymbol{\omega}_{body}} + \mathbf{J}_{body}\mathbf{z}_{body}$$

Selection of Method

- We have observed the (highly non-intuitive) behavior of the gyroscopic effect, already last lecture:



Connector Design

- The translational components can be more conveniently described in the inertial system.
- The rotational components are preferably resolved w.r.t. to the body system.

Connector Design

- In the MultiBody library, the connector is designed as follows:
- Vectors and matrices are supported natively by Modelica and used for the connector variables.

```
connector Frame

  SI.Position r_0[3];
  Real T[3, 3];
  SI.AngularVelocity w[3]

  flow SI.Force f[3];
  flow SI.Torque t[3];

end Frame;
```


Connector Design

- In the MultiBody library, the connector is designed as follows:
- Resolved w.r.t. to the inertial system:

  r_0, T
- Resolved w.r.t. to the body system (T):

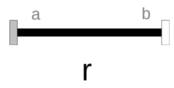
  w, t, and f (why ever...)

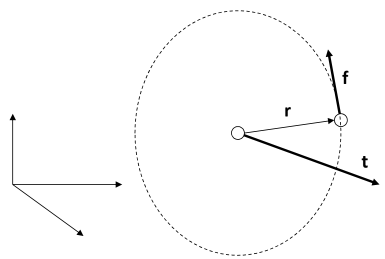
```
connector Frame

  SI.Position r_0[3];
  Real T[3, 3];
  SI.AngularVelocity w[3]

  flow SI.Force f[3];
  flow SI.Torque t[3];

end Frame;
```


An example component

Let us look at the fixed translation component:

- It essentially represents the lever principle.



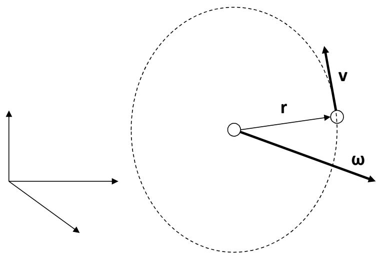


t = r x f

An example component

Let us look at the fixed translation component:

- It essentially represents the lever principle.



$$\mathbf{v} = \mathbf{r} \times \boldsymbol{\omega}$$

An example component

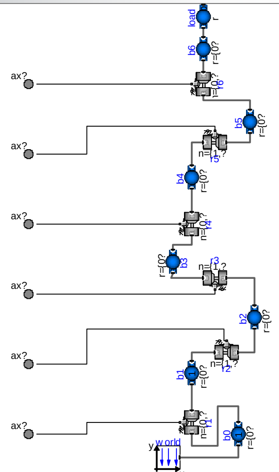
Let us look at the fixed translation component:

- It essentially represents the lever principle.

```
model FixedTranslation
  parameter SI.Position r[3] = {0,0,0};

  frame_b.r_0 = frame_a.r_0 + transpose(frame_a.T)*r;
  frame_b.T = frame_a.T;
  frame_b.w = frame_a.w;

  zeros(3) = frame_a.f + frame_b.f;
  zeros(3) = frame_a.t + frame_b.t + cross(r, frame_b.f);

end FixedTranslation;
```

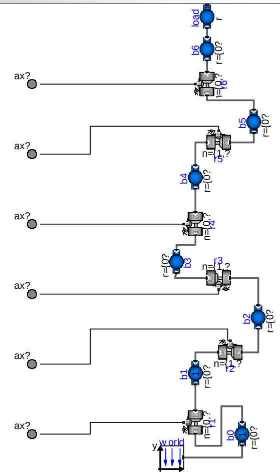

An example system: The robot

- Here, the multibody components are used to assemble a robot.



An Example System: The Robot

- Here, the multibody components are used to assemble a robot.
- It essentially consists out of fixed translations combined with body parts and actuated revolute joints.



Redundant Connector Variables

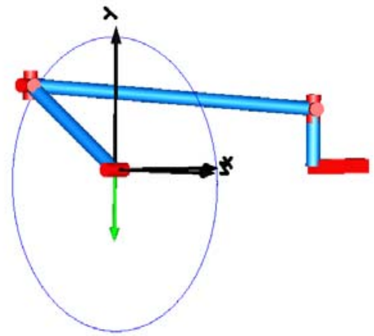
Robotics and Mechatronics Centre

- The potential variables of the Multibody connector are highly redundant.
- Only 3 variables are sufficient to describe the 3D-rotation.
- But the connector contains 3*3 + 3 = 12 potential variables for the rotational part.

```
connector Frame

  SI.Position r_0[3];
  Real T[3, 3];
  SI.AngularVelocity w[3]

  flow SI.Force f[3];
  flow SI.Torque t[3];

end Frame;
```

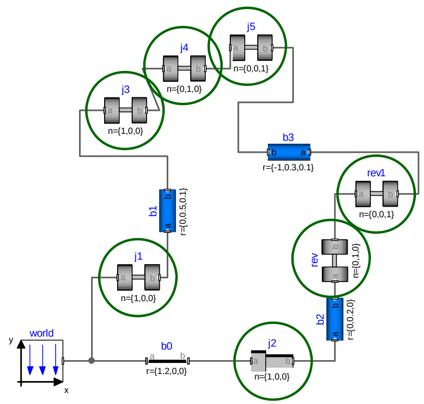

Kinematic Loops

Robotics and Mechatronics Centre



Kinematic Loops

Robotics and Mechatronics Centre

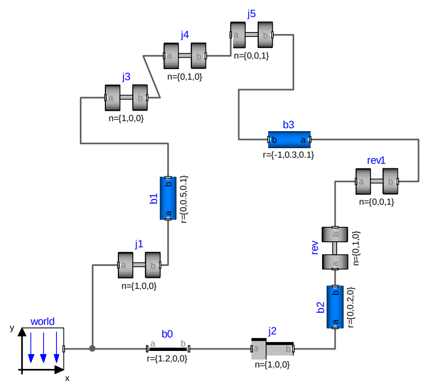


7 degrees of freedom – 6 constraint equations = 1 degree of freedom

Kinematic Loops

Robotics and Mechatronics Centre

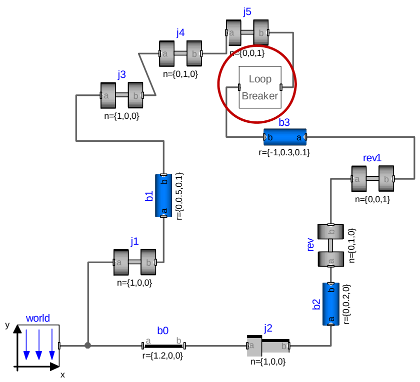
- This redundancy causes severe problems in case of kinematic loops.
- Closing a kinematic loop establishes 6 constraint equations.
- But the redundant connector set leads to 15 constraint equations (these are 9 too many).



Kinematic Loops

Robotics and Mechatronics Centre

- In the “old days”, the loops had to manually closed with the aid of a loop-breaker.
- The loop-breaker is a model that contains just the necessary 6 constraint equations
  (and the balance of force and torque, naturally)



Kinematic Loops

Robotics and Mechatronics Centre

- In the “old days”, the loops had to manually closed with the aid of a loop-breaker.
- The loop-breaker is a model that contains just the necessary 6 constraint equations
  (and the balance of force and torque, naturally)

```
model LoopBreaker
  Interfaces.Frame_a frame_a;
  Interfaces.Frame_b frame_b;

equation
  frame_a.r_0 frame_b.r_0;
  cross(frame_a.T[1, :], frame_a.T[2, :])*frame_b.T[2, :] = 0;
  -cross(frame_a.T[1, :],frame_a.T[2, :])*frame_b.T[1, :] = 0;
 frame_a.T[2, :]*frame_b.T[1, :] = 0
 frame_a.f + frame_b.f = zeros(3);
 frame_a.t + frame_b.t = zeros(3);
end LoopBreaker
```


Kinematic Loops

Robotics and Mechatronics Centre

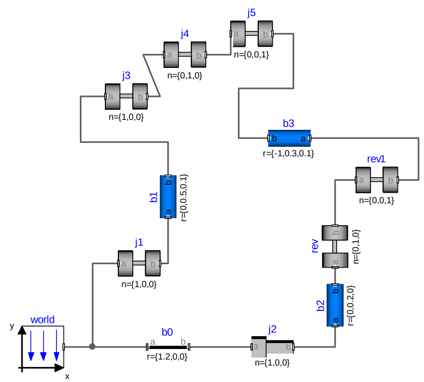
- In the “old days”, the loops had to manually closed with the aid of a loop-breaker.
- The loop-breaker is a model that contains just the necessary 6 constraint equations
  (and the balance of force and torque, naturally)

```
model LoopBreaker
  Interfaces.Frame_a frame_a;
  Interfaces.Frame_b frame_b;

equation
  frame_a.r_0 frame_b.r_0;
  cross(frame_a.T[1, :], frame_a.T[2, :])*frame_b.T[2, :] = 0;
  -cross(frame_a.T[1, :],frame_a.T[2, :])*frame_b.T[1, :] = 0;
 frame_a.T[2, :]*frame_b.T[1, :] = 0
 frame_a.f + frame_b.f = zeros(3);
 frame_a.t + frame_b.t = zeros(3);
end LoopBreaker
```


Kinematic Loops

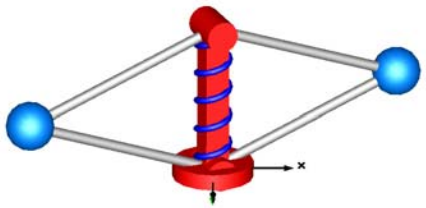
Robotics and Mechatronics Centre

- Nowadays, the loop breaker is not necessary anymore.
- The process has been automated (by introducing a whole new set of irritating language constructs).



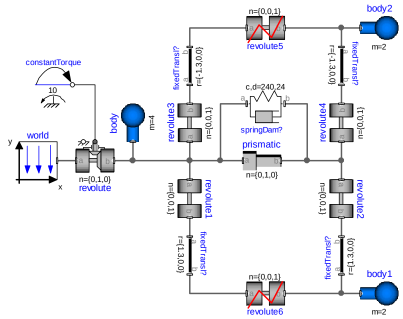
Planar kinematic loops

- Another special case are planar kinematic loops within 3D mechanics.
- Even if we apply the correct set of constraint equations, we get a singular system.
- Let us look at an example…



Planar kinematic loops

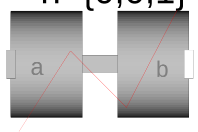
- The problem is the following:
- There are two planar closed kinematic loops each defined by three revolute joints and a prismatic joint.
- Two revolute joints with the same rotation axis suffice to restrict the freedom of motion to a single axis. The constraint of the third revolute joint is therefore superfluous, which leads to an additional redundancy

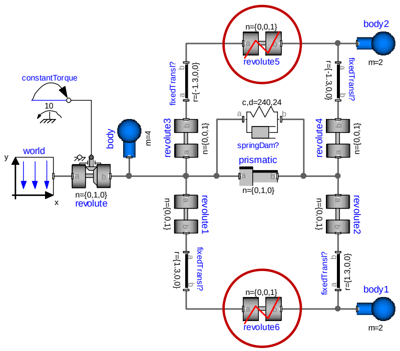


Planar kinematic loops

- To this end, there is a special revolute joint to cut the planar loop.

n={0,0,1}





Questions ?