Developer's Notes

Siconos Development Team

July 5, 2017

Contents

Chapter 1

OneStepNSProblem formalisation for several interactions

author	F. Pérignon
date	May 16, 2006
version	?

1.1 LinearDS - Linear Time Invariant Relations

1.1.1 General notations

We consider n dynamical systems of the form:

$$\dot{x}_i = A_i x_i + R_i \tag{1.1}$$

Each system if of dimension n_i, and we denote $N = \sum_{i=1}^{n} n_i$.
An interaction, I_α is composed with a non smooth law, $nslaw_\alpha$ and a relation:

$$y_\alpha = C_\alpha X_\alpha + D_\alpha \lambda_\alpha \tag{1.2}$$

The "dimension" of the interaction, ie the size of vector y_α, is denoted m_α and we set:

$$M = \sum_{\alpha=1}^{m} m_\alpha$$

m being the number of interactions in the Non Smooth Dynamical System.
X_α is a vector that represents the DS concerned by the interaction. Its dimension is noted N_α, this for n_α systems in the interaction.
C_α is a $m_\alpha \times N_\alpha$ row-blocks matrix and D_α a $m_\alpha \times m_\alpha$ square matrix.

$$C_\alpha = \begin{bmatrix} C_\alpha^i & C_\alpha^j & \dots \end{bmatrix} \tag{1.3}$$

with $i, j, \dots \in \mathcal{DS}_\alpha$ which is the set of DS belonging to interaction α.
We also have the following relation:

$$\begin{bmatrix} R_\alpha^i \\ R_\alpha^j \\ \dots \end{bmatrix} = B_\alpha \lambda_\alpha = \begin{bmatrix} B_\alpha^i \\ B_\alpha^j \\ \dots \end{bmatrix} \lambda_\alpha \tag{1.4}$$

R_α^i represents the contribution of interaction α on the reaction of the dynamical system i, and B_α^i is a $n_i \times m_\alpha$ block matrix.
And so:

$$R_i = \sum_{\beta \in \mathcal{I}_i} R_\beta^i = \sum_{\beta \in \mathcal{I}_i} B_\beta^i \lambda_\beta \tag{1.5}$$

with $\mathcal{I}_i$ the set of interactions in which dynamical system number i is involved.
Introducing the time discretization, we get:

$$x_i^{k+1} - x_i^k = hA_i x_i^{k+1} + hR_i^{k+1} \tag{1.6}$$

$$y_\alpha^{k+1} = C_\alpha X_\alpha^{k+1} + D_\alpha \lambda_\alpha^{k+1} \tag{1.7}$$

$$R_i^{k+1} = \sum_{\beta \in \mathcal{I}_i} B_\beta^i \lambda_\beta^{k+1} \tag{1.8}$$

ie, with $W_i = (I - hA_i)^{-1}$:

$$x_i^{k+1} = W_i x_i^k + hW_i R_i^{k+1} \tag{1.9}$$

$$y_\alpha^{k+1} = C_\alpha W_\alpha X_\alpha^k + C_\alpha h W_\alpha \sum_{\beta \in \mathcal{I}_i} B_\beta^i \lambda_\beta^{k+1} + D_\alpha \lambda_\alpha^{k+1} \tag{1.10}$$

$$= C_\alpha W_\alpha X_\alpha^k + (C_\alpha h W_\alpha B_\alpha + D_\alpha)\lambda_\alpha^{k+1} + \sum_{\beta \neq \alpha} \Big(\sum_{i \in \mathcal{DS}_\alpha \cap \in \mathcal{DS}_\beta} h C_\alpha^i W_i B_\beta^i \lambda_\beta^{k+1} \Big) \tag{1.11}$$

with

$$W_\alpha = \begin{bmatrix} W_i & 0 & \dots \\ 0 & W_j & \dots \\ 0 & \dots & \dots \end{bmatrix} \tag{1.12}$$ {Walpha}

the block-diagonal matrix of all the W for the dynamical systems involved in interaction α.
The global-assembled Y vector, of dimension M, composed by m y_α subvectors, is given by:

$$Y_{k+1} = q_{OSNSP} + M_{OSNSP} \Lambda_{k+1} \tag{1.13}$$

or,

$$Y_{k+1} = \begin{bmatrix} y_1 \\ \dots \\ y_m \end{bmatrix}_{k+1} = \begin{bmatrix} C_1^1 & \dots & C_1^n \\ \vdots & \dots & \vdots \\ C_m^1 & \dots & C_m^n \end{bmatrix} \begin{bmatrix} W_1 & 0 & \dots & 0 \\ 0 & W_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ & & 0 & W_n \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{bmatrix}_k \tag{1.14}$$

$$+ \begin{bmatrix} D_1 + h \sum_{j \in \mathcal{DS}_1} C_1^j W_j B_1^j & h \sum_{j \in \mathcal{DS}_1 \cap \mathcal{DS}_2} C_1^j W_j B_2^j & \dots \\ \vdots & \ddots & \\ & h \sum_{j \in \mathcal{DS}_m} C_m^j W_j B_{m-1}^j & D_m + h \sum_{j \in \mathcal{DS}_m \cap \mathcal{DS}_{m-1}} C_m^j W_j B_m^j \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{bmatrix}_{k+1}$$

To sum it up, the block-diagonal term of matrix M_{OSNSP}, for block-row α is:

$$D_\alpha + h \sum_{j \in \mathcal{DS}_\alpha} C_\alpha^j W_j B_\alpha^j \tag{1.15}$$

This is an $m_\alpha \times m_\alpha$ square matrix. The extra-diagonal block term, in position (α, β) is:

$$h \sum_{j \in \mathcal{DS}_\alpha \cap \mathcal{DS}_\beta} C_\alpha^j W_j B_\beta^j \tag{1.16}$$

and is a $m_\alpha \times m_\beta$ matrix. This matrix differs from 0 when interactions α and β are coupled, ie have common DS.

Or, for the relation l of interaction α, we get:

$$D_{\alpha,l} + h \sum_{j \in \mathcal{DS}_\alpha} C^j_{\alpha,l} W_j B^j_\alpha \tag{1.17}$$

for the diagonal, and

$$h \sum_{j \in \mathcal{DS}_\alpha \cap \mathcal{DS}_\beta} C^j_{\alpha,l} W_j B^j_\beta \tag{1.18}$$

for extra-diagonal terms.
$D_{\alpha,l}$, row number l of D_α, the same for $C_{\alpha,l}$

Finally, the linked-Interaction map provides, for each interaction (named "current interaction"), the list of all the interactions (named "linked interaction") that have common dynamical system with the "current interaction".

1.1.2 A simple example

We consider $n = 3$ dynamical systems and $m = 2$ interactions:

$$\begin{aligned} I_\mu &\rightarrow \mathcal{DS}_\mu = \{DS_1, DS_3\}, m_\mu = 3 \\ I_\theta &\rightarrow \mathcal{DS}_\theta = \{DS_2, DS_3\}, m_\theta = 1 \end{aligned}$$

The linked-interaction map is :

$$\begin{aligned} I_\mu &\rightarrow I_\theta, commonDS = DS_3 \\ I_\theta &\rightarrow I_\mu, commonDS = DS_3 \end{aligned}$$

And:

$$\begin{aligned} M &= 4, N = \sum_{i=1}^{3} n_i \\ \mathcal{I}_1 &= \{I_\mu\} \\ \mathcal{I}_2 &= \{I_\theta\} \\ \mathcal{I}_3 &= \{I_\mu, I_\theta\} \end{aligned}$$

$$y_1 = \begin{bmatrix} C^1_1 & C^3_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_3 \end{bmatrix} + D_1 \lambda_1 \tag{1.19}$$

$$y_2 = \begin{bmatrix} C^2_2 & C^3_2 \end{bmatrix} \begin{bmatrix} x_2 \\ x_3 \end{bmatrix} + D_2 \lambda_2 \tag{1.20}$$

$$\begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} = \begin{bmatrix} B^1_1 \lambda_1 \\ B^2_2 \lambda_2 \\ B^3_1 \lambda_1 + B^3_2 \lambda_2 \end{bmatrix} \tag{1.21}$$

$$M_{OSNSP} = \begin{bmatrix} D_1 + hC^1_1 W_1 B^1_1 + hC^3_1 W_3 B^3_1 & hC^3_1 W_3 B^3_2 \\ hC^3_2 W_3 B^3_1 & D_2 + hC^2_2 W_2 B^2_2 + hC^3_2 W_3 B^3_2 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}_{k+1} \tag{1.22}$$

1.1.3 relative degree

Let us consider the global vector

$$Y = \begin{bmatrix} y_1 \\ ... \\ y_M \end{bmatrix} = CX + D\Lambda \tag{1.23}$$

We denote by r_j the relative degree of equation j, $j \in [1..M]$. We have:

$$y_j = \sum_{i=1}^{n} C_j^i x_i + D_{j,j}\lambda_j + \sum_{i\neq j, i=1}^{m} D_{j,i}\lambda_i \tag{1.24}$$

$D_{j,i}$ a scalar and C_j^i a $1 \times n_i$ line-vector.
If $D_{jj} \neq 0$, then $r_j = 0$. Else, we should consider the first derivative of y_j.
Before that, recall that:

$$R_i = \sum_{k=1}^{M} B_k^i \lambda_j \tag{1.25}$$

Through many of the B_j^i are equal to zero, we keep them all in the following lines.
Then:

$$\begin{aligned} \dot{y}_j &= \sum_{i=1}^{n} C_j^i (A_i x_i + \sum_{k=1}^{M} B_k^i \lambda_k) + f(\lambda_k)_{k\neq j} && (1.26)\\ &= \sum_{i=1}^{n} C_j^i (A_i x_i + B_j^i \lambda_j + \sum_{k=1, k\neq j}^{M} B_k^i \lambda_k) + \dots && (1.27) \end{aligned}$$

So, if $\sum_{i=1}^{n} C_j^i B_j^i \neq 0$ (note that this corresponds to the product between line j of C and column j of B) then $r_j = 1$ else we consider the next derivative, and so on.
In derivative r, the coefficient of λ_j will be:

$$coeff_j = \sum_{i=1}^{n} C_j^i (A_i)^{r-1} B_j^i \tag{1.28}$$

if $coeff_j \neq 0$ then $r_j = r$.

1.2 LagrangianDS - Lagrangian Linear Relations

1.2.1 General notations

We consider n dynamical systems, lagrangian and non linear, of the form:

$$M_i(q_i)\ddot{q}_i + N_i(\dot{q}_i, q_i) = F_{Int,i}(\dot{q}_i, q_i, t) + F_{Ext,i}(t) + p_i \tag{1.29}$$

Each system if of dimension n_i, and we denote $N = \sum_{i=1}^{n} n_i$.
An interaction, I_α is composed with a non smooth law, $nslaw_\alpha$ and a relation:

$$y_\alpha = H_\alpha Q_\alpha + b_\alpha \tag{1.30}$$

The "dimension" of the interaction, ie the size of vector y_α, is denoted m_α and we set:

$$M_y = \sum_{\alpha=1}^{m} m_\alpha$$

m being the number of interactions in the Non Smooth Dynamical System.
Q_α is a vector that represents the DS concerned by the interaction. Its dimension is noted N_α, this for n_α systems in the interaction.
H_α is a $m_\alpha \times N_\alpha$ row-blocks matrix and b_α a m_α vector.

$$H_\alpha = \left[\begin{array}{cccc} H_\alpha^i & H_\alpha^j & ... \end{array} \right] \tag{1.31}$$

with $i, j, ... \in \mathcal{DS}_\alpha$ which is the set of DS belonging to interaction α.
We also have the following relation:

$$\left[\begin{array}{c} R_\alpha^i \\ R_\alpha^j \\ ... \end{array} \right] = {}^tH_\alpha \lambda_\alpha = \left[\begin{array}{c} {}^tH_\alpha^i \\ {}^tH_\alpha^j \\ ... \end{array} \right] \lambda_\alpha \tag{1.32}$$

R_α^i represents the contribution of interaction α on the reaction of the dynamical system i, and ${}^tH_\alpha^i$ is a $n_i \times m_\alpha$ block matrix.
And so:

$$R_i = \sum_{\beta \in \mathcal{I}_i} R_\beta^i = \sum_{\beta \in \mathcal{I}_i} H_\beta^i \lambda_\beta \tag{1.33}$$

with $\mathcal{I}_i$ the set of interactions in which dynamical system number i is involved.
Introducing the time dicretisation, we get:

$$\begin{aligned} \dot{q}_i^{k+1} &= \dot{q}_{free,i} + W_i R_i^{k+1} \\ y_\alpha^{k+1} &= H_\alpha \dot{Q}_\alpha^{k+1} \end{aligned} \tag{1.34}$$

$$R_i^{k+1} = \sum_{\beta \in \mathcal{I}_i} H_\beta^i \lambda_\beta^{k+1} \tag{1.35}$$

ie,

$$y_\alpha^{k+1} = H_\alpha Q_\alpha^{free} + H_\alpha W_\alpha {}^tH_\alpha \lambda_\alpha + \sum_{i \in \mathcal{DS}_\alpha} \sum_{\beta \in \mathcal{I}_i, \alpha \neq \beta} H_\alpha^i W_i H_\beta^j \lambda_\beta \tag{1.36}$$

with W_α given by (1.12).

The global-assembled Y vector, of dimension M, composed by m y_α subvectors, is given by:

$$Y_{k+1} = q_{OSNSP} + M_{OSNSP} \Lambda_{k+1} \tag{1.37}$$

with:

$$q_{OSNSP}^\alpha = H_\alpha Q_\alpha^{free} \tag{1.38}$$

and for M_{OSNSP}, the block-diagonal term for block-row α is

$$\sum_{j \in \mathcal{DS}_\alpha} H_\alpha^j W_j {}^tH_\alpha^j \tag{1.39}$$

an $m_\alpha \times m_\alpha$ square matrix. The extra-diagonal block term, in position (α, β) is:

$$\sum_{j \in \mathcal{DS}_\alpha \cap \mathcal{DS}_\beta} H_\alpha^j W_j {}^tH_\beta^j \tag{1.40}$$

and is a $m_\alpha \times m_\beta$ matrix. This matrix differs from 0 when interactions α and β are coupled, ie have common DS.

Or, for the relation l of interaction α, we get:

$$\sum_{j \in \mathcal{DS}_\alpha} H^j_{\alpha,l} {W_j}^t H^j_\alpha \tag{1.41}$$

for the diagonal, and

$$\sum_{j \in \mathcal{DS}_\alpha \cap \mathcal{DS}_\beta} H^j_{\alpha,l} {W_j}^t H^j_\beta \tag{1.42}$$

for extra-diagonal terms.
$H_{\alpha,l}$, row number l of H_α.

WARNING: depending on linear and non linear case for the DS, there should be a factor h ahead W. See Bouncing Ball template.

1.3 Block matrix approach

The built of the OSNSProblem matrix could be computed using block matrix structure. This section describe these matrices. It is not implemented in Siconos. Using previous notations, n is the number of DS. m the number of interations.

1.3.1 Block matrix of DS

$$\boldsymbol{M}\dot{\boldsymbol{X}} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{R}$$

where $\boldsymbol{M} = diag(M_1, ...M_n)$ and $\boldsymbol{A} = diag(A_1, .., A_n)$.

$$\boldsymbol{R} = \boldsymbol{B}\boldsymbol{\lambda}$$

$$\boldsymbol{B} = \begin{pmatrix} B^1_1 ... B^1_m \\ . \\ . \\ B^n_1 ... B^n_m \end{pmatrix}$$

Some of B^i_j doesn't exist.

1.3.2 Block matrix of interaction

$$\boldsymbol{Y} = \boldsymbol{C}\boldsymbol{X} + \boldsymbol{D}\boldsymbol{\lambda}$$

with $\boldsymbol{D} = diag(D_1..D_m)$ and

$$\boldsymbol{C} = \begin{pmatrix} C^1_1 .. C^n_1 \\ . \\ . \\ C^1_m ... C^n_m \end{pmatrix}$$

Some of C^i_j doesn't exist.

1.3.3 OSNSProblem using block matrices

The Matrix of the OSNS Problem could be assembled using the following block-product-matrices $\boldsymbol{CWB}$.

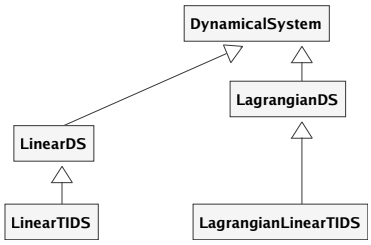
Chapter 2

Dynamical Systems formulations in Siconos.

author	F. Pérignon
date	March 22, 2006
version	Kernel 1.1.4

2.1 Class Diagram

There are four possible formulation for dynamical systems in Siconos, two for first order systems and two for second order Lagrangian systems. The main class is DynamicalSystem, all other derived from this one, as shown in the following diagram:



{DSDiagram}

2.2 General non linear first order dynamical systems $\rightarrow$ class *DynamicalSystem*

This is the top class for dynamical systems. All other systems classes derived from this one.

A general dynamical systems is described by the following set of n equations, completed with initial conditions:

$$\dot{x} = f(x,t) + T(x)u(x,\dot{x},t) + r \quad (2.1)$$

$$x(t_0) = x_0 \quad (2.2)$$

- x: state of the system - Vector of size n.
- $f(x,t)$: vector field - Vector of size n.

- $u(x, \dot{x}, t)$: control term - Vector of size $uSize$.
- $T(x)$: $n \times uSize$ matrix, related to control term.
- r: input due to non-smooth behavior - Vector of size n.

The Jacobian matrix, $\nabla_x f(x,t)$, of f according to x, $n \times n$ square matrix, is also a member of the class.

Initial conditions are given by the member x_0, vector of size n. This corresponds to x value when simulation is starting, ie after a call to strategy->initialize().

There are plug-in functions in this class for f (vectorField), $jacobianX$, u and T. All of them can handle a vector of user-defined parameters.

2.3 First order linear dynamical systems $\rightarrow$ class *LinearDS*

Derived from DynamicalSystem, described by the set of n equations and initial conditions:

$$\dot{x} = A(t)x(t) + Tu(t) + b(t) + r \tag{2.3}$$
$$x(t_0) = x_0 \tag{2.4}$$

With:

- $A(t)$: $n \times n$ matrix, state independent but possibly time-dependent.
- $b(t)$: Vector of size n, possibly time-dependent.

Other variables are those of DynamicalSystem class.
A and B have corresponding plug-in functions.

Warning: time dependence for A and b is not available at the time in the simulation part for this kind of dynamical systems.

Links with vectorField and its Jacobian are:

$$f(x,t) = A(t)x(t) + b(t) \tag{2.5}$$
$$jacobianX = \nabla_x f(x,t) = A(t) \tag{2.6}$$

2.4 Second order non linear Lagrangian dynamical systems $\rightarrow$ class *LagrangianDS*

Lagrangian second order non linear systems are described by the following set of$nDof$ equations + initial conditions:

$$M(q)\ddot{q} + NNL(\dot{q}, q) + F_{Int}(\dot{q}, q, t) = F_{Ext}(t) + p \tag{2.7}$$
$$q(t_0) = q0 \tag{2.8}$$
$$\dot{q}(t_0) = velocity0 \tag{2.9}$$

With:

- $M(q)$: $nDof \times nDof$ matrix of inertia.
- q: state of the system - Vector of size $nDof$.
- $\dot{q}$ or $velocity$: derivative of the state according to time - Vector of size $nDof$.

- $NNL(\dot{q}, q)$: non linear terms, time-independent - Vector of size $nDof$.
- $F_{Int}(\dot{q}, q, t)$: time-dependent linear terms - Vector of size $nDof$.
- $F_{Ext}(t)$: external forces, time-dependent BUT do not depend on state - Vector of size $nDof$.
- p: input due to non-smooth behavior - Vector of size $nDof$.

The following Jacobian are also member of this class:

- jacobianQFInt = $\nabla_q F_{Int}(t, q, \dot{q})$ - $nDof \times nDof$ matrix.
- jacobianVelocityFInt = $\nabla_{\dot{q}} F_{Int}(t, q, \dot{q})$ - $nDof \times nDof$ matrix.
- jacobianQNNL = $\nabla_q NNL(q, \dot{q})$ - $nDof \times nDof$ matrix.
- jacobianVelocityNNL = $\nabla_{\dot{q}} NNL(q, \dot{q})$ - $nDof \times nDof$ matrix.

There are plug-in functions in this class for F_{int}, F_{Ext}, M, NNL and the four Jacobian matrices. All of them can handle a vector of user-defined parameters.

Links with first order dynamical system are:

$$n = 2nDof \tag{2.10}$$

$$x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} \tag{2.11}$$

$$f(x,t) = \begin{bmatrix} \dot{q} \\ M^{-1}(F_{Ext} - F_{Int} - NNL) \end{bmatrix} \tag{2.12}$$

$$\tag{2.13}$$

$$\nabla_x f(x,t) = \begin{bmatrix} 0_{nDof \times nDof} & I_{nDof \times nDof} \\ \nabla_q(M^{-1})(F_{Ext} - F_{Int} - NNL) - M^{-1}\nabla_q(F_{Int} + NNL) & -M^{-1}\nabla_{\dot{q}}(F_{Int} + NNL) \end{bmatrix} \tag{2.14}$$

$$r = \begin{bmatrix} 0_{nDof} \\ p \end{bmatrix} \tag{2.15}$$

$$u(x, \dot{x}, t) = u_L(\dot{q}, q, t) \text{ (not yet implemented)} \tag{2.16}$$

$$T(x) = \begin{bmatrix} 0_{nDof} \\ T_L(q) \end{bmatrix} \text{ (not yet implemented)} \tag{2.17}$$

$$\tag{2.18}$$

With 0_n a vector of zero of size n, $0_{n \times m}$ a $n \times m$ zero matrix and $I_{n \times n}$, identity $n \times n$ matrix.

Warning: control terms (Tu) are not fully implemented in Lagrangian systems. This will be part of future version.

2.5 Second order linear and time-invariant Lagrangian dynamical systems → class *LagrangianLinearTIDS*

{Sec:LagrangianL

$$M\ddot{q} + C\dot{q} + Kq = F_{Ext}(t) + p \tag{2.19}$$

With:

- C: constant viscosity $nDof \times nDof$ matrix
- K: constant rigidity $nDof \times nDof$ matrix

And:

$$F_{Int} = C\dot{q} + Kq \tag{2.20}$$

$$NNL = 0_{nDof} \tag{2.21}$$

Chapter 3

Dynamical Systems implementation in Siconos.

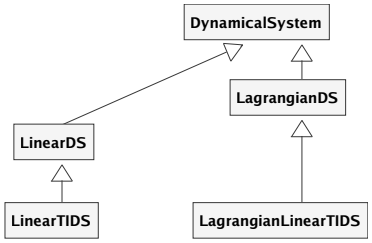
author	F. Pérignon
date	November 7, 2006
version	Kernel 1.3.0

3.1 Introduction

This document is only a sequel of notes and remarks on the way Dynamical Systems are implemented in Siconos.
It has to be completed, reviewed, reorganized etc etc for a future Developpers'Guide.
See also documentation in Doc/User/DynamicalSystemsInSiconos for a description of various dynamical systems types.

3.2 Class Diagram

There are four possible formulation for dynamical systems in Siconos, two for first order systems and two for second order Lagrangian systems. The main class is DynamicalSystem, all other derived from this one, as shown in the following diagram:



{DSDiagram}

3.3 Construction

Each constructor must:

- initialize all the members of the class and of the top-class if it exists

- allocate memory and set value for all required inputs
- allocate memory and set value for optional input if they are given as argument (in xml for example)
- check that given data are coherent and that the system is complete (for example, in the LagrangianDS if the internal forces are given as a plug-in, their Jacobian are also required. If they are not given, this leads to an exception).

No memory allocation is made for unused members ⇒ requires if statements in simulation. (if!=NULL ...).

3.3.1 DynamicalSystem

Required data:
n, x0, f, jacobianXF
Optional:
T,u

Always allocated in constructor:
x, x0, xFree, r, rhs, jacobianXF

Warning: default constructor is always private or protected and apart from the others and previous rules or remarks do not always apply to it. This for DS class and any of the derived ones.

3.3.2 LagrangianDS

Required data:
ndof, q0, velocity0, mass
Optional:
fInt and its Jacobian, fExt, NNL and its Jacobian.

Always allocated in constructor:
mass, q, q0, qFree, velocity, velocity0, velocityFree, p.
All other pointers to vectors/matrices are set to NULL by default.
Memory vectors are required but allocated during call to initMemory function.

Various rules:

- fInt (NNL) given as a plug-in ⇒ check that JacobianQ/Velocity are present (matrices or plug-in)
- any of the four Jacobian present ⇒ allocate memory for block-matrix jacobianX (connectToDS function)
-

check: end of constructor or in initialize?
computeF and JacobianF + corresponding set functions: virtual or not?

3.4 Specific flags or members

- isAllocatedIn: to check inside-class memory allocation
- isPlugin: to check if operators are computed with plug-in or just directly set as a matrix or vector
- workMatrix: used to save some specific matrices in order to avoid recomputation if possible (inverse of mass ...)

3.5 plug-in management

DynamicalSystem class has a member named parameterList which is a $map < string, SimpleVector* >$, ie a list of pointers to SimpleVector*, with a string as a key to identified them. For example, $parametersList["mass"]$ is a SimpleVector*, which corresponds to the last argument given in mass plug-in function.
By default, each parameters vectors must be initialized with a SimpleVector of size 1, as soon as the plug-in is declared. Moreover, to each vector corresponds a flag in isAllocatedIn map, to check if the corresponding vector has been allocated inside the class or not.
For example, in DynamicalSystem, if $isPlugin["vectorField"] == true$, then, during call to constructor or set function, it is necessary to defined the corresponding parameter:
$parametersList["vectorField"] = newSimpleVector(1)$
and to complete the $isAllocatedIn$ flag:
$isAllocatedIn["parameter_for_vectorField"] = true$.

Chapter 4

Interactions

author	F. Pérignon
date	November 7, 2006
version	Kernel 1.3.0

4.1 Introduction

This document is only a sequel of notes and remarks on the way Interactions are implemented in Siconos.
It has to be completed, reviewed, reorganized etc etc for a future Developpers'Guide.
See also documentation in Doc/User/Interaction.

4.2 Class Diagram

4.3 Description

4.3.1 Redaction note F. PERIGNON
review of interactions (for EventDriven implementation) 17th May 2006.

- variable *nInter* renamed in *interactionSize*: represents the size of y and λ. NOT the number of relations !!

- add a variable *nsLawSize* that depends on the non-smooth law type.
  Examples:
  - NewtonImpact -> *nsLawSize* = 1
  - Friction 2D -> *nsLawSize* = 2
  - Friction 3D -> *nsLawSize* = 3
  - ...
  - *nsLawSize* = n with n dim of matrix D in : $y = Cx + D\lambda$, D supposed to be a full-ranked matrix.
    Warning: this case is represented by only one relation of size n.

- *numberOfRelations*: number of relations in the interaction, *numberOfRelations* = $\frac{interactionSize}{nsLawSize}$.

Chapter 5

Notes on the Non Smooth Dynamical System construction

author	F. Pérignon
date	November 7, 2006
version	Kernel 1.3.0

5.1 Introduction

5.2 Class Diagram

5.3 Description

Objects must be constructed in the following order:

- DynamicalSystems
- NonSmoothLaw: depends on nothing
- Relation: no link with an interaction during construction, this will be done during initialization.
- Interaction: default constructor is private and copy is forbidden. Two constructors: xml and from data. Required data are a DSSet, a NonSmoothLaw and a Relation (+ dim of the Interaction and a number).
  Interaction has an initialize function which allocates memory for y and lambda, links correctly the relation and initializes it .... This function is called at the end of the constructor. That may be better to call it in simulation->initialize? Pb: xml constructor needs memory allocation for y and lambda if they are provided in the input xml file.
- NonSmoothDynamicalSystem: default is private, copy fobidden. Two constructors: xml and from data. Required data are the DSSet and the InteractionsSet. The topology is declared and constructed (but empty) during constructor call of the nsds, but initialize in the Simulation, this because it can not be initialize until the nsds has been fully described (ie this to allow user to add DS, Inter ...) at any time in the model, but before simulation initialization).

5.4 misc

- no need to keep a number for Interactions? Only used in xml for OSI, to know which Interactions it holds.

- pb: the number of saved derivatives for y and lambda in Interactions is set to 2. This must depends on the relative degree which is computes during Simulation initialize and thus too late. It is so not available when memory is allocated (Interaction construction). Problem-> to be reviewed.

Chapter 6

OneStepIntegrator and derived classes.

author	F. Pérignon
date	November 7, 2006
version	Kernel 1.3.0

6.1 Introduction

This document is only a sequel of notes and remarks on the way OneStepIntegrators are implemented in Siconos.
It has to be completed, reviewed, reorganized etc etc for a future Developpers'Guide.
See also documentation in Doc/User/OneStepIntegrator for a description of various OSI.

6.2 Class Diagram

6.3 Misc

OSI review for consistency between Lsodar and Moreau:

- add set of DynamicalSystem*
- add set of Interaction*
- add link to strategy that owns the OSI
- remove td object in OSI -> future: replace it by a set of td (one per ds)
- add strat in constructors arg list

osi -> strat -> Model -> nsds -> topology
osi -> strat -> timeDiscretisation

let a timeDiscretisation object in the OSI? set of td (one per ds)?
create a class of object that corresponds to DS on the simulation side ?
will contain the DS, its discretization, theta for Moreau ... ?
Allow setStrategyPtr operation? Warning: need reinitialisation.

Required input by user:

- list of DS or list of Interactions ?
- pointer to strategy
- ...

6.4 Construction

Each constructor must:

-

6.4.1 Moreau

Two maps: one for W, and one for theta. To each DS corresponds a theta and a W.
Strategy arg in each constructor.

Required data:

Optional:

Always allocated in constructor:

Warning: default constructor is always private or protected and apart from the others and previous rules or remarks do not always apply to it.

6.4.2 Lsodar

Required data:

Optional:

Always allocated in constructor:

Chapter 7

First Order Nonlinear Relation

author	0. Bonnefon
date	July, 1 2009
version	Kernel 3.0.0

Chapter 8

Computation of the number of Index Set and various levels

author	V. Acary
date	Septembre 16, 2011
version	Kernel 3.3.0

In this chapter, we give some hints on the automatic computation of the number of index sets, the number of derivatives in the `Interaction` and the levelMin and LevelMax.

8.1 Why is the relative degree not relevant ?

In this section, we give a very brief overview of the notion of relative degree.

8.1.1 First order Linear complementary systems

A Linear Complementarity System (LCS) is defined by

$$\begin{cases} \dot{x} = Ax + B\lambda \\ y = Cx + D\lambda \\ 0 \leq y \perp \lambda \geq 0 \end{cases} \tag{8.1}$$ {eq:LCS-bis}

Definition 1 (Relative degree in the SISO case) *Let us consider a linear system in state representation given by the quadruplet* $(A, B, C, D) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{n\times m} \times \mathbb{R}^{m\times n} \times \mathbb{R}^{m\times m}$*:*

$$\begin{cases} \dot{x} = Ax + B\lambda \\ y = Cx + D\lambda \end{cases} \tag{8.2}$$ {eq:LS}

- *In the Single Input/ Single Output (SISO) case* ($m = 1$)*, the relative degree is defined by the first non zero Markov parameters :*

$$D, CB, CAB, CA^2B, \ldots, CA^{r-1}B, \ldots \tag{8.3}$$ {eq:Markov-Parame

- *In the multiple input/multiple output (MIMO) case* ($m > 1$)*, an uniform relative degree is defined as follows. If* D *is non singular, the relative degree is equal to* 0*. Otherwise, it is assumed to be the first positive integer* r *such that*

$$CA^iB = 0, \quad i = 0 \ldots q-2 \tag{8.4}$$ {eq:mimo-r}

while

$$CA^{r-1}B \text{ is non singular.} \tag{8.5}$$ {eq:mimo-r2}

The Markov parameters arise naturally when we derive with respect to time the output y,

$$\begin{aligned} y &= Cx + D\lambda \\ \dot{y} &= CAx + CB\lambda, \text{ if } D = 0 \\ \ddot{y} &= CA^2x + CAB\lambda, \text{ if } D = 0, CB = 0 \\ &\dots \\ y^{(r)} &= CA^rx + CA^{r-1}B\lambda, \text{ if } D = 0, CB = 0, CA^{r-2}B = 0, r = 1 \dots r-2 \\ &\dots \end{aligned}$$

and the first non zero Markov parameter allows us to define the output y directly in terms of the input λ.

In continuous time, the nature of solutions depends strongly on the relative degree. When we want to perform the time–integration of such systems, we need also to reduce the relative degree or to known it to correctly operate.

We can observe that the relative degree 0 is well defined only by the relation (D nonsingular) and by the nonsmooth law. Indeed, let us imagine that the nonsmooth law is defined by $0 \leq \dot{y} \perp \lambda \geq 0$. We can easily see that the relative degree is reduced.

In the MIMO, the computation of non uniform relative degree is hard task. This is also the case for nonlinear systems.

8.1.2 Second order Lagrangian systems

Let us consider a second order linear and time-invariant Lagrangian dynamical system (see § 2.5)

$$\begin{cases} M\dot{v} + Cv + Kq = F_{Ext}(t) + p \\ \dot{q} = v \end{cases} \tag{8.6}$$ {eq:rd1}

together with a Lagrangian linear relation

$$y = Cq + e + D\lambda + Fz, \tag{8.7}$$ {eq:rd2}

$$p = C^t\lambda \tag{8.8}$$ {eq:rd3}

and a simple nonsmooth law,

$$0 \leq y \perp \lambda \geq 0 \tag{8.9}$$ {eq:rd4}

If $D > 0$, the relative degree is uniformly zero and the system can be solved without deriving the output (8.7). Indeed, we known that the solution of the LCP

$$0 \leq Cq + e + D\lambda + Fz, \perp \lambda \geq 0 \tag{8.10}$$ {eq:rd5}

is unique and Lipschitz with respect to q. It can be denoted as $\lambda(q) = \text{SOL}(D, Cq + e + Fz)$. Therefore, the differential equation (8.6) reduces to a standard ODE with a Lipschitz RHS

$$\begin{cases} M\dot{v} + Cv + Kq = F_{Ext}(t) + C^t\lambda(q) \\ \dot{q} = v \end{cases} \tag{8.11}$$ {eq:rd6}

In the case that we deal with unilateral contact, we usually have $D = 0$ and the relative degree of the system is 2. In this case, the output has to be differentiated as

$$\dot{y} = C\dot{q}, \tag{8.12}$$ {eq:rd7}

and an impact law has to added, for instance the newton's impact law

$$\text{if } y = 0, \text{ when } \dot{y}^+ = -e\dot{y}^- \tag{8.13}$$ {eq:rd8}

In the same vein, the equations of motion (8.6) is not sufficient since the velocity may encounter jumps. The dynamics is usually replaced by a measure differential equation of the form

$$\begin{cases} Mdv + Cv^+(t)dt + Kq(t)dt = F_{Ext}(t)dt + di \\ \dot{q} = v \end{cases} \tag{8.14}$$ {eq:rd10}

where di is the measure that can be related to p thanks to

$$di = pdt + \sigma\delta_{t^*} \tag{8.15}$$ {eq:rd11}

is only one jump is expected at t^*.

8.1.3 Conclusion for the implementation

From the continuous time mathematical analysis, the relative degree is very important to know if we have to compute the derivatives of the output $y^{(n)}$ and to consider various levels for the input $p, \sigma, ...$

However in the numerical practice, the time –discretization makes an assumption on the relative degree and treats the nonsmooth law at different levels. The resulting time discretized system posseses more or less variables.

Consider for instance (8.6) in the case of the Moreau scheme

$$\begin{cases} M(v_{k+1} - v_k) + h(Kq_{k+\theta} + Cv_{k+\theta}) = p_{k+1} = G(q_{k+1})\lambda_{k+1}, & \text{(8.16a)} \\ q_{k+1} = q_k + hv_{k+\theta}, & \text{(8.16b)} \\ \dot{y}_{k+1} = G^\top(q_{k+1})\, v_{k+1} & \text{(8.16c)} \\ \text{if} \quad \bar{y}^\alpha_{k+1} \le 0 \text{ then } 0 \le \dot{y}^\alpha_{k+1} + e\dot{y}^\alpha_k \perp \lambda^\alpha_{k+1} \ge 0, \\ \text{otherwise } \lambda^\alpha_{k+1} = 0. \end{cases} , \alpha \in \mathcal{I} \quad \text{(8.16d)}$$ {eq:MoreauTS} {eq:MoreauTSd}

and the Schatzman–Paoli scheme

$$\begin{cases} M(q_{k+1} - 2q_k + q_{k-1}) + h^2(Kq_{k+\theta} + Cv_{k+\theta}) = p_{k+1}, & \text{(8.17a)} \\ v_{k+1} = \dfrac{q_{k+1} - q_{k-1}}{2h}, & \text{(8.17b)} \\ y_{k+1} = h\left(\dfrac{q_{k+1} + eq_{k-1}}{1+e}\right) & \text{(8.17c)} \\ p_{k+1} = G\left(\dfrac{q_{k+1} + eq_{k-1}}{1+e}\right)\lambda_{k+1} & \text{(8.17d)} \\ 0 \le y_{k+1} \perp \lambda_{k+1} \ge 0. & \text{(8.17e)} \end{cases}$$

We can see easily that the number of derivatives (or levels) that we store for y and λ is independent on the relative degree but is chosen by the `OneStepIntegrator` with respect to the type of systems.

8.2 How to define and compute the various levels and the number of indexSets

8.2.1 y related variables

The size of the vector `y` in the `Interaction` depends on

- the `OneStepIntegrator` type.
  - see the difference between the Moreau and Schatzman Paoli scheme,

  - plan the time–discontinuous Galerkin scheme
  - plan the Higher Order Moreau sweeping process (HOSP)

- the `Simulation` type.
  - In `Timestepping` or `Event-driven` we do not need the same number of stored y

- the `NonSmoothLaw` type.
  - If we consider some cases with or without friction in `Timestepping` or `Event-driven`, we need to adapt the number of stored y

Since the various levels of `y` are used to build the index sets we will need from 0 to a computed size that depends on the previous criteria. Only a part will be used in the `OneStepNSProblem`.

8.2.2 λ related variables

The size of the vector `lambda` in the `Interaction` depends on the same criteria than in the previous section. Only, the number of lambda is not the same as `y` since a multiplier `lambda[i]` is not necessarily related to `y[i]`

8.3 Rules for implementation

We can define new members in `Interaction`:

- `_lowerlevelForOutput`, this value is to 0 by default
- `_upperlevelForOutput`, this value must be computed at initialization with respect to the previous criteria
- `_lowerlevelForInput`, this value must be computed at initialization with respect to the previous criteria
- `_upperlevelForInput`, this value must be computed at initialization with respect to the previous criteria

This level are computed in `Simulation::ComputeLevelsForInputAndOutput`. A visitor is used for the `OneStepIntegrator`. Furthermore, four global levels are computed

- _levelMinForOutput this value is the minimum level for the output `Interaction::_lowerlevelForOutput` for all the interactions
- _levelMaxForOutput this value is the maximum level for the output `Interaction::_upperlevelForOutput` for all the interactions
- _levelMinForInput this value is the minimum level for the output `Interaction::_lowerlevelForInput` for all the interactions
- _levelMaxForInput this value is the maximum level for the output `Interaction::_upperlevelForInput` for all the interactions
- the values `y[i]` must be initialized from `_lowerlevelForOutput` to `_upperlevelForOutput`.
- the values `lamdba[i]` must be initialized from `_lowerlevelForInput` to `_upperlevelForInput`.
- the values `y[i]` in `Interaction` must be used in priority to store the i-th derivative of y. When it is needed, higher index i can be used for other triggering variables. For instance, for an Event–Driven scheme with a Lagrangian systems with friction, sliding velocity must be stored.

- the values of `lamdba[i]` must stored the various multiplier for the nonsmooth law. We affect the same index *i* as for the level of `y[i]` present in the corresponding nonsmooth law.
- The number of `IndexSets` should follows `_levelMaxForY`.

For the dynamical systems :

- The number of levels for `_r` and `_p` in the DS should follow `_lowerlevelForInput` and `_upperlevelForOutput` of the associated interactions. This is done in `Interaction::initialize`.
- A new variable should be added in the LagrangianDS to store the multiplier at the position level (`_tau` ?) to avoid the use of `_p[0]`. Indeed, we will continue to assume that `_p` is the input in the equation of motion. For `lambda` we can use `lambda[0]`

TODO LIST AND QUESTIONS

- What about the case of multiples interactions on a DS with various `_lowerlevelForInput` and `_upperlevelForOutput` ? Normally, all the levels should be correctly initialized thanks to the proposed implementation (r2821)
- `DynamicalSystem::_r` should be a VectorOfVectors
- `DynamicalSystem::_r` is split in `LagrangianDS`. a first part is `LagrangianDS::_p`. The other is not implemented !! `LagrangianDS::_tau` ?

Chapter 9

Newton's linearization for First Order Systems

author	O.Bonnefon, V. Acary
date last update	Sept, 07, 2007 Feb, 2011 April, 2014
version	

This section is devoted to the implementation and the study of the algorithm. The interval of integration is $[0, T]$, $T > 0$, and a grid $t_0 = 0$, $t_{k+1} = t_k + h$, $k \geq 0$, $t_N = T$ is constructed. The approximation of a function $f(\cdot)$ on $[0, T]$ is denoted as $f^N(\cdot)$, and is a piecewise constant function, constant on the intervals $[t_k, t_{k+1})$. We denote $f^N(t_k)$ as f_k. The time-step is $h > 0$.

9.1 Various first order dynamical systems with input/output relations

FirstOrderR. Fully nonlinear case Let us introduce the following system,

$$\begin{aligned} M\dot{x}(t) &= f(x(t), t) + r(t) \\ y(t) &= h(t, x(t), \lambda(t)) \\ r(t) &= g(t, x(t), \lambda(t)) \end{aligned} \tag{9.1}$$ {first-DS}

where $\lambda(t) \in \mathbb{R}^m$ and $y(t) \in \mathbb{R}^m$ are complementary variables related through a multi-valued mapping. According to the class of systems, we are studying, the function f and g are defined by a fully nonlinear framework or by affine functions. We have decided to present the time-discretization in its full generality and specialize the algorithms for each cases in Section **??**. This fully nonlinear case is not implemented in Siconos yet. This fully general case is not yet implemented in Siconos.

This case is implemented in Siconos with the relation `FirstOrderR` using the subtype NonLinearR

FirstOrderType1R Let us introduce a new notation,

$$\begin{aligned} M\dot{x}(t) &= f(x(t), t) + r(t) \\ y(t) &= h(t, x(t)) \\ r(t) &= g(t, \lambda(t)) \end{aligned} \tag{9.2}$$ {first-DS1}

This case is implemented in Siconos with the relation `FirstOrderType1R`.

FirstOrderType2R Let us introduce a new notation,

$$\begin{aligned} M\dot{x}(t) &= f(x(t),t) + r(t) \\ y(t) &= h(t,x(t),\lambda(t)) \\ r(t) &= g(t,\lambda(t)) \end{aligned} \tag{9.3}$$ {first-DS2}

This case is implemented in Siconos with the relation `FirstOrderType2R`.

Linear case Let us introduce a new notation,

$$\begin{aligned} M\dot{x}(t) &= Ax(t) + r(t) + b(t) \\ y(t) &= h(x(t),\lambda(t),z) = Cx + Fz + D\lambda \\ r(t) &= g(t,\lambda(t)) = B\lambda \end{aligned} \tag{9.4}$$ {first-DS3}

9.2 Time–discretizations

9.2.1 Standard $\theta - \gamma$ scheme.

Let us now proceed with the time discretization of (9.4) by a fully implicit scheme :

$$\begin{aligned} &Mx_{k+1} = Mx_k + h\theta f(x_{k+1},t_{k+1}) + h(1-\theta)f(x_k,t_k) + h\gamma r(t_{k+1}) + h(1-\gamma)r(t_k) \\ &y_{k+1} = h(t_{k+1},x_{k+1},\lambda_{k+1}) \\ &r_{k+1} = g(t_{k+1},x_{k+1},\lambda_{k+1}) \\ &\text{NsLaw}(y_{k+1},\lambda_{k+1}) \end{aligned} \tag{9.5}$$ {eq:toto1}

where $\theta = [0,1]$ and $\gamma \in [0,1]$. As in [**?**], we call the problem (9.5) the "one–step nonsmooth problem".

In the Siconos/Kernel module, the use of γ is activated in the class `EulerMoreauOSI` by the boolean `_useGamma`.

This time-discretization is slightly more general than a standard implicit Euler scheme. The main discrepancy lies in the choice of a θ-method to integrate the nonlinear term. For $\theta = 0$, we retrieve the explicit integration of the smooth and single valued term f. Moreover for $\gamma = 0$, the term g is explicitly evaluated. The flexibility in the choice of θ and γ allows the user to improve and control the accuracy, the stability and the numerical damping of the proposed method. For instance, if the smooth dynamics given by f is stiff, or if we have to use big step sizes for practical reasons, the choice of $\theta > 1/2$ offers better stability with the respect to h.

9.2.2 Full $\theta - \gamma$ scheme

Another possible time–discretization is as follows.

$$\begin{aligned} &Mx_{k+1} = Mx_k + h\theta f(x_{k+1},t_{k+1}) + h(1-\theta)f(x_k,t_k) + hr(t_{k+\gamma}) \\ &y_{k+\gamma} = h(t_{k+\gamma},x_{k+\gamma},\lambda_{k+\gamma}) \\ &r_{k+\gamma} = g(t_{k+\gamma},x_{k+\gamma},\lambda_{k+\gamma}) \\ &\text{NsLaw}(y_{k+\gamma},\lambda_{k+\gamma}) \end{aligned} \tag{9.6}$$ {eq:toto1-ter}

We call the scheme (9.6) the full $\theta - \gamma$ scheme since it uses also the evaluation at $t_{k+\gamma}$ for the relation.

In the Siconos/Kernel module, the time–stepping scheme is activated in the class `EulerMoreauOSI` by the boolean `_useGammaForRelation`.

Another possibility for the time discretization in the nonlinear case would be

$$\begin{aligned} Mx_{k+1} &= Mx_k + hf(x_{k+\theta}, t_{k+\theta}) + hr(t_{k+\gamma}) \\ y_{k+\gamma} &= h(t_{k+\gamma}, x_{k+\gamma}, \lambda_{k+\gamma}) \\ r_{k+\gamma} &= g(t_{k+\gamma}, x_{k+\gamma}, \lambda_{k+\gamma}) \\ &\text{NsLaw}(y_{k+\gamma}, \lambda_{k+\gamma}) \end{aligned} \tag{9.7}$$ {eq:toto1-quat}

This scheme has not been yet implemented in Siconos/Kernel.

9.3 Newton's linearization of (9.5)

Due to the fact that two of the studied classes of systems that are studied in this paper are affine functions in terms of f and g, we propose to solve the "one–step nonsmooth problem" (9.5) by performing an external Newton linearization.

Newton's linearization of the first line of (9.5) The first line of the problem (9.5) can be written under the form of a residue $\mathcal{R}$ depending only on x_{k+1} and r_{k+1} such that

$$\mathcal{R}(x_{k+1}, r_{k+1}) = 0 \tag{9.8}$$ {eq:NL3}

with

$$\mathcal{R}(x, r) = M(x - x_k) - h\theta f(x, t_{k+1}) - h(1-\theta) f(x_k, t_k) - h\gamma r - h(1-\gamma) r_k. \tag{9.9}$$

The solution of this system of nonlinear equations is sought as a limit of the sequence $\{x^\alpha_{k+1}, r^\alpha_{k+1}\}_{\alpha \in \mathbb{N}}$ such that

$$\begin{cases} x^0_{k+1} = x_k \\ \\ r^0_{k+1} = r_k \\ \\ \mathcal{R}_L(x^{\alpha+1}_{k+1}, r^{\alpha+1}_{k+1}) = \mathcal{R}(x^\alpha_{k+1}, r^\alpha_{k+1}) + \left[\nabla_x \mathcal{R}(x^\alpha_{k+1}, r^\alpha_{k+1})\right](x^{\alpha+1}_{k+1} - x^\alpha_{k+1}) + \left[\nabla_r \mathcal{R}(x^\alpha_{k+1}, r^\alpha_{k+1})\right](r^{\alpha+1}_{k+1} - r^\alpha_{k+1}) = 0 \end{cases} \tag{9.10}$$ {eq:NL7}

9.3.1 Redaction note V. ACARY
What about r^0_{k+1} ?

The residu free$\mathcal{R}_{\text{free}}$ is also defined (useful for implementation only):

$$\mathcal{R}_{\text{free}}(x) \stackrel{\Delta}{=} M(x - x_k) - h\theta f(x, t_{k+1}) - h(1-\theta) f(x_k, t_k),$$

which yields

$$\mathcal{R}(x, r) = \mathcal{R}_{\text{free}}(x) - h\gamma r - h(1-\gamma) r_k.$$

$$\mathcal{R}(x^\alpha_{k+1}, r^\alpha_{k+1}) = \boxed{\mathcal{R}^\alpha_{k+1} \stackrel{\Delta}{=} \mathcal{R}_{\text{free}}(x^\alpha_{k+1}) - h\gamma r^\alpha_{k+1} - h(1-\gamma) r_k} \tag{9.11}$$ {eq:rfree-1}

$$\mathcal{R}_{\text{free}}(x^\alpha_{k+1}, r^\alpha_{k+1}) = \boxed{\mathcal{R}^\alpha_{\text{free},k+1} \stackrel{\Delta}{=} M(x^\alpha_{k+1} - x_k) - h\theta f(x^\alpha_{k+1}, t_{k+1}) - h(1-\theta) f(x_k, t_k)}$$

At each time–step, we have to solve the following linearized problem,

$$\mathcal{R}^\alpha_{k+1} + (M - h\theta A^\alpha_{k+1})(x^{\alpha+1}_{k+1} - x^\alpha_{k+1}) - h\gamma(r^{\alpha+1}_{k+1} - r^\alpha_{k+1}) = 0, \tag{9.12}$$ {eq:NL10}

with

$$A^\alpha_{k+1} = \nabla_x f(t_{k+1}, x^\alpha_{k+1}) \tag{9.13}$$

By using (9.11), we get

$$\mathcal{R}_{\text{free}}(x^\alpha_{k+1}, r^\alpha_{k+1}) - h\gamma r^{\alpha+1}_{k+1} - h(1-\gamma) r_k + (M - h\theta A^\alpha_{k+1})(x^{\alpha+1}_{k+1} - x^\alpha_{k+1}) = 0 \tag{9.14}$$ {eq:rfree-2}

The matrix W is clearly non singular for small h.

Newton's linearization of the second line of (9.5) The same operation is performed with the second equation of (9.5)

$$\mathcal{R}_y(x, y, \lambda) = y - h(t_{k+1}, x, \lambda) = 0 \tag{9.15}$$

which is linearized as

$$\begin{aligned}\mathcal{R}_{Ly}(x_{k+1}^{\alpha+1}, y_{k+1}^{\alpha+1}, \lambda_{k+1}^{\alpha+1}) = \mathcal{R}_y(x_{k+1}^{\alpha}, y_{k+1}^{\alpha}, \lambda_{k+1}^{\alpha}) + (y_{k+1}^{\alpha+1} - y_{k+1}^{\alpha}) - \\ C_{k+1}^{\alpha}(x_{k+1}^{\alpha+1} - x_{k+1}^{\alpha}) - D_{k+1}^{\alpha}(\lambda_{k+1}^{\alpha+1} - \lambda_{k+1}^{\alpha}) = 0\end{aligned} \tag{9.16}$$ {eq:NL9}

This leads to the following linear equation

$$\boxed{y_{k+1}^{\alpha+1} = y_{k+1}^{\alpha} - \mathcal{R}_{yk+1}^{\alpha} + C_{k+1}^{\alpha}(x_{k+1}^{\alpha+1} - x_{k+1}^{\alpha}) + D_{k+1}^{\alpha}(\lambda_{k+1}^{\alpha+1} - \lambda_{k+1}^{\alpha})}. \tag{9.17}$$ {eq:NL11y}

with,

$$\begin{aligned}C_{k+1}^{\alpha} &= \nabla_x h(t_{k+1}, x_{k+1}^{\alpha}, \lambda_{k+1}^{\alpha}) \\ D_{k+1}^{\alpha} &= \nabla_\lambda h(t_{k+1}, x_{k+1}^{\alpha}, \lambda_{k+1}^{\alpha})\end{aligned} \tag{9.18}$$

and

$$\boxed{\mathcal{R}_{yk+1}^{\alpha} \stackrel{\Delta}{=} y_{k+1}^{\alpha} - h(x_{k+1}^{\alpha}, \lambda_{k+1}^{\alpha})} \tag{9.19}$$

Newton's linearization of the third line of (9.5) The same operation is performed with the third equation of (9.5)

$$\mathcal{R}_r(r, x, \lambda) = r - g(t_{k+1}, x, \lambda) = 0 \tag{9.20}$$

which is linearized as

$$\mathcal{R}_{Lr}(r_{k+1}^{\alpha+1}, x_{k+1}^{\alpha+1}, \lambda_{k+1}^{\alpha+1}) = \mathcal{R}_{rk+1}^{\alpha} + (r_{k+1}^{\alpha+1} - r_{k+1}^{\alpha}) - K_{k+1}^{\alpha}(x_{k+1}^{\alpha+1} - x_{k+1}^{\alpha}) - B_{k+1}^{\alpha}(\lambda_{k+1}^{\alpha+1} - \lambda_{k+1}^{\alpha}) = 0 \tag{9.21}$$ {eq:NL9}

$$\boxed{r_{k+1}^{\alpha+1} = g(t_{k+1}, x_{k+1}^{\alpha}, \lambda_{k+1}^{\alpha}) + K_{k+1}^{\alpha}(x_{k+1}^{\alpha+1} - x_{k+1}^{\alpha}) + B_{k+1}^{\alpha}(\lambda_{k+1}^{\alpha+1} - \lambda_{k+1}^{\alpha})} \tag{9.22}$$ {eq:rrL}

with,

$$\begin{aligned}K_{k+1}^{\alpha} &= \nabla_x g(t_{k+1}, x_{k+1}^{\alpha}, \lambda_{k+1}^{\alpha}) \\ B_{k+1}^{\alpha} &= \nabla_\lambda g(t_{k+1}, x_{k+1}^{\alpha}, \lambda_{k+1}^{\alpha})\end{aligned} \tag{9.23}$$

and the residue for r:

$$\boxed{\mathcal{R}_{rk+1}^{\alpha} = r_{k+1}^{\alpha} - g(t_{k+1}, x_{k+1}^{\alpha}, \lambda_{k+1}^{\alpha})} \tag{9.24}$$

Reduction to a linear relation between $x_{k+1}^{\alpha+1}$ and $\lambda_{k+1}^{\alpha+1}$ Inserting (9.22) into (9.14), we get the following linear relation between $x_{k+1}^{\alpha+1}$ and $\lambda_{k+1}^{\alpha+1}$,

$$\begin{aligned}\mathcal{R}_{\mathsf{free},k+1}^{\alpha} - h\gamma\left[g(t_{k+1}, x_{k+1}^{\alpha}, \lambda_{k+1}^{\alpha}) + B_{k+1}^{\alpha}(\lambda_{k+1}^{\alpha+1} - \lambda_{k+1}^{\alpha}) + K_{k+1}^{\alpha}(x_{k+1}^{\alpha+1} - x_{k+1}^{\alpha})\right] \\ -h(1-\gamma)r_k + (M - h\theta A_{k+1}^{\alpha})(x_{k+1}^{\alpha+1} - x_{k+1}^{\alpha}) = 0\end{aligned} \tag{9.25}$$ {eq:rfree-3}

that is

$$(M - h\theta A^{\alpha}_{k+1} - h\gamma K^{\alpha}_{k+1})(x^{\alpha+1}_{k+1} - x^{\alpha}_{k+1}) = -\mathcal{R}^{\alpha}_{\text{free},k+1} - h(1-\gamma)r_k + h\gamma\left[g(t_{k+1}, x^{\alpha}_{k+1}, \lambda^{\alpha}_{k+1}) + B^{\alpha}_{k+1}(\lambda^{\alpha+1}_{k+1} - \lambda^{\alpha}_{k+1})\right] \quad (9.26)$$ {eq:rfree-4}

Let us introduce some intermediate notation:

$$W^{\alpha}_{k+1} \stackrel{\Delta}{=} M - h\theta A^{\alpha}_{k+1} - h\gamma K^{\alpha}_{k+1}) \quad (9.27)$$ {eq:NL9}

$$\boxed{x^{\alpha}_{\text{free}} \stackrel{\Delta}{=} x^{\alpha}_{k+1} - (W^{\alpha}_{k+1})^{-1}(R^{\alpha}_{\text{free},k+1} - h(1-\gamma)r_k)} \quad (9.28)$$ {eq:rfree-12}

and

$$\boxed{x^{\alpha}_{p} \stackrel{\Delta}{=} h\gamma(W^{\alpha}_{k+1})^{-1}\left[g(t_{k+1}, x^{\alpha}_{k+1}, \lambda^{\alpha}_{k+1}) - B^{\alpha}_{k+1}(\lambda^{\alpha}_{k+1})\right] + x^{\alpha}_{\text{free}}}. \quad (9.29)$$

The relation (9.26) can be written as

$$\boxed{x^{\alpha+1}_{k+1} \stackrel{\Delta}{=} x^{\alpha}_{p} + \left[h\gamma(W^{\alpha}_{k+1})^{-1}B^{\alpha}_{k+1}\lambda^{\alpha+1}_{k+1}\right]} \quad (9.30)$$ {eq:rfree-13}

Reduction to a linear relation between $y^{\alpha+1}_{k+1}$ and $\lambda^{\alpha+1}_{k+1}$. Inserting (9.30) into (9.17), we get the following linear relation between $y^{\alpha+1}_{k+1}$ and $\lambda^{\alpha+1}_{k+1}$,

$$y^{\alpha+1}_{k+1} = y_p + \left[h\gamma C^{\alpha}_{k+1}(W^{\alpha}_{k+1})^{-1}B^{\alpha}_{k+1} + D^{\alpha}_{k+1}\right]\lambda^{\alpha+1}_{k+1} \quad (9.31)$$

with

$$\boxed{y_p = y^{\alpha}_{k+1} - \mathcal{R}^{\alpha}_{yk+1} + C^{\alpha}_{k+1}(x_q) - D^{\alpha}_{k+1}\lambda^{\alpha}_{k+1}} \quad (9.32)$$

$$\boxed{x_q = x^{\alpha}_{p} - x^{\alpha}_{k+1}} \quad (9.33)$$ {eq:xqq}

Mixed linear complementarity problem (MLCP) To summarize, the problem to be solved in each Newton iteration is:

$$\begin{cases} y^{\alpha+1}_{k+1} = W^{\alpha}_{mlcpk+1}\lambda^{\alpha+1}_{k+1} + b^{\alpha}_{k+1} \\ -y^{\alpha+1}_{k+1} \in N_{[l,u]}(\lambda^{\alpha+1}_{k+1}). \end{cases} \quad (9.34)$$ {eq:NL14}

with $W_{mlcpk+1} \in \mathbb{R}^{m\times m}$ and $b \in \mathbb{R}^{m}$ defined by

$$\begin{aligned} W^{\alpha}_{mlcpk+1} &= h\gamma C^{\alpha}_{k+1}(W^{\alpha}_{k+1})^{-1}B^{\alpha}_{k+1} + D^{\alpha}_{k+1} \\ b^{\alpha}_{k+1} &= y_p \end{aligned} \quad (9.35)$$ {eq:NL15}

The problem (10.41) is equivalent to a Mixed Linear Complementarity Problem (MLCP) which can be solved under suitable assumptions by many linear complementarity solvers such as pivoting techniques, interior point techniques and splitting/projection strategies. The reformulation into a standard

MLCP follows the same line as for the MCP in the previous section. One obtains,

$$\begin{aligned} & y_{k+1}^{\alpha+1} = -W_{k+1}^{\alpha}\lambda_{k+1}^{\alpha+1} + b_{k+1}^{\alpha} \\ & (y_{k+1}^{\alpha+1})_i = 0 \qquad \text{for } i \in \{1..n\} \\ & 0 \leq (\lambda_{k+1}^{\alpha+1})_i \perp (y_{k+1}^{\alpha+1})_i \geq 0 \qquad \text{for } i \in \{n..n+m\} \end{aligned} \tag{9.36}$$ {eq:MLCP1}

9.3.1 The special case of Newton's linearization of (9.5) with FirstOrderType2R (9.3)

Let us now proceed with the time discretization of (9.5) with FirstOrderType2R (9.3) by a fully implicit scheme :

$$\begin{aligned} & Mx_{k+1} = Mx_k + h\theta f(x_{k+1}, t_{k+1}) + h(1-\theta)f(x_k, t_k) + h\gamma r(t_{k+1}) + h(1-\gamma)r(t_k) \\ & y_{k+1} = h(t_{k+1}, x_{k+1}, \lambda_{k+1}) \\ & r_{k+1} = g(t_{k+1}, \lambda_{k+1}) \end{aligned} \tag{9.37}$$ {eq:mlcp2-toto1-

Newton's linearization of the first line of (9.37) The linearization of the first line of the problem (9.37) is similar to the previous case so that (9.14) is still valid.

Newton's linearization of the second line of (9.37) The linearization of the second line of the problem (9.37) is similar to the previous case so that (9.17) is still valid.

Newton's linearization of the third line of (9.37) Since $K_{k+1}^{\alpha} = \nabla_x g(t_{k+1}, \lambda_{k+1}^{\alpha}) = 0$, the linearization of the third line of (9.37) reads as

$$\boxed{r_{k+1}^{\alpha+1} = g(t_{k+1}, \lambda_{k+1}^{\alpha}) + B_{k+1}^{\alpha}(\lambda^{\alpha+1} - \lambda_{k+1}^{\alpha})} \tag{9.38}$$ {eq:mlcp2-rrL}

Reduction to a linear relation between $x_{k+1}^{\alpha+1}$ and $\lambda_{k+1}^{\alpha+1}$ Inserting (9.38) into (**??**), we get the following linear relation between $x_{k+1}^{\alpha+1}$ and $\lambda_{k+1}^{\alpha+1}$, we get the linear relation

$$\boxed{x_{k+1}^{\alpha+1} \triangleq x_p^{\alpha} + \left[h\gamma(W_{k+1}^{\alpha})^{-1}B_{k+1}^{\alpha}\lambda_{k+1}^{\alpha+1}\right]} \tag{9.39}$$ {eq:mlcp2-rfree-

with

$$\boxed{x_p^{\alpha} \triangleq h\gamma(W_{k+1}^{\alpha})^{-1}\left[g(t_{k+1}, \lambda_{k+1}^{\alpha}) - B_{k+1}^{\alpha}(\lambda_{k+1}^{\alpha})\right] + x_{\text{free}}^{\alpha}} \tag{9.40}$$

and

$$W_{k+1}^{\alpha} \triangleq M - h\theta A_{k+1}^{\alpha} \tag{9.41}$$ {eq:mlcp2-NL9}

Reduction to a linear relation between $y_{k+1}^{\alpha+1}$ and $\lambda_{k+1}^{\alpha+1}$ Inserting (9.39) into (9.17), we get the following linear relation between $y_{k+1}^{\alpha+1}$ and $\lambda_{k+1}^{\alpha+1}$,

$$y_{k+1}^{\alpha+1} = y_p + \left[h\gamma C_{k+1}^{\alpha}(W_{k+1}^{\alpha})^{-1}B_{k+1}^{\alpha} + D_{k+1}^{\alpha}\right]\lambda_{k+1}^{\alpha+1} \tag{9.42}$$

with

$$y_p = y^\alpha_{k+1} - \mathcal{R}^\alpha_{yk+1} + C^\alpha_{k+1}(x_q) - D^\alpha_{k+1}\lambda^\alpha_{k+1} \tag{9.43}$$

$$x^\alpha_q = x^\alpha_p - x^\alpha_{k+1} \tag{9.44}$$ {eq:mlcp2-xqq}

9.3.2 The special case of Newton's linearization of (9.5) with FirstOrderType1R (9.2)

Let us now proceed with the time discretization of (9.5) with FirstOrderType1R (9.2) by a fully implicit scheme :

$$\begin{aligned} Mx_{k+1} &= Mx_k + h\theta f(x_{k+1}, t_{k+1}) + h(1-\theta)f(x_k, t_k) + h\gamma r(t_{k+1}) + h(1-\gamma)r(t_k) \\ y_{k+1} &= h(t_{k+1}, x_{k+1}) \\ r_{k+1} &= g(t_{k+1}\lambda_{k+1}) \end{aligned} \tag{9.45}$$ {eq:mlcp3-toto1-

The previous derivation is valid with $D^\alpha_{k+1} = 0$.

9.3.3 Time–discretization of the linear case (9.4)

Let us now proceed with the time discretization of (9.5) with FirstOrderLinearR (9.4) by a fully implicit scheme :

$$\begin{aligned} Mx^{\alpha+1}_{k+1} &= Mx_k + h\theta Ax^{\alpha+1}_{k+1} + h(1-\theta)Ax_k + h\gamma r^{\alpha+1}_{k+1} + h(1-\gamma)r(t_k) + hb \\ y^{\alpha+1}_{k+1} &= Cx^{\alpha+1}_{k+1} + D\lambda^{\alpha+1}_{k+1} + Fz + e \\ r^{\alpha+1}_{k+1} &= B\lambda^{\alpha+1}_{k+1} \end{aligned} \tag{9.46}$$ {eq:toto1-DS3}

$$R_{\text{free}} = M(x^\alpha_{k+1} - x_k) - h\theta Ax^\alpha_{k+1} - h(1-\theta)Ax_k - hb_{k+1}$$
$$R_{\text{free}} = W(x^\alpha_{k+1} - x_k) - hAx_k - hb_{k+1}$$

Resulting Newton step (only one step) For the sake of simplicity, let us assume that $\gamma = 1$

$$\begin{aligned} (M - h\theta A)x^{\alpha+1}_{k+1} &= Mx_k + h(1-\theta)Ax_k + hr^{\alpha+1}_{k+1} + hb \\ y^{\alpha+1}_{k+1} &= Cx^{\alpha+1}_{k+1} + D\lambda^{\alpha+1}_{k+1} + Fz + e \\ r^{\alpha+1}_{k+1} &= B\lambda^{\alpha+1}_{k+1} \end{aligned} \tag{9.47}$$

that lead to with: $(M - h\theta A) = W$

$$\begin{aligned} x^{\alpha+1}_{k+1} &= W^{-1}(Mx_k + h(1-\theta)Ax_k + r^{\alpha+1}_{k+1} + hb) = x\text{free} + W^{-1}(r^{\alpha+1}_{k+1}) \\ y^{\alpha+1}_{k+1} &= (D + hCW^{-1}B)\lambda^{\alpha+1}_{k+1} + Fz + CW^{-1}(Mx_k + h(1-\theta)Ax_k + hb) + e \end{aligned} \tag{9.48}$$

with $x_{\text{free}} = x^\alpha_{k+1} + W^{-1}(-R_{\text{free}}) = x^\alpha_{k+1} - W^{-1}(W(x^\alpha_{k+1} - x_k) - hAx_k - hb_{k+1}) = W^{-1}(Mx_k + h(1-\theta)Ax_k + hb_{k+1})$

$$\begin{aligned} y^{\alpha+1}_{k+1} &= (D + hCW^{-1}B)\lambda^{\alpha+1}_{k+1} + Fz + Cx_{\text{free}} + e \\ r^{\alpha+1}_{k+1} &= B\lambda^{\alpha+1}_{k+1} \end{aligned} \tag{9.49}$$

Coherence with previous formulation

$$y_p = y^{\alpha}_{k+1} - \mathcal{R}^{\alpha}_{yk+1} + C^{\alpha}_{k+1}(x_p - x^{\alpha}_{k+1}) - D^{\alpha}_{k+1}\lambda^{\alpha}_{k+1}$$

$$y_p = Cx_k + D\lambda_k + C(\tilde{x}_{\mathsf{free}}) - D\lambda_k + Fz + e$$

$$y_p = Cx_k + C(\tilde{x}_{\mathsf{free}}) + Fz + e$$

$$y_p = Cx_k + C(\tilde{x}_{\mathsf{free}}) + Fz + e$$

$$y_p = C(x_{\mathsf{free}}) + Fz + e$$

9.4 Newton's linearization of (9.6)

In this section, we deal with only with the FirstOrderType2R case.

$$\begin{aligned} Mx_{k+1} &= Mx_k + h\theta f(x_{k+1}, t_{k+1}) + h(1-\theta) f(x_k, t_k) + hr_{k+\gamma} \\ y_{k+\gamma} &= h(t_{k+\gamma}, x_{k+\gamma}, \lambda_{k+\gamma}) \\ r_{k+\gamma} &= g(t_{k+\gamma}, \lambda_{k+\gamma}) \end{aligned} \tag{9.50}$$ {eq:full-toto1-t

Newton's linearization of the first line of (9.50) The first line of the problem (9.50) can be written under the form of a residue $\mathcal{R}$ depending only on x_{k+1} and $r_{k+\gamma}$ such that

$$\mathcal{R}(x_{k+1}, r_{k+\gamma}) = 0 \tag{9.51}$$ {eq:full-NL3}

with

$$\mathcal{R}(x, r) = M(x - x_k) - h\theta f(x, t_{k+1}) - h(1-\theta) f(x_k, t_k) - hr.$$

The solution of this system of nonlinear equations is sought as a limit of the sequence $\{x^\alpha_{k+1}, r^\alpha_{k+\gamma}\}_{\alpha \in \mathbb{N}}$ such that

$$\begin{cases} x^0_{k+1} = x_k \\ \\ r^0_{k+\gamma} = (1-\gamma) r_k + \gamma r^0_{k+1} = r_k \\ \\ \mathcal{R}_L(x^{\alpha+1}_{k+1}, r^{\alpha+1}_{k+\gamma}) = \mathcal{R}(x^\alpha_{k+1}, r^\alpha_{k+\gamma}) + \left[\nabla_x \mathcal{R}(x^\alpha_{k+1}, r^\alpha_{k+\gamma})\right](x^{\alpha+1}_{k+1} - x^\alpha_{k+1}) + \\ \qquad\qquad \left[\nabla_r \mathcal{R}(x^\alpha_{k+1}, r^\alpha_{k+\gamma})\right](r^{\alpha+1}_{k+\gamma} - r^\alpha_{k+\gamma}) = 0 \end{cases} \tag{9.52}$$ {eq:full-NL7}

9.4.1 Redaction note V. ACARY
What about $r^0_{k+\gamma}$?

The residu free is also defined (useful for implementation only):

$$\mathcal{R}_{\text{free}}(x) \stackrel{\Delta}{=} M(x - x_k) - h\theta f(x, t_{k+1}) - h(1-\theta) f(x_k, t_k).$$

We get

$$\mathcal{R}(x^\alpha_{k+1}, r^\alpha_{k+\gamma}) = \boxed{\mathcal{R}^\alpha_{k+1} \stackrel{\Delta}{=} \mathcal{R}_{\text{free}}(x^\alpha_{k+1}) - hr^\alpha_{k+\gamma}} \tag{9.53}$$ {eq:full-rfree-1

$$\mathcal{R}_{\text{free}}(x^\alpha_{k+1}) = \boxed{\mathcal{R}^\alpha_{\text{free},k+1} \stackrel{\Delta}{=} M(x^\alpha_{k+1} - x_k) - h\theta f(x^\alpha_{k+1}, t_{k+1}) - h(1-\theta) f(x_k, t_k)}$$

The computation of the Jacobian of $\mathcal{R}$ with respect to x, denoted by W^α_{k+1} leads to

$$W^\alpha_{k+1} \stackrel{\Delta}{=} \nabla_x \mathcal{R}(x^\alpha_{k+1}) = M - h\theta \nabla_x f(x^\alpha_{k+1}, t_{k+1}). \tag{9.54}$$ {eq:full-NL9}

At each time–step, we have to solve the following linearized problem,

$$\mathcal{R}^\alpha_{k+1} + W^\alpha_{k+1}(x^{\alpha+1}_{k+1} - x^\alpha_{k+1}) - h(r^{\alpha+1}_{k+\gamma} - r^\alpha_{k+\gamma}) = 0, \tag{9.55}$$ {eq:full-NL10}

By using (9.53), we get

$$\mathcal{R}_{\text{free}}(x^\alpha_{k+1}) - hr^{\alpha+1}_{k+\gamma} + W^\alpha_{k+1}(x^{\alpha+1}_{k+1} - x^\alpha_{k+1}) = 0 \tag{9.56}$$ {eq:full-rfree-2

$$\boxed{x_{k+1}^{\alpha+1} = h(W_{k+1}^{\alpha})^{-1} r_{\gamma+1}^{\alpha+1} + x_{\text{free}}^{\alpha}} \tag{9.57}$$

with :

$$\boxed{x_{\text{free}}^{\alpha} \stackrel{\Delta}{=} x_{k+1}^{\alpha} - (W_{k+1}^{\alpha})^{-1} \mathcal{R}_{\text{free},k+1}^{\alpha}} \tag{9.58}$$ {eq:full-rfree-1

The matrix W is clearly non singular for small h.

Note that the linearization is equivalent to the case (9.14) and (9.28) with $\gamma = 1$ and replacing r_{k+1} by $r_{k+\gamma}$.

Newton's linearization of the second line of (9.50) The same operation is performed with the second equation of (9.50)

$$\mathcal{R}_y(x, y, \lambda) = y - h(t_{k+\gamma}, \gamma x + (1-\gamma)x_k, \lambda) = 0 \tag{9.59}$$

which is linearized as

$$\begin{aligned} \mathcal{R}_{Ly}(x_{k+1}^{\alpha+1}, y_{k+\gamma}^{\alpha+1}, \lambda_{k+\gamma}^{\alpha+1}) = \mathcal{R}_y(x_{k+1}^{\alpha}, y_{k+\gamma}^{\alpha}, \lambda_{k+\gamma}^{\alpha}) + (y_{k+\gamma}^{\alpha+1} - y_{k+\gamma}^{\alpha}) - \\ \gamma C_{k+1}^{\alpha}(x_{k+1}^{\alpha+1} - x_{k+1}^{\alpha}) - D_{k+\gamma}^{\alpha}(\lambda_{k+\gamma}^{\alpha+1} - \lambda_{k+\gamma}^{\alpha}) = 0 \end{aligned} \tag{9.60}$$ {eq:full-NL9}

This leads to the following linear equation

$$\boxed{y_{k+\gamma}^{\alpha+1} = y_{k+\gamma}^{\alpha} - \mathcal{R}_{y,k+1}^{\alpha} + \gamma C_{k+1}^{\alpha}(x_{k+1}^{\alpha+1} - x_{k+1}^{\alpha}) + D_{k+\gamma}^{\alpha}(\lambda_{k+\gamma}^{\alpha+1} - \lambda_{k+\gamma}^{\alpha})}. \tag{9.61}$$ {eq:full-NL11y}

with,

$$\begin{aligned} C_{k+\gamma}^{\alpha} &= \nabla_x h(t_{k+1}, x_{k+\gamma}^{\alpha}, \lambda_{k+\gamma}^{\alpha}) \\ D_{k+\gamma}^{\alpha} &= \nabla_\lambda h(t_{k+1}, x_{k+\gamma}^{\alpha}, \lambda_{k+\gamma}^{\alpha}) \end{aligned} \tag{9.62}$$

and

$$\boxed{\mathcal{R}_{yk+1}^{\alpha} \stackrel{\Delta}{=} y_{k+\gamma}^{\alpha} - h(x_{k+\gamma}^{\alpha}, \lambda_{k+\gamma}^{\alpha})} \tag{9.63}$$

Note that the linearization is equivalent to the case (9.17) by replacing λ_{k+1} by $\lambda_{k+\gamma}$ and x_{k+1} by $x_{k+\gamma}$.

Newton's linearization of the third line of (9.50) The same operation is performed with the third equation of (9.50)

$$\mathcal{R}_r(r, \lambda) = r - g(\lambda, t_{k+1}) = 0 \tag{9.64}$$

which is linearized as

$$\mathcal{R}_{L\lambda}(r_{k+\gamma}^{\alpha+1}, \lambda_{k+\gamma}^{\alpha+1}) = \mathcal{R}_{r,k+\gamma}^{\alpha} + (r_{k+\gamma}^{\alpha+1} - r_{k+\gamma}^{\alpha}) - B_{k+\gamma}^{\alpha}(\lambda_{k+\gamma}^{\alpha+1} - \lambda_{k+\gamma}^{\alpha}) = 0 \tag{9.65}$$ {eq:full-NL9}

$$\boxed{r_{k+\gamma}^{\alpha+1} = g(\lambda_{k+\gamma}^{\alpha}, t_{k+\gamma}) - B_{k+\gamma}^{\alpha}\lambda_{k+\gamma}^{\alpha} + B_{k+\gamma}^{\alpha}\lambda_{k+\gamma}^{\alpha+1}} \tag{9.66}$$ {eq:full-rrL}

with,

$$B^{\alpha}_{k+\gamma} = \nabla_{\lambda} g(\lambda^{\alpha}_{k+\gamma}, t_{k+\gamma}) \tag{9.67}$$

and the residue for r:

$$\boxed{\mathcal{R}^{\alpha}_{rk+\gamma} = r^{\alpha}_{k+\gamma} - g(\lambda^{\alpha}_{k+\gamma}, t_{k+\gamma})} \tag{9.68}$$

Note that the linearization is equivalent to the case (9.22) by replacing λ_{k+1} by $\lambda_{k+\gamma}$ and x_{k+1} by $x_{k+\gamma}$.

Reduction to a linear relation between $x^{\alpha+1}_{k+1}$ and $\lambda^{\alpha+1}_{k+\gamma}$ Inserting (9.66) into (9.58), we get the following linear relation between $x^{\alpha+1}_{k+1}$ and $\lambda^{\alpha+1}_{k+1}$,

$$x^{\alpha+1}_{k+1} = h(W^{\alpha}_{k+1})^{-1} \left[g(\lambda^{\alpha}_{k+\gamma}, t_{k+\gamma}) + B^{\alpha}_{k+\gamma}(\lambda^{\alpha+1}_{k+\gamma} - \lambda^{\alpha}_{k+\gamma}) \right] + x^{\alpha}_{free} \tag{9.69}$$

that is

$$\boxed{x^{\alpha+1}_{k+1} = x_p + h(W^{\alpha}_{k+1})^{-1} B^{\alpha}_{k+\gamma} \lambda^{\alpha+1}_{k+\gamma}} \tag{9.70}$$ {eq:full-rfree-1

with

$$\boxed{x_p \triangleq h(W^{\alpha}_{k+1})^{-1} \left[g(\lambda^{\alpha}_{k+\gamma}, t_{k+\gamma}) - B^{\alpha}_{k+\gamma}(\lambda^{\alpha}_{k+\gamma}) \right] + x^{\alpha}_{free}} \tag{9.71}$$

Reduction to a linear relation between $y^{\alpha+1}_{k+\gamma}$ and $\lambda^{\alpha+1}_{k+\gamma}$ Inserting (9.70) into (9.61), we get the following linear relation between $y^{\alpha+1}_{k+1}$ and $\lambda^{\alpha+1}_{k+1}$,

$$y^{\alpha+1}_{k+1} = y_p + \left[h\gamma C^{\alpha}_{k+\gamma} (W^{\alpha}_{k+1})^{-1} B^{\alpha}_{k+1} + D^{\alpha}_{k+1} \right] \lambda^{\alpha+1}_{k+1} \tag{9.72}$$

with

$$y_p = y^{\alpha}_{k+1} - \mathcal{R}^{\alpha}_{yk+1} + \gamma C^{\alpha}_{k+1}(x_q) - D^{\alpha}_{k+1} \lambda^{\alpha}_{k+1} \tag{9.73}$$

that is

$$\boxed{y_p = h(x^{\alpha}_{k+\gamma}, \lambda^{\alpha}_{k+\gamma}) + \gamma C^{\alpha}_{k+1}(x_q) - D^{\alpha}_{k+1} \lambda^{\alpha}_{k+1}} \tag{9.74}$$

$$\boxed{x_q = (x_p - x^{\alpha}_{k+1})} \tag{9.75}$$ {eq:full-xqq}

The linear case

$$\begin{array}{rcl} y_p & = & h(x^{\alpha}_{k+\gamma}, \lambda^{\alpha}_{k+\gamma}) + \gamma C^{\alpha}_{k+1}(x_q) - D^{\alpha}_{k+1} \lambda^{\alpha}_{k+1} \\ & = & C^{\alpha}_{k+1} x^{\alpha}_{k+\gamma} + D^{\alpha}_{k+1} \lambda^{\alpha}_{k+\gamma} + \gamma C^{\alpha}_{k+1}(x_q) - D^{\alpha}_{k+1} \lambda^{\alpha}_{k+1} \\ & = & C^{\alpha}_{k+1}(x^{\alpha}_{k+\gamma} + \gamma x_p - \gamma x^{\alpha}_{k+1}) \\ & = & C^{\alpha}_{k+1}((1-\gamma)x_k + \gamma x_{free}) \text{since } x_p = x_{free} \end{array} \tag{9.76}$$

Implementation details For the moment (Feb. 2011), we set $x_q = (1-\gamma)x_k + \gamma x_{free}$ in the linear case The nonlinear case is not yet implemented since we need to change the management of `H_alpha` in Relation to be able to compute the mid–point values. things that remain to do

- implement the function `BlockVector computeg(t,lambda)` and `SimpleVector computeh(t,x,lambda)` which takes into account the values of the argument and return and vector
- remove temporary computation in Relation of `Xq`,`g_alpha`and `H_alpha`. This should be stored somewhere else. (in the node of the graph)

Chapter 10

Newton's linearization for Lagrangian systems

author	V. Acary
date	Sept, 20, 2011
version	

This section is devoted to the implementation and the study of the algorithm. The interval of integration is $[0, T]$, $T > 0$, and a grid $t_0 = 0$, $t_{k+1} = t_k + h$, $k \geq 0$, $t_N = T$ is constructed. The approximation of a function $f(\cdot)$ on $[0, T]$ is denoted as $f^N(\cdot)$, and is a piecewise constant function, constant on the intervals $[t_k, t_{k+1})$. We denote $f^N(t_k)$ as f_k. The time-step is $h > 0$.

10.1 Various second order dynamical systems with input/output relations

10.1.1 Lagrangian dynamical systems

The class `LagrangianDS` defines and computes a generic ndof-dimensional Lagrangian Non Linear Dynamical System of the form :

$$\begin{cases} M(q,z)\dot{v} + N(v,q,z) + F_{Int}(v,q,t,z) = F_{Ext}(t,z) + p \\ \dot{q} = v \end{cases} \tag{10.1}$$

where

- $q \in R^{ndof}$ is the set of the generalized coordinates,
- $\dot{q} = v \in R^{ndof}$ the velocity, i. e. the time derivative of the generalized coordinates (Lagrangian systems).
- $\ddot{q} = \dot{v} \in R^{ndof}$ the acceleration, i. e. the second time derivative of the generalized coordinates.
- $p \in R^{ndof}$ the reaction forces due to the Non Smooth Interaction.
- $M(q) \in R^{ndof \times ndof}$ is the inertia term saved in the SiconosMatrix mass.
- $N(\dot{q}, q) \in R^{ndof}$ is the non linear inertia term saved in the `SiconosVector _NNL`.
- $F_{Int}(\dot{q}, q, t) \in R^{ndof}$ are the internal forces saved in the SiconosVector fInt.

- $F_{Ext}(t) \in R^{ndof}$ are the external forces saved in the SiconosVector fExt.
- $z \in R^{zSize}$ is a vector of arbitrary algebraic variables, some sort of discrete state.

The equation of motion is also shortly denoted as:

$$M(q,z)\dot{v} = F(v,q,t,z) + p \tag{10.2}$$

where $F(v,q,t,z) \in R^{ndof}$ collects the total forces acting on the system, that is

$$F(v,q,t,z) = F_{Ext}(t,z) - NNL(v,q,z) + F_{Int}(v,q,t,z) \tag{10.3}$$

This vector is stored in the `SiconosVector _Forces`

10.1.2 Fully nonlinear case

Let us introduce the following system,

$$\begin{cases} M(q,z)\dot{v} = F(v,q,t,z) + p \\ \dot{q} = v \\ y = h(t,q,\lambda) \\ p = g(t,q,\lambda) \end{cases} \tag{10.4}$$ {eq:FullyNonLinea

where $\lambda(t) \in I\!R^m$ and $y(t) \in I\!R^m$ are complementary variables related through a multi-valued mapping. According to the class of systems, we are studying, the function F , h and g are defined by a fully nonlinear framework or by affine functions. This fully nonlinear case is not implemented in Siconos yet. This fully general case is not yet implemented in Siconos.

10.1.3 Lagrangian Rheonomous relations

$$\begin{cases} M(q,z)\dot{v} = F(v,q,t,z) + p \\ \dot{q} = v \\ y = h(t,q) \\ p = G(t,q)\lambda) \end{cases} \tag{10.5}$$ {eq:RheonomousNo

10.1.4 Lagrangian Scleronomous relations

$$\begin{cases} M(q,z)\dot{v} = F(v,q,t,z) + p \\ \dot{q} = v \\ y = h(q) \\ p = G(q)\lambda \end{cases} \tag{10.6}$$ {eq:Scleronomous

Fully Linear case

$$\begin{cases} M\dot{v} + Cv + Kq = F_{Ext}(t,z) + p \\ \dot{q} = v \\ y = Cq + e + D\lambda + Fz \\ p = C^T\lambda \end{cases} \tag{10.7}$$ {eq:FullyLinear}

10.2 Moreau–Jean event-capturing scheme

In this section, a time-discretization method of the Lagrange dynamical equation (11.17), consistent with the nonsmooth character of the solution, is presented. It is assumed in this section, as in the other sections, that $v^+(\cdot) = \dot{q}^+(\cdot)$ is a locally bounded variation function. The equation of motion reads as,

$$\begin{cases} M(q(t))dv + N(q(t), v^+(t))dt + F_{\text{int}}(t, q(t), v^+(t))\,dt = F_{\text{ext}}(t)\,dt + dr \\ \\ v^+(t) = \dot{q}^+(t) \\ \\ q(0) = q_0 \in \mathcal{C}(0),\ \dot{q}(0^-) = \dot{q}_0 \end{cases} \tag{10.8}$$ {eq:11-b}

We also assume that $F_{\text{int}}(\cdot)$ and $F_{\text{ext}}(\cdot)$ are continuous with respect to time. This assumption is made for the sake of simplicity to avoid the notation $F^+_{\text{int}}(\cdot)$ and $F^+_{\text{ext}}(\cdot)$. Finally, we will condense the nonlinear inertia terms and the internal forces to lighten the notation. We obtain

$$\begin{cases} M(q(t))dv + F(t, q(t), v^+(t))\,dt = F_{\text{ext}}(t)\,dt + dr \\ \\ v^+(t) = \dot{q}^+(t) \\ \\ q(0) = q_0 \in \mathcal{C}(0),\ \dot{q}(0^-) = \dot{q}_0 \end{cases} \tag{10.9}$$ {eq:11-c}

The NSCD method, also known as the Contact Dynamics (CD) is due to the seminal works of J.J. [**? ? ? ? ?**] and M. [**? ?**] (See also [**? ? ?**]). A lot of improvements and variants have been proposed over the years. In this Section, we take liberties with these original works, but we choose to present a version of the NSCD method which preserves the essential of the original work. Some extra developments and interpretations are added which are only under our responsibility. To come back to the source of the NSCD method, we encourage to read the above references.

10.2.1 The Linear Time-invariant NonSmooth Lagrangian Dynamics

{section11.1.1}

For the sake of simplicity of the presentation, the linear time-invariant case is considered first. The nonlinear case will be examined later in this chapter.

$$\begin{cases} Mdv + (Kq(t) + Cv^+(t))\,dt = F_{\text{ext}}(t)\,dt + dr \\ \\ v^+(t) = \dot{q}^+(t) \end{cases} \tag{10.10}$$ {eq:11-a}

10.2.1.a Time–discretization of the Dynamics

Integrating both sides of this equation over a time step $(t_k, t_{k+1}]$ of length $h > 0$, one obtains

$$\begin{cases} \displaystyle\int_{(t_k,t_{k+1}]} Mdv + \int_{t_k}^{t_{k+1}} (Cv^+(t) + Kq(t))\,dt = \int_{t_k}^{t_{k+1}} F_{\text{ext}}\,dt + \int_{(t_k,t_{k+1}]} dr\,, \\ \\ q(t_{k+1}) = q(t_k) + \displaystyle\int_{t_k}^{t_{k+1}} v^+(t)\,dt \end{cases} \tag{10.11}$$

By definition of the differential measure dv, we obtain

$$\int_{(t_k,t_{k+1}]} M\,dv = M \int_{(t_k,t_{k+1}]} dv = M\,(v^+(t_{k+1}) - v^+(t_k)) \tag{10.12}$$ {eq:19}

Note that the right velocities are involved in this formulation. The impulsion $\int_{(t_k,t_{k+1}]} dr$ of the reaction on the time interval $(t_k, t_{k+1}]$ emerges as a natural unknown. The equation of the nonsmooth motion can be written under an integral form as:

$$\begin{cases} M\left(v(t_{k+1}) - v(t_k)\right) = \int_{t_k}^{t_{k+1}} \left(-Cv^+(t) - Kq(t) + F_{\text{ext}}(t)\right) dt + \int_{(t_k,t_{k+1}]} dr\,, \\ \\ q(t_{k+1}) = q(t_k) + \int_{t_k}^{t_{k+1}} v^+(t)\, dt \end{cases} \tag{10.13}$$

Choosing a numerical method boils down to choose a method of approximation for the remaining integral terms. Since discontinuities of the derivative $v(\cdot)$ are to be expected if some shocks are occurring, i.e.. dr has some atoms within the interval $(t_k, t_{k+1}]$, it is not relevant to use high order approximations integration schemes for dr (this was pointed out in remark **??**). It may be shown on some examples that, on the contrary, such high order schemes may generate artefact numerical oscillations (see [**?**]).

The following notation will be used:

- q_k is an approximation of $q(t_k)$ and q_{k+1} is an approximation of $q(t_{k+1})$,
- v_k is an approximation of $v^+(t_k)$ and v_{k+1} is an approximation of $v^+(t_{k+1})$,
- p_{k+1} is an approximation of $\int_{(t_k,t_{k+1}]} dr$.

A popular first order numerical scheme, the so called θ-method, is used for the term supposed to be sufficiently smooth:

$$\begin{aligned} \int_{t_k}^{t_{k+1}} Cv + Kq\, dt &\approx h\left[\theta(Cv_{k+1} + Kq_{k+1}) + (1-\theta)(Cv_k + Kq_k)\right] \\ \int_{t_k}^{t_{k+1}} F_{\text{ext}}(t)\, dt &\approx h\left[\theta(F_{\text{ext}})_{k+1} + (1-\theta)(F_{\text{ext}})_k\right] \end{aligned}$$

The displacement, assumed to be absolutely continuous, is approximated by:

$$q_{k+1} = q_k + h\ \left[\theta v_{k+1} + (1-\theta)v_k\right]$$

Taking into account all these discretizations, the following time-discretized equation of motion is obtained:

$$\begin{cases} M(v_{k+1} - v_k) + h\left[\theta(Cv_{k+1} + Kq_{k+1}) + (1-\theta)(Cv_k + Kq_k)\right] = \\ \\ \qquad = h\left[\theta(F_{\text{ext}})_{k+1} + (1-\theta)(F_{\text{ext}})_k\right] + p_{k+1} \\ \\ q_{k+1} = q_k + h\left[\theta v_{k+1} + (1-\theta)v_k\right] \end{cases} \tag{10.14}$$ {eq:NSCD-discret

Finally, introducing the expression of q_{k+1} in the first equation of (10.14), one obtains:

$$\left[M + h\theta C + h^2\theta^2 K\right](v_{k+1} - v_k) = -hCv_k - hKq_k - h^2\theta K v_k$$

$$+h\left[\theta(F_{\text{ext}})_{k+1}) + (1-\theta)(F_{\text{ext}})_k\right] + p_{k+1}\,, \tag{10.15}$$ {eq:23}

which can be written as:

$$v_{k+1} = v_{\text{free}} + \widehat{M}^{-1} p_{k+1} \tag{10.16}$$ {eq:24}

where,

- the matrix
$$\widehat{M} = \left[M + h\theta C + h^2\theta^2 K\right] \tag{10.17}$$ {eq:2002}
is usually called the iteration matrix.

- The vector
$$\begin{aligned} v_{\text{free}} \quad &= v_k + \widehat{M}^{-1}\left[-hCv_k - hKq_k - h^2\theta K v_k\right. \\ &\left. + h\left[\theta(F_{\text{ext}})_{k+1}) + (1-\theta)(F_{\text{ext}})_k\right]\right] \end{aligned} \tag{10.18}$$ {eq:2003}
is the so-called "free" velocity, i.e., the velocity of the system when reaction forces are null.

10.2.1.b Comments

Let us make some comments on the above developments:

- The iteration matrix $\widehat{M} = \left[M + h\theta C + h^2\theta^2 K\right]$ is supposed to be invertible, since the mass matrix M is usually positive definite and h is supposed to be small enough. The matrices C and K are usually semi-definite positive since rigid motions are allowed to bodies.
- When $\theta = 0$, the θ-scheme is the explicit Euler scheme. When $\theta = 1$, the θ-scheme is the fully implicit Euler scheme. When dealing with a plain ODE
$$M\ddot{q}(t) + C\dot{q}(t) + Kq(t) = F(t) \tag{10.19}$$
the $\theta-$scheme is unconditionally stable for $0.5 < \theta \leq 1$. It is conditionally stable otherwise.
- The equation (11.35) is a linear form of the dynamical equation. It appears as an affine relation between the two unknowns, v_{k+1} that is an approximation of the right derivative of the Lagrange variable at time t_{k+1}, and the impulse p_{k+1}. Notice that this scheme is fully implicit. Nonsmooth laws have to be treated by implicit methods.
- From a numerical point of view, two major features appear. First, the different terms in the numerical algorithm will keep finite values. When the time step h vanishes, the scheme copes with finite jumps. Secondly, the use of differential measures of the time interval $(t_k, t_{k+1}]$, i.e.., $dv((t_k, t_{k+1}]) = v^+(t_{k+1}) - v^+(t_k)$ and $dr((t_k, t_{k+1}])$, offers a rigorous treatment of the nonsmooth evolutions. It is to be noticed that approximations of the acceleration are ignored.

These remarks on the contact dynamics method might be viewed only as some numerical tricks. In fact, the mathematical study of the second order MDI by Moreau provides a sound mathematical ground to this numerical scheme. It is noteworthy that convergence results have been proved for such time-stepping schemes [**? ? ? ?**], see below.

10.2.2 The Nonlinear NonSmooth Lagrangian Dynamics

{section11.1.2}

10.2.2.a Time–discretization of the Dynamics

Starting from the nonlinear dynamics (10.9), the integration of both sides of this equation over a time step $(t_k, t_{k+1}]$ of length $h > 0$ yields

$$\begin{cases} \displaystyle\int_{(t_k,t_{k+1}]} M(q)dv + \int_{t_k}^{t_{k+1}} F(t, q(t), v^+(t))\,dt = \int_{t_k}^{t_{k+1}} F_{\text{ext}}(t)\,dt + \int_{(t_k,t_{k+1}]} dr\,, \\ \\ \displaystyle q(t_{k+1}) = q(t_k) + \int_{t_k}^{t_{k+1}} v^+(t)\,dt \end{cases} \tag{10.20}$$

The first term is generally approximated by

$$\int_{(t_k,t_{k+1}]} M(q)\,dv \approx M(q_{k+\gamma})\,(v_{k+1}-v_k) \tag{10.21}$$ {eq:19-NL}

where $q_{k+\gamma}$ generalizes the standard notation for $\gamma \in [0,1]$ such that

$$q_{k+\gamma} = (1-\gamma)q_k + \gamma\, q_{k+1} \tag{10.22}$$ {eq:NL1}

The *a priori* smooth terms are evaluated with a θ-method, chosen in this context for its energy conservation ability,

$$\int_{t_k}^{t_{k+1}} F(t,q,v)\,dt \quad \approx \quad h\tilde{F}_{k+\theta} \tag{10.23}$$

where $\tilde{F}_{k+\theta}$ is an approximation with the following dependencies

$$\tilde{F}(t_k, q_k, v_k, t_{k+1}, q_{k+1}, v_{k+1}, t_{k+\theta}, q_{k+\theta}, v_{k+\theta})$$

The mid-values $t_{k+\theta}, q_{k+\theta}, v_{k+\theta}$ are defined by

$$\left\{\begin{array}{l} t_{k+\theta} = \theta t_{k+1} + (1-\theta)t_k \\ q_{k+\theta} = \theta q_{k+1} + (1-\theta)q_k \\ v_{k+\theta} = \theta v_{k+1} + (1-\theta)v_k \end{array}\right. , \quad \theta \in [0,1] \tag{10.24}$$ {eq:NSCD-discret

{eq:Simo}

Remark 1 *The choice of the approximated function $\tilde{F}(\cdot)$ strongly depends on the nature of the internal forces that are modeled. For the linear elastic behavior of homogeneous continuum media, this approximation can be made by:*

$$\tilde{F}_{k+\theta} = \frac{1}{2}K\colon [E(q_k) + E(q_{k+1})] : F(q_{k+1/2}) \tag{10.25}$$

*where $E(:\ cdot)$ is the Green-Lagrange strain tensor, which leads to an energy conserving algorithm as in [**?**]. For nonlinear elastic other smooth nonlinear behaviors, we refer to the work of [**? ?**] and references therein for the choice of the discretization and the value of θ.*

The displacement, assumed to be absolutely continuous is approximated by:

$$q_{k+1} = q_k + h\, v_{k+\theta}$$

The following nonlinear time–discretized equation of motion is obtained:

$$\left\{\begin{array}{l} M(q_{k+\gamma})(v_{k+1}-v_k) + h\tilde{F}_{k+\theta} = p_{k+1} \\ \\ q_{k+1} = q_k + hv_{k+\theta} \end{array}\right. \tag{10.26}$$ {eq:NSCD-discret

In its full generality and at least formally, substituting the expression of $q_{k+\gamma}$, q_{k+1} and $q_{k+\theta}$, the first line of the problem can be written under the form of a residue $\mathcal{R}$ depending only on v_{k+1} such that

$$\mathcal{R}(v_{k+1}) = p_{k+1} \tag{10.27}$$ {eq:NL3}

In the last expression, we have omitted the dependence to the known values at the beginning the time–step, i.e. q_k and v_k.

10.2.2.b Linearizing the Dynamics

The system of equations (10.27) for v_{k+1} and p_{k+1} can be linearized yielding a Newton's procedure for solving it. This linearization needs the knowledge of the Jacobian matrix $\nabla\mathcal{R}(\cdot)$ with respect to its argument to construct the tangent linear model.

Let us consider that the we have to solve the following equations,

$$\mathcal{R}(u) = 0 \tag{10.28}$$ {eq:NL4}

by a Newton's method where

$$\mathcal{R}(u) = M(q_{k+\gamma})(v_{k+1} - v_k) + h\tilde{F}_{k+\theta} \tag{10.29}$$ {eq:NL6}

The solution of this system of nonlinear equations is sought as a limit of the sequence $\{u_{k+1}^{\tau}\}_{\tau\in\mathbb{N}}$ such that

$$\begin{cases} u_{k+1}^{0} = v_k \\ \\ \mathcal{R}_L(u_{k+1}^{\tau+1}) = \mathcal{R}(u_{k+1}^{\tau}) + \nabla\mathcal{R}(u_{k+1}^{\tau})(u_{k+1}^{\tau+1} - u_{k+1}^{\tau}) = 0 \end{cases} \tag{10.30}$$ {eq:NL7}

In practice, all the nonlinearities are not treated in the same manner and the Jacobian matrices for the nonlinear terms involved in the Newton's algorithm are only computed in their natural variables. In the following, we consider some of the most widely used approaches.

The Nonlinear Mass Matrix The derivation of the Jacobian of the first term of $\mathcal{R}(\cdot)$ implies to compute

$$\nabla_u \left(M(q_{k+\gamma}(u))(u - v_k)\right) \text{ with } q_{k+\gamma}(u) = q_k + \gamma h[(1-\theta)v_k + \theta u]. \tag{10.31}$$ {eq:NL2000}

One gets

$$\begin{aligned} \nabla_u \left(M(q_{k+\gamma}(u))(u - v_k)\right) &= M(q_{k+\gamma}(u)) + \left[\nabla_u M(q_{k+\gamma}(u))\right](u - v_k) \\ &= M(q_{k+\gamma}(u)) + \left[h\gamma\theta\nabla_q M(q_{k+\gamma}(u))\right](u - v_k) \end{aligned} \tag{10.32}$$ {eq:NL8}

Remark 2 *The notation $\nabla_u M(q_{k+\gamma}(u))(u - v_k)$ is to be understood as follows:*

$$\nabla_u M(q_{k+\gamma}(u))(u - v_k) = \frac{\partial}{\partial u}[M(q_{k+\gamma}(u))(u - v_k)]$$

which is denoted as $\dfrac{\partial M_{ij}}{\partial q^l}(q_{k+\gamma}(u))(u^l - v_k^l)$ in tensorial notation. {remarkBABAS}

A very common approximation consists in considering that the mass matrix evolves slowly with the configuration in a single time–step, that is, the term $\nabla_q M(q_{k+\gamma})$ is neglected and one gets,

$$\nabla_u (M(q_{k+\gamma}(u))(u - v_k)) \approx M(q_{k+\gamma}(u)) \tag{10.33}$$ {eq:NL9}

The Jacobian matrix $\nabla\mathcal{R}(\cdot)$ is evaluated in u_{k+1}^{τ} which yields for the equation (10.33)

$$\nabla_u (M(q_{k+\gamma})(u_{k+1}^{\tau} - v_k)) \approx M(q_k + \gamma h[(1-\theta)v_k + \theta u_{k+1}^{\tau}])) \tag{10.34}$$ {eq:NL10}

The prediction of the position which plays an important role will be denoted by

$$\tilde{q}_{k+1}^{\tau} = q_k + \gamma h[(1-\theta)v_k + \theta u_{k+1}^{\tau}] \tag{10.35}$$ {eq:NL555}

Very often, the matrix $M(q_{k+\gamma})$ is only evaluated at the first Newton's iteration with $u_{k+1}^{0} = v_k$ leading the approximation for the whole step:

$$M(q_k + \gamma h[(1-\theta)v_k + \theta u_{k+1}^{\tau}])) \approx M(q_k + h\gamma v_k) \tag{10.36}$$ {eq:NL11}

Another way to interpret the approximation (10.36) is to remark that this evaluation is just an explicit evaluation of the predictive position (10.35) given by $\theta = 0$:

$$\tilde{q}_{k+1} = q_k + h\gamma v_k \tag{10.37}$$ {eq:NL5}

Using this prediction, the problem (10.26) is written as follows:

$$\begin{cases} M(\tilde{q}_{k+1})(v_{k+1} - v_k) + h\tilde{F}_{k+\theta} = p_{k+1} \\ \\ q_{k+1} = q_k + hv_{k+\theta} \\ \\ \tilde{q}_{k+1} = q_k + h\gamma v_k \end{cases} \tag{10.38}$$ {eq:NSCD-discret1

The Nonlinear Term $F(t,q,v)$ The remaining nonlinear term is linearized providing the Jacobian matrices of $F(t,q,v)$ with respect to q and v. This expression depends strongly on the choice of the approximation $\tilde{F}_{k+\theta}$. Let us consider a pedagogical example, which is not necessarily the best as the Remark 1 suggests but which is one of the simplest,

$$\tilde{F}_{k+\theta} = (1-\theta)F(t_k,q_k,v_k) + \theta F(t_{k+1},q_{k+1},v_{k+1}) \tag{10.39}$$ {eq:NL13}

The computation of the Jacobian of $\tilde{F}_{k+\theta}(t,q(u),u)$ for

$$q(u) = q_k + h[(1-\theta)v_k + \theta u]$$

is given for this example by

$$\begin{aligned}\nabla_u \tilde{F}_{k+\theta}(t,q,u) &= \theta \nabla_u F(t,q(u),u)\\ &= \theta \nabla_q F(t_{k+1},q(u),u)\nabla_u q(u) + \theta \nabla_u F(t,q(u),u)\\ &= h\theta^2 \nabla_q F(t,q(u),u) + \theta \nabla_u F(t,q(u),u)\end{aligned} \tag{10.40}$$ {eq:NL12}

The standard tangent stiffness and damping matrices K_t and C_t are defined by

$$\begin{aligned}K_t(t,q,u) &= \nabla_q F(t,q,u)\\ C_t(t,q,u) &= \nabla_u F(t,q,u)\end{aligned} \tag{10.41}$$ {eq:NL14}

In this case, the Jacobian of $\tilde{F}_{k+\theta}(t,q(u),u)$ may be written as

$$\nabla_u \tilde{F}_{k+\theta}(t,q,u) = h\theta^2 K_t(t,q,u) + \theta C_t(t,q,u) \tag{10.42}$$ {eq:NL15}

The complete Newton's iteration can then be written as

$$\widehat{M}_{k+1}^{\tau+1}(u_{k+1}^{\tau+1} - u_{k+1}^{\tau}) = \mathcal{R}(u_{k+1}^{\tau}) + p_{k+1}^{\tau+1} \tag{10.43}$$ {eq:NL16}

where the iteration matrix is evaluated as

$$\widehat{M}_{k+1}^{\tau+1} = (M(\tilde{q}_{k+1}^{\tau}) + h^2\theta^2 K_t(t_{k+1},q_{k+1}^{\tau},u_{k+1}^{\tau}) + \theta h C_t(t,q_{k+1}^{\tau},u_{k+1}^{\tau})) \tag{10.44}$$ {eq:NL17}

(compare with (10.17)).

Remark 3 *The choice of $\theta = 0$ leads to an explicit evaluation of the position and the nonlinear forces terms. This choice can be interesting if the time–step has to be chosen relatively small due to the presence a very rapid dynamical process. This can be the case in crashes applications or in fracture dynamics [**?**]. In this case, the iteration matrix reduces to $\widehat{M}_{k+1}^{\tau+1} = M(\tilde{q}_{k+1}^{\tau})$ avoiding the expensive evaluation of the tangent operator at each time–step.*

This choice must not be misunderstood. The treatment of the nonsmooth dynamics continues to be implicit.

10.3 Schatzman–Paoli 'scheme and its linearizations

10.3.1 The scheme

$$\begin{cases} M(q_k)(q_{k+1} - 2q_k + q_{k-1}) - h^2 F(v_{k+\theta}, q_{k+\theta}, t_{k+theta}) = p_{k+1}, & \text{(10.45a)}\\ v_{k+1} = \dfrac{q_{k+1} - q_{k-1}}{2h}, & \text{(10.45b)}\\ y_{k+1} = h\left(\dfrac{q_{k+1} + eq_{k-1}}{1+e}\right) & \text{(10.45c)}\\ p_{k+1} = G\left(\dfrac{q_{k+1} + eq_{k-1}}{1+e}\right)\lambda_{k+1} & \text{(10.45d)}\\ 0 \le y_{k+1} \perp \lambda_{k+1} \ge 0. & \text{(10.45e)}\end{cases}$$

10.3.1 Redaction note V. ACARY

Should we have

$$v_{k+1} = \frac{q_{k+1} - q_{k-1}}{2h}$$

or

$$v_{k+1} = \frac{q_{k+1} - q_k}{h}$$

? This question is particularly important for the initialization and the proposed θ-scheme

10.3.2 The Newton linearization

Let us define the residu on q

$$\mathcal{R}(q) = M(q)(q - 2q_k + q_{k-1}) + h^2 F((\theta v(q) + (1-\theta)v_k), \theta q + (1-\theta)q_k), t_{k+\theta}) - p_{k+1} \tag{10.46}$$ {eq:residu}

with

$$v(q) = \frac{q - q_{k-1}}{2h} \tag{10.47}$$ {eq:residu-linq1}

that is

$$\mathcal{R}(q) = M(q)(q - 2q_k + q_{k-1}) + h^2 F((\theta \frac{q - q_{k-1}}{2h} + (1-\theta)v_k), \theta q + (1-\theta)q_k), t_{k+\theta}) - p_{k+1} \tag{10.48}$$ {eq:residu-linq2}

Neglecting $\nabla_q M(q)$ we get

$$\nabla_q \mathcal{R}(q^\nu) = M(q^\nu) + h^2 \theta K(q^\nu, v^\nu) + \frac{1}{2} h \theta C(q^\nu, v^\nu) \tag{10.49}$$ {eq:iterationmat

and we have to solve

$$\nabla_q \mathcal{R}(q^\nu)(q^{\nu+1} - q^\nu) = -\mathcal{R}(q^\nu). \tag{10.50}$$ {eq:iterationloo

10.3.3 Linear version of the scheme

$$\begin{cases} M(q_{k+1} - 2q_k + q_{k-1}) + h^2(Kq_{k+\theta} + Cv_{k+\theta}) = p_{k+1}, & (10.51a) \\ v_{k+1} = \frac{q_{k+1} - q_{k-1}}{2h}, & (10.51b) \\ y_{k+1} = h\left(\frac{q_{k+1} + eq_{k-1}}{1+e}\right) & (10.51c) \\ p_{k+1} = G\left(\frac{q_{k+1} + eq_{k-1}}{1+e}\right)\lambda_{k+1} & (10.51d) \\ 0 \le y_{k+1} \perp \lambda_{k+1} \ge 0. & (10.51e) \end{cases}$$

Let us define the residu on q

$$\mathcal{R}(q) = M(q - 2q_k + q_{k-1}) + h^2(K(\theta q + (1-\theta)q_k)) + C(\theta v(q) + (1-\theta)v_k)) - p_{k+1} \tag{10.52}$$ {eq:residu-linq}

with

$$v(q) = \frac{q - q_{k-1}}{2h} \tag{10.53}$$ {eq:residu-linq1}

that is

$$\mathcal{R}(q) = M(q - 2q_k + q_{k-1}) + h^2(K(\theta q + (1-\theta)q_k))) + h^2 C(\theta \frac{q - q_{k-1}}{2h} + (1-\theta)v_k)) - p_{k+1} \tag{10.54}$$ {eq:residu-linq2}

In this linear case, assuming that $q^0 = q^\nu = q_k$, we get

$$\mathcal{R}(q^\nu) = M(-q_k + q_{k-1}) + h^2(Kq_k) + h^2 C(\theta \frac{q_k - q_{k-1}}{2h} + (1-\theta)v_k)) - p_{k+1} \tag{10.55}$$ {eq:residu-linq2}

10.4 What about mixing `OnestepIntegrator` in Simulation?

{Sec:MisingOSI}

Let us consider that we have two simple linear Lagrangian Dynamical systems

$$\begin{cases} M_1\dot{v}_1 = F_{1,Ext}(t) + p_1 \\ \dot{q}_1 = v_1 \end{cases} \tag{10.56}$$ {eq:FullyLinear1}

and

$$\begin{cases} M_2\dot{v}_2 = F_{2,Ext}(t) + p_2 \\ \dot{q}_2 = v_2 \end{cases} \tag{10.57}$$ {eq:FullyLinear1}

These Dynamical systems (10.57) and (10.57) might numerically solved by choosing two different time–stepping schemes. Let us choose for instance Moreau's scheme for(10.57)

$$\begin{cases} M_1(v_{1,k+1} - v_{1,k}) = F_{1,Ext}(t_{k+1}) + p_{1,k+1} \\ q_{1,k+1} = q_k + hv_{1,k+\theta} \end{cases} \tag{10.58}$$ {eq:FullyLinear1

and Schatzman–Paoli's sheme for (10.57)

$$\begin{cases} M_2(q_{2,k+1} - 2q_{2,k} + q_{2,k-1}) = F_{2,Ext}(t_{k+1}) + p_{2,k+1} \\ v_{2,k+1} = \dfrac{q_{2,k+1} - q_{2,k-1}}{2h} \end{cases} \tag{10.59}$$ {eq:FullyLinear1

Let us consider known that we have a `LagrangianLinearTIR` between this two DSs such that

$$\begin{aligned} y &= q_1 - q_2 \geq 0 \\ p &= \begin{bmatrix} 1 \\ -1 \end{bmatrix} \lambda \end{aligned} \tag{10.60}$$ {eq:LTIR-2DS}

and a complementarity condition

$$0 \leq y \perp \lambda \geq 0 \tag{10.61}$$ {eq:CP}

Many questions are raised when we want to deal with the discrete systems:

- Which rules should we use for the discretization of (10.61) ?

$$\text{if } \bar{y}_{k+1} \leq 0, \text{ then } 0 \leq \dot{y}_{k+1} + e\dot{y}_k \perp \hat{\lambda}_{k+1} \geq 0 \tag{10.62}$$ {eq:CP-TS1}

  or

$$0 \leq y_{k+1} + ey_{k-1} \perp \tilde{\lambda}_{k+1} \geq 0 \tag{10.63}$$ {eq:CP-TS2}

- Should we assume that $y_{k+1} = q_{1,k+1} - q_{2,k+1}$ and $\dot{y}_{k+1} = v_{1,k+1} - v_{2,k+1}$
- How can we link $\hat{\lambda}_{k+1}$ and $\tilde{\lambda}_{k+1}$ with $p_{1,k+1}$ and $p_{2,k+1}$?

The third is the more difficult question and is seems that it is not reasonable to deal with two DS related by one interaction with different osi.In practice, this should be avoided in Siconos.

Chapter 11

NewtonEuler Dynamical Systems

Author	O. Bonnefon	2010
Revision	section **??** to **??** V. Acary	05/09/2011
Revision	section **??** V. Acary	01/06/2016
Revision	complete edition V. Acary	06/01/2017

11.1 The equations of motion

In the maximal coordinates framework, the most natural choice for the kinematic variables and for the formulation of the equations of motion is the Newton/Euler formalism, where the equation of motion describes the translational and rotational dynamics of each body using a specific choice of parameters. For the translational motion, the position of the center of mass $x_{\mathrm{g}} \in I\!R^3$ and its velocity $v_{\mathrm{g}} = \dot{x}_{\mathrm{g}} \in I\!R^3$ is usually chosen. For the orientation of the body is usually defined by the rotation matrix R of the body-fixed frame with respect to a given inertial frame.

For the rotational motion, a common choice is to choose the rotational velocity $\Omega \in I\!R^3$ of the body expressed in the body–fixed frame. This choice comes from the formulation of a rigid body motion of a point X in the inertial frame as

$$x(t) = \Phi(t, X) = x_{\mathrm{g}}(t) + R(t)X. \tag{11.1}$$ {eq:1}

The velocity of this point can be written as

$$\dot{x}(t) = v_{\mathrm{g}}(t) + \dot{R}(t)X \tag{11.2}$$ {eq:2}

Since $R^\top R = I$, we get $R^\top \dot{R} + \dot{R}^\top R = 0$. We can conclude that it exists a matrix $\tilde{\Omega} := R^\top \dot{R}$ such that $\tilde{\Omega} + \tilde{\Omega}^\top = 0$, i.e. a skew symmetric matrix. The notation $\tilde{\Omega}$ comes from the fact that there is a bijection between the skew symmetric matrix in $I\!R^{3\times 3}$ and $I\!R^3$ such that

$$\tilde{\Omega} x = \Omega \times x, \quad \forall x \in I\!R^3. \tag{11.3}$$ {eq:3}

The rotational velocity is then related to the R by :

$$\widetilde{\Omega} = R^\top \dot{R}, \text{ or equivalently, } \dot{R} = R\widetilde{\Omega} \tag{11.4}$$ {eq:angularveloc

Using these coordinates, the equations of motion are given by

$$\left\{ \begin{array}{rcl} m\,\dot{v}_{\mathrm{g}} & = & f(t, x_{\mathrm{g}}, v_{\mathrm{g}}, R, \Omega) \\ I\dot{\Omega} + \Omega \times I\Omega & = & M(t, x_{\mathrm{g}}, v_{\mathrm{g}}, R, \Omega) \\ \dot{x}_{\mathrm{g}} & = & v_{\mathrm{g}} \\ \dot{R} & = & R\widetilde{\Omega} \end{array} \right. \tag{11.5}$$ {eq:motion-Newto

where $m > 0$ is the mass, $I \in I\!R^{3\times 3}$ is the matrix of moments of inertia around the center of mass and the axis of the body–fixed frame.

The vectors $f(\cdot) \in I\!R^3$ and $M(\cdot) \in I\!R^3$ are the total forces and torques applied to the body. It is important to outline that the total applied forces $f(\cdot)$ has to be expressed in a consistent frame w.r.t. to v_g. In our case, it hae to be expressed in the inertial frame. The same applies for the moment M that has to be expressed in the body-fixed frame. If we consider a moment $m(\cdot)$ expressed in the inertial frame, then is has to be convected to the body–fixed frame thanks to

$$M(\cdot) = R^\top m(\cdot) \tag{11.6}$$ {eq:convected_mo

Remark 4 *If we perform the time derivation of $RR^\top = I$ rather than $R^\top R = I$, we get $R\dot{R}^\top + \dot{R}R^\top = 0$. We can conclude that it exists a matrix $\tilde{\omega} := \dot{R}R^\top$ such that $\tilde{\omega} + \tilde{\omega}^\top = 0$, i.e. a skew symmetric matrix. Clearly, we have*

$$\tilde{\omega} = R\tilde{\Omega}R^\top \tag{11.7}$$ {eq:4}

and it can be proved that is equivalent to $\omega = R\Omega$. The vector ω is the rotational velocity expressed in the inertial frame. The equation of motion can also be expressed in the inertial frame as follows

$$\begin{cases} m\,\dot{v}_g &= f(t, x_g, v_g, R, R^T\omega) \\ J(R)\dot{\omega} + \omega \times J(R)\omega &= m(t, x_g, v_g, R, \omega) \\ \dot{x}_g &= v_g \\ \dot{R} &= \widetilde{\omega}R \end{cases} \tag{11.8}$$ {eq:motion-Newto

where the matrix $J(R) = RIR^T$ is the inertia matrix in the inertial frame. Defining the angular momentum with respect to the inertial frame as

$$\pi(t) = J(R(t))\omega(t) \tag{11.9}$$ {eq:1}

the equation of the angular motion is derived from the balance equation of the angular momentum

$$\dot{\pi}(t) = m(t, x_g, v_g, R, \omega)). \tag{11.10}$$ {eq:5}

For a given constant (time invariant) $\tilde{\Omega}$, let us consider the differential equation

$$\begin{cases} \dot{R}(t) = R\tilde{\Omega} \\ R(0) = I \end{cases} \tag{11.11}$$ {eq:5}

Let us recall the definition of the matrix exponential,

$$\exp(A) = \sum_{k=0}^{\infty} \frac{1}{k!} A^k \tag{11.12}$$ {eq:6}

A trivial solution of (11.11) is $R(t) = \exp(t\tilde{\Omega})$ since

$$\frac{d}{dt}(\exp(At)) = \exp(At)A. \tag{11.13}$$ {eq:7}

More generally, with the initial condition $R(t_0) = R_0$, we get the solution

$$R(t) = R_0 \exp((t - t_0)\tilde{\Omega}) \tag{11.14}$$ {eq:8}

Another interpretation is as follows. From a (incremental) rotation vector, Ω and its associated matrix $\tilde{\Omega}$, we obtain a rotation matrix by the exponentation of $\tilde{\Omega}$:

$$R = \exp(\tilde{\Omega}). \tag{11.15}$$ {eq:9}

Since we note that $\tilde{\Omega}^3 = -\theta^2\tilde{\Omega}$ with $\theta = \|\Omega\|$, it is possible to get a closed form of the matrix exponential of $\tilde{\Omega}$

$$\begin{aligned} \exp(\tilde{\Omega}) &= \sum_{k=0}^{\infty} \frac{1}{k!}(\tilde{\Omega})^k \\ &= I_{3\times 3} + \sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{(2k-1)!}\theta^{2k-1}\tilde{\Omega} + (\sum_{k=0}^{\infty} \frac{(-1)^{k-1}}{(2k)!}\theta)^{2k-2}\tilde{\Omega}^2 \\ &= I_{3\times 3} + \frac{\sin\theta}{\theta}\tilde{\Omega} + \frac{(\cos\theta - 1)}{\theta^2}\tilde{\Omega}^2 \end{aligned} \tag{11.16}$$ {eq:10}

that is

$$R = I_{3\times 3} + \frac{\sin\theta}{\theta}\tilde{\Omega} + \frac{(\cos\theta - 1)}{\theta^2}\tilde{\Omega}^2 \tag{11.17}$$ {eq:11}

The formula (11.17) is the Euler–Rodrigues formula that allows to compute the rotation matrix on closed form.

11.1.1 Redaction note V. ACARY

todo :

- add the formulation in the inertial frame of the Euler equation with $\omega = R\Omega$.
- Check that (11.16) is the Euler-Rodrigues formula and not the Olinde Rodrigues formula. (division by θ)

In the numerical practice, the choice of the rotation matrix is not convenient since it introduces redundant parameters. Since R must belong to $SO^+(3)$, we have also to satisfy $\det(R) = 1$ and $R^{-1} = R^\top$. In general, we use a reduced vector of parameters $p \in \mathbb{R}^{n_p}$ such $R = \Phi(p)$ and $\dot{p} = \psi(p)\Omega$. We denote by q the vector of coordinates of the position and the orientation of the body, and by v the body twist:

$$q := \begin{bmatrix} x_{\mathrm{g}} \\ p \end{bmatrix}, \quad v := \begin{bmatrix} v_{\mathrm{g}} \\ \Omega \end{bmatrix}. \tag{11.18}$$

The relation between v and the time derivative of q is

$$\dot{q} = \begin{bmatrix} \dot{x}_{\mathrm{g}} \\ \psi(p)\dot{p} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & \psi(p) \end{bmatrix} v := T(q)v \tag{11.19}$$ {eq:TT}

with $T(q) \in \mathbb{R}^{3+n_p\times 6}$. Note that the twist v is not directly the time derivative of the coordinate vector as a major difference with Lagrangian systems.

The Newton-Euler equation in compact form may be written as:

$$\boxed{\begin{cases} \dot{q} = T(q)v, \\ M\dot{v} = F(t,q,v) \end{cases}} \tag{11.20}$$ {eq:Newton-Euler

where $M \in \mathbb{R}^{6\times 6}$ is the total inertia matrix

$$M := \begin{pmatrix} mI_{3\times 3} & 0 \\ 0 & I \end{pmatrix}, \tag{11.21}$$

and $F(t,q,v) \in \mathbb{R}^6$ collects all the forces and torques applied to the body

$$F(t,q,v) := \begin{pmatrix} f(t, x_{\mathrm{g}}, v_{\mathrm{g}}, R, \Omega) \\ I\Omega \times \Omega + M(t, x_{\mathrm{g}}, v_{\mathrm{g}}, R, \Omega) \end{pmatrix}. \tag{11.22}$$

When a collection of bodies is considered, we will use the same notation as in (11.20) extending the definition of the variables q, v and the operators M, F in a straightforward way.

11.2 Basic elements of Lie groups and Lie algebras theory.

Let us recall the definitions of the Lie group Theory taken from [2] and [4].

11.2.1 Differential equation (evolving) on a manifold $\mathcal{M}$

Definition 2 *A d-dimensional manifold $\mathcal{M}$ is a d-dimensional smooth surface $M \subset \mathbb{R}^n$ for some $n \geq d$.*

Definition 3 *Let $\mathcal{M}$ be a d-dimensional manifold and suppose that $\rho(t) \in \mathcal{M}$ is a smooth curve such that $\rho(0) = p$. A tangent vector at p is defined as*

$$a = \frac{d}{dt}(\rho(t))\Big|_{t=0}. \tag{11.23}$$ {eq:12}

The set of all tangents at p is called the tangent space at p and denoted by $T\mathcal{M}|_p$. It has the structure of a linear space.

Definition 4 *A (tangent) vector field on $\mathcal{M}$ is a smooth function $F : \mathcal{M} \to T\mathcal{M}$ such that $F(p) \in T\mathcal{M}|_p$ for all $p \in \mathcal{M}$. The collection of all vector fields on $\mathcal{M}$ is denoted by $\mathcal{X}(\mathcal{M})$.*

Definition 5 (Differential equation (evolving) on $\mathcal{M}$) *Let F be a tangent vector field on $\mathcal{M}$. By a differential equation (evolving) on $\mathcal{M}$ we mean a differential equation of the form*

$$\dot{y} = F(y), t \geq 0, y(0) \in \mathcal{M} \tag{11.24}$$ {eq:13}

where $F \in \mathcal{X}(\mathcal{M})$. Whenever convenient, we allow F in (11.24) *to be a function of time, $F = F(t, y)$. The flow of F is the solution operator $\Psi_{t,F} : \mathcal{M} \to \mathcal{M}$ such that*

$$y(t) = \Psi_{t,F}(y0). \tag{11.25}$$ {eq:14}

11.2.2 Lie algebra and Lie group

Definition 6 (commutator) *Given two vector fields F, G on $\mathbb{R}^n$, the commutator $H = [F, G]$ can be computed componentwise at a given point $y\mathbb{R}^n$ as*

$$H_i(y) = \sum_{j=1}^{n} G_j(y)\frac{\partial F_i(y)}{\partial y_j} F_j(y)\frac{\partial G_i(y)}{\partial y_j}. \tag{11.26}$$ {eq:15}

{lemma:LieBracke

Lemma 1 *The commutator of vector fields satisfies the identities*

$$\begin{array}{lcll} [F, G] & = & [G, F] & (skewsymmetry), \\ [\alpha F, G] & = & \alpha[F, G], \alpha \in \mathbb{R} & \\ [F + G, H] & = & [F, H] + [G, H] & (bilinearity), \\ 0 & = & [F, [G, H]] + [G, [H, F]] + [H, [F, G]] & (Jacobisidentity). \end{array} \tag{11.27}$$ {eq:16}

Definition 7 *A Lie algebra of vector fields is a collection of vector fields which is closed under linear combination and commutation. In other words, letting $\mathfrak{g}$ denote the Lie algebra,*

$$\begin{array}{l} B \in \mathfrak{g} \implies \alpha B \in \mathfrak{g} \text{ for all } \alpha R. \\ B_1, B_2 \in \mathfrak{g} \implies B_1 + B_2, [B_1, B_2] \in \mathfrak{g} \end{array} \tag{11.28}$$ {eq:17}

Given a collection of vector fields $B = B_1, B_2, \ldots$, the least Lie algebra of vector fields containing B is called the Lie algebra generated by B

Definition 8 *A Lie algebra is a linear space V equipped with a Lie bracket, a bilinear, skew-symmetric mapping*

$$[\cdot, \cdot] : V \times V \to V \tag{11.29}$$ {eq:18}

that obeys identities (11.27) *from Lemma 1*

Definition 9 ((General) Lie algebra) *A Lie algebra homomorphism is a linear map between two Lie algebras, $\varphi : \mathfrak{g} \to \mathfrak{h}$, satisfying the identity*

$$\varphi([v, w]_{\mathfrak{g}}) = [\varphi(v), \varphi(w)]_{\mathfrak{h}}, v, win\mathfrak{g}. \tag{11.30}$$ {eq:19}

An invertible homomorphism is called an isomorphism.

Definition 10 *A Lie group is a differential manifold $\mathcal{G}$ equipped with a product $\cdot : \mathcal{G} \times \mathcal{G} \to \mathcal{G}$ satisfying*

$$\begin{array}{ll} p \cdot (q \cdot r) = (p \cdot q) \cdot r, \forall p, q, r \mathcal{G} & \textit{(associativity)} \\ \exists I \in \mathcal{G} \text{ such that } I \cdot p = p \cdot I = p, \forall p \in \mathcal{G} & \textit{(identity element)} \\ \forall p \in \mathcal{G}, \exists p^{-1} \in \mathcal{G} \text{ such that } p^{-1} \cdot p = I & \textit{(inverse)} \\ \textit{The maps} (p, r) \to p \cdot r \textit{ and } p \to p^{-1} \textit{are smooth functions} & \textit{(smoothness)} \end{array} \tag{11.31}$$ {eq:20}

Definition 11 (Lie algebra $\mathfrak{g}$ of a Lie group $\mathcal{G}$) *The Lie algebra $\mathfrak{g}$ of a Lie group $\mathcal{G}$ is defined as the linear space of all tangents to G at the identity I. The Lie bracket in $\mathfrak{g}$ is defined as*

$$[a, b] = \frac{\partial^2}{\partial s \partial t} \rho(s) \sigma(t) \rho(-s) \Big|_{s=t=0} \tag{11.32}$$ {eq:21}

where $\rho(s)$ and $\sigma(t)$ are two smooth curves on $\mathcal{G}$ such that $\rho(0) = \sigma(0) = I$, and $\dot{\rho}(0) = a$ and $\dot{\sigma}(0) = b$.

11.2.3 Actions of a group $\mathcal{G}$ on manifold $\mathcal{M}$

Definition 12 *A left action of Lie Group $\mathcal{G}$ on a manifold $\mathcal{M}$ is a smooth map $\Lambda^l : \mathcal{G} \times \mathcal{M} \to \mathcal{M}$ satisfying*

$$\begin{array}{rcl} \Lambda^l(I, y) & = & y, \quad \forall y \in \mathcal{M} \\ \Lambda^l(p, \Lambda(r, y)) & = & \Lambda^l(p \cdot r, y), \quad \forall p, r \in \mathcal{G}, \quad \forall y \in \mathcal{M}. \end{array} \tag{11.33}$$ {eq:22}

Definition 13 *A right action of Lie Group $\mathcal{G}$ on a manifold $\mathcal{M}$ is a smooth map $\Lambda^r : \mathcal{M} \times \mathcal{G} \to \mathcal{M}$ satisfying*

$$\begin{array}{rcl} \Lambda^r(y, I) & = & y, \quad \forall y \in \mathcal{M} \\ \Lambda^r(\Lambda(y, r), p) & = & \Lambda^r(y, r \cdot p), \quad \forall p, r \in \mathcal{G}, \quad \forall y \in \mathcal{M}. \end{array} \tag{11.34}$$ {eq:23}

A given smooth curve $S(\cdot) : t \in \mathbb{R} \mapsto S(t) \in \mathcal{G}$ in $\mathcal{G}$ such that $S(0) = I$ produces a flow $\Lambda^l(S(t), \cdot)$ (resp. $\Lambda^r(\cdot, S(t))$) on $\mathcal{M}$ and by differentiation we find a tangent vector field

$$F(y) = \frac{d}{dt}(\Lambda^l(S(t), y) \Big|_{t=0} \quad (\text{resp. } F(y) = \frac{d}{dt}(\Lambda^r(y, S(t)) \Big|_{t=0}) \tag{11.35}$$ {eq:24}

that defines a ordinary differential equation on a Lie Group

$$\dot{y}(t) = F(y(t)) = \frac{d}{dt}(\Lambda^l(S(t), y) \Big|_{t=0} \quad (\text{resp. } \dot{y}(t) = F(y(t)) = \frac{d}{dt}(\Lambda^r(y, S(t)) \Big|_{t=0}) \tag{11.36}$$ {eq:25}

Lemma 2 *Let $\lambda^l_* : \mathfrak{g} \to \mathcal{X}(\mathcal{M})$ (resp. $\lambda^r_* : \mathfrak{g} \to \mathcal{X}(\mathcal{M})$ be defined as*

$$\lambda^l_*(a)(y) = \frac{d}{ds} \Lambda^l(\rho(s), y) \Big|_{s=0} \quad (\textit{ resp. } \lambda^r_*(a)(y) = \frac{d}{ds} \Lambda^r(y, \rho(s)) \Big|_{s=0}) \tag{11.37}$$ {eq:26}

where $\rho(s)$ is a curve in $\mathcal{G}$ such that $\rho(0) = I$ and $\dot{\rho}(0) = a$. Then λ^l_8 is a linear map between Lie algebras such that

$$[a, b]_{\mathfrak{g}} = [\lambda^l_*(a), \lambda^l_*(b)]_{\mathcal{X}(\mathcal{M})}. \tag{11.38}$$ {eq:27}

The following product between an element of an algebra $a \in \mathfrak{g}$ with an element of a group $\sigma \in \mathcal{G}$ can be defined. This will served as a basis for defining the exponential map.

Definition 14 *We define the left product $(\cdot, \cdot)^l : \mathfrak{g} \times \mathcal{G} \to \mathcal{G}$ of an element of an algebra $a \in \mathfrak{g}$ with an element of a group $\sigma \in \mathcal{G}$ as*

$$(a, \sigma)^l = a \cdot \sigma = \frac{d}{ds} \rho(s) \cdot \sigma \Big|_{s=0} \tag{11.39}$$ {eq:28}

where $\rho(s)$ is a smooth curve such that $\dot{\rho}(0) = a$ and $\rho(0) = I$. In the same way, we can define the right product $(\cdot, \cdot)^r : \mathcal{G} \times \mathfrak{g} \to \mathcal{G}$

$$(\sigma, a)^r = \sigma \cdot a = \frac{d}{ds} \sigma \cdot \rho(s) \Big|_{s=0} \tag{11.40}$$ {eq:29}

11.2.4 Exponential map

Definition 15 *Let $\mathcal{G}$ be a Lie group and $\mathfrak{g}$ its Lie algebra. The exponential mapping $exp : \mathfrak{g} \to \mathcal{G}$ is defined as $\exp(a) = \sigma(1)$ where $\sigma(t)$ satisfies the differential equation*

$$\dot{\sigma}(t) = a \cdot \sigma(t), \quad \sigma(0) = I. \tag{11.41}$$ {eq:30}

Let us define a^k as

$$\begin{cases} a^k = \underbrace{a \cdot a \cdot \ldots a \cdot a}_{k \text{ times}} \text{ for } k \geq 1 \\ a^0 = I \end{cases} \tag{11.42}$$ {eq:31}

The exponential map can be expressed as

$$\exp(at) = \sum_{k=0}^{\infty} \frac{(ta)^k}{k!} \tag{11.43}$$ {eq:32}

since it is a solution of (11.41). A simple computation allows to check this claim:

$$\frac{d}{dt}\exp(at) = \sum_{k=1}^{\infty} k t^{k-1} \frac{a^k}{k!} = a \cdot \sum_{k=0}^{\infty} t^k \frac{a^k}{k!} = a \cdot \exp(at). \tag{11.44}$$ {eq:33}

A similar computation gives

$$\frac{d}{dt}\exp(at) = \sum_{k=0}^{\infty} t^k \frac{a^k}{k!} \cdot a = \exp(at) \cdot a. \tag{11.45}$$ {eq:34}

The exponential mapping $exp : \mathfrak{g} \to \mathcal{G}$ can also be defined as $\exp(a) = \sigma(1)$ where $\sigma(t)$ satisfies the differential equation

$$\dot{\sigma}(t) = \sigma(t) \cdot a, \quad \sigma(0) = I. \tag{11.46}$$ {eq:35}

{Theorem:solutio

Theorem 1 *Let $\Lambda^l : \mathcal{G} \times \mathcal{M} \to \mathcal{M}$ be a left group action and $\lambda^l : \mathfrak{g} \to \mathcal{X}(\mathcal{M})$ the corresponding Lie algebra homomorphism. For any $a \in \mathfrak{g}$ the flow of the vector field $F = \lambda^l_a(a)$, i.e. the solution of the equation*

$$\dot{y}(t) = F(y(t)) = \lambda^l_*(a)(y(t)), \quad t \geq 0, y(0) = y_0 \in \mathcal{M}, \tag{11.47}$$ {eq:36}

is given as

$$y(t) = \Lambda^l(\exp(ta), y_0). \tag{11.48}$$ {eq:37}

Let $\Lambda^r : \mathcal{M} \times \mathcal{G} \to \mathcal{M}$ be a right group action and $\lambda^r : \mathfrak{g} \to \mathcal{X}(\mathcal{M})$ the corresponding Lie algebra homomorphism. For any $a \in \mathfrak{g}$ the flow of the vector field $F = \lambda^r_*(a)$, i.e. the solution of the equation

$$\dot{y}(t) = F(y(t)) = \lambda^r_*(a)(y(t)), \quad t \geq 0, y(0) = y_0 \in \mathcal{M}, \tag{11.49}$$ {eq:38}

is given as

$$y(t) = \Lambda^r(y_0, \exp(ta)). \tag{11.50}$$ {eq:39}

11.2.5 Translation (Trivialization) maps

The left and right translation maps defined by

$$\begin{array}{rccl} L_z : \mathcal{G} \times \mathcal{G} & \to & \mathcal{G} & \text{(left translation map)} \\ y & \mapsto & z \cdot y & \end{array} \tag{11.51}$$ {eq:148}

and

$$\begin{array}{rccl} R_z(y) : \mathcal{G} \times \mathcal{G} & \to & \mathcal{G} & \text{(right translation map)} \\ y & \mapsto & y \cdot z & \end{array} \tag{11.52}$$ {eq:149}

If we identify the manifold $\mathcal{M}$ with the group $\mathcal{G}$, The left and right translations can be interpreted as the simplest example of group action on the manifold. Note that the left translation map can be viewed as a left or right action on the group.

If we consider $L_z(y)$ as a right group action $L_z(y) = \Lambda^r(z, y) = z \cdot y$, by differentiation we get a $L'_z : T\mathfrak{g} \cong \mathfrak{g} \to T_z\mathcal{G}$ with $\dot{\rho}(0) = a$ such that

$$\lambda^r_*(a)(z) = L'_z(a) = \frac{d}{ds}\Lambda^r(z, \rho(s))\Big|_{s=0} = z \cdot a \tag{11.53}$$ {eq:150}

The map

$$\begin{array}{rccc} L'_z : & \mathfrak{g} & \to & T_z\mathcal{G} \\ & a & \mapsto & z \cdot a \end{array} \tag{11.54}$$ {eq:152}

determines an isomorphism of $\mathfrak{g}$ with the tangent space $T_z\mathcal{G}$. In other words, the tangent space can be identified to $\mathfrak{g}$ as

$$T_z\mathcal{G} = \{L'_z(a) = z \cdot a \mid a \in \mathfrak{g}\} \tag{11.55}$$ {eq:153}

Respectively, if we consider $R_z(y)$ as a left group action $R_z(y) = \Lambda^l(y, z) = y \cdot z$, by differentiation we get a $R'_z : T\mathfrak{g} \cong \mathfrak{g} \to T_z\mathcal{G}$ with $\dot{\rho}(0) = a$ such that

$$\lambda^l_*(a)(z) = R'_z(a) = \frac{d}{ds}\Lambda^l(\rho(s), z)\Big|_{s=0} = a \cdot z \tag{11.56}$$ {eq:150}

The map

$$\begin{array}{rccc} R'_z : & \mathfrak{g} & \to & T_z\mathcal{G} \\ & a & \mapsto & a \cdot z \end{array} \tag{11.57}$$ {eq:152}

determines an isomorphism of $\mathfrak{g}$ with the tangent space $T_z\mathcal{G}$. In other words, the tangent space can be identified to $\mathfrak{g}$ as

$$T_z\mathcal{G} = \{R'_z(a) = a \cdot z \mid a \in \mathfrak{g}\} \tag{11.58}$$ {eq:153}

Any tangent vector $F : \mathcal{G} \to T_z\mathcal{G}$ can be written in either of the forms

$$F(z) = L'_z(f(a)) = R'_z(g(z)) \tag{11.59}$$ {eq:155}

where $f, g\mathcal{G} \to \mathfrak{g}$.

11.2.6 Adjoint representation

Definition 16 *Let $p \in \mathcal{G}$ and let $\sigma(t)$ be a smooth curve on $\mathcal{G}$ such that $\sigma(0) = I$ and $\dot{\sigma}(0) = b \in \mathfrak{g}$. The adjoint representation is defined as*

$$\mathrm{Ad}_p(b) = \frac{d}{dt}p\sigma(t)p^{-1}\Big|_{t=0} \tag{11.60}$$ {eq:40}

The derivative of Ad *with respect to the first argument is denoted* ad. *Let $\rho(s)$ be a smooth curve on $\mathcal{G}$ such that $\rho(0) = I$ and $\dot{\rho}(0) = a$, it yields:*

$$\mathrm{ad}_a(b) = \frac{d}{ds}\,\mathrm{Ad}_{\rho(s)}(b)\Big|_{s=0} = [a, b] \tag{11.61}$$ {eq:41}

The adjoint representation can also be expressed with the map

$$\mathrm{Ad}_p(b) = (L_p \cdot R_{p^{-1}})'(b) = (L'_p \cdot R'_{p^{-1}})(b) = p \cdot b \cdot p^{-1} \tag{11.62}$$ {eq:154}

For a tangent vector given in (11.59), we have

$$g(z) = Ad_z(f(z)) \tag{11.63}$$ {eq:151}

Another important relation relating Ad, ad and exp is

$$\mathrm{Ad}_{\exp(a)} = \exp \mathrm{ad}_a \tag{11.64}$$ {eq:164}

11.2.7 Differential of the exponential map

There are multiple ways to represent the differential of $\exp(\cdot)$ at a point $a \in \mathfrak{g}$. Let us start by the following definition of the differential map at $a \in \mathfrak{g}$

$$\begin{aligned} \exp'_a \quad : \quad & \mathfrak{g} \to T_{exp(a)}\mathcal{G} \\ & v \mapsto \exp'_a(v) = \left.\frac{d}{dt}\exp(a+tv)\right|_{t=0} \end{aligned} \tag{11.65}$$ {eq:147}

The definition is very similar to the definition of the directional derivative of exp in the direction $v \in \mathfrak{g}$ at a point $a \in \mathfrak{g}$. Using the expression (11.58) of the tangent space at $\exp(a)$, we can defined another expression of the differential map denoted as $\mathrm{d}^{\mathrm{l}}\exp_a : \mathfrak{g} \to \mathfrak{g}$ such that

$$\mathrm{d}^{\mathrm{l}}\exp_a = L'_{\exp^{-1}(a)} \cdot \exp'_a = L'_{\exp(-a)} \cdot \exp'_a \tag{11.66}$$ {eq:156}

This expression appears as a trivialization of the differential map $\exp'_a$. Using the expression of L'_z in (11.57). In [4, Theorem 2.14.13], an explicit formula relates $\mathrm{d}^{\mathrm{l}}\exp_a$ to the iteration of the adjoint operator:

$$\mathrm{d}^{\mathrm{l}}\exp_a(b) = \sum_{k=0}^{\infty} \frac{(-1)^k}{(k+1)!}(\mathrm{ad}_a(b))^k := \frac{e - \exp \cdot \mathrm{ad}_a}{\mathrm{ad}_a}(b) \tag{11.67}$$ {eq:43}

where $(\mathrm{ad}_a)^k$ is the kth iteration of the adjoint operator:

$$\begin{cases} (\mathrm{ad}_a)^k(b) = \underbrace{[a,[a,[\ldots,a,[a,b]]]}_{k \text{ times}} \text{ for } k \geq 1 \\ (\mathrm{ad}_a)^0(b) = b \end{cases} \tag{11.68}$$ {eq:44}

It is also possible to define the right trivialized differential of the exponential map

$$\mathrm{d}^{\mathrm{r}}\exp_a = R'_{\exp^{-1}(a)} \cdot \exp'_a = R'_{\exp(-a)} \cdot \exp'_a \tag{11.69}$$ {eq:162}

that is

$$\mathrm{d}^{\mathrm{r}}\exp_a(b) = \exp'_a(b) \cdot \exp(-a) \tag{11.70}$$ {eq:163}

With these expression, we have equivalently for

$$\exp'_a(b) = \exp_a \cdot \mathrm{d}^{\mathrm{l}}\exp_a(b) \quad \text{ and } \exp'_a(b) = \mathrm{d}^{\mathrm{r}}\exp_a(b) \cdot \exp(a) \tag{11.71}$$ {eq:157}

To avoid to burden to much the notation, we introduced the unified definition of the differential map that corresponds to $\mathrm{dexp} = \mathrm{d}^{\mathrm{r}}\exp$

Definition 17 *The differential of the exponential mapping, denoted by* $\mathrm{dexp}_a : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$ *is defined as the "right trivialized" tangent of the exponential map*

$$\frac{d}{dt}(\exp(a(t))) = \mathrm{dexp}_{a(t)}(a'(t))\exp(a(t)) \tag{11.72}$$ {eq:42}

An explicit formula relates dexp_a to the iteration of the adjoint operator:

$$\mathrm{dexp}_a(b) = \sum_{k=0}^{\infty} \frac{1}{(k+1)!}(\mathrm{ad}_a(b))^k := \frac{\exp \cdot \mathrm{ad}_a - e}{\mathrm{ad}_a}(b) \tag{11.73}$$ {eq:43}

11.2.1 Redaction note V. ACARY
Say what is not the Jacobian in $\mathbb{R}^4$

As for Ad_a and ad_a, the mapping $\mathrm{dexp}_a(b)$ is a linear mapping in its second argument for a fixed a. Using the relation (11.64), we can also relate the right and the lest trivialization tangent

$$\mathrm{d}^1\exp_a(b) = (\mathrm{Ad}_{\exp(a)} \cdot \mathrm{dexp}(a))(b) = (\exp(\mathrm{ad}_{-a}) \cdot \frac{e - \exp \cdot \mathrm{ad}_a}{\mathrm{ad}_a})(b) = \frac{e - \exp \cdot \mathrm{ad}_{-a}}{\mathrm{ad}_a}(b) = \mathrm{dexp}_{-a}(b) \tag{11.74}$$ {eq:165}

It is also possible to define the the "left trivialized" tangent of the exponential map

$$\frac{d}{dt}(\exp(a(t))) = \exp(a(t))\, \mathrm{d}^1\exp_{a(t)}(a'(t)) = \exp(a(t))\, \mathrm{dexp}_{-a(t)}(a'(t)) \tag{11.75}$$ {eq:46}

11.2.2 Redaction note V. ACARY
other notation and Lie derivative

$$Df \cdot \widehat{\Omega}(p) = (\widehat{\Omega}^r f)(p) \tag{11.76}$$ {eq:178}

Inverse of the exponential map The function dexp_a is an analytical function so it possible to invert it to get

$$\mathrm{dexp}_a^{-1} = \sum_{k=0}^{\infty} \frac{B_k}{(k)!}(\mathrm{ad}_a)^k(b) \tag{11.77}$$ {eq:45}

where B_k are the Bernouilli number.

11.2.8 Differential of a map $f : \mathcal{G} \to \mathfrak{g}$

We follow the notation developed in [3]. Let us first define the differential of the map $f : \mathcal{G} \to \mathfrak{g}$ as

$$\begin{array}{rcl} f'_z : T_z\mathcal{G} & \to & T_{f(z)}\mathfrak{g} \cong \mathfrak{g} \\ b & \mapsto & \left.\dfrac{d}{dt} f(z \cdot \exp(tL'_{z^{-1}}(b)))\right|_{t=0} \end{array} \tag{11.78}$$ {eq:166}

The image of b by f'_z is obtained by first identifying b with an element of $v \in \mathfrak{g}$ thanks to the left representation of $T_{f(z)}\mathfrak{g}$ view the left translation map $v = tL'_z(b)$. The exponential mapping transforms v an element y of the Lie Group $\mathcal{G}$. Then f'_z is obtained by

$$f'_z(b) = \lim_{t \to 0} \frac{f(z \cdot y) - f(z)}{t} \tag{11.79}$$ {eq:167}

As we have done for the exponential mapping, it is possible to get a left trivialization of

$$\mathrm{d}\, f_z = (f \cdot L_z)' = f'_z \cdot L'_z \tag{11.80}$$ {eq:169}

thus

$$\mathrm{d}\, f_z(a) = f'_z \cdot L'_z(a) = f'_z(L'_z(a)) = \left.\frac{d}{dt} f(z \cdot \exp(ta))\right|_{t=0} \tag{11.81}$$ {eq:170}

Newton Method Let us imagine that we want to solve $f(y) = 0$ for $y \in \mathcal{G}$. A newton method can be written as

11.3 Lie group $SO(3)$ of finite rotations and Lie algebra $\mathfrak{so}(3)$ of infinitesimal rotations

The presentation is this section follows the notation and the developments taken from [2**?**]. For more details on Lie groups and Lie algebra, we refer to [4] and [**?**].

The Lie group $SO(3)$ is the group of linear proper orthogonal transformations in $\mathbb{R}^3$ that may be represented by a set of matrices in $\mathbb{R}^{3\times 3}$ as

$$SO(3) = \{R \in \mathbb{R}^{3\times 3} \mid R^T R = I, det(R) = +1\} \tag{11.82}$$ {eq:47}

with the group law given by $R_1 \cdot R_2 = R_1 R_2$ for $R_1, R_2 \in SO(3)$. The identity element is $e = I_{3\times 3}$. At any point of $R \in SO(3)$, the tangent space $T_R SO(3)$ is the set of tangent vectors at a point R.

Left representation of the tangent space at R, $T_R SO(3)$ Let $S(t)$ be a smooth curve $S(\cdot) : \mathbb{R} \to SO(3)$ in $SO(3)$. An element a of the tangent space at R is given by

$$a = \left.\frac{d}{dt} S(t)\right|_{t=0} \tag{11.83}$$ {eq:174}

such that $S(0) = R$. Since $S(t) \in SO(3)$, we have $\frac{d}{dt}(S(t)) = \dot{S}(t)S^T(t) + S(t)\dot{S}^T(t) = 0$. At $t = 0$, we get $aR^T + Ra^T = 0$. We conclude that it exists a skew–symmetric matrix $\tilde{\Omega} = R^T a$ such that $\tilde{\Omega}^T + \tilde{\Omega} = 0$. Hence, a possible representation of $T_R SO(3)$ is

$$T_R SO(3) = \{a = R\tilde{\Omega} \in \mathbb{R}^{3\times 3} \mid \tilde{\Omega}^T + \tilde{\Omega} = 0\}. \tag{11.84}$$ {eq:49}

For $R = I$, the tangent space is directly given by the set of skew–symmetric matrices:

$$T_I SO(3) = \{\tilde{\Omega} \in \mathbb{R}^{3\times 3} \mid \tilde{\Omega}^T + \tilde{\Omega} = 0\}. \tag{11.85}$$ {eq:50}

The tangent space $T_I SO(3)$ with the Lie Bracket $[\cdot, \cdot]$ defined by the matrix commutator

$$[A, B] = AB - BA \tag{11.86}$$ {eq:51}

is a Lie algebra that is denoted by

$$\mathfrak{so}(3) = \{\Omega \in \mathbb{R}^{3\times 3} \mid \Omega + \tilde{\Omega}^T = 0\}. \tag{11.87}$$ {eq:53}

For skew symmetric matrices, the commutator can be expressed with the cross product in $\mathbb{R}^3$

$$[\tilde{\Omega}, \tilde{\Gamma}] = \tilde{\Omega}\tilde{\Gamma} - \tilde{\Gamma}\tilde{\Omega} = \widetilde{\Omega \times \Gamma} \tag{11.88}$$ {eq:52}

We use $T_I SO(3) \cong \mathfrak{so}(3)$ whenever there is no ambiguity.

The notation $\tilde{\Omega}$ is implied by the fact that the Lie algebra is isomorphic to $\mathbb{R}^3$ thanks to the operator $\widetilde{(\cdot)} : \mathbb{R}^3 \to \mathfrak{so}(3)$ and defined by

$$\widetilde{(\cdot)} : \Omega \mapsto \tilde{\Omega} = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \tag{11.89}$$ {eq:54}

Note that $\tilde{\Omega} x = \Omega \times x$.

A special (right) action of Lie Group $\mathcal{G}$ on a manifold $\mathcal{M}$. Let us come back to the representation of $T_R SO(3)$ given in (11.84). It is clear it can expressed with a representation that relies on $\mathfrak{so}(3)$

$$T_R SO(3) = \{a = R\tilde{\Omega} \in \mathbb{R}^{3\times 3} \mid \tilde{\Omega} \in \mathfrak{so}(3)\}. \tag{11.90}$$ {eq:58}

With (11.90), we see that there is a linear map that relates $T_R SO(3)$ to $\mathfrak{so}(3)$. This can be formalize by noting that the left translation map for a point $R \in SO(3)$

$$\begin{array}{rcl} L_R & : & SO(3) \to SO(3) \\ & & S \mapsto L_R(S) = R \cdot S = RS \end{array} \tag{11.91}$$ {eq:59}

which is diffeomorphism on $SO(3)$ is a group action. In our case, we identify the manifold and the group. Hence, the mapping L_R can be viewed as a left or a right group action. We choose a right action such that $\Lambda^r(R, S) = L_R(S) = R \cdot S$. By differentiation, we get a mapping $L'_R : T_I\mathfrak{so}(3) \cong \mathfrak{so}(3) \to T_R SO(3)$. For a given $\tilde{\Omega} \in \mathfrak{so}(3)$ and a point R, the differential L'_R by computing the tangent vector field $\lambda^r_*(a)(R)$ of the group action $\Lambda^r(R, S)$ for a smooth curve $S(t) : \mathbb{R} \to S0(3)$ such that $\dot{S}(0) = \tilde{\Omega}$:

$$\lambda^r_*(a)(R) := \frac{d}{dt}\Lambda^r(R, S(t))\Big|_{t=0} = \frac{d}{dt}L_R(S(t))\Big|_{t=0} = \frac{d}{dt}R \cdot S(t)\Big|_{t=0} = R \cdot \dot{S}(0) = R\tilde{\Omega} \in X(\mathcal{M}) \tag{11.92}$$ {eq:60}

Therefore, the vector field in (11.92) is a tangent vector field that defines a Lie-Type ordinary differential equation

$$\dot{R}(t) = \lambda^r_*(a)(R(t)) = R(t)\tilde{\Omega} \tag{11.93}$$ {eq:61}

In [1], the linear operator $\lambda^r_*(a)$ is defined as the directional derivative with respect to S an denoted $DL_R(S)$. It defines a diffeomorphism between $T_S SO(3)$ and $T_{RS} SO(3)$. In particular, for $S = I_{3\times 3}$, we get

$$\begin{array}{rcl} DL_R(I_{3\times 3}) : \mathfrak{so}(3) & \to & T_R SO(3) \\ \tilde{\Omega} & \mapsto & DL_R(I_{3\times 3}).\tilde{\Omega} = R\tilde{\Omega} \end{array} \tag{11.94}$$ {eq:62}

We end up with a possible representation of $T_R SO(3)$ as

$$T_R SO(3) = \{\tilde{\Omega}_R \mid \tilde{\Omega}_R = DL_R(I_{3\times 3}).\tilde{\Omega} = R\tilde{\Omega}, \tilde{\Omega} \in \mathfrak{so}(3)\}. \tag{11.95}$$ {eq:63}

In other words, a tangent vector $\tilde{\Omega} \in \mathfrak{so}(3)$ defines a left invariant vector field on $SO(3)$ at the point R given by $R\tilde{\Omega}$.

11.3.1 Redaction note V. ACARY

what happens at $S(0) = R$, with $a = R\tilde{\Omega} = \dot{S}(0)$ and then $\dot{y}(t) = F(y(t)) = R\tilde{\Omega}y(t) = R\Omega \times y(t) = \dot{S}(0)y(t)$. What else ?

Exponential map $\exp_{\mathfrak{so}(3)} \to SO(3)$ The relations (11.35) and (11.36) shows that is possible to define tangent vector field from a group action. We can directly apply Theorem 1 and we get that the solution of

$$\begin{cases} \dot{R}(t) = \lambda^r_*(a)(R(t)) = R(t)\tilde{\Omega} \\ R(0) = R_0 \end{cases} \tag{11.96}$$ {eq:130}

is

$$R(t) = R_0 \exp(t\tilde{\Omega}) \tag{11.97}$$ {eq:138}

Let us do the computation in this case. Let us assume that the solution can be sought as $R(t) = \Lambda^r(y_o, S(t))$. The initial condition imposes that $R(0) = R_0 = \Lambda(R_0, I) = \Lambda(R_0, S(0))$ that implies

$S(0) = I$. Since $\Lambda(R_0, S(t))$ is the flow that is produces by $S(t)$ and let us try to find the relation satisfied by $S(\cdot)$. For a smooth curve $T(s) \in SO(3)$ such that $\dot{T}(0) = \tilde{\Omega}$, we have

$$\begin{aligned} \dot{R}(t) = \lambda^r_*(\tilde{\Omega})(R(t)) &= \frac{d}{ds}\Lambda^r(R(t), T(s))\Big|_{s=0} \\ &= \frac{d}{ds}\Lambda^r(\Lambda(R_0, S(t)), T(s))\Big|_{s=0} \\ &= \frac{d}{ds}(\Lambda^r(R_0, S(t)\cdot T(s))\Big|_{s=0} \\ &= D_2\Lambda^r(R_0, \cdot S(t)\cdot \dot{T}(0)) \\ &= D_2\Lambda^r(R_0, S(t)\cdot\tilde{\Omega}) \end{aligned} \tag{11.98}$$ {eq:64}

On the other side, the relation $y(t) = \Lambda^r(y_0, S(t))$ gives $\dot{y}(t) = D_2\Lambda^r(y_0, S'(t))$ and we conclude that

$$\begin{cases} \dot{S}(t) = S(t)\cdot\tilde{\Omega} = S(t)\tilde{\Omega} \\ S(0) = I. \end{cases} \tag{11.99}$$ {eq:65}

The ordinary differential equation (11.99) is a matrix ODE that admits the following solution

$$S(t) = \mathrm{expm}(t\tilde{\Omega}) \tag{11.100}$$ {eq:66}

where $\exp : \boldsymbol{R}^{3\times3} \to \boldsymbol{R}^{3\times3}$ is the matrix exponential defined by

$$\mathrm{expm}(A) = \sum_{k=0}^{\infty}\frac{1}{k!}(A)^k. \tag{11.101}$$ {eq:67}

We conclude that $R(t) = \Lambda(R_0, S(t)) = R_0\,\mathrm{expm}(t\tilde{\Omega})$ is the solution of (11.46).

We can use the closed form solution for the matrix exponential of $t\tilde{\Omega} \in \mathfrak{so}(3)$ as

$$\mathrm{expm}(t\tilde{\Omega}) = I_{3\times3} + \frac{\sin t\theta}{\theta}\tilde{\Omega} + \frac{(\cos t\theta - 1)}{\theta^2}\tilde{\Omega}^2 \tag{11.102}$$ {eq:68}

with $\theta = \|\Omega\|$. For given $\tilde{\Omega} \in \mathfrak{so}(3)$, we have

$$\det(\tilde{\Omega}) = \det(\tilde{\Omega}^T) = \det(-\tilde{\Omega}^T) = (-1)^3\det(\tilde{\Omega}) = -\det(\tilde{\Omega}) \tag{11.103}$$ {eq:69}

that implies that $\det(\tilde{\Omega}) = 0$. From (11.102), we conclude that

$$\det(\mathrm{expm}(t\tilde{\Omega})) = 1. \tag{11.104}$$ {eq:70}

Furthermore, we have $\mathrm{expm}(t\tilde{\Omega})\,\mathrm{expm}(-t\tilde{\Omega}) = \mathrm{expm}(t(\tilde{\Omega} - \tilde{\Omega})) = I$. We can verify that $\mathrm{expm}(t\tilde{\Omega}) \in SO(3)$.

Adjoint representation In the case of $SO(3)$, the definition of the operator Ad gives

$$\mathrm{Ad}_R(\tilde{\Omega}) = R\tilde{\Omega}R^T \tag{11.105}$$ {eq:121}

and then mapping $\mathrm{ad}_{\tilde{\Omega}}(\tilde{\Gamma})$ is defined by

$$\mathrm{ad}_{\tilde{\Omega}}(\tilde{\Gamma}) = \tilde{\Omega}\tilde{\Gamma} - \tilde{\Gamma}\tilde{\Omega} = [\tilde{\Omega}, \tilde{\Gamma}] = \widetilde{\Omega\times\Gamma}. \tag{11.106}$$ {eq:56}

Using the isomorphism between $\mathfrak{so}(3)$ and $\boldsymbol{R}^3$, it possible the define the mapping $\mathrm{ad}_\Omega(\Gamma) : \boldsymbol{R}^3 \times \boldsymbol{R}^3 \to \boldsymbol{R}^3$ with the realization of the Lie algebra in $\boldsymbol{R}^3$ as

$$\mathrm{ad}_\Omega(\Gamma) = \tilde{\Omega}\Gamma = \Omega\times\Gamma \tag{11.107}$$ {eq:55}

Differential of the exponential map dexpm The differential of the exponential mapping, denoted by dexpm is defined as the 'right trivialized' tangent of the exponential map

$$\frac{d}{dt}(\exp(\tilde{\Omega}(t))) = \mathrm{dexp}_{\tilde{\Omega}(t)}(\frac{d\tilde{\Omega}(t)}{dt})\exp(\tilde{\Omega}(t)) \tag{11.108}$$ {eq:71}

The differential of the exponential mapping, denoted by dexpm is defined as the 'left trivialized' tangent of the exponential map

$$\frac{d}{dt}(\exp(\tilde{\Omega}(t))) = \mathrm{dexp}_{\tilde{\Omega}(t)}(\frac{d\tilde{\Omega}(t)}{dt})\exp(\tilde{\Omega}(t)) \tag{11.109}$$ {eq:72}

Using the formula (11.73) and the fact that $\mathrm{ad}_{\Omega}(\Gamma) = \tilde{\Omega}\Gamma$, we can write the differential as

$$\begin{aligned} \mathrm{dexp}_{\tilde{\Omega}}(\tilde{\Gamma}) &= \sum_{k=0}^{\infty} \frac{1}{(k+1)!}\,\mathrm{ad}^k_{\tilde{\Omega}}(\tilde{\Gamma}) \\ &= \sum_{k=0}^{\infty} \frac{1}{(k+1)!}\tilde{\Omega}^k\tilde{\Gamma} \end{aligned} \tag{11.110}$$ {eq:122}

Using again the fact that $\tilde{\Omega}^3 = -\theta^2\tilde{\Omega}$, we get

$$\begin{aligned} \mathrm{dexp}_{\tilde{\Omega}} &= \sum_{k=0}^{\infty} \frac{1}{(k+1)!}\tilde{\Omega}^k \\ &= I + \sum_{k=0}^{\infty} \frac{(-1)^k}{((2(k+1))!}\theta^{2k}\tilde{\Omega} + \sum_{k=0}^{\infty} \frac{(-1)^k}{((2(k+1)+1)!}\theta^{2k}\tilde{\Omega}^2 \end{aligned} \tag{11.111}$$ {eq:123}

Hence, we get

$$\mathrm{dexp}_{\tilde{\Omega}} = I + \frac{(1-\cos(\theta))}{\theta^2}\tilde{\Omega} + \frac{(\theta-\sin(\theta))}{\theta^3}\tilde{\Omega}^2 \tag{11.112}$$ {eq:124}

Since $\mathrm{dexp}_{\tilde{\Omega}}$ is a linear mapping from $\mathfrak{so}(3)$ to $\mathfrak{so}(3)$, we will use the following notation

$$\mathrm{dexp}_{\tilde{\Omega}}\,\tilde{\Gamma} := T(\Omega)\tilde{\Gamma} \tag{11.113}$$ {eq:172}

with

$$T(\Omega) := I + \frac{(1-\cos(\theta))}{\theta^2}\tilde{\Omega} + \frac{(\theta-\sin(\theta))}{\theta^3}\tilde{\Omega}^2 \in I\!R^{3\times 3} \tag{11.114}$$ {eq:173}

11.3.1 Newton method and differential of a map $f : \mathcal{G} \to \mathfrak{g}$

Finally, let us define the differential of the map $f : SO(3) \to \mathfrak{so}(3)$ as

$$\begin{aligned} f'_R : T_R SO(3) &\to T_{f(R)}\mathfrak{so}(3) \cong \mathfrak{so}(3) \\ a &\mapsto \frac{d}{dt} f(R \cdot \mathrm{expm}(tL'_{R^{-1}}(a)))\Big|_{t=0} \end{aligned} \tag{11.115}$$ {eq:183}

The image of b by f'_z is obtained by first identifying a with an element of $\tilde{\Omega} \in \mathfrak{so}(3)$ thanks to the left representation of $T_{f(R)}\mathfrak{so}(3)$ view the left translation map $\tilde{\Omega} = tL'_R(b)$. The exponential mapping transforms $\tilde{\Omega}$ an element S of the Lie Group $SO(3)$. Then f'_z is obtained by

$$f'_R(b) = \lim_{t\to 0} \frac{f(R\cdot S) - f(R)}{t} \tag{11.116}$$ {eq:184}

As we have done for the exponential mapping, it is possible to get a left trivialization of

$$\mathrm{d}\, f_R = (f \circ L_R)' = f'_R \circ L'_R \tag{11.117}$$ {eq:185}

thus

$$\mathrm{d}\, f_R(\tilde{\Omega}) = f'_R \circ L'_R(\tilde{\Omega}) = f'_R(L'_R(\tilde{\Omega})) = \frac{d}{dt} f(R \cdot \mathrm{expm}(t\tilde{\Omega}))\Big|_{t=0} \tag{11.118}$$ {eq:186}

11.3.2 Redaction note V. ACARY

The computation of this differential is non linear with respect to $\tilde{\Omega}$.
not clear if we write $\mathrm{d}\, f_R(\tilde{\Omega})$. Better understand the link with $\mathrm{dexp}_{\tilde{\Omega}}$

Sometimes, it can be formally written as

$$\mathrm{d}\, f_R(\tilde{\Omega}) = C(\tilde{\Omega})\tilde{\Omega} \tag{11.119}$$ {eq:180}

Nevertheless, an explicit expression of $C(\cdot)$ is not necessarily trivial.

Let us consider a first simple example of a mapping $f(R) = \widetilde{Rx}$ for a given $x \in I\!R^3$. The computation yields

$$\begin{aligned}
\mathrm{d}\, f_R(\tilde{\Omega}) &= \frac{d}{dt} R \widetilde{\exp(t\tilde{\Omega})x}\Big|_{t=0} \\
&= R\, \frac{d}{dt} \widetilde{\exp(t\tilde{\Omega})}\Big|_{t=0} x \\
&= R\, \widetilde{\mathrm{dexp}_{\tilde{\Omega}}(\tilde{\Omega})\exp(t\tilde{\Omega})}\big|_{t=0}\, x \\
&= R\, \widetilde{\mathrm{dexp}_{\tilde{\Omega}}(\tilde{\Omega})x} \\
&= R\widetilde{T(\Omega)\tilde{\Omega}x} \\
&= -R\widetilde{T(\Omega)\tilde{x}\Omega}
\end{aligned} \tag{11.120}$$ {eq:181}

In that case, it is difficult to find a expression as in (11.119), but considering the function $g(R)$ such that $f(R) = \tilde{g}(x)$ we get

$$\mathrm{d}\, g_R(\tilde{\Omega}) = -RT(\Omega)\tilde{x}\Omega = C(\Omega)\Omega \tag{11.121}$$ {eq:181}

with

$$C(\Omega) = -RT(\Omega)\tilde{x} \tag{11.122}$$ {eq:182}

11.4 Lie group of unit quaternions $I\!H_1$ and pure imaginary quaternions $I\!H_p$.

In Siconos we choose to parametrize the rotation with a unit quaternion $p \in I\!H$ such that $R = \Phi(p)$. This parameterization has no singularity and has only one redundant variable that is determined by imposing $\|p\| = 1$.

Quaternion definition. There is many ways to define quaternions. The most convenient one is to define a quaternion as a 2×2 complex matrix, that is an element of $\mathbb{C}^{2\times 2}$. For this end, we write for $z \in \mathbb{C}$, $z = a + ib$ with $a, b \in I\!R^2$ and $i^2 = -1$ and its conjugate $\bar{z} = a - ib$. Let $e, , \mathbf{i}, \mathbf{j}, \mathbf{k}$ the following matrices in $\mathbb{C}^{2\times 2}$

$$e = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{i} = \begin{bmatrix} i & 0 \\ 0 & -i \end{bmatrix} \quad \mathbf{j} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad \mathbf{k} = \begin{bmatrix} 0 & i \\ i & 0 \end{bmatrix} \tag{11.123}$$ {eq:127}

Definition 18 *Let $I\!H$ be the set of all matrices of the form*

$$p_0 e + p_1 \mathbf{i} + p_2 \mathbf{j} + p_3 \mathbf{k} \tag{11.124}$$ {eq:128}

where $(p_0, p_1, p_2, p_3) \in I\!R^4$. Every Matrix in $I\!H$ is of the form

$$\begin{bmatrix} x & y \\ -\bar{y} & \bar{x} \end{bmatrix} \tag{11.125}$$ {eq:129}

where $x = p_0 + ip_1$ and $y = p_2 + ip_3$. The matrices in $I\!H$ are called quaternions.

Definition 19 *The null quaternion generated by* $[0,0,0,0] \in \mathbb{R}^4$ *is denoted by* 0 *. Quaternions of the form* $p_1\mathbf{i} + \mathbf{p_2}\mathbf{j} + \mathbf{p_3}\mathbf{k}$ *are called pure quaternions. The set of pure quaternions is denoted by* $\mathbb{H}_p$*.*

With the definition of $\mathbb{H}$ as a set of complex matrices, It can be show that $\mathbb{H}$ is a real vector space of dimension 4 with basis $e, , \mathbf{i}, \mathbf{j}, \mathbf{k}$. Furthermore, with the matrix product, $\mathbb{H}$ is a real algebra.

Representation of quaternions Thanks to the equations (11.124), (11.124) and (11.125), we see that there are several manner to represent a quaternion $p \in \mathbb{H}$. It can be represented as a complex matrix as in (11.125). It can also be represented as a vector in $\mathbb{R}^4$, $p = [p_0, p_1, p_2, p_3]$ with the isomorphism (11.124). In other words, $\mathbb{H}$ is isomorphic to $\mathbb{R}^4$. The first element p_0 can also be viewed as a scalar and three last ones as a vector in $\mathbb{R}^3$ denoted by $\vec{p} = [p_1, p_2, p_3]$, and in that case, $\mathbb{H}$ is viewed as $\mathbb{R} \times \mathbb{R}^3$. The quaternion can be written as $p = (p_0, \vec{p})$.

Quaternion product The quaternion product denoted by $p \cdot q$ for $p, q \in \mathbb{H}_1$ is naturally defined as the product of complex matrices. With its representation in $\mathbb{R} \times \mathbb{R}^3$, the quaternion product is defined by

$$p \cdot q = \begin{bmatrix} p_o q_o - \vec{p}\,\vec{q} \\ p_0 \vec{q} + q_o \vec{p} + \vec{p} \times \vec{q} \end{bmatrix}. \tag{11.126}$$ {eq:73}

Since the product is a matrix product, it is not communicative, but it is associative. The identity element for the quaternion product is

$$e = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = (1, \vec{0}). \tag{11.127}$$ {eq:57}

Let us note that

$$(0, \vec{p}) \cdot (0, \vec{(}q)) = -(0, \vec{q}) \cdot (0, \vec{p}). \tag{11.128}$$ {eq:74}

The quaternion multiplication can also be represented as a matrix operation in $\mathbb{R}^{4\times 4}$. Indeed, we have

$$p \cdot q = \begin{bmatrix} q_0 p_0 - q_1 p_1 - q_2 p_2 - q_3 p_3 \\ q_0 p_1 + q_1 p_0 - q_2 p_3 + q_3 p_2 \\ q_0 p_2 + q_1 p_3 + q_2 p_0 - q_3 p_1 \\ q_0 p_3 - q_1 p_2 + q_2 p_1 + q_3 p_0 \end{bmatrix} \tag{11.129}$$ {eq:75}

that can be represented as

$$p \cdot q = \begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} := [p.]q \tag{11.130}$$ {eq:76}

or

$$p \cdot q = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} := [.q]p \tag{11.131}$$ {eq:77}

Adjoint quaternion, inverse and norm The adjoint quaternion of p is denoted by

$$p^\star = \overline{\begin{bmatrix} x & y \\ -\bar{y} & \bar{x} \end{bmatrix}}^T = \begin{bmatrix} \bar{x} & -y \\ \bar{y} & x \end{bmatrix} = [p_0, -p_1, -p_2, -p_3] = (p_0, -\vec{p}) \tag{11.132}$$

We note that

$$p \cdot p^\star = \begin{bmatrix} x & y \\ -\bar{y} & \bar{x} \end{bmatrix} \begin{bmatrix} \bar{x} & -y \\ \bar{y} & x \end{bmatrix} = \det(\begin{bmatrix} x & y \\ -\bar{y} & \bar{x} \end{bmatrix}) e = (x\bar{x} + y\bar{y})e = (p_0^2 + p_1^2 + p_2^2 + p_3^2)e \tag{11.133}$$ {eq:131}

The norm of a quaternion is given by $|p|^2 = p^\top p = p_o^2 + p_1^2 + p_2^2 + p_3^2$. In particular, we have $p \cdot p^\star = p^\star \cdot p = |p|^2 e$. This allows to define the reciprocal of a non zero quaternion by

$$p^{-1} = \frac{1}{|p|^2} p^\star \tag{11.134}$$ {eq:78}

A quaternion p is said to be unit if $|p| = 1$.

Unit quaternion and rotation For two vectors $x \in I\!R^3$ and $x' \in I\!R^3$, we define the quaternion $p_x = (0, x) \in I\!H_p$ and $p_{x'} = (0, x') \in I\!H_p$. For a given unit quaternion p, the transformation

$$p_{x'} = p \cdot p_x \cdot p^\star \tag{11.135}$$ {eq:79}

defines a rotation R such that $x' = Rx$ given by

$$x' = (p_0^2 - p^\top \vec{p})x + 2p_0(\vec{p} \times x) + 2(\vec{p}^\top x)p = Rx \tag{11.136}$$ {eq:80}

The rotation matrix may be computed as

$$R = \Phi(p) = \begin{bmatrix} 1 - 2p_2^2 - 2p_3^2 & 2(p_1p_2 - p_3p_0) & 2(p_1p_3 + p_2p_0) \\ 2(p_1p_2 + p_3p_0) & 1 - 2p_1^2 - 2p_3^2 & 2(p_2p_3 - p_1p_0) \\ 2(p_1p_3 - p_2p_0) & 2(p_2p_3 + p_1p_0) & 1 - 2p_1^2 - 2p_2^2 \end{bmatrix} \tag{11.137}$$ {eq:81}

Computation of the time derivative of a unit quaternion associated with a rotation. The derivation with respect to time can obtained as follows. The rotation transformation for a unit quaternion is given by

$$p_{x'}(t) = p(t) \cdot p_x \cdot p^\star(t) = p(t) \cdot p_x \cdot p^{-1}(t) \tag{11.138}$$ {eq:82}

and can be derived as

$$\begin{aligned} \dot{p}_{x'}(t) &= \dot{p}(t) \cdot p_x \cdot p^{-1}(t) + p(t) \cdot p_x \cdot \dot{p}^{-1}(t) \\ &= \dot{p}(t) \cdot p^{-1}(t) \cdot p_{x'}(t) + p_{x'}(t) \cdot p(t) \cdot \dot{p}^{-1}(t) \end{aligned} \tag{11.139}$$ {eq:83}

From $p(t) \cdot p^{-1}(t) = e$, we get

$$\dot{p}(t) \cdot p^{-1}(t) + p \cdot \dot{p}^{-1}(t) = 0 \tag{11.140}$$ {eq:84}

so (11.138) can be rewritten

$$\dot{p}_{x'}(t) = \dot{p}(t) \cdot p^{-1}(t) \cdot p_{x'}(t) - p_{x'}(t) \cdot \dot{p}(t) \cdot p^{-1}(t) \tag{11.141}$$ {eq:85}

The scalar part of $\dot{p}(t) \cdot p^{-1}(t)$ is $(\dot{p}(t) \cdot p^{-1}(t))_0 = p_o\dot{p}_0 + \vec{p}^T\vec{\dot{p}}$. Since p is a unit quaternion, we have

$$|p| = 1 \implies \frac{d}{dt}(p^\top p) = 0 = \dot{p}^\top p + p^\top \dot{p} = 2(p_o\dot{p}_0 + \vec{p}^T\vec{\dot{p}}). \tag{11.142}$$ {eq:86}

Therefore, the scalar part $(\dot{p}(t) \cdot p^{-1}(t))_0 = 0$. The quaternion product $\dot{p}(t) \cdot p^{-1}(t)$ and $p_{x'}(t)$ is a product of quaternions with zero scalar part (see (11.128)), so we have

$$\dot{p}_{x'}(t) = 2\dot{p}(t) \cdot p^{-1}(t) \cdot p_{x'}(t). \tag{11.143}$$ {eq:87}

In terms of vector of $I\!R^3$, this corresponds to

$$\dot{x}'(t) = 2\overrightarrow{\dot{p}(t) \cdot p^{-1}(t)} \times x'(t). \tag{11.144}$$ {eq:88}

Since $x'(t) = R(t)x$, we have $\dot{x}' = \dot{R}(t)x = \tilde{\omega}(t)R(t)x = \tilde{\omega}(t)x'(t)$. Comparing (11.143) and (11.144), we get

$$\tilde{\omega}(t) = 2\overrightarrow{\dot{p}(t) \cdot p^{-1}(t)} \tag{11.145}$$ {eq:89}

or equivalently

$$\dot{p}(t) \cdot p^{-1}(t) = (0, \frac{\omega(t)}{2}) \tag{11.146}$$ {eq:90}

Finally, we can conclude that

$$\dot{p}(t) = (0, \frac{\omega(t)}{2}) \cdot p(t). \tag{11.147}$$ {eq:91}

Since $\omega(t) = R(t)\Omega(t)$, we have

$$(0, \omega(t)) = (0, R(t)\Omega(t)) = p(t) \cdot (0, \Omega(t)) \cdot \bar{p}(t) = p(t) \cdot (0, \Omega(t)) \cdot p^{-1}(t) \tag{11.148}$$ {eq:92}

and then

$$\dot{p}(t) = \frac{1}{2} p(t) \cdot (0, \Omega(t)). \tag{11.149}$$ {eq:93}

The time derivation is compactly written

$$\dot{p} = \frac{1}{2} p \cdot (0, \frac{\Omega}{2}) = [p.] p_{\frac{\Omega}{2}} = \Psi(p) \frac{\Omega}{2}, \tag{11.150}$$ {eq:94}

and using the matrix representation of product of quaternion we get

$$\Psi(p) = \begin{bmatrix} -p_1 & -p_2 & -p_3 \\ p_0 & -p_3 & p_2 \\ p_3 & p_0 & -p_1 \\ -p_2 & p_1 & p_0 \end{bmatrix} \tag{11.151}$$ {eq:95}

The relation (11.149) can be also inverted by writing

$$(0, \Omega(t)) = 2p^{-1}(t) \cdot \dot{p}(t) \tag{11.152}$$ {eq:96}

Using again matrix representation of product of quaternion, we get

$$\Omega(t) = \overrightarrow{2p^{-1}(t) \cdot \dot{p}(t)} = 2 \begin{bmatrix} -p_1 & p_0 & p_3 & -p_2 \\ -p_2 & -p_3 & p_0 & p_1 \\ -p_3 & p_2 & -p_1 & p_0 \end{bmatrix} \dot{p}(t) = 2\Psi(p)^\top \dot{p}(t) \tag{11.153}$$ {eq:97}

Note that we have $\Psi^\top(p)\Psi(p) = I_{4\times 4}$ and $\Psi(p)\Psi^\top(p) = I_{3\times 3}$

Lie group structure of unit quaternions. In terms of complex matrices, an unit quaternion p satisfies

$$\det\left(\begin{bmatrix} x & y \\ -\bar{y} & \bar{x} \end{bmatrix}\right) = 1 \tag{11.154}$$ {eq:125}

The set of all unit quaternions that we denote $\mathbb{H}_1$ is the set of unitary matrices of determinant equal to 1. From (11.133), we get that

$$p \cdot p^\star = e \tag{11.155}$$ {eq:126}

It implies that the set $\mathbb{H}_1$ is the set of special unitary complex matrices. The set is a Lie group usually denoted as $SU(2)$. Since we used multiple representation of a quaternion, we continue to use $\mathbb{H}_1 \cong SU(2)$ as a notation but with the Lie group structure implied by $SU(2)$.

Let us compute the tangent vector at a point $p \in \mathbb{H}_1$. Let $q(t)$ be a smooth curve $q(\cdot) : t \in \mathbb{R} \mapsto q(t) \in H_1$ in H_1 such that $q(O) = p$. Since $q(t) \in H_1$, we have $|q(t)| = 1$ and then $\frac{d}{dt}|q(t)| = 2(q_0(0)\dot{q}_0(0) + \vec{q}^T(0)\dot{\vec{q}}(0)) = 0$. At $t = 0$, we get

$$2(p_0 a_0 + \vec{p}^T \vec{a}) = 0. \tag{11.156}$$ {eq:48}

This relation imposes that the quaternions $2p^\star \cdot a \in H_1$ and $2a \cdot p^\star \in H_1$, that is, have to be pure quaternions. Therefore, it exists $\omega \in \mathbb{R}^3$ and $\Omega \in \mathbb{R}^3$ such that

$$(0, \Omega) = 2p^\star \cdot a \tag{11.157}$$ {eq:132}

and

$$(0, \omega) = 2a \cdot p^\star \tag{11.158}$$ {eq:132}

In other terms, the tangent vector spaces at $p \in \mathbb{H}_1$ can be represented as a left representation

$$T_p\mathbb{H}_1 = \{a \mid a = p \cdot (0, \frac{\Omega}{2}), \Omega \in \mathbb{R}^3\} \tag{11.159}$$ {eq:133}

or a right representation

$$T_p\mathbb{H}_1 = \{a \mid a = (0, \frac{\omega}{2}) \cdot p, \omega \in \mathbb{R}^3\} \tag{11.160}$$ {eq:1330}

At $p = e$, we get the Lie algebra defined by

$$\mathfrak{h}_1 = T_e\mathbb{H}_1 = \{a = (0, \frac{\Omega}{2}), \Omega \in \mathbb{R}^3\} \tag{11.161}$$ {eq:134}

equipped with the Lie bracket given by the commutator

$$[p, q] = p \cdot q - q \cdot p. \tag{11.162}$$ {eq:135}

We can easily verify that for $a = (0, \frac{\Omega}{2})$, $b = (0, \frac{\Gamma}{2}) \in \mathfrak{h}_1$, we have

$$[a, b] = (0, \frac{\Omega}{2}) \cdot (0, \frac{\Gamma}{2}) - (0, \frac{\Gamma}{2}) \cdot (0, \frac{\Omega}{2}) = (0, \frac{\Omega \times \Gamma}{2}) \in \mathfrak{h}_1 \tag{11.163}$$ {eq:136}

As for $\mathfrak{so}(3)$, the Lie algebra $\mathfrak{h}_1$ is isomorphic to $\mathbb{R}^3$ thanks to the operator $\widehat{(\cdot)} : \mathbb{R}^3 \to \mathfrak{h}_1$ and defined by

$$\widehat{(\cdot)} : \Omega \mapsto \widehat{\Omega} = (0, \frac{\Omega}{2}) \tag{11.164}$$ {eq:54}

With this operator, the Lie Bracket can be written

$$[\widehat{\Omega}, \widehat{\Gamma}] = \widehat{\Omega \times \Gamma} \tag{11.165}$$ {eq:137}

A special (right) action of Lie Group $\mathcal{G}$ on a manifold $\mathcal{M}$. Let us come back to the representation of $T_p\mathbb{H}_1$ given in (11.159). It is clear it can expressed with a representation that relies on $\mathfrak{h}_1$

$$T_R SO(3) = \{a = p \cdot \widehat{\Omega} \mid \widehat{\Omega} \in \mathfrak{h}_1\}. \tag{11.166}$$ {eq:158}

With (11.90), we see that there is a linear map that relates $T_p\mathbb{H}_1$ to $\mathfrak{h}_1$. This linear map defines a vector field. A special group action is defined by the left translation map for a point $p \in \mathbb{H}_1$

$$\begin{array}{rcl} L_p & : & \mathbb{H}_1 \to \mathbb{H}_1 \\ & & q \mapsto L_p(q) = p \cdot q \end{array} \tag{11.167}$$ {eq:159}

which is diffeomorphism on $\mathbb{H}_1$. In that case, we identify the manifold and the group. So, L_p can be viewed as a left or a right group action. We choose a right action. For our application where $\mathcal{G} = \mathcal{M} = \mathbb{H}_1$ and $\Lambda^r(p, q) = L_p(q) = p \cdot q$, we get

$$\lambda^r_*(a)(p) = \left.\frac{d}{dt} L_p(q(t))\right|_{t=0} = \left.\frac{d}{dt} p \cdot q(t)\right|_{t=0} = p \cdot \dot{q}(0) = p \cdot \dot{q}(0) \in X(\mathcal{M}) \tag{11.168}$$ {eq:160}

for a smooth curve $q(t)$ in $\mathbb{H}_1$. Since $q(\cdot)$ is a smooth curve in $\mathbb{H}_1$, $\dot{q}(0)$ is a tangent vector at the point $q(0) = I$, that is an element $a = \widehat{\Omega} \in \mathfrak{h}_1$ defined by the relation (11.44). Therefore, the vector field in (11.168) is a tangent vector field and we get

$$\dot{p}(t) = \lambda^r_*(a)(p(t)) = p(t) \cdot \widehat{\Omega} \tag{11.169}$$ {eq:161}

Exponential map $\exp q : \mathfrak{h}_1 \to I\!H_1$ We can directly apply Theorem 1 and we get that the solution of

$$\begin{cases} \dot{p}(t) = \lambda^r_*(a)(p(t)) = p(t) \cdot \widehat{\Omega} \\ p(0) = Rp_0 \end{cases} \tag{11.170}$$ {eq:130}

is

$$p(t) = p_0 \exp q(t\widehat{\Omega}) \tag{11.171}$$ {eq:138}

The exponential mapping $\exp q : \mathfrak{h}_1 \to I\!H_1$ can also be defined as $\exp q(\widehat{\Omega}) = q(1)$ where $q(t)$ satisfies the differential equation

$$\dot{q}(t) = q(t) \cdot \widehat{\Omega}, \quad q(0) = e. \tag{11.172}$$ {eq:235}

Using the quaternion product, the exponential map can be expressed as

$$\exp q(t\widehat{\Omega}) = \sum_{k=0}^{\infty} \frac{(t\widehat{\Omega})^k}{k!} \tag{11.173}$$ {eq:232}

since it is a solution of (11.170). A simple computation allows to check this claim:

$$\frac{d}{dt} \exp q(t\widehat{\Omega}) = \sum_{k=1}^{\infty} k t^{k-1} \frac{\widehat{\Omega}^k}{k!} = \sum_{k=0}^{\infty} t^k \frac{t\widehat{\Omega}^k}{k!} \cdot \widehat{\Omega} = \exp q(t\widehat{\Omega}) \cdot \widehat{\Omega}. \tag{11.174}$$ {eq:233}

A closed form relation for the form the quaternion exponential can also be found by noting that

$$\widehat{\Omega}^2 = -\left(\frac{\theta}{2}\right)^2 e, \text{ and } \widehat{\Omega}^3 = -\left(\frac{\theta}{2}\right)^2 \widehat{\Omega}. \tag{11.175}$$ {eq:140}

A simple expansion of (11.173) at $t = 1$ equals

$$\begin{aligned} \exp q(\widehat{\Omega}) &= \sum_{k=0}^{\infty} \frac{(\widehat{\Omega})^k}{k!} \\ &= \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k)!} \left(\frac{\theta}{2}\right)^{2k} e + \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)!} \left(\frac{\theta}{2}\right)^{2k+1} \widehat{\Omega} \\ &= \cos(\frac{\theta}{2}) e + \frac{\sin(\frac{\theta}{2})}{\frac{\theta}{2}} \widehat{\Omega} \end{aligned} \tag{11.176}$$ {eq:141}

that is

$$\exp q(\widehat{\Omega}) = (\cos(\frac{\theta}{2}), \sin(\frac{\theta}{2}) \frac{\Omega}{\theta}). \tag{11.177}$$ {eq:144}

Adjoint representation In the case of $I\!H_1$, the definition of the operator Ad gives

$$\mathrm{Ad}_p(\widehat{\Omega}) = p \cdot \widehat{\Omega} p^\star \tag{11.178}$$ {eq:121}

and then mapping $\mathrm{ad}_{\widehat{\Omega}}(\widehat{\Gamma})$ is defined by

$$\mathrm{ad}_{\widehat{\Omega}}(\widehat{\Gamma}) = \widehat{\Omega}\widehat{\Gamma} - \widehat{\Gamma}\widehat{\Omega} = [\widehat{\Omega}, \widehat{\Gamma}] = \widehat{\Omega \times \Gamma}. \tag{11.179}$$ {eq:56}

Using the isomorphism between $\mathfrak{h}_1$ and $I\!R^3$, we can use the the mapping $\mathrm{ad}_\Omega(\Gamma) : I\!R^3 \times I\!R^3 \to I\!R^3$ given by (11.107) to get

$$\mathrm{ad}_{\widehat{\Omega}}(\widehat{\Gamma}) = \widehat{\Omega \times \Gamma} = \widehat{\mathrm{ad}_\Omega(\Gamma)} = \widehat{\Omega\Gamma} \tag{11.180}$$ {eq:145}

Differential of the exponential map dexpq The differential of the exponential mapping, denoted by dexpq is defined as the 'right trivialized' tangent of the exponential map

$$\frac{d}{dt}(\text{expq}(\widehat{\Omega}(t))) = \text{dexpq}_{\widehat{\Omega}(t)}(\frac{d\widehat{\Omega}(t)}{dt})\,\text{expq}(\widehat{\Omega}(t)) \tag{11.181}$$ {eq:71}

An explicit expression of $\text{dexp}_{\widehat{\Omega}}(\widehat{\Gamma})$ can also be developed either by developing the expansion and (11.165).

$$\text{dexpq}_{\widehat{\Omega}}(\Gamma) = \sum_{k=0}^{\infty} \frac{1}{(k+1)!}\,\text{ad}^k_{\widehat{\Omega}}(\widehat{\Gamma}) = \widehat{T(\Omega)\Gamma} \tag{11.182}$$ {eq:168}

Remark 5 *Note that the time derivative in $\mathbb{R}^4$ is not differential mapping. The standard time derivative of* expq *in the expression* (11.177) *gives*

$$\frac{d}{dt}\,\text{expq}(\widehat{\Gamma}(t)) = (-\frac{\sin(\theta)}{\theta}\Omega^T\Gamma, \frac{\sin(\theta)}{\theta}\Gamma + \frac{\theta\cos(\theta)-\sin(\theta)}{\theta^3}\Omega^T\Omega\Gamma) \tag{11.183}$$ {eq:171}

that can be expressed in $\mathbb{R}^4$ by

$$\frac{d}{dt}\,\text{expq}(\widehat{\Gamma}(t)) = \nabla\,\text{expq}(\widehat{\Omega})\dot{\widehat{\Omega}} \tag{11.184}$$ {eq:175}

with

$$\nabla\,\text{expq}(\widehat{\Omega}) = \begin{bmatrix} -\frac{\sin(\theta)}{\theta}\Omega^T \\ \frac{\sin(\theta)}{\theta}I + \frac{\theta\cos(\theta)-\sin(\theta)}{\theta^3}\Omega^T\Omega \end{bmatrix} \tag{11.185}$$ {eq:176}

Clearly, we have

$$\nabla\,\text{expq}(\widehat{\Omega}) \neq \text{dexpq}_{\widehat{\Omega}} \tag{11.186}$$ {eq:177}

Directional derivative and Jacobians of functions of a quaternion

11.4.1 Redaction note V. ACARY
experimental

Let $f : \mathbb{H}_1 \to \mathbb{R}$ be a mapping from the group to $\mathbb{R}^3$. The directional derivative of f in the direction $\widehat{\Omega} \in \mathfrak{h}_1$ at $p \in \mathbb{H}_1$ is defined by

$$df_p(\widehat{\Omega}) = \frac{d}{dt} f(p \cdot \text{expq}(t\widehat{\Omega}))\Big|_{t=0} \tag{11.187}$$ {eq:139}

As a first simple example let us choose $f(p) = \overrightarrow{p \cdot p_x \cdot p^\star}$ for a given $x \in \mathbb{R}^3$, we get

$$\begin{aligned} DId \cdot \widehat{\Omega}(p) = (\widehat{\Omega}^r f)(p) &= \frac{d}{dt}\overrightarrow{p \cdot \text{expq}(t\widehat{\Omega}) \cdot p_x \cdot (p \cdot \text{expq}(t\widehat{\Omega}))^\star}\Big|_{t=0} \\ &= \overrightarrow{p \cdot \frac{d}{dt}\,\text{expq}(t\widehat{\Omega})\Big|_{t=0} \cdot p_x \cdot p^\star + p \cdot p_x \cdot (p \cdot \frac{d}{dt}\,\text{expq}(t\widehat{\Omega})\Big|_{t=0})^\star} \end{aligned} \tag{11.188}$$ {eq:142}

We have form the definition of the time derivative of the exponential

$$\begin{aligned} \frac{d}{dt}\,\text{expq}(t\widehat{\Omega})\Big|_{t=0} &= \text{dexpq}_{\widehat{\Omega}}(\widehat{\Omega})\,\text{expq}(t\widehat{\Omega})\Big|_{t=0} \\ &= \text{dexpq}_{\widehat{\Omega}}(\widehat{\Omega}) \end{aligned} \tag{11.189}$$ {eq:143}

Then, the directional derivative can be written

$$\begin{aligned} DId \cdot \widehat{\Omega}(p) &= \overrightarrow{p \cdot \text{dexpq}_{\widehat{\Omega}}(\widehat{\Omega}) \cdot p_x \cdot p^\star + p \cdot p_x \cdot (\text{dexpq}_{\widehat{\Omega}}(\widehat{\Omega}))^* \cdot p^\star} \\ &= \overrightarrow{p \cdot (\text{dexpq}_{\widehat{\Omega}}(\widehat{\Omega}) \cdot p_x + p_x \cdot (\text{dexpq}_{\widehat{\Omega}}(\widehat{\Omega}))^*) \cdot p^\star} \end{aligned} \tag{11.190}$$ {eq:146}

11.5 Newton-Euler equation in quaternion form

Computation of T for unit quaternion The operator $T(q)$ is directly obtained as

$$T(q) = \frac{1}{2}\begin{bmatrix} 2I_{3\times 3} & & 0_{3\times 3} & \\ & -p_1 & -p_2 & -p_3 \\ 0_{4\times 3} & p_0 & -p_3 & p_2 \\ & p_3 & p_0 & -p_1 \\ & -p_2 & p_1 & p_0 \end{bmatrix} \tag{11.191}$$ {eq:98}

11.5.1 Redaction note V. ACARY

todo :

- computation of the directional derivative of $R(\Omega) = exp(\tilde{\Omega})$ in the direction $\tilde{\Omega}$, to get $T(\Omega)$

Quaternion representation If the Lie group is described by unit quaternion, we get

$$SO(3) = \{p = (p_0, \vec{p}) \in \mathbb{R}^4 \mid |p| = 1\} \tag{11.192}$$ {eq:99}

with the composition law $p_1 \cdot p_2$ given by the quaternion product.

Note that the concept of exponential map for Lie group that are not parameterized by matrices is also possible.

11.5.1 Mechanical systems with bilateral and unilateral constraints

{section22}

Let us consider that the system (11.20) is subjected to m constraints, with m_e holonomic bilateral constraints

$$h^\alpha(q) = 0, \alpha \in \mathcal{E} \subset \mathbb{N}, |\mathcal{E}| = m_e, \tag{11.193}$$ {eq:bilateral-co

and m_i unilateral constraints

$$g_{\mathrm{N}}^\alpha(q) \geq 0, \alpha \in \mathcal{I} \subset \mathbb{N}, |\mathcal{I}| = m_i. \tag{11.194}$$ {eq:unilateral-c

Let us denote as $J_h^\alpha(q) = \nabla_q^\top h^\alpha(q)$ the Jacobian matrix of the bilateral constraint $h^\alpha(q)$ with respect to q and as $J_{g_{\mathrm{N}}}^\alpha(q)$ respectively for $g_{\mathrm{N}}^\alpha(q)$. The bilateral constraints at the velocity level can be obtained as:

$$0 = \dot{h}^\alpha(q) = J_h^\alpha(q)\dot{q} = J_h^\alpha(q)T(q)v \coloneqq H^\alpha(q)v, \quad \alpha \in \mathcal{E}. \tag{11.195}$$ {eq:bilateral-co

By duality and introducing a Lagrange multiplier $\lambda^\alpha, \alpha \in \mathcal{E}$, the constraint generates a force applied to the body equal to $H^{\alpha,\top}(q)\lambda^\alpha$. For the unilateral constraints, a Lagrange multiplier $\lambda_{\mathrm{N}}^\alpha, \alpha \in \mathcal{I}$ is also associated and the constraints at the velocity level can also be derived as

$$0 \leq \dot{g}_{\mathrm{N}}^\alpha(q) = J_{g_{\mathrm{N}}}^\alpha(q)\dot{q} = J_{g_{\mathrm{N}}}^\alpha(q)T(q)v, \text{ if } g_{\mathrm{N}}^\alpha(q) = 0, \quad \alpha \in \mathcal{I}. \tag{11.196}$$ {eq:unilateral-c

Again, the force applied to the body is given by $(J_{g_{\mathrm{N}}}^\alpha(q)T(q))^\top \lambda_{\mathrm{N}}^\alpha$. Nevertheless, there is no reason that $\lambda_{\mathrm{N}}^\alpha = r_{\mathrm{N}}^\alpha$ and $u_{\mathrm{N}} = J_{g_{\mathrm{N}}}^\alpha(q)T(q)v$ if the g_n is not chosen as the signed distance (the gap function). This is the reason why we prefer directly define the normal and the tangential local relative velocity with respect to the twist vector as

$$u_{\mathrm{N}}^\alpha \coloneqq G_{\mathrm{N}}^\alpha(q)v, \quad u_{\mathrm{T}}^\alpha \coloneqq G_{\mathrm{T}}^\alpha(q)v, \quad \alpha \in \mathcal{I}, \tag{11.197}$$ {eq:unilateral-c

and the associated force as $G_{\mathrm{N}}^{\alpha,\top}(q)r_{\mathrm{N}}^\alpha$ and $G_{\mathrm{T}}^{\alpha,\top}(q)r_{\mathrm{T}}^\alpha$. For the sake of simplicity, we use the notation $u^\alpha \coloneqq G^\alpha(q)v$ and its associated total force generated by the contact α as $G^{\alpha,\top}(q)r^\alpha \coloneqq G_{\mathrm{N}}^{\alpha,\top}(q)r_{\mathrm{N}}^\alpha + G_{\mathrm{T}}^{\alpha,\top}(q)r_{\mathrm{T}}^\alpha$.

The complete system of equation of motion can finally be written as

$$\left\{\begin{array}{ll} \dot{q} = T(q)v, & \\ M\dot{v} = F(t,q,v) + H^\top(q)\lambda + G^\top(q)r, & \\ H^\alpha(q)v = 0, & \alpha \in \mathcal{E} \\ \left.\begin{array}{ll} r^\alpha = 0, & \text{if } g_{\mathrm{N}}^\alpha(q) > 0, \\ K^{\alpha,*} \ni \widehat{u}^\alpha \perp r^\alpha \in K^\alpha, & \text{if } g_{\mathrm{N}}^\alpha(q) = 0, \\ u_{\mathrm{N}}^{\alpha,+} = -e_r^\alpha u_{\mathrm{N}}^{\alpha,-}, & \text{if } g_{\mathrm{N}}^\alpha(q) = 0 \text{ and } u_{\mathrm{N}}^{\alpha,-} \leq 0, \end{array}\right\} & \alpha \in \mathcal{I} \end{array}\right. \tag{11.198}$$

{eq:NewtonEuler-u

where the definition of the variables $\lambda \in \mathbb{R}^{m_e}, r \in \mathbb{R}^{3m_i}$ and the operators H, G are extended to collect all the variables for each constraints.

Note that all the constraints are written at the velocity integrators. Another strong advantage is the straightforward introduction of the contact dissipation processes that are naturally written at the velocity level such as the Newton impact law and the Coulomb friction. Indeed, in Mechanics, dissipation processes are always given in terms of rates of changes, or if we prefer, in terms of velocities.

Siconos Notation In the siconos notation, we have for the applied torques on the system the following decomposition

$$F(t,q,v) := \begin{pmatrix} f(t, x_{\mathrm{g}}, v_{\mathrm{g}}, R, \Omega) \\ I\Omega \times \Omega + M(t, x_{\mathrm{g}}, v_{\mathrm{g}}, R, \Omega) \end{pmatrix} := \begin{pmatrix} f_{ext}(t) - f_{int}(x_{\mathrm{g}}, v_{\mathrm{g}}, R, \Omega) \\ -M_{gyr}(\Omega) + M_{ext}(t) - M_{int}(x_{\mathrm{g}}, v_{\mathrm{g}}, R, \Omega) \end{pmatrix}. \tag{11.199}$$

with

$$M_{gyr} := \left(\Omega \times I\Omega\right) \tag{11.200}$$

In the siconos notation, we have for the relation

$$C = J^\alpha(q) \quad CT = J^\alpha(q)T(q) \tag{11.201}$$

{eq:100}

11.6 Time integration scheme in scheme

11.6.1 Moreau–Jean scheme based on a θ-method

The complete Moreau–Jean scheme based on a θ-method is written as follows

$$\left\{\begin{array}{ll} q_{k+1} = q_k + hT(q_{k+\theta})v_{k+\theta} & \\ M(v_{k+1} - v_k) - hF(t_{k+\theta}, q_{k+\theta}, v_{k+\theta}) = H^\top(q_{k+1})Q_{k+1} + G^\top(q_{k+1})P_{k+1}, & \\ \left.\begin{array}{ll} H^\alpha(q_{k+1})v_{k+1} = 0 & \end{array}\right\} & \alpha \in \mathcal{E} \\ \left.\begin{array}{ll} P_{k+1}^\alpha = 0, & \end{array}\right\} & \alpha \notin \mathcal{I}^\nu \\ \left.\begin{array}{ll} K^{\alpha,*} \ni \widehat{u}_{k+1}^\alpha \perp P_{k+1}^\alpha \in K^\alpha & \end{array}\right\} & \alpha \in \mathcal{I}^\nu \end{array}\right. \tag{11.202}$$

{eq:Moreau-Jean-t

where $\mathcal{I}^\nu$ is the set of forecast constraints, that may be evaluated as

$$\mathcal{I}^\nu = \{\alpha \mid \bar{g}_{\mathrm{N}}^\alpha := g_{\mathrm{N}} + \frac{h}{2}u_{\mathrm{N}}^\alpha \leq 0\}. \tag{11.203}$$

{eq:101}

11.6.2 Semi-explicit version Moreau–Jean scheme based on a θ-method

$$\left\{\begin{array}{ll} q_{k+1} = q_k + hT(q_k)v_{k+\theta} & \\ M(v_{k+1}-v_k) - hF(t_k,q_k,v_k) = H^\top(q_k)Q_{k+1} + G^\top(q_k)P_{k+1}, & \\ H^\alpha(q_{k+1})v_{k+1} = 0 & \left.\right\} \quad \alpha \in \mathcal{E} \\ P^\alpha_{k+1} = 0, & \left.\right\} \quad \alpha \notin \mathcal{I}^\nu \\ K^{\alpha,*} \ni \widehat{u}^\alpha_{k+1} \perp P^\alpha_{k+1} \in K^\alpha & \left.\right\} \quad \alpha \in \mathcal{I}^\nu \end{array}\right. \tag{11.204}$$ {eq:Moreau-Jean-

In this version, the new velocity v_{k+1} can be computed explicitly, assuming that the inverse of M is easily written, as

$$v_{k+1} = v_k + M^{-1}hF(t_k,q_k,v_k) + M^{-1}(H^\top(q_k)Q_{k+1} + G^\top(q_k)P_{k+1}) \tag{11.205}$$ {eq:Moreau-Jean-

11.6.3 Nearly implicit version Moreau–Jean scheme based on a θ-method implemented in siconos

A first simplification is made considering a given value of q_{k+1} in $T()$, $H()$ and $G()$ denoted by $\bar{q}_k$. This limits the computation of the Jacobians of this operators with respect to q.

$$\left\{\begin{array}{ll} q_{k+1} = q_k + hT(\bar{q}_k)v_{k+\theta} & \\ M(v_{k+1}-v_k) - h\theta F(t_{k+1},q_{k+1},v_{k+1}) - h(1-\theta)F(t_k,q_k,v_k) = H^\top(\bar{q}_k)Q_{k+1} + G^\top(\bar{q}_k)P_{k+1}, & \\ H^\alpha(\bar{q}_k)v_{k+1} = 0 & \left.\right\} \quad \alpha \in \mathcal{E} \\ P^\alpha_{k+1} = 0, & \left.\right\} \quad \alpha \notin \mathcal{I}^\nu \\ K^{\alpha,*} \ni \widehat{u}^\alpha_{k+1} \perp P^\alpha_{k+1} \in K^\alpha & \left.\right\} \quad \alpha \in \mathcal{I}^\nu \end{array}\right. \tag{11.206}$$ {eq:Moreau-Jean-

The nonlinear residu is defined as

$$\mathcal{R}(v) = M(v-v_k) - h\theta F(t_{k+1},q(v),v) - h(1-\theta)F(t_k,q_k,v_k) - H^\top(\bar{q}_k)Q_{k+1} - G^\top(\bar{q}_k)P_{k+1} \tag{11.207}$$ {eq:Moreau-Jean-

with

$$q(v) = q_k + hT(\bar{q}_k))((1-\theta)v_k + \theta v). \tag{11.208}$$ {eq:Moreau-Jean-

At each time step, we have to solve

$$\mathcal{R}(v_{k+1}) = 0 \tag{11.209}$$ {eq:Moreau-Jean-

together with the constraints.

Let us write a linearization of the problem to design a Newton procedure:

$$\nabla_v^\top \mathcal{R}(v^\tau_{k+1})(v^{\tau+1}_{k+1} - v^\tau_{k+1}) = -\mathcal{R}(v^\tau_{k+1}). \tag{11.210}$$ {eq:Moreau-Jean-

The computation of $\nabla_v^\top \mathcal{R}(v^\tau_{k+1})$ is as follows

$$\nabla_v^\top \mathcal{R}(v) = M - h\theta \nabla_v F(t_{k+1},q(v),v) \tag{11.211}$$ {eq:102}

with

$$\begin{array}{rcl} \nabla_v F(t_{k+1},q(v),v) & = & D_2F(t_{k+1},q(v),v)\nabla_v q(v) + D_3F(t_{k+1},q(v),v) \\ & = & h\theta D_2F(t_{k+1},q(v),v)T(\bar{q}_k) + D_3F(t_{k+1},q(v),v) \end{array} \tag{11.212}$$ {eq:103}

where D_i denotes the derivation with respect the i^{th} variable. The complete Jacobian is then given by

$$\nabla_v^\top \mathcal{R}(v) = M - h\theta D_3F(t_{k+1},q(v),v) - h^2\theta^2 D_2F(t_{k+1},q(v),v)T(\bar{q}_k) \tag{11.213}$$ {eq:104}

In siconos, we ask the user to provide the functions $D_3F(t_{k+1}, q, v)$ and $D_2F(t_{k+1}, q, v)$.

Let us denote by W^τ the inverse of Jacobian of the residu,

$$W^\tau = (M - h\theta D_3F(t_{k+1}, q(v), v) - h^2\theta^2 D_2F(t_{k+1}, q(v), v)T(\bar{q}_k))^{-1}. \tag{11.214}$$ {eq:105}

and by $\mathcal{R}_{free}(v)$ the free residu,

$$\mathcal{R}_{free}(v) = M(v - v_k) - h\theta F(t_{k+1}, q(v), v) - h(1-\theta)F(t_k, q_k, v_k). \tag{11.215}$$ {eq:106}

The linear equation 11.210 that we have to solve is equivalent to

$$\boxed{v^{\tau+1}_{k+1} = v^\tau_{k+1} - W\mathcal{R}_{free}(v^\tau_{k+1}) + WH^\top(\bar{q}_k)Q^{\tau+1}_{k+1} + WG^\top(\bar{q}_k)P^{\tau+1}_{k+1}} \tag{11.216}$$ {eq:107}

We define v_{free} as

$$v_{free} = v^\tau_{k+1} - W\mathcal{R}_{free}(v^\tau_{k+1}) \tag{11.217}$$ {eq:108}

The local velocity at contact can be written

$$u^{\tau+1}_{\mathrm{N},k+1} = G(\bar{q}_k)[v^\tau_{free} + WH^\top(\bar{q}_k)Q^{\tau+1}_{k+1} + WG^\top(\bar{q}_k)P^{\tau+1}_{k+1}] \tag{11.218}$$ {eq:109}

and for the equality constraints

$$u^{\tau+1}_{k+1} = H(\bar{q}_k)[v^\tau_{free} + WH^\top(\bar{q}_k)Q^{\tau+1}_{k+1} + WG^\top(\bar{q}_k)P^{\tau+1}_{k+1}] \tag{11.219}$$ {eq:110}

Finally, we get a linear relation between $u^{\tau+1}_{\mathrm{N},k+1}$ and the multiplier

$$\boxed{u^{\tau+1}_{k+1} = \begin{bmatrix} H(\bar{q}_k) \\ G(\bar{q}_k) \end{bmatrix} v^\tau_{free} + \begin{bmatrix} H(\bar{q}_k)WH^\top(\bar{q}_k) & H(\bar{q}_k)WG^\top(\bar{q}_k) \\ G(\bar{q}_k)WH^\top(\bar{q}_k) & G(\bar{q}_k)WG^\top(\bar{q}_k) \end{bmatrix} \begin{bmatrix} Q^{\tau+1}_{k+1} \\ P^{\tau+1}_{k+1} \end{bmatrix}} \tag{11.220}$$ {eq:111}

choices for $\bar{q}_k$ Two choices are possible for $\bar{q}_k$

1. $\bar{q}_k = q_k$
2. $\bar{q}_k = q^\tau_{k+1}$

11.6.1 Redaction note V. ACARY

todo list:

- add the projection step for the unit quaternion
- describe the computation of H and G that can be hybrid

11.6.4 Computation of the Jacobian in special case

Moment of gyroscopic forces Let us denote by the basis vector e_i given the i^{th} column of the identity matrix $I_{3\times 3}$. The Jacobian of M_{gyr} is given by

$$\nabla^\top_\Omega M_{gyr}(\Omega) = \nabla^\top_\Omega(\Omega \times I\Omega) = \left[e_i \times I\Omega + \Omega \times Ie_i, i = 1, 2, 3\right] \tag{11.221}$$ {eq:112}

Linear internal wrench If the internal wrench is given by

$$F_{int}(t,q,v) = \begin{bmatrix} f_{int}(t,q,v) \\ M_{int}(t,q,v) \end{bmatrix} = Cv + Kq, \quad C \in I\!R^{6\times 6}, \quad K \in I\!R^{6\times 7} \tag{11.222}$$ {eq:113}

we get

$$\begin{array}{lcl} \nabla_v F(t_{k+1}, q(v), v) & = & h\theta KT(\bar{q}_k) + C \\ \nabla_v^\top \mathcal{R}(v) & = & M - h\theta C - h^2\theta^2 KT(\bar{q}_k) \end{array} \tag{11.223}$$ {eq:114}

External moment given in the inertial frame If the external moment denoted by $m_{ext}(t)$ is expressed in inertial frame, we have

$$M_{ext}(q,t) = R^T m_{ext}(t) = \Phi(p) m_{ext}(t) \tag{11.224}$$ {eq:115}

In that case, $M_{ext}(q,t)$ appears as a function q and we need to compute its Jacobian w.r.t q. This computation needs the computation of

$$\nabla_p M_{ext}(q,t) = \nabla_p \Phi(p) m_{ext}(t) \tag{11.225}$$ {eq:116}

Let us compute first

$$\Phi(p) m_{ext}(t) = \begin{bmatrix} (1-2p_2^2-2p_3^2)m_{ext,1} + 2(p_1p_2 - p_3p_0)m_{ext,2} + 2(p_1p_3 + p_2p_0)m_{ext,3} \\ 2(p_1p_2 + p_3p_0)m_{ext,1} + (1-2p_1^2-2p_3^2)m_{ext,2} + 2(p_2p_3 - p_1p_0)m_{ext,3} \\ 2(p_1p_3 - p_2p_0)m_{ext,1} + 2(p_2p_3 + p_1p_0)m_{ext,2} + (1-2p_1^2-2p_2^2)m_{ext,3} \end{bmatrix} \tag{11.226}$$ {eq:117}

Then we get

$$\nabla_p \Phi(p) m_{ext}(t) = \begin{bmatrix} -2p_3 m_{ext,2} + 2p_2 m_{ext,3} & 2p_2 m_{ext,2} + 2p_3 m_{ext,3} & -4p_2 m_{ext,1} + 2p_1 m_{ext,2} + 2p_0 m_{ext,3} & -3p_3 m_{ext,1} - 2p_0 m_{ext,2} \\ 2p_3 m_{ext,1} - 2p_1 m_{ext,3} & 2p_2 m_{ext,1} - 4p_1 m_{ext,2} - 2p_1 m_{ext,3} & & \end{bmatrix} \tag{11.227}$$ {eq:118}

11.6.5 Siconos implementation

The expression: $\mathcal{R}_{free}(v^\tau_{k+1}) = M(v - v_k) - h\theta F(t_{k+1}, q(v^\tau_{k+1}), v^\tau_{k+1}) - h(1-\theta)F(t_k, q_k, v_k)$ is computed in `MoreauJeanOSI::computeResidu()` and saved in `ds->workspace(DynamicalSystem::freeresidu)`

The expression: $\mathcal{R}(v^\tau_{k+1}) = \mathcal{R}_{free}(v^\tau_{k+1}) - h(1-\theta)F(t_k, q_k, v_k) - H^\top(\bar{q}_k)Q_{k+1} - G^\top(\bar{q}_k)P_{k+1}$ is computed in `MoreauJeanOSI::computeResidu()` and saved in `ds->workspace(DynamicalSystem::free)`.

11.6.2 Redaction note V. ACARY

really a bad name for the buffer ds->workspace(DynamicalSystem::free). Why we are chosing this name ? to save some memory ?

The expression: $v_{free} = v^\tau_{k+1} - W\mathcal{R}_{free}(v^\tau_{k+1})$ is compute in `MoreauJeanOSI::computeFreeState()` and saved in `d->workspace(DynamicalSystem::free)`.

The computation: $v^{\tau+1}_{k+1} = v_{free} + WH^\top(\bar{q}_k)Q^{\tau+1}_{k+1} + WG^\top(\bar{q}_k)P^{\tau+1}_{k+1}$ is done in `MoreauJeanOSI::updateState` and stored in `d->twist()`.

Chapter 12

NewtonEulerR: computation of $\nabla_q H$

12.0.1 Gradient computation, case of NewtonEuler with quaternion

In the section, q is the quaternion of the dynamical system.

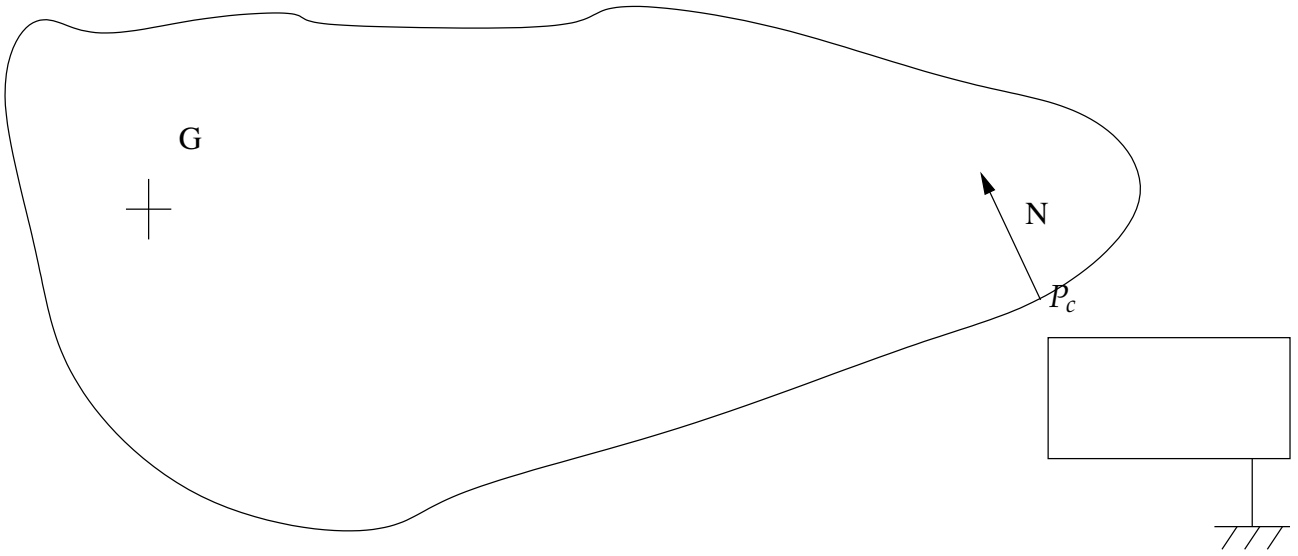


Figure 12.1: Impact of one DS.

{figCase}

The normal vector N is view as a constant.

$$\tilde{h}(q) = P_c(\frac{q}{\|q\|})$$

$$^t\nabla h(q)(\delta q) = \lim_{e\to 0} \frac{(\tilde{h}(q+e\delta q) - \tilde{h}(q)).N}{e}$$

$\nabla_q h$ consist in computing $P_c(\frac{q+\delta q}{\|q+\delta q\|}) - P_c(q)$.

$$GP(q) = qG_0P_0\ ^c q$$

$$GP(\frac{q+\delta q}{\|q+\delta q\|}) = (q+\delta q)G_0P_0\ ^c(q+\delta q)\frac{1}{\|q+\delta q\|^2}$$

$$= (q+\delta q)\ ^c qGP(q)q\ ^c(q+\delta q)\frac{1}{\|q+\delta q\|^2}$$

$$= ((1,0,0,0) + \delta q\ {}^c q) GP(q) ((1,0,0,0) + q\ {}^c \delta q) \frac{1}{\|q+\delta q\|^2}$$

$$= GP(q) + \delta q\ {}^c q GP(q) + GP(q) q\ {}^c \delta q + 0(\delta q)^2 \frac{1}{\|q+\delta q\|^2}$$

So, because G is independant of q:

$$P(\frac{q+\delta q}{\|q+\delta q\|}) - P(q) = qGP(\frac{q+\delta q}{\|q+\delta q\|}) - GP(q) = \delta q\ {}^c q GP(q) + GP(q) q\ {}^c \delta q + 0(\delta q)^2 + GP(q) \frac{1}{\|q+\delta q\|^2}$$

For the directional derivation, we chose $\delta q = \epsilon * (1,0,0,0)$. using a equivalent to $\frac{1}{1+\epsilon}$

$$\lim_{\epsilon \to 0} \frac{P(\frac{q+\delta q}{\|q+\delta q\|}) - P(q)}{\epsilon} = {}^c q GP(q) + GP(q) q - 2 q_i GP(q)$$

For the directional derivation, we chose $\delta q = \epsilon * (0,1,0,0) = \epsilon * e_i$

$$\lim_{\epsilon \to 0} \frac{P(\frac{q+\delta q}{\|q+\delta q\|}) - P(q)}{\epsilon} = e_i\ {}^c q GP(q) - GP(q) q e_i - 2 q_i GP(q)$$

Application to the NewtonEulerRImpact:

$$H : \mathbb{R}^7 \to \mathbb{R}$$

$$\nabla_q H \in \mathcal{M}^{1,7}$$

$$\nabla_q H = \begin{pmatrix} N_x \\ N_y \\ N_z \\ ({}^c q GP(q) + GP(q) q - 2 q_0 GP(q)).N \\ (e_2\ {}^c q GP(q) - GP(q) q e_2 - 2 q_1 GP(q)).N \\ (e_3\ {}^c q GP(q) - GP(q) q e_3 - 2 q_2 GP(q)).N \\ (e_4\ {}^c q GP(q) - GP(q) q e_4 - 2 q_3 GP(q)).N \end{pmatrix}$$

12.0.2 Ball case

It is the case where $GP = -N$: for $e2$:

$$(0,1,0,0).(q_0, -\underline{p}).(0,-N) =$$

$$\left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} .\underline{p}, \begin{pmatrix} q_0 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} * \underline{p} \right) .(0,-N) =$$

$$\left(?, -\underline{p}_x\ N - \left(\begin{pmatrix} q_0 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} * \underline{p} \right) * N \right) =$$

and:

$$(0,-N).(q_0, \underline{p}).(0,1,0,0) =$$

$$(N.\underline{p}, -q_0 N - N * \underline{p}).(0,1,0,0) =$$

$$\left(?, (N.\underline{p}) \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} * (q_0 N + N * \underline{p}) \right) =$$

$$\left(?, (N.\underline{p}) \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + q_0 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} * N + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} * (N * \underline{p})\right)$$

sub then and get the resulting vector.N:

$$\left[-\underline{p}_x\ N - N.\underline{p} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + () * N - \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} * (N * \underline{p})\right].N =$$

$$-\underline{p}_x - N_x N.\underline{p} + 0 - (\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} * (N * \underline{p})).N =$$

using $a * (b * c) = b(a.c) - c(a.b)$ leads to

$$-q_1 - N_x N.\underline{p} - (q_1\ N - N_x\ \underline{p}).N =$$
$$-q_1 - N_x N.\underline{p} - q_1 + N_x N.\underline{p} = -2q_1$$

for $e1 = (1, 0, 0, 0)$:

$$(q_0, -\underline{p}).(0, -N) = (?, -q_0 N + \underline{p} * N)$$
$$(0, -N).(q_0, \underline{p}) = (?, -q_0 N - \underline{p} * N)$$

So

$$\nabla_q H = \begin{pmatrix} N_x \\ N_y \\ N_z \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

12.0.3 Case FC3D: using the local frame and momentum

$$\begin{pmatrix} m\dot{V} \\ I\dot{\Omega} + \Omega I\Omega \end{pmatrix} = \begin{pmatrix} Fect + R \\ Mext_{R_{obj}} + (R * PG)_{R_{obj}} \end{pmatrix}$$

with * vectoriel product, R reaction in the globla frame. P the point of contact. r is the reaction in the local frame. $M_{R_{obj}toR_{abs}} = M^t_{R_{abs}toR_{obj}} r = R$ with:

$$M_{R_C toR_{abs}} = \begin{pmatrix} nx & t_1 x & t_2 x \\ ny & t_1 y & t_2 y \\ nz & t_1 z & t_2 z \end{pmatrix}$$

we have :

$$\begin{pmatrix} R \\ (R * PG)_{R_{obj}} \end{pmatrix} = \begin{pmatrix} I_3 \\ M_{R_{abs}toR_{obj}} N_{PG} \end{pmatrix}.R$$
$$= \begin{pmatrix} I_3 \\ M_{R_{abs}toR_{obj}} N_{PG} \end{pmatrix}.M_{R_{obj}toR_{abs}} r$$
$$N_{PG} = \begin{pmatrix} 0 & PG_z & -PG_y \\ -PG_z & 0 & PG_x \\ PG_y & -PG_X & 0 \end{pmatrix}$$

that is:

$$\begin{pmatrix} m\dot{V} \\ I\dot{\Omega} + \Omega I\Omega \end{pmatrix} = \begin{pmatrix} M_{R_C toR_{abs}} \\ M_{R_{abs}toR_{obj}} N_{PG} M_{R_C toR_{abs}} \end{pmatrix} r$$

So $jachqt = MN$

12.0.4 Case FC3D: using the local frame local velocities

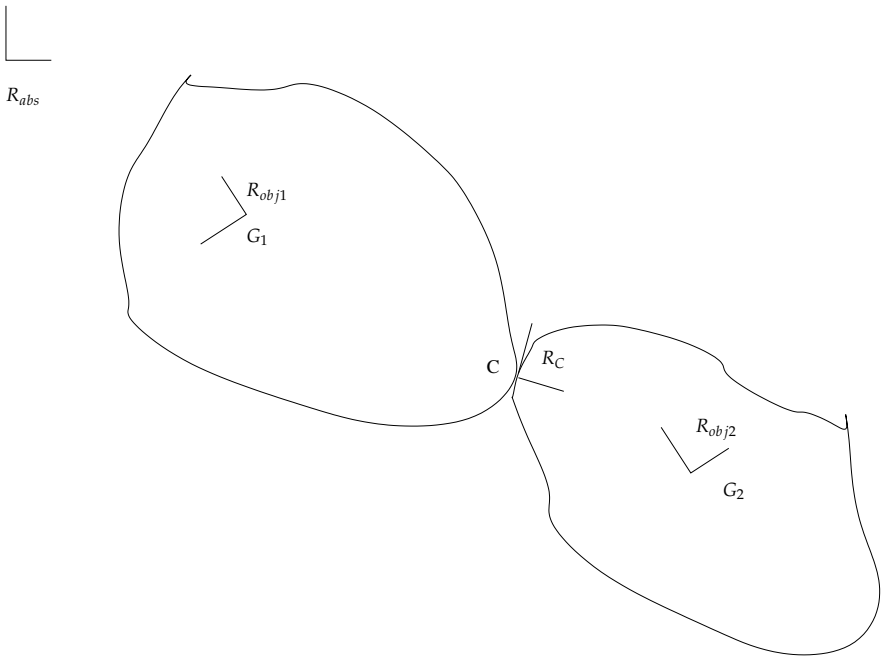


Figure 12.2: Two objects colliding. {figCase}

We are looking for an operator named CT such that:

$$V_C = \begin{pmatrix} V_N \\ V_T \\ V_S \end{pmatrix}_{R_C} = CT \begin{pmatrix} V_{G1\ R_{abs}} \\ \Omega_{1\ R_{obj1}} \\ V_{G2\ R_{abs}} \\ \Omega_{2\ R_{obj2}} \end{pmatrix}$$

$$V_c = V_{G1\ R_{abs}} + w_1 * G_1P_{\ R_{abs}} - (V_{G2\ R_{abs}} + w_2 * G_1P_{\ R_{abs}})$$

where w_1 and w_2 are given in R_{abs}. We note $M_{R_{obj1}toR_{abs}}$ the matrice converting the object 1 coordinate to the absolute coordinate. We note N_{GP} the matrice such that $w_1 * G_1P_{\ R_{abs}} = N_{GC}w_1$. Endly, we note $M_{R_{abs}toR_C}$ converting the absolute coordinate to the R_C frame. we get:

$$CT = M_{R_{abs}toR_C} \begin{pmatrix} I_3 & N_{G_1C}M_{R_{obj1}toR_{abs}} & -I_3 & -N_{G_2C}M_{R_{obj2}toR_{abs}} \end{pmatrix}$$

12.0.4.a Expression of $M_{R_{obj1}toR_{abs}}$

Using quaternion, we get :

$$M_{R_{obj1}toR_{abs}} = \begin{pmatrix} q\begin{pmatrix}1\\0\\0\end{pmatrix}{}^cq & q\begin{pmatrix}0\\1\\0\end{pmatrix}{}^cq & q\begin{pmatrix}0\\0\\1\end{pmatrix}{}^cq \end{pmatrix} \quad (12.1)$$ {eq:newton_Mobjt

12.0.4.b Expression of N_1

$$N_{GC} = \begin{pmatrix} 0 & G_1C_z & -G_1C_y \\ -G_1C_z & 0 & G_1C_x \\ G_1C_y & -G_1C_X & 0 \end{pmatrix}$$

Chapter 13

Projection On constraints

13.0.1 Velocity formulation

The first step consists in doing a velocity formulation of the system:

$$\begin{array}{l} M\dot{v} = F_{ext} + B\lambda \\ \dot{q} = Tv \\ y = h(q) \\ NSLAW(y, \lambda, ...) \end{array} \tag{13.1}$$ {NE_Dyn1}

The constraint $\dot{q} = Tv$ is suffisiant to keep a normal quaternion. Because of the speed formulation, $h(q)$ could violate the NSLAW. A solution coul be to add a formulation in position. We must underline that the constraints $\mid Q \mid = 1$ is implicit in this system. Endeed, the direction $\dot{q} = Tv$ is tangential to the sphere.

13.0.2 Posion formulation

It consists in writting a position formulation on the system:

$$h(q) = \begin{pmatrix} HI(q) \\ HE(q) \end{pmatrix} \tag{13.2}$$ {NE_Dyn1}

13.0.2.a Approach using q

We are looking for q_1 from q_0:

$$q_1 = q_0 + \nabla HI\Lambda_I + \nabla HE\Lambda_E \tag{13.3}$$

Assume that $h(q_0)$ doesn't satisfy the constraints, ie $HI(q_0) \ngeq 0$ or $HE(q_0) \neq 0$). Linearize h leads to:

$$0 \leq HI(q_0) + \nabla^t HI(\nabla HI\Lambda_I + \nabla HE\Lambda_E) \perp \Lambda_I \geq 0 \tag{13.4}$$

$$0 = HE(q_0) + \nabla^t HE(\nabla HI\Lambda_I + \nabla HE\Lambda_E) \tag{13.5}$$

The getting system could be written has a MLCP:

$$C \ni h(q_0) + \nabla^t h(\nabla h\Lambda), \Lambda \in C^* \tag{13.6}$$

In the case of a quaternion Q for the rotation representation, it is noteworthy that this system doesn't deal with the constraints $\mid Q \mid = 1$. Thus, the direction (q_1, q_0) can be normal to this constraint, in that case this approach doesn't work. (It happens in practice) The solution that consists in normaliaed q after this formulation is not convenient because, it could be incompatible with $\mid Q \mid = 1$. A better approach is to add this constraint.

The constraint $\mid Q \mid = 1$ in the system HE:

$$\tilde{HE}(q) = \begin{pmatrix} HE(q) \\ \mid Q \mid -1 \end{pmatrix} \tag{13.7}$$

The formulation described above can be done.

13.0.2.b Approach using V

It consists in building the OSNSP using CT instead of C.

$$h(q_1) = h(q_0) + \nabla^t H \delta q \tag{13.8}$$ {NE_projV}

ie:

$$h(q_1) = h(q_0) + \nabla^t HTV \tag{13.9}$$ {NE_projV}

We are looking for q_1 such that:

$$q_1 - q_0 = \nabla H \Lambda \tag{13.10}$$

We have

$$\delta q = TV, \qquad {}^tT\delta q = {}^t TTV, \qquad ({}^tTT)^{-1}\, {}^tT\delta q = V$$

ie

$$h(q_1) = h(q_0) + {}^t \nabla_q hT({}^tTT)^{-1}\, {}^tT\nabla_q h\Lambda \tag{13.11}$$

With $C = {}^t \nabla_q h$ leading to the prolem:

$$K \ni h(q_0) + CT\, ({}^tTT)^{-1}\, {}^t(CT)\Lambda \in K^* \tag{13.12}$$

Chapter 14

Simulation of a Cam Follower System

Main Contributors: *Mario di Bernardo, Gustavo Osorio, Stefania Santini*
University of Naples Federico II, Italy.

The free body dynamics can be described by a linear second order system. An external input is considered acting directly on the follower. This input is a non linear forcing component coming from the valve. The follower motion is constrained to a phase space region bounded by the cam position. The non conservative Newton restitution law is used for the computation of the post impact velocity. The cam is assumed to be massive therefore only rotational displacement is allowed. Under these assumptions, the free body dynamics of the follower can be described by

$$\mu\frac{d^2u(t)}{dt^2}+\zeta\frac{du(t)}{dt}+\kappa u(t)=f_v(t),\qquad \text{if}\quad u(t)>c(t). \tag{14.1}$$ {eq:sols}

where μ, ζ and κ are constant parameters for the follower mass, friction viscous damping and spring stiffness respectively. The state of the follower is given by the position $u(t)$ and velocity $v(t)=\dfrac{du}{dt}$. The external forcing is given by $f_v(t)$. The cam angular position determines $c(t)$ that defines the holonomic (i.e. constraint only on the position) rheonomic (i.e. time varying) constraint. The dynamic behavior when impacts occurs (i.e. $u(t)=c(t)$) is modelled via Newton's impact law that in this case is given by

$$v(t^+)=\frac{dc}{dt}-r\left(v(t^-)-\frac{dc}{dt}\right)=(1+r)\frac{dc}{dt}-rv(t^-),\ \text{ if }\ u(t)=c(t). \tag{14.2}$$ {eq:il}

where $v(t^+)$ and $v(t^-)$ are the post and pre impact velocities respectively, $\dfrac{dc}{dt}$ is the velocity vector of the cam at the contact point with the follower, and $r\in[0,1]$ is the restitution coefficient to model from plastic to elastic impacts. In Figure 14.1 is presented the schematic diagram of the physical cam-follower system. In Figure 14.1.a for $t=0$, 14.1.b for $t=\beta$, and 14.1.c the profile of the constraint position $\delta c(t)$, velocity $\dfrac{dc}{dt}(t)$ and acceleration $\dfrac{d^2c}{dt^2}(t)$. It is possible to visualize the follower displacement as a function of the cam position. It is also important to notice that different types of cams and followers profiles are used in practical applications.

14.0.1 The cam-follower as a Lagrangian NSDS.

It is possible to completely describe the cam-follower system as a driven impact oscillator into the framework of *Lagrangian NSDS* using a translation in space. Setting $\hat{u}(t)=u(t)-c(t)$ and $\hat{v}(t)=v(t)-dc/dt$, then equations (14.1) and (14.2) can be expressed as (the argument t will not be explicitly written)

$$\mu\frac{d^2\hat{u}}{dt^2}+\zeta\frac{d\hat{u}}{dt}+\kappa\hat{u}=f_v-\left(\mu\frac{d^2c}{dt^2}+\zeta\frac{dc}{dt}+\kappa c\right)\ \equiv\ \hat{f},\qquad \text{if}\quad \hat{u}>0. \tag{14.3}$$ {eq:trans}

$$\hat{v}^+\ =\ -r\hat{v}^-,\ \text{ if }\ \hat{u}=0. \tag{14.4}$$

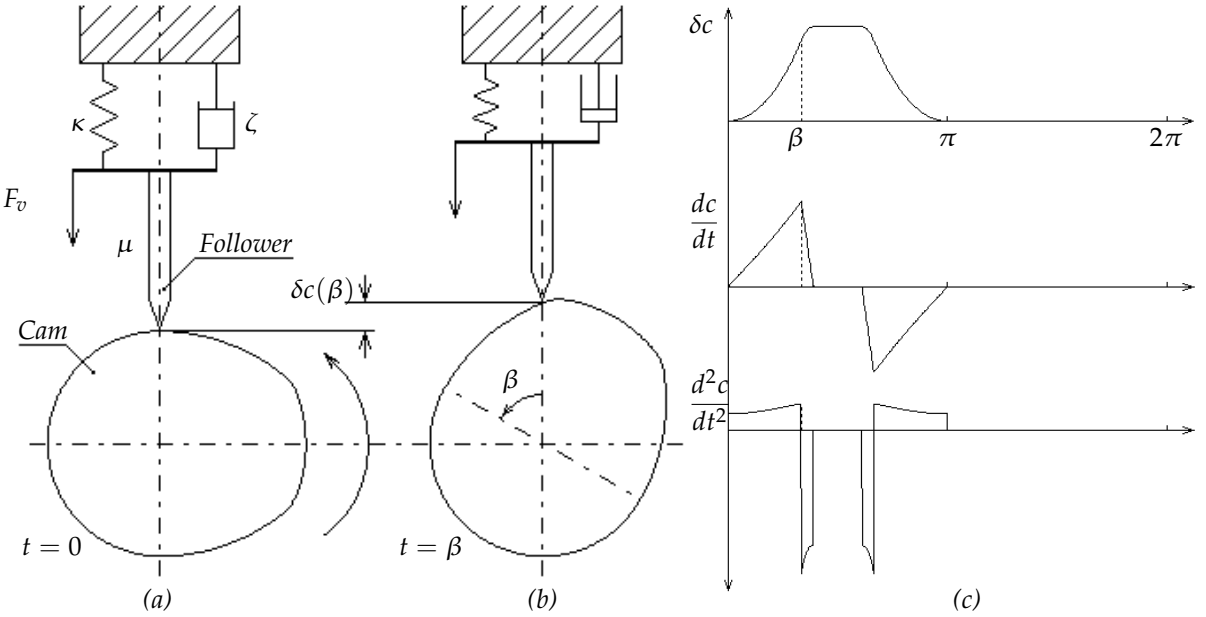


Figure 14.1: Cam-Shaft's schematics. *(a)* t=0. *(b)* t=β. *(c)* Constraint position $\delta c(t)$, velocity $\frac{dc}{dt}(t)$ and acceleration $\frac{d^2c}{dt}(t^2)$. {Fig:cam-shaft}

Using the framework presented in [2] we have that the equation of motion of a Lagrangian system may be stated as follows :

$$M(q)\ddot{q} + Q(q,\dot{q}) + F(\dot{q},q,t) = F_{ext}(t) + R \tag{14.5}$$ {eq:lag1}

From the (14.3) we can derive all of the terms which define a Lagrangian NSDS. In our case the model is completely linear:

$$\begin{aligned} q &= \begin{bmatrix} \hat{u} \end{bmatrix} \\ M(q) &= \begin{bmatrix} \mu \end{bmatrix} \\ Q(q,\dot{q}) &= \begin{bmatrix} 0 \end{bmatrix} \\ F(q,\dot{q}) &= \begin{bmatrix} \zeta \end{bmatrix}\dot{q} + \begin{bmatrix} \kappa \end{bmatrix} q \\ F_{ext} &= \begin{bmatrix} \hat{f} \end{bmatrix} \end{aligned} \tag{14.6}$$ {eq:lag2}

The unilateral constraint requires that:

$$\hat{u} \geq 0$$

so we can obtain

$$\begin{aligned} y &= H^T q + b \\ H^T &= \begin{bmatrix} 1 \end{bmatrix} \\ b &= 0 \end{aligned} \tag{14.7}$$ {eq:constr}

In the same way, the reaction force due to the constraint is written as follows:

$$R = H\lambda, \qquad \text{with } H = \begin{bmatrix} 1 \end{bmatrix}$$

The unilataral contact law may be formulated as follow:

$$0 \leq y \perp \lambda \geq 0 \tag{14.8}$$ {eq:119}

and the Newton's impact law:

$$\text{If } y = 0, \dot{y}^+ = -r\dot{y}^- \tag{14.9}$$ {eq:120}

14.0.2 Implementation in the platform

For the simulation of the cam follower system follow the steps

1. Move to the working directory `sample/CamFollower`

   ```
   $cd sample/CamFollower
   ```

2. Clean the directory form binary files using the `siconos` command

   ```
   $siconos -c
   ```

3. Compile the file `CamFollowerNoXml.cpp` in the sample folder (*See* the code at the end of the section)

   ```
   $siconos CamFollowerNoXml.cpp
   ```

4. Change the simulation parameters (*i.e.* Follower initial position and velocity, cam initial angle, simulations time, cam rotational speed in rpm, etc.) in the file `CamFollowerNoXml.cpp`.

Next we present the sample code for the `CamFollowerNoXml.cpp` file:

```
int main(int argc, char* argv[]) {
  {
    // ======== Creation of the model ==============
    // User-defined main parameters
    double rpm=358;
    double phi_0=0;
    unsigned int dsNumber = 1;      // the Follower and the ground
    unsigned int nDof = 1;          // degrees of freedom for the Follower
    double t0 = 0;                  // initial computation time
    double T = 5;                   // final computation time
    double h = 0.0001;              // time step
    int Kplot;
    Kplot=(int)(Tplot/h);
    double position_init = 0.4;     // initial position for lowest bead.
    double velocity_init = 0.4;     // initial velocity for lowest bead.

    // ======= Dynamical systems =========

    vector<DynamicalSystem *> vectorDS; // the list of DS
    vectorDS.resize(dsNumber,NULL);

    SiconosMatrix *Mass, *K, *C; // mass/rigidity/viscosity
    Mass = new SiconosMatrix(nDof,nDof);
    (*Mass)(0,0) = 1.221;
    K = new SiconosMatrix(nDof,nDof);
    (*K)(0,0) = 1430.8;
    C = new SiconosMatrix(nDof,nDof);
    (*C)(0,0) = 0;

    // Initial positions and velocities
    vector<SimpleVector *> position_0;
    vector<SimpleVector *> velocity_0;
    position_0.resize(dsNumber,NULL);
    velocity_0.resize(dsNumber,NULL);
    position_0[0] = new SimpleVector(nDof);
    velocity_0[0] = new SimpleVector(nDof);
    (*(position_0[0]))(0) = position_init;
```

```
(*(velocity_0[0]))(0) = velocity_init;

vectorDS[0] =
new LagrangianLinearTIDS(0,nDof,*(position_0[0]),*(velocity_0[0]),*Mass,*K,*C);

static_cast<LagrangianDS*>(vectorDS[0])
                    ->setComputeFExtFunction("FollowerPlugin.so", "FollowerFExt");

// Example to set a list of parameters in FExt function.
// 1 - Create a simple vector that contains the required parameters.

// Here we set two parameters, the DS number.
SimpleVector * param = new SimpleVector(2);

(*param)(0)=rpm;
(*param)(1)=phi_0;
// 2 - Assign this param to the function FExt
static_cast<LagrangianDS*>(vectorDS[0])->setParametersListPtr(param,2);
// 2 corresponds to the position of FExt in the stl vector of possible parameters.
// 0 is mass, 1 FInt.
// Now the cam rotational velocity in rpms will be available in FExt plugin.

// ===== Interactions =====

vector<Interaction*> interactionVector;
interactionVector.resize(1,NULL);
vector<DynamicalSystem*> *dsConcerned =
                    new vector<DynamicalSystem*>(dsNumber);

// ===== Non Smooth Law =====
double e = 0.8;
// Interaction Follower-floor
SiconosMatrix *H = new SiconosMatrix(1,nDof);
(*H)(0,0) = 1.0;
NonSmoothLaw * nslaw = new NewtonImpactLawNSL(e);
Relation * relation = new LagrangianLinearR(*H);
(*dsConcerned)[0] = vectorDS[0];
interactionVector[0] = new Interaction("Follower-Ground",0,1, dsConcerned);
interactionVector[0]->setRelationPtr(relation);
interactionVector[0]->setNonSmoothLawPtr(nslaw);
// ===== Interactions =====

// ===== NonSmoothDynamicalSystem =====
bool isBVP =0;
NonSmoothDynamicalSystem * nsds =
                        new NonSmoothDynamicalSystem(isBVP);

// Set DS of this NonSmoothDynamicalSystem
nsds->setDynamicalSystems(vectorDS);
// Set interactions of the NonSmoothDynamicalSystem
nsds->setInteractions(interactionVector);

// ===== Model =====
```

```
Model * Follower = new Model(t0,T);
// set NonSmoothDynamicalSystem of this model
Follower->setNonSmoothDynamicalSystemPtr(nsds);

// ====== Strategy ======

double theta = 0.5;           // theta for Moreau integrator
string solverName = "QP" ;

Strategy* S = new TimeStepping(Follower);

// – Time discretisation –
TimeDiscretisation * t = new TimeDiscretisation(h,S);

// – OneStepIntegrators –
vector<OneStepIntegrator *> vOSI;
vOSI.resize(dsNumber,NULL);
vOSI[0] = new Moreau(t,vectorDS[0],theta);
S->setOneStepIntegrators(vOSI);

// – OneStepNsProblem –
OneStepNSProblem * osnspb = new LCP(S,solverName,101, 0.0001,"max",0.6);
S->setOneStepNSProblemPtr(osnspb); // set OneStepNSProblem of the strategy
cout « "=== End of model loading === " « endl;
// ==== End of model definition======



// ========= Computation============

// — Strategy initialization —
S->initialize();
cout «"End of strategy initialisation" « endl;

int k = t->getK();                  // Current step
int N = t->getNSteps();             // Number of time steps

// — Get the values to be plotted —
// -> saved in a matrix dataPlot
unsigned int outputSize = 8;

SiconosMatrix DataPlot(Kplot+1,outputSize );
// For the initial time step:

// time
DataPlot(k,0) = k*t->getH();

DataPlot(k,1) = static_cast<LagrangianDS*>(vectorDS[0])->getQ()(0);
DataPlot(k,2) = static_cast<LagrangianDS*>(vectorDS[0])->getVelocity()(0);
DataPlot(k,3) = (Follower->getNonSmoothDynamicalSystemPtr()->
                 getInteractionPtr(0)->getLambda(1))(0);
DataPlot(k,4) = static_cast<LagrangianDS*>(vectorDS[0])->getFExt()(0);

// State of the Cam
```

```
double CamEqForce,CamPosition,CamVelocity,CamAcceleration;
CamEqForce=
        CamState(k*t->getH(),rpm,CamPosition,CamVelocity,CamAcceleration);
// Position of the Cam
DataPlot(k, 5) = CamPosition;
// Velocity of the Cam
DataPlot(k, 6) = CamVelocity;
// Acceleration of the Cam
DataPlot(k, 7) =
        CamPosition+static_cast<LagrangianDS*>(vectorDS[0])->getQ()(0);

// — Time loop —
cout « "Start computation ... " « endl;
while(k < N)
{

    // — Get values to be plotted —
    DataPlot(k,0) = k*t->getH();

    DataPlot(k,1) =
        static_cast<LagrangianDS*>(vectorDS[0])->getQ()(0);
    DataPlot(k,2) =
        static_cast<LagrangianDS*>(vectorDS[0])->getVelocity()(0);
    DataPlot(k,3) =
        (Follower->getNonSmoothDynamicalSystemPtr()->
        getInteractionPtr(0)->getLambda(1))(0);
    DataPlot(k,4) = static_cast<LagrangianDS*>(vectorDS[0])->getFExt()(0);

    CamEqForce=
    CamState(k*t->getH(),rpm,CamPosition,CamVelocity,CamAcceleration);

    DataPlot(k, 5) = CamPosition;
    DataPlot(k, 6) = CamVelocity;
    DataPlot(k, 7) = CamPosition+
        static_cast<LagrangianDS*>(vectorDS[0])->getQ()(0);
    // transfer of state i+1 into state i and time incrementation
    S->nextStep();
    // get current time step
    k = t->getK();
    // solve ...
    S->computeFreeState();
    S->computeOneStepNSProblem();
    // update
    S->update();
}
// — Output files —
DataPlot.rawWrite("result.dat", "ascii");
// — Free memory —
delete osnspb;
delete vOSI[0];
delete t;
delete S;
delete Follower;
delete nsds;
```

```
    delete interactionVector[0];
    delete relation;
    delete nslaw;
    delete H;
    delete dsConcerned;
    delete vectorDS[0];
    delete position_0[0];
    delete velocity_0[0];
    delete C;
    delete K;
    delete Mass;

}
```

14.0.3 Simulation

We have perform the simulation of the cam follower system for different values of the cam rotational speed with the SICONOS software package using a time-stepping numerical scheme with step size ($h = 1e^{-4}$) and an event-driven scheme with minimum step size ($h_{min} = 1e^{-12}$). Fig. 14.2 and 14.3 show the time simulations for different values of the cam rotational speed and Fig. 14.4 show the chaotic attractor at $rpm = 660$ for impact and stroboscopic Poincarè sections.

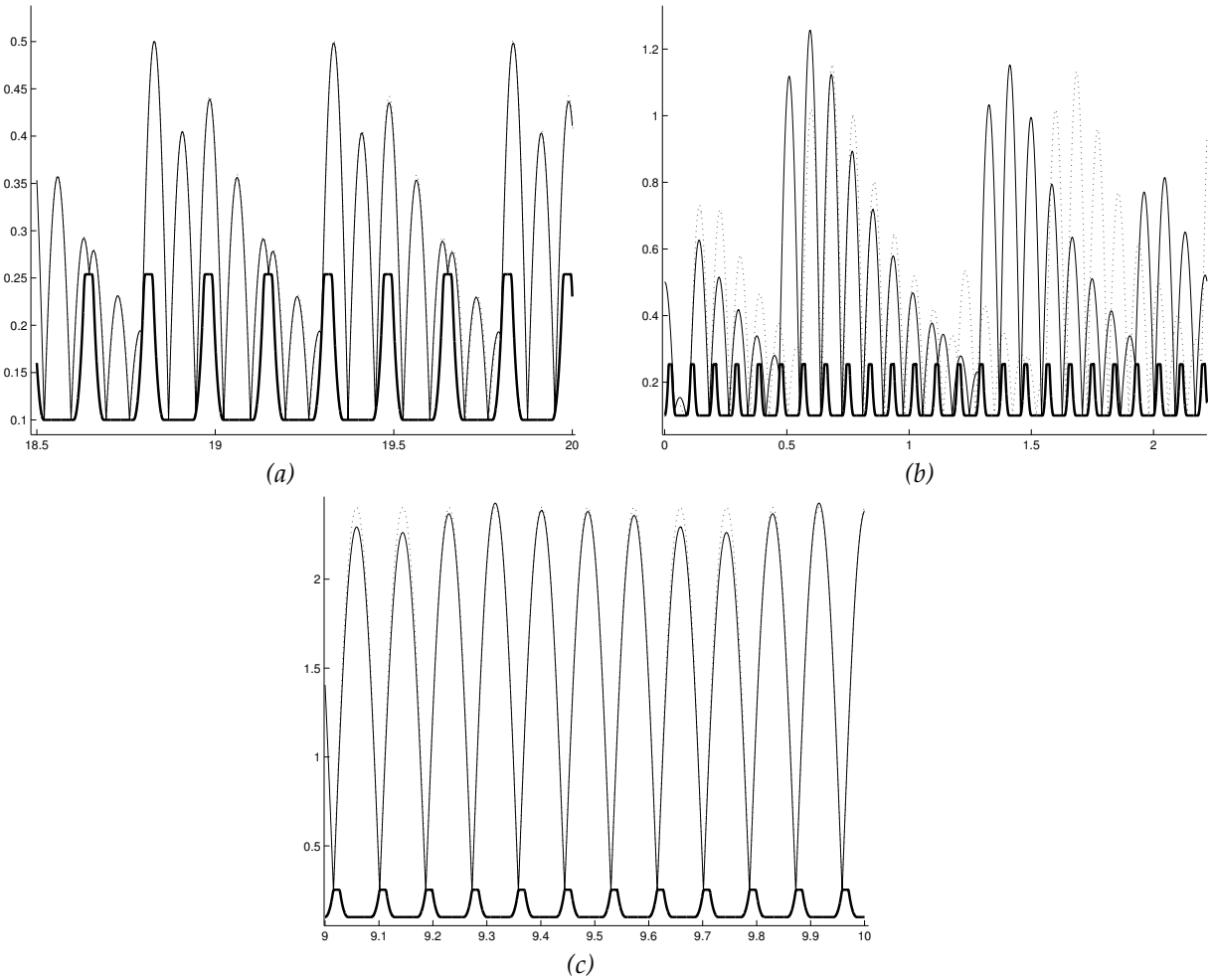


Figure 14.2: Time series using SICONOS platform. Time-stepping scheme (continuous line). Event-driven scheme (dashed line) *(a)* rpm=358. *(b)* rpm=660. *(c)* rpm=700. {Fig:time_compari

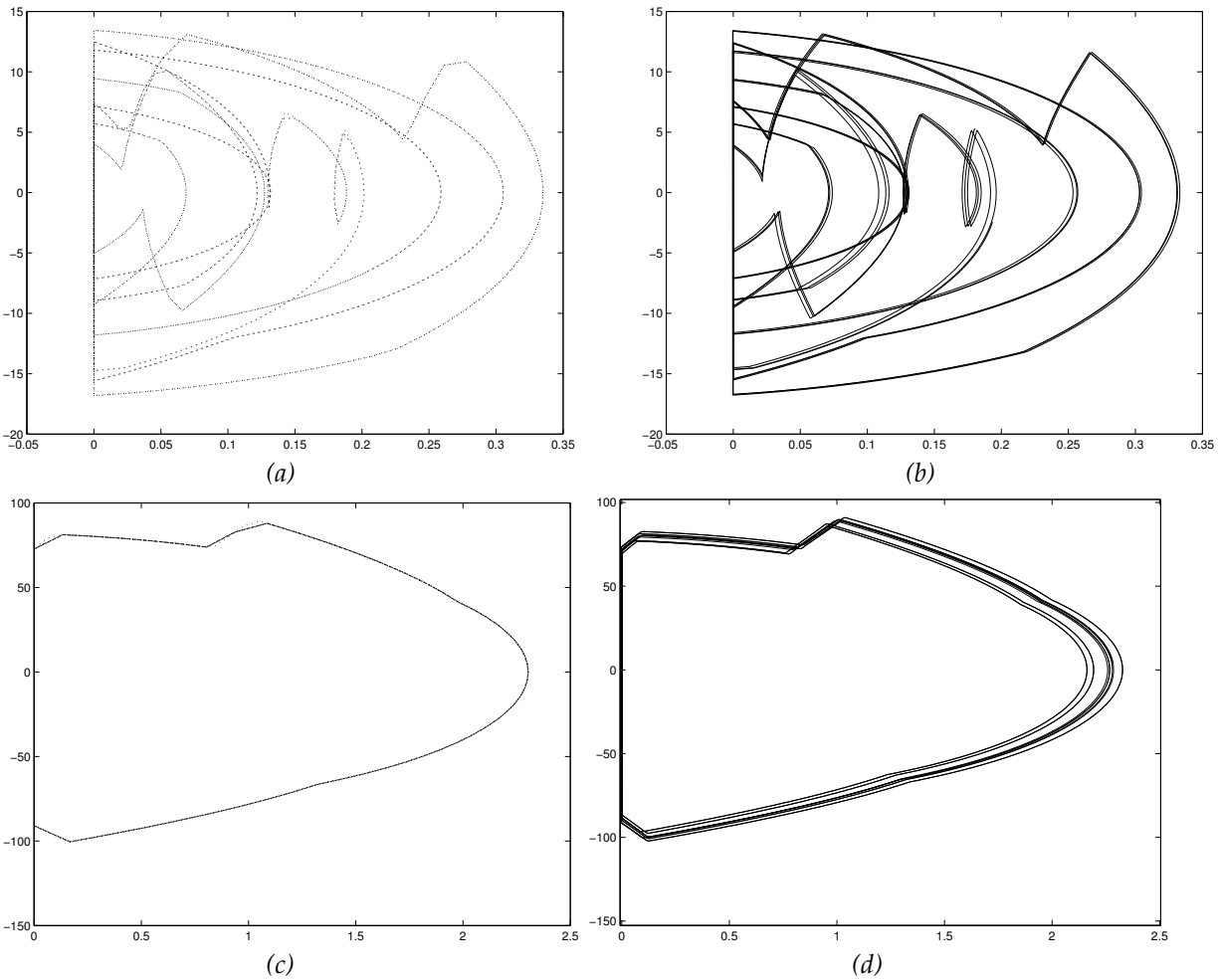


Figure 14.3: State space comparison using SICONOS platform. *(a)* rpm=358. Event Driven *(b)* rpm=358. Time Stepping ($h = 1e^{-4}$)*(c)* rpm=700. Event Driven *(d)* rpm=700. Time Stepping ($h = 1e^{-4}$)

{Fig:state_compa

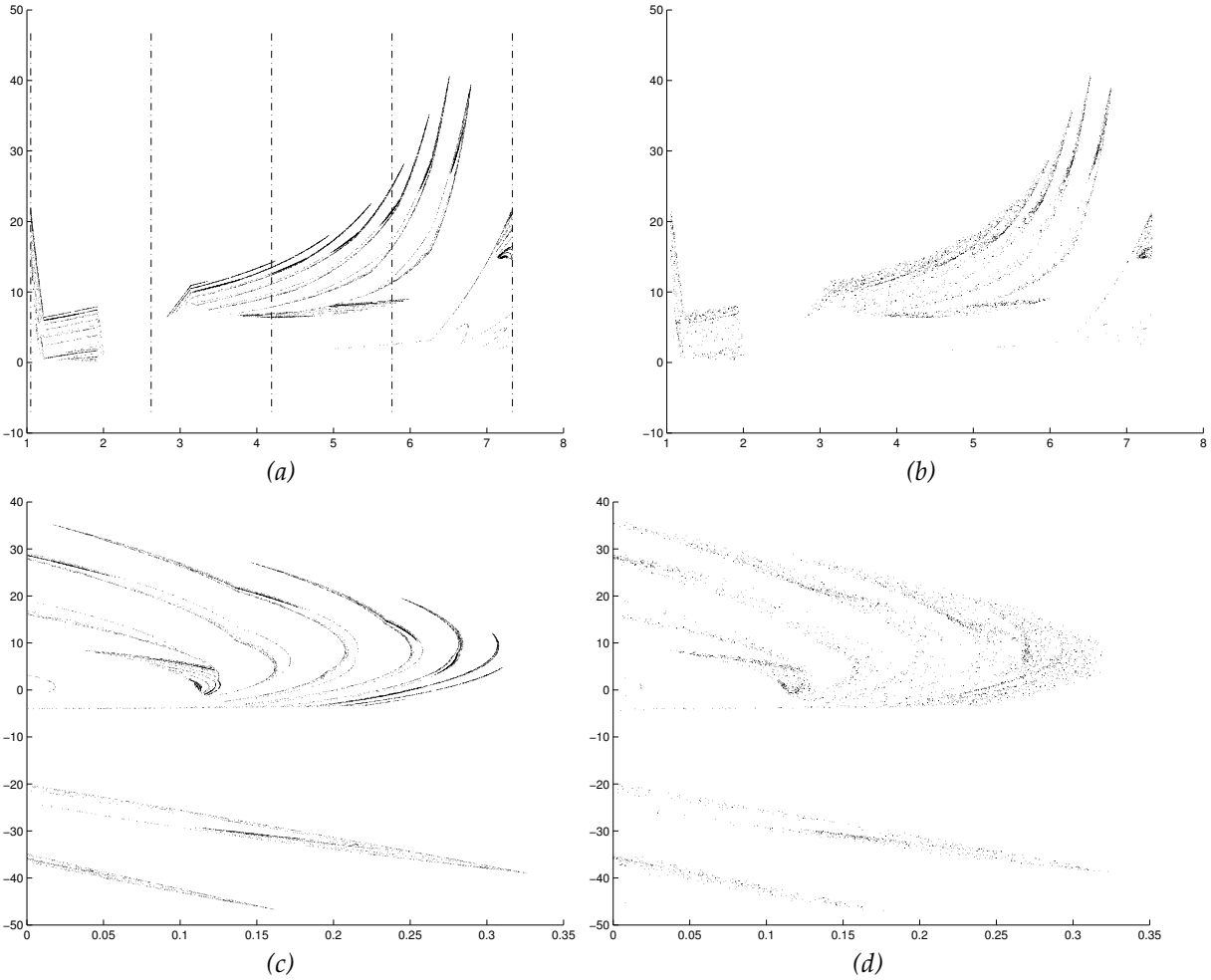


Figure 14.4: Attractors comparison using SICONOS platform at rpm=660. *(a)* Impact map. (Event Driven) *(b)* Impact Map. Time Stepping ($h = 1e^{-4}$)*(a)* Stroboscopic map. (Event Driven) *(b)* Stroboscopic Map. Time Stepping ($h = 1e^{-4}$)

{Fig:attractor_c

Chapter 15

Quartic Formulation

15.0.1 Slidding ?

It consists in finding $\alpha > 0$ and $R \in \partial K_\mu$ such that $-\alpha \begin{pmatrix} 0 \\ R_T \end{pmatrix} = MR + q$. That is :

$$\left[M + \begin{pmatrix} 0 & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & \alpha \end{pmatrix} \right] R + q = 0 \tag{15.1}$$ {eq_quartic1}

15.0.1.a R_T is on a conic

The first line of the system 15.1 and the $R \in \partial K_\mu$ is the intersection between a plan and a cone in $\mathbb{R}^3$, endeed:

$$\begin{aligned} \mu R_N &= \| R_T \| \\ \frac{M_{11}}{\mu} \| R_T \| &= -q_1 - M_{12}R_{T1} - M_{13}R_{T2} \end{aligned} \tag{15.2}$$ {eq_quartic2}

That is:

$$\begin{aligned} \mu^2 R_N^2 &= (R_{T1}^2 + R_{T1}^2) \\ \frac{M_{11}^2}{\mu^2}(R_{T1}^2 + R_{T1}^2) &= (-q_1 - M_{12}R_{T1} - M_{13}R_{T2})^2 \end{aligned} \tag{15.3}$$ {eq_quartic2}

That means that R_T is contained in a conic, focus and directrice are:

$$\begin{aligned} &\mathcal{D} : q_1 + M_{12}R_{T1} + M_{13}R_{T2} = 0 \\ &focus : \mathcal{O} \\ &\frac{M_{11}^2}{\mu^2} Dist(\mathcal{O}, R_T)^2 = Dist(\mathcal{D}, R_T)^2 (M_{12}^2 + M_{13}^2) \\ &\frac{Dist(\mathcal{O}, R_T)}{Dist(\mathcal{D}, R_T)} = \frac{\mu\sqrt{(M_{12}^2 + M_{13}^2)}}{M_{11}} = e \end{aligned} \tag{15.4}$$ {eq_quartic3}

The parametric equation is:

$$\begin{aligned} R_{T1} &= rcos(\theta) \\ R_{T2} &= rsin(\theta) \\ r &= \frac{p}{1 + ecos(\theta - \phi)} \end{aligned} \tag{15.5}$$ {eq_quartic4}

With p an simple expression of M_{11}, M_{12}, M_{13}, and ϕ a constant angle between $\mathcal{D}$ and (O, R_{T1})

15.0.1.b The two last line of the system 15.1

$$\frac{\parallel R_T \parallel}{\mu}\tilde{M}_{1.} + \left(\tilde{M} + \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix}\right) R_T + \tilde{q} = 0 \tag{15.6}$$ {eq_quartic5}

$\tilde{M}$ is symetric, so it exists a unitary matrix V such that $V\tilde{M}V^T = \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix}$. One can get:

$$\frac{\parallel R_T \parallel}{\mu}V\tilde{M}_{1.} + V\left(\tilde{M} + \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix}\right) V^T V R_T + V\tilde{q} = 0 \tag{15.7}$$ {eq_quartic6}

Rename:

$$\frac{\parallel \bar{R}_T \parallel}{\mu}\bar{M}_{1.} + \begin{pmatrix} d_1+\alpha & 0 \\ 0 & d_2+\alpha \end{pmatrix} \overline{R}_T + \bar{q} = 0 \tag{15.8}$$ {eq_quartic7}

In the plan, either V is a rotation or a symetrie. So $\bar{R}_T = VR_T$ is a conic with the same focus and a rotated directrice, it means that it exists ϕ_1 such that :

$$\begin{array}{l} \bar{R}_{T1} = rcos(\theta) \\ \bar{R}_{T2} = rsin(\theta) \\ r = \dfrac{p}{1+ecos(\theta-\phi_1)} \end{array} \tag{15.9}$$ {eq_quartic8}

The equation 15.8 is :

$$\begin{array}{l} (d_1+\alpha)\bar{R}_{T1} = -\bar{q}_1 + a_1 \parallel R_T \parallel \\ (d_2+\alpha)\bar{R}_{T2} = -\bar{q}_2 + a_2 \parallel R_T \parallel \end{array} \tag{15.10}$$ {eq_quartic9}

The case ($\bar{R}_{T1} = 0$ or $\bar{R}_{T2} = 0$) has to be examine. We try to eliminate $alpha$:

$$\begin{array}{l} d_1\bar{R}_{T1}\bar{R}_{T2} + \alpha\bar{R}_{T1}\bar{R}_{T2} = -\bar{q}_1\bar{R}_{T2} + a_1\bar{R}_{T2} \parallel R_T \parallel \\ d_2\bar{R}_{T1}\bar{R}_{T2} + \alpha\bar{R}_{T1}\bar{R}_{T2} = -\bar{q}_2\bar{R}_{T1} + a_2\bar{R}_{T1} \parallel R_T \parallel \end{array} \tag{15.11}$$ {eq_quartic10}

that leads to:

$$(d_1-d_2)\bar{R}_{T1}\bar{R}_{T2} = -\bar{q}_1\bar{R}_{T2} + \bar{q}_2\bar{R}_{T1} + (a_1\bar{R}_{T2} - a_2\bar{R}_{T1}) \parallel R_T \parallel \tag{15.12}$$ {eq_quartic10}

The parametric expression of $\bar{R}_T$ leads to:

$$\begin{array}{l} (d_1-d_2)r^2cos(\theta)sin(\theta) = -\bar{q}_1 rsin(\theta) + \bar{q}_2 rcos(\theta) + r(a_1 rsin(\theta) - a_2 rcos(\theta)) \\ \text{ie:}(d_1-d_2)rcos(\theta)sin(\theta) = -\bar{q}_1 sin(\theta) + \bar{q}_2 cos(\theta) + r(a_1 sin(\theta) - a_2 cos(\theta)) \end{array} \tag{15.13}$$ {eq_quartic11}

with the expression of r:

$$\begin{array}{l} (d_1-d_2)\dfrac{p}{1+ecos(\theta-\phi_1)}cos(\theta)sin(\theta) = \\ -\bar{q}_1 sin(\theta) + \bar{q}_2 cos(\theta) + \dfrac{p}{1+ecos(\theta-\phi_1)}(a_1 sin(\theta) - a_2 cos(\theta)) \end{array}$$

$$\begin{array}{l} \text{ie:}(d_1-d_2)pcos(\theta)sin(\theta) = \\ (1+ecos(\theta-\phi_1))(-\bar{q}_1 sin(\theta) + \bar{q}_2 cos(\theta)) + p(a_1 sin(\theta) - a_2 cos(\theta)) \end{array} \tag{15.14}$$ {eq_quartic12}

$$\begin{array}{l} \text{ie:}(d_1-d_2)pcos(\theta)sin(\theta) = \\ (1+e(cos(\theta)cos(\phi_1) + sin(\theta)sin(\phi_1)))(-\bar{q}_1 sin(\theta) + \bar{q}_2 cos(\theta)) + p(a_1 sin(\theta) - a_2 cos(\theta)) \end{array}$$

$$\begin{array}{l} \text{ie:}(d_1-d_2)pcos(\theta)sin(\theta) + \\ (1+ecos(\theta)cos(\phi_1) + esin(\theta)sin(\phi_1))(\bar{q}_1 sin(\theta) - \bar{q}_2 cos(\theta)) + p(-a_1 sin(\theta) + a_2 cos(\theta)) = 0 \end{array}$$

rename :

$$Acos(\theta)^2 + Bsin(\theta)^2 + Csin(\theta)cos(\theta) + Dsin(\theta) + Ecos(\theta) = 0 \tag{15.15}$$ {eq_quartic13}

with

$$\begin{aligned} A &= -e\bar{q}_2 cos(\phi_1) \\ B &= e\bar{q}_1 sin(\phi_1) \\ C &= (d_1 - d_2)p + ecos(\phi_1)\bar{q}_1 - esin(\phi_1)\bar{q}_2 \\ D &= \bar{q}_1 - pa_1 \\ E &= -\bar{q}_2 + pa_2 \end{aligned} \tag{15.16}$$ {eq_quartic12}

rename : Using the following set of unknown :

$$\begin{aligned} t &= tan(\theta/2) \\ sin(\theta) &= \frac{2t}{1+t^2} \\ cos(\theta) &= \frac{1-t^2}{1+t^2} \end{aligned} \tag{15.17}$$ {eq_quartic14}

leads to:

$$\begin{aligned} &A\frac{(1-t^2)^2}{1+t^2} + B\frac{4t^2}{1+t^2} + C\frac{2t(1-t^2)}{1+t^2} + D2t + E(1-t^2) = 0 \\ &\text{ie:} A(1-t^2)^2 + 4Bt^2 + C2t(1-t^2) + 2Dt(1+t^2) + E(1-t^2)(1+t^2) = 0 \end{aligned} \tag{15.18}$$ {eq_quartic13}

ie:$P_4 = A - E \qquad P_3 = -2C + 2D \qquad P_2 = 4B - 2A \qquad P_1 = 2C + 2D \qquad P_0 = A + E$

Finally, we get 4 possible values for R_T, checking the sign of α and R_N selects the solutions.

15.0.1.c case $R_{T12} = 0$

From 15.10, R_{T1} leads to:

$$\begin{aligned} \| R_T \| &= |\bar{R}_{T2}| = \frac{\bar{q}_1}{a_1} \\ \bar{R}_T &= \begin{pmatrix} 0 \\ \pm\frac{\bar{q}_1}{a_1} \end{pmatrix} \end{aligned} \tag{15.19}$$ {eq_quartic14}

From 15.10, R_{T2} leads to:

$$\begin{aligned} \| R_T \| &= |\bar{R}_{T1}| = \frac{\bar{q}_2}{a_2} \\ \bar{R}_T &= \begin{pmatrix} \pm\frac{\bar{q}_2}{a_2} \\ 0 \end{pmatrix} \end{aligned} \tag{15.20}$$ {eq_quartic14}

From $\bar{R}_T$, we have to check the coherence with the equation 15.9. If it is on the conic, we compute R, and the sign condition of the equation 15.1 must be check.

Chapter 16

Alart–Curnier Formulation

16.1 Reduced formulation to local variables.

16.1.1 Formulation

Let us start with

$$\begin{aligned}\Phi_1(U,P) &= -U_{k+1} + \widehat{W} P_{k+1} + U_{\text{free}} \\ \Phi_2(U,P) &= P_{\text{N}} - \text{proj}_{\mathbb{R}^a_+}(P_{\text{N}} - \rho_{\text{N}} \circ (U_{\text{N}} + e \circ U_{\text{N},k})) \\ \Phi_3(U,P) &= P_{\text{T}} - \text{proj}_{\widehat{\mathbf{D}}(P_{\text{N}},U_{\text{N}})}(P_{\text{T}} - \rho_{\text{T}} \circ U_{\text{T}})\end{aligned} \tag{16.1}$$ {eq:AC-L7}

where the modified friction disk for a contact α is

$$\widehat{\mathbf{D}}^{\alpha}(P^{\alpha}_{\text{N},k+1}, U^{\alpha}_{\text{N},k+1}) = \mathbf{D}(\mu(\text{proj}_{\mathbb{R}_+}(P^{\alpha}_{\text{N},k+1} - \rho^{\alpha}_{\text{N}}\,(U^{\alpha}_{\text{N},k+1} + e^{\alpha} U^{\alpha}_{\text{N},k}))). \tag{16.2}$$ {eq:AC-L3}

16.1.2 Structure of the Jacobians

Let us denote the one element of the generalized Jacobian by $H(U,P) \in \partial\Phi(U,P)$ which has the structure

$$H(U,P) = \begin{bmatrix} -I & 0 & \widehat{W}_{\text{NN}} & \widehat{W}_{\text{NT}} \\ 0 & -I & \widehat{W}_{\text{TN}} & \widehat{W}_{\text{TT}} \\ \partial_{U_{\text{N}}}\Phi_2(U,P) & 0 & \partial_{P_{\text{N}}}\Phi_2(U,P) & 0 \\ \partial_{U_{\text{N}}}\Phi_3(U,P) & \partial_{U_{\text{T}}}\Phi_3(U,P) & \partial_{P_{\text{N}}}\Phi_3(U,P) & \partial_{P_{\text{T}}}\Phi_3(U,P) \end{bmatrix} \tag{16.3}$$ {eq:AC-L6}

16.1.3 Computation of the gradients

Let us consider the single contact case.

Computation of the gradients of Φ_2

$$\Phi_2(U,P) = P_{\text{N}} - \text{proj}_{\mathbb{R}^a_+}(P_{\text{N}} - \rho_{\text{N}}(U_{\text{N}} + eU_{\text{N},k})) \tag{16.4}$$ {eq:AC-T1}

- **If** $P_{\text{N}} - \rho_{\text{N}}(U_{\text{N}} + eU_{\text{N},k}) \geq 0$, we get

$$\Phi_2(U,P) = +\rho_{\text{N}}(U_{\text{N}} + eU_{\text{N},k}) \tag{16.5}$$ {eq:AC-T2}

and

$$\partial_{U_N}\Phi_2(U,P) = +\rho_N$$
$$\partial_{P_N}\Phi_2(U,P) = 0 \tag{16.6}$$ {eq:AC-T3}

- **If** $P_N - \rho_N(U_N + eU_{N,k}) < 0$, we get

$$\Phi_2(U,P) = P_N \tag{16.7}$$ {eq:AC-T4}

and

$$\partial_{U_N}\Phi_2(U,P) = 0$$
$$\partial_{P_N}\Phi_2(U,P) = 1 \tag{16.8}$$ {eq:AC-T5}

Computation of the gradients of Φ_3

$$\Phi_3(U,P) = P_T - \text{proj}_{\widehat{\mathbf{D}}(P_N,U_N)}(P_T - \rho_T U_T) \tag{16.9}$$ {eq:AC-TT1}

- **If** $\|P_T - \rho_T U_T\| \leq \mu \max(0, P_N - \rho_N(U_N + eU_{N,k}))$, we get

$$\Phi_3(U,P) = +\rho_T U_T \tag{16.10}$$ {eq:AC-TT2}

and

$$\partial_{U_N}\Phi_3(U,P) = 0$$
$$\partial_{P_N}\Phi_3(U,P) = 0$$
$$\partial_{U_T}\Phi_3(U,P) = +\rho_T$$
$$\partial_{P_T}\Phi_3(U,P) = 0 \tag{16.11}$$ {eq:AC-TT3}

- **If** $\|P_T - \rho_T U_T\| > \mu \max(0, P_N - \rho_N(U_N + eU_{N,k}))$, we get

$$\Phi_3(U,P) = P_T - \mu \max(0, P_N - \rho_N(U_N + eU_{N,k}))\frac{P_T - \rho_T U_T}{\|P_T - \rho_T U_T\|} \tag{16.12}$$ {eq:AC-TT4}

  - **If** $P_N - \rho_N(U_N + eU_{N,k}) \leq 0$, we get

$$\Phi_3(U,P) = P_T \tag{16.13}$$ {eq:AC-TT5}

and

$$\partial_{U_N}\Phi_3(U,P) = 0$$
$$\partial_{P_N}\Phi_3(U,P) = 0$$
$$\partial_{U_T}\Phi_3(U,P) = 0$$
$$\partial_{P_T}\Phi_3(U,P) = I_2 \tag{16.14}$$ {eq:AC-TT6}

  - **If** $P_N - \rho_N(U_N + eU_{N,k}) > 0$, we get

$$\Phi_3(U,P) = P_T - \mu(P_N - \rho_N(U_N + eU_{N,k}))\frac{P_T - \rho_T U_T}{\|P_T - \rho_T U_T\|} \tag{16.15}$$ {eq:AC-TT7}

and

$$
\begin{aligned}
\partial_{U_N}\Phi_3(U,P) &= \mu\rho_N\frac{P_T-\rho_T U_T}{\|P_T-\rho_T U_T\|}\textbf{WARNING case was not taken into account}\\
\partial_{P_N}\Phi_3(U,P) &= -\mu\frac{P_T-\rho_T U_T}{\|P_T-\rho_T U_T\|}\\
\partial_{U_T}\Phi_3(U,P) &= \mu\rho_T(P_N-\rho_N(U_N+eU_{N,k}))\Gamma(P_T-\rho_T U_T)\\
\partial_{P_T}\Phi_3(U,P) &= I_2-\mu(P_N-\rho_N(U_N+eU_{N,k}))\Gamma(P_T-\rho_T U_T)
\end{aligned}
\tag{16.16}
$$

{eq:AC-TT8}

16.1.4 Rearranging the cases

TO BE COMPLETED

16.2 Formulation with global variables.

16.2.1 Formulation

Let us start with

$$
\begin{aligned}
\Psi_1^a(v,U,P) &= -\widehat{M}v_{k+1}+HP_{k+1}+q\\
\Psi_1^b(v,U,P) &= -U_{k+1}+H^\top v_{k+1}+b\\
\Psi_2(v,U,P) &= P_N-\mathrm{proj}_{\mathbb{R}^a_+}(P_N-\rho_N\circ(U_N+e\circ U_{N,k}))\\
\Psi_3(v,U,P) &= P_T-\mathrm{proj}_{\widehat{\mathbf{D}}(P_N,U_N)}(P_T-\rho_T\circ U_T)
\end{aligned}
\tag{16.17}
$$

{eq:GAC-L1}

where the modified friction disk for a contact α is

$$
\widehat{\mathbf{D}}^\alpha(P^\alpha_{N,k+1},U^\alpha_{N,k+1})=\mathbf{D}(\mu(\mathrm{proj}_{\mathbb{R}_+}(P^\alpha_{N,k+1}-\rho^\alpha_N(U^\alpha_{N,k+1}+e^\alpha U^\alpha_{N,k}))).
\tag{16.18}
$$

{eq:GAC-L2}

16.2.2 Structure of the Jacobians

Let us denote the one element of the generalized Jacobian by $H(v,U,P)\in\partial\Psi(s,U,P)$ which has the structure

$$
H(v,U,P)=\begin{bmatrix}
-\widehat{M} & 0 & 0 & H_N & H_T\\
H_N^\top & -I & 0 & 0 & 0\\
H_T^\top & 0 & -I & 0 & 0\\
0 & \partial_{U_N}\Psi_2(v,U,P) & 0 & \partial_{P_N}\Psi_2(v,U,P) & 0\\
0 & \partial_{U_N}\Psi_3(v,U,P) & \partial_{U_T}\Psi_3(v,U,P) & \partial_{P_N}\Psi_3(v,U,P) & \partial_{P_T}\Psi_3(v,U,P)
\end{bmatrix}
\tag{16.19}
$$

{eq:GAC-L3}

We clearly have

$$
\begin{aligned}
\partial_U\Psi_2(v,U,P) &= \partial_U\Phi_2(U,P)\\
\partial_P\Psi_2(v,U,P) &= \partial_P\Phi_2(U,P)\\
\partial_U\Psi_3(v,U,P) &= \partial_U\Phi_3(U,P)\\
\partial_P\Psi_3(v,U,P) &= \partial_P\Phi_3(U,P)
\end{aligned}
\tag{16.20}
$$

{eq:equivalentJa

and we get

$$H(v,U,P) = \begin{bmatrix} -\widehat{M} & 0 & 0 & H_{\text{N}} & H_{\text{T}} \\ H_{\text{N}}^{\top} & -I & 0 & 0 & 0 \\ H_{\text{T}}^{\top} & 0 & -I & 0 & 0 \\ 0 & \partial_{U_{\text{N}}}\Phi_2(U,P) & 0 & \partial_{P_{\text{N}}}\Phi_2(U,P) & 0 \\ 0 & \partial_{U_{\text{N}}}\Phi_3(U,P) & \partial_{U_{\text{T}}}\Phi_3(U,P) & \partial_{P_{\text{N}}}\Phi_3(U,P) & \partial_{P_{\text{T}}}\Phi_3(U,P) \end{bmatrix} \tag{16.21}$$ {eq:GAC-L4}

16.2.3 Simplification ?

Since the second line Ψ_1^b is linear, we should be able to derive a reduced Jacobian using the chain rule. Let us define $\widetilde{\Psi}$

$$\widetilde{\Psi}(v,P) = \Psi(v, H^{\top}v + b, P) \tag{16.22}$$ {eq:chainrule}

$$\begin{aligned} \widetilde{\Psi}_1(v,P) &= -\widehat{M}v_{k+1} + HP_{k+1} + q \\ \widetilde{\Psi}_2(v,P) &= P_{\text{N}} - \text{proj}_{\mathbb{R}^q_+}(P_{\text{N}} - \rho_{\text{N}} \circ (H_{\text{N}}^{\top}v + b_{\text{N}} + e \circ U_{\text{N},k})) \\ \widetilde{\Psi}_3(v,P) &= P_{\text{T}} - \text{proj}_{\widehat{\mathbf{D}}(P_{\text{N}},U_{\text{N}})}(P_{\text{T}} - \rho_{\text{T}} \circ (H_{\text{T}}^{\top}v + b_{\text{T}})) \end{aligned} \tag{16.23}$$ {eq:GAC-L5}

Chain rule

$$\begin{aligned} \partial_v\widetilde{\Psi}_{2,3}(v,P) &= \partial_v\Psi_{2,3}(v, H^{\top}v + b, P) \\ &= H_{\text{N}}^{\top}\partial_{U_{\text{N}}}\Phi_{2,3}(H^{\top}v + b, P) + H_{\text{T}}^{\top}\partial_{U_{\text{T}}}\Phi_{2,3}(H^{\top}v + b, P) \end{aligned} \tag{16.24}$$ {eq:chainrule1}

$$H(v,P) = \begin{bmatrix} -\widehat{M} & H_{\text{N}} & H_{\text{T}} \\ H_{\text{N}}^{\top}\partial_{U_{\text{N}}}\Phi_2(H^{\top}v + b, P) & \partial_{P_{\text{N}}}\Phi_2(H^{\top}v + b, P) & 0 \\ H_{\text{N}}^{\top}\partial_{U_{\text{N}}}\Phi_3(H^{\top}v + b, P) \\ \quad + H_{\text{T}}^{\top}\partial_{U_{\text{T}}}\Phi_3(H^{\top}v + b, P) & \partial_{P_{\text{N}}}\Phi_3(H^{\top}v + b, P) & \partial_{P_{\text{T}}}\Phi_3(H^{\top}v + b, P) \end{bmatrix} \tag{16.25}$$ {eq:GAC-L6}

discussion

- Formulae has to be checked carefully
- I do not known if there an interest in the simplification. With sparse matrices, it is perhaps easier to deal with (16.21)

References

[1] O. Brüls and A. Cardona. On the use of Lie group time integrators in multibody dynamics. *ASME Journal of Computational and Nonlinear Dynamics*, 5(031002), 2010.

[2] Arieh Iserles, Hans Z Munthe-Kaas, Syvert P Nørsett, and Antonella Zanna. Lie-group methods. *Acta Numerica 2000*, 9:215–365, 2000.

[3] B. Owren and B. Welfert. The Newton iteration on Lie groups. *BIT*, 40(1):121–145, 2000.

[4] V. S. Varadarajan. *Lie Groups, Lie Algebras, and Their Representations*. Graduate Texts in Mathematics 102. Springer-Verlag New York, 1 edition, 1984.