Siconos

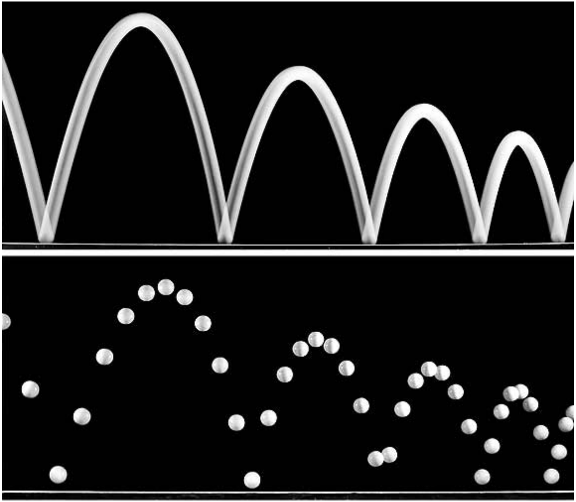
An opensource software platform for the modeling, the simulation and the control of nonsmooth mechanical and electrical systems

V. Acary, O. Bonnefon, M. Brémond, O. Huber, F. Pérignon & S. Sinclair
siconos-team@lists.gforge.inria.fr

Tripop team. INRIA Rhône–Alpes, Grenoble. & LJK

June 1, 2018

What is a Non Smooth Dynamical System (NSDS) ?



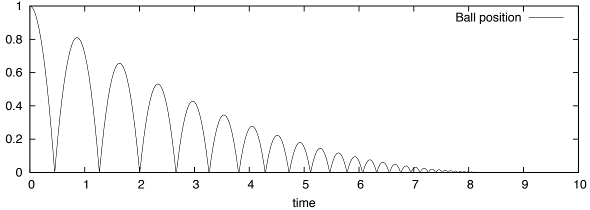
What is a Non Smooth Dynamical System (NSDS) ?

A NSDS is a dynamical system characterized by two correlated features:

- a non smooth evolution with the respect to time:
  - Jumps in the state and/or in its derivatives w.r.t. time
  - Generalized solutions (distributions)
- a set of non smooth laws (Generalized equations, inclusions) constraining the state x

NSDS are a special class of Hybrid Systems coupling:

- A set of continuous dynamical systems (modes)
- A set of discrete rules governing the mode selection.



Non Smooth modeling vs. General Hybrid Modeling

NSDS: a special class of Hybrid Systems, but ...

A NSDS is a special class of Hybrid Systems with

- a strong mathematical structure
- well–posedness results (existence, uniqueness, continuity with the respect to data)
- Efficient simulation tools

Two examples

- Use of mathematical programming (Optimization) formulations and techniques (LCP, QP)
  - Better than enumerative algorithm for conditional statement
  - polynomial complexity for well-posed physical systems.
- Use of specific time-stepping schemes without explicit event handling.
  - Better than Event-driven strategies for a huge number of discrete events.
  - Ability to handle functions of bounded variations (finite accumulations of discontinuities.)
  - Definition of global solutions in the space of distributions.

Typical examples

- Differential inclusions & variational inequalities
- Mechanical systems with unilateral contact, Coulomb's Friction and impacts
- Complementarity systems
- Optimal control with state constraints
- Sliding Mode Control
- ...

Application fields

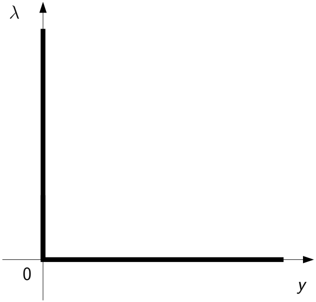
- Non-smooth Mechanical systems
- Non smooth Electrical Circuits
- MEMS and NEMS
- Computer Graphics, Virtual Reality and Haptic systems
- Genetic regulatory networks
- Macro-economic dynamical model
- ...

Typical examples

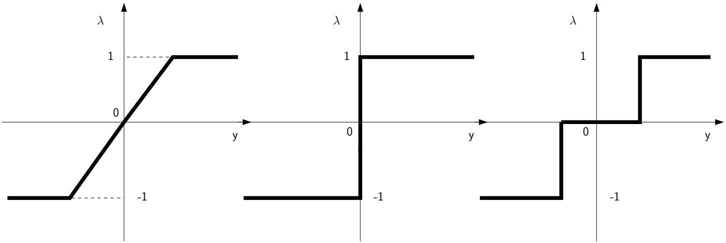
Linear Complementarity Systems (LCS)

$$\begin{cases} \dot{x} = Ax + B\lambda, \quad x \in \mathbb{R}^n, \lambda \in \mathbb{R}^m \\ y = Cx + D\lambda \\ 0 \leqslant y \perp \lambda \geqslant 0 \end{cases} \tag{1}$$

with $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{n\times m}, C \in \mathbb{R}^{m\times n}, D \in \mathbb{R}^{m\times m}$, for m constraints.



Piecewise linear systems



Saturation Relay Relay with dead zone

Mixed complementarity systems

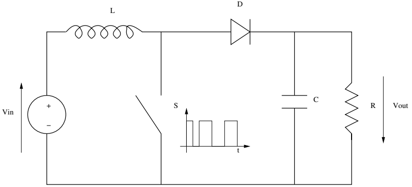
$$\begin{cases} M\dot{x} = f(x,t) + g(x,\lambda,t), \quad x \in \mathbb{R}^n, \lambda \in \mathbb{R}^m \\ y = h(x,\lambda,t) \\ -y \in N_K(\lambda) \end{cases} \tag{2}$$

with $K = \prod_i [l_i, u_i]$ and M may singular. (The relative degree is assumed to be less than 1)

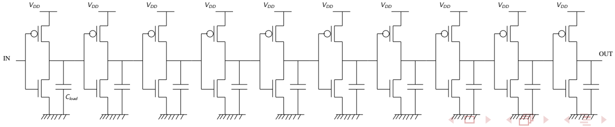
Applications

Electrical networks

Simulation, modeling and control of electrical networks with idealized components (diodes, transistors, switch, ...)



DC-DC Boost Converter with Sliding mode control



Lagrangian systems with unilateral contact and Coulomb's friction

Lagrangian dynamical systems

$$M(q)\ddot{q} + Q(\dot{q}, q) + F(\dot{q}, q, t) = F_{ext}(t) + R$$

- $q \in \mathbb{R}^n$: generalized coordinates vector.
- $M \in \mathbb{R}^{n\times n}$: the inertia matrix
- $Q(\dot{q}, q)$: The non linear inertial term (Coriolis)
- $F(\dot{q}, q, t)$: the internal forces
- $F_{ext}(t) : \mathbb{R} \mapsto \mathbb{R}^n$: given external load,
- $R \in \mathbb{R}^n$ is the force due the non smooth law.

Lagrangian systems with unilateral contact and Coulomb's friction

Lagrangian dynamical systems

$$M(q)\ddot{q} + Q(\dot{q}, q) + F(\dot{q}, q, t) = F_{ext}(t) + R$$

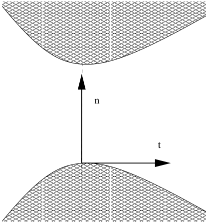
- $q \in \mathbb{R}^n$: generalized coordinates vector.
- $M \in \mathbb{R}^{n\times n}$: the inertia matrix
- $Q(\dot{q}, q)$: The non linear inertial term (Coriolis)
- $F(\dot{q}, q, t)$: the internal forces
- $F_{ext}(t) : \mathbb{R} \mapsto \mathbb{R}^n$: given external load,
- $R \in \mathbb{R}^n$ is the force due the non smooth law.

Kinematic linear relations

- Kinematic laws from the generalized coordinates to the local coordinates at contact.
$$y = H^T q + b, \dot{y} = H^T \dot{q}$$
Mapping H : Restriction mapping composed with a change of frame
- By duality,
$$R = H\lambda$$

Lagrangian systems with unilateral contact and Coulomb's friction

- Local frame at contact : $(\boldsymbol{n}, \boldsymbol{t})$

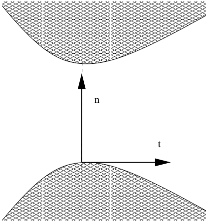


$$y = y_{\boldsymbol{n}}\boldsymbol{n} + y_{\boldsymbol{t}}, \quad \dot{y} = \dot{y}_{\boldsymbol{n}}\boldsymbol{n} + \dot{y}_{\boldsymbol{t}}$$

$$\lambda = \lambda_{\boldsymbol{n}}\boldsymbol{n} + \lambda_{\boldsymbol{t}},$$

Lagrangian systems with unilateral contact and Coulomb's friction

- Local frame at contact : $(\boldsymbol{n}, \boldsymbol{t})$



$$y = y_{\boldsymbol{n}}\boldsymbol{n} + y_{\boldsymbol{t}}, \quad \dot{y} = \dot{y}_{\boldsymbol{n}}\boldsymbol{n} + \dot{y}_{\boldsymbol{t}}$$

$$\lambda = \lambda_{\boldsymbol{n}}\boldsymbol{n} + \lambda_{\boldsymbol{t}},$$

- Unilateral contact :

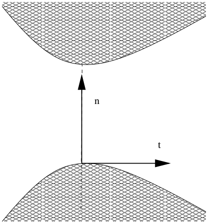
$$0 \leqslant y_{\boldsymbol{n}} \perp \lambda_{\boldsymbol{n}} \geqslant 0 \quad \Longleftrightarrow \quad -\lambda_{\boldsymbol{n}} \in \partial\Psi_{\mathbb{R}^+}(y_{\boldsymbol{n}})$$

Lagrangian systems with unilateral contact and Coulomb's friction

- Local frame at contact : $(\boldsymbol{n}, \boldsymbol{t})$

$$y = y_{\boldsymbol{n}}\boldsymbol{n} + y_{\boldsymbol{t}}, \quad \dot{y} = \dot{y}_{\boldsymbol{n}}\boldsymbol{n} + \dot{y}_{\boldsymbol{t}}$$

$$\lambda = \lambda_{\boldsymbol{n}}\boldsymbol{n} + \lambda_{\boldsymbol{t}},$$



- Unilateral contact :

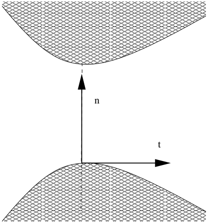
$$0 \leqslant y_{\boldsymbol{n}} \perp \lambda_{\boldsymbol{n}} \geqslant 0 \quad \Longleftrightarrow \quad -\lambda_{\boldsymbol{n}} \in \partial\Psi_{\mathbb{R}^+}(y_{\boldsymbol{n}})$$

- Coulomb's Friction, μ Coefficient of friction, $\mathcal{C}(\mu\lambda_{\boldsymbol{n}}) = \{\lambda_{\boldsymbol{t}}, \|\lambda_{\boldsymbol{t}}\| \leqslant \mu\lambda_{\boldsymbol{n}}\}$

$$\begin{cases} \dot{y}_{\boldsymbol{t}} = 0, \|\lambda_{\boldsymbol{t}}\| \leqslant \mu\lambda_{\boldsymbol{n}} \\ \dot{y}_{\boldsymbol{t}} \neq 0, \lambda_{\boldsymbol{t}} = -\mu\lambda_{\boldsymbol{n}}\mathrm{sign}(\dot{y}_{\boldsymbol{t}}) \end{cases} \Longleftrightarrow \dot{y}_{\boldsymbol{t}} \in \partial\Psi_{\mathcal{C}(\mu\lambda_{\boldsymbol{n}})}(-\lambda_{\boldsymbol{t}}) \Longleftrightarrow -\lambda_{\boldsymbol{t}} \in \partial\Psi^*_{\mathcal{C}(\mu\lambda_{\boldsymbol{n}})}(\dot{y}_{\boldsymbol{t}})$$

Lagrangian systems with unilateral contact and Coulomb's friction

- Local frame at contact : $(\boldsymbol{n}, \boldsymbol{t})$



$$y = y_{\boldsymbol{n}}\boldsymbol{n} + y_{\boldsymbol{t}}, \quad \dot{y} = \dot{y}_{\boldsymbol{n}}\boldsymbol{n} + \dot{y}_{\boldsymbol{t}}$$

$$\lambda = \lambda_{\boldsymbol{n}}\boldsymbol{n} + \lambda_{\boldsymbol{t}},$$

- Unilateral contact :

$$0 \leqslant y_{\boldsymbol{n}} \perp \lambda_{\boldsymbol{n}} \geqslant 0 \quad \Longleftrightarrow \quad -\lambda_{\boldsymbol{n}} \in \partial \Psi_{\mathbb{R}^+}(y_{\boldsymbol{n}})$$

- Coulomb's Friction, μ Coefficient of friction, $\mathcal{C}(\mu\lambda_{\boldsymbol{n}}) = \{\lambda_{\boldsymbol{t}}, \|\lambda_{\boldsymbol{t}}\| \leqslant \mu\lambda_{\boldsymbol{n}}\}$

$$\begin{cases} \dot{y}_{\boldsymbol{t}} = 0, \|\lambda_{\boldsymbol{t}}\| \leqslant \mu\lambda_{\boldsymbol{n}} \\ \dot{y}_{\boldsymbol{t}} \neq 0, \lambda_{\boldsymbol{t}} = -\mu\lambda_{\boldsymbol{n}}\mathrm{sign}(\dot{y}_{\boldsymbol{t}}) \end{cases} \Longleftrightarrow \dot{y}_{\boldsymbol{t}} \in \partial\Psi_{\mathcal{C}(\mu\lambda_{\boldsymbol{n}})}(-\lambda_{\boldsymbol{t}}) \Longleftrightarrow -\lambda_{\boldsymbol{t}} \in \partial\Psi^*_{\mathcal{C}(\mu\lambda_{\boldsymbol{n}})}(\dot{y}_{\boldsymbol{t}})$$

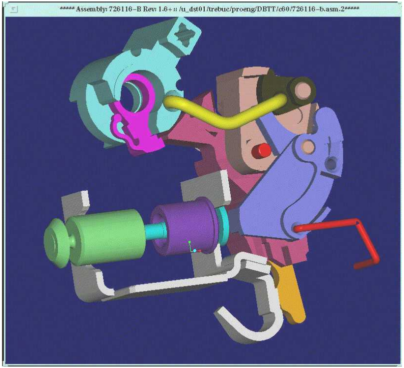
- (Newton) Impact law, if necessary, e coefficient of restitution

$$\dot{y}_{\boldsymbol{n}}(t^+) = -e\dot{y}_{\boldsymbol{n}}(t^-)$$

Mechanical systems with contact, impact and friction

Mechanical systems with contact, impact and friction

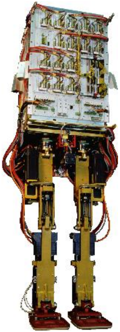
Multi-body systems : Simulation of electrical circuit breakers



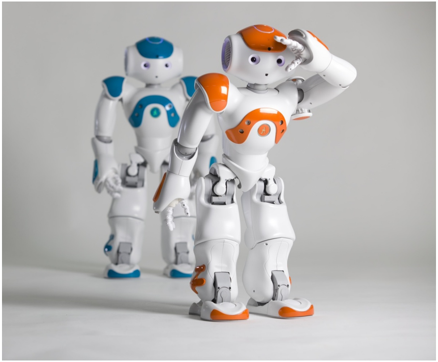
INRIA/Schneider Electric

Mechanical systems with contact, impact and friction

Robotic and Haptic systems



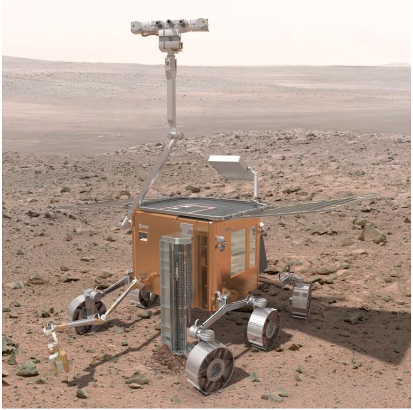
Biped Robot INRIA BIPOP



Aldebaran Robotics NAO

Mechanical systems with contact, impact and friction

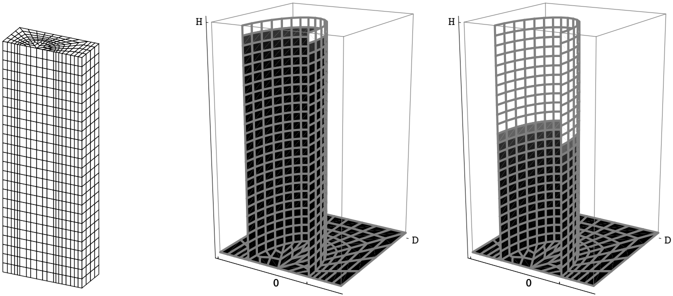
Robotic and Haptic systems



Simulation of the ExoMars Rover (INRIA/Trasys Space/ESA)

Mechanical systems with contact, impact and friction

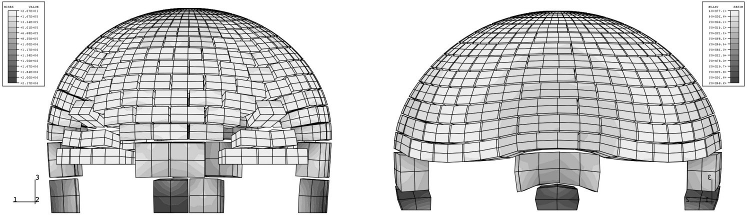
Mechanics of Solids and Structures



FEM cohesive zone modeling of composite. Contact, friction cohesion, etc...
Joint work with Y. Monerie, IRSN.

Mechanical systems with contact, impact and friction

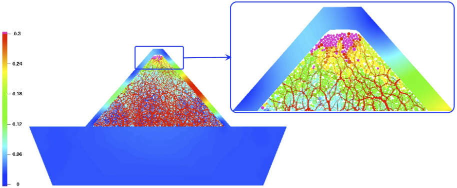
Mechanics of Solids and Structures



Divided Materials and Masonry

Mechanical systems with contact, impact and friction

Dam made of blocks (Saladyn project)



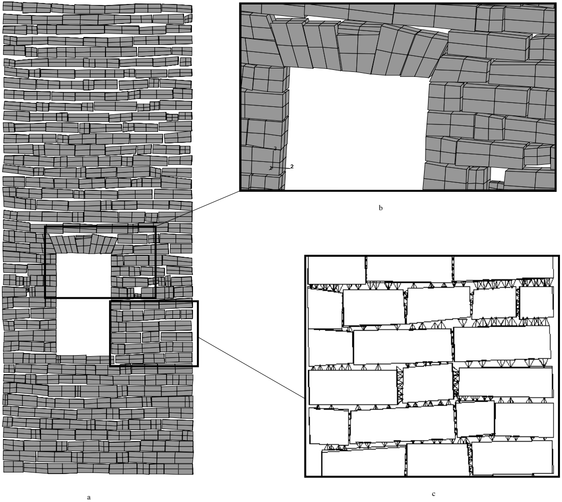
Simulation: Code_Aster + Siconos +LMGC90

Mechanical systems with contact, impact and friction

Mechanics of Solids and Structures. Masonry.

La tour Saint Laurent du palais des Papes à Avignon

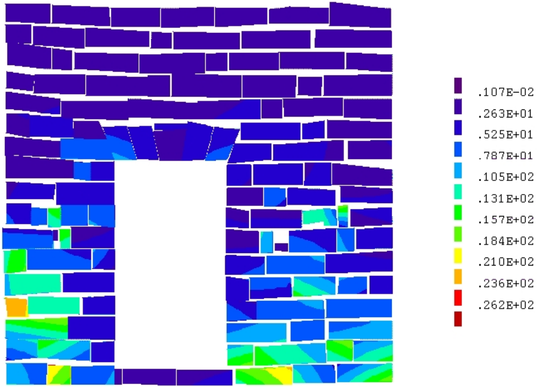
Mechanical systems with contact, impact and friction



Photogrammetric survey and mesh generation

Mechanical systems with contact, impact and friction

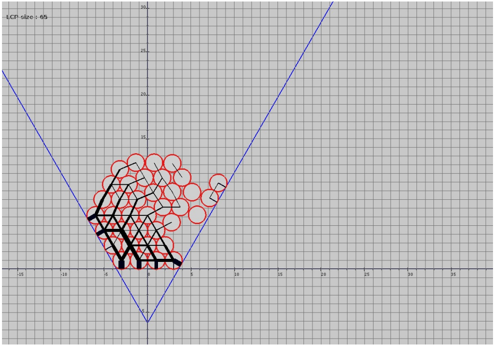
Mechanics of Solids and Structures. Masonry



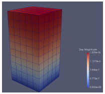
Mechanical stress computation

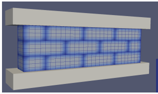
Mechanical systems with contact, impact and friction

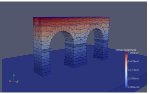
Granular matter

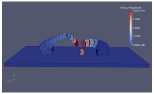


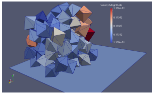
Stack of beads with perturbation

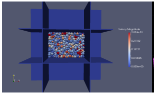

(a) Cubes_H8

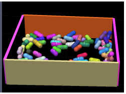

(b) LowWall_FEM


(c) Aqueduct_PR

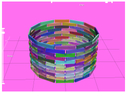

(d) Bridge_PR


(e) 100_PR_Periobox


(f) 945_SP_Box_PL


(g) Capsules


(h) Chain


(i) KaplasTower


(j) BoxesStack

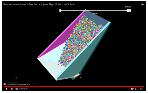
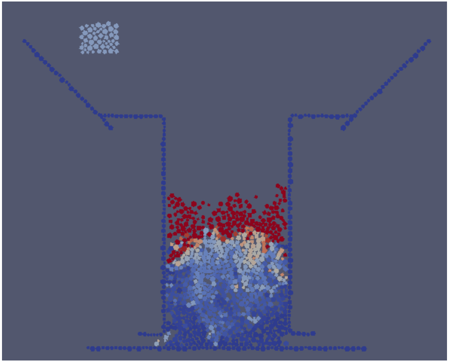

(k) Chute_1000, Chute_4000, Chute_local_problems

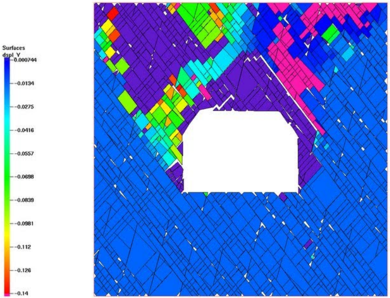
Figure: Illustrations of the FClib test problems

A first application at INRIA Chile. Mining industries

Simulation of granular flows



A first application at INRIA Chile. Mining industries



- Simulation and analysis of granular rock flows.
- Optimization of block caving techniques
  - Role of the preconditioning
  - Fracture processes.

Sliding Mode Control

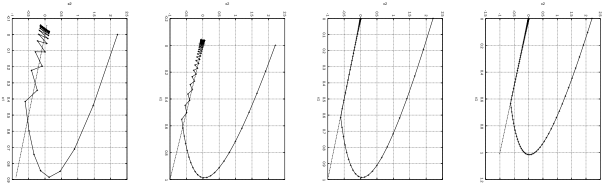
Academic example

$$\dot{x} = -\mathrm{sgn}(x) \tag{3}$$

Chattering-free stabilization

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & -c_1 \end{bmatrix} x - \begin{bmatrix} 0 \\ \alpha \end{bmatrix} \mathrm{sgn}(\begin{bmatrix} c_1 & 1 \end{bmatrix} x). \tag{4}$$

Difference between explicit and implicit time integration



(a) $h = 0.3$. Explicit Euler (b) $h = 0.1$. Explicit Euler (c) $h = 0.1$. Implicit Euler (d) $h = 0.05$. Implicit Euler

Figure: Equivalent control based SMC, $c_1 = 1, \alpha = 1$ and $x_0 = [0, 2.21]^T$. State $x_1(t)$ versus $x_2(t)$.

Matlab/Simulink – Scilab/Scicos

Scilab/Scicos. METALAU project.

- Simulink (Scilab) is a graphical dynamical system modelers and simulator toolbox included in the Matlab (Scilab) engineering and scientific computation software.
- Block diagrams editor to model and simulate the dynamics of hybrid dynamical systems and compile your models into executable code.
- New extensions allow generation of component based modeling of electrical and hydraulic circuits using the Modelica language for scilab.

Modelica

Modelica

- The object-oriented modeling language Modelica is designed to allow convenient, component-oriented modeling of complex physical systems, e.g., systems containing mechanical, electrical, electronic, hydraulic, thermal, control, electric power or process-oriented subcomponents

Modelica compiler and Simulator

- Scicos
- OpenDymola
- Dymola (Dynasim/Dassault Systems)

Simulation challenges for hybrid systems

Simulation approach in Hybrid dynamical system modeler

- Loose coupling between a continuous dynamical simulator (ODE or DAE solvers) and a discrete event simulator
- Event–Driven approach with only external events
- Complex combinatorics to decide the right mode after an event.
- Huge problems of scalability.

Difficulties

- High number of events
- Sliding modes control
- Nonsmooth events due to the lack of regularity in models.
- Difficulties in finding consistent initial conditions

Siconos and some hybrid systems

Hybrid systems issued form a physical modeling

A lot of hybrid systems are issued form a physical modeling: Main part of the system are only "fake" logical dynamics.

- Such systems can be formulated as nonsmooth dynamical systems (Friction, Relay, diode, ...)
- We take benefits from the nonsmooth approach to better simulate these systems.

Original Motivations

Context

The Siconos Platform is one of the main outcome of the Siconos EU project.

Goal: Modeling, simulation, analysis and control of NSDS

There is no other general, common and open software suitable for the modeling and the simulation all of these NSDS

Constraints

- various modeling habits and formulations
- various application fields
- various mathematical and numerical tools

Links and interfaces with existing software

- Matlab or Scilab dedicated user toolbox
- Low-level numerical libraries (BLAS, LAPACK, ODEPACK,. . .)
- Simulation tools for a given application field:
  - Scicos, Simulink
  - FEM and DEM Software (LMGC90, Aster, . . .)
  - Hybrid modeling Language (Modelica, . . .)

Some figures

- 2003. Beginning of the project
- Around 100000 lines of Open–source code in C++, C, Fortran, Python (GPL Licence)
- two APP deposits.
- Around 30 users and contributors
- Human efforts for design and development

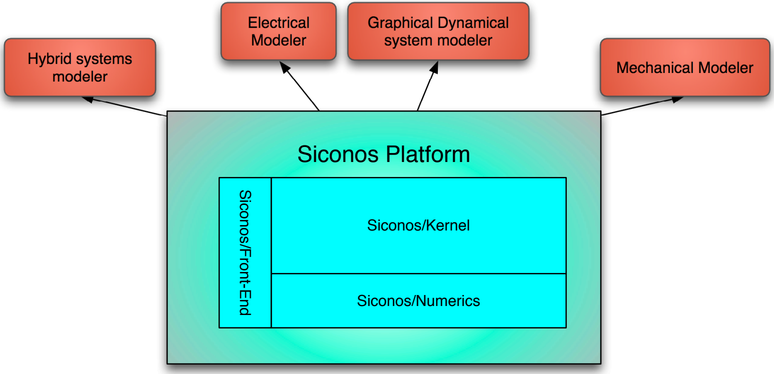
	person/year	type	funds
	3	Software Engineers	SICONOS
	3	Expert Engineers	SICONOS
	2	Researcher	INRIA
	1	PHD thesis	SICONOS
Total	9		

- Human efforts for application and validation

	person/year	type	funds
	3	Expert Engineer	INRIA
	0,5	Expert Engineer	ANR VAL-AMS
	0,5	PHD thesis	UJF
	1	Post Doc	INRIA
Total	5		

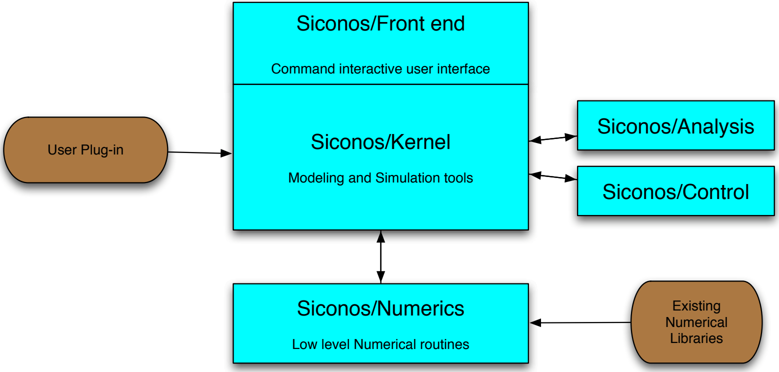
Architecture and Design

Numerical simulation Kernel for various modelers:



Architecture and Design

Siconos Modules



Architecture and Design

SICONOS/Numerics library

- API C
- Shared dynamic library.
- Scilab and Matlab interfaces (Obsolete).

SICONOS/Kernel library

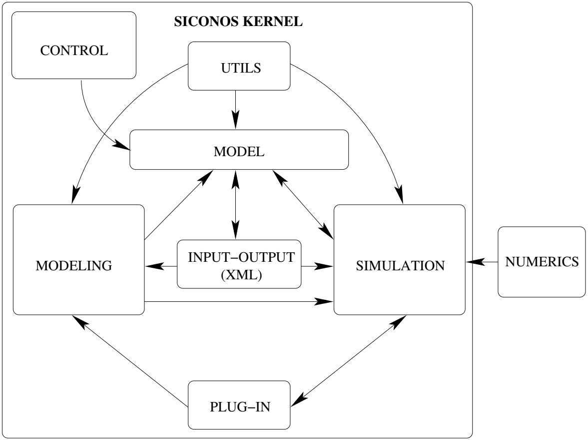
- API C++: Shared dynamic library in other modeling environment.
- API C++: Compiled command files with high level methods (C++ Constructors and/or XML file data loading.)
- API C : Shared dynamic library in low-level environment.

SICONOS/Frond-End

- API Python: Interactive environment (SWIG wrapping).
- API C: Scilab and Matlab interface (Obsolete).

Architecture and Design

Majors functionnalities and modules



Software quality

Substantial effort for a high quality software

- Work in collaboration with SED from the beginning of the project
- Use of the ESA standards for the software quality method

Extensive Software Documentation

1. Project overview
2. Project proposal
3. Software Requirements Specification
   - Functional and non functional requirements
   - Feature set by functionalities and by release and priority
   - Use cases
4. Architectural and Detailed design
   - Description of components
   - Software development methodology
5. Quality assurance plan
   - Project management plan (Organization, Work Breakdown structure, Tasks, Milestones)
   - Configuration Management plan
   - Verification and Validation plan

Siconos/Numerics

Independent collection of solvers in C for standard nonsmooth problem :

- LCP/MLCP/Relay
  - Lemke's method, PSOR, PGS, Enumerative (based on simplex), Semismooth newton, ...
- MCP/VI
  - projection/splitting methods, interface to PATH solver, semismooth Newton based on fischer-Burmeister formulation.
- FrictionContact
  - Nonsmooth newton (Alart-Curnier, Christensen et al.), PGS with local solvers, Extragradient, hyperplane, projection/splitting methods, optimization based on Tresca's formulation, ...
- QP
- ODE/DAE integrators:
  - Lsode suite with LSODAR (Hindmarsh, Alan C., (LLNL))
  - HEM5 DAE solver (Hairer, Ernst, Université de Genève)

Modeling Principle:

Nonsmooth Dynamical System

Dynamical system

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

Modeling Principle:

Nonsmooth Dynamical System

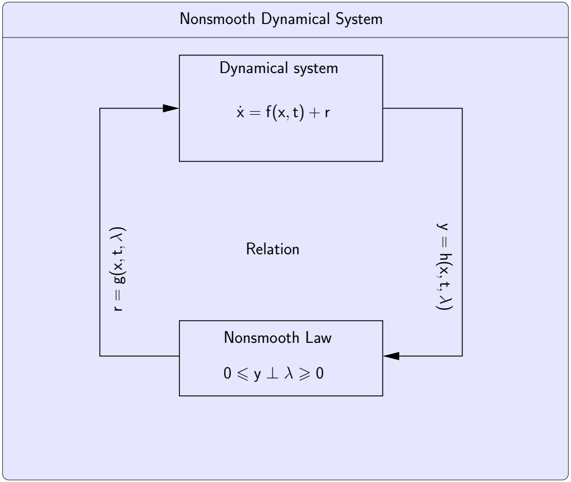
Dynamical system

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

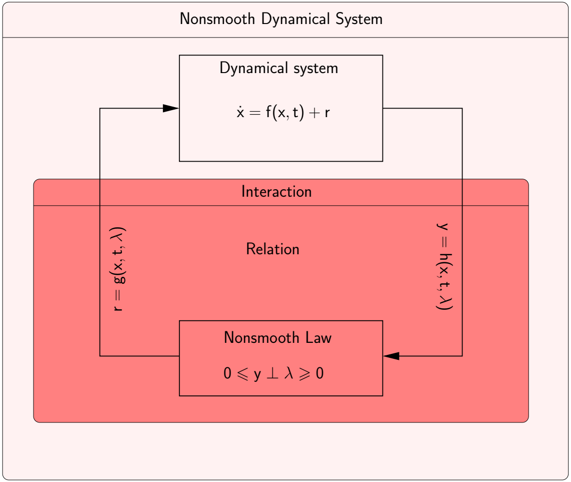
Nonsmooth Law

$$0 \leqslant y \perp \lambda \geqslant 0$$

Modeling Principle:



Modeling Principle:

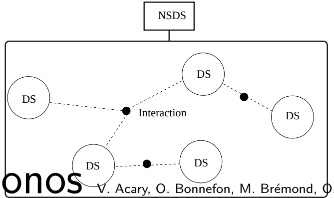


Kernel Modeling Part

A Nonsmooth Dynamical System :

a directed graph of Dynamical systems and Interactions

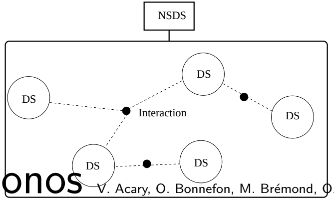
- **DynamicalSystem**: a set of ODEs
- **Interaction**: a set of input/output relations and a non-smooth law



Kernel Modeling Part

A Nonsmooth Dynamical System :

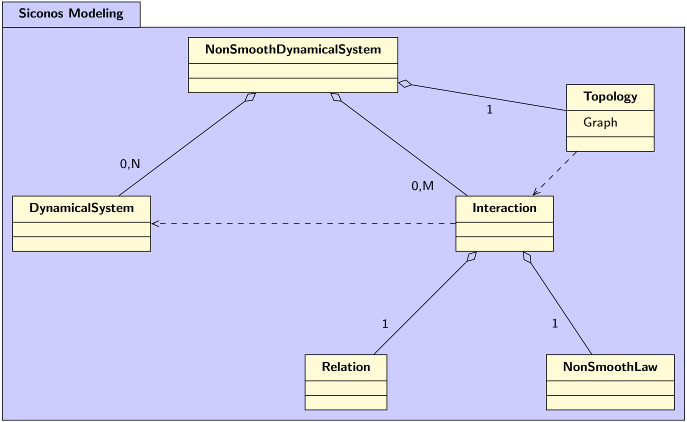
a directed graph of Dynamical systems and Interactions



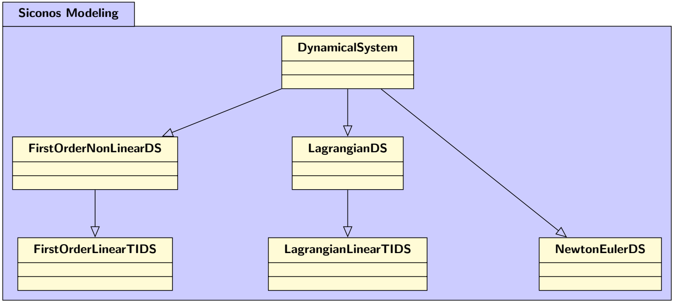
- **DynamicalSystem**: a set of ODEs
- **Interaction**: a set of input/output relations and a non-smooth law
- **Topology**: A directed graph that links the dynamical systems with the Interaction and that handles relative degrees, index sets ...

Kernel Modeling Part

Simplified Modeling Tools class diagram:

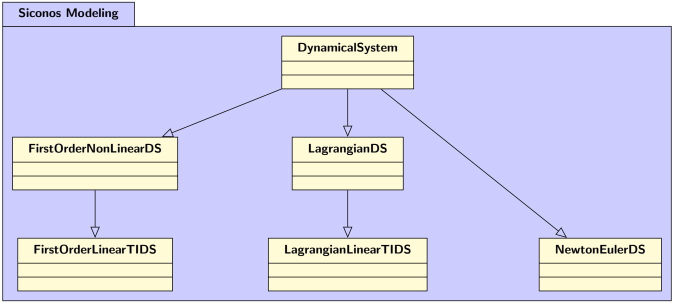


Dynamical Systems in Siconos/Kernel



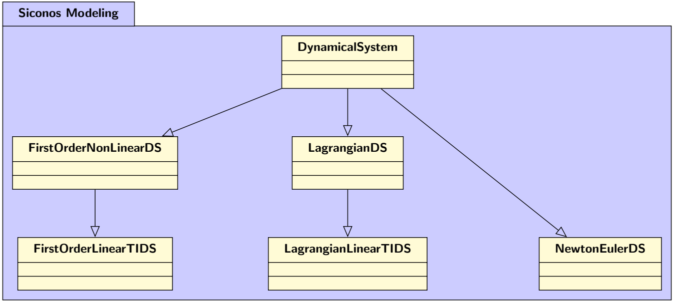
- Parent Class **DynamicalSystem** $\quad g(\dot{x}, x, t, z) = 0$

Dynamical Systems in Siconos/Kernel



- Parent Class **DynamicalSystem** $\quad g(\dot{x}, x, t, z) = 0$
  - **FirstOrderNonLinearDS** Linear Dynamical Systems
    $$M\dot{x} = f(x, t, z) + r$$
  - **FirstOrderLinearDS** Linear Dynamical Systems
    $$M\dot{x} = A(t, z)x + b(t, z) + r$$

Dynamical Systems in Siconos/Kernel



- Parent Class **DynamicalSystem** $g(\dot{x}, x, t, z) = 0$
- Derived Classes
  - **LagrangianDS** Lagrangian Dynamical Systems

$$M(q)\ddot{q} + NNL(q, \dot{q}) + F_{int}(\dot{q}, q, t) = F_{ext}(t) + T(q)u(q, t) + p$$

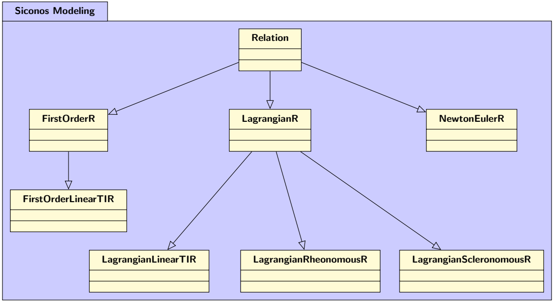
  - **LagrangianLinearTIDS** Lagrangian Linear Time Invariant Systems

$$M\ddot{q} + C\dot{q} + Kq = F_{ext}(t) + Tu(t) + p$$

  - **NewtonEulerDS** Newton/Euler Systems

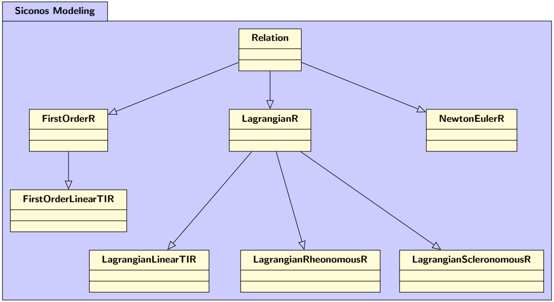
Note: all operators ($f(x, t)$, $M(q)$, ...) can be set either as matrices (when constant) or with a user-defined external function (plug-in).

Relations



- Parent Class **Relation** $\quad y = h(x, t, \lambda, z), r = g(\lambda, t, x, z)$

Relations



- Parent Class **Relation** $y = h(x, t, \lambda, z), r = g(\lambda, t, x, z)$
- Derived Classes:
  - **FirstOrderLinearTIR** First Order LTI Relation

$$y = Cx + Fu + D\lambda + e, \quad r = B\lambda$$

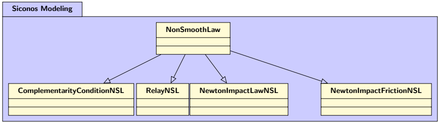
  - **LagrangianR** Lagrangian Relation

$$\dot{y} = H(q, t, \ldots)\dot{q}, \quad p = H^t(q, t, \ldots)\lambda$$

  - **LagrangianLinearR** Lagrangian Linear Relation

$$\dot{y} = H\dot{q} + b, \quad p = H^t\lambda$$

Non Smooth laws



- Parent Class **NonSmoothLaw**
- Derived Classes
  - **ComplementarityConditionNSL** Complementarity condition or unilateral contact

$$0 \leqslant y \perp \lambda \geqslant 0$$

  - **Relay** condition.

$$\begin{cases} \dot{y} = 0, |\lambda| \leqslant 1 \\ \dot{y} \neq 0, \lambda = \text{sign}(y) \end{cases}$$

  - **NewtonImpactLawNSL** Newton impact Law.

$$\text{if } y(t) = 0, \quad 0 \leqslant \dot{y}(t^+) + e\dot{y}(t^-) \perp \lambda \geqslant 0$$

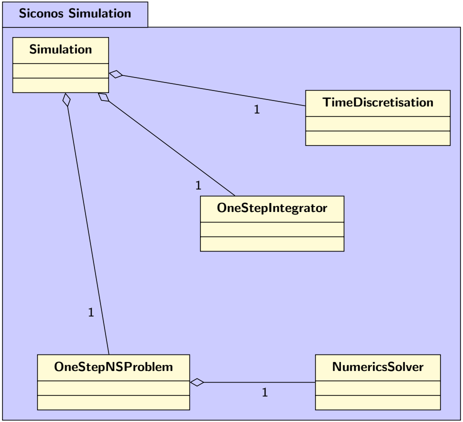
  - **NewtonImpactFrictionNSL** Newton impact and Friction (Coulomb) Law.

```
t0 = 0       # start time
T = 10       # end time
h = 0.005    # time step
r = 0.1      # ball radius
g = 9.81     # gravity
m = 1        # ball mass
e = 0.9      # restitution coeficient
theta = 0.5 # theta scheme
#
# dynamical system
#
x = [1,0,0] # initial position
v = [0,0,0] # initial velocity
mass = eye(3)       # mass matrix
mass[2,2]=3./5 * r * r
# the dynamical system
ball = LagrangianLinearTIDS(x, v, mass)

# set external forces
weight = [-m * g, 0, 0]
ball.setFExtPtr(weight)
```

```python
# Interactions
#
# ball-floor
H = [[1,0,0]]
nslaw = NewtonImpactNSL(e)
relation = LagrangianLinearTIR(H)
inter = Interaction(1, nslaw, relation)

#
# Model
#
bouncingBall = Model(t0,T)

# add the dynamical system to the non smooth dynamical system
bouncingBall.nonSmoothDynamicalSystem().insertDynamicalSystem(ball)

# link the interaction and the dynamical system
bouncingBall.nonSmoothDynamicalSystem().link(inter,ball);
```

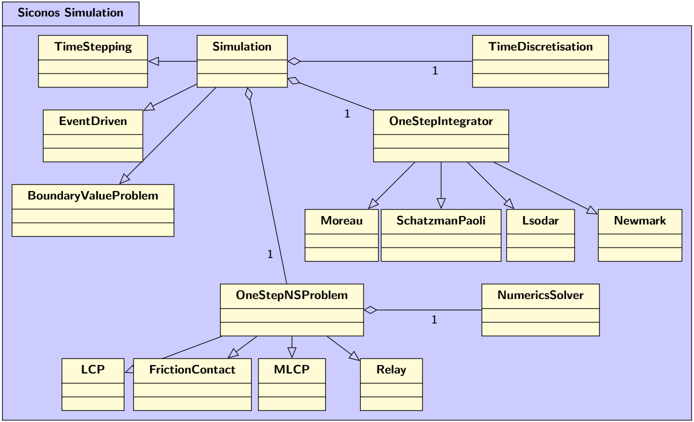
Kernel Simulation Part

Simplified Modeling Tools class diagram:



Kernel Simulation Part

Simplified Modeling Tools class diagram:



OneStepIntegrator:

- **Moreau**: Moreau–Jean Time-stepping integrator
- **SchatzmanPaoli**: Schatzman–Paoli Time-stepping integrator
- **D1MinusLinear**: Time–Discontinuous Galerkin method.
- **Lsodar**: Numerical integration scheme based on the Livermore Solver for Ordinary Differential Equations with root finding.
- **HEM5**: Half–explicit method of Brasey & Hairer for index-2 mechanical systems.

OnestepNSproblem: Numerical one step non smooth problem formulation and solver.

- **LCP** Linear Complementarity Problem

$$\begin{cases} w = Mz + q \\ 0 \leqslant w \perp z \geqslant 0 \end{cases}$$

- **FrictionContact** Two(three)-dimensional contact friction problem
- **QP** Quadratic programming problem

$$\begin{cases} \min \frac{1}{2} z^T Q z + z^T p \\ z \geqslant 0 \end{cases}$$

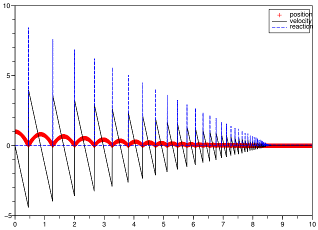
- **Relay**

```
# (1) OneStepIntegrators
OSI = Moreau(theta)
OSI.insertDynamicalSystem(ball)

# (2) Time discretisation --
t = TimeDiscretisation(t0,h)

# (3) one step non smooth problem
osnspb = LCP()

# (4) Simulation setup with (1) (2) (3)
s = TimeStepping(t)
s.insertIntegrator(OSI)
s.insertNonSmoothProblem(osnspb)
```

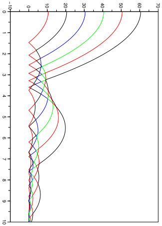
```python
#
# computation
#
# simulation initialization
bouncingBall.initialize(s)
# time loop
while(s.nextTime() < T):
    s.computeOneStep()
    s.nextStep()
    print s.nextTime()
```

Model: Lagrangian Linear Time Invariant Dynamical Systems with Lagrangian Linear Relations, Newton Impact Law.
Simulation: Moreau's Time Stepping or Event Driven.

Bouncing Ball

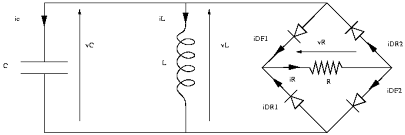


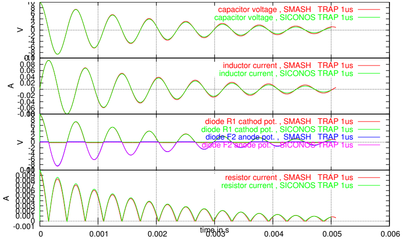
Beads column



A 4 diodes bridge wave rectifier.

Model: Linear Dynamical System with Linear Relations, Complementarity Condition Non Smooth Law.
Simulation: Moreau's Time Stepping

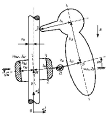


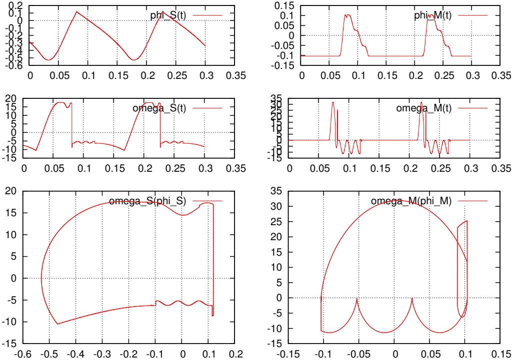


Comparison between the SICONOS Platform (Non Smooth LCS model) and SPICE simulator (Smooth Diode model).

Woodpecker toy (sample from Michael Moeller (CR10))

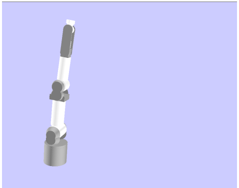
Model: Lagrangian Linear Dynamical System, Lagrangian Linear Relations, Newton impact-friction law.
Simulation: Moreau's Time Stepping

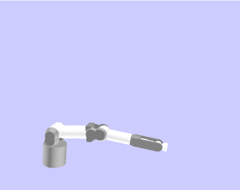


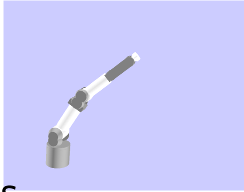


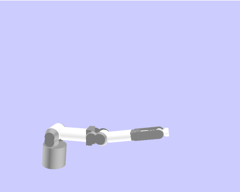
A Robotic Arm (Pa10)

Model: Lagrangian Non Linear Dynamical System with Lagrangian Non Linear Relations, Newton impact.
Simulation: Moreau's Time Stepping









Proximity detection

- threshold bounding box
- spatial hashing of the bounding box

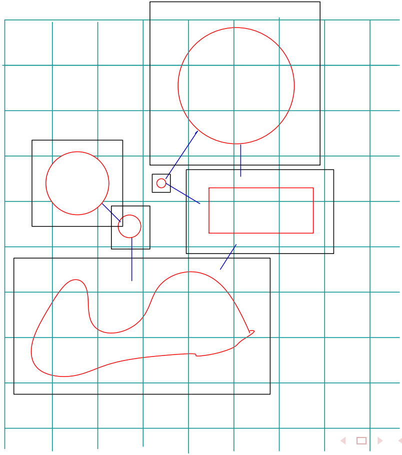


Figure: proximity detection with spatial hashing

Siconos internal graphs

- dynamical systems as nodes, interactions as edges
- interactions as nodes, dynamical systems as edges

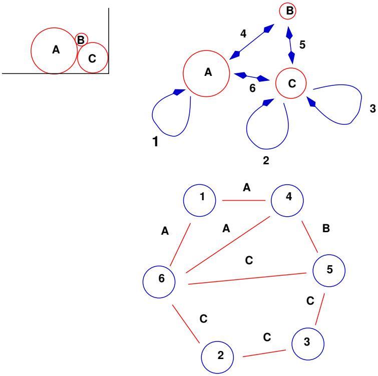


Figure: adjoint graph construction

Newton Euler Formalism

Dynamical system of a rigid body

$$
\begin{array}{rcl}
q & = & (x_G, Q)^T, \\
\begin{pmatrix} M\dot{v}_G \\ I\Omega + \Omega \times I\Omega \end{pmatrix} & = & \begin{pmatrix} F_{ext}(t, q, \Omega, v_G) \\ M_{ext}(t, q, \Omega, v_G) \end{pmatrix} + R, \\
v_G & = & \dot{x}_G, \\
\dot{q} & = & T(q)(v_G, \Omega)^T
\end{array}
$$

- $q \in \mathbb{R}^7$: absolute coordinates vector.
- $x_G \in \mathbb{R}^3$: coordinates of the center of mass.
- $Q \in \mathbb{R}^4$: unit quaternion representing the absolute orientation.
- $\Omega \in \mathbb{R}^3$: angular speed vector relative to the solid.
- $M = mI_{3x3}$: mass matrix.
- $I \in \mathbb{R}^{3\times 3}$: inertia matrix.
- $F_{ext}(t, q, \Omega, v_G) : \mathbb{R} \times \mathbb{R}^7 \times \mathbb{R}^3 \times \mathbb{R}^3 \mapsto \mathbb{R}^6$: given external forces,
- $M_{ext}(t, q, \Omega, v_G) : \mathbb{R} \times \mathbb{R}^7 \times \mathbb{R}^3 \times \mathbb{R}^3 \mapsto \mathbb{R}^6$: given external moments,
- $R \in \mathbb{R}^6$ is the force due the non smooth law.
- $T(q) \in \mathbb{R}^{6\times 7}$

Newton Euler Formalism

Constraints between two rigid bodies

$$y = h(q_1, q_2)$$

$$\dot{y} = \nabla_q h^T(q) \begin{pmatrix} T(q_1) & 0 \\ 0 & T(q_2) \end{pmatrix} (v_{1G}, \Omega_1, v_{2G}, \Omega_2)^T$$

with the contact law

- $y = 0$ in the case of a joint.
- $0 \leqslant y \perp \lambda \geqslant 0$ in the case of an unilateral constraint.
- NonSmoothLaw(y, λ) in more general case

The reaction force R

$$R = \begin{pmatrix} T(q_1)^T & 0 \\ 0 & T(q_2)^T \end{pmatrix} \nabla_q h(q)\, \lambda$$

Coupling with the 3D modeling library, Open CASCADE.

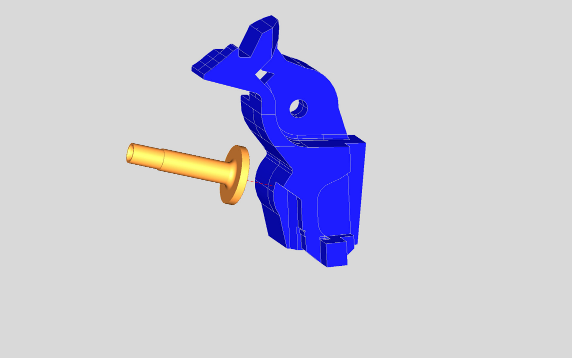


Figure: Parts of a Circuit breakers (Schneider Electric).

Open CASCADE provides the following features:

- To load a mechanism from CAD files (step, iges...).
- To compute the geometrical informations needed by the Nonsmoothlaw.

It allows to simulate industrial mechanisms using the Siconos technology.

Picker example.

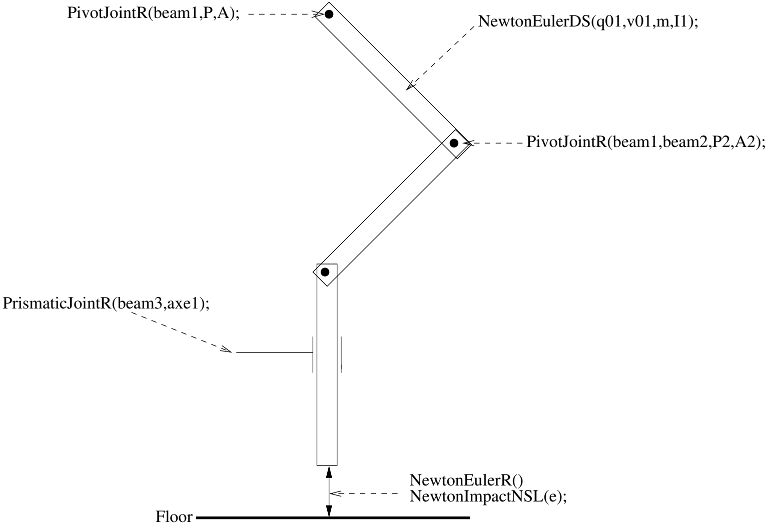


Figure: A simple example.

Circuit breakers example.

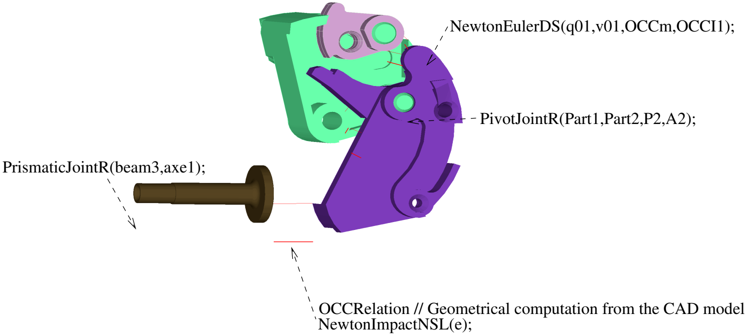


Figure: Modeling of a Circuit breakers using SICONOS and Open CASCADE.

- The matrices of mass and the geometrical informations are computed from the geometrical model.

Help and Documentation

- Doxygen tools for automatic documentation in Numerics and Kernel
- Users, developers and theoretical manuals (in progress ...)
- Web pages, Bug tracker, forum ... on Gforge.
- Examples library as templates (more than 60 simple examples).

Diffusion

- The SICONOS platform is distributed under GPL licence.
- Visit the Gforge Web site for
  - Documentations
  - Mailing lists
  - Downloads
  - Bug tracker
  - Contributing, . . .

  http://gforge.inria.fr/projects/siconos/