Projective Dynamics with Dry Frictional Contact

MICKAËL LY, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, France
JEAN JOUVE, ENS Rennes and Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, France
LAURENCE BOISSIEUX, Univ. Grenoble Alpes, Inria, France
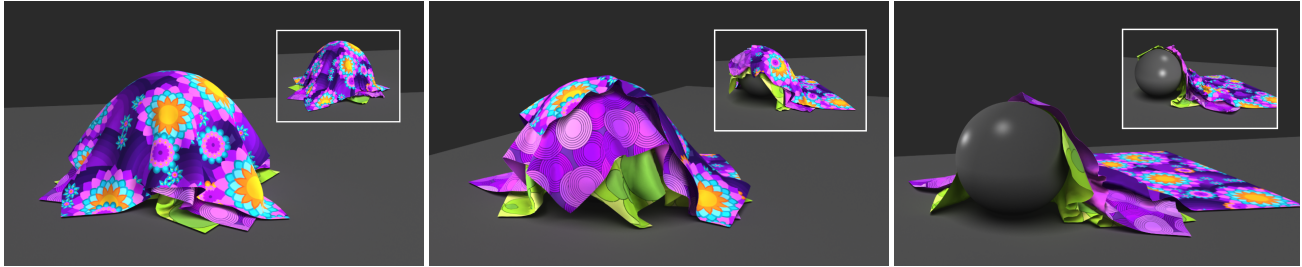FLORENCE BERTAILS-DESCOUBES, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, France

Fig. 1. Our frictional contact algorithm embedded inside Projective dynamics [Bouaziz et al. 2014] makes it possible to simulate challenging self-contacting scenarios, even at a modest number of iterations (20 iterations here). Compared to reference implicit contact solvers of the literature (here, Argus [Li et al. 2018] in inset, with fixed mesh resolution), it performs an order of magnitude faster while generating similar visual effects at the macroscopic scale.

Projective dynamics was introduced a few years ago as a fast method to yield an approximate yet stable solution to the dynamics of nodal systems subject to stiff internal forces. Previous attempts to include contact forces in that framework considered adding a quadratic penalty energy to the global system, which however broke the simple – constant matrix – structure of the global linear equation, while failing to treat contact in an implicit manner. In this paper we propose a simple yet effective method to integrate in a unified and semi-implicit way contact as well as dry frictional forces into the nested architecture of Projective dynamics. Assuming that contacts apply to nodes only, the key is to split the global matrix into a diagonal and a positive matrix, and use this splitting in the local step so as to make a good prediction of frictional contact forces at next iteration. Each frictional contact force is refined independently in the local step, while the original efficient structure of the global step is left unchanged. We apply our algorithm to cloth simulation and show that contact and dry friction can be captured at a reasonable precision within a few iterations only, hence one order of magnitude faster compared to global implicit contact solvers of the literature.

CCS Concepts: • **Computing methodologies → Animation**; **Physical simulation**.

Additional Key Words and Phrases: Projective Dynamics, dry frictional contact, cloth simulation, interactive simulation

Authors' addresses: Mickaël Ly, mickael.ly@inria.fr, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000, Grenoble, France; Jean Jouve, jean.jouve@ens-rennes.fr, ENS Rennes, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000, Grenoble, France; Laurence Boissieux, laurence.boissieux@inria.fr, Univ. Grenoble Alpes, Inria, 38000, Grenoble, France; Florence Bertails-Descoubes, florence.descoubes@inria.fr, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000, Grenoble, France.

1 INTRODUCTION

Properly accounting for contact and friction is a major challenge when simulating dynamic solid objects, such as 3D soft bodies, cloth, or hair. During motion, objects should not interpenetrate, but still interact in a stable way. At contact, they should dissipate energy in a realistic manner, following the Coulomb law for dry friction, which captures the typical threshold effect between stick and slip. In the last decades, a number of implicit algorithms have been designed so as to satisfy these major requirements in a stable and robust way [Daviet et al. 2011; Li et al. 2018; Otaduy et al. 2009]. However, such solvers currently remain inaccessible to interactive scenarios. Instead, interactivity is often obtained at the price of a simplification and/or a decoupled treatment of the interaction model [Kavan et al. 2011; Macklin et al. 2014]. Our goal in this paper is to build a realistic numerical model for frictional contact, fully coupled with the dynamics, which gets closer to interactive needs.

Consistently degradable model. Simulating complex deformable objects in a both realistic and interactive fashion is more than ever a topical issue, as new applications like virtual try on or fast prototyping, requiring both faithfulness and speed, are being deployed [Bartle et al. 2016; Wang 2018]. In addition, possessing tools to design an animation sequence interactively before proceeding to (offline) high-accuracy simulation constitutes a longstanding issue in the field of feature film production [Barbič et al. 2012]. In all cases, having access to a *consistently degradable* model, that is, a discrete model which converges to the desired continuous model while generating *satisfactory* (in a sense to be defined) simulations at low cost, is truly beneficial. In our case, we will qualify a simulation of satisfactory if

it yields visually convincing motions while being free of popping artifacts; and, in case of contact, if it strictly avoids penetration while qualitatively matching sticking and sliding behaviors compared to the high precision simulation.

Projective dynamics framework. In the case of nodal and contact-free systems, Projective dynamics [Bouaziz et al. 2014] appears as an interesting degradable model, able to yield an accurate solution to the dynamics if enough time budget is allocated, while computing an approximate yet stable solution in a very short amount of time – almost one order of magnitude faster compared to an incomplete Newton solve reaching the same precision. The key of the method is to approximate internal forces as linear terms of the positions, and to alternate iteratively between fast (linear) implicit global solves and local projection steps where forces are refined in parallel.

The common way to handle contact in that framework is to add a stiff elastic potential which penalizes interpenetrating vertices [Bouaziz et al. 2014]. However, the frequent activation/deactivation of contacts prevents from an efficient prefactoring of the global step, thus downgrading the overall performance of the base algorithm unless parallel GPU-based strategies are designed [Fratarcangeli et al. 2016; Komaritzan and Botsch 2019; Wang 2015]. Moreover, as contact is handled explicitly, weights associated to the contact forces must be chosen carefully to avoid penetration and popping artifacts. Finally, we are not aware of any algorithm incorporating dry friction in that framework. Recent attempts to include dry friction rely on the more general ADMM framework [Brown et al. 2018; Overby et al. 2017] and are not directly transferable to Projective dynamics. Besides, in these works, friction is decoupled from the normal force at each iteration, so that frictional contact is properly captured only when converging at a good precision. In contrast, in our approach we treat contact and friction simultaneously, while fully relying on the light and simple Projective dynamics framework.

Contribution. We propose a robust and efficient decoupled algorithm for incorporating frictional contact forces into Projective dynamics while keeping the global matrix constant. Assuming that contacts apply to nodes only, we rely on a simple splitting strategy in the local step so as to satisfy simultaneously non-penetration and Coulomb constraints, in a semi-implicit way (Section 3). Additionally, we extend our approach to self-contacts and carefully design a contact sorting scheme to handle multiple contact stacking in a stable way (Section 4). We obtain satisfactory simulations within 10 to 30 local/global iterations, even for complex scenarios involving multiple self-contact layers, and finally show that our method reaches a ×15 to ×36 speed gain compared to the reference code ARGUS [Li et al. 2018] used with fixed resolution (Section 5).

2 BACKGROUND

We consider a nodal system made of m vertices whose positions $\mathbf{q}(t)$ and velocities $\mathbf{v}(t) = \frac{d\mathbf{q}}{dt}$ are two functions from $\mathbb{R}_+$ to $\mathbb{R}^{3m}$. The system is subject to uniform external forces $\mathbf{f}_{\text{ext}}(t)$, such as gravity, as well as to stiff internal forces $\mathbf{f}_{\text{int}}(\mathbf{q}(t))$, such as elastic forces, assumed to be position-dependent, as in [Bouaziz et al. 2014]. The system may also undergo unknown frictional contact forces so as to satisfy constraints for non-penetration and Coulomb friction.

2.1 Implicit integration without contact

The time evolution of the system is dictated by the Newton second law, which is discretized in time using the implicit Euler scheme,

$$\begin{cases} \mathbf{q}_{n+1} = \mathbf{q}_n + h\mathbf{v}_{n+1} \\ \mathbf{v}_{n+1} = \mathbf{v}_n + h\mathbf{M}^{-1}\left(\mathbf{f}_{\text{int}}(\mathbf{q}_{n+1}) + \mathbf{f}_{\text{ext}}\right), \end{cases} \quad (1)$$

where $h > 0$ is the integration timestep, $(\mathbf{q}_n, \mathbf{v}_n) \in \mathbb{R}^{3m} \times \mathbb{R}^{3m}$ the state of the system at discrete time t_n, and $\mathbf{M}$ the (positive) diagonal mass matrix of the nodal system, of size $(3m, 3m)$.

Optimization view. We further assume, as in [Bouaziz et al. 2014], that internal forces derive from a potential energy, so that $\mathbf{f}_{\text{int}}(\mathbf{q}) = -\sum_i \nabla W_i(\mathbf{q})$. Equation (1) can then be interpreted as the necessary optimal conditions of the following optimization problem,

$$\min_{\mathbf{q}_{n+1}} \frac{1}{2h^2}(\mathbf{q}_{n+1} - \mathbf{s}_n)^{\mathsf{T}}\mathbf{M}(\mathbf{q}_{n+1} - \mathbf{s}_n) + \sum_i W_i(\mathbf{q}_{n+1}), \quad (2)$$

where $\mathbf{s}_n = \mathbf{q}_n + h\mathbf{v}_n + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}$. Projective dynamics [Bouaziz et al. 2014] computes an approximate yet stable solution to this problem that is almost one order of magnitude faster compared to a Newton solve with similarly low precision (see Section 2.3).

2.2 Implicit integration with frictional contact

We now assume the system to be subject to N_c dry frictional contacts $j = 1, \ldots, N_c$. Non-penetration constraints together with Coulomb friction conditions are compactly expressed by the so-called *Signorini-Coulomb law*. For each contact j, the Signorini–Coulomb law constrains each pair of relative velocity $\mathbf{u}_j \in \mathbb{R}^3$ and local contact force $\mathbf{r}_j \in \mathbb{R}^3$ at contact j (i.e. the relative velocity and contact force expressed in the local contact basis $\mathbf{R}_j$, see inset figure) to lie in a certain set C_{μ_j}, such that $(\mathbf{r}_j, \mathbf{u}_j) \in C_{\mu_j}$ if and only if

$$\begin{cases} \text{(take off) } \mathbf{r}_j = 0 \text{ and } \mathbf{u}_{j|N} \geq 0, \\ \text{or (stick) } \|\mathbf{r}_{j|T}\| \leq \mu_j \mathbf{r}_{j|N} \text{ and } \mathbf{u}_j = 0, \\ \text{or (slip) } \|\mathbf{r}_{j|T}\| = \mu_j \mathbf{r}_{j|N}, \mathbf{u}_{j|N} = 0, \text{ and } \exists \alpha_j > 0, \mathbf{r}_{j|T} = -\alpha_j \mathbf{u}_{j|T}, \end{cases}$$

where the normal and tangential components of any local vector $\mathbf{y}_j \in \mathbb{R}^3$ are denoted by $\mathbf{y}_{j|N} = \mathbf{y}_j^{\mathsf{T}}\mathbf{R}_j^{\mathsf{T}}\mathbf{e}_j \in \mathbb{R}$ and $\mathbf{y}_{j|T} = \mathbf{y}_j - \mathbf{y}_{j|N}\mathbf{R}_j^{\mathsf{T}}\mathbf{e}_j \in \mathbb{R}^3$ respectively, with $\mathbf{e}_j$ the contact normal expressed in the world frame (hence $\mathbf{R}_j^{\mathsf{T}}\mathbf{e}_j$ is the local contact normal).

The Signorini–Coulomb law captures three disjoint cases: taking-off, sticking, and slipping (see inset figure). We refer the reader to [Brogliato 2016; Daviet et al. 2011] for more details and various equivalent formulations of this law.

$$\mathbf{R}_j = (\mathbf{e}_j, \mathbf{t}_j)$$
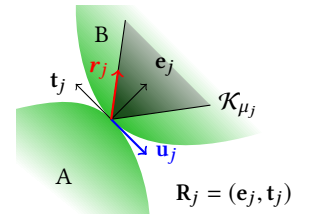
Implicit integration problem. Following Moreau's implicit time-stepping scheme [1988], the dynamic equations can be integrated over an arbitrary time step h as

$$\begin{cases} \mathbf{q}_{n+1} = \mathbf{q}_n + h\mathbf{v}_{n+1} & (3a) \\ \mathbf{v}_{n+1} = \mathbf{v}_n + h\mathbf{M}^{-1}(\mathbf{f}_{\text{int}}(\mathbf{q}_{n+1}) + \mathbf{f}_{\text{ext}} + \underbrace{\mathbf{J}^{\mathsf{T}}\mathbf{r}_{n+1}}_{\xi(\mathbf{r}_{n+1})}) & (3b) \\ \mathbf{u}_{n+1} = \mathbf{J}\mathbf{v}_{n+1} + \mathbf{u}_{\text{f}} & (3c) \\ \forall j = 1 \ldots N_c, \left(\mathbf{r}_{n+1,j}, \mathbf{u}_{n+1,j}\right) \in C_{\mu_j}, & (3d) \end{cases}$$

where a global unknown force $\xi(\mathbf{r}_{n+1})$, stemming from the Signorini-Coulomb constraint (3d), has been added to the linear momentum equation (3b). A linear kinematic law (3c) relates node velocities $\mathbf{v} \in \mathbb{R}^{3m}$ to contact velocities $\mathbf{u} \in \mathbb{R}^{3N_c}$ through the gradient matrix at contact $\mathbf{J} = \frac{\partial \mathbf{u}}{\partial \mathbf{v}}$, of size $(3N_c, 3m)$. The constant term $\boldsymbol{u}_{\mathrm{f}}$ is the value of $\mathbf{u}$ when $\mathbf{v} = 0$, for example, in the presence of external moving obstacles. Compared to Equation (1), the set of unknowns has been augmented in Equation (3) since now, not only the m node velocities $\mathbf{v}$ have to be solved for, but also the N_c local contact velocities and forces, $\mathbf{u} \in \mathbb{R}^{3N_c}$ and $\mathbf{r} \in \mathbb{R}^{3N_c}$.

A common way to solve the discrete frictional contact problem (3) robustly is first to linearize the internal force in (3b), then eliminate $\mathbf{v}$ by introducing the so-called *Delassus* operator $\mathbf{D} := \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^{\mathsf{T}}$, and finally employ an implicit nonsmooth solver of the literature [Daviet et al. 2011; Jean 1999; Kaufman et al. 2008; Otaduy et al. 2009] to solve the problem in $(\mathbf{u}, \mathbf{r})$. Of course this solving can be performed iteratively while linearizing the internal forces. Overall, such a method can be seen as an extension of the Newton method for a constrained system, hence it is not well-suited for fast simulation. In the following, we mention two interesting views of the problem which will help us design an accelerated algorithm for solving (3), in line with Projective dynamics.

Optimization view. Interestingly, Cadoux and colleagues [Acary et al. 2011; Cadoux 2009] have extended the common optimization interpretation of constraint-free implicit dynamics, recalled in Section 2.1, to the case of frictional contact. They have recast System (3) as a sequence of quadratic problems subject to conical constraints. Each iteration is much simpler to solve compared to the full problem, but it still remains costly compared to a linear solve. Inspired by this approach, and motivated by the gain of efficiency at low resolution, we have striven to design yet another iterative algorithm, leading this time to a *linear* global step.

Special case of a nodal system. It is noteworthy that in the case of a nodal system with contact occurring at nodes only, the gradient $\mathbf{J}$ takes a simple form as each one of its $3N_c$ lines j contains one non-zero 3×3 block at index $3i$ consisting of the transposed local basis $\mathbf{R}_j$ of contact j at node i, such that $\mathbf{u}_j = (\mathbf{R}_j)^{\mathsf{T}}\mathbf{v}_i + \mathbf{u}_{\mathrm{f},j}$. Daviet et al. [2015] have exploited this property to build an accelerated solver dedicated to nodal systems, which was then improved and combined with adaptive simulation [Li et al. 2018]. In that framework however, self-contacts cannot be incorporated easily as they break the simple invertible structure of $\mathbf{J}$. To keep the simple structure of $\mathbf{J}$, the authors suggest duplicating nodes sharing several contacts, and linking them through artificial pin constraints. This new set of constraints however considerably increases the solving cost. In contrast, our local contact algorithm supports several contacts per node, and proves extremely stable even at low precision (see Section 4).

2.3 Projective dynamics method

Projective dynamics [Bouaziz et al. 2014] has striven to outperform the traditional Newton method for solving the contact-free dynamics (1), at least in the range of low precision. The general idea is to reformulate the optimization problem (2) as a sequence of constraint-free quadratic programs with constant symmetric positive-definite matrix $\mathbf{P}$ – or equivalently, to reformulate the nonlinear equation (1)

as a sequence of linear problems $\mathbf{P}\mathbf{q}_{n+1} = \mathbf{b}$. To achieve this goal, the key is to express potential energies W_i as quadratic distance measures with respect to an (unknown) rest state $\mathbf{p}_i \in E_i$,

$$W_i(\mathbf{q}, \mathbf{p}_i) = \frac{\omega_i}{2}\|\mathbf{A}_i\mathbf{S}_i\mathbf{q} - \mathbf{B}_i\mathbf{p}_i\|^2 + \delta_{E_i}(\mathbf{p}_i),$$

where $\mathbf{A}_i$, $\mathbf{B}_i$ and $\mathbf{S}_i$ are constant matrices, ω_i is a nonnegative weight, and δ_{E_i} is the characteristic function of the set E_i (in the sense of convex analysis), that is, $\delta_{E_i}(\mathbf{p}_i) = 0$ if $\mathbf{p}_i \in E_i$, and $\delta_{E_i}(\mathbf{p}_i) = +\infty$ otherwise. This way, nonlinearity of the internal forces $\mathbf{f}_{\mathrm{int}}$ w.r.t. $\mathbf{q}$ is transferred to the local search of a suitable rest state $\mathbf{p}_i$, by solving, at fixed $\mathbf{q}$, each local and independent subproblem

$$\min_{\mathbf{p}_i} W_i(\mathbf{q}, \mathbf{p}_i) = \min_{\mathbf{p}_i \in E_i} \frac{\omega_i}{2}\|\mathbf{A}_i\mathbf{S}_i\mathbf{q} - \mathbf{B}_i\mathbf{p}_i\|^2. \qquad (4)$$

Main algorithm. The Projective dynamics method then boils down to an iterative algorithm alternating between parallel local solves of optimization problems (4) and the global solve of the linear system

$$\underbrace{\left(\mathbf{M} + h^2 \sum_i \omega_i \mathbf{S}_i^{\mathsf{T}}\mathbf{A}_i^{\mathsf{T}}\mathbf{A}_i\mathbf{S}_i\right)}_{\mathbf{P}} \mathbf{q}_{n+1} = \underbrace{\mathbf{M}\mathbf{s}_n + h^2 \sum_i \omega_i \mathbf{S}_i^{\mathsf{T}}\mathbf{A}_i^{\mathsf{T}}\mathbf{B}_i\mathbf{p}_i}_{\mathbf{b}_n(\mathbf{p})}. \qquad (5)$$

As local solves can be done in parallel and the global solve can benefit from a single precomputation of the constant inverse $\mathbf{P}^{-1}$, the method proves to be extremely fast in the absence of contact, even for a high number of degrees of freedom. The iterative Projective dynamics algorithm is outlined in Algorithm 1 (in black), with our modifications (in orange).

ALGORITHM 1: Algorithm of Projective Dynamics for computing positions and velocities at time t_{n+1}, augmented with our computation of frictional contact forces.

$\mathbf{q}_{n+1}^0 \leftarrow \mathbf{q}_n + h\mathbf{v}_n + h^2\mathbf{M}^{-1}\mathbf{f}_{\mathrm{ext}}$;

Detect the contacts;

Sort self-contacts (see section 4.2);

for k *from* 0 *to* $max\_iter - 1$ **do**

    Project $\mathbf{q}_{n+1}^k$ to obtain $\mathbf{p}_i^{k+1}$ using (4);

    Compute and add the friction forces (see sections 3.3 and 4);

    Solve the linear system (5) with fixed $\mathbf{p}_i^{k+1}$ to obtain $\mathbf{q}_{n+1}^{k+1}$;

end

$\mathbf{q}_{n+1} \leftarrow \mathbf{q}_{n+1}^{max\_iter}$;

$\mathbf{v}_{n+1} \leftarrow \frac{1}{h}(\mathbf{q}_{n+1} - \mathbf{q}_n)$;

Adding contacts. As mentioned earlier, in [Bouaziz et al. 2014] non-penetration is handled by adding a new elastic potential which pulls out the involved vertices. However, this scheme modifies the left-hand side of the global step, and thus removes the benefit of factorizing $\mathbf{P}^{-1}$ once for all. Furthermore, as contact forces are computed as explicit penalties, they may require a careful and scenario-dependent tuning of their associated weight to prevent interpenetration effectively. Finally, dry friction cannot be enforced robustly with this scheme. We depart from this approach and in the same spirit as what was recently done in the context of ADMM [Brown et al. 2018; Overby et al. 2017], we compute the unknown frictional

contact forces by refining them progressively in the local step. In contrast to the latter works however, we *fully* couple the normal and tangential components of each force, and compute the resultant in a *semi-implicit* way. This allows us to fulfil the Signorini-Coulomb constraints satisfactorily even at a low number of iterations.

3 SIGNORINI-COULOMB IN PROJECTIVE DYNAMICS

3.1 Velocity-based Projective dynamics

As seen in Section 2.2, the Signorini-Coulomb constraints are preferably expressed in terms of velocities (not positions). To include these constraints to the Projective dynamics framework, we thus start by reformulating the Projective dynamics algorithm in terms of velocities. While the local solves (4) are left unchanged, the global linear system (5) becomes

$$\mathbf{P}\mathbf{v}_{n+1} = \overbrace{\frac{1}{h}\left(\mathbf{b}_n(\mathbf{p}) - \mathbf{P}\mathbf{q}_n\right)}^{\tilde{\mathbf{b}}_n(\mathbf{p})}. \tag{6}$$

We denote by $\tilde{\mathbf{b}}_n(\mathbf{p})$ the new right-hand side term of the dynamics, given in Equation (6). Note that apart from this modification, the global solve is identical to the original system (5) expressed on positions. Finally, after having computed the new velocities $\mathbf{v}_{n+1}$, positions $\mathbf{q}_{n+1}$ are updated trivially as $\mathbf{q}_{n+1} := \mathbf{q}_n + h\mathbf{v}_{n+1}$. In practice, we have not noted any difference in the results yielded by this velocity-based algorithm compared to the original one.

3.2 Local vs global constraints

As in Section 2.2, we now consider the nodal system to be subject to N_c frictional contacts. For the moment we limit ourselves to the case where each contact j occurs between a single node i and a still obstacle (nodal contact, no self-contact and $\mathbf{u}_f = 0$). We shall only consider nodal contact in this paper but will alleviate other assumptions in Section 4.

As we now focus on the iterations of Projective dynamics, at *fixed* time step $n + 1$, we drop the $n + 1$ index from our notation for the sake of clarity. Local/global iterations of the Projective dynamics algorithm will be denoted by the letter k.

The dynamic equations (3) include a new force ξ, the global contact force, which is related to the unknown local contact forces $\mathbf{r}$ through the matrix $\mathbf{J}$. If the node i is involved in a contact j, then $\xi_i = \mathbf{R}_j\mathbf{r}_j$, where $\mathbf{R}_j$ is the local contact basis of j. Otherwise, $\xi_i = 0$.

First idea: global constraint. A straightforward idea is to note that at iteration $k + 1$, once the $\mathbf{p}_i$ have been updated, the global solve takes the form of a linearized dynamic equation subject to the Signorini-Coulomb constraints,

$$\begin{cases} \mathbf{P}\mathbf{v}^{k+1} = \tilde{\mathbf{b}}(\mathbf{p}) + \xi(\mathbf{r}^{k+1}) & \text{(7a)} \\ \mathbf{u}^{k+1} = \mathbf{J}\mathbf{v}^{k+1} + \mathbf{u}_f & \text{(7b)} \\ \forall j = 1 \dots N_c, \left(\mathbf{r}_j^{k+1}, \mathbf{u}_j^{k+1}\right) \in C_{\mu_j}, & \text{(7c)} \end{cases}$$

with the velocities $\mathbf{v}^{k+1}, \mathbf{u}^{k+1}$ and the contact impulses $\mathbf{r}^{k+1}$ as unknowns. Hence, we could use an implicit frictional contact solver of the literature, for instance the nodal solver of [Daviet et al. 2015; Li et al. 2018] to solve the global step. Although such a method works well, we have unfortunately observed that the global step is

slowed down by a factor 15, which makes it hardly compatible with interactive needs even for a low number of iterations.

Second idea: progressive refinement of the local contact forces. Observing Equation (7), we note that similarly to internal forces which depend on the local unknowns $\mathbf{p}_i$, the global contact force depends on the local unknowns $\mathbf{r}_j$. Our idea is thus to progressively refine the value of $\xi(\mathbf{r})$, by updating *locally* the values of the $\mathbf{r}_j$, at each iteration of Projective dynamics. That is, starting from the local step at iteration k, we first compute the $\mathbf{p}^{k+1}$ as in [Bouaziz et al. 2014], which are simply denoted by $\mathbf{p}$ in the following. Then, we aim at computing the $\mathbf{r}_j^{k+1}$ in parallel (in a manner that remains to be defined). If all these local solves can be done properly, the complex global solve (7) can be replaced with the following *linear* global solve, $\mathbf{P}\mathbf{v}^{k+1} = \tilde{\mathbf{b}}(\mathbf{p}) + \xi(\mathbf{r})$, which has exactly the same cost as the original global solve of Projective dynamics without contact.

We now explain how the contact forces can indeed be updated locally to approach the Signorini-Coulomb constraints globally.

3.3 Local update of the frictional contact forces

Our goal is to update the frictional contact forces $\mathbf{r}_j$ so that each pair $(\mathbf{u}_j^{k+1}, \mathbf{r}_j^{k+1})$ satisfies the Signorini-Coulomb law at iteration $k + 1$. Unlike the minimization of distance potentials, the Signorini-Coulomb law involves velocities $\mathbf{v}$ of the body as we have $\mathbf{u}_j = (\mathbf{R}_j)^\mathsf{T}\mathbf{v}_i$. To obtain stable results even at low precision, we propose to do this update *semi-implicitly*. To this aim, we seek to find an approximate equation relating $\mathbf{u}_j$ to $\mathbf{r}_j$ for each contact j, and then set the values of $\mathbf{r}_j$ so as to satisfy the Signorini-Coulomb constraint at each contact.

Splitting scheme. Inspired by the popular Jacobi scheme for solving linear systems, our key idea is to approximate Equation (7a) through a *splitting* operation. More precisely, we search for a decomposition of $\mathbf{P}$ as a sum of a positive diagonal matrix and a positive matrix. Such a decomposition is straightforward, as we readily have $\mathbf{P} = \mathbf{M} + \mathbf{C}$ with $\mathbf{M}$ the (diagonal positive) mass-matrix, and $\mathbf{C} = h^2 \sum_i \omega_i \mathbf{S}_i^\mathsf{T} \mathbf{A}_i^\mathsf{T} \mathbf{A}_i \mathbf{S}_i$ a positive matrix. At iteration $k + 1$ we thus obtain the splitting scheme

$$\mathbf{M}\mathbf{v}^{k+1} = \overbrace{\tilde{\mathbf{b}}(\mathbf{p}) - \mathbf{C}\mathbf{v}^k}^{\mathbf{f}} + \xi(\mathbf{r}^{k+1}), \tag{8}$$

yielding the nodal equation

$$\forall 0 \le i \le m \quad M_i\mathbf{v}_i^{k+1} = \begin{cases} \mathbf{f}_i & \text{if } i \text{ not in contact} \\ \mathbf{f}_i + \mathbf{R}_j\mathbf{r}_j^{k+1} & \text{if } i \text{ in contact } j, \end{cases}$$

where $M_i > 0$ is the mass of the ith node.

For each contacting node i (involved in the contact j), we use the nodal contact assumption $\mathbf{u}_j = (\mathbf{R}_j)^\mathsf{T}\mathbf{v}_i$ and finally obtain an approximate local equation relating $\mathbf{u}_j$ to $\mathbf{r}_j$ at next iteration $k + 1$,

$$\forall 0 \le j \le N_c \quad M_i\mathbf{u}_j^{k+1} = \underbrace{\mathbf{R}_j^\mathsf{T}\mathbf{f}_i}_{\mathbf{d}_j} + \mathbf{r}_j^{k+1}. \tag{9}$$

Enforcing Signorini-Coulomb. We can now enforce the Signorini-Coulomb law at iteration $k + 1$ by examining the normal and tangential components of the term $\mathbf{d}_j$. Indeed,

- If $\mathbf{d}_{j|N} \ge 0$ then the body should be taking off at next iteration, that is, we set $\mathbf{r}_j^{k+1} = 0$ (and we indeed have $\mathbf{u}_{j|N}^{k+1} \ge 0$);

- Otherwise, the object should lie in contact (without penetration) at next iteration, that is we set $\mathbf{r}_{j|N}^{k+1} = -\mathbf{d}_{j|N} \geq 0$ so that $\mathbf{u}_{j|N} = 0$. Moreover,
  - If $\|\mathbf{d}_{j|T}\| \leq \mu_j \mathbf{r}_{j|N}$ then the contact should be sticking at next iteration, that is, we set $\mathbf{r}_{j|T}^{k+1} = -\mathbf{d}_{j|T}$ so that $\mathbf{u}_j = 0$ (and the contact force correctly lies inside the cone);
  - Otherwise, the contact should be slipping at next iteration, that is, we set $\mathbf{r}_{j|T}^{k+1} = -\mu_j \mathbf{r}_{j|N}^{k+1} \frac{\mathbf{d}_{j|T}}{\|\mathbf{d}_{j|T}\|}$, so that $\mathbf{u}_{j|T}$ is in the direction opposed to $\mathbf{r}_{j|T}$ (and the contact force correctly lies on the boundary of the cone).

Convergence. We prove in Appendix A that if our algorithm converges, then at convergence the contact forces and velocities satisfy the Signorini-Coulomb law. Studying the exact conditions of convergence are left for future work. In practice, we have observed an excellent behavior of our algorithm even for a low number of iterations. In Section 5, we provide some numerical studies demonstrating the experimental convergence of our method.

4 EXTENSIONS

4.1 Moving obstacle and self-contact

Moving obstacle. The relation between the two frames becomes affine, $\mathbf{u} = \mathbf{J}\mathbf{v} + \mathbf{u}_f$, with $\mathbf{u}_f$ the translation velocities of the contact frames, i.e. the opposite of the velocities of the contact points *on the obstacle*. Equation (9) is slightly modified as

$$\forall 0 \leq j \leq N_c \quad M_i \mathbf{u}_j^{k+1} = \mathbf{R}_j^\mathsf{T} \mathbf{f}_i + M_i \mathbf{u}_{f,j} + \mathbf{r}_j^{k+1}. \tag{10}$$

The previous reasoning remains valid with now $\mathbf{d}_j := \mathbf{R}_j^\mathsf{T} \mathbf{f}_i + M_i \mathbf{u}_{f,j}$.

Self-contact. We assume that the jth contact is a self-contact between nodes i_A and i_B. The nodes are thus submitted to the contact forces $\xi(\mathbf{r}_j)$ and $-\xi(\mathbf{r}_j)$, respectively, which are related to the local velocity $\mathbf{u}_j = R_j^\mathsf{T}(\mathbf{v}_{i_A} - \mathbf{v}_{i_A})$. We are therefore looking for an equation similar to Equation (9), involving the difference of velocities. From Equation (8), the two nodes are described by the equations

$$\begin{cases} M_{i_A} \mathbf{v}_{i_A}^{k+1} &= \mathbf{f}_{i_A} + \mathbf{R}_j \mathbf{r}_j^{k+1} \\ M_{i_B} \mathbf{v}_{i_B}^{k+1} &= \mathbf{f}_{i_B} - \mathbf{R}_j \mathbf{r}_j^{k+1}, \end{cases} \tag{11}$$

from which we can compute $\mathbf{u}_j^{k+1}$ as

$$\mathbf{u}_j^{k+1} = \mathbf{R}_j^\mathsf{T} \left(\mathbf{v}_{i_A}^{k+1} - \mathbf{v}_{i_B}^{k+1} \right) = \mathbf{d}_j + \frac{M_{i_A} + M_{i_B}}{M_{i_A} M_{i_B}} \mathbf{r}_j^{k+1}, \tag{12}$$

where $\mathbf{d}_j := \mathbf{R}_j^\mathsf{T} \left(\frac{1}{M_{i_A}} \mathbf{f}_{i_A} - \frac{1}{M_{i_B}} \mathbf{f}_{i_B} \right)$. The local force $\mathbf{r}_j^{k+1}$ can then be enforced as in Section 3.3, with appropriate scaling by $\frac{M_{i_A} M_{i_B}}{M_{i_A} + M_{i_B}}$.

4.2 Dealing with multiple contacts per node

In our current scheme, contact forces are considered as external forces which are all updated in parallel after the update of the $\mathbf{p}_i$. However, in the case when a node is subject to multiple contacts (multi-layering case), we found this algorithm to be pretty unstable. Indeed, instead of stabilizing nodes on the surface of contact, non-penetration forces would instead "overreact" and diverge due to their knowledge of other non-penetration forces only at previous iteration, leading to some popping artifacts.

To overcome this issue, we sort the contact forces in several "layers" (batches) that are processed sequentially to avoid inconsistencies, while forces in the same layer can safely be processed in parallel. More precisely,
we process the list of contacts and organize them as follows:

- Contacts between two nodes that are not involved in other contacts are safe and added to the first parallel batch (1);
- Contact between a node and an external obstacle are also added to the first layer (1). Indeed, the external object does not react to contact forces applied on it; the latter can thus serve as the "origin" for the next layers (inset figure, left);
- Then we process the remaining contacts by traversing our graph of contacts, and build the different contact "layers";
- Finally, for layers of self-contacts that are not in contact with an external object, we start arbitrarily from one "side" and build the layers through to the other side (inset figure, right).
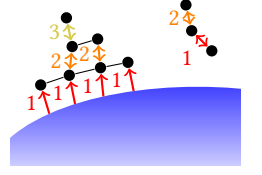
5 RESULTS AND COMPARISONS

We have implemented the Projective dynamics algorithm in C/C++ using *OpenMP* for parallelization. Collision detection is performed through simple proximity queries, using an acceleration structure. All the examples were run on a desktop computer featuring 4 dual-core Intel i7-5600U processors running at 2.60GHz.

Examples. Our results focus on cloth examples, which are particularly challenging in terms of frictional contact, but our technique is general and can be applied to any nodal system satisfying Projective dynamics energies. We have tested our method on five scenarios:

- **Ribbon**: a ribbon falling on an inclined plane, causing multiple layered self-contacts;
- **Square1**: a flat sheet falling on a rotating sphere, showcasing impacts and frictional contact with a moving obstacle;
- **Square3**: three stacked flat sheets falling on a rotating sphere, involving additional multi-layered contact compared to the previous example;
- **Arabesque**: the dress of a dancing character, freely available from [Li et al. 2018], combining impacts with a moving obstacle and stick-slip thresholding behavior.
- **Crinoline**: a highly-detailed gown with puff sleeves and complex folds, subject to walking and turning motions.

Table 1 gives the configuration for each one of these scenarios. For better realism in **Arabesque** and **Crinoline**, we added air damping forces modeled explicitly as $\xi_i^{k+1} = -\nu \mathbf{v}_i^k$ for each non-contacting node, with ν the damping coefficient.

Sample images of our examples are provided in Figures 3 and 4. Please also refer to the accompanying video for the full animations.

Table 1. Main parameters used for our five examples.

Example	Nb of vertices	Mass density (kg·m⁻²)	Stretch. weight (N·m⁻¹)	Bend. weight (N·m)	Air damp. coeff. (N·s·m⁻¹)	Time step (ms)
Ribbon	5946	0.25	20	$2 \cdot 10^{-2}$	0	4
Square1	5996	3	200	$5 \cdot 10^{-4}$	0	5
Square3	17964	1.5	200	$5 \cdot 10^{-4}$	0	5
Arabesque	15842	0.01	4	$5 \cdot 10^{-7}$	$3 \cdot 10^{-6}$	2
Crinoline	53097	0.01	5	$5 \cdot 10^{-6}$	$2.5 \cdot 10^{-7}$	2

5.1 Qualitative evaluation

Influence of the number of iterations. Figure 2 shows the effect of varying the number of global/local iterations for **Ribbon** and **Square1**. For both examples, simulations degrade consistently down to 10 iterations, below which **Square1** stops converging properly, while **Ribbon** continues to work consistently. In terms of accuracy, no significant difference is observed above 20 iterations. In the sequel, we thus chose 20 iterations for these two examples along with **Square3**. For **Arabesque** and **Crinoline**, which involve higher resolution meshes and stiffer materials, we used 30 iterations.

Varying the friction coefficient and comparison with Argus. As shown in Figure 3, we have run our four examples with both low and large values of the friction coefficient to demonstrate the macroscopic impact of sticking *vs.* sliding behaviors. We have reproduced these experiments using the accurate and freely available Argus simulator [Li et al. 2018], using the same timesteps and a fixed mesh resolution. Due to the specific form of energies required by Projective dynamics, we were not able to match exactly the same material parameters as those used for Argus, but we picked stretching and bending weights so as to obtain a reasonable match. Despite these differences in material, it is noteworthy that the response to frictional contact yielded by our method is similar to that generated by Argus, as shown in our accompanying video. Moreover, in case of multi-layered contacts, Argus in its fixed resolution version depicts some obvious artifacts (presumably due to an overly coarse mesh resolution in folding areas), whereas our method generates perfectly stable results.
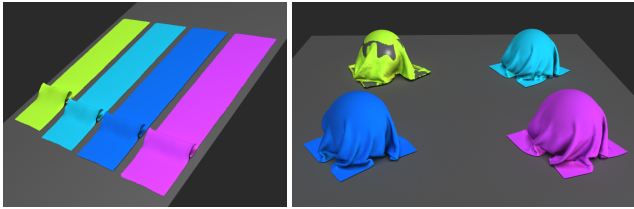

Fig. 2. Simulating the **Ribbon** (left, $\mu = 0.3$) and **Square1** (right, $\mu = 0.1$) examples using a various number of iterations. In green: 5 iterations, in light blue: 10 iterations, in dark blue: 20 iterations, in purple: 30 iterations.

5.2 Convergence

Analytical example. As in [Li et al. 2018], we have evaluated our method on the scenario of a falling sheet parallel to an inclined plane, illustrated in Figures 5a and 5b. In this setup, an analytical model for the dynamic of the nodes can be derived, and we show in Figures 5c and 5d that our numerical results match the analytical curves. Moreover, we have measured a global penetration and friction error using the so-called Alart-Curnier function [Alart and Curnier 1991; Bertails-Descoubes et al. 2011] and we have found out that, in this simple scenario, for each time-step the error vanishes after just one local/global iteration. Indeed, as all nodes have the same motion due to the initial configuration, no internal force is applied, i.e. the terms in **C** in both sides of the equation cancel each other, meaning that no error is introduced by the splitting scheme.
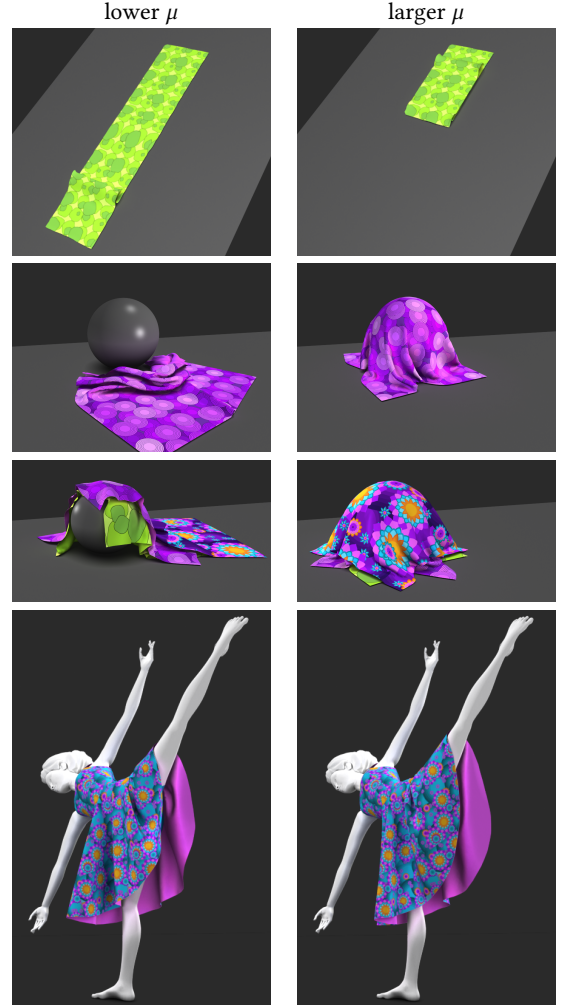
Fig. 3. Varying the friction coefficient in our four examples **Ribbon** ($\mu = 0.3$ and $\mu = 0.6$), **Square1** and **Square3** ($\mu = 0.1$ and $\mu = 0.3$), and **Arabesque** ($\mu = 0.0$ and $\mu = 0.3$), using 20 to 30 iterations per timestep. Results are visually close to those generated by the accurate Argus simulator [Li et al. 2018] (see accompanying video), while being computed 15 to 36 times faster.

General case. In most of the cases where the terms in **C** are not negligible, our scheme still manages to decrease the Alart-Curnier error, yielding the visually good results described in Section 5.1.

In Figure 6, we show the evolution of the normal and tangential errors (i.e. the amount of deviation from the non-penetration and Coulomb friction constraints, respectively) w.r.t the number of iterations for four different time steps selected from the **Sphere1** example. We see that both errors quickly decrease in the first iterations, before reaching a plateau at moderate precision with a much lower slope. Also note that the tangential part responsively adjusts to the bound set by the normal component in the Coulomb law.

5.3 Performance

Table 2 shows that our method fits in well with the Projective dynamics algorithm. Indeed, compared to the native algorithm without
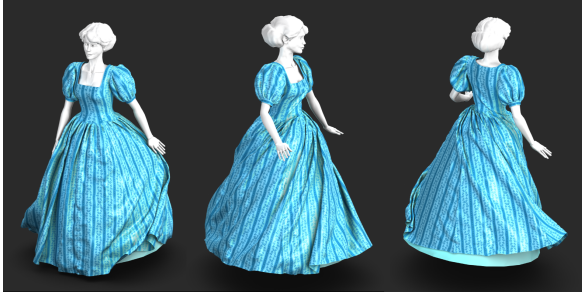
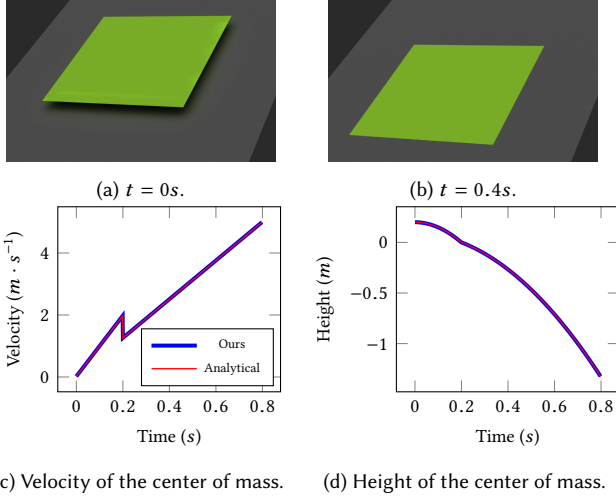Fig. 4. Highly-detailed **Crinoline** example ($\mu = 0.3$, 30 iterations).

(a) $t = 0s$.

(b) $t = 0.4s$.

(c) Velocity of the center of mass.

(d) Height of the center of mass.

Fig. 5. Comparison to an analytical scenario.

(a) Normal part.

(b) Tangential part (zoom on the 100 first local/global iterations).
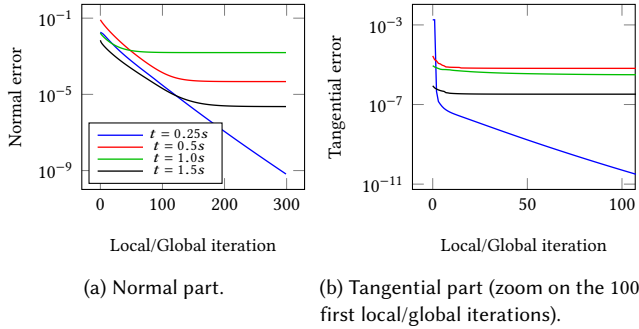
Fig. 6. Evolution of the mean Alart-Curnier error w.r.t the local/global iterations on the **Sphere1** scenario.

contact, it only adds a small overhead when computing the right-hand side of the global equation and locally updating contact forces using our sorting algorithm (columns in orange). Additionally, as we do not need to modify the left-hand side of the global solve, we preserve the inherent speed of Projective dynamics.

Comparison with penalty contact forces. We have run **Square1** by computing contacts as penalty forces, similarly to [Bouaziz et al. 2014] and most follow-up papers. We have observed a speed gain

$g = \frac{t_{\text{Bouaziz}}}{t_{\text{ours}}} = 1.1$ brought by our approach, even though ours is richer and more robust as it fully captures the Signorini-Coulomb law in a semi-implicit way. This overhead in [Bouaziz et al. 2014] is due to the Cholesky factorization which needs to be performed at each global step, and eventually represents a cost of 21% of the global step on average.

Comparison with Argus *[Li et al. 2018].* Table 2 reports the average cost of Argus's timesteps for **Square1**, **Square3**, and **Arabesque** run with the same number of vertices and the same timestep values used by our method, and without adaptivity. It turns out that our method runs more than one order of magnitude faster, with a speed gain[1] comprised between ×15 and ×36. We have noticed that the Argus solver is especially penalized when many self-contacts are involved (**Square3** example), due to their handling through vertex duplication and artificial pin constraints.

5.4 Limitation

Our method of course inherits limitations of Projective dynamics, and in particular the lack of a simple rule to ensure convergence. In order to obtain stable simulations, the user needs in a preliminary step to adjust the number of iterations required, depending on the mesh size and the material used.

Second and more importantly, our algorithm only considers vertex-vertex contact, which can be insufficient for handling properly some specific scenarios, such as cloth contacting an obstacle with corners or sharp edges. However, treating a contact point that is not a node is not straightforward. In such a case, the block lines of the matrix $\mathbf{J}$ relating velocities of the contact points $\mathbf{u}$ to the degrees of freedom $\mathbf{v}$ do not only contain one rotation, but a linear combination of rotations. Thus, per contact point, the system is not invertible as it is with nodal contacts. Including vertex-face and edge-edge contact in our framework hence remains an open direction of research.

Finally, our experiments on convergence suggest that the accuracy of our method degrades as the variation of internal forces increases. In the future we plan to investigate how to improve numerical convergence in these cases.

6 CONCLUDING REMARKS

We have presented a simple method to introduce dry frictional contact into the Projective dynamics framework in a robust yet cheap way. Our technique preserves the global step Cholesky factorization that is a keystone of the speed of the method, and only adds a small overhead when assembling the equations and updating contact forces. There are still many ways to improve the efficiency of our method, such as mesh adaptivity [Narain et al. 2012], GPU implementation [Wang 2015], and efficient detection collision scheme. In future work we plan to build upon these orthogonal works in the hope of reaching full interactivity of the approach.

---

[1]With adaptivity enabled in Argus, the speed gain lies between ×6.5 and ×8.

Table 2. Performance of our solver for all our examples

Example	μ	$\bar{n}_{\text{ext}}{}^2$	$\bar{n}_{\text{self}}{}^2$	$\bar{t}_{\text{rhs}}{}^3$	$\bar{t}_{\text{ext}}{}^3$	$\bar{t}_{\text{self}}{}^3$	$\bar{t}_{\text{solve}}{}^3$	$\bar{t}_i{}^3$	$\bar{t}_{\text{contact}}{}^4$	$\bar{t}_{\text{self-contact}}{}^4$	$\bar{t}_{\text{sorting}}{}^4$	$\bar{t}_p{}^5$	$\bar{t}_{\text{Argus}}{}^6$	$\bar{g}{}^6$
Ribbon	0.3 [1]	4210	455	1.9	0.29	0.07	1.02	3.3	3.1	38.4	0.17	109	–	–
	0.6 [1]	2194	1769	1.9	0.27	0.179	1.02	3.4	2.8	49.8	0.29	122	–	–
Square1	0.1	2273	1009	2.0	0.29	0.142	1.24	3.8	5.9	69.5	0.25	153	2244	**14.7**
	0.3	1665	301	2.1	0.28	0.071	1.23	3.7	6.6	62.6	0.16	144	4828	**33.5**
Square3	0.1	4034	5840	6.2	0.79	0.66	3.60	11.4	11.8	283.6	1.23	526	18713	**35.6**
	0.3	2222	5311	6.2	0.78	0.59	3.60	11.3	13.1	283.4	1.26	525	19233	**36.6**
Arabesque	0.0	3354	155	5.3	0.62	0.121	4.33	10.4	51.3	161.2	0.28	530	9069	**17.1**
	0.3	3673	102	5.4	0.68	0.093	4.39	10.6	58.0	152.7	0.30	543	15899	**29.3**
Crinoline	0.3	5977	1052	15.6	2.22	0.427	17.0	35.5	200	483	1.04	1751		

[1] Self-friction coefficient only. In the **Ribbon** example, the friction coefficient with the inclined plane is 0.7.

[2] Average number of contact points with external objects ($\bar{n}_{\text{ext}}$) and with the object itself ($\bar{n}_{\text{self}}$).

[3] Average time in ms per iteration ($\bar{t}_i$), including the time for assembling the right-hand side ($\bar{t}_{\text{rhs}}$), the computation of the frictional contact forces with external obstacles ($\bar{t}_{\text{ext}}$) and with the object itself ($\bar{t}_{\text{self}}$) and the global step solve ($\bar{t}_{\text{solve}}$).

[4] Average time in ms to detect the collisions $\bar{t}_{\text{contact}}$, the self-collisions $\bar{t}_{\text{self-contact}}$ and to perform the contact sorting $\bar{t}_{\text{sorting}}$.

[5] Average time in ms per timestep ($\bar{t}_p = \bar{t}_{\text{contact}} + \bar{t}_{\text{self-contact}} + \bar{t}_{\text{sorting}} + n_{\text{iter}} \times \bar{t}_i$) with $n_{\text{iter}} = 20$ for the three first examples and $n_{\text{iter}} = 30$ for the last two.

[6] Average time in ms per timestep ($\bar{t}_{\text{Argus}}$) for the ARGUS solver [Li et al. 2018] used with a fixed mesh resolution, and speed gain $\left(\bar{g} = \frac{\bar{t}_{\text{Argus}}}{\bar{t}_p} \right)$ of our method.

REFERENCES

V. Acary, F. Cadoux, C. Lemaréchal, and J. Malick. 2011. A formulation of the linear discrete Coulomb friction problem via convex optimization. *ZAMM* 91 (02 2011), 155–175. http://hal.inria.fr/inria-00495734/en/

P. Alart and A. Curnier. 1991. A mixed formulation for frictional contact problems prone to Newton like solution methods. *Comput. Methods Appl. Mech. Eng.* 92, 3 (1991), 353–375.

J. Barbič, F. Sin, and E. Grinspun. 2012. Interactive Editing of Deformable Simulations. *ACM Trans. Graph.* 31, 4, Article 70 (July 2012), 8 pages.

A. Bartle, A. Sheffer, V. Kim, D. Kaufman, N. Vining, and F. Berthouzoz. 2016. Physics-driven Pattern Adjustment for Direct 3D Garment Editing. *ACM Trans. Graph.* 35, 4, Article 50 (July 2016), 11 pages.

F. Bertails-Descoubes, F. Cadoux, G. Daviet, and V. Acary. 2011. A nonsmooth Newton solver for capturing exact Coulomb friction in fiber assemblies. *ACM Trans. Graph.* 30, Article 6 (February 2011), 14 pages. Issue 1.

S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly. 2014. Projective Dynamics: Fusing Constraint Projections for Fast Simulation. *ACM Trans. Graph.* 33, 4, Article 154 (July 2014), 11 pages.

B. Brogliato. 2016. *Nonsmooth Mechanics.* Springer International Publishing Switzerland. https://doi.org/10.1007/978-3-319-28664-8 Third edition.

G. Brown, M. Overby, Z. Forootaninia, and R. Narain. 2018. Accurate Dissipative Forces in Optimization Integrators. *ACM Trans. Graph.* 37, 6, Article 282 (Dec. 2018), 14 pages.

F. Cadoux. 2009. *Méthodes d'optimisation pour la dynamique non-régulière.* Ph.D. Dissertation. Université Joseph Fourier.

G. Daviet, F. Bertails-Descoubes, and L. Boissieux. 2011. A Hybrid Iterative Solver for Robustly Capturing Coulomb Friction in Hair Dynamics. *ACM Trans. Graph.* 30, 6 (Dec. 2011), 1–12.

G. Daviet, F. Bertails-Descoubes, and R. Casati. 2015. Fast Cloth Simulation with Implicit Contact and Exact Coulomb Friction. ACM SIGGRAPH / Eurographics Symposium on Computer Animation. https://hal.inria.fr/hal-01180756 Poster.

M. Fratarcangeli, V. Tibaldo, and F. Pellacini. 2016. Vivace: A Practical Gauss-Seidel Method for Stable Soft Body Dynamics. *ACM Trans. Graph.* 35, 6, Article 214 (Nov. 2016), 9 pages.

M. Jean. 1999. The Non Smooth Contact Dynamics Method. *Computer Methods in Applied Mechanics and Engineering* 177 (1999), 235–257. Special issue on computational modeling of contact and friction, J.A.C. Martins and A. Klarbring, editors.

D. Kaufman, S. Sueda, D. James, and D. Pai. 2008. Staggered Projections for Frictional Contact in Multibody Systems. *ACM Trans. Graph.* 27, 5, Article 164 (Dec. 2008), 11 pages.

L. Kavan, D. Gerszewski, A. W. Bargteil, and P. Sloan. 2011. Physics-inspired Upsampling for Cloth Simulation in Games. *ACM Trans. Graph.* 30, 4, Article 93 (July 2011), 10 pages. https://doi.org/10.1145/2010324.1964988

M. Komaritzan and M. Botsch. 2019. Fast Projective Skinning. In *Motion, Interaction and Games (MIG '19).* ACM, New York, NY, USA, Article 22, 10 pages.

J. Li, G. Daviet, R. Narain, F. Bertails-Descoubes, M. Overby, G. Brown, and L. Boissieux. 2018. An Implicit Frictional Contact Solver for Adaptive Cloth Simulation. *ACM Trans. Graph.* 37, 4, Article 52 (Aug. 2018), 15 pages.

M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim. 2014. Unified Particle Physics for Real-Time Applications. *ACM Trans. Graph.* 33, 4, Article 153 (July 2014), 12 pages.

J.-J. Moreau. 1988. Unilateral contact and dry friction in finite freedom dynamics. Nonsmooth mechanics and applications, CISM Courses Lect. 302, 1-82 (1988)..

R. Narain, A. Samii, and J. O'Brien. 2012. Adaptive Anisotropic Remeshing for Cloth Simulation. *ACM Trans. Graph.* 31, 6, Article 152 (Nov. 2012), 10 pages.

M. Otaduy, R. Tamstorf, D. Steinemann, and M. Gross. 2009. Implicit Contact Handling for Deformable Objects. *Computer Graphics Forum* 28, 2 (2009), 559–568.

M. Overby, G. E. Brown, J. Li, and R. Narain. 2017. ADMM ⊇ Projective Dynamics: Fast Simulation of Hyperelastic Models with Dynamic Constraints. *IEEE Trans. Vis. and Comp. Graph.* 23, 10 (Oct 2017), 2222–2234.

H. Wang. 2015. A Chebyshev Semi-Iterative Approach for Accelerating Projective and Position-Based Dynamics. *ACM Trans. Graph.* 34, 6, Article 246 (Oct. 2015), 9 pages.

H. Wang. 2018. Rule-Free Sewing Pattern Adjustment with Precision and Efficiency. *ACM Trans. Graph.* 37, 4, Article 53 (July 2018), 13 pages.

A SIGNORINI-COULOMB AT CONVERGENCE

We prove that if our algorithm converges, then the Signorini-Coulomb law is satisfied at convergence. The global step at a fixed point $\mathbf{v}^*$ (which is assumed to exist) reads $(\mathbf{M} + \mathbf{C})\,\mathbf{v}^* = \tilde{\mathbf{b}}(\mathbf{p}^*) + \xi[\mathbf{f}(\mathbf{v}^*)]$, where ξ is the global contact force estimated in Section 3.3, which depends on $\mathbf{v}^*$ through $\mathbf{f}(\mathbf{v}^*) = \tilde{\mathbf{b}}(\mathbf{p}^*) - \mathbf{C}\mathbf{v}^*$, with $\tilde{\mathbf{b}}(\mathbf{p}^*)$ fixed as $\mathbf{p}^*$ directly depends on $\mathbf{v}^*$. As in Eq. (9), we take the ith block corresponding to the jth contact. Considering for now only the normal part of the constraint (Signorini), we project this 3D equation onto the contact normal $\mathbf{e}_j$. Since $(\mathbf{M}\mathbf{v}^*)_i = M_i \mathbf{v}_i^*$ ($\mathbf{M}$ diagonal), and $(\mathbf{v}_i^*)^\mathsf{T} \mathbf{e}_j = \left(\mathbf{R}_j \mathbf{u}_j^*\right)^\mathsf{T} \mathbf{e}_j = \mathbf{u}_{j|N}^*$ and $\xi_i^\mathsf{T} \mathbf{e}_j = (\mathbf{R}_j \mathbf{r}_j)^\mathsf{T} \mathbf{e}_j = \mathbf{r}_{j|N}$ (nodal contact), we obtain the scalar equation

$$M_i \mathbf{u}_{j|N}^* = \mathbf{d}_{j|N}(\mathbf{v}^*) + \mathbf{r}_{j|N}[\mathbf{d}_{j|N}(\mathbf{v}^*)] \quad \text{where } \mathbf{d}_{j|N}(\mathbf{v}^*) = \mathbf{f}_i(\mathbf{v}^*)^\mathsf{T}\mathbf{e}_j. \quad (13)$$

To obtain all admissible values for $\mathbf{u}_{j|N}^*$ and $\mathbf{r}_{j|N}^* := \mathbf{r}_{j|N}[\mathbf{d}_{j|N}(\mathbf{v}^*)]$, we consider two different cases. If $\mathbf{d}_{j|N}(\mathbf{v}^*) \geq 0$, then necessarily $\mathbf{r}_{j|N}^* = 0$, which leads to $\mathbf{u}_{j|N}^* \geq 0$ in (13): this corresponds to the take-off case. Otherwise ($\mathbf{d}_{j|N}(\mathbf{v}^*) < 0$), we have $\mathbf{r}_{j|N}^* = -\mathbf{d}_{j|N}(\mathbf{v}^*) \geq 0$, which yields $\mathbf{u}_{j|N}^* = 0$ in (13): this matches the contact case. No other cases are possible.

The same reasoning holds for the tangential part. We project our fixed point equation onto the tangential part, and examine the contact case. If $\|\mathbf{d}_{j|T}(\mathbf{v}^*)\| \leq -\mu_j \mathbf{d}_{j|N}(\mathbf{v}^*)$, then $\mathbf{r}_{j|T}^* = -\mathbf{d}_{j|T}(\mathbf{v}^*)$, yielding $\mathbf{u}_{j|T}^* = 0$ (stick). Otherwise we must have $\mathbf{r}_{j|T}^* = \mu_j \mathbf{d}_{j|N} \frac{\mathbf{d}_{j|T}}{\|\mathbf{d}_{j|T}\|}$, which gives $\mathbf{u}_{j|T}^* = \alpha \mathbf{d}_{j|T}$ with $\alpha > 0$, meaning that $\mathbf{u}_{j|T}^*$ is aligned and opposed to $\mathbf{r}_{j|T}^*$ (slide).