A Fast NCP Solver for Large Rigid-Body Problems with Contacts, Friction, and Joints

Alessandro Tasora[1] and Mihai Anitescu[2]

[1] Università degli Studi di Parma, Dipartimento di Ingegneria Industriale, 43100 Parma, Italy tasora@ied.unipr.it

[2] Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439, USA, anitescu@mcs.anl.gov

The simulation of multibody systems with rigid contacts entails the solution of nonsmooth equations of motion. The dynamics is nonsmooth because of the discontinuous nature of noninterpenetration, collision, and adhesion constraints.

We propose a solver that is able to handle the simulation of multibody systems of vast complexity, with more than 100,000 colliding rigid bodies. The huge number of nonsmooth constraints arising from unilateral contacts with friction gives rise to a nonlinear complementarity problem (NCP), which we solve by means of a high-performance iterative method.

The method has been implemented as a high-performance software library, written in C++. Complex simulation scenarios involving thousands of moving parts have been extensively tested, showing a remarkable performance of the numerical scheme compared to other algorithms.

1 Introduction

When simulating nonsmooth systems affected by discontinuities, such as those imposed by frictionless or frictional contacts, the direct application of numerical methods for ordinary differential equations and differential algebraic equations can be hard, if not impossible [1]. Some naive numerical approaches try to circumvent the complexity of nonsmooth dynamics by introducing a smooth and stiff approximation of the problem, however at a cost of requiring prohibitively small time steps to achieve stability.

Other approaches try to solve the nonsmooth integral by piecewise integration, but such methods cannot be applied to systems with a large number of contacts because stop-and-restart schemes can easily enter infinite or ill-posed loops.

These limitations motivate our research on innovative methods that deal directly with the discontinuities of the equations of motion. Different from the case of smooth dynamics, where for the past decade efficient numerical methods have been able to compute solutions in linear time [2], the case of nonsmooth dynamics introduces many numerical difficulties that are still under active and fertile investigation in the mathematical community.

Among the various potential applications of nonsmooth dynamics are granular flows, rock soil dynamics, refueling of pebble-bed nuclear reactors, interaction between robot and large environments, and real-time simulations for augmented reality problems that are challenging or still impossible to simulate because of the severe computational requirements.

We formulate the problem in terms of differential inclusions over the speed-impulses space [11], a fertile approach that expresses the problem as a differential complementarity problem (DCP), requiring the solution of a single NCP per time step [7]. Posing the problem in terms of speeds and impulses rather than in terms of accelerations and reaction forces has various advantages. However, here we do not enter into the details of the integration scheme, except to mention that our focus is a recently-proposed optimization-based scheme [9], for which details will be presented as needed for setting up the NCPs. For additional information about such schemes the interested reader can examine the bibliography [3]. Since the main bottleneck is by far represented by the NCP, which must be solved at each step, we will focus on this part.

The difficulty of this process rests with the fact that the more the contacts and collisions, the more the complementarity constraints, and the higher the computational overhead [4]. Previous efforts in solving NCP problems arising in nonsmooth dynamics were based on approximating them by Linear Complementarity Problems (LCPs), since solvers for LCPs are available, such as the simplex-based pivoting methods of Lemke or Dantzing; nonetheless these pivoting methods suffer exponential explosion and cannot practically handle systems with more than a few hundred contacts [13].

A more promising way to handle NCP problems, followed in this work, is iterative methods, in particular, the projected-SOR and projected-Gauss-Seidel [14][15]. Although the idea of solving frictional problems by means of iterative schemes with projections is not new [5], our iterative method is based on an improved projected-block-symmetric-overrelaxed scheme which aims at the best computational performance when applied to a large number of contacts with friction. With some requirements on the spectral radius of the iteration matrix, and since the scheme from [9] results in convex subproblems, we can prove that, when applied to our formulation, the iteration is a fixed-point contraction that converges monotonically to the NCP solution [6].

2 Implementation

Let us introduce the position vector $\mathbf{q} \in \mathbb{R}^{n_q}$, the speed vector $\dot{\mathbf{q}} \in \mathbb{R}^{n_q}$, the mass matrix $[\mathrm{M}] \in \mathbb{R}^{n_q \times n_q}$, and the sum of external applied forces and inertial forces $\mathbf{f}_t(t^{(l)}, \mathbf{q}^{(l)}, \dot{\mathbf{q}}^{(l)}) \in \mathbb{R}^{n_q}$.

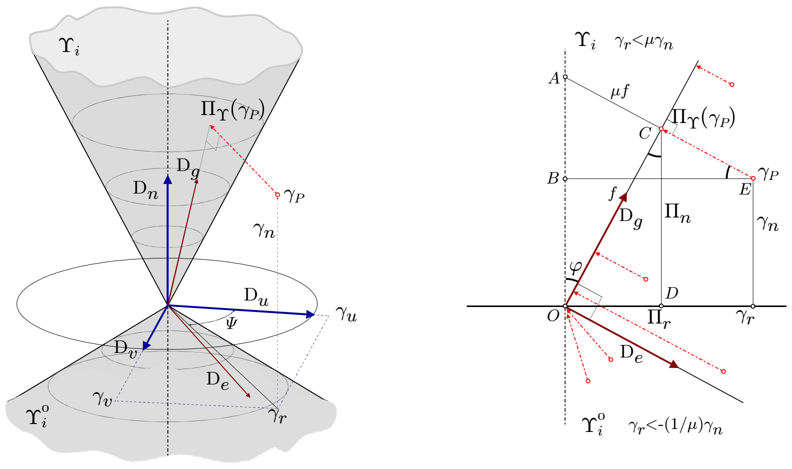


Fig. 1. Nonexpansive projection on the ith convex friction cone.

The vector of Lagrangian multipliers, representing the unknown impulses in constraints and in contact points, is the vector $\gamma \in \mathbb{R}^{n_c}$, which can be partitioned into p subvectors $\gamma_i \in \mathbb{R}^{n_i}$ such that $\gamma = \left\{\gamma_1^T, \gamma_2^T, \ldots, \gamma_p^T\right\}^T$.

In detail, for each bilateral scalar constraint there is a corresponding one-dimensional impulse $\gamma_i \in \mathbb{R}^1, \forall i \in \mathcal{G}_C$, along with the corresponding bilateral constraint equation $\mathbf{C}_i(\mathbf{q}, t) = \mathbf{0} \in \mathbb{R}^1$ and Jacobian $[\mathrm{C_q}(\mathbf{q}, \mathrm{t})]_i = \nabla_q \mathbf{C}_i(\mathbf{q}, t)^T \in \mathbb{R}^{n_q \times 1}$.

Similarly, for each unilateral constraint we have a corresponding one-dimensional impulse $\gamma_i \in \mathbb{R}^1, \forall i \in \mathcal{G}_D$, along with the corresponding bilateral constraint equation $\mathbf{D}_i(\mathbf{q}, t) > \mathbf{0} \in \mathbb{R}^1$ and Jacobian $[\mathrm{D_q}(\mathbf{q}, \mathrm{t})]_i = \nabla_q \mathbf{D}_i(\mathbf{q}, t)^T \in \mathbb{R}^{n_q \times 1}$.

If we also introduce contact constraints with nonlinear friction cones in 3D space, the γ vector includes the vectors of unknown normal and tangential impulses, that is, the three-dimensional $\gamma_i \in \mathbb{R}^3, \forall i \in \mathcal{G}_F$, along with the corresponding equations $\mathbf{F}_i(\mathbf{q}) = \mathbf{0} \in \mathbb{R}^3$ and Jacobian $[\mathrm{F_q}(\mathbf{q})]_i = \nabla_q \mathbf{F}_i(\mathbf{q}, t)^T \in \mathbb{R}^{n_q \times 3}$.

These vectors and matrices have a block structure coresponding to the three components n, u, v (normal, u-tangential, v-tangential), that is, $\gamma_i = \{\gamma_{i,n}, \gamma_{i,u}, \gamma_{i,v}\}^T$ and $[\mathrm{F_q}]_i = \left[F_{q_{i,n}}^T | F_{q_{i,u}}^T | F_{q_{i,v}}^T\right]^T$. Each contact point has a friction coefficient $\mu_i, \forall i \in \mathcal{G}_F$.

With a time step h and a stabilization coefficient $0 < K < 1$, we expand the original first-order DCP formulation of [7] as follows.

$$
\begin{aligned}
M(\dot{\mathbf{q}}^{(l+1)} - \dot{\mathbf{q}}^{l}) &= \sum_{i \in \mathcal{G}_C} \left([\mathrm{C_q}]_i^T \gamma_i \right) + \sum_{i \in \mathcal{G}_D} \left([\mathrm{D_q}]_i^T \gamma_i \right) + \sum_{i \in \mathcal{G}_F} \left([\mathrm{F_q}]_i^T \gamma_i \right) + \\
&\quad + h \mathbf{f}_t(t^{(l)}, \mathbf{q}^{(l)}, \dot{\mathbf{q}}^{(l)}) && (1) \\
0 &= \frac{K}{h} \mathbf{C}_i(q^{(l)}, \dot{\mathbf{q}}^{(l)}, t) + \frac{\partial \mathbf{C}_i}{\partial t} + [\mathrm{C_q}]_i \dot{\mathbf{q}}^{(l+1)}, \quad i \in \mathcal{G}_C && (2) \\
0 &\leq \frac{K}{h} \mathbf{D}_i(q^{(l)}) + [\mathrm{D_q}]_i \dot{\mathbf{q}}^{(l+1)} \quad \perp \gamma_i \geq 0, \quad i \in \mathcal{G}_D && (3) \\
0 &\leq \frac{K}{h} \mathbf{F}_{i,n}(q^{(l)}) + [\mathrm{F_q}]_{i,n} \dot{\mathbf{q}}^{(l+1)} \quad \perp \gamma_{i,n} \geq 0, \quad i \in \mathcal{G}_F && (4) \\
(\gamma_{i,u}, \gamma_{i,v})_{i \in \mathcal{G}_F} &= \mathrm{argmin}_{\mu_i \gamma_n \geq \sqrt{(\gamma_u)^2 + (\gamma_v)^2}} \left[\dot{\mathbf{q}}^T (\gamma_{i,u} [\mathrm{F_q}]_{i,u}^T + \gamma_{i,v} [\mathrm{F_q}]_{i,v}^T) \right] && (5) \\
\mathbf{q}^{(l+1)} - \mathbf{q}^{(l)} &= h \dot{\mathbf{q}}^{(l+1)}. && (6)
\end{aligned}
$$

The DCP problem above is affected by complementarity constraints (3) and (4), where only one of the inequalities can hold at a single time. An alternative way to write complementarity constraints $\mathbf{a} > 0 \perp \mathbf{b} > 0$ is by inner product: $\mathbf{a} > 0, \mathbf{b} > 0, \langle \mathbf{a}, \mathbf{b} \rangle = 0$.

The extremely nonlinear nature of complementarity conditions is the main cause of computational overhead. Moreover, the introduction of the original Coloumb model in the DCP leads to an NCP problem that be nonconvex under some circumstances [8]. A possible workaround is to relax the Coloumb original assumptions and to replace (4) with the following:

$$
\begin{aligned}
&0 \leq \frac{K}{h} \mathbf{F}_{i,n}(q^{(l)}) + [\mathrm{F_q}]_{i,n} \dot{\mathbf{q}}^{(l+1)} - \mu_i \sqrt{([\mathrm{F_q}]_{i,u} \dot{\mathbf{q}})^2 + ([\mathrm{F_q}]_{i,v} \dot{\mathbf{q}})^2} \ \perp \\
&0 \leq \gamma_{i,n}, \quad i \in \mathcal{G}_F.
\end{aligned} \qquad (7)
$$

This modification to the original scheme has negligible effects for small values of $\dot{\mathbf{q}}$ or μ, converges to the same class of weak solutions as the original scheme (described in [10, 3] for $h \to 0$, and has the positive effect of making the NCP problem always convex [9]. The resulting NCP is also a convex cone complementarity problem (CCP), hence a special case of convex VI, variational inequality.

For a more compact notation, we assume that constraints are ordered such that $i \in \mathcal{G}_C < i \in \mathcal{G}_D < i \in \mathcal{G}_F$. Hence, we can introduce the following vectors and matrices:

$$
\begin{aligned}
\gamma_E &= \left\{ \gamma_C^T | \gamma_D^T | \gamma_T^T \right\}^T \quad \in \mathbb{R}^{n_c} && (8) \\
[\mathrm{E_q}] &= \left[[\mathrm{C_q}]^T | [\mathrm{D_q}]^T | [\mathrm{F_q}]^T \right]^T \quad \in \mathbb{R}^{n_q \times n_c} && (9) \\
\mathbf{b}_E &= \left\{ \mathbf{b}_C^T | \mathbf{b}_D^T | \mathbf{b}_T^T \right\}^T \quad \in \mathbb{R}^{n_c}, && (10)
\end{aligned}
$$

where, if we assume m_C bilateral constraints, m_D unilateral constraints, and m_F frictional contacts, we have the following:

$$\mathbf{b}_C = \left\{ -\frac{K}{h}C_1 - \frac{\partial C_1}{\partial t}, -\frac{K}{h}C_2 - \frac{\partial C_2}{\partial t}, \ldots, -\frac{K}{h}C_{m_C} - \frac{\partial C_{m_C}}{\partial t} \right\}^T \tag{11}$$

$$\mathbf{b}_D = \left\{ -\frac{K}{h}D_1, -\frac{K}{h}D_2, \ldots, -\frac{K}{h}D_{m_D} \right\}^T \tag{12}$$

$$\mathbf{b}_F = \left\{ -\frac{K}{h}F_{1,n}, 0, 0, -\frac{K}{h}F_{2,n}, 0, 0, \ldots, -\frac{K}{h}F_{m_F,n}, 0, 0 \right\}^T . \tag{13}$$

We now introduce $\mathbf{k} = [\mathrm{M}]\dot{\mathbf{q}}^l + h\mathbf{f}_t$, $[\mathrm{N}] = [\mathrm{E_q}][\mathrm{M}]^{-1}[\mathrm{E_q}]^T$ and $\mathbf{r} = [\mathrm{E_q}][\mathrm{M}]^{-1}\mathbf{k} - \mathbf{b}_E$. We also introduce the convex cones Υ_i, defined as follows

$$\Upsilon_i = \begin{cases} \mathbb{R}, & i \in \mathcal{G}_C \\ \mathbb{R}^+, & i \in \mathcal{G}_D \\ \left\{ (\gamma_n, \gamma_u, \gamma_v) \in \mathbb{R} \mid \mu^i \gamma_n \geq \sqrt{\gamma_u^2 + \gamma_v^2} \right\}, & i \in \mathcal{G}_F . \end{cases} \tag{14}$$

Their polar cones $\Lambda_i = \Upsilon_i^\circ$, where we denote by $^\circ$ the polar set, can be immediately calculated as

$$-\Lambda_i = \begin{cases} 0, & i \in \mathcal{G}_C \\ \mathbb{R}^+, & i \in \mathcal{G}_D \\ \left\{ (s_n, s_u, s_v) \in \mathbb{R} \mid s_n \geq \mu^i \sqrt{s_u^2 + s_v^2} \right\}, & i \in \mathcal{G}_F . \end{cases} \tag{15}$$

Denoting the total index set $\mathcal{G} = \mathcal{G}_C \cup \mathcal{G}_D \cup \mathcal{G}_F$, the direct sum of these cones $\Upsilon = \bigoplus_{i \in \mathcal{G}} \Upsilon_i$, and $\Lambda = \bigoplus_{i \in \mathcal{G}} \Lambda_i = \Upsilon^o$, and using equations (1)- (15) we obtain a canonical CCP (for full details, see [6]):

$$([\mathrm{N}]\gamma_E + \mathbf{r}) \in -\Lambda, \quad \gamma_E \in \Upsilon, \quad \langle ([\mathrm{N}]\gamma_E + \mathbf{r}), \gamma_E \rangle = 0, \tag{16}$$

which can be solved for unknowns γ_E. One can then obtain the unknown speeds with the simple formula

$$\dot{\mathbf{q}} = [\mathrm{M}]^{-1}(\mathbf{k} + [\mathrm{E_q}]^T \gamma_E). \tag{17}$$

To solve (16), which represents the most relevant computational burden, we propose an iterative scheme. We make the following assumptions:

A1 The matrix [N] of the NCP problem is symmetric and positive semidefinite.
A2 There exists a positive number $\alpha > 0$ such that, at any iteration r, $r = 0, 1, 2 \ldots$, we have that $B^r \succ \alpha I$.
A3 There exists a positive number $\beta > 0$ such that, at any iteration r, $r = 0, 1, 2 \ldots$, we have that $(\gamma^{r+1} - \gamma^r)^T \left((\lambda\omega[\mathrm{B}]^r)^{-1} - \frac{[\mathrm{N}]}{2} \right) (\gamma^{r+1} - \gamma^r) \geq \beta \left\| \gamma^{r+1} - \gamma^r \right\|^2$.

With these assumptions, we can prove the convergence to the NCP solution by means of the fixed-point iteration

$$\gamma_E^{r+1} = \Pi_{\Upsilon^+} \left(\gamma_E^r - \omega [\mathrm{B}^\mathrm{r}] \left([\mathrm{N}] \gamma_E^r + \mathbf{r} \right) \right) , \tag{18}$$

where we introduced the nonsmooth projection mapping $\Pi_{\Upsilon_i}(\cdot) : \mathbb{R}^{n_c} \rightarrow \mathbb{R}^{n_c}$ onto the boundary of a n_ith dimensional convex cone Υ. See [6] for a detailed proof.

For the iteration matrix $[\mathrm{B}^\mathrm{r}]$ we use the following block-diagonal structure:

$$[\mathrm{B}^\mathrm{r}] = \begin{bmatrix} \eta_1 I_{n_1} & 0 & \cdots & 0 \\ 0 & \eta_2 I_{n_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \eta_{n_i} I_{n_{n_i}} \end{bmatrix} . \tag{19}$$

The complete projection operator $\Pi_{\Upsilon^+} : \mathbb{R}^{n_c} \rightarrow \mathbb{R}^{n_c}$ can be expressed as

$$\Pi_{\Upsilon^+} = \left\{ \Pi_{\Upsilon_1}(\gamma_1)^T, \ldots \Pi_{\Upsilon_p}(\gamma_p)^T \right\}^T .$$

Each single $\Pi_{\Upsilon_i}(\cdot)$ projection operator behaves as $\Pi_{\Upsilon}(\gamma_i) = \mathrm{argmin}_{\zeta \in \Upsilon_i} || \gamma_i - \zeta ||$ in order to be globally nonexpansive for projection on convex subspace Υ_i.

For the multipliers introduced by friction constraints, it is enough to introduce a straightforward mapping $\Pi_{\Upsilon_i}(\cdot) : \mathbb{R}^3 \rightarrow \mathbb{R}^3, i \in \mathcal{G}_F$, to be applied multiple times on all the triplets of contact multipliers $\gamma_i, i \in \mathcal{G}_F$. Such a projection can be implemented as depicted in Fig.1, where three subcases are detected:

- When γ_i is inside the Υ_i cone, the vector is left untouched.
- When γ_i is inside the Υ_i^o polar cone, it maps to the origin $\{0, 0, 0\}$.
- Otherwise γ_i is projected to the nearest point on the Υ_i cone.

One can verify that such a mapping exhibits $||\Pi_{\Upsilon}(a) - \Pi_{\Upsilon}(b)|| \leq ||a - b||$. Hence the nonexpansive property holds.

This orthogonal projection onto the surface of the cone can be easily obtained by means of a few operations. First we compute $\gamma_r = \sqrt{\gamma_u^2 + \gamma_v^2}$. The reaction is inside the friction cone for $\gamma_r < \mu \gamma_n$ and inside the polar cone for $\gamma_r < -\frac{1}{\mu} \gamma_n$. For the remaining case, applying the Pitagora theorem on triangle OCA and similarity between triangles OCA and OCD, we easily get

$$\Pi_n = \frac{\gamma_r \mu + \gamma_n}{\mu^2 + 1} \ \ , \ \ \Pi_r = \mu \Pi_n \ \ , \ \ \Pi_u = \gamma_u \frac{\Pi_r}{\gamma_r} \ \ , \ \ \Pi_v = \gamma_v \frac{\Pi_r}{\gamma_r} . \tag{20}$$

The projective operator, modified for the complete case including bilateral constraints $\mathbf{C}$, generic unilateral constraints $\mathbf{D}$, and frictional contacts $\mathbf{F}$, becomes the following nonsmooth mapping:

$$\Pi_{\Upsilon^+} \begin{cases} \forall i \in \mathcal{G}_C & \Pi_i = \gamma_i \\ \forall i \in \mathcal{G}_D \wedge \gamma_i > 0 & \Pi_i = \gamma_i \\ \forall i \in \mathcal{G}_D \wedge \gamma_i \leq 0 & \Pi_i = 0 \\ \forall i \in \mathcal{G}_F \wedge \gamma_r < \mu_i \gamma_n & \Pi_i = \gamma_i \\ \forall i \in \mathcal{G}_F \wedge \gamma_r < -\frac{1}{\mu_i}\gamma_n & \Pi_i = \{0,0,0\}^T \\ \forall i \in \mathcal{G}_F \wedge \gamma_r > \mu_i \gamma_n \wedge \gamma_r > -\frac{1}{\mu_i}\gamma_n & \Pi_{i,n} = \frac{\gamma_r \mu_i + \gamma_n}{\mu_i^2 + 1} \\ & \Pi_{i,u} = \gamma_u \frac{\mu_i \Pi_{i,n}}{\gamma_r} \\ & \Pi_{i,v} = \gamma_v \frac{\mu_i \Pi_{i,n}}{\gamma_r} \end{cases} . \tag{21}$$

Iterations can be computed storing the [N] matrix explicitly, except for the sparse Jacobians of the constraints, hence resulting in strictly O(n) space requirements. To this end, we have developed efficient data structures and sparsity-preserving multiplication algorithms.

Below we present the final algorithm with some improvements, most noticeably, with the addition of an acceleration parameter λ and implementing immediate symmetric updates as in SSOR fixed-point methods. For reasons of space, we omit the simplifications that allow us to compute the iteration without explicitly building the [N] matrix, and we do not discuss the details of how to choose the η values.

ALGORITHM

```
1. For $i = 1, 2, \ldots, p$ compute the matrices $\mathbf{s}_i = [\mathrm{M}]^{-1}[\mathrm{E_q}]_i^T$ and matrices
   $g_i = [\mathrm{E_q}]_i \mathbf{s}_i$ (the latter being simple scalars for $i \in \mathcal{G}_C \cup \mathcal{G}_D$).
2. For $i \in \mathcal{G}_F$, compute $\eta_i = \frac{3}{\mathrm{Trace}(g_i)}$
   For $i \in \mathcal{G}_C \cup \mathcal{G}_D$, compute $\eta_i = \frac{1}{g_i}$
3. If warm starting with with some initial guess $\gamma^*$, initialize reactions as
   $\gamma^0 = \gamma^*$, otherwise $\gamma^0 = \mathbf{0}$.
4. Initialize speeds: $\dot{\mathbf{q}} = \sum_{i=1}^{n_c} \mathbf{s}_i \gamma_i^0 + [\mathrm{M}]^{-1}\mathbf{k}$.
5. For $i = 1, 2, \ldots p$, perform the updates
   $\delta_i^r = (\gamma_i^r - \omega\eta_i ([\mathrm{E_q}]_i \dot{\mathbf{q}}^r + b_i))$;
   $\gamma_i^{r+1} = \lambda \Pi_\Upsilon (\delta_i^r) + (1-\lambda)\gamma_i^r$ ;
   $\Delta\gamma_i^{r+1} = \gamma_i^{r+1} - \gamma_i^r$ ;
   $\dot{\mathbf{q}} := \dot{\mathbf{q}} + \mathbf{s}_i \Delta\gamma_i^{r+1}$.
6. $r := r + 1$
7. For $i = p, \ldots, 2, 1$, optionally perform symmetric updates
   $\delta_i^r = (\gamma_i^r - \omega\eta_i ([\mathrm{E_q}]_i \dot{\mathbf{q}}^r + b_i))$;
   $\gamma_i^{r+1} = \lambda \Pi_\Upsilon (\delta_i^r) + (1-\lambda)\gamma_i^r$ ;
   $\Delta\gamma_i^{r+1} = \gamma_i^{r+1} - \gamma_i^r$ ;
   $\dot{\mathbf{q}} := \dot{\mathbf{q}} + \mathbf{s}_i \Delta\gamma_i^{r+1}$.
8. $r := r + 1$
9. Repeat the loop 5 until convergence or until maximum number of itera-
   tions.
```

3 Examples

In this section, we present results from the application of our algorithm to two examples: a small-scale spider robot simulation example and a large-scale granular flow example.

3.1 Spider Robot

A walking robot has been modeled and simulated (see Fig. 2). The robot has six legs, each being actuated by two servomotors, for a total of 12 actuators. Multiple linkages and revolute joints are used to constrain the gait motion of the leg, hence leading to a total of 37 moving parts.

Each leg can collide with the environment. In detail, the feet collide with the ground in an intermittent pattern, introducing nonsmooth phenomena that must be handled by the solver. The rigid contacts are affected by a friction coefficient $\mu = 0.6$. Since this numerical approach does not introduce artificial stiffness into the system (unlike approaches that use spring-dashpot regularization at the point of contact), large integration time steps are allowed: we use $h = 0.01[s]$.

Although 50 iterations were used for each NCP problem, the CPU time for computing each time step is about 1 ms on a 2GHz laptop. That is, the simulation is almost ten times faster than the real-time requirement.

For simpler systems (pendulum, four-bar linkages, etc.) where a smaller number of constraints are used and a lower number of iterations can meet the tolerance requirements, the simulation can run almost one hundred times faster than real time.

3.2 Granular Flow

As a benchmark for measuring the performance of the solver, we used the granular flow simulation depicted in Fig. 3. A box is filled with a cascade of small spheres, which fill the inner space and slowly flow out from a lateral opening. All spheres are subject to rigid contacts with friction.

This benchmark represents one of the most critical scenarios because, in general, the stability of the integration and the speed of convergence of the solver are both affected negatively by dense packing of objects.

Multiple tests were performed with an increasing number of spheres, up to 50,000 colliding objects. The solver is able to handle problems with more than two millions unknowns. On average, the CPU time for a single time step, with 120 iterations and 50,000 objects, is 18s.

Note that the large number of iterations is mandatory if a decent tolerance is required: in this benchmark, the error of interpenetration between the colliding geometries does not exceed $0.002d$, with d being the radius of the spheres. If a less precise solution were tolerated, a smaller number of iterations could be used, hence improving the CPU performance even further.

As a side note, we remark that the NCP solver acts on potential contacts fed by a collision-detection algorithm. This algorithm should be as efficient as possible; otherwise the time spent in computing the points of contacts will be comparable to the time used for solving the NCP problem. [3]

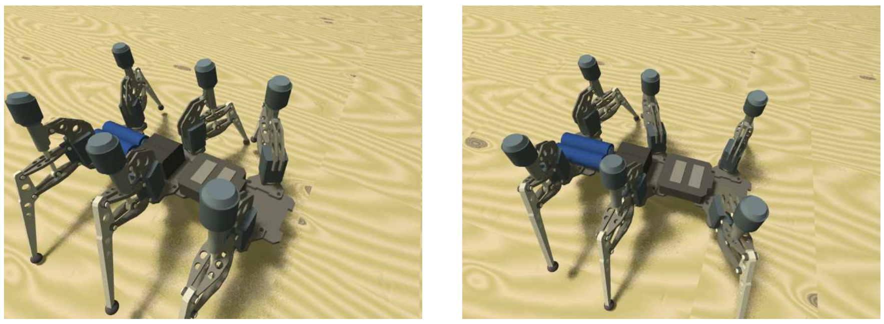
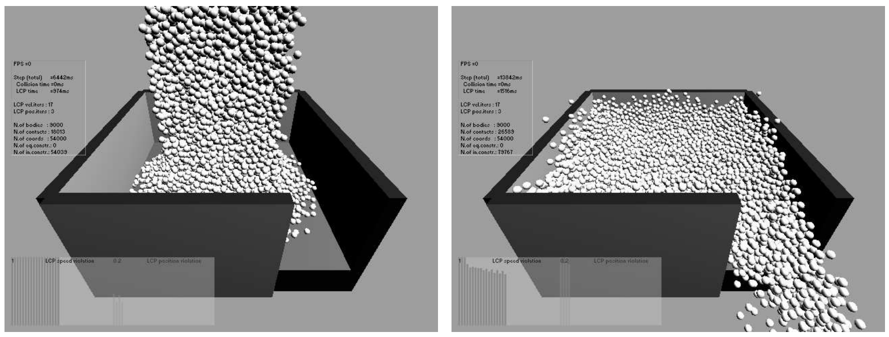


Fig. 2. Example: frames from a real-time simulation of a walking robot with 37 articulated parts, 12 motors, and frictional contacts between legs and floor.



```
FPS =0

Step (total)     =6442ms
  Collision time =0ms
  LCP time       =974ms

LCP vel.iters : 17
LCP pos.iters : 3

N.of bodies   : 9000
N.of contacts : 18013
N.of coords   : 54000
N.of eq.constr.: 0
N.of in.constr.: 54039
```

```
FPS =0

Step (total)     =13842ms
  Collision time =0ms
  LCP time       =1516ms

LCP vel.iters : 17
LCP pos.iters : 3

N.of bodies   : 9000
N.of contacts : 26589
N.of coords   : 54000
N.of eq.constr.: 0
N.of in.constr.: 79767
```

Fig. 3. Stress test: granular flow of spheres falling into a shaker. The figure shows the case with 9,000 spheres. A system of more than 100,000 complementarity constraints is solved at each step.

[3] Although this case of contact between spheres is trivial insofar complexity of contact features, we stress the importance of implementing a fast and reliable collision-detection engine for handling also more general cases. For example, our collision engine is able to compute collisions between generic compound shapes, convex or not, and exploits a sweep-and-prune broad phase that avoids superlinear algorithmic complexity.

4 Conclusions

We have developed a fast computational method to handle multibody systems encompassing nonsmooth phenomena, such as those caused by contacts and friction.

The method has been implemented as a C++ API in the Chrono::Engine middleware and extensively tested with complex simulation scenarios [17]. Benchmarks and comparisons show the superior performance of the method compared to other algorithms.

Our algorithm can run in O(n) space and time. Thus it can simulate in real-time even the most complex mechanical systems.

Acknowledgments

Mihai Anitescu was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, Office of Science, U.S. Dept. of Energy, under Contract DE-AC02-06CH11357.

References

1. Pfeiffer F, Glocker C (1996) Multibody Dynamics with Unilateral Contacts. John Wiley.
2. Tasora A (2001) An optimized Lagrangian-multiplier approach for interactive multibody simulation in kinematic and dynamical digital prototyping. In: Casolo F (ed.) Proceedings of VIII ISCSB. CLUP, Milano. 1–12.
3. Stewart E (1998) Convergence of a time-stepping scheme for rigid body dynamics and resolution of Painleve's problems, Archive Rational Mechanics and Analysis, 145(3), 215–260.
4. Tasora A (2006) An iterative fixed-point method for solving large complementarity problems in multibody systems. In: Proceedings of XVI GIMC, Bologna.
5. Alart P, Curnier A (1991) A mixed formulation for frictional contact problems prone to Newton-like solution methods. Comput. Methods Appl. Mech. Engrg. 92, 353-375.
6. Anitescu M, Tasora A (2007) An iterative approach for cone complementarity problems for nonsmooth dynamics, Argonne National Laboratory preprint ANL/MCS-P1413-0507.
7. Anitescu M, Hart GD (2004) A constraint-stabilized time-stepping approach for rigid multibody dynamics with joints, contact and friction. International Journal for Numerical Methods in Engineering 60(14), 2335–2371.
8. Anitescu M, Hart GD (2003) Solving nonconvex problems of multibody dynamics with contact and small friction by sequential convex relaxation. Mechanics Based Design of Machines and Structures 31(3), 335-356.
9. Anitescu M (2004) Optimization-based simulation of nonsmooth dynamics, Mathematical Programming, 105(1), 113–143.

10. Anitescu M, Potra FA (1997) Formulating dynamic multi-rigid-body contact problems with friction as solvable Linear Complementarity Problems, Nonlinear Dynamics , 14 , 231–247.
11. Monteiro-Marques MDP (1993) Differential inclusions in nonsmooth mechanical problems: Shocks and dry friction. In: Progress in Nonlinear Differential Equations and Their Applications, 9.
12. Tonge R, Zhang L, Sequeira D (2006) Method and program solving LCPs for rigid body dynamics. U.S. patent 7079145 B2.
13. Tasora A, Manconi E, Silvestri M (2006) Un nuovo metodo del simplesso per il problema di complementarit linerare mista in sistemi multibody. In: Proceedings of AIMETA 2005, 11-15 September 2005, Firenze.
14. Mangasarian OL (1977) Solution of symmetric linear complementarity problems by iterative methods. J. Optim. Theory Appl. 22:465–485.
15. Kocvara M, Zowe J (1994) An iterative two-step algorithm for linear complementarity problems. Numerische Mathematik 68:95–107.
16. Bolz J, Farmer I, Grinspun E, Shroeder P (2003) Sparse matrix solvers on the GPU: Conjugate gradients and multigrid. In: Computer Graphics SIGGRAPH'03 Proceedings, ACM, New York.
17. Tasora A (2006) Chrono::Engine web site: www.deltaknowledge.com/chronoengine.