Compliant Reference

Maxime Tournier

August 27, 2014

Abstract

Quick notes about the mathematical framework used in Compliant. We review the time-stepping scheme, kinematic constraint handling, potential forces as compliant constraints, and KKT systems.

TODO

- Code cheat sheet
- finish sections :-)
- references
- wikipedia links
- introduction ?

1 Lagrangian Dynamics

Given an n-dimensional state space Q, we consider a Lagrangian $\mathcal{L}$ defined as usual:

$$\mathcal{L}(q, v) = \frac{1}{2} v^T M v - V(q) \tag{1}$$

where $V(q)$ a the potential energy of the form:

$$V(q) = \frac{1}{2} ||g(q)||_N^2 \tag{2}$$

for a matrix norm N in the suitable deformation space $Im(g)$. To be fully general, we include a Rayleigh dissipation function $\frac{1}{2} v^T D v$, where D is PSD. The Euler-Lagrange equations of motions are, in this case:

$$\frac{d}{dt} \frac{\partial \mathcal{L}}{\partial v} = \frac{\partial \mathcal{L}}{\partial q} - \frac{\partial D}{\partial v} \tag{3}$$

For simplicity, we restrict to the case where M is constant, which yields:

$$M \dot{v} = -\nabla_q V - D v \tag{4}$$

2 Time Stepping

We now discretize time using a fixed time step h. We use forward differences for $\dot{v}$ as follows:

$$h\,\dot{v}_{k+1} \approx v_{k+1} - v_k \tag{5}$$

Equation (4) becomes:

$$Mv_{k+1} = Mv_k - h\left(\nabla_q V_{k+1} + Dv_{k+1}\right) \tag{6}$$

Or, calling $p_k = Mv_k$ the momentum of the system and $f_k = -\nabla_q V_k$ the potential forces:

$$\left(M + h\,D\right)v_{k+1} = p_k + hf_{k+1} \tag{7}$$

In order to control how implicit our integrator is on forces, we introduced a blending parameter $\alpha \in [0, 1]$ as follows:

$$\left(M + h\alpha D\right)v_{k+1} = p_k - h(1-\alpha)Dv_k + h\left((1-\alpha)f_k + \alpha f_{k+1}\right) \tag{8}$$

We linearize the above using the Hessian of the potential energy:

$$\nabla_q V_{k+1} \approx \nabla_q V_k + \nabla_q^2 V_k\left(q_{k+1} - q_k\right) \tag{9}$$

so that:

$$(1-\alpha)f_k + \alpha f_{k+1} = f_k - \alpha\nabla_q^2 V_k\left(q_{k+1} - q_k\right) \tag{10}$$

Finally, we use the following position time-stepping scheme:

$$q_{k+1} = q_k + h\left((1-\beta)v_k + \beta v_{k+1}\right) \tag{11}$$

where $\beta \in [0, 1]$ is again a blending parameter. Letting the stiffness matrix be $K = \nabla_q^2 V_k$ [1] and $d_k = -Dv_k$ be the damping forces, we put everything together as:

$$\left(M + h\alpha D + h^2\alpha\beta K\right)v_{k+1} = p_k + h\left(f_k + (1-\alpha)d_k\right) - \alpha(1-\beta)h^2 K v_k \tag{12}$$

(and yes, this is very ugly). Fortunately, everyone would use the much friendlier, fully implicit scheme $\alpha = \beta = 1$ as follows:

$$\left(M + hD + h^2 K\right)v_{k+1} = p_k + hf_k \tag{13}$$

3 Response Matrix, Net Momentum

From now on, we will refer to the *response matrix* defined as follows:

$$W = \left(M + h\alpha D + h^2\alpha\beta K\right) \tag{14}$$

[1] This is the opposite of the stiffness matrix as defined in SOFA, *i.e.* $\nabla f(q) = -\nabla_q^2 V$. I find it easier to work with PSD matrices.

We will also refer to the *net momentum* of the system at time step k:

$$c_k = p_k + h\left(f_k + (1-\alpha)d_k\right) - \alpha(1-\beta)h^2 K v_k \tag{15}$$

The time-stepping scheme (12) thus involves solving the following linear system:

$$W v_{k+1} = c_k \tag{16}$$

The numerical solvers for time-stepping will be described in section 11.

4 Constraints

We now introduce holonomic constraints of the form:

$$g(q) = 0 \tag{17}$$

where g again maps kinematic DOFs to a suitable deformation space $Im(g)$. Such constraints are satisfied by introducing constraint forces $J^T\lambda$, where $J = dg$ is the constraint Jacobian matrix, and λ are the *Lagrange multipliers*:

$$M\dot{v} = -\nabla_q V - Dv + J^T\lambda \tag{18}$$
$$g(q) = 0 \tag{19}$$

Again, we discretize time and $\dot{v}$ as follows:

$$M v_{k+1} = p_k + h\left(f_{k+1} + d_{k+1} + J_{k+1}^T\lambda_{k+1}\right) \tag{20}$$
$$g\left(q_{k+1}\right) = 0 \tag{21}$$

And again:

$$g(q_{k+1}) = g_k + hJ_k\left((1-\beta)v_k + \beta v_{k+1}\right) \tag{22}$$

At this point we become lazy so we approximate constraints as *affine* functions, meaning that $J_{k+1} \approx J_k$, otherwise computing the constraint Hessian d^2g would be too costly, and would result in a non-linear system to solve anyways (*i.e.* with terms involving $v_{k+1}^T d^2 g_k \lambda_{k+1}$). As we will see later, such approximation is consistent with our treatment of potential forces as compliant constraints. The time-discrete system is then:

$$M v_{k+1} = p_k + h\left(f_{k+1} + d_{k+1} + J_k^T\lambda_{k+1}\right) \tag{23}$$
$$J_k v_{k+1} = -\frac{g_k}{\beta h} - \frac{1-\beta}{\beta} J_k v_k \tag{24}$$

At this point, we *could* introduce an implicit blending between λ_k and λ_{k+1}, but this would result in unneeded complication as λ_{k+1} would need to be computed anyways. The blended force would then be:

$$J_k^T\left((1-\alpha)\lambda_k + \alpha\lambda_{k+1}\right)$$

which simply offsets λ_{k+1} with respect to λ_k. We will thus happily ignore such blending. The final linear system to solve (for v_{k+1} and λ_{k+1}) becomes:

$$W v_{k+1} - J_k^T \lambda_{k+1} = c_k \tag{25}$$

$$J_k v_{k+1} = -b_k \tag{26}$$

where $b_k = \frac{g_k}{\beta h} + \frac{1-\beta}{\beta} J_k v_k$. Before leaving, note the sneaky accounting of h inside λ_{k+1}.

5 Compliance

We will now establish a connection between elastic forces and constraint forces through a compliance matrix. Remember that we defined potential energy as:

$$V(q) = \frac{1}{2} ||g(q)||_N^2 \tag{27}$$

where g maps kinematic DOFs to an appropriate deformation space $Im(g)$. This means the potential forces are:

$$f = -\nabla_q V = J^T N g(q) \tag{28}$$

Now, the Hessian or stiffness matrix is :

$$K = \nabla_q^2 V = dJ^T N g(q) + J^T N J \tag{29}$$

As in section 4, we are too lazy for computing second derivatives so we approximate deformation mappings as affine maps, meaning that $dJ \approx 0$. We are left with the following response matrix (14):

$$W = \left(M + h\alpha D + h^2 \alpha\beta J_k^T N J_k\right) \tag{30}$$

The net momentum (15) becomes:

$$c_k = p_k + h(1-\alpha)d_k - hJ_k^T N \left(g_k + h\alpha(1-\beta)J_k v_k\right) \tag{31}$$

If we write the potential force as $J_k^T \lambda_{k+1}$, we have:

$$\lambda_{k+1} = -N \left(h^2 \alpha\beta J_k v_{k+1} + h g_k + h^2 \alpha(1-\beta) J_k v_k\right) \tag{32}$$

It turns out that this system can be rewritten as a *larger* system:

$$\begin{pmatrix} M + h\alpha D & -J_k \\ J_k & \frac{N^{-1}}{h^2\alpha\beta} \end{pmatrix} \begin{pmatrix} v_{k+1} \\ \lambda_{k+1} \end{pmatrix} = \begin{pmatrix} p_k + h(1-\alpha)d_k \\ -\frac{g_k}{h\alpha\beta} - \frac{(1-\beta)}{\beta} J_k v_k \end{pmatrix} \tag{33}$$

(pffew !) One can notice that this system is *almost* exactly the one we obtained with kinematic constraints in (25), with $\alpha = 1$, with the exception of the N^{-1} term in the $(2,2)$ block. We call matrix $C = N^{-1}$ the *compliance* matrix. We see that kinematic constraints simply correspond to a zero compliance matrix, *i.e.* infinite stiffness, as one would intuitively expect. It can be checked that taking α into account for constraint forces produces the same system with $C = 0$. (TODO) We now recall the main results for the unified elastic/constraint treatment:

Linear System

$$\begin{pmatrix} W & -J \\ -J & -\frac{C}{h^2\alpha\beta} \end{pmatrix} \begin{pmatrix} v \\ \lambda \end{pmatrix} = \begin{pmatrix} c_k \\ b_k \end{pmatrix} \tag{34}$$

Response Matrix

$$W = M + h\alpha D \tag{35}$$

Net Momentum

$$c_k = p_k + h(1-\alpha)d_k \tag{36}$$

Constraint Value

$$b_k = \frac{g_k}{h\alpha\beta} + \frac{(1-\beta)}{\beta} J_k v_k \tag{37}$$

TODO pourquoi c'est cool: transition seamless entre contraintes et elasticite, traitement unifié contraintes/elasticité, conditionnement pour les sytemes très raides. par contre ca fait des systèmes plus gros.

6 KKT Systems, Compliance and Relaxation

At this point, it is probably a good idea to introduce a few notions on saddle-point (or KKT) systems. Such systems are (for now, linear) systems of the form:

$$\begin{pmatrix} Q & -A^T \\ -A & 0 \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} c \\ b \end{pmatrix} \tag{38}$$

KKT systems typically arise from the Karush-Kuhn-Tucker (hence the name) optimality conditions of constrained optimization problems. For instance, the KKT system (38) corresponds to the following equality-constrained Quadratic Program (QP):

$$\underset{Ax+b=0}{\operatorname{argmin}} \quad \frac{1}{2} x^T Q x - c^T x \tag{39}$$

The KKT system summarizes the optimality conditions for the following unconstrained function (also called the *Lagrangian*, this time in the context of optimization):

$$\mathcal{L}(x,\lambda) \quad = \quad \frac{1}{2} x^T Q x - c^T x \quad - \quad (Ax+b)^T \lambda \tag{40}$$

and whose critical points (in fact, saddle-points) solve the constrained problem (39). As one can see, the constrained integrator (25) is an example of such KKT systems:

$$\begin{pmatrix} W & -J \\ -J & 0 \end{pmatrix} \begin{pmatrix} v \\ \lambda \end{pmatrix} = \begin{pmatrix} c \\ b \end{pmatrix} \tag{41}$$

where we dropped subscripts for the sake of readability. As symmetric *indefinite* systems, they tend to be more difficult to solve than positive definite ones: CG and Cholesky LDL^T factorizations may breakdown on such systems. However, when the $(1,1)$ block (here, matrix Q) is invertible, one can use *pivoting* to obtain the following smaller system:

$$x = Q^{-1}\left(c - A^T\lambda\right) \tag{42}$$

$$-Ax = AQ^{-1}A^T\lambda - AQ^{-1}c = b \tag{43}$$

$$AQ^{-1}A^T\lambda = b - AQ^{-1}c \tag{44}$$

This smaller system (44) is known as the *Schur* complement of the KKT system. It is always SPD as long as A^T is full column-rank, but requires to invert Q which might be costly in practice. The Schur complement system also corresponds to an (unconstrained) optimization problem:

$$\underset{\lambda}{\text{argmin}} \quad \frac{1}{2}\lambda^T S\lambda - s^T\lambda \tag{45}$$

where $S = AQ^{-1}A^T$ is the Schur complement, and $s = b + AQ^{-1}c$. The minimized quantity has a physical interpretation: typically, holonomic constraints minimize the total kinetic energy. If we now consider systems with compliance, such as:

$$\begin{pmatrix} Q & -A^T \\ -A & -D \end{pmatrix}\begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} c \\ b \end{pmatrix} \tag{46}$$

we see that they correspond to the following Lagrangian:

$$\mathcal{L}(x,\lambda) \quad = \quad \frac{1}{2}x^T Qx - c^T x \quad - \quad (Ax+b)^T\lambda \quad - \quad \frac{1}{2}\lambda^T D\lambda \tag{47}$$

One can check that the Schur complement becomes $S + D$, and corresponds to:

$$\underset{\lambda}{\text{argmin}} \quad \frac{1}{2}\lambda^T S\lambda - s^T\lambda \quad + \quad \frac{1}{2}\lambda^T D\lambda \tag{48}$$

We see that D acts as a form of numerical damping on the resolution of constraints, by biasing the solution of the constrained system towards zero. It is a form of *constraint relaxation*. Physically, elastic forces minimize a mix between the kinetic energy of the system and the D-norm of the forces λ. Under their most general form, KKT systems include unilateral and complementarity constraints. We will develop these in more details in section 8.

7 Geometric Stiffness

For both constraint and elastic forces, we happily neglected second-order derivatives of the deformation mapping g. In the case of elastic forces, this resulted in a nice stiffness matrix of the form:

$$\nabla_q^2 V(q) \approx J^T N J = K \tag{49}$$

which enabled us to treat elastic forces as a compliant kinematic constraints. But what about the second order terms $\tilde{K} = (dJ)^T N g(q)$? First of all, when the configuration space is a vector space Q and g is $\mathcal{C}^2$-continuous, the Schwarz' theorem ensures that $\nabla_q^2 V$ is symmetric, hence so is $\tilde{K}$. We call $\tilde{K}$ the *geometric stiffness*, as it is induced by the variation of the deformation mapping g. We do not know whether in the general case, it is possible to factor *both* K and $\tilde{K}$ as:

$$\nabla_q^2 V(q) = K + \tilde{K} \stackrel{?}{=} J^T \left(N + \tilde{N} \right) J \tag{50}$$

even though some specific examples exist (*e.g.* mass spring systems). Of course, it is always possible to apply a Cholesky factorization directly on $\nabla_q^2 V(q)$ as:

$$\nabla_q^2 V(q) = L D L^T \tag{51}$$

and get back on our feet, but this would be highly inefficient in practice. Therefore, unless an ad-hoc derivation provides the needed factorization, we are left with only one alternative: treat the geometric stiffness as a regular stiffness instead of compliance.

8 Unilateral Constraints

We now consider *unilateral constraints* (inequality) instead of bilateral ones:

$$g(q) \geq 0 \tag{52}$$

Examples of such constraints include non-penetration constraints, or angular limits for an articulated rigid body. Just like in the bilateral case, the constraints are enforced by the addition of constraint forces $J^T \lambda$, satisfying velocity constraints of the form:

$$J v_{k+1} \geq -b_k \tag{53}$$

obtained by differentiation of (52), according to the time-stepping scheme. Furthermore, reaction forces must satisfy additional requirements known as the Signorini conditions:

- constraint forces must be repulsive: $\lambda_{k+1} \geq 0$
- constraint forces don't act when the constraint is inactive, and conversely:

$$J_k v_{k+1} > -b_k \Rightarrow \lambda_{k+1} = 0, \quad \lambda_{k+1} > 0 \Rightarrow J_k v_{k+1} = -b_k$$

Intuitively, a non-penetration contact force is not allowed to push bodies further apart when they are already separating. The requirements on constraint forces are summarized by the following *complementarity* constraint:

$$0 \leq J_k v_{k+1} + b_k \perp \lambda_{k+1} \geq 0 \tag{54}$$

The time-stepping scheme with constraint forces is, as before:

$$Wv_{k+1} - J_k^T \lambda_{k+1} = c_k \tag{55}$$

It turns out that the complementarity constraints (54) together with time-stepping equation (55) form the KKT conditions of the following *inequality-constrained* Quadratic Program:

$$v_{k+1} = \underset{J_k v + b_k \geq 0}{\operatorname{argmin}} \quad \frac{1}{2} v^T W v - c_k^T v \tag{56}$$

Therefore, time-stepping in the presence of unilateral constraints can be readily solved by a general QP solver. As in the bilateral case, when W is easily invertible, it is possible to compute the Schur complement in order to obtain an equivalent, but smaller problem:

$$0 \leq JW^{-1}J^T \lambda_{k+1} + b_k - AW^{-1}c_k \perp \lambda_{k+1} \geq 0 \tag{57}$$

(TODO: check formula) Such problem is known as a *Linear Complementarity Problem* (LCP) and can be solved by various algorithms, some of which will be presented in section 11.

9 Stabilization

10 Restitution

TODO: velocity constraints for contact with restitution (rigid), mention Generalized Reflections

11 Numerical Solvers

TODO: CG, Cholesky, Minres, GS, PGS, Sequential Impulses

12 Code Cheat Sheet

A quick reminder of who does what where in the code:

12.1 CompliantImplicitSolver

- Performs KKT system assembly
- Builds right-hand sides (correction/dyamics) computation
- Integrate system solution, update graph and stuff

12.2 Right-hand side computation

Lots of cases:

- Bilateral (default), unilateral, friction constraint type
- Elastic constraints, kinematic constraints (can be stabilized) (default), restitution constraints, . . . ?

A constraint has a BaseConstraintValue constraint value (if not, the default is assigned) that produces a correct constraint violation term (the right-hand side for constraints in the KKT system) depending on the correction/dynamics phase.

Elastic constraints have zero rhs during correction, and -error / dt during dynamics.

Stabilized constraints (Stablization component) correct the error during stabilization, but have zero rhs during dynamics (holonomic kinematic constraints)

Restitution constraints: stabilize when velocity is small and when the constraint is not violated, otherwise flip relative velocity.

Depending on the type (bilateral, unilateral, friction), the constraint also has an associated Projector, that you can find in the KKT system.

12.3 KKTSolver

Solves a KKT system (AssemledSystem), for both correction/dynamics cases.