Lecture 15: Modelica/Dymola: The Multibody library, Friction

- Newton-Euler equations of motion
- Software
  - Dymola and the Modelica.Mechanics.Multibody library
- Friction

Book: 7.3, 5

Newton-Euler equations of motion

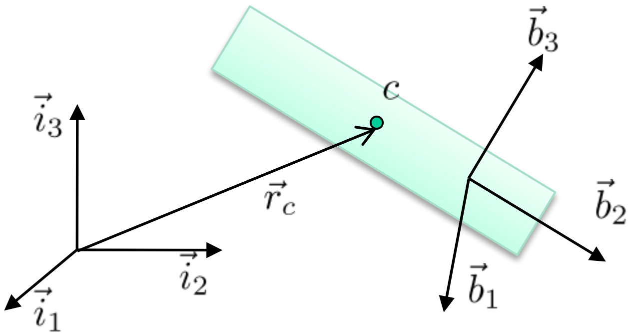
- Newton's law (for particle *k*)

$$m_k \vec{a}_k = \vec{F}^{(r)}$$

- Newton-Euler EoM for rigid bodies:
  - Integrate Newton's law over body, define center of mass
  - Define torque/moment and angular momentum to handle forces that give rotation about center of mass
  - Define inertia dyadic/matrix

$$\vec{F}_{bc} = m\vec{a}_c$$

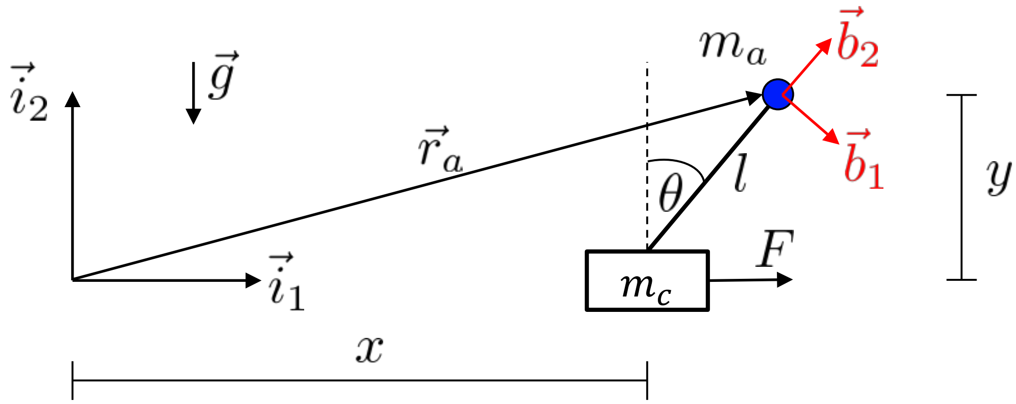
$$\vec{T}_{bc} = \vec{M}_{b/c} \cdot \vec{\alpha}_{ib} + \vec{\omega}_{ib} \times \left(\vec{M}_{b/c} \cdot \vec{\omega}_{ib} \right)$$



(Here: Referenced to center of mass)

- Implemented in e.g. Dymola (Modelica.Multibody library)

Example: Inverted pendulum - kinematics



$$\vec{r}_a = (x + l\sin\theta)\,\vec{i}_1 + l\cos\theta\,\vec{i}_2$$

$$\vec{i}_1 = \vec{b}_1 \cos\theta + \sin\theta\,\vec{b}_2$$

$$\vec{i}_2 = -\vec{b}_1 \sin\theta + \cos\theta\,\vec{b}_2$$

$$\vec{b}_1 = \vec{i}_1 \bigcirc + \bigcirc\,\vec{i}_2$$

$$\vec{b}_2 = \vec{i}_1 \bigcirc + \vec{i}_2 \bigcirc$$

$$\begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} \times \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

$\vec{r}_a$: position of m_a

$$\vec{r}_a = x \cdot \vec{i}_1 + l\,\vec{b}_2$$

$$\vec{v}_a = \frac{{}^i d}{dt}\vec{r}_a = \frac{{}^i d}{dt}(x \cdot \vec{i}_1) + \frac{{}^i d}{dt}(l\,\vec{b}_2)$$

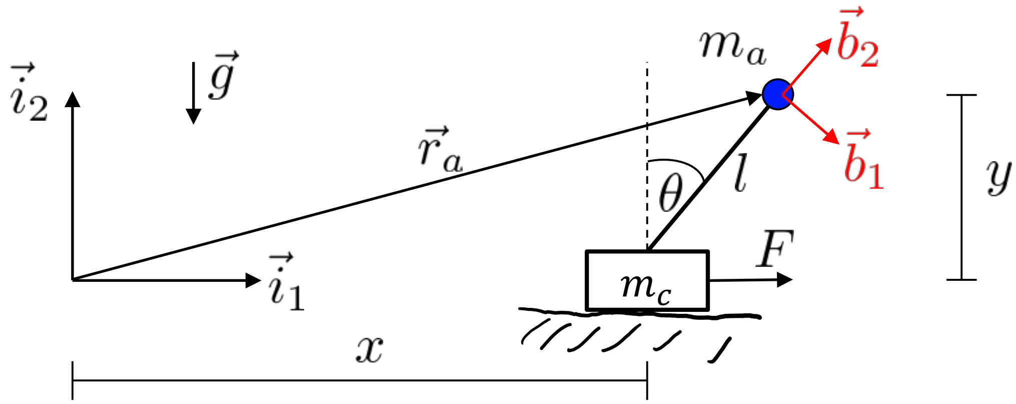
$$= \dot{x}\,\vec{i}_1 + \frac{{}^b d}{dt}(l\,\vec{b}_2) + \vec{\omega}_{ib} \times l\,\vec{b}_2$$

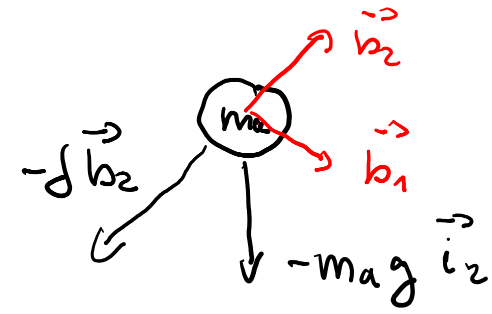
$$= \dot{x}\,\vec{i}_1 + l\dot{\theta}\,\vec{b}_1$$

$$\vec{a}_a = \ddot{x}\,\vec{i}_1 + l\ddot{\theta}\,\vec{b}_1 + \vec{\omega}_{ib} \times l\dot{\theta}\,\vec{b}_1$$

$$= \ddot{x}\,\vec{i}_1 + \underbrace{l\ddot{\theta}\,\vec{b}_1}_{\text{tangential}} - \underbrace{l\dot{\theta}^2\,\vec{b}_2}_{\text{centripetal}}$$

Example: Inverted pendulum – kinetics I



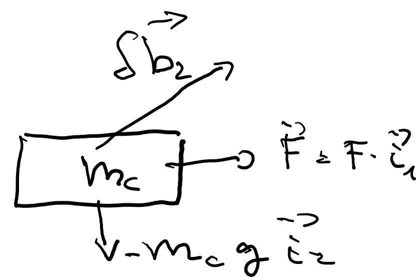


Newton's law m_a

$$(1) \quad m_a \vec{a}_a = -m_a g \vec{i}_2 - \delta \vec{b}_2$$

$$(1)\cdot \vec{i}_1 \quad m_a \ddot{x} + m_a l \ddot{\theta} \cos\theta - m_a l \dot{\theta}^2 \sin\theta = -\delta \sin\theta$$

$$(1)\cdot \vec{b}_2 \quad m_a \ddot{x} \cos\theta + m l \ddot{\theta} = m_a g \sin\theta$$



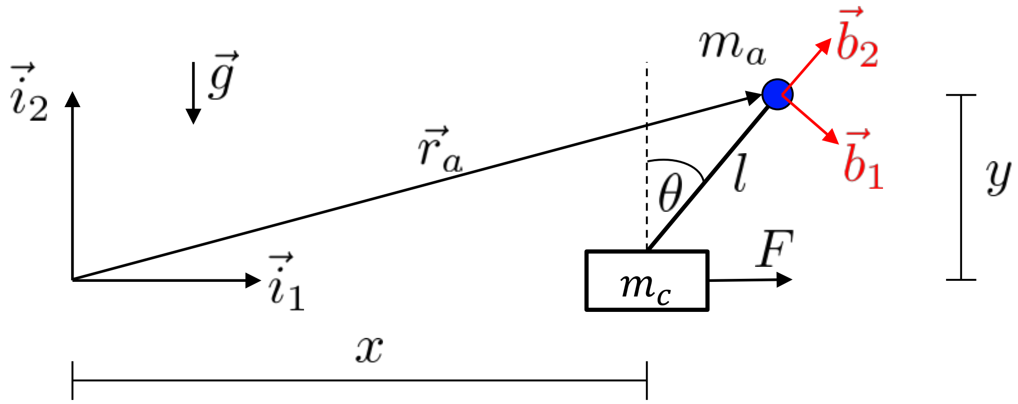
Newton's law m_c

$$(2) \quad m_c \vec{a}_c = F \vec{i}_1 + \delta \vec{b}_2 - m_c g \vec{i}_2$$

$$(2)\cdot \vec{i}_1 \quad m_c \ddot{x} = F + \delta \sin\theta$$

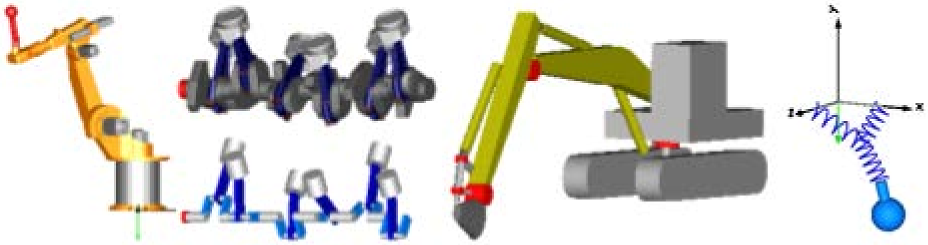
$$\left[(2)\cdot i_2 \quad 0 = \delta \cos\theta - m_c g \right]$$

Example: Inverted pendulum – kinetics II



$$(x) + (x)(x) \qquad (m_c + m_a)\ddot{x} + m_a l \ddot{\theta} \cos\theta - m_a l \dot{\theta}^2 \sin\theta = F$$

$$m_a g \sin\theta = m l \ddot{\theta} + m_a \ddot{x} \cos\theta$$

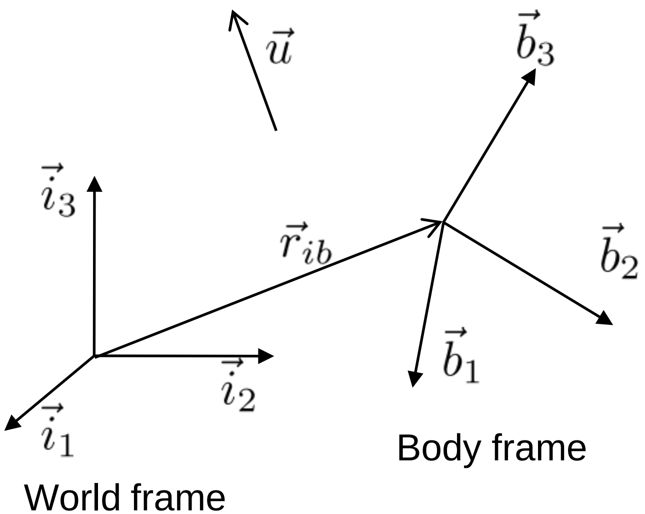


Modelica Multibody introduction

Adapted from slides by Andreas Heckmann, DLR

Modelica Multibody: Orientation

- Orientation and position of coordinate systems (*frames*)

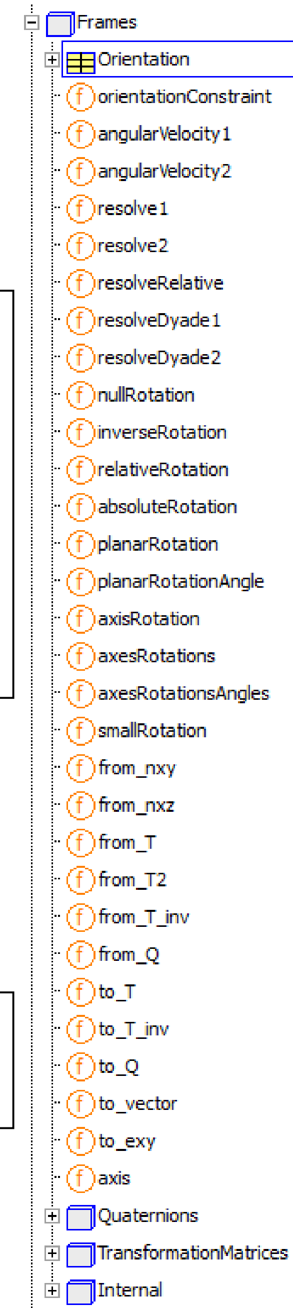



```
model ...
 import Modelica.Mechanics.MultiBody.Frames;
 Frames.Orientation Rib;
 Real[3] ui "vector u resolved in frame i";
 Real[3] ub "vector u resolved in frame b";
 ...
equation
 ...
 ui = Frames.resolve1(Rib, ub); // ui = Rib*ub
 ub = Frames.resolve2(Rib, ui); // ub = Rib'*ui
```

- Orientation object $\mathbf{R}_b^i$
  - Describes orientation of system *b* wrt *i* (transforms from *b* to *i*)
  - Contains:

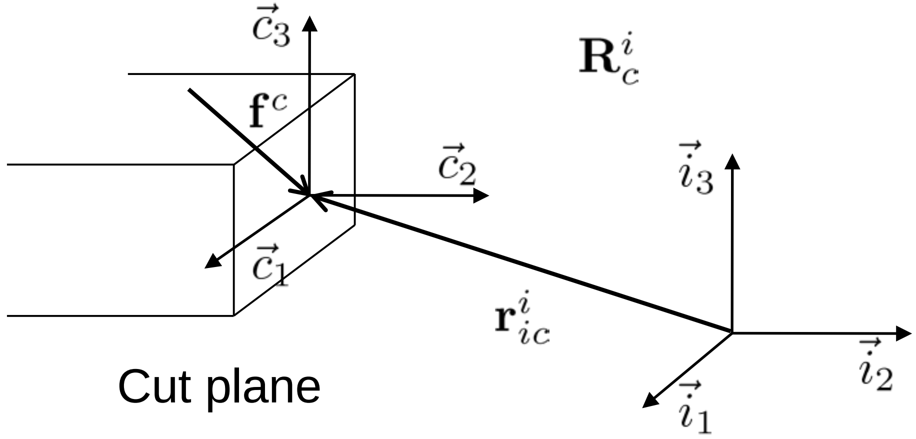
```
Real T[3, 3] "Transformation matrix from world frame to local frame";
SI.AngularVelocity w[3]
   "Absolute angular velocity of local frame, resolved in local frame";
```

  - Can be specified using Euler angles or Euler parameters/quaternions
- Many functions to operate on orientation objects



Modelica Multibody: Connectors I

- Connectors: To connect different rigid bodies
  - Position is resolved in world frame
  - Forces and torques are resolved in local frame



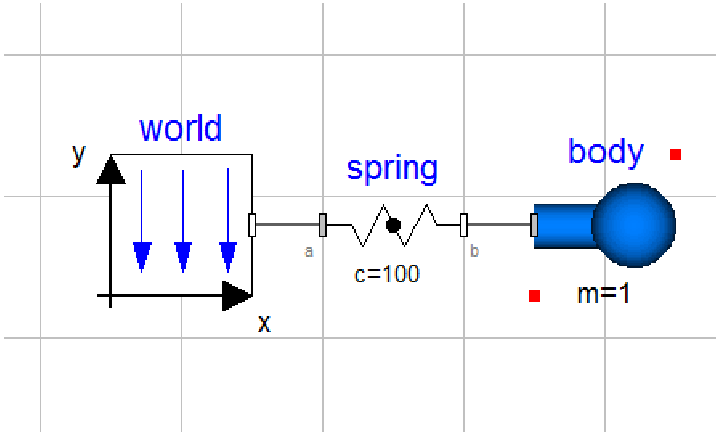
"No flow" variables

```
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";
  flow SI.Torque t[3] "Cut-torque resolved in connector frame";
end Frame;
```

"Flow" variables

Modelica Multibody: Connectors II

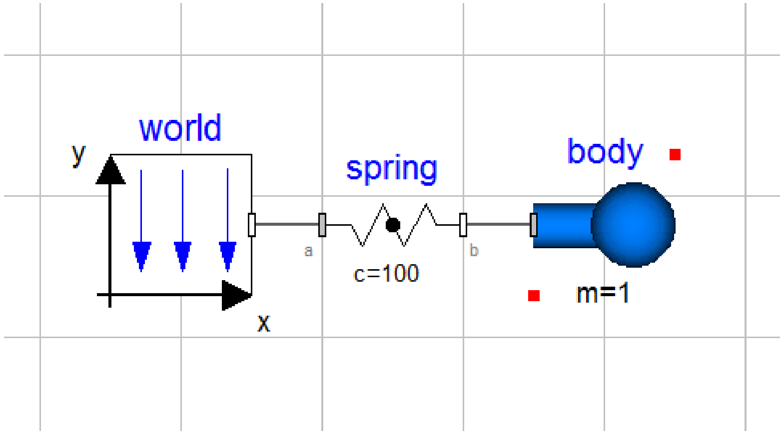
```
model SpringMass
 inner Modelica.Mechanics.MultiBody.World world;
 Modelica.Mechanics.MultiBody.Parts.Body body(
    m=1,
    r_CM={0,1,0}, // In frame a
    r_0(fixed=true, start={0,0.5,0})); // In world frame
Modelica.Mechanics.MultiBody.Forces.Spring spring(c=100);
equation
 connect(spring.frame_a, world.frame_b);
 connect(spring.frame_b, body.frame_a);
end SpringMass;
```



- Connection rules
  - *Non-flow* variables set equal (that is: frames coincides)
  - *Flow* variables sum to zero (Newton’s third law)

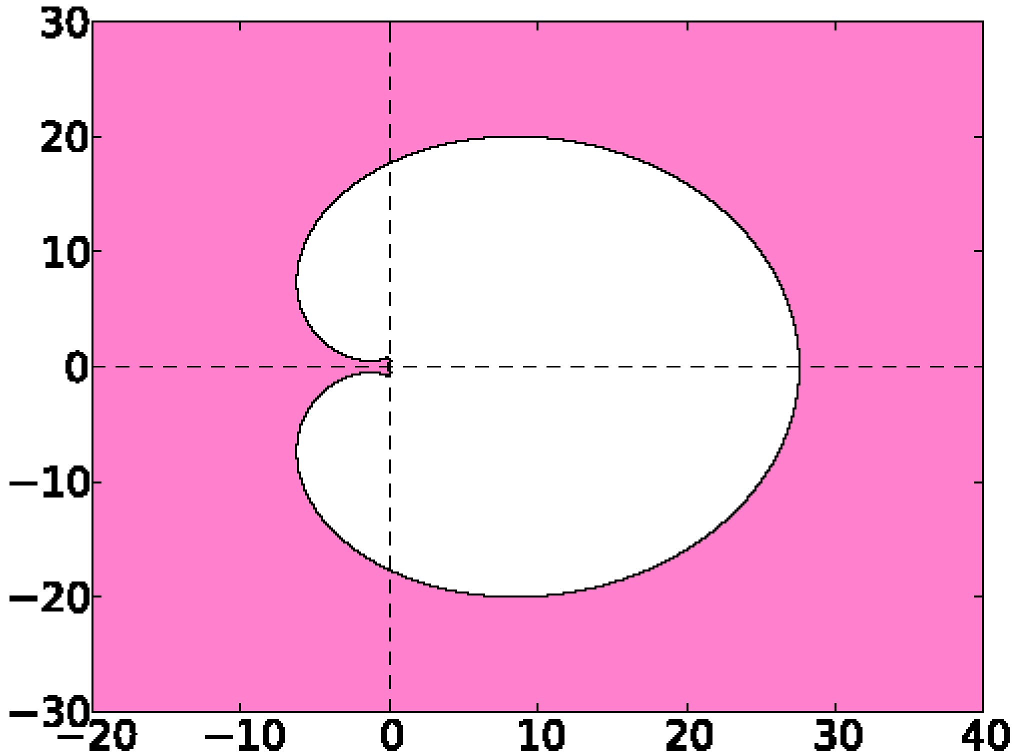
Modelica.Multibody: Generic body component

- Make SpringMass in Dymola



- Show
  - Parameters (mass, r_cm = (0,-0.5,0), Inertia matrix)
  - Initial values
  - Euler angles

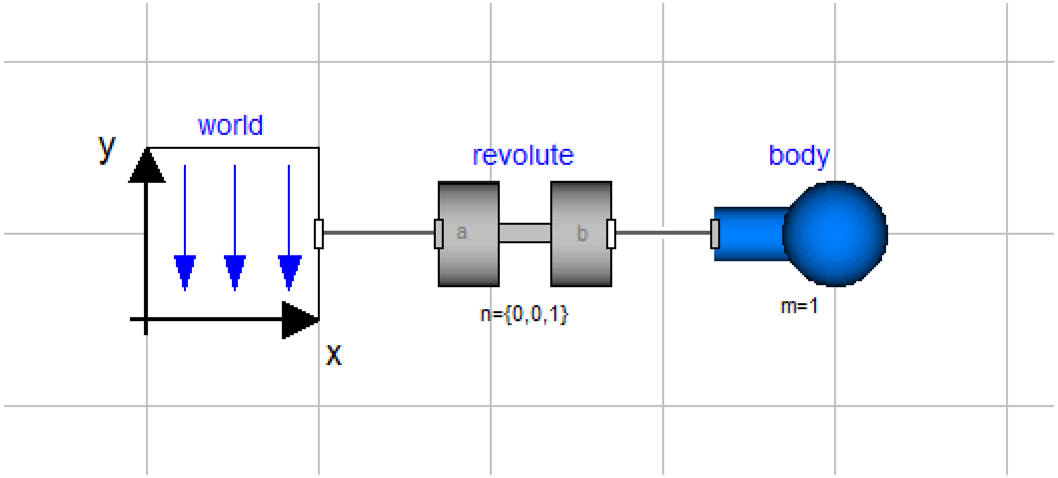
Stability Region BDF



BDF 6

Modelica.Multibody: Rotations

- Make simple pendulum

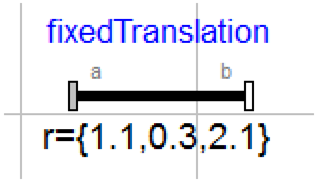


- Show body.frame_a.R (rotation object)

Modelica Multibody: Kinematics

- Equations inside the component provide relations between the connector variables <u>on position level</u>

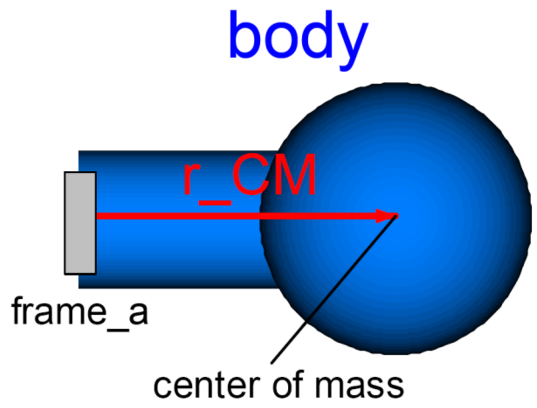
- Example: MultiBody.Parts.FixedTranslation
  - Fixed translation of frame_b with respect to frame_a




```
model FixedTranslation
  "Fixed translation of frame_b with respect to frame_a”
  ...
equation

  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);
end FixedTranslation;
```

- Dymola differentiates these equations twice for (velocity and) accelerations

Modelica Multibody: Kinetics



- Newton-Euler equations
  - Accelerations

$$\vec{a}_p = \vec{a}_o + \vec{\alpha}_{ib} \times \vec{r} + \vec{\omega}_{ib} \times (\vec{\omega}_{ib} \times \vec{r}), \quad \vec{r} \text{ fixed.}$$

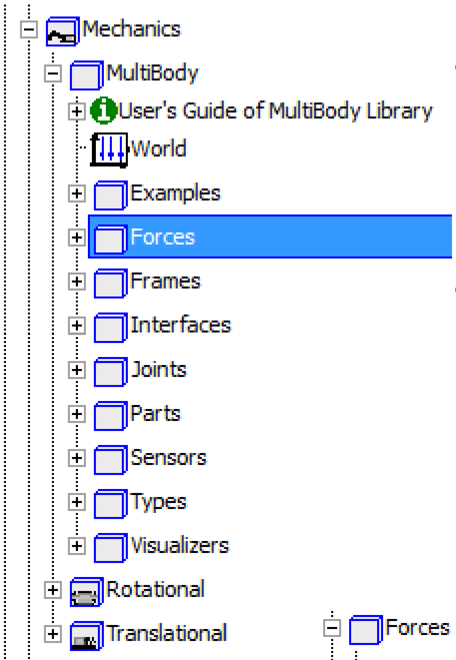
  - Kinetics

$$\vec{F}_{bc} = m\vec{a}_c$$

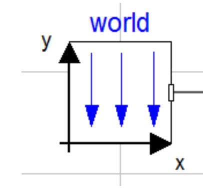
$$\vec{T}_{bc} = \vec{M}_{b/c} \cdot \vec{\alpha}_{ib} + \vec{\omega}_{ib} \times (\vec{M}_{b/c} \cdot \vec{r})$$

```
model body
  "Rigid body with mass, inertia tensor and one frame connector (12 potential states)"
  ...
equation
  // translational kinematic differential equations
  v_0 = der(frame_a.r_0);                       // r_0, v_0 resolved in world frame
  a_a = Frames.resolve2(frame_a.R,der(v_0)); // a_a resolved in frame_a

  // rotational kinematic differential equations
  w_a = 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 - g_a);
            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);
end body;
```

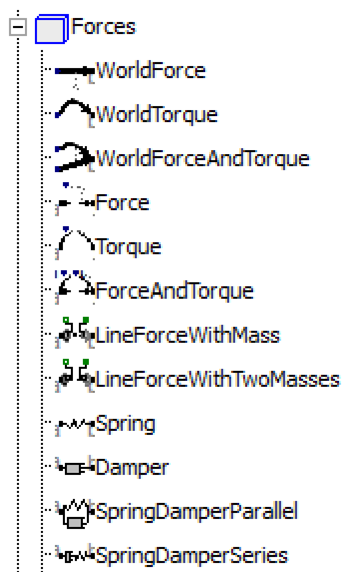
Modelica Multibody: Elementary components I



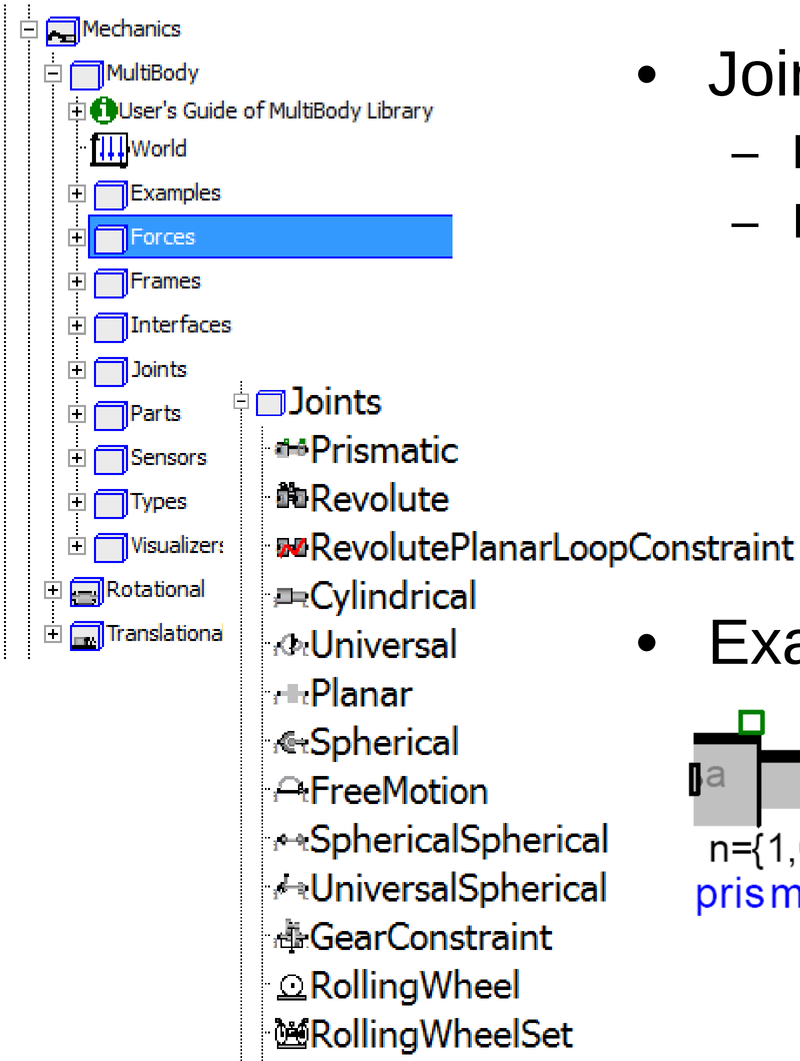
- Modelica.Mechanics.Multibody.World
  - Defines inertial frame, gravity, animation defaults



- Modelica.Mechanics.MultiBody.Forces
  - External forces and torques, resolved in body- or inertial frame
  - Interface to Real input functions
  - Several spring/damper configurations

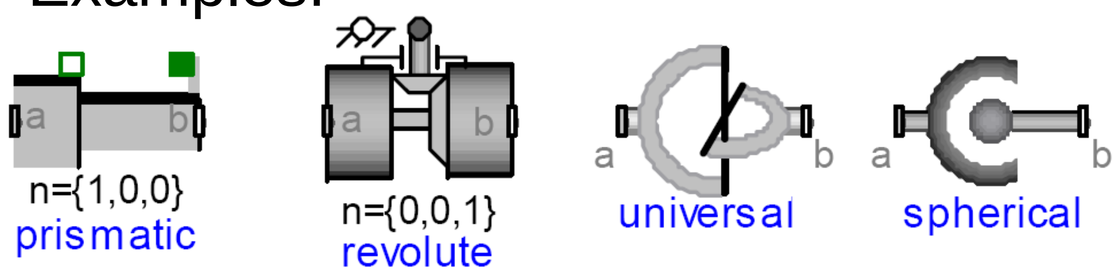


Modelica Multibody: Elementary components II

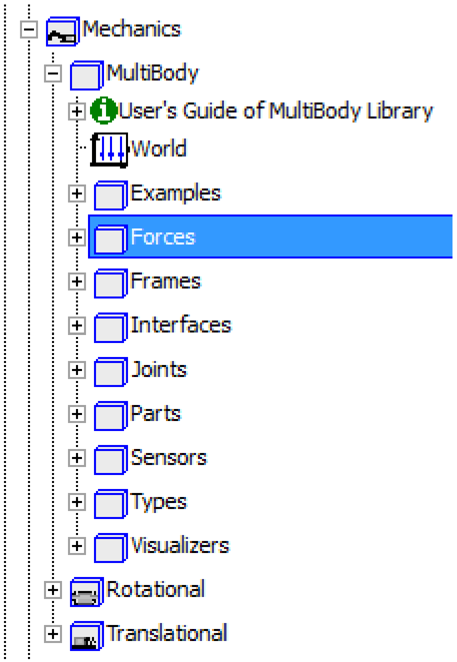


- Joints
  - Define specific degrees of freedom
  - Interface to 1D mechanics
    - (rotational/translational)

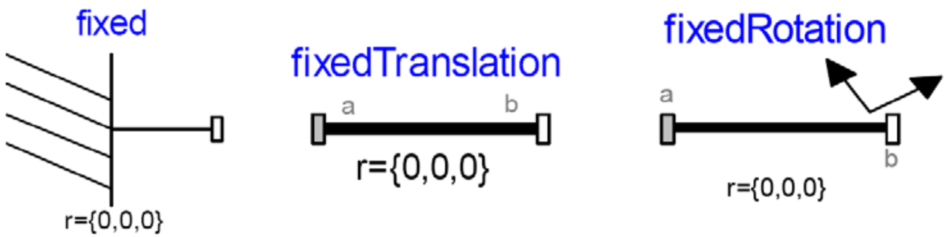
- Examples:



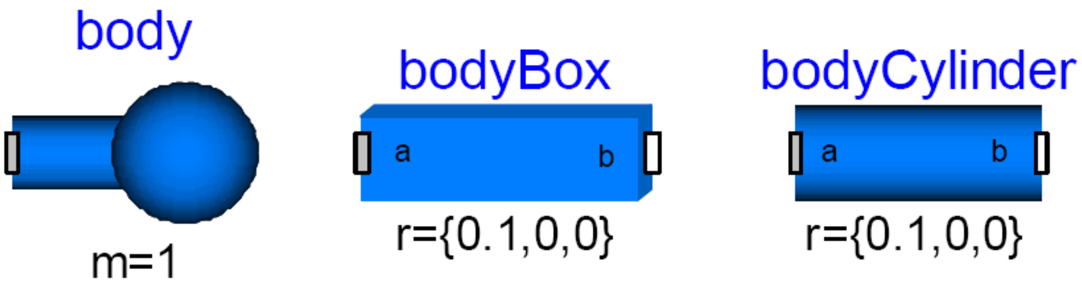
Modelica Multibody: Elementary components III



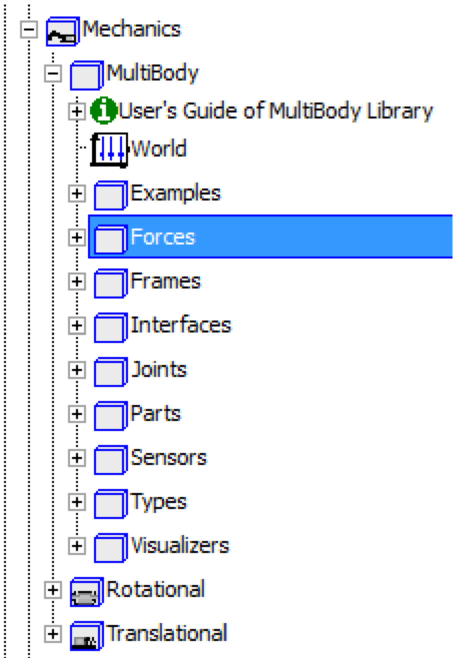
- Modelica.Mechanics.MultiBody.Parts
  - Fixed, Fixed Translation and Fixed Rotation



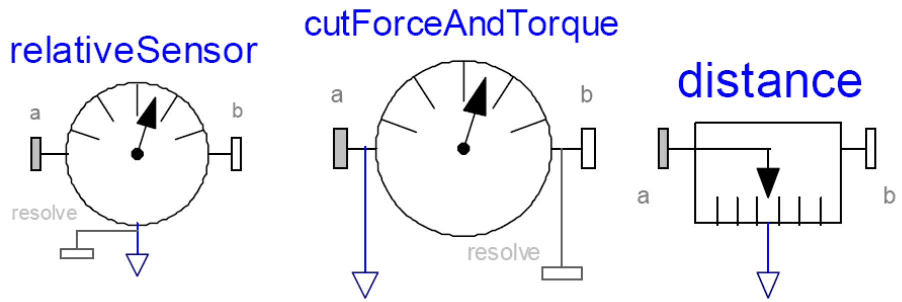
- Rigid bodies with predefined geometric shapes



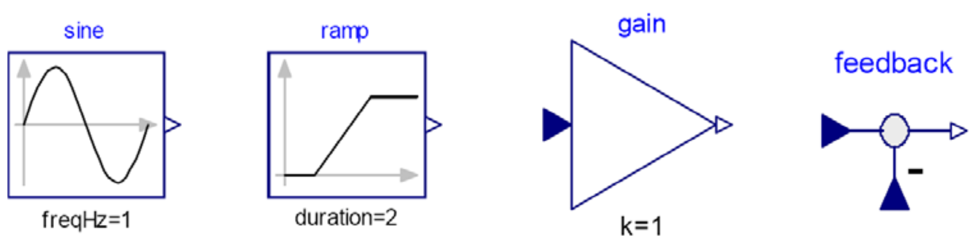
Modelica Multibody: Elementary components IV



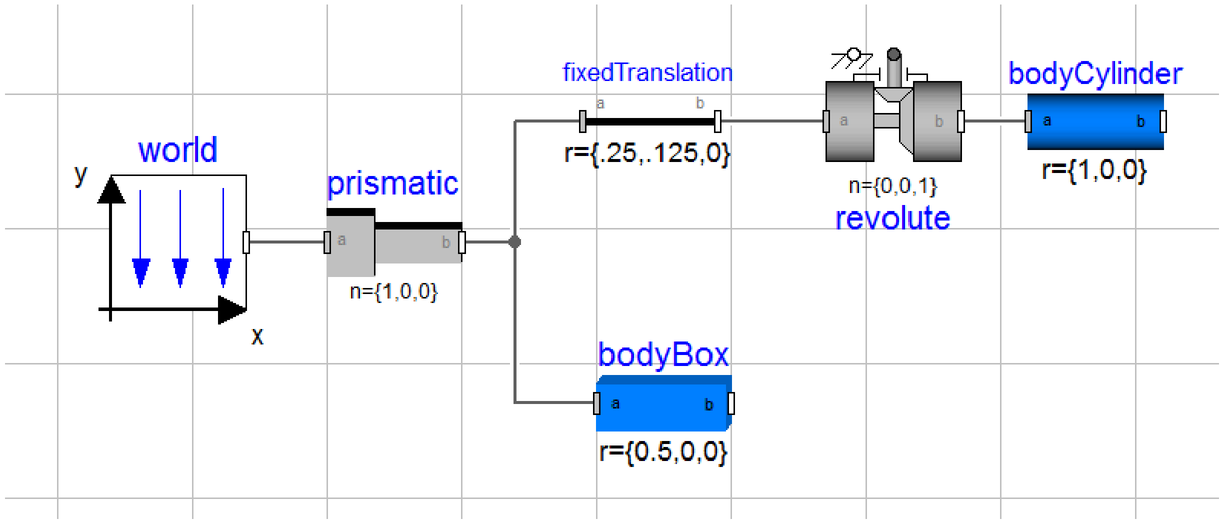
- Modelica.Mechanics.Multibody.Sensors
  - For control and validation purposes



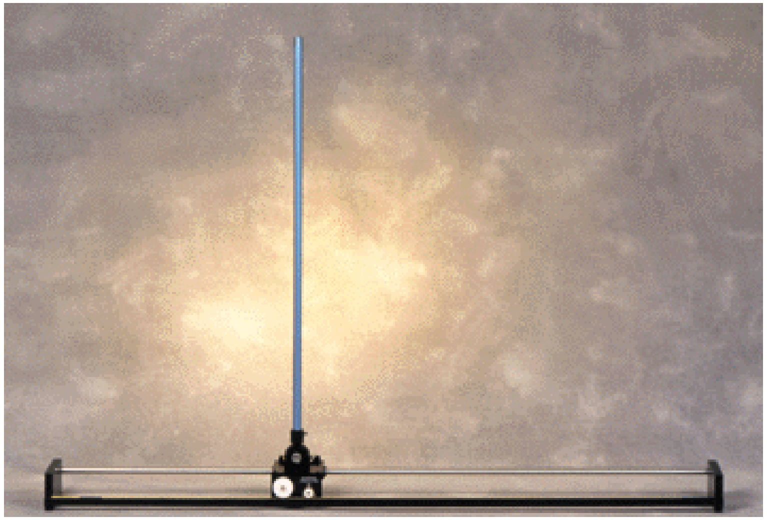
- Modelica.Blocks.Sources + Modelica.Blocks.Math



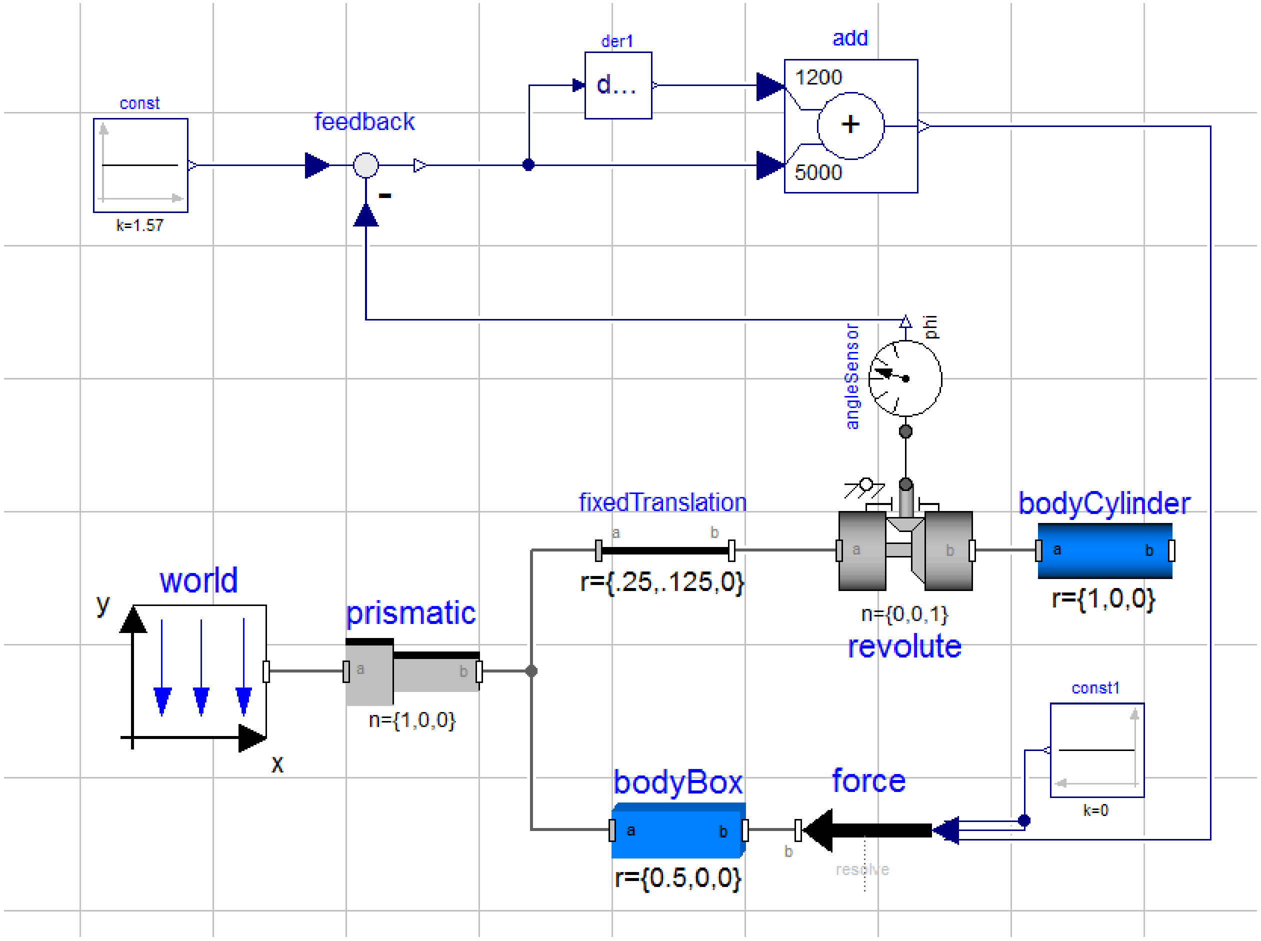
Example: Inverted pendulum, modeling



- Box: 0.5m x 0.25m x 0.25m
- Cylinder: L = 1m, r = 0.05m



Example: Inverted pendulum, PD control

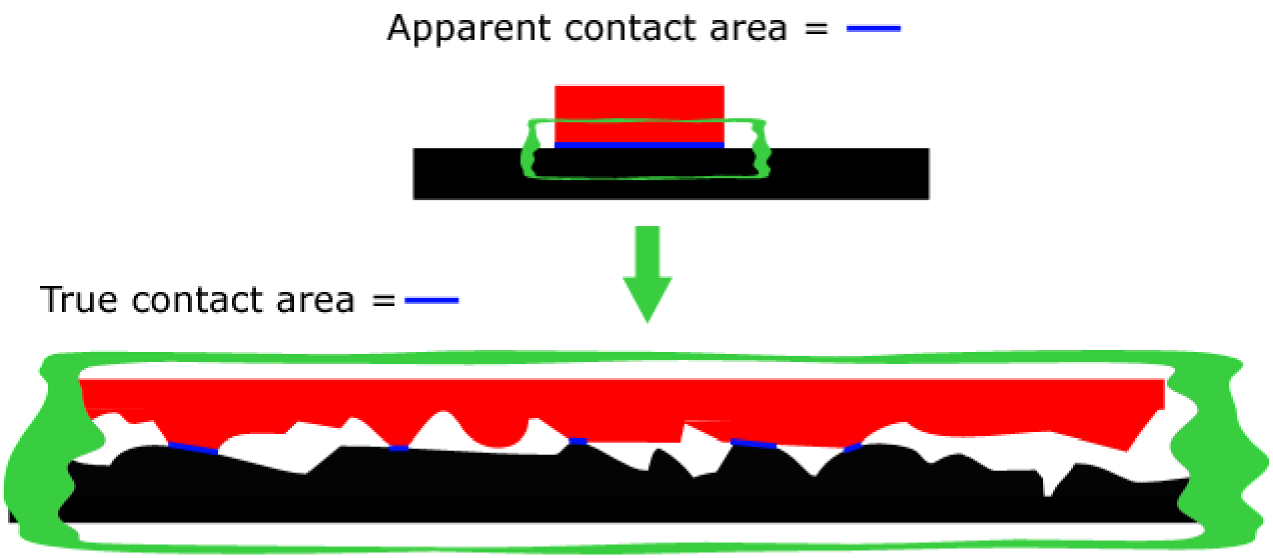


Friction

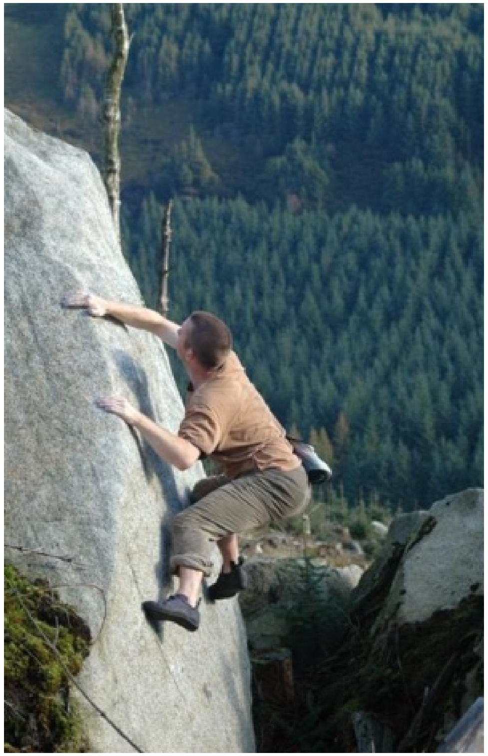
- What is friction?
- Why is it important to know about friction for a control engineer?
- Static friction models
- Dynamic friction models

Book: Ch. 5

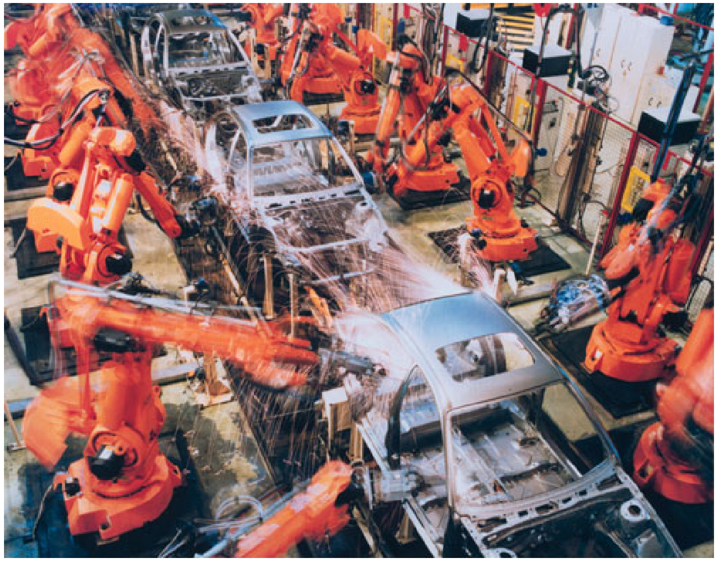
What is friction?



- “The evil of all motion”:
  - No matter which direction something is pushed, friction pulls it the other way
- But not all bad: Without friction we cannot move
  - Walking, cycling, driving, flying, …

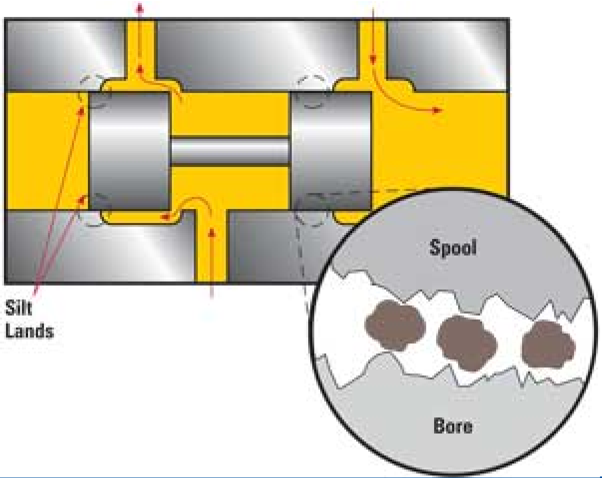


Control systems with friction I



Friction is a problem for

- Control systems for positioning
  - Electrical and hydraulic actuators
  - Translational or rotational
- In process systems: Valves with friction
  - Often “stiction”



Control systems with friction II

Friction can be used to control motion

- Electronic stability control (ESC), ”anti-skidding”

Without ESC:

With ESC:

- Also ABS systems exploits friction characteristics

Static friction models

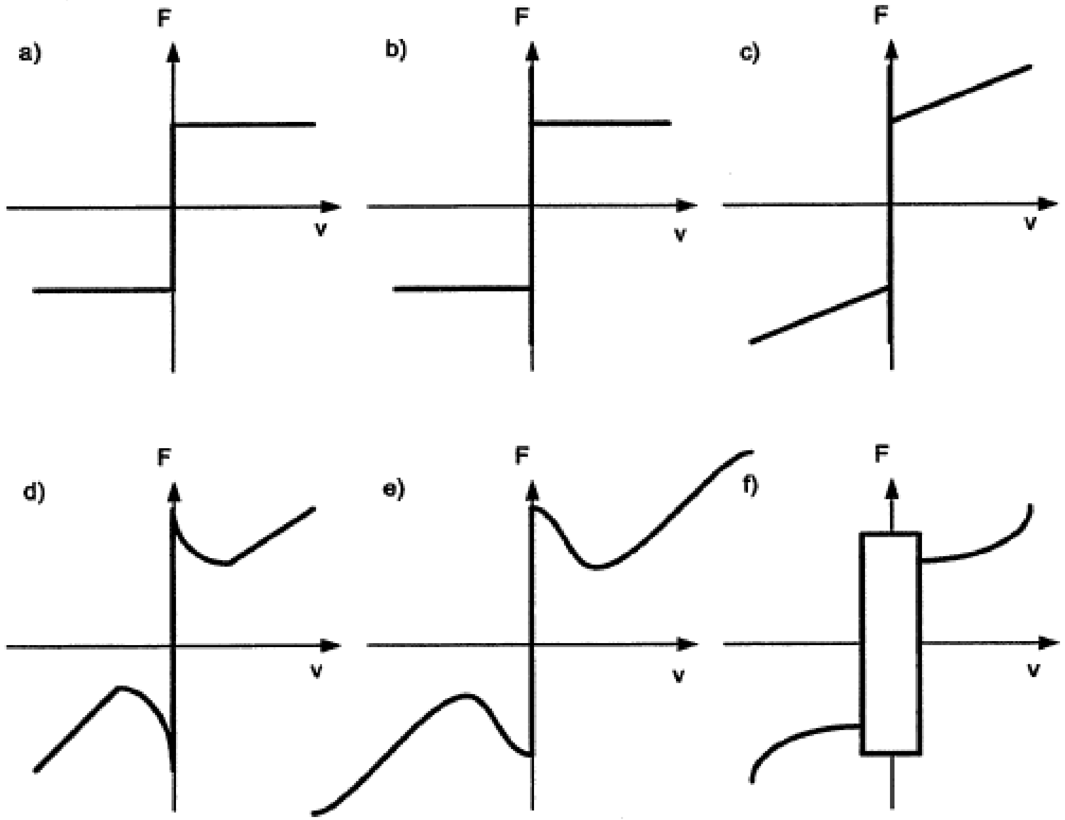
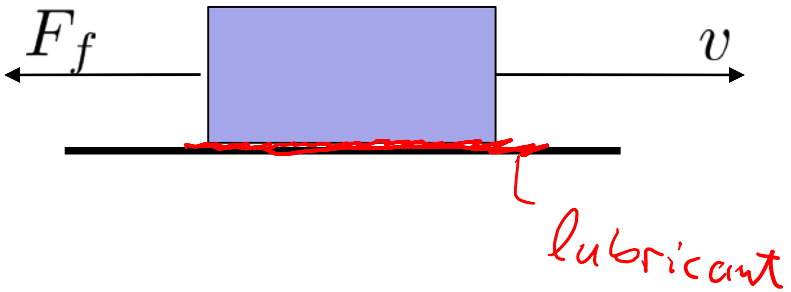
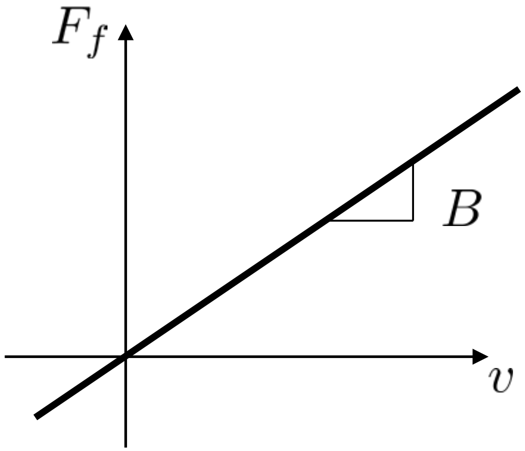


Figure 5.3: Static friction models: a) Colomb friction b) Coulomb+stiction c) Coulomb+stiction+viscous d) Stribeck effect e) Hess and Soom; Armstrong f) Karnopp model

Viscous friction



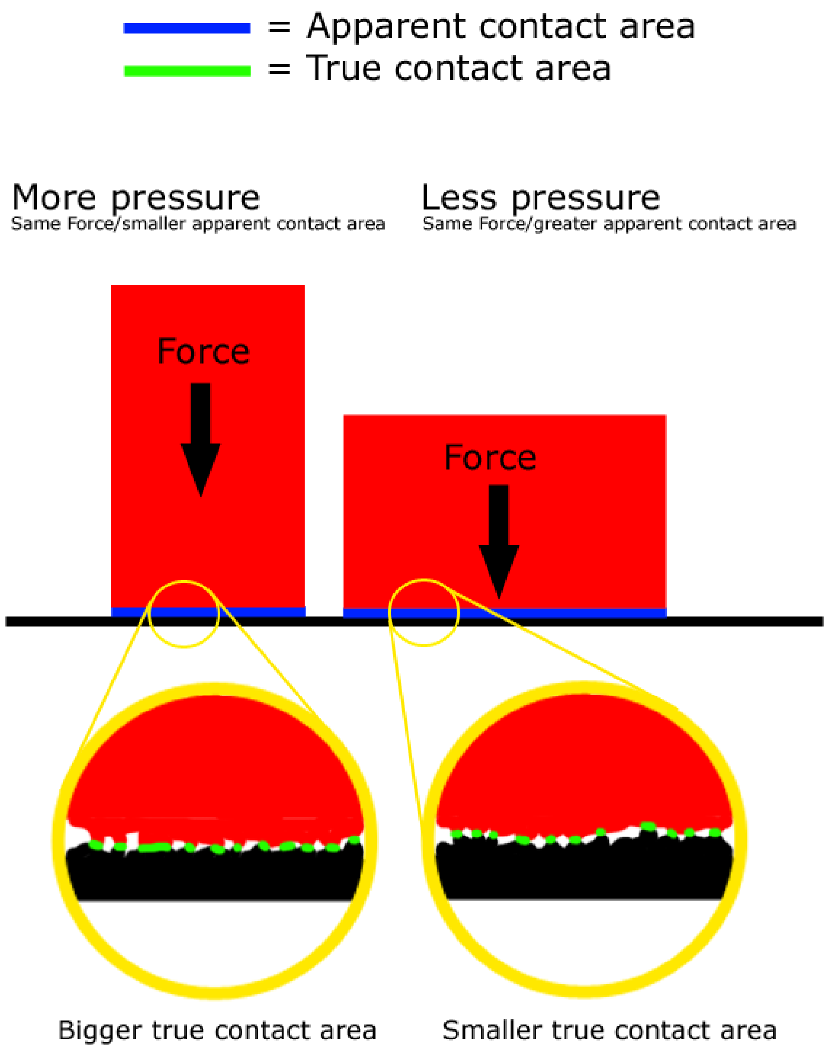
$$F_f = Bv$$



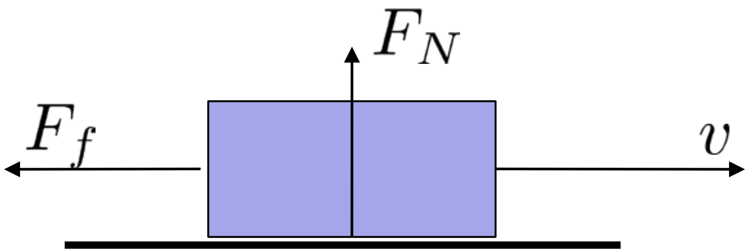
Dry friction is

- Independent of area
  - Da Vinci
- Proportional to normal force
  - Amonton, Euler
- Independent of velocity
  - Coloumb

$$F = \mu F_N$$

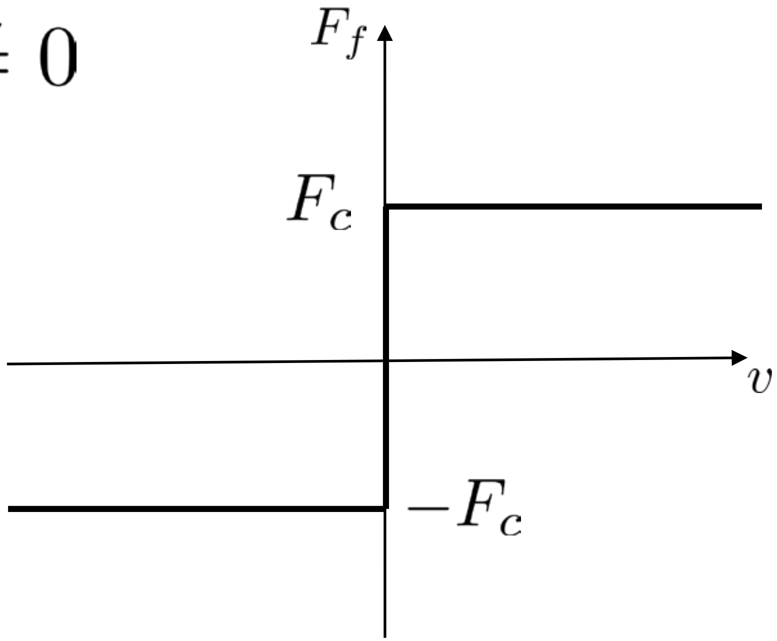


Coulomb friction

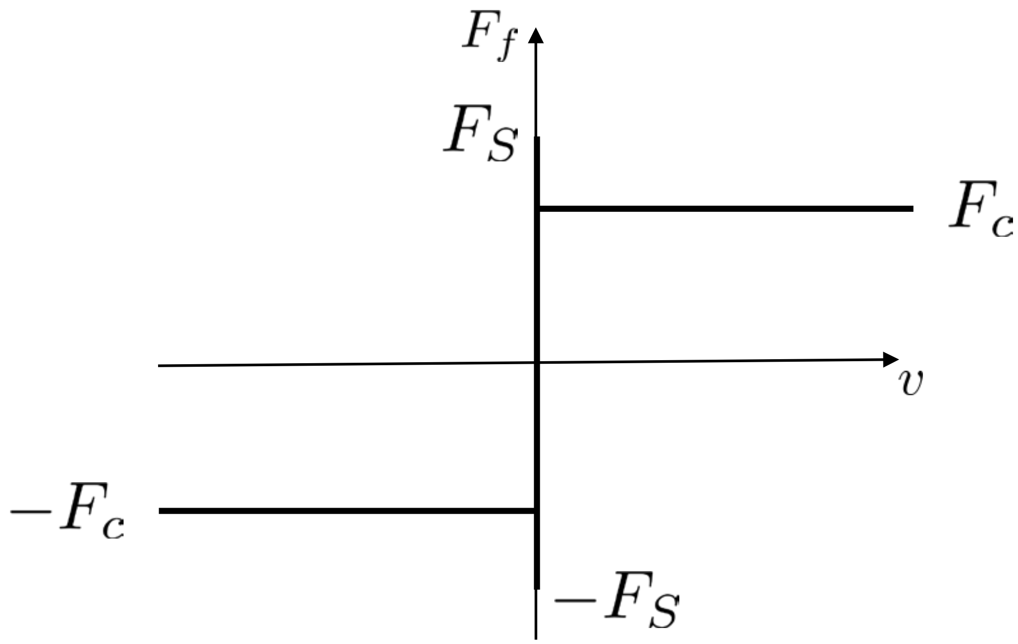


$$F_c = \mu F_N$$

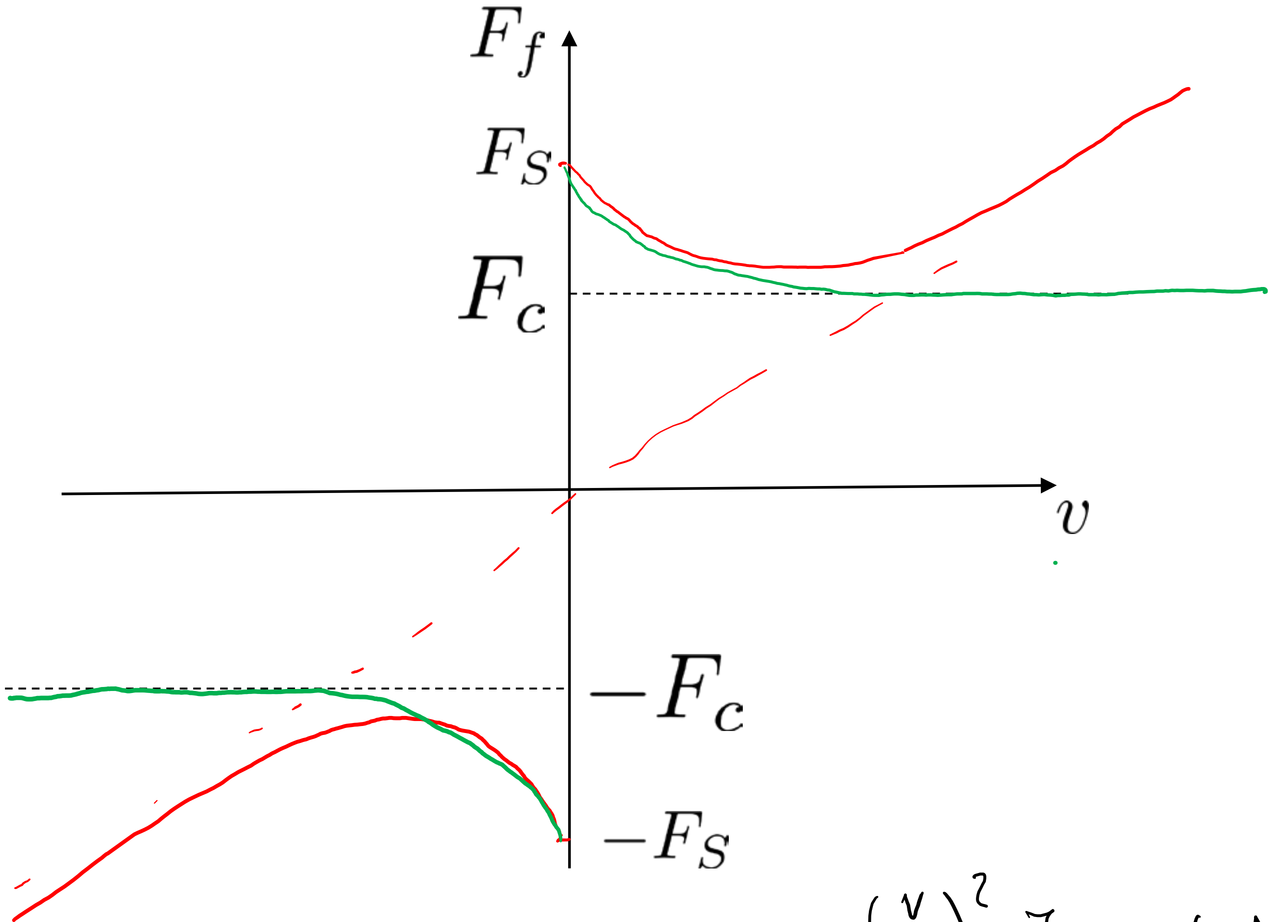
$$F_f = F_c \, \mathrm{sign}(v); \quad v \neq 0$$



Static friction (stiction)



Stribeck-effect



$$F_f = \left[F_c + (F_s - F_c) e^{-\left(\frac{v}{v_s}\right)^2} \right] \mathrm{sgn}(v) + \beta v$$

Generalized Stribeck curve

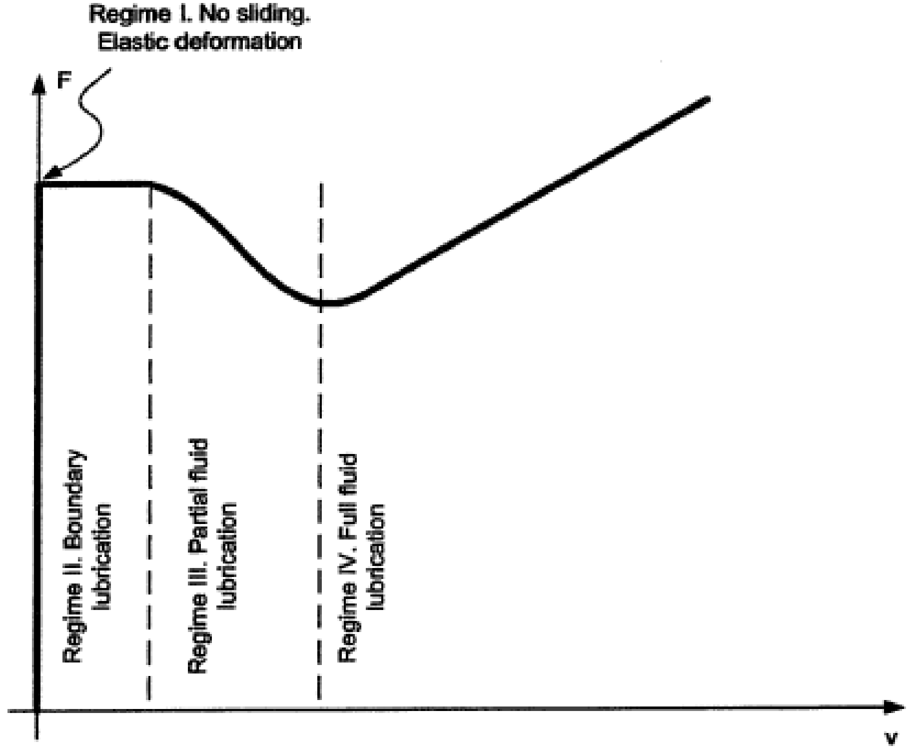
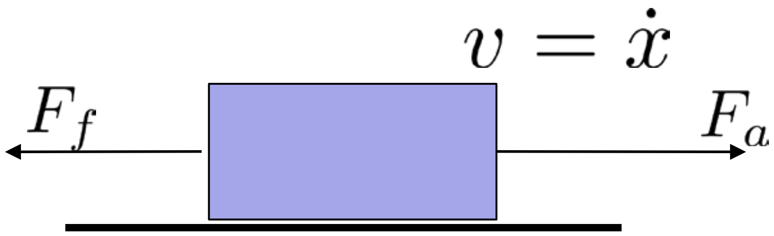
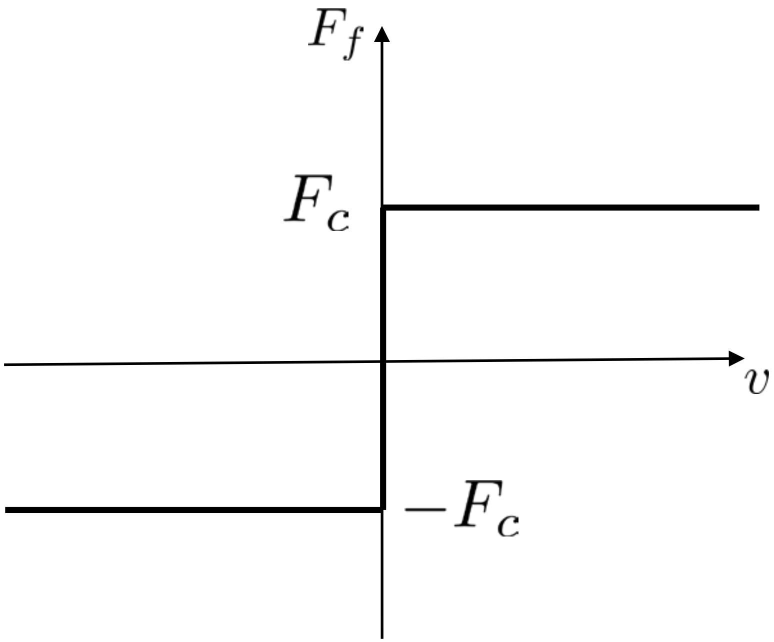


Figure 5.2: The generalized Stribeck curve, showing friction as a function of velocity for low velocities, (Armstrong-Hélouvry et al. 1994).

Problems with the signum terms at zero velocity I



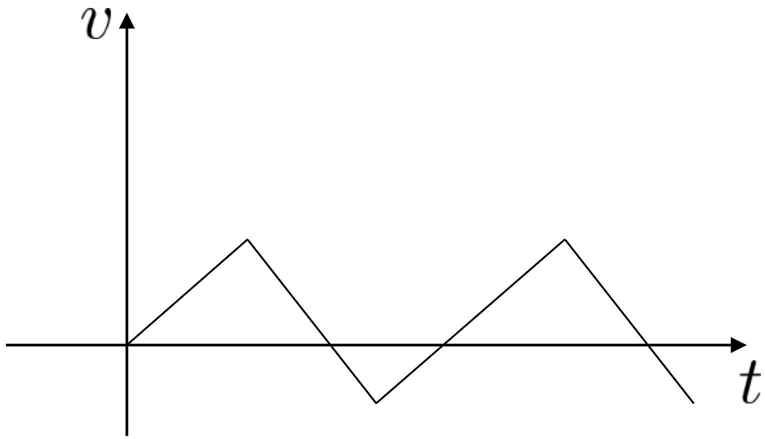


Newton's law:

$$m\dot{v} = F_a - F_f$$

$$F_f = F_c \,\mathrm{sign}(v) = \begin{cases} -F_c, & v < 0 \\ 0, & v = 0 \\ F_c, & v > 0 \end{cases}$$

Problems with the signum terms at zero velocity II



Assume: $0 < F_a < F_f$

$v = 0$ $\quad F_f = 0$ $\quad m\dot{v} = F_a > 0 \rightarrow v$ increases

$v > 0$ $\quad m\dot{v} = F_a - F_f < 0 \rightarrow v$ decreased

Karnopp's model

$$m\dot{v} = F_a - F_f \qquad \begin{cases} F_a + F_c & v < 0 \\ F_a - F_c & v > 0 \end{cases}$$

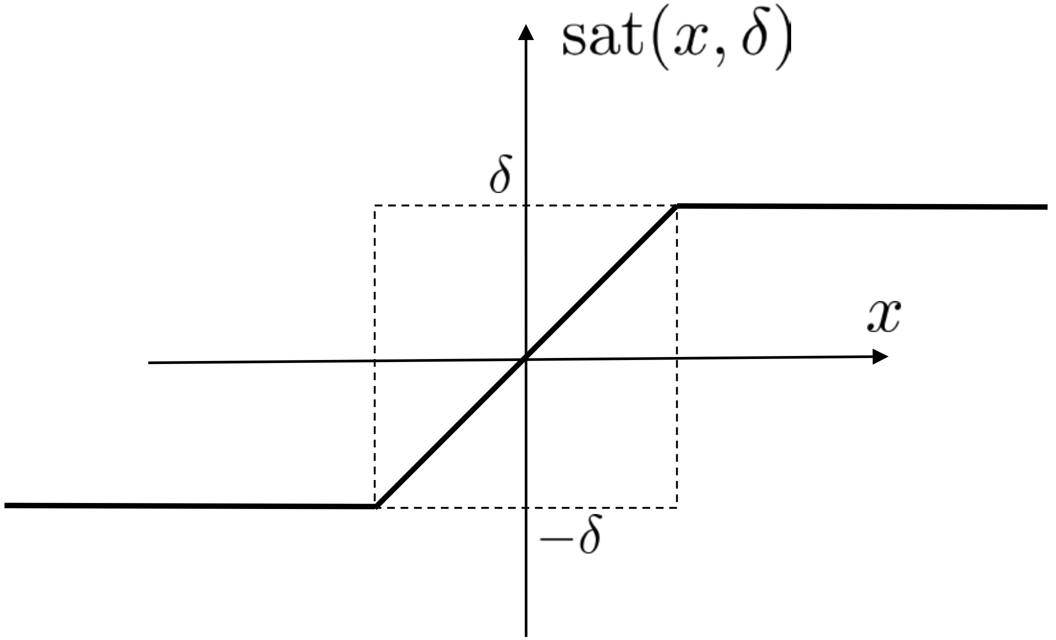
We want $m\dot{v} = 0$ if $v = 0$ and $|F_a| < F_c$

Therefore $F_f = F_a$ if $v = 0$ and $|F_a| < F_c$

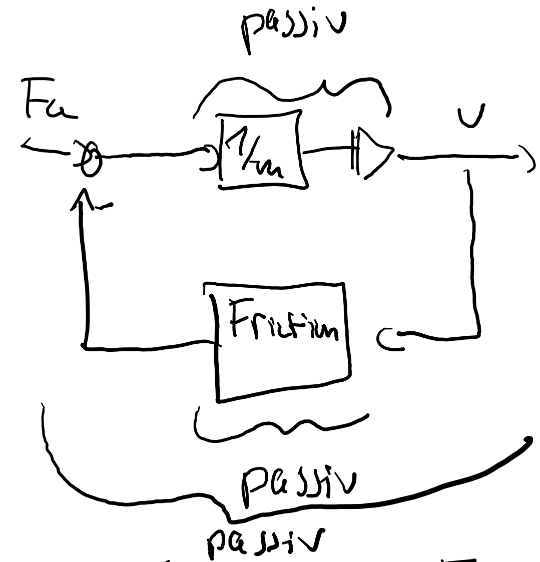
$$F_f = \begin{cases} \mathrm{sat}(F_a, F_c) \; ; & v = 0 \\ F_c \,\mathrm{sgn}(v) \; ; & v \neq 0 \end{cases}$$

Saturation function

$$\operatorname{sat}(x,\delta) = \begin{cases} x, & |x| \leq \delta \\ \delta \operatorname{sgn}(x), & |x| > \delta \end{cases}$$

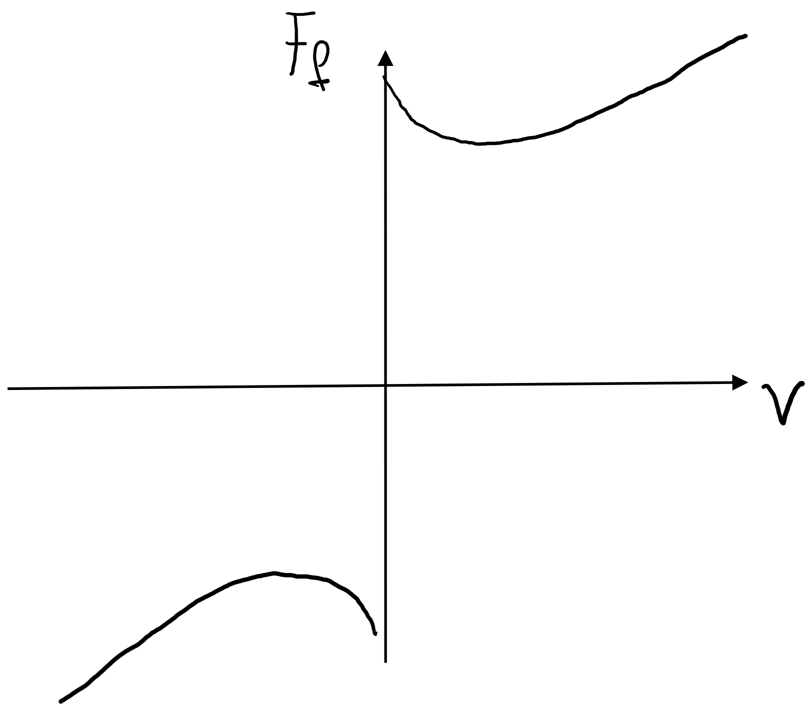


Passivity of friction models



passiv if

$$\int_{t_0}^{t} F_f(\tau)\cdot v(\tau)\, d\tau \geq -E_0$$



$$\Rightarrow \int_{t_0}^{t} F_f \cdot v \, d\tau \geq 0$$

Homework

- Implement the inverted pendulum with PID controller in Modelica

- Read 3.1-3.2

Dynamic friction models

The Dahl model

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \sigma \left(v - |v| \frac{F}{F_c} \right)$$

The LuGre model

$$F = \sigma_0 z + \sigma_1 \frac{\mathrm{d}z}{\mathrm{d}t} + \sigma_2 v$$

$$\frac{\mathrm{d}z}{\mathrm{d}t} = v - \sigma_0 \frac{|v|}{g(v)} z$$

$$g(v) = F_c + (F_s - F_c) e^{-\left(\frac{v}{v_s}\right)^2}$$

Why dynamic friction models?

- Easier to simulate
- Easier to analyze
- They reproduce (to some extent) dynamic friction phenomena
  - Presliding displacement
    - friction force act as a spring in sticking region
  - Frictional lag
    - Dynamic friction force depends on direction of velocity
  - Varying break-away force
    - Break-away force depends on rate-of-change of applied force

Dahl's model

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \sigma \left(v - |v| \frac{F}{F_c} \right)$$

Passivity of Dahl's model $\frac{\mathrm{d}F}{\mathrm{d}t} = \sigma\left(v - |v|\frac{F}{F_c}\right)$

LuGre model I

$$F = \sigma_0 z + \sigma_z \dot{z} + \sigma_2 v$$

$$\dot{z} = v - \sigma_0 \frac{|v|}{g(v)} z \qquad g(v) = F_c + (F_s - F_c) e^{-\left(\frac{v}{v_s}\right)^2}$$

LuGre model II

- «time constant»:

$$T = \frac{g(v)}{\sigma_0 |v|} \rightarrow \inf, \quad \text{if} \quad |v| \rightarrow 0$$

- Same advantageous/disadvantegous as Dahl’s model
- Possible more realistic dynamic behaviour
- LuGre-model is passive from v to F if σ_1 is small enough

ABS-system – blokkeringsfrie bremser

- Hva er det som gjør at **bremsing**, **gass**, **styring** får bilen til å endre hastighet?

Friksjon mellom hjul og vei

- Hva bestemmer friksjon?
  - Tyngde
  - Underlag og egenskaper ved dekk
    - tørr asfalt, våt asfalt, snø, is
  - **Relativ hastighetsforskjell** mellom bil og hjul
    - langsgående (longitudinal) slipp, side- (lateral) slipp

Slipp – relativ hastighetsforskjell

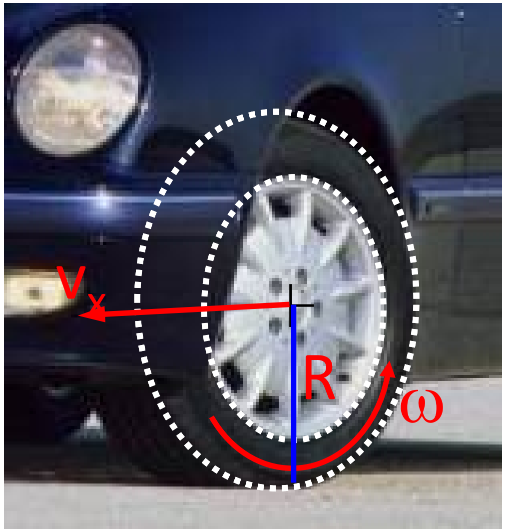
- I langsretning:

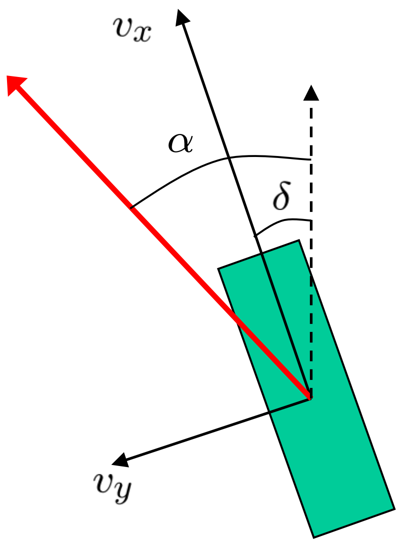
$$\lambda_x := \frac{v_x - R\omega}{v_x}$$

- I sideretning:

$$\lambda_y := \sin\alpha$$

$$\alpha := \delta + \arctan\frac{v_y}{v_x}$$





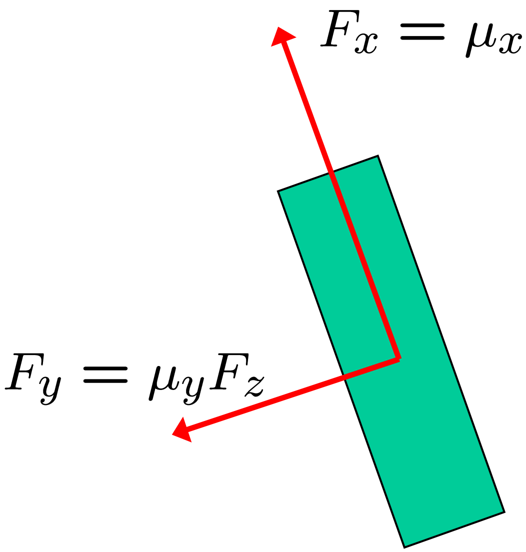
Friksjonskrefter

Coloumbs lov:

- Friksjonskrefter gitt av vertikale krefter og friksjonskoeffisient

- Friksjonskoeffisient gitt av slipp og underlag

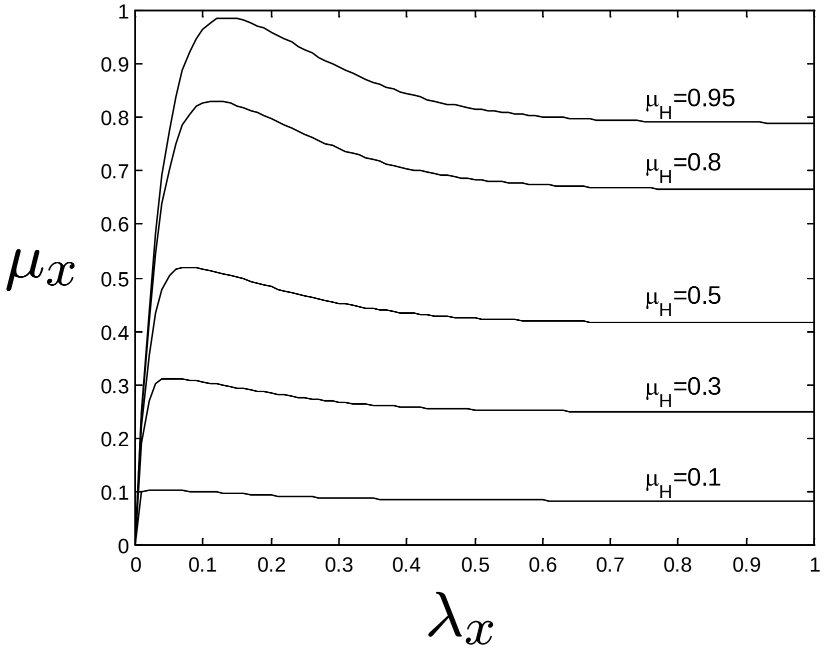
$$\mu_x \approx \mu_x(\lambda_x, \lambda_y, \mu_H)$$
$$\mu_y \approx \mu_y(\lambda_y, \lambda_x, \mu_H)$$

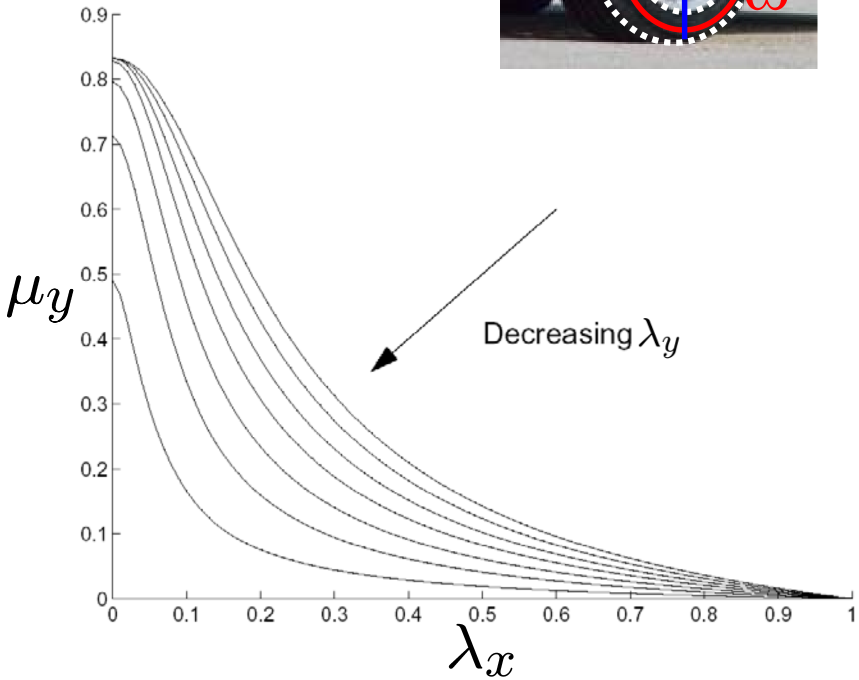


Friksjonskoeffisienter under bremsing

- Bremsing reduserer hjulhastighet i forhold til bilhastighet

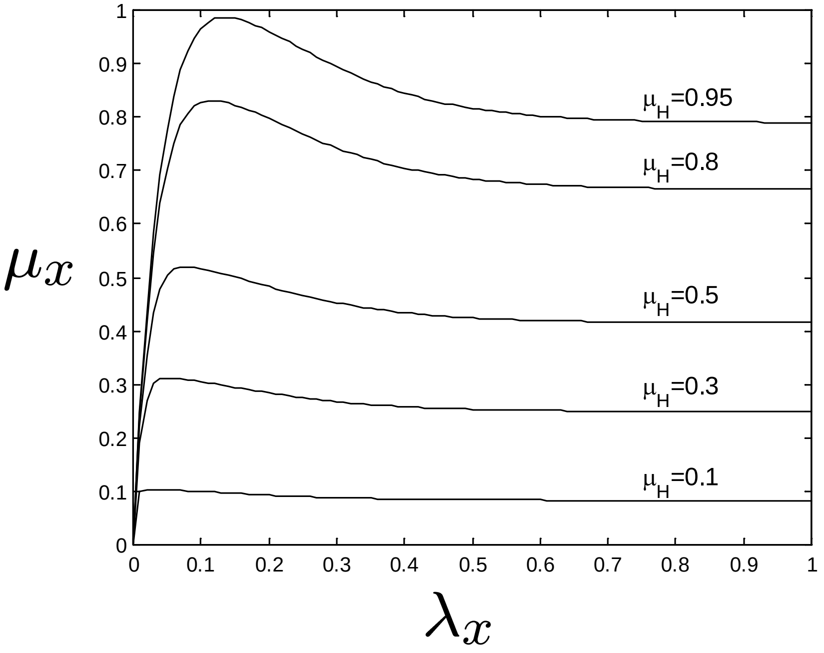
$$\lambda_x := \frac{v_x - R\omega}{v_x}$$

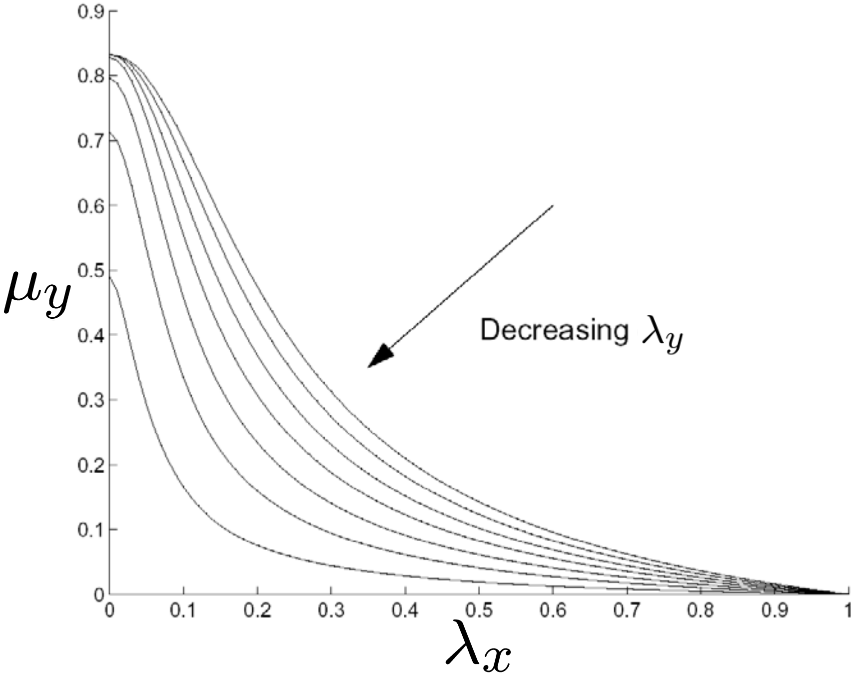




Blokkeringsfrie bremser – ABS

- Ønsker konstant lav slipp under bremsing fordi
  - Det gjør bremsing mest effektivt
  - Kan styre bilen under bremsing





ABS i praksis

Bremselengde:

Unnamanøver: