Incremental Potential Contact: Intersection- and Inversion-free, Large-Deformation Dynamics

MINCHEN LI, University of Pennsylvania & Adobe Research
ZACHARY FERGUSON and TESEO SCHNEIDER, New York University
TIMOTHY LANGLOIS, Adobe Research
DENIS ZORIN and DANIELE PANOZZO, New York University
CHENFANFU JIANG, University of Pennsylvania
DANNY M. KAUFMAN, Adobe Research

Fig. 1. **Squeeze out:** Incremental Potential Contact (IPC) enables high-rate time stepping, here with $h = 0.01$s, of extreme nonlinear elastodynamics with contact that is intersection- and inversion-free at all time steps, irrespective of the degree of compression and contact. Here a plate compresses and then forces a collection of complex soft elastic FE models (181K tetrahedra in total, with a neo-Hookean material) through a thin, codimensional obstacle tube. The models are then compressed entirely together forming a tight mush to fit through the gap and then once through they cleanly separate into a stable pile.

Contacts weave through every aspect of our physical world, from daily household chores to acts of nature. Modeling and predictive computation of these phenomena for solid mechanics is important to every discipline concerned with the motion of mechanical systems, including engineering and animation. Nevertheless, efficiently time-stepping accurate and consistent simulations of real-world contacting elastica remains an outstanding computational challenge. To model the complex interaction of deforming solids in contact we propose Incremental Potential Contact (IPC) – a new model and algorithm for variationally solving implicitly time-stepped nonlinear elastodynamics. IPC maintains an intersection- and inversion-free trajectory regardless of material parameters, time step sizes, impact velocities, severity of deformation, or boundary conditions enforced.

Constructed with a custom nonlinear solver, IPC enables efficient resolution of time-stepping problems with separate, user-exposed accuracy tolerances that allow independent specification of the physical accuracy of the dynamics and the geometric accuracy of surface-to-surface conformation. This enables users to decouple, as needed per application, desired accuracies for a simulation's dynamics and geometry.

The resulting time stepper solves contact problems that are intersection-free (and thus robust), inversion-free, efficient (at speeds comparable to or faster than available methods that lack both convergence and feasibility), and accurate (solved to user-specified accuracies). To our knowledge this is the first implicit time-stepping method, across both the engineering and graphics literature that can consistently enforce these guarantees as we vary simulation parameters.

In an extensive comparison of available simulation methods, research libraries and commercial codes we confirm that available engineering and computer graphics methods, while each succeeding admirably in custom-tuned regimes, often fail with instabilities, egregious constraint violations and/or inaccurate and implausible solutions, as we vary input materials, contact numbers and time step. We also exercise IPC across a wide range

Authors' addresses: Minchen Li, University of Pennsylvania & Adobe Research, minchernl@gmail.com; Zachary Ferguson; Teseo Schneider, New York University; Timothy Langlois, Adobe Research; Denis Zorin; Daniele Panozzo, New York University; Chenfanfu Jiang, University of Pennsylvania; Danny M. Kaufman, Adobe Research, dannykaufman@gmail.com.

0730-0301/2020/7-ART49 $15.00
https://doi.org/10.1145/3386569.3392425

of existing and new benchmark tests and demonstrate its accurate solution over a broad sweep of reasonable time-step sizes and beyond (up to $h = 2$s) across challenging large-deformation, large-contact stress-test scenarios with meshes composed of up to 2.3M tetrahedra and processing up to 498K contacts per time step. For applications requiring high-accuracy we demonstrate tight convergence on all measures. While, for applications requiring lower accuracies, e.g. animation, we confirm IPC can ensure feasibility and plausibility even when specified tolerances are lowered for efficiency.

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

Additional Key Words and Phrases: Contact Mechanics, Elastodynamics, Friction, Constrained Optimization

ACM Reference Format:
Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. 2020. Incremental Potential Contact: Intersection- and Inversion-free, Large-Deformation Dynamics. *ACM Trans. Graph.* 39, 4, Article 49 (July 2020), 20 pages. https://doi.org/10.1145/3386569.3392425

1 INTRODUCTION

Contact is ubiquitous and often unavoidable and yet modeling contacting systems continues to stretch the limits of available computational tools. In part this is due to the unique hurdles posed by contact problems. There are several intricately intertwined physical and geometric factors that make contact computations hard, especially in the presence of friction and nonlinear elasticity.

Real-world contact and friction forces are effectively discontinuous, immediately making the time-stepping problems very stiff, especially if the contact constraints are enforced exactly. On the other hand, even small violations of exact contact constraints (which are nonconvex) can lead to impossible to untangle geometric configurations, with a direct impact on physical accuracy and stability. In addition, stiff contact forces often lead to extreme deformations, resulting in element inversions for mesh-based discretization. Friction modeling then introduces further challenges with asymmetric force coupling and rapid switching between sliding and sticking modes.

In this work, our goal is to achieve very high robustness (by which we mean the absence of catastrophic failures or stagnation) for contact modeling even for the most challenging elastodynamic contact problems with friction. Robustness should be obtained independent of user-controllable accuracy in time-stepping, spatial discretization and contact resolution, while maintaining efficiency required to solve large-scale problems. At the same time we wish to also ensure that all accuracies – across the board – are efficiently attainable (of course with additional cost) when required.

With these goals in mind, we reexamine the contact problem formulation, discretization and numerical methods from scratch, building on numerous ideas and observations from prior work.

Our Incremental Potential Contact (IPC) solver is constructed for mesh-based discretizations of nonlinear volumetric elastodynamic problems supporting large nonlinear deformations, implicit time-stepping with contact, friction and boundaries of arbitrary codimension (points, curves, surfaces, and volumes). A key principle we follow is that while the physics and shape can be approximated arbitrarily coarsely, the geometric constraints (absence of intersections of the approximate geometry and inversions of elements) are maintained exactly at all times. We achieve this for essentially arbitrary target time steps and spatial discretization resolution.

The key element of our approach is the formulation of the contact problem and the customized numerical method to solve it. As a starting point, we use an exact contact constraint formulation, described in terms of an *unsigned* distance function, and rate-based maximal dissipation for friction.

For every time step, we solve the discrete nonlinear contact problem with a given tolerance using a smoothed barrier method, ensuring that the solution remains intersection-free at all intermediate steps. We use a comparably smoothed, arbitrarily close, approximation to static friction, also eliminating the need for an explicit Coulomb constraint, and cast friction forces at every time step in a dissipative potential form, using an alternating lagged formulation. All forces can then be solved by unconstrained minimization.

Our barrier formulation for contact has several important properties: 1) it is an almost everywhere C^2 function of the unsigned distances between mesh elements, C^1 continuous for a measure-zero set of configurations; 2) its support is restricted to a small part of the configuration space close to configurations with contact. The former property makes it possible to use rapidly converging Newton-type unconstrained optimization methods to solve the barrier approximation of the problem, the latter ensures that additional contact forces are applied highly locally *and* that only a small set of terms of the barrier function need to be computed explicitly during optimization. Jointly they enable stable, conforming contact between geometries.

To guarantee a collision-free state at every time step, feasibility is maintained throughout all nonlinear solver iterations: the line search in our customized Newton-based solver is augmented with efficient, filtered continuous collision detection (CCD) accelerated by a conservative CFL-type contact bound on line search step sizes. Friction forces are resolved directly in the same solver via our lagged potential with geometric accuracy improved by alternating updates.

1.1 Contributions

In summary, IPC solves nonlinear elastodynamic trajectories that are intersection- and inversion-free, efficient and accurate (solved to user-specified accuracies) while resolving collisions with both nonsmooth and codimensional obstacles. To our knowledge, this is the first implicit time-stepping method, across both the engineering and graphics literature, with these properties.

We demonstrate the efficacy of IPC with stress tests containing large deformations, many contact primitive pairs, large friction, tight collisions as well as sharp and codimensional obstacles. Our technical contributions include

- A contact model based on the unsigned distance function;
- An almost everywhere C^2, C^1-continuous barrier formulation, approximating the contact problem with arbitrary accuracy, with barrier support localized in the configuration space, enabling efficient time-stepping;
- Contact-aware line search that continuously guarantees penetration-free descent steps with CCD evaluations accelerated by a conservative-bound contact-specific CFL-inspired filter;

- A new variational friction model with smoothed static friction, formulated as a lagged dissipative potential, robustly resolving challenging frictional contact behaviors; and
- A new benchmark of simulation test sets with careful evaluation of constraint and time stepping formulations along with an extensive evaluation of existing contact solvers.

2 CONTACT MODEL

We focus on solving numerical time-integration for nonlinear volumetric elastodynamic models with contact. These models can interact with fixed and moving obstacles which can be of arbitrary dimension (surfaces, curves and points). The simulation domain is discretized with finite elements. Given n nodal positions, x, finite-element mass matrix, M, and a hyper-elastic deformation energy, $\Psi(x)$, the contact problem extremizes the extended-value action

$$S(x) = \int_0^{\mathrm{T}} \left(\frac{1}{2}\dot{x}^T M\dot{x} - \Psi(x) + x^T(f_e + f_d)\right)dt.$$

on an *admissible set of trajectories* $\mathcal{A}$, which we discuss below. Here f_e are external forces and f_d are dissipative frictional forces. We assume, for simplicity, that all object geometry is discretized with n-dimensional piecewise-linear elements, $n = 1, 2, 3$.

Admissible trajectories. We construct a new definition of admissibility based on unsigned distance functions that has a number of advantages. Most importantly, in the context of our work, it naturally allows us to formulate exact contact constraints in terms of constraints on collisions between pairs of primitives (triangles, vertices and edges), and can be defined in exactly the same way for objects of any dimensions (points, curves, surfaces and volumes).

Specifically we define trajectories $x(t)$, with $x \in \mathbb{R}^{3n}$ as *intersection-free*, if for all moments t, $x(t)$ ensures that the distance $d(p, q)$ between any distinct points p and q on the boundaries of objects is positive. In the space of trajectories, the set of intersection-free trajectories forms an open set $\mathcal{A}_I$, as it is defined by strict inequalities. Optimization problems may not have solutions in this set; for this reason, we add the limit trajectories to it, which involve contact. Specifically, we define the *set of admissible trajectories* $\mathcal{A}$ as the *closure* of $\mathcal{A}_I$. In other words, a trajectory is admissible, if it is intersection-free, *or* there is an intersection-free trajectory arbitrarily close.

Note that this closure is not equivalent to replacing the constraint $d(p, q) > 0$ with $d(p, q) \geq 0$; the latter is always satisfied for unsigned distances, so that all trajectories would be admissible. This is not the case for our definition. Consider for example, a point moving towards a plane. If its trajectory touches the plane and then turns back, an arbitrarily small perturbation makes it intersection-free, and the trajectory is in $\mathcal{A}$. However, if the trajectory crosses the plane small perturbations do not make it intersection-free. This highlights the need for our treatment even in the volumetric setting as the boundaries of our mesh upon which we impose constraint are exactly surfaces whose potential collisions include the point-face case above.

We can describe $\mathcal{A}_I$ directly in terms of constraints on unsigned distances d between surface primitives (vertices, edges, and faces in the simulation surface mesh and domain boundaries). We denote this set of mesh primitives $\mathcal{T}$. Equivalently to the more general definition above, a piecewise-linear trajectory $x(t)$ starting in an intersection-free state x_0 is admissible, if for all times t, the configuration $x(t)$ satisfies positive distance constraints $d_{ij}(x(t)) > 0$ for all $\{i, j\} \in \mathcal{B}$, where $\mathcal{B} \subset \mathcal{T} \times \mathcal{T}$ is the set of all non-adjacent and non-incident surface mesh primitive pairs.

We then observe that the distance between any pair of primitives is bounded from below by the distance for triangle-vertex and edge-edge pairs, *if* there are no intersections. For this reason, it is sufficient to enforce constraints $d_k(x(t)) > 0$ *continuously* in time, for all $k \in \mathcal{C} \subset \mathcal{B}$ where $\mathcal{C}$ contains all *non-incident point-triangle and all non-adjacent edge-edge pairs* in the surface mesh.

Time discretization. Discretizing in time, we can directly construct discrete energies whose stationary points give an unconstrained time step method's update [Ortiz and Stainier 1999]. Concretely, given nodal positions x^t and velocities v^t, at time step t, we formulate the time step update for new positions x^{t+1} as the minimization of an Incremental Potential (IP) [Kane et al. 2000], $E(x, x^t, v^t)$, over valid $x \in \mathbb{R}^{3n}$. For example the IP for implicit Euler is then simply

$$E(x, x^t, v^t) = \tfrac{1}{2}(x - \hat{x})^T M(x - \hat{x}) - h^2 x^T f_d + h^2\Psi(x), \qquad (1)$$

where h is the time step size and $\hat{x} = x^t + hv^t + h^2M^{-1}f_e$. IPs for implicit Newmark (see Section 7) and many other integrators follow similarly by simply treating their update rule as stationarity conditions of a potential with respect to variations of x^{t+1}.

Addition of contact constraints restricts minimization of the IP to admissible trajectories [Kane et al. 1999; Kaufman and Pai 2012] and so yields for our model the following time step problem:

$$x^{t+1} = \underset{x}{\operatorname{argmin}}\, E(x, x^t, v^t),\ x^\tau \in \mathcal{A}, \qquad (2)$$

where x^τ, $\tau \in [t, t+1]$, is the linear trajectory between x^t and x^{t+1}.

Our goal is to define a numerical method for approximating the solution of this problem in (2). Solving it is challenging due to the complex nonlinearity of the admissibility constraint, especially in the context of large deformations.

In turn, when frictional forces in (2) include frictional contact, solving the time step problem becomes all the more challenging as f_d is now governed by the Maximal Dissipation Principle [Moreau 1973] and so must satisfy further challenging, asymmetric and strongly nonlinear consistency conditions [Simó and Hughes 1998]. We present a friction formulation in Section 5 that is naturally integrated into this formulation via a lagged dissipative potential.

We further define a set of piecewise-linear surfaces as ϵ-separated, if the distance between two boundary points of the set is at least ϵ, unless these are on the same element of the boundary. An ϵ-separated trajectory is then a trajectory for which surfaces stay ϵ-separated. We denote the set of such trajectories $\mathcal{A}_\epsilon$.

To handle contact constraints, in our algorithm, we use the following overall approach: (a) the IP function E is unmodified on $\mathcal{A}_\epsilon$ – the set of trajectories for which any ϵ-separated trajectory extremizes the action are preserved; (b) we introduce a barrier term that vanishes for trajectories in $\mathcal{A}_\epsilon$ and diverges as the distance between any two boundary points vanishes, converting the problem to an unconstrained optimization problem. This barrier, together with

continuous collision detection within minimization steps, ensure that trajectories remain in $\mathcal{A}_I$.

This algorithm then guarantees that the trajectories are modified, compared to the exact solution, in an arbitrarily small, user-controlled (by ϵ) region near object boundaries and, at the same time, always remain admissible.

2.1 Trajectory accuracies

A discrete contacting trajectory is accurate if it satisfies 1) admissibility, 2) discrete momentum balance, 3) positivity, 4) injectivity and 5) complementarity.

In the discrete setting, *momentum balance* requires that the gradient of the incremental potential, $\nabla_x E(x)$, balance against the time-integrated contact forces. Its accuracy, is then exactly measured by the residual error in the optimization of the constrained incremental potential. We simply and directly control accuracy of momentum balance by setting stopping tolerance in our nonlinear optimization; see Section 4.5.

In turn *positivity* means that contact forces' signed magnitudes, λ_k, per contact pair $k \in C$, are always non-negative and so push surfaces but do not pull. Our method guarantees exact positivity.

Combined with *admissibility*, *injectivity* requires positive volumes for all tetrahedra in the simulation mesh. This invariant is enforced when a non-inverting energy density function, e.g. neo-Hookean, is modeled [1].

Finally, the classic definition of *complementarity* in contact mechanics [Kikuchi and Oden 1988] is the requirement that contact forces enforcing admissibility can only be exerted between surfaces if they are touching with no distance between them. We do not allow $d_k(x) = 0$, and so define a comparable measure of discrete ϵ-complementarity requiring

$$\lambda_k \max(0, d_k(x) - \epsilon) = 0, \ \forall k \in C \tag{3}$$

to measure how well contact accuracy is achieved. Discrete complementarity is then satisfied whenever distances between all contact pairs, defined as surface pairs with nonzero contact forces, are less than the ϵ and converge to complementarity as we reduce ϵ.

3 RELATED WORK

Computational contact modeling is a fundamental and long studied task in mechanics well covered from diverse perspectives in engineering, robotics and computer graphics [Brogliato 1999; Kikuchi and Oden 1988; Stewart 2001; Wriggers 1995]. At its core the contact problem combines enforcement of significant and challenging *geometric* non-intersection constraints with the resolution of a deformable solid's momentum balance. The latter task is well-explored, often independent of contact [Belytschko et al. 2000; Stuart and Humphries 1996]. We focus below on related works in defining contact constraints, implicitly time stepping with contact and friction, and barriers.

[1] When an invertible deformation model, e.g. fixed corotational, is modeled, injectivity need not be preserved in computation. We primarily focus on non-inverting neo-Hookean but will also demonstrate the weaker invertible case with fixed corotational.

3.1 Constraints and constraint proxies

Contact simulation requires a computable model of admissibility and so a choice of contact constraint representation. For volumetric models, admissibility generally begins with description of a signed distance function. This allows a clean formulation of the continuous model. However, when it comes to computing non-intersection on deformable meshes, choices for representing non-intersection must be made and a diversity of constraint representations exist. Contact constraints for deformable meshes, in both engineering [Belytschko et al. 2000; Wriggers 1995] and graphics [Bridson et al. 2002; Daviet et al. 2011; Harmon et al. 2009, 2008; Otaduy et al. 2009; Verschoor and Jalba 2019] are most commonly defined pairwise between matched surface primitives.

Existing methods most often define a local, signed distance evaluation using a diverse array of nonlinear proxy functions as well as their linearizations. These include linear gap functions, linearized constraints built from geometric normals, as well as a number of oriented volume constraints [Kane et al. 1999; Sifakis et al. 2008]. These nonlinear proxies, such as the tetrahedral volumes formed between surface point-face and edge-edge pairs, are only locally valid. They can introduce artificial ghost contact forces when sheared, false positives when rotated (e.g. for edge-edge tetrahedra), discontinuities when traversing surface element boundaries, and, in many methods, must still be further linearized and so introduce additional levels of approximation in order to solve a constrained time step.

Alternately gap functions and other related methods approximate signed distance functions for pairs of primitives by locally projecting a linearized distance measure between pairwise surface primitives onto a fixed geometric normal [Otaduy et al. 2009; Wriggers 1995]. As discussed in Erleben [Erleben 2018] these "contact point and normal" based constraint functions can be inconsistent over successive iterations and so are highly sensitive to surface and meshing variations with well known failure modes if care is not taken. Indeed, as we investigate in Section 8.3, even with iterative updates of these linear constraints inside SQP-type methods, time stepping with gap functions and related representations produces highly varied results whose success or failure is largely dependent on the scene simulated. In turn all of these challenges are only further exacerbated when simulations encounter the sharp, nonsmooth and even codimensional collisions imposed by meshed obstacles [Kane et al. 1999]; see e.g. Figure 2.

Recent fictitious domain methods [Jiang et al. 2017; Misztal and Bærentzen 2012; Müller et al. 2015; Pagano and Alart 2008; Zhang et al. 2005] offer a promising alternative. In these methods, motivated by global injectivity conditions [Ball 1981] negative space is separately discretized by a compatible discretization sometimes called an air-mesh [Müller et al. 2015].

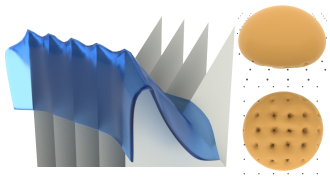

Fig. 2. **Nonsmooth, codimensional collisions.** Left: thin volumetric mat falls on codimensional (triangle) obstacles. Right: a soft ball falls on a matrix of point obstacles, front and bottom views.

Maintaining a non-negative volume on elements of this mesh then guarantees non-inversion. However, as with locally defined proxy volumes, the globally defined mesh introduces increasingly severe errors, e.g., shearing and locking forces, as it distorts with the material mesh. In 2D this can be alleviated by local [Müller et al. 2015] or global [Jiang et al. 2017] remeshing, however this is highly inefficient in 3D, does not provide a continuous constraint representation for optimization, nor, even with remeshing, can it resolve sliding and resting contact where air elements must necessarily be degenerate [Li et al. 2018].

Alternately, discrete signed distance fields (SDF) representations can be constructed via a number of approximation strategies over spatial meshes [Jones et al. 2006]. However, while state of the art adaptive SDF methods now gain high-resolution accuracy for sampling against a fixed meshes [Koschier et al. 2017], they can not yet be practically updated at rates suitable for deformable time steps, much less at rates suitable for querying deformations at every iterate within a single implicit time step solve [Koschier et al. 2017]. At the same time, discontinuities across element boundaries, while improved in recent works, still preclude smooth optimization methods.

We observe that while approximating signed distance pairwise between surface mesh elements is problematic, unsigned distance is well defined. We then design a new contact model for exact admissibility constraints in terms of unsigned distances between mesh-element pairs. This model of constraint is constructed sufficiently smooth to enable efficient, super-linear Newton-type optimization, maintains exact constraint satisfaction guarantees throughout all steps (time steps and iterations) and requires evaluation of just mesh-surface primitive pairs.

3.2 Implicit Time Step Algorithms for Contact

With choice of contact constraint proxy, $g(x) \geq 0$, the solve for the implicit time-step update is then the minimization of the contact-constrained IP [Doyen et al. 2011; Kane et al. 1999; Kaufman and Pai 2012],

$$\min_x E(x, x^t, v^t) \quad \text{s.t.} \quad g(x) \geq 0. \tag{4}$$

The variational problem (4), or its approximation is then minimized to compute the configuration at each time step.

This is typically done with off-the-shelf [Nocedal and Wright 2006] or customized constrained optimization techniques. In engineering, commonly used methods include sequential quadratic programming (SQP) [Kane et al. 1999], augmented Lagrangian and occasionally interior point methods [Belytschko et al. 2000]. All such methods iteratively linearize constraint functions and elasticity. However, both nonlinear constraint functions and their linearizations are generally valid only in local regions of evaluation and so can lead to intersections due to errors at larger time steps, faster speeds and/or larger deformations. For example Kane et al.'s [1999] volumes are only valid under a strong assumption of the relative position of contact primitive pairs.

In turn linearization of the full constraint set can also introduce additional error, result in infeasible sub-problems, locking and/or constraint drift [Erleben 2018]. This often requires complex and challenging (re-)evaluations of constraints in inter-penetrating states. Even when such obstructions are not present, iterated constraint linearization generally can not guarantee interpenetration-free state except upon convergence and so often must resort to small time steps and non-physical fail-safes in order to limit damage caused by missed constraint enforcement.

Although SQP- [Kane et al. 1999] and LCP/QP-based contact solvers [Kaufman et al. 2008] support and generally employ a variety of constraint-set culling and active-set update strategies, e.g., incrementally adding newly detected collisions at each iteration [Otaduy et al. 2009; Verschoor and Jalba 2019], they also can become infeasible and generate constraint drift when linearizing and filtering constraints.

Irrespective of how the contact-IP is solved and constraints are enforced, we then remain faced with combinatorial explosion in the number of contact constraints to handle. Determining the active set, i.e., finding which constraints are necessary for admissibility and so can not be ignored, remains an outstanding computational challenges. At the same time, to take large time steps or handle large deformation, we must resolve strongly nonlinear deformation energies in balance with contact forces. This requires line search. However, for constrained optimization methods, e.g., SQP, efficient line search in the presence of large numbers of active constraints remains an open problem [Bertsekas 2016; Nocedal and Wright 2006]. For this reason, existing methods in graphics currently avoid line search altogether and are, as a consequence, mostly restricted to quadratic energy models per time step [Otaduy et al. 2009; Verschoor and Jalba 2019] and, often, small time step sizes for even moderate material stiffness [Verschoor and Jalba 2019].

3.3 Friction

The addition of accurate friction with stiction only increases the computational challenge for time stepping deformation [Wriggers 1995]. Friction forces are tightly coupled to the computation of both deformation and the contact forces that prevent intersection. These side conditions are generally formulated by their own governing variational Maximal Dissipation Principle (MDP) [Goyal et al. 1991; Moreau 1973] and thus introduce additional nonlinear, nonsmooth and asymmetric relationships to dynamics. In transitions between sticking and sliding modes large, nonsmooth jumps in both magnitude and direction are made possible by frictional contact model. Asymmetry, in turn, is a direct consequence of MDP: frictional forces are not uniquely defined by the velocities they oppose, and are also determined by additional consistency conditions and constraints, e.g., Coulomb's law. One critical consequence is that there is no well-defined potential that we can add to an IP to directly produce contact friction via minimization.

To address these challenges, frictional contact is often solved by seeking a joint solution to the optimality conditions of MDP together with the discretized equations of motion (the latter are equivalent to optimality conditions for E). This requires, however, simultaneously solving for primal velocity unknowns together with a large additional number of dual contact and friction force unknowns. These latter variables then scale in the number of active contacts and their number grows large for even moderately sized simulation meshes.

To solve these systems it is generally standard to apply iterative per-contact, nonlinear Gauss-Seidel-type methods [Alart and Curnier 1991; Bridson et al. 2002; Daviet et al. 2011; Jean and Moreau 1992; Kaufman et al. 2014]. Here elasticity is again often, but not always, linearized per time step, while contact and friction constraints are similarly often approximated per iteration with a range of linear and nonlinear proxies. Alternate iteration strategies [Kaufman et al. 2008; Otaduy et al. 2009] have also been applied. However, as in the frictionless setting, all such splittings remain challenging to solve with guarantees for complex, real-world scenarios. Most recently, the same discrete formulation has been solved with new custom-designed algorithms – both with nonsmooth Newton-type strategies [Bertails-Descoubes et al. 2011; Macklin et al. 2019] and an extension of the Conjugate Residual method [Verschoor and Jalba 2019] with improved accuracy and efficiency.

3.4 Barrier Functions

Barrier functions are commonly applied in nonlinear optimization, especially in interior-point methods [Nocedal and Wright 2006]. Here primal-dual interior point methods are generally favored with Lagrange multipliers as additional unknowns for improved convergence. For contact problems, this impractically enlarges system sizes by orders-of-magnitude. Here we focus on a primal solution suited for contact problems. Similarly, the vast majority of the literature focuses on globally supported functions, which are not viable for contact due to the quadratic set (collision primitive pairs) of constraints that must be considered. Recently, a few works have begun exploiting *locally supported* barriers [Harmon et al. 2009; Schüller et al. 2013; Smith and Schaefer 2015]. Harmon et al. [2009] propose a set of layered discrete penalty barriers that grow unbounded as the configuration reaches toward contact. While well-suited for small time-step explicit methods, the incremental construction of the barriers challenge application in implicit time integration with Newton-type optimization. Most recently methods in geometry processing [Schüller et al. 2013; Smith and Schaefer 2015] propose locally supported barriers in the context of 2D mesh parametrization to prevent element inversion and overlap. Our formulation builds on a similar idea. Here we design smoothed, local barriers custom-constructed for the challenges of resolving contact-response and preventing intersection between 3D mesh-primitives.

3.5 Summary

In summary, state of the art methods for contact simulation are often highly effective per example. However, in order to do so they generally require significant hand-tuning per simulation set-up in order to obtain successful simulation output, i.e., stable, nonintersecting, plausible, or predictive output. Currently many of the tuned parameters, as we discuss in Section 8.3, are not physical but rather guided by expected intersection constraint violation errors and stability needs, and so need to be experimentally determined by many simulation test runs. Thus, to date, direct, fully automated simulation has not been available for contact simulation – despite contact's fundamental role in many design, engineering, robotics, learning and animation tasks. Towards a direct, "plug-and-play" contact simulation framework we propose IPC . Across a wide range of mesh resolutions, time step sizes, physical conditions, material parameters and extreme deformations we confirm IPC performs and completes simulations to requested accuracy without algorithm parameter tuning.

4 PRIMAL BARRIER CONTACT MECHANICS

In this section, we describe how we solve our time step problem (2) formulated in Section 2. We defer consideration of friction to Section 5, focusing on handling contact dynamics here. We solve the minimization problem (2), with primitive-pair admissibility constraints using a carefully designed barrier-augmented incremental potential that can be evaluated efficiently. In turn, to solve this potential we design a custom, contact-aware, Newton-type solver, outlined in Algorithm 1, with constraint culling for efficient evaluation of objectives, gradients and Hessians (Section 4.3).

Algorithm 1 Barrier Aware Projected Newton

procedure BarrierAwareProjectedNewton(x^t, ϵ)
  $x \leftarrow x^t$
  $\hat{C} \leftarrow$ ComputeConstraintSet($x, \hat{d}$) ▷ Section 4.6, 6.1
  $E_{\text{prev}} \leftarrow B_t(x, \hat{d}, \hat{C})$
  $x_{\text{prev}} \leftarrow x$
  do
    $H \leftarrow \text{SPDProject}(\nabla_x^2 B_t(x, \hat{d}, \hat{C}))$ ▷ Section 4.3
    $p \leftarrow -H^{-1} \nabla_x B_t(x, \hat{d}, \hat{C})$
    // CCD line search: ▷ Section 4.4
    $\alpha \leftarrow \min(1, \text{StepSizeUpperBound}(x, p, \hat{C}))$
    do
      $x \leftarrow x_{\text{prev}} + \alpha p$
      $\hat{C} \leftarrow$ ComputeConstraintSet($x, \hat{d}$)
      $\alpha \leftarrow \alpha/2$
    while $B_t(x, \hat{d}, \hat{C}) > E_{\text{prev}}$
    $E_{\text{prev}} \leftarrow B_t(x, \hat{d}, \hat{C})$
    $x_{\text{prev}} \leftarrow x$
    Update κ, BCs and equality constraints ▷ Supplemental
  while $\frac{1}{h}\|p\|_\infty > \epsilon_d$
  return x

4.1 Barrier-augmented incremental potential

To enforce distance constraints $d_k(t) > 0$, for all $k \in C$, we construct a continuous barrier energy b (Section 4.2), that creates a highly localized repulsion force, affecting motion only when primitives are close to collision, and vanishing if primitives are a small user-specified distance apart. We then augment the time step potential $E(x, x^t, v^t)$ with a sum of these barriers over all possible pairs in C. The barrier-augmented IP is then

$$B_t(x) = E(x, x^t, v^t) + \kappa \sum_{k \in C} b\big(d_k(x)\big), \tag{5}$$

with $\kappa > 0$ an adaptive conditioning parameter automatically controlling the barrier stiffness (see Section 4.3 and our Supplemental for details.).

Minimizing (5) enables the solution of contact-constrained dynamics with unconstrained optimization. Computing the energy

naively, however, would require evaluation of the barrier functions for all $O(|\mathcal{T}|^2)$ pairs. To address similar challenges many simulation methods simply remove constraints corresponding to distant primitives that are hoped to be unnecessary for the current solution. However, this tempting operation is dangerous, as significant errors and instabilities can be introduced when constraint sets are modified and critical collisions can also be missed (see Sections 3 and 8). Instead, we design smooth barrier functions that allow us to compute the barrier energy exactly and efficiently for all constraints while evaluating distances only for a small subset of pairs of primitives that are close and simultaneously ensuring that the rest smoothly evaluate to zero.

4.2 Smoothly clamped barriers

We begin by defining a smooth barrier function composed of terms that are *local* for every primitive pair, that is each term is exactly zero if the two primitives are far away, enabling reliable and efficient pruning of pairs in C without change to the solution.

We start by defining a computational distance accuracy target, $\hat{d} > 0$ (corresponding to ϵ in Section 2) that specifies the maximum distance at which contact repulsions can act. We then construct a barrier potential that approaches infinity at zero distance, initiates contact forces for pairs closer than the target distance, $\hat{d}$, and applies no repulsion at distances greater than $\hat{d}$.

Considering the smooth log-barrier function commonly applied in optimization [Boyd and Vandenberghe 2004] gives $\ln(d/\hat{d})$, where d is the unsigned distance evaluation between a primitive pair. However, simply truncating this function produces an unacceptably non-smooth energy which cannot be efficiently optimized and is effectively no better than simply discarding constraints. Some examples of problems this generates in optimization are covered in the supplemental. We thus propose a smoothly clamped barrier to regain superlinear convergence for Newton-type methods

$$b(d,\hat{d}) = \begin{cases} -(d-\hat{d})^2 \ln\left(\frac{d}{\hat{d}}\right), & 0 < d < \hat{d} \\ 0 & d \geq \hat{d}. \end{cases} \tag{6}$$

Our barrier function is now C^2 at the clamping threshold, and it is exactly zero for pairs beyond the target accuracy (see Figure 3). Now, without harm, at any configuration x, we only need to evaluate barrier terms for the *culled constraint set*

$$\hat{C}(x) = \{k \in C \,:\, d_k(x) \leq \hat{d}\},$$

composed of barriers between close primitives. As we increase accuracy by specifying smaller $\hat{d}$ we then need to evaluate increasingly smaller numbers of contact barriers, albeit with increased cost in nonlinearity.

Next, while the barrier function $b(d,\hat{d})$ itself is now C^2, the distance function it evaluates between primitives will be C^0 for certain unavoidable configurations; i.e., parallel edge-edge collisions – see Figure 9. For this reason, we multiply the barrier terms for edge-edge collisions by a mollifier that ensure our distance function is C^1 (and piecewise C^2) for all primitive pair types. Distance evaluation and mollifier are discussed in detail in Section 6. Additional important considerations related to numerical stability and roundoff error in distance evaluation are then detailed further in the Supplemental.

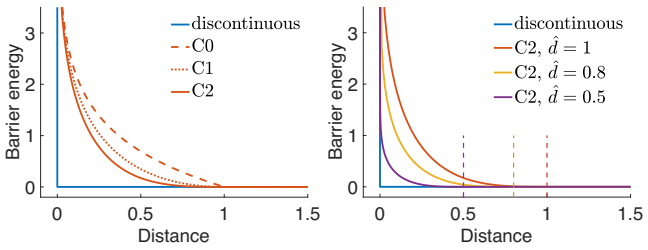

Fig. 3. **Barriers.** Left: log barrier function clamped with varying continuity. We can augment the barrier to make clamping arbitrarily smooth (see our Supplemental). We apply our C^2 variant for best tradeoff: smoother clamping improves approximation of the discontinuous function while higher-order continuity introduces more computational work. Right: our C^2 clamped barrier improves approximation to the discontinuous function as we make our geometric accuracy threshold, $\hat{d}$, smaller.

4.3 Newton-type barrier solver

Projected Newton (PN) methods are second-order unconstrained optimization strategies for minimizing nonlinear nonconvex functions where the Hessian may be indefinite. Here we apply and customize PN to the barrier-augmented IP (5). At each iteration, we project each local energy stencil's Hessian to the cone of symmetric positive semi-definite (PSD) matrices (see SPDProject function in Algorithm 1) prior to assembly. Specifically, following Teran et al. [2005] we project per-element elasticity Hessians to PSD. We then comparably project the Hessian of each barrier to PSD. Each barrier Hessian has the form

$$\frac{\partial^2 b}{\partial d^2}\nabla_x d(\nabla_x d)^T + \frac{\partial b}{\partial d}\nabla_x^2 d \tag{7}$$

and so can be constructed as a small matrix restricted to the vertices in the stencil of the barrier's primitive pair. The addition of mass matrix terms then ensures that the assembled total IP Hessian is symmetric positive definite (SPD). Originally we also investigated a Gauss-Newton approximation to the above barrier Hessian, taking only the first, SPD term in the sum. However, we find that resulting search directions are far less efficient than using the full projected barrier Hessian.

Termination. For termination of the solver we check the infinity norm of the Newton search direction scaled by time step (but *unscaled* by line-search step size). Specifically we solve each time step's barrier IP to an accuracy satisfying $\frac{1}{h}\|H^{-1}\nabla B_t(x)\|_\infty < \boldsymbol{\epsilon}_d$. This provides affine invariance and a characteristic measure using the Hessian's natural scaling as metric. Accuracy is then directly defined by $\boldsymbol{\epsilon}_d$ in physical units of velocity (and so is independent of time-step size applied) and consistently measures quadratically approximated distance to local optima across examples with varying scales and conditions.

Barrier stiffness adaptation. We automatically adapt our barrier stiffness to provide repulsive scaling that balances necessary distances against conditioning from the barrier stiffness. Our barrier-augmented potential, B_t, has two key parameters: $\hat{d}$ and κ, that jointly scale the effective stiffness of each contact barrier. The

strength of our barriers' contact forces (equivalently Lagrange multipliers) are directly determined during minimization by evaluating distances, d_k, and stiffness, κ. When κ is too small, contact-pair distances must become tiny to exert sufficient repulsion. On the other hand, when κ is too large, contact-pair distances must be below $\hat{d}$ in order to exert non-zero force, but at the same time remain exceedingly close to $\hat{d}$ so as to not exert too large a repulsion. Both cases thus generate unnecessary ill-conditioning and nonsmoothness that challenge efficiency. As we directly control geometric accuracy by setting $\hat{d}$, this frees κ to adaptively condition our Newton-solver to improve convergence. While conceptually one could imagine finding improved scalings of κ by hand, per example, this is unacceptable and inefficient for an automated simulation pipeline. Instead, in our Supplemental, we derive our stiffness update algorithm that automatically adapts barrier stiffness per iterate for improved conditioning.

Relation to homotopy solves. While in IPC we directly set and solve for a desired target accuracy $\hat{d}$, a natural alternative is to solve with a homotopy as is typical in interior point methods. We initially experimented with this approach: solving for larger distances (and so less stiff systems) and then decreasing to the target distance, $\hat{d}$, over successive nonlinear solves. We find, however, that this is unnecessary for elastodynamics where the direct barrier solves we employ are much more efficient. In part, this is because we typically have a good warm start available from the prior time step.

4.4 Intersection-aware line search

While our barrier energy is infinite for contact, this by itself does not guarantee that constraints $d_k(t) > 0$ are not violated by the solver. Standard line search [Nocedal and Wright 2006], e.g, back-tracking with Wolfe conditions, can find an energy decrease in configurations that have passed through intersection, resulting in a step that takes the configuration out of the admissible set.

Smith and Schaefer's [2015] line-search filter computes the largest step size in 2D per triangle and per boundary point-edge pair that first triggers inversion or overlap, and then take the minimum as a step size upper bound for the current Newton iteration to stay feasible. Taking inspiration from this line-search filter we propose a continuous, intersection-aware line search filter for 3D meshes. In each line search we first apply CCD to conservatively compute a large feasible step size along the descent step. We then apply back-tracking line search from this step size upper bound to obtain energy decrease. CCD then certifies that each step taken is always valid. When we apply barrier-based energy densities (our default) for our elasticity potential, Ψ, i.e., neo-Hookean, we combine the inversion-aware line search filter [Smith and Schaefer 2015] with our intersection-aware filter to obtain descent steps. In combination this guarantees that every step of position update in our solver (and so simulation) maintains an inversion- and intersection-free trajectory.

4.5 IPC solution accuracy

Revisiting accuracy we confirm *momentum balance* is directly satisfied by IPC after convergence. For example, for implicit Euler we have

$$\nabla_x B_t(x, \hat{d}) = 0 \implies M(\frac{x - \hat{x}}{h^2}) = -\nabla\Psi(x) + \sum_{k \in C} \lambda_k \nabla d_k(x), \quad (8)$$

where our contact forces λ_k are given by barrier derivatives

$$\lambda_k = -\frac{\kappa}{h^2} \frac{\partial b}{\partial d_k}. \quad (9)$$

Comparable discrete momentum balance follows when we apply alternate time integration methods, e.g. implicit Newmark. *Positivity* is then confirmed directly by (9) and observing that our barrier function definition guarantees $\frac{\partial b}{\partial d_k} \leq 0$. In turn our above line-search filters guarantee *admissibility* and, when applicable for barrier-type elasticity energy densities, *injectivity*. Finally, our barrier definition ensures *discrete complimentarity* is always satisfied as contact forces can not be applied at distance more than $\epsilon = \hat{d}$ away.

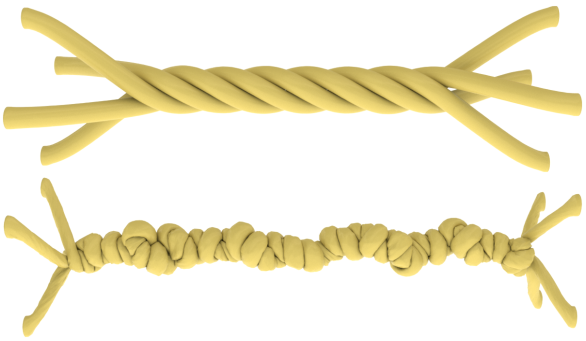

Fig. 4. **Extreme stress test: rod twist for 100s.** We simulate the twisting of a bundle of thin volumetric rod models at both ends for 100s. IPC efficiently captures the increasingly conforming contact and expected buckling while maintaining an intersection- and inversion-free simulation throughout. Top: at 5.5s, before buckling. Bottom: at 73.6s, after significant repeated buckling is resolved.

4.6 Constraint set update and CCD acceleration

Every line search, prior to backtracking, performs CCD to guarantee non-intersection, while every evaluation of energies and their derivatives compute distances to update the culled constraint set, $\hat{C}(x)$. To accelerate these computations, we construct a combined spatial hash and distance filtering structure to efficiently reduce the number of primitive-pair distance checks. Then, to further accelerate intersection-free stepping along each Newton iterate's descent direction, p, we derive an efficient conservative bound motivated by CFL conditions [Courant et al. 1967]. As in force evaluations we aim to avoid unnecessary and expensive CCD computation on primitive pairs *not in* $\hat{C}$. We leverage the fact that all contact pairs *not* in $\hat{C}$ are at distances greater than $\hat{d}$, and use the maximal relative search step in p of each such pair to obtain a conservative upper bound on step size. We then need only perform the CCD tests on primitive pairs in $\hat{C}$. This CCD culling generally provides an average 50% speed-up for all CCD costs across our simulations, with negligible increase in Newton iterations and an overall impact of 10% improvement in simulation times. Details on these accelerations and our adaptive

application of this bound (to avoid taking overly conservative steps) are detailed in our Supplemental.

5 VARIATIONAL FRICTION FORCES

Frictional contact adds contact-dependent dissipative forcing to our system. At macroscale these friction forces are modeled by the Maximal Dissipation Principle (MDP) [Moreau 1973]. MDP posits that frictional forces maximize rate of dissipation in relative motion directions orthogonal to contact constraints up to a maximum magnitude imposed by limit surfaces, e.g. as modeled by Coulomb's constraint [Goyal et al. 1991].

5.1 Discrete friction

To include frictional contact in our time stepping, we add local friction forces F_k for every active surface primitive pair, $k \in \hat{C}(x)$. For each such pair k, at current state x, we construct a consistently oriented sliding basis $T_k(x) \in \mathbb{R}^{3n\times 2}$. Each T_k is built so that $u_k = T_k(x)^T(x - x^t) \in \mathbb{R}^2$ gives the local relative sliding displacement at contact k, in the frame orthogonal to the distance vector between closest points on the two primitives defining $d_k(x)$. See Section 6 and our supplemental document for details on construction of $T_k(x)$. The corresponding sliding velocity is then $v_k = u_k/h \in \mathbb{R}^2$.

Maximizing dissipation rate subject to the Coulomb constraint defines friction forces variationally

$$F_k(x,\lambda) = T_k(x) \operatorname*{argmin}_{\beta\in\mathbb{R}^2} \beta^T v_k \quad \text{s.t.} \quad \|\beta\| \le \mu\lambda_k \tag{10}$$

where λ_k is the contact force magnitude and μ is the local friction coefficient.

Friction forces governed by (10) are bimodal. If $\|v_k\| > 0$, there is sliding and the corresponding friction force opposes it with $F_k = -\mu\lambda_k T_k(x)\frac{u_k}{\|u_k\|}$. If $\|v_k\| = 0$, there is sticking and the corresponding static friction force is $F_k = -\mu\lambda_k T_k(x) f$, where the friction direction f can take any value in the closed 2D unit disk.

5.2 Challenges to computation

Friction forces F_k are then challenging to solve for in three interconnected ways. First, F_k is *nonsmooth*. In transitions between sticking and sliding modes, nonsmooth jumps in both magnitude and direction are possible. Second, because of sticking modes, F_k in MDP is not uniquely defined by displacements until we have found a solution satisfying stationarity:

$$\nabla B_t(x) - h^2 \sum_{k\in C} F_k(x,\lambda) = 0. \tag{11}$$

Third, there is no well defined dissipation potential whose spatial gradient will generate friction forces. As a consequence, frictional contact forces do not naturally fit into variational time-stepping frameworks.

To tackle these challenges, we first examine F_k as a nonsmooth function of u_k. Next, as in our barrier treatment of contact, we smooth the friction function with controlled and bounded accuracy. Then, in order to apply friction as an energy potential in our variational solve, we lag updates of the sliding bases T_k and contact forces λ_k over nonlinear solves within each time step (or over time steps). This allows us to define a smooth dissipative potential for friction that can be consistently integrated into our Newton-type solver.

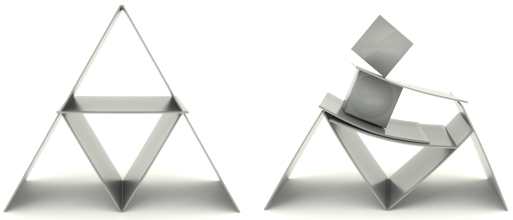

Fig. 5. **Friction benchmark: Stiff card house.** Left: we simulate a frictionally stable "card" house with $0.5m\times0.5m\times4mm$ stiff boards ($E = 0.1GPa$). Right: we impact the house at high-speed from above with two blocks; elasticity is now highlighted as the thin boards rapidly bend and rebound.

5.3 Smoothed static friction

During each of our Newton iterations any transitions of sliding displacements to or from sticking conditions will introduce large and sudden jumps in friction forces, F_k. These discontinuities, if left unmollified, would severely slow and even break convergence of gradient-based optimization; see Section 7. To enable efficient and stable optimization, we smooth the friction-velocity relation in the transition to static friction.

We start with a useful and equivalent (re-)expression for friction forces:

$$F_k = -\mu\lambda_k T_k(x) f(\|u_k\|)\, s(u_k), \tag{12}$$

with $s(u_k) = \frac{u_k}{\|u_k\|}$ when $\|u_k\| > 0$, while $s(u_k)$ takes any 2D unit vector when $\|u_k\| = 0$. The friction magnitude function, f, is then correspondingly nonsmooth with $f(\|u_k\|) = 1$ when $\|u_k\| > 0$, and $f(\|u_k\|) \in [0,1]$ when $\|u_k\| = 0$.

To smooth f and so (12) with bounded approximation error, we first define a velocity magnitude bound ϵ_v (in units of m/s) below which sliding velocities $v_k = u_k/h$ are treated as static. Then, we define a smoothed approximation of f with f_1. We maintain $f_1(y) = 1$ for all $y > h\epsilon_v$, (sliding) while for $y \in [0, h\epsilon_v]$, we require $f_1(y)$ to smoothly and monotonically transition from 1 to 0 over a finite range. This forms a bijective map from velocity magnitudes to friction magnitudes for velocities below the ϵ_v limit. For smoothing we experiment with satisfying interpolating polynomials ranging from C^0 to C^2. Increased continuity order introduces greater smoothing and faster error reduction for decreasing ϵ_v, at the cost of introducing greater nonlinearity into the IP solve. In the end, we find that our C^1 interpolant

$$f_1(y) = \begin{cases} -\frac{y^2}{\epsilon_v^2 h^2} + \frac{2y}{\epsilon_v h}, & y \in (0, h\epsilon_v) \\ 1, & y \ge h\epsilon_v, \end{cases} \tag{13}$$

provides best balance – yielding a continuous force Jacobian while introducing less nonlinearity and so fewer overall iterations in testing. See Figure 6 and our discussion in the Supplemental.

5.4 Variationally approximated friction

With a smooth and uniquely defined F_k for each u_k, we are now able to define friction forces solely based on nodal displacement

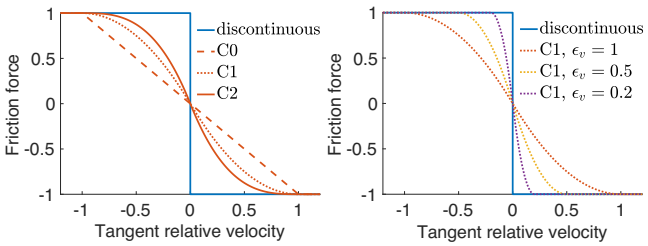

Fig. 6. **Friction smoothing in 1D.** Left: increasing orders of our polynomials better approximate the friction-velocity relation with increasing smoothness. Right: Our C^1 construction improves approximation to the exact relation as we make our frictional accuracy threshold, ϵ_v, and so the size of static friction zone, smaller.

unknowns. A next natural step would then be to define a so-called dissipative potential [Kane et al. 2000; Pandolfi et al. 2002] for inclusion in our optimization. An ideal potential would be a scalar function with respect to x whose gradient returns F_k. However, even with our smoothing, no well-defined displacement-based potential for friction exists, and F_k cannot be approximated by a potential force without introducing significant approximation errors. In other words, we do not have a variational form of friction that we can yet minimize.

We start by making dependence of our friction on both $T_k(x)$ and $\lambda_k(x)$ explicit:

$$F_k(x, \lambda_k, T_k) = -\mu\lambda_k T_k f_1(||u_k||)\frac{u_k}{||u_k||}. \tag{14}$$

Now, if we set $T_k = T_k(x)$ and $\lambda_k = \lambda_k(x)$ this friction evaluation is exact. However, if we decouple dependence of the evaluated sliding basis and contact force from x and instead lag them to values, λ^n, T^n, from a prior nonlinear solve (or previous time step) n, then all remaining terms in the expression for friction are integrable. The lagged friction force is then $F_k(x, \lambda_k^n, T_k^n)$ and provides a simple and compact friction potential,

$$D_k(x) = \mu\lambda_k^n f_0(||u_k||). \tag{15}$$

Here f_0 is given by the relations $f_0' = f_1$ and $f_0(\epsilon_v h) = \epsilon_v h$ so that $F_k(x) = -\nabla_x D_k(x)$. This potential provides easy-to-compute Hessian, $\nabla_x^2 D_k(x)$, and energy contributions to the barrier potential, described in detail in the supplemental document. Our full friction potential is then $D(x) = h^2 \sum_{k\in C} D_k(x)$, and the frictional barrier-IP potential for the time step $t + 1$ is

$$B_t(x, \hat{d}) + D(x). \tag{16}$$

Friction Hessian projection. For our Newton method (Section 4.3), we again need to project the friction potential Hessian to the space of PSD matrices. The friction Hessian structure is similar to that of elasticity, in that it can be written as a product of the T_k matrices. This allows us to apply the same strategy as used for elasticity Hessians, and so we need only perform a 2×2 PSD projection for each friction term per primitive pair. This is detailed in our Supplemental.

5.5 Frictional contact accuracy

Accuracy of friction forces generated by each solution of our IP (16) are defined by the static threshold, sliding basis and contact force magnitudes.

Static friction threshold. As we apply smaller ϵ_v we decrease the range of sliding velocities that we exert static friction upon and correspondingly sharpen the friction forces towards the exact non-smooth friction relation. Decreasing ϵ_v thus reduces stiction error while increasing compute times as we introduce a sharper nonlinearity in a tighter range; see Figure 6. For accurate reproduction of dynamic behaviors with friction and for visually plausible results, we observed that $\epsilon_v = 10^{-3}\ell m/s$, where ℓ is characteristic length (i.e. bounding box size), works well as a default across a wide range of examples with friction coefficients. See e.g., Figure 8. As static accuracy becomes important, we then find solutions with $\epsilon_v = 10^{-5}m/s$ work well. We have further confirmed IPC convergence down to $\epsilon_v = 10^{-9}m/s$. See, for example, our reproduction of the stable frictional contact structures in the masonry arch and card house examples in Figures 5 and 7.

Friction direction and magnitude. We improve accuracy of the direction and magnitude of the friction forces by solving successive minimizations of (16) within each time step. For each solve we update the lagged T^n and λ^n (warm-starting from the previous time step) with results from the last nonlinear solve. Convergence of lagged iterations is then achieved when we reach approximate momentum balance with

$$\|\nabla B_t(x^{t+1}) - h^2 \sum_{k\in C} F_k(x^{t+1}, \lambda^{t+1}, T^{t+1})\| \le \epsilon_d, \tag{17}$$

where ϵ_d is the targeted dynamics accuracy.

We confirm lagged iterations rapidly converge over nonlinear solves with our FE models for the well-known, standard frictional benchmarks, e.g., block-slopes, catenary arches and card houses. See Figures 7 and 5 and Section 7. However, we emphasize that we do not have convergence guarantees for lagging. In particular, we have identified cases with large deformation and/or high speed impacts where we do not reach convergence for T and λ in the friction

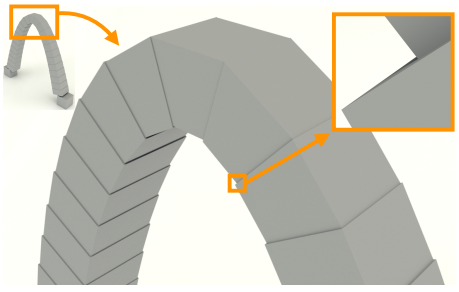

Fig. 7. **Friction benchmark: Masonry arch.** IPC captures the static stable equilibrium of a 20m high cement ($\rho = 2300kg/m^3$, $E = 20GPa$, $\nu = 0.2$) arch with tight geometric, $\hat{d} = 1\mu m$, and friction, $\epsilon_v = 10^{-5}m/s$ accuracy. Decreasing μ then obtains the expected instability and the stable arch does not form (see our supplemental videos). Inset: zoomed 100× (orange) highlights the minimal gaps with a geometric accuracy of small $\hat{d}$.

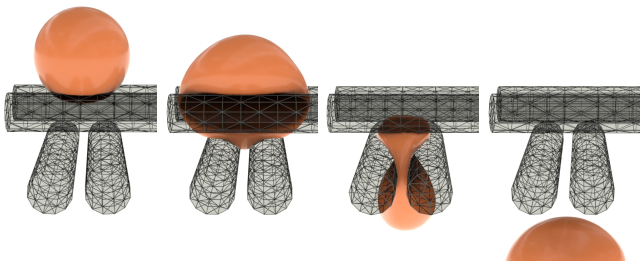

Fig. 8. **Large deformation, frictional contact test.** We drop a soft ball ($E = 10^4 Pa$) on a roller (made transparent to highlight friction-driven deformation). Here IPC simulates the ball's pull through the rollers with extreme compression and large friction ($\mu = 0.5$).

forces. Thus, in our large-deformation frictional examples we apply just a single lagging iteration. In these cases, sliding directions and contact force magnitudes in the friction force evaluation may not match. However, even in these cases, all other guarantees, including non-intersection, momentum balance (as in the frictionless case) and accurate stiction are maintained. More generally, we observe high-quality, predictive frictional results for large deformation examples independent of the number of lagging iterations applied; see e.g. Figure 16. We also emphasize that for frictionless models, IPC continues to guarantee convergence for contact and elasticity with just a single nonlinear solve per time step.

6 DISTANCE COMPUTATION

Evaluating unsigned distance functions between point-triangle and edge-edge pairs requires care as closed-form distance formulas change with relative position of surface primitives.

6.1 Combinatorial distance computation

Unsigned distances are given by the closest points on the two primitives evaluated.

Distance between a point v_P and a triangle $T = (v_{T1}, v_{T2}, v_{T3})$ can be formulated as a constrained optimization problem,

$$\begin{aligned} \mathcal{D}^{\mathrm{PT}} = &\min_{\beta_1,\beta_2} ||v_P - (v_{T1} + \beta_1(v_{T2} - v_{T1}) + \beta_2(v_{T3} - v_{T1}))|| \\ &s.t. \quad \beta_1 \geq 0, \quad \beta_2 \geq 0, \quad \beta_1 + \beta_2 \leq 1. \end{aligned} \tag{18}$$

Similarly the distance between edges v_{11}-v_{12} and v_{21}-v_{22} is

$$\begin{aligned} \mathcal{D}^{\mathrm{EE}} = &\min_{\gamma_1,\gamma_2} ||v_{11} + \gamma_1(v_{12} - v_{11}) - (v_{21} + \gamma_2(v_{22} - v_{21}))|| \\ &s.t. \quad 0 \leq \gamma_1, \gamma_2 \leq 1. \end{aligned} \tag{19}$$

Each possible *active set* of these two minimizations corresponds to a closed-form distance formula. In each, at most two constraints can be active at the same time.

- When *two* constraints are active in either (18) or (19), the distance between primitives is a point-point distance evaluation:

$$d^{PP} = ||v_a - v_b||. \tag{20}$$

  Here v_a and v_b correspond to v_P and a v_{Ti} for (18), or to the two endpoints of the edges in the edge-edge pair for (19).

- When a *single* constraint is active in either (18) or (19), the distance in both cases becomes a point-edge distance evaluation:

$$d^{PE} = \frac{||(v_a - v_c) \times (v_b - v_c)||}{||v_a - v_b||}. \tag{21}$$

  Here (v_a, v_b) corresponds to one of the triangle edges of T and $v_c = v_P$ for (18), or else (v_a, v_b) corresponds to one of the edges in the edge-edge pair and v_c corresponds to an endpoint of the other edge for (19).

- When *no* constraints are active in either (18) or (19), distance computations are simply parallel-plane distance evaluations. For the point-triangle pairing in (18) this is

$$d^{PT} = |(v_P - v_{T1}) \cdot \frac{(v_{T2} - v_{T1}) \times (v_{T3} - v_{T1})}{||(v_{T2} - v_{T1}) \times (v_{T3} - v_{T1})||}|, \tag{22}$$

  while for the edge-edge pairing in (19) it is

$$d^{EE} = |(v_{11} - v_{21}) \cdot \frac{(v_{12} - v_{11}) \times (v_{22} - v_{21})}{||(v_{12} - v_{11}) \times (v_{22} - v_{21})||}|. \tag{23}$$

For evaluations of d, ∇d, and $\nabla^2 d$, we apply the currently valid, closed-form distance formula (either PP, PE, PT, or EE above) and its analytic derivatives. The formula to apply, at each evaluation of a surface pair, is determined by the active constraint subset defined by the current relative positions of the pair's primitives. This information is computed and stored together with our culled constraint set $\hat{C}$ data, and so is then available for direct use whenever computing barrier energies and derivatives. This treatment is analogous to storing and reusing singular value decompositions of deformation gradients for elasticity computations. As in elasticity, our distance state and evaluations can efficiently be reused for all energy and derivative evaluations at the same nodal positions. Correspondingly, having now reduced general point-triangle and edge-edge distance evaluations to the above closed-form formulas, we can directly compute and store our sliding bases, $T_k(x)$, for friction computation with respect to each case; please see our Supplemental for details.

6.2 Differentiabilty of d

In collision-resolution methods, close-to-parallel edge-edge contacts are notorious failure modes – to the extent that existing methods often ignore this case by throwing out all corresponding constraints [Harmon et al. 2008]. However, despite the challenges imposed, these constraint cases cannot be removed, as doing so would lead to intersection. The reason for the difficulty in these cases is the (lack of) differentiability of the distance function for some configurations. Each above analytic formula for distances corresponds to a subset of the relative configuration space of

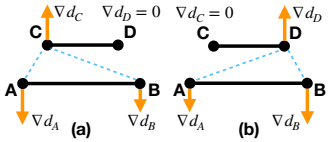

Fig. 9. **Nonsmoothness of parallel edge-edge distance.** When edge AB and CD are parallel, the distance computation can be reduced to either (a) C–AB point-edge or (b) D–AB point-edge. Then for the trajectory of C moving down from above D, the distance gradient is not continuous at the parallel point even though the distance is always continuously varying.

a primitive pair. For example, for vertex-triangle pairs, relative configurations are completely characterized by fixing the triangle and varying v_P positions. If the projection of v_P to T is in the triangle interior, no constraints are active, while if the projection lies on the interior of a triangle edge then one constraint is active. Otherwise, two constraints are active.

Each of these geometric criteria defines a subset of $\mathbf{R}^3$, where one of the three analytic formulas is valid. The distance function is C^∞ inside each such domain, and, in general, is C^1 at the boundaries between domains. However, the critical exception is in parallel edge-edge configurations: at these points, the distance function is not differentiable (see Figure 9). Configurations close to these parallel edge-edge conditions, when reached, lead to unacceptably slow convergence of Newton iterations or even convergence failures altogether. Numerically, the issue is similar to the C^0-continuous friction problem we faced in Section 5.2. To resolve this issue, we once again apply a local smoothing solution to mollify the barrier corresponding to nearly parallel edge-edge contact conditions.

We smooth by multiplying all edge-edge barrier terms by a piecewise-polynomial mollifier closely analogous to our static-friction smoother; recall Figure 6. Here, for each edge-edge contact pair k, we define $e_k(x)$ to vanish when edges $(v_{11}v_{12} - v_{21}v_{22})$ are parallel and to smoothly grow to 1 as the edge-pair become far from parallel,

$$e_k(x) = \begin{cases} -\frac{1}{\epsilon_\times^2}c^2 + \frac{2}{\epsilon_\times}c & c < \epsilon_\times, \\ 1 & c \geq \epsilon_\times, \end{cases} \tag{24}$$

where $c = ||(v_{12} - v_{11}) \times (v_{22} - v_{21})||^2$ and $\epsilon_\times = 10^{-3}||v'_{12} - v'_{11}||^2||v'_{22} - v'_{21}||^2$ is defined with respect to edge-edge vertex-pair rest positions v'.

Our mollified edge-edge barriers are then $e_k(x)b\big(d_k(x)\big)$ and so now extend our barrier potentials to a piecewise C^∞, *everywhere* C^1-continuous (for nonintersecting configurations) barrier formulation. At the same time our barriers now remain sufficient to guarantee that no collisions are missed: there are always point-triangle contact pairs at distance no more than the parallel edge-edge distance; see our Supplemental for details on this. In turn, our construction of the parallel-edge mollifier then minimizes its impact on edge-edge pair barriers as they move away from degeneracy. While in principle increasing smoothness to C^1 is sufficient to avoid most dramatic degeneracy failures, there are additional numerical stability issues to be addressed related to nearly parallel edges. Please see our Supplemental for details.

Now, with this third and last smoothing in place we have an overall time-stepping potential for contact and friction that can leverage superlinear convergence and robustness of Newton-type stepping. As we analyze in Sections 7 and 8 below (see especially 7.1) this gains robust simulation against failure – even when simulating challenging conditions with unavoidable numbers of degenerate evaluations.

7 EVALUATION

Our IPC code is implemented in C++, parallelizing assembly and evaluations with Intel TBB, applying CHOLMOD [Chen et al. 2008] with AMD reordering for linear system solves in all examples (with the exception of the squishy ball example – see below) and Eigen [Guennebaud et al. 2010] for linear algebra routines. We run most experiments on a 4-core 3.5GHz Intel Core i7, a 4-core 2.9 GHz Intel Core i7, and a 8-core 3.0 GHz Intel Xeon machine. Machine use per example is summarized along with performance statistics and problem parameters in Figure 23 and in our Supplemental. The reference implementation, scripts used to generate these results and our benchmarks are released as an open-source project.

Linear system computations and solves. We compute elasticity and barrier Hessians (with PSD projections) in parallel, and have designed and implemented a custom multi-threaded, sparse matrix data structure construction routine that, given the connectivity graph of nodes, efficiently builds the CSR format with index entries ready. While we utilize efficient symbolic factorization and parallel numerical factorization routines in CHOLMOD [Chen et al. 2008] compiled with MKL LAPACK and BLAS, we also tested IPC with AMGCL [Demidov 2019] – a multigrid preconditioned solver. Here, we found behavior is as might be expected, less memory overhead and faster linear solves by avoiding direct factorization. However, for majority of examples the large deformations and many contacts generate poorly conditioned systems. We then found AMGCL requires extensive parameter tuning to perform well and still can not compete, in general, with the parallel direct solver. All examples in the following then apply CHOLMOD for linear solves, with the exception of our largest, squishy ball example (Figure 22), where we apply AMGCL.

Models and practical considerations. We primarily employ the non-inverting, neo-Hookean (NH) elasticity model and implicit Euler time stepping. In the following examples we also apply and evaluate implicit Newmark time stepping, as well as the invertible fixed corotational elasticity (FCR) elasticity model. While for clarity in the preceding we derive IPC with unmodified distance evaluations, for numerical accuracy and efficiency our implementation applies squared distances for evaluations of the barrier, we use $b(d^2, \hat{d}^2)$, and related computations, thus avoiding squared roots. In turn expressions for contact forces, λ_k, and related terms must be modified, from our direct exposition and derivations above. To do so we rescale for consistent dimensions and units in our implementation; see our Supplemental for details. Finally and importantly we note that IPC's barrier formulation requires nonzero separation distances to be strictly satisfied at initialization and then guarantees it throughout simulation. Exact initialization at zero distance is neither possible (as the barrier of course diverges) nor for that matter physically meaningful. Contact, including resting contact, instead occurs around the specified geometric distance accuracy given by the user. Here we demonstrate simulated configurations with distances down to $10^{-8}m$ reached in simulation (e.g. arch in Figure 7) or initialized by users.

Evaluation and tests. Below we first introduce a set of unit tests for seemingly simple yet challenging scenarios with nonsmooth, aligned and close contacts (Section 7.1), stress tests involving large deformation and high velocities (Section 7.2), and friction (Section 7.3). We next study IPC 's scaling, run time, and accuracy behavior as we vary simulation problem parameters (Section 7.4). Finally, we

present an extensive, quantitative comparison with previous works in Section 8.

7.1 Unit tests

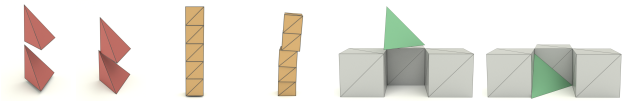

Fig. 10. **Aligned, close and nonsmooth contact tests.** Pairs of before and after frames of deformable geometries initialized with exact alignment of corners and/or asperities; dropped under gravity. We confirm both non-smooth and conforming collisions are accurately and stably resolved.

Aligned, close and nonsmooth contact. We apply a set of unit tests exercising closely aligned, conforming and nonsmooth contact known to stress contact algorithms. We build them with two simple models: a single tetrahedron and an 8-node unit cube; see Figure 10 and Figures 11. For contact handling, these seemingly simple tests are designed to trigger degenerate edge cases that often cause failure in existing methods (see Section 8). IPC resolves all cases including those in which we exercise *exact* parallel edge-edge (e.g., Figure 10 middle) and point-point (e.g., Figure 10, left) collisions. For unit tests like Figure 10 right we drop objects into slotted obstacles so that they fit tightly with tiny gaps; here IPC retrieves a tight conforming fit into a 1μm gap.

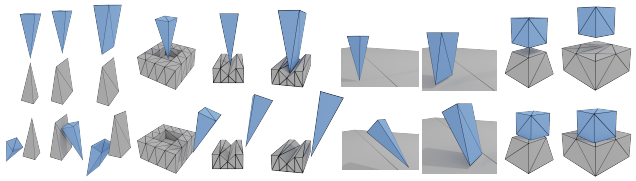

Fig. 11. **Erleben tests** Top: fundamental test cases to challenge mesh-based collision handling algorithms proposed by Erleben [2018]. Bottom: IPC robustly passes all these tests even when stepped at frame-rate size time steps.

Erleben fundamental cases. Erleben [2018] proposes unit tests (see Figure 11 top row) for contact constraint failure testing. Here these tests are again simple but designed to challenge mesh-based collision-handling algorithms. IPC again resolves all tests robustly (see Figure 11, bottom row), even when stepped at frame-rate size time steps.

Tunneling. Tunneling through obstacles when simulating high-speed velocities is a common failure mode in dynamic contact modeling. We thus add an example to our unit tests: we fire an elastic ball (diameter 0.1m) at a fixed 0.02m thin board at successive speeds of 10m/s, 100m/s, and 1000m/s stepped at $h = 0.02s$. IPC accurately rebounds at large time step without tunneling in all cases.

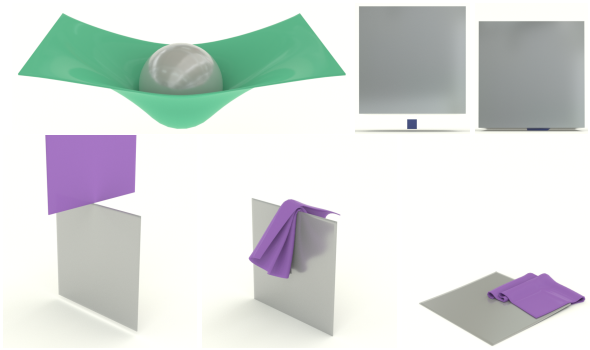

Fig. 12. **Large Mass and Stiffness ratios.**

Large mass and stiffness ratio tests. Contact resolution between objects with largely varying scale, mass, and/or stiffness ratios has long-challenged time stepping methods due to ill-conditioning. In Figure 12, we simulate IPC dropping of a range of objects upon each other with widely varying weight and stiffness. Here we apply $E = 0.1GPa$ for the sphere, board, and large cube, $E = 1MPa$ for the small cube and the mat holding the sphere, and $E = 10KPa$ for the mat dropped on boards. For the stiff ball and large cube, we set their respective densities to 2× and 10× that of softer objects ($1000kg/m^3$) to add large mass ratios to the challenge. Regardless of these different ill-conditioned settings, IPC simulates all scenes robustly and efficiently without any artifacts; see also our supplemental videos.

Chains. While resolving transient collisions exercises stability, large numbers of persistent, coupled contacts, as in a long chain of elastic links, exercises contact constraint accuracy. A small amount of constraint error integrated over time will cause such chains to break. We simulate chains of 100 elastic links under gravity, observe stable oscillations and shock-propagation while shorter chains stably bounce – all preserve constraints; see our supplemental videos.

7.2 Stress tests

We next consider IPC's ability to resolve a range of extreme stress-test examples motivated by well-known pre-existing challenges and previously proposed benchmarks.

Funnel. To confirm contact resolution under strong boundary conditions, extreme compression, and elongation, we pull a stiff NH material dolphin model through a codimensional funnel mesh obstacle. We step IPC at large time steps of $h = 0.04$s with up to 32.3K contacts per step. The resulting simulation is intersection- and inversion-free throughout with the model regaining its rest shape once pulled through (Figure 13).

Thin volumetric meshes. Thin geometries notoriously stress contact simulations. Likewise, as more simulation nodes are involved in collision stencils, simulation challenges grow. Here we test IPC's handling of extreme cases with both challenges, by simulating single layer meshes of tetrahedra. Here IPC robustly handles the contacts with accurate solutions at all time steps across a range of large deformation contact examples (Figures 5, 12 and 14).

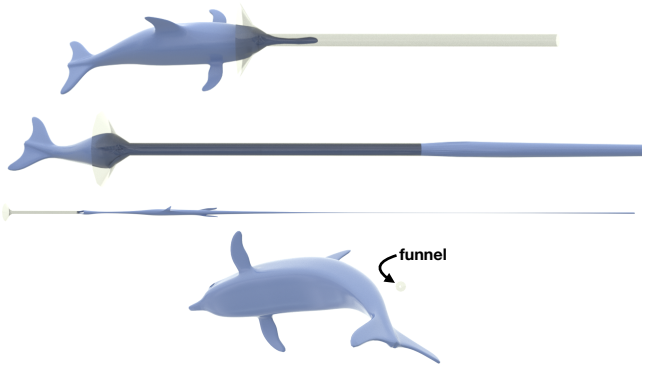

Fig. 13. **Funnel test.** Top: the tip of a stiff neo-Hookean dolphin model is dragged through a long, tight funnel (a codimensional mesh obstacle). Middle top: due to material stiffness and tightness of fit the tip of the model is elongated well before the tail pulls through. Middle bottom: extremity of the deformation is highlighted as we zoom out immediately after the model passes through the funnel. Bottom: finally, soon after pulling through, the dolphin safely recovers its rest shape. We confirm that the simulation is both intersection- and inversion-free throughout all time steps.

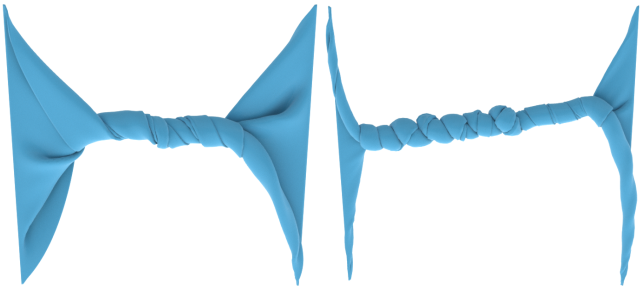

Fig. 14. **Stress test: extreme twisting of a volumetric mat for 100s.** Left: IPC simulation at 10s after 2 rounds of twisting at both ends. Right: at 40s after 8 rounds of twisting. This model, designed to stress IPC, has all of its 45K simulation nodes lying on the mesh surface.

Extreme and extended twisting. As large deformation high-contact examples, we twist thin mats (Figure 14), rods (Figure 4), and Armadillos (Figure 21, bottom) with rotating speeds of $72°/s$ at both ends. We simulate the twist of both the rods and mats for 100s – efficiently capturing increasingly tight conforming contact and expected buckling in all simulations.

Compactor test. In Figure 15 we test "trash" compactor-type examples from Harmon et al. [2009]. After releasing the compactor from the extreme compression point we clearly see that the tentacles of the octocat model and correspondingly the sphere, mat, and bunnies models are all cleanly separated.

Rollers compression and stick-slip instability. To combine extreme deformation with friction, we match the set-up of the kinematic roller test from Vershoor and Jalba [2019] with the same originally applied, high friction coefficient $\mu = 0.5$ (Figure 16). This scene is highly challenging due to the competing large magnitude of the friction and the large compression induced by the rollers. Here,

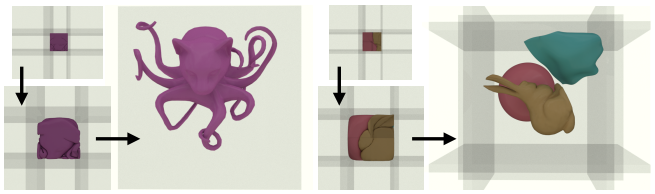

Fig. 15. **Trash Compactor.** An octocat (left) and a collection of models (right) are compressed to a small cube by 6 moving walls and then released. Here, under extreme compression IPC remains able to preserve intersection- and inversion- free trajectories solved to requested accuracies.

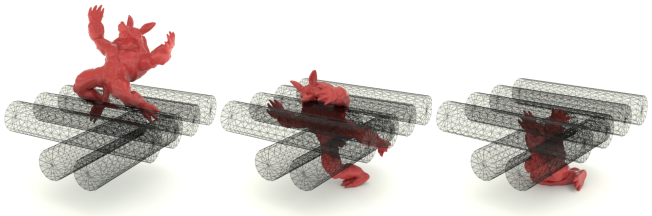

Fig. 16. **Roller tests.** Simulating the Armadillo roller from Verschoor and Jalba [2019] (same material parameters) in IPC now captures the expected stick-slip behavior for the high-friction, moderate stiffness conditions.

with a moderately stiff material ($E = 5 \times 10^5 Pa$ Young's modulus) we observe that IPC with our friction model obtains the expected stick-slip instability effects that such competition should generate. In simulation we observe deformation grows in opposition to static friction in the rollers until stress overcomes static friction and we observe slip – this process is then repeated. This stick-slip effect is captured by our Armadillo with moderate stiffness when tested with both the NH and FCR elasticity models (see our supplemental videos for the motion). We also note, as expected, when we subsequently test with softer material, i.e., $E = 10^5 Pa$, we get smooth rolling behavior for the Armadillo, as expected, without stick slip.

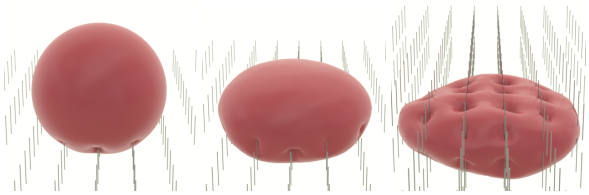

Fig. 17. **Codimensional collision objects: pin-cushions** We drop a soft ball onto pins composed of codimensional line-segments and then torture it further by pressing down with another set of codimensional pins to compress from above. IPC robustly simulates the ball to a "safe", stable resting state under compression against the pins.

Codimensional collision obstacles. Collision obstacles, especially in animation and gaming, are often most easily expressed in their default form as triangle meshes or even unorganized triangle soups. While highly desirable in applications, codimensional collision types are not generally supported by available simulation methods, which

often suffer tunneling, snagging, and resulting instabilities when exposed to them. To our knowledge IPC is the first algorithm to stably and accurately resolve collisions between volumes and codimensional collision objects. We perform a set of tests dropping different objects on planes, segments, and points, see e.g. Figures 2, 17 and 18. Collisions are stably resolved and we see tight compliance to the sharp poking obstacles in contacting regions.

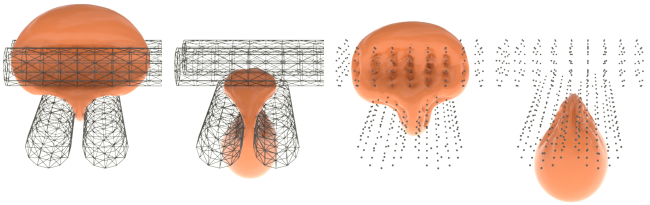

Fig. 18. **Codimensional collision objects: rollers** We modify the ball roller example from Figure 8 by using only the edge segments (left) or even just vertices (right) for the moving roller obstacle. For these extremely challenging tests IPC continues robust simulation exhibiting tight compliant shapes in contact regions pressed by the sharp obstacles.

Codimensional rollers. What if we modify the roller test in Figure 8, leaving only the edges or even only the points for the roller obstacles? This leads to our codimensional roller tests (Figure 18). Here, with solely codimensional wire (just edges) and points (just vertices) rollers, the big ball still pulls inwards, forming tightly pressed geometries in the contact regions as it is compressed and pulled against and then out of the codimensional rollers (Figure 18). For sharp point rollers we require negligible friction (for points we apply $\mu = 10^{-3}$) to pull the ball inwards as the sharp points directly grab the deforming surface.

Squeeze out stress test. A plate compresses and then forces a collection of complex soft material models into a tight conforming mush through a thin co-dimensional tube obstacle. Once through they cleanly separate (Figure 1).

High speed impact test. To examine IPC's fidelity in capturing high-speed dynamics we match the reported material properties and firing speed of an experiment of foam practice ball fired at high-speed towards a fixed steel wall. In Figure 19 top, we show key frames of a high-speed capture of the event. Middle: we visualize velocity magnitudes simulated by IPC , stepped with implicit Newmark and the NH material, at the same corresponding times in the simulation, and bottom the IPC-simulated geometry. Here we observe both the expected shockwave propagating through the sphere during the finite-time collision as well as the overall matching dynamics and shape across the simulation. Please see our supplemental video for complete simulation moving through the phases of inelastic collision impact: compression (first shockwave), restoration (second shockwave), and release.

7.3 Frictional contact tests

To examine IPC's frictional model we simulate a set of increasingly challenging frictional benchmark tests. All utilize a tight accuracy of $\epsilon_v = 10^{-5} m/s$ and apply lagged iterations to update sliding bases and normal forces until the system is confirmed as fully converged by satisfying (11).

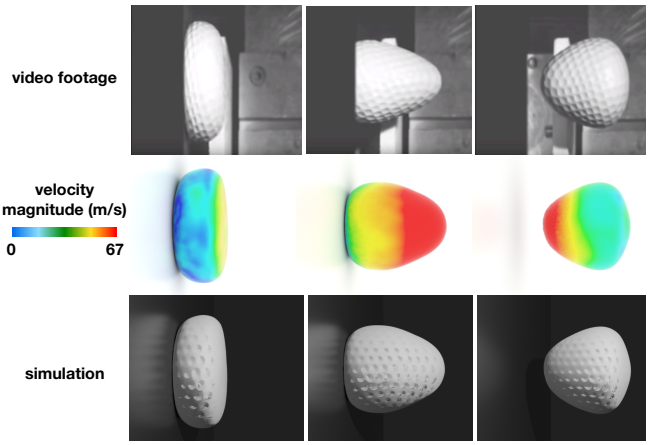

Fig. 19. **High-speed impact test.** Top: we show key frames from a high-speed video capture of a foam practice ball fired at a fixed plate. Matching reported material properties (0.04m diameter, $E = 10^7$Pa, $\nu = 0.45$, $\rho = 1150$kg/m^3) and firing speed ($v_0 = 67$m/s), we apply IPC to simulate the set-up with Newmark time stepping at $h = 2 \times 10^{-5}$s to capture the high-frequency behaviors. Middle and bottom: IPC-simulated frames at times corresponding to the video frames showing respectively, visualization of the simulated velocity magnitudes (middle) and geometry (bottom).

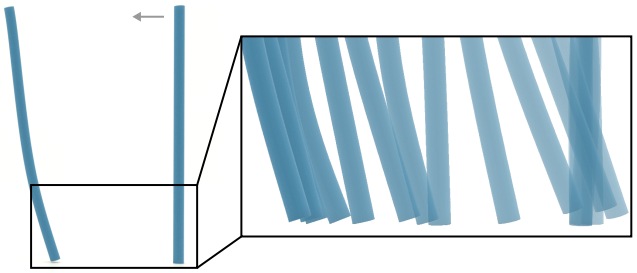

Fig. 20. **Stick-slip** oscillations with friction simulated with IPC by dragging an elastic rod along a surface.

Block tests. We start by placing stiff elastic blocks on a slope with tangent at 0.5. Here for $\mu = 0.5$, IPC generates the expected result of frictional equilibrium – the block does not slide. Switching to $\mu = 0.49$, IPC then immediately sets the block sliding, again matching the analytic solution.

Frictionally dependent structures. We test IPC on the challenging, frictionally dependent stable structure tests from Kaufman et al. [Kaufman et al. 2008]. We model both the card house (Figure 5) and masonry arch (Figure 7) with stiff deformable materials. We further extend the challenge of the arch with a precarious base balanced on sharp edges. We obtain long-term stable structures with $\mu = 0.5$ and $\mu = 0.2$ respectively and confirm that they fall apart as we reduce to $\mu = 0.2$ and $\mu = 0.1$ respectively (see our supplementals for statistics and videos).

Table 1. **Increasing time step sizes to frame-rate and beyond.** Here we demonstrate tradeoffs in varying time step sizes for the same tight twisted rods example (2.5K nodes, 6.9K tets, 4.6K faces, $E = 10^4 Pa$) with a 4-core 2.9GHz Intel Core i7 CPU, 16GB memory. # iters is the number of Newton iterations *per time step* or in *total* for the simulation sequences.

h (s)	# constraints	per time step		total	
	avg (max)	t (s)	# iters	t (s)	# iters
0.002	137 (430)	0.29	2.12	862	6351
0.005	194 (584)	0.36	2.37	435	2843
0.01	269 (707)	0.38	2.65	229	1591
0.025	435 (1.0K)	0.38	2.69	91	645
0.05	551 (1.2K)	0.46	3.06	56	368
0.1	597 (1.3K)	0.73	4.75	44	285
0.2	607 (1.2K)	1.79	14.37	54	431
0.5	653 (1.4K)	11.39	100.58	137	1207
1	708 (1.3K)	18.41	188.17	110	1129
2	843 (1.3K)	52.02	522.00	156	1566

Stick-slip instability. Finally, we script the motion of the top of a thin, volumetric elastic rod pushed slightly down towards, and then along a surface ($\mu = 0.35$) to test stick-slip oscillations. As in the Armadillo roller example, large static friction creates a buildup of elastic energy in the rod which is released when the friction force, opposing sliding contact, is exceeded by the tangential stiffness at the contact. This interaction between the friction forces and the sliding velocities becomes periodic, and so induces self-excited oscillations that buildup and dissipate energy; see Figure 20 and our supplemental video.

7.4 Scaling, Performance, and Accuracy

Varying time step sizes. Existing contact-resolution methods generally rely on small time step sizes for simulation success. As demonstrated above, IPC is able to simulate across a wide range of time step sizes h and so can capture a range of different frequency effects. Choice of time step size for IPC is then simply a question of accuracy required per application as balanced against efficiency needed, rather than a predicate required for success. To investigate the effect of varying time step size, h, in IPC we simulate the tight twisted rods example (Figure 4) for 6s. We range h from $0.002s$ to $2s$. In Table 1 we observe that transitioning from large to small time step sizes, our method improves its *per time-step* performance – but not by orders of magnitude. This is because the costs of intersection-free time stepping, distance computation and CCD do not change much. Since we do not miss any contacts, the number of constraints we process decrease only sublinearly as we decrease time step sizes. This is a key computational feature to ensure feasibility and robustness. On the other hand, we happily observe that our method is robust even well beyond standard time step sizes. While, in general, such excessively large step sizes beyond frame-rate are not useful for dynamics, this offers a robust opportunity for quasi-statically computing equilibria subject to challenging contact conditions. When we deploy IPC with implicit Euler IP (taking advantage of numerical dissipation), these very-large time steps rapidly compute equilibria with extreme contact conditions in just a few steps.

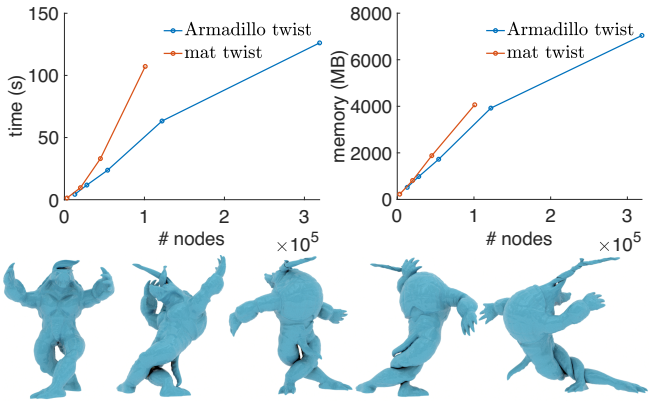

Fig. 21. **Scaling tests.** Top: applying increasing resolution meshes ranging from 3K to 219K nodes we examine the time (left) and memory (right) scaling behavior of IPC on a range of resolutions of the twisting Armadillo and twisting mat (Figure 14) examples. Bottom: frames from the highest-resolution twisting Armadillo example (219K nodes, 928K tets).

Scaling. In Figure 21 top, we study scaling behavior of IPC , with twisting mat (Figure 14) and twisting Armadillo (Figure 21, bottom) simulations of increasing-resolution meshes ranging from 3K to 219K nodes. Armadillo is a representative volumetric model while the single-layer mat is an extreme example designed to especially stress IPC . The mat meshes importantly have all simulation nodes on their surfaces and so, as contacts tighten in the twisting mat, they can form arbitrarily dense Hessians. For the mat we observe iteration count, memory and contact counts increase linearly with resolution, while timing increases in a slight superlinear trend. For the more standard volumetric Armadillo model we observe iteration count remains flat as we increase resolution, while timing and memory increase linearly.

In addition, when mesh sizes and contacts grow large, available memory can potentially preclude application of direct linear solvers. To confirm IPC applicability in these settings we simulate the firing of a 688K node, 2.3M tetrahedra, squishy ball model from Zheng and James [2012] at a glass board using AMGCL's [Demidov 2019] multigrid-preconditioned iterative linear solver. Here both the large element count and the large numbers of collisions enabled by the toy's many colliding tendrils introduce very large system solves during the most contact-rich steps colliding against the glass (Figure 22).

Performance. Comprehensive statistics on all simulations, models, parameters and performance are reported in Table 23 and in our Supplemental. For reference dynamics please see our supplemental videos.

Accuracy. User-facing parameters in IPC have three accuracies that can be specified: 1) dynamics accuracy (ϵ_d), defining how well dynamics are resolved; 2) geometric accuracy ($\hat{d}$), defining how close objects can come to touching; and 3) stiction accuracy (ϵ_v), defining how well static friction is resolved. All three provide users direct and intuitive control (with meaningful physical units) of the trade-off between accuracy and compute cost.

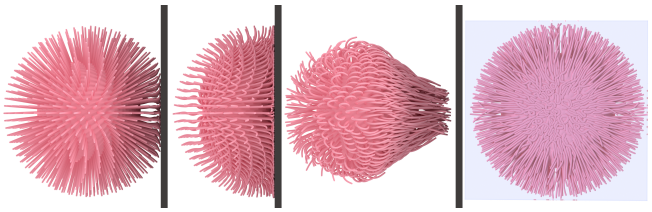

Fig. 22. **Squishy ball test:** Simulated by IPC an elastic squishy ball toy model (688K nodes, 2.3M tets) is thrown at a glass wall. The left three frames show side views before, at, and after the moment of maximal compression during impact. The right-most frame then shows the view behind the glass during the moment of maximal compression, highlighting how all of the toy's intricately intertwined tendrils remain intersection free.

In our extensive testing, IPC converges to satisfy these requested accuracies while always maintaining an intersection- and inversion-free state. These guarantees (non-intersection, non-inversion) hold even as we radically increase speed of collision at large time step, apply extreme deformations, and model highly stiff materials. We have tested this across a wide range of test examples with material stiffnesses up to $E = 2 \times 10^{11} Pa$ and have confirmed our IPC implementation's ability to converge to tight tolerances for all these measures when requested with ϵ_d down to 10^{-7}m/s, ϵ_v down to $10^{-8} m/s$, and $\hat{d}$ down to $1\mu m$.

As we discuss and demonstrate in Sections 3 and 8, all previously available methods introduce computational error for these accuracy measures; to our knowledge, IPC is the first to provide and expose direct and separable control of them. Our singular exception, as detailed in Section 5 above, is the number of frictional lagging iterations applied. When accurate friction is required, e.g., our arch, stick-slip and card house experiments, we set no upper bound on this parameter. Then as discussed above, in these examples IPC fully converges and is entirely parameter-free. However, (as detailed above) we do not have convergence guarantees for lagging, and in our large-deformation frictional examples we apply a single lagging iteration. In these cases, as discussed in Section 5.5, sliding directions and contact forces in the friction may not match. However, even in such cases all other guarantees, including non-intersection are maintained. We observe high-quality results regardless of number of lagging iterations applied or accuracies specified.

Finally, on the other end of the spectrum in many applications, e.g. animation, it can be desirable to trade accuracy for efficiency. We confirm robust, plausible behavior for IPC when we set very large, loose tolerances on all the above parameters, e.g. with $\epsilon_d = 10^{-1}$m/s, while still maintaining feasible (non-inverting, non-intersecting) trajectory guarantees.

Exact CCD admissibility check. IPC's collision aware line search ensures intersection-free trajectories. Our implementation applies standard floating-point CCD[2] combined with the conservative advancement strategies detailed in Section 4 and our Supplemental to ensure efficient, intersection-free stepping. Exact CCD then offers the possibility for aggressive advancement of intersection-free steps and so improved efficiency. To this end we tested the robust CCD methods from both Bridson et al. [2002] and Tang et al. [2014] but found the reference implementations for each missed critical intersections in degeneracies. We then reimplemented Bridson et al. [2002] with rationals. While this version now guarantees exactness, it is much slower (~30x) than our floating-point implementation. Currently we apply this exact CCD just for re-analysis as a post step check after every Newton iterate to test three of our challenging contact stress tests: octocat on codimensional "knives", ball roller and mat twist. We confirm that every step taken in every time step was intersection free in these examples.

[2]https://github.com/evouga/collisiondetection

Varying material model. A general expectation from unconstrained simulation is that modeling with non-invertible materials like NH should be more costly than comparable set-ups with invertible materials like FCR. However, when studying our large deformation examples with contact we find that the picture is more complex. Here the larger bottleneck is generally resolving contact barrier terms. In many examples we then observe that simulations with NH and FCR have comparable cost. In a number of other simulations with extreme contact conditions (e.g. pin-cushion and mat twist) element degeneracies allowed by FCR actually increase overall cost of simulation well over the same simulations with the NH material. Finally, in other cases where stress is most extreme (e.g. armadillo roller and dolphin funnel), NH entails more cost than the comparable simulation with FCR.

8 COMPARISONS

We perform extensive quantitative comparisons with existing algorithms and commercial codes used in both computer graphics (Section 8.1) and mechanical engineering (Section 8.2). Then, to more fairly compare across a large class of previous contacts algorithms based on SQP-type methods, we implement their core contact resolution procedures in a single framework, and perform a large scale comparison on our benchmark test set (Section 8.3). While our implementations are not finely tuned as for the first two sets of comparisons, this approach allows us to compare the core algorithmic components in a common, objective and unbiased context.

8.1 Computer Graphics Comparisons

Contact algorithms in graphics often target performance with small compute budgets and so admirably face many efficiency challenges in balancing fidelity against speed. We investigate what happens if we push these methods' settings to be most accurate without regard to speed, e.g., max iteration caps of 1M per step and time steps down to $10^{-5}s$. Here, nevertheless, we still document failures, e.g., tunneling, non-convergence, instabilities and ghost forces, even on very simple test examples.

Verschoor and Jalba [2019]. We apply the reference implementation [Verschoor and Jalba 2019] to reproduce available scenes from Verschoor and Jalba with their default and reported input parameters. Here we observe that small adjustments to time step sizes and material parameters lead to divergent simulations. Specifically, the Armadillo roller example does not converge when applying the implementation's default time step of $h = 10^{-3}$ for a range of stiffnesses of $E = 5 \times 10^4, 5 \times 10^5$, and 5×10^6Pa, nor when applying

the default material setting $E = 5 \times 10^5 Pa$ for a range of time step sizes of $h = 10^{-3}, 2 \times 10^{-3}, 4 \times 10^{-3}$, and 10^{-2}s. In all these cases the implementation maxes out at its default max-iteration cap of 1M.

We extract the Armadillo mesh, roller models and replicate the same example in IPC with identical scene settings. Here it is noteworthy that IPC applies fully nonlinear NH and FCR models with variational friction while the reference code (matching paper) linearizes elasticity once per time step. As covered in Section 7.2, IPC obtains the stick-slip oscillations expected in this setting (see also our video), when rolling the Armadillo. This does not match the Verschoor and Jalba reference code nor paper video. Artificial material softening due to the per-step linearization of Verschoor and Jalba's elasticity likely explains the difference. We confirm this in Section 7.2 where IPC's fully nonlinear simulation of the Armadillo roller, with a softer $E = 10^5 Pa$ (5× softer) does not, as expected, stick-slip.

SOFA. SOFA [Faure et al. 2012] is an open-source simulation framework featuring a range of physical models. These include deformable models via FEM. We modify a SOFA demo scene to simulate the five-link chain example with the top link fixed and four free FE links. We use the linear elasticity model (most robust) and found SOFA to provide a stable solution for the chain with large time steps up to $h = 10^{-2}$s. We extend the chain to ten links and are unable to find a converging time step size (tested down to $h = 10^{-4}$s). Please see our Supplemental for the full SOFA simulation settings.

Houdini. Houdini [SideFX 2020] is a widely used VFX tool that provides two performant simulation methods for deformable volumes: 1) a FE solver with co-rotated linear and neo-Hookean materials, and 2) Vellum, a state-of-the-art PBD solver. While capable of producing impressive effects – especially for rapid collision denting and bouncing, we find that both solvers suffer in different ways when enforcing contact constraints accurately is critical. As a simple demonstration we again apply the chain example.

Trying a simple, lower-stiffness, 5-link chain we aim Houdini's FE solver towards robustness over speed by finely tetrahedralizing the link rings (∼ 8000 tets per ring), applying small time steps (we tried increasing solver substeps to $h \approx 1ms$), and increasing collision passes (up to 16). Up to and including these maximum settings we observe rings tunneling through. We verify the same tunneling with both FE solvers provided in Houdini 18 (GNL,GSL), with both available materials. With similar stretchy material, IPC is able to accurately resolve the chain collisions even with a much coarser mesh (∼ 500 tets per ring), and frame-rate size time steps, e.g. $h = 0.04s$.

For the same 5-link scene, Houdini's Vellum PBD system does better, avoiding tunneling. However, as we increase numbers of links different tradeoffs (expected of PBD) are exposed. For example, a 35-link chain, requires collision passes and/or substeps to be increased quite high to prevent tunneling. However, this unavoidably changes the material (stiffer) and introduces biasing, in this case with sideways ghost forces. Careful experimentation with substep, smoothing, and constraint iteration parameters do not help alleviate these issues. For long chains (e.g. 100 links) we confirm IPC produces stable results, with accurate physical effects (e.g. shockwaves). See our supplemental videos.

8.2 Comparison with engineering codes

We compare IPC with two commercial engineering codes, COMSOL [2020] and ANSYS [2020], and one open-source engineering simulation framework [Krause and Zulian 2016]. For all three codes we set up exceedingly simple scenes involving small numbers of objects. All three methods generate intersection during simulation and exhibit instabilities highly dependent on parameters and tuning choices. In stark contrast to these three engineering solutions, IPC resolves a range of contact problems, demonstrates robust output across parameters, and ensures feasible trajectories. Please see our Supplemental for details on this comparison set.

8.3 Large scale benchmark testing with SQP-type methods

We focus on frictionless contact to compare a wide range of recently developed, implicit time-stepping algorithms. Removing the various and diverse treatments for friction allows us to carefully consider behavior with contact for a broad set of recent methods [Daviet et al. 2011; Harmon et al. 2008; Jean and Moreau 1992; Kane et al. 1999; Kaufman et al. 2008, 2014; Macklin et al. 2019; Otaduy et al. 2009; Verschoor and Jalba 2019] in a common test-harness framework. This is because all these methods, once friction is removed, follow a common iterated, Newton-type process to solve each time step as follows: 1) To help reduce constraint violation heuristic distance offsets/thickenings are applied to constraints; 2) at the start of each time step collision detection is performed to update a running estimate of active constraints; 3) The currently determined active (and possibly offset) constraint set and the IP energy are respectively approximated by first and second-order expansions; 4) The resulting quadratic energy is minimized subject to the linearized inequality constraints. This is a QP problem and so a bottleneck. A wide range of algorithms thus focus particularly on the efficient solution of this QP with custom approaches including QP, CR, LCP and nonsmooth-Newton strategies. Given the common sequential QP structure, we will jointly refer to them going forward as SQP-type. 5) A resulting displacement is then found and applied to the current iterate. This entire process is then repeated until a termination criteria is reached.

The above methods then differ in amount of offset, choice of constraint function, active set update strategy, IP approximations – most in graphics use just a fixed quadratic energy approximation (and so linearized elasticity) per time step, and choice of QP solver.

Here we focus on the ability of these methods to achieve convergent and accurate solves on a benchmark composed of our unit tests from Section 7.1 and a few additional low-resolution examples. To eliminate uncertainty of errors from the wide range of QP methods, we use the same state-of-the-art, albeit slow, QP solver Gurobi [Gurobi Optimization 2019] for all methods and test each simulation method across a grid of variations on an HPC cluster.

We implement three common constraint types: the projected gap function, see e.g., Harmon et al. [2008]; the volume based proxy of Kane et al. [1999]; and the CCD-based gap function, see e.g. Otaduy et al. [2009] and Vershoor and Jalba [2019]. For each constraint type we test on a 3D sweep of (a) time steps (10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}s), (b) constraint offsets (10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}), and (c) both fully nonlinear SQP and the graphics-standard of per time-step

fixed quadratic approximation of the elastic energy with nonlinear constraints.

A general pattern appears in our results (entire output is provided in the Supplemental): for simulations to succeed all methods require small time step and/or large constraint offset. With large time steps accuracy of the constraint linearization diminishes, thus larger constraint offsets are necessary to compensate for constraint violations. A too large constraint offset leads to failures as the local QP may become infeasible. Additionally, with large constraint offset, a constraint pair may initially violate the constraints (a common-case for self-collision due to arbitrarily small distances between elements). While it is possible to recover from such initial constraint violations, this rarely happens in our experiments. In contrast, we (re-)confirm IPC is unconditionally robust across all test cases and time steps in the benchmark.

9 CONCLUSION

In summary, IPC provides an exceedingly flexible, efficient, and unconditionally feasible solution for volumetric, mesh-based nonlinear elasticity simulations with self or external, volumetric or codimensional contacts. Guaranteeing intersection- and inversion-free output, IPC allows both computer graphics and engineering applications to run simulations by directly specifying just physically and geometrically meaningful parameters and tolerances as required per application.

At the same time much more remains to be done. While we have enabled a first of its kind "plug-and-play" contact simulation framework that provides convergent, intersection- and inversion-free simulation, clearly costs rise as scene complexity (both in contacts enforced and mesh resolutions) increase. There are thus many promising directions for future improvement that are exciting directions for exploration including further customized Newton-type methods, practical speed exact CCD, extensions to higher-order elements and improved convergence for frictional contact. We emphasize that we have no guarantee for convergence of lagged friction for λ and T (although we do for stiction) and so another meaningful avenue of future development is better exploration and understanding of its behavior.

Our hope is to enable engineers, designers, and artists to utilize predicative, expressive, and *differentiable* simulation, free from having to perform extra per-scene algorithmic tuning or deviation from real-world physical parameters. We look forward to enabling design, machine learning, robotics, and other processes reliant on automated and reliable simulation output across parameter sweeps and iterations and hope to better enable artists to use real-world materials and settings as useful design tools for creative exploration.

ACKNOWLEDGMENTS

We thank Xinlei Wang, Yu Fang and Francisca T. Gil-Ureta for valuable discussions, Yixin Zhu and NYU IT High Performance Computing for simulation and rendering run assistance and resources, and Mickeal Verschoor for generous assistance in generating comparisons. This work was supported in part by the NSF (CAREER Awards 1943199 and 1652515, and grants CCF-1813624, IIS-1320635, DMS-1436591, DMS-1821334, OAC-1835712, OIA-1937043, CHS-1908767, and CHS-1901091), DOE ORNL (subcontract 4000171342), and gifts from Houdini, nTopology, Adobe, NVIDIA, and AMD.

REFERENCES

Pierre Alart and Alain Curnier. 1991. A mixed formulation for frictional contact problems prone to Newton like solution methods. *CMAME* 92, 3 (Nov. 1991).
ANSYS Group. 2020. *ANSYS.* https://www.ansys.com/
John M Ball. 1981. Global invertibility of Sobolev functions and the interpenetration of matter. *Proc. of the Royal Society of Edinburgh: Section A Math.* 88, 3-4 (1981).
Ted Belytschko, Wing Kam Liu, and Brian Moran. 2000. *Nonlinear Finite Elements for Continua and Structures.* John Wiley & Sons, Ltd.
Florence Bertails-Descoubes, Florent Cadoux, Gilles Daviet, and Vincent Acary. 2011. A Nonsmooth Newton Solver for Capturing Exact Coulomb Friction in Fiber Assemblies. *ACM Trans. on Graph.* 30, 1, Article 6 (Feb. 2011).
Dimitri P. Bertsekas. 2016. *Nonlinear Programming.* Athena Scientific.
Stephen Boyd and Lieven Vandenberghe. 2004. *Convex Optimization.* Cambridge University Press, USA.
Robert Bridson, Ronald Fedkiw, and John Anderson. 2002. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. *ACM Trans. on Graph.* 21 (05 2002).
Bernard Brogliato. 1999. *Nonsmooth Mechanics.* Springer-Verlag.
Yanqing Chen, Timothy A Davis, William W Hager, and Sivasankaran Rajamanickam. 2008. Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Trans. on Math. Software (TOMS)* 35, 3 (2008).
COMSOL Group. 2020. *COMSOL Multiphysics.* https://www.comsol.com/
Richard Courant, Kurt Friedrichs, and Hans Lewy. 1967. On the partial difference equations of mathematical physics. *IBM J. of Research and Development* 11, 2 (1967).
Gilles Daviet, Florence Bertails-Descoubes, and Laurence Boissieux. 2011. A Hybrid Iterative Solver for Robustly Capturing Coulomb Friction in Hair Dynamics. *ACM Trans. on Graph.* 30 (12 2011).
Denis Demidov. 2019. AMGCL: An Efficient, Flexible, and Extensible Algebraic Multigrid Implementation. *Lobachevskii J. of Math.* 40, 5 (01 May 2019).
David Doyen, Alexandre Ern, and Serge Piperno. 2011. Time-Integration Schemes for the Finite Element Dynamic Signorini Problem. *SIAM J. on Sci. Comp.* 33 (01 2011).
Kenny Erleben. 2018. Methodology for Assessing Mesh-Based Contact Point Methods. *ACM Trans. on Graph. (TOG)* 37, 3 (2018).
François Faure, Christian Duriez, Hervé Delingette, Jérémie Allard, Benjamin Gilles, Stéphanie Marchesseau, Hugo Talbot, Hadrien Courtecuisse, Guillaume Bousquet, Igor Peterlik, et al. 2012. Sofa: A multi-model framework for interactive physical simulation. In *Soft tissue biomechanical modeling for computer assisted surgery.* Springer.
Suresh Goyal, Andy Ruina, and Jim Papadopoulos. 1991. Planar sliding with dry friction, Part 2. Dynamics of motion. *Wear* 143 (1991).
Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3.
LLC Gurobi Optimization. 2019. Gurobi Optimizer Reference Manual. http://www.gurobi.com
David Harmon, Etienne Vouga, Breannan Smith, Rasmus Tamstorf, and Eitan Grinspun. 2009. Asynchronous contact mnumpageechanics. In *ACM Trans. on Graph. (TOG)*, Vol. 28. ACM.
David Harmon, Etienne Vouga, Rasmus Tamstorf, and Eitan Grinspun. 2008. Robust Treatment of Simultaneous Collisions. *SIGGRAPH (ACM Trans. on Graph.)* 27, 3 (2008).
Michel Jean and Jean Jacques Moreau. 1992. Unilaterality and dry friction in the dynamics of rigid body collections. In *Proc. of Contact Mech. Int. Symp.*, Vol. 1.
Zhongshi Jiang, Scott Schaefer, and Daniele Panozzo. 2017. Simplicial Complex Augmentation Framework for Bijective Maps. *ACM Trans. on Graph.* 36, 6, Article 186 (Nov. 2017).
Mark W. Jones, J. Andreas Baerentzen, and Milos Sramek. 2006. 3D Distance Fields: A Survey of Techniques and Applications. *IEEE Trans. on Vis. and Comp. Graph.* 12, 4 (July 2006).
Couro Kane, Jerrold E Marsden, Michael Ortiz, and Matthew West. 2000. Variational integrators and the Newmark algorithm for conservative and dissipative mechanical systems. *Int. J. for Numer. Meth. in Eng.* 49, 10 (2000).
Couro Kane, Eduardo A Repetto, Michael Ortiz, and Jerrold E Marsden. 1999. Finite element analysis of nonsmooth contact. *CMAME* 180, 1-2 (1999).
Danny M Kaufman and Dinesh K Pai. 2012. Geometric numerical integration of inequality constrained, nonsmooth Hamiltonian systems. *SIAM J. on Sci. Comp.* 34, 5 (2012).
Danny M. Kaufman, Shinjiro Sueda, Doug L. James, and Dinesh K. Pai. 2008. Staggered Projections for Frictional Contact in Multibody Systems. *ACM Trans. on Graph. (SIGGRAPH Asia 2008)* 27, 5 (2008).
Danny M. Kaufman, Rasmus Tamstorf, Breannan Smith, Jean-Marie Aubry, and Eitan Grinspun. 2014. Adaptive Nonlinearity for Collisions in Complex Rod Assemblies. *ACM Trans. on Graph.* 33, 4, Article 123 (July 2014).

Fig. 23. **Simulation statistics** for IPC on a subset of our benchmark examples. Complete benchmark statistics are summarized in our supplemental documents. For each simulation we report geometry, time step, materials, accuracies solved to ($\hat{d}$, ϵ_d and ϵ_v are generally set w.r.t. to bounding box diagonal length l), number of contacts processed per time step, machine, memory, as well as average timing and number of Newton iterations per time step solve. When applicable, for friction we additionally report number of lagged iterations, with number of iterations set to $*$ indicating lagged iterations are applied until convergence until (17) is satisfied. We apply implicit Euler time stepping and the neo-Hookean material by default unless specified in example name; i.e., "NM" for implicit Newmark time stepping and "FCR" for the fixed-corotational material model.

Example	nodes, tets, faces	h(s)	ρ(kg/m^3), E(Pa), ν	$\hat{d}$(m)	μ, ϵ_v(m/s), friction iterations	ϵ_d(m/s)	contacts avg. (max.) (per timestep)	machine	memory (MB)	timing (s), iterations (per timestep)
Ball on points	7K, 28K, 10K	0.04	1000, 1e4, 0.4	1e-3l	N/A	1E-02l	126 (182)	4-core 2.9 GHz Intel Core i7, 16 GB memory	229	2.8, 6.6
Mat on knives	3.2K, 9.1K, 6.4K	0.04	1000 2e4, 0.4	1e-3l	N/A	1E-02l	291 (472)	4-core 2.9 GHz Intel Core i7, 16 GB memory	147	1.4, 5.5
100 chains	20K, 49K, 40K	0.04	500, 1e7, 0.4	1e-3l	N/A	1E-02l	40K (53K)	4-core 2.9 GHz Intel Core i7, 16 GB memory	450	4.0, 2.4
Dolphin funnel	8K, 36K, 10K	0.04	1000 1e4, 0.4	1e-3l	N/A	1E-02l	7K (31K)	4-core 2.9 GHz Intel Core i7, 16 GB memory	357	27.9, 39.7
Pin-cushion compress	9K, 28K, 10K	0.04	1000 1e4, 0.4	1e-3l	N/A	1E-02l	317 (496)	4-core 2.9 GHz Intel Core i7, 16 GB memory	233	3.7, 9.5
Golf ball (NM)	29K, 118K, 38K	2E-05	1150, 1e7, 0.45	1e-3l	N/A	1E-02l	1K (4K)	4-core 2.9 GHz Intel Core i7, 16 GB memory	861	12.1, 9.3
Mat twist (100s)	45K, 133K, 90K	0.04	1000 2e4, 0.4	1e-3l	N/A	1E-02l	264K (439K)	8-core 3.0 GHz Intel Xeon, 32GB memory	4,546	776.2, 34.5
Rods twist (100s)	53K, 202K, 80K	0.025	1000 1e4, 0.4	1e-3l	N/A	1E-02l	243K (498K)	8-core 3.0 GHz Intel Xeon, 32GB memory	2,638	141.5 ,14.1
Trash compactor: ball, mat and bunny	15K, 56K, 22K	0.01	1000 1e4, 0.4	1e-3l	N/A	1E-02l	6K (132K)	8-core 3.0 GHz Intel Xeon, 32GB memory	638	61.9, 29.4
Squeeze out	45K, 181K, 60K	0.01	1000, 5e4, 0.4	1e-3l	N/A	1E-02l	37K (277K)	8-core 3.0 GHz Intel Xeon, 32GB memory	1,700	252, 42.5
Ball mesh roller	7K, 28K, 11K	0.01	1000, 1e4, 0.4	1e-3l	0.5, 1e-3l, 1	1E-02l	2.3K (5.6K)	4-core 2.9 GHz Intel Core i7, 16 GB memory	215	63.3, 58.6
Hit board house	6K, 15K, 11K	0.025	1000, 1e8, 0.4	1e-4l	1.0, 1e-5l, 2	1E-02l	7K (13K)	4-core 2.9 GHz Intel Core i7, 16 GB memory	186	10.0, 16.6
Cement Arch	216, 150, 324	0.01	2300, 2e10, 0.2	1E-06	0.5, 1e-5l, *	1E-04l	101 (118)	4-core 3.6 GHz Intel Core i7, 32 GB memory	54	0.05, 5.7
Stick-slip Armadillo roller (FCR)	67K, 386K, 24K	0.025	1000, 5e5, 0.2	1e-3l	0.5, 1e-3l, 1	1E-02l	8K (33K)	4-core 3.6 GHz Intel Core i7, 32 GB memory	3,651	346, 66.8
Squishy ball (AMGCL)	688K, 2314K, 1064K	1E-03	1000, 7e4, 0.4	1e-4l	N/A	4E-02l	3.6K (105K)	8-core 3.6 GHz Intel Core i9, 64GB memory	19,463	328.3, 12.2

Noboru Kikuchi and John Tinsley Oden. 1988. *Contact Problems in Elasticity: A Study of Variational Inequalities and Finite Element Methods.* SIAM Studies in App. and Numer. Math., Vol. 8. Society for Industrial and Applied Mathematics.

Dan Koschier, Crispin Deul, Magnus Brand, and Jan Bender. 2017. An hp-Adaptive Discretization Algorithm for Signed Distance Field Generation. *IEEE Trans. on Vis. and Comp. Graph.* 23, 10 (2017).

Rolf. Krause and Patrick. Zulian. 2016. A Parallel Approach to the Variational Transfer of Discrete Fields between Arbitrarily Distributed Unstructured Finite Element Meshes. *SIAM J. on Sci. Comp.* 38, 3 (2016).

Minchen Li, Danny M. Kaufman, Vladimir G. Kim, Justin Solomon, and Alla Sheffer. 2018. OptCuts: Joint Optimization of Surface Cuts and Parameterization. *ACM Trans. on Graph.* 37, 6, Article 247 (Dec. 2018).

Miles Macklin, Kenny Erleben, Matthias Müller, Nuttapong Chentanez, Stefan Jeschke, and Viktor Makoviychuk. 2019. Non-Smooth Newton Methods for Deformable Multi-Body Dynamics. (07 2019).

Marek Krzysztof Misztal and Jakob Andreas Bærentzen. 2012. Topology-Adaptive Interface Tracking Using the Deformable Simplicial Complex. *ACM Trans. on Graph.* 31, 3, Article 24 (June 2012).

Jean Jacques Moreau. 1973. On Unilateral Constraints, Friction and PLasticity. *New Variational Tech. in Math. Phys.* (1973).

Matthias Müller, Nuttapong Chentanez, Tae-Yong Kim, and Miles Macklin. 2015. Air Meshes for Robust Collision Handling. *ACM Trans. on Graph.* 34, 4, Article 133 (July 2015).

Jorge Nocedal and Stephen Wright. 2006. *Numerical optimization.* Springer Science & Business Media.

Michael Ortiz and Laurent Stainier. 1999. The variational formulation of viscoplastic constitutive updates. *CMAME* 171, 3-4 (1999).

Miguel Otaduy, Rasmus Tamstorf, Denis Steinemann, and Markus Gross. 2009. Implicit Contact Handling for Deformable Objects. *Comp. Graph. Forum* 28 (04 2009).

Stéphane Pagano and Pierre Alart. 2008. Self-contact and fictitious domain using a difference convex approach. *Int. J. for Numer. Meth. in Eng.* 75 (07 2008).

Anna Pandolfi, Couro Kane, Jerrold E Marsden, and Michael Ortiz. 2002. Time-discretized variational formulation of non-smooth frictional contact. *Int. J. for Numer. Meth. in Eng.* 53, 8 (2002).

Christian Schüller, Ladislav Kavan, Daniele Panozzo, and Olga Sorkine-Hornung. 2013. Locally Injective Mappings. *Comp. Graph. Forum (Proc. of EUROGRAPHICS/ACM SIGGRAPH Symp. on Geom. Proc.)* 32, 5 (2013).

SideFX. 2020. *Houdini.* https://www.sidefx.com/products/houdini/

Eftychios Sifakis, Sebastian Marino, and Joseph Teran. 2008. Globally Coupled Collision Handling Using Volume Preserving Impulses. In *Proc. of the 2008 ACM SIGGRAPH/Eurographics Symp. on Comp. Anim. (SCA 2008).*

Juan C. Simó and Thomas J. R. Hughes. 1998. *Computational Inelasticity.* Springer.

Jason Smith and Scott Schaefer. 2015. Bijective parameterization with free boundaries. *ACM Trans. on Graph. (TOG)* 34, 4 (2015).

David E Stewart. 2001. Finite-dimensional contact mechanics. *Phil. Trans. R. Soc. Lond. A* 359 (2001).

Andrew Stuart and Anthony R Humphries. 1996. *Dynamical Systems and Numerical Analysis.* Cambridge Univ. Press.

Min Tang, Ruofeng Tong, Zhendong Wang, and Dinesh Manocha. 2014. Fast and exact continuous collision detection with bernstein sign classification. *ACM Trans. on Graph. (TOG)* 33, 6 (2014).

Joseph Teran, Eftychios Sifakis, Geoffrey Irving, and Ronald Fedkiw. 2005. Robust quasistatic finite elements and flesh simulation. In *Proc. of the 2005 ACM SIGGRAPH/Eurographics Symp. on Comp. Anim.* ACM.

Mickeal Verschoor and Andrei C Jalba. 2019. Efficient and accurate collision response for elastically deformable models. *ACM Trans. on Graph. (TOG)* 38, 2 (2019).

Peter Wriggers. 1995. Finite Element Algorithms for Contact Problems. *Archives of Comp. Meth. in Eng.* 2 (12 1995).

Eugene Zhang, Konstantin Mischaikow, and Greg Turk. 2005. Feature-Based Surface Parameterization and Texture Mapping. *ACM Trans. on Graph.* 24, 1 (Jan. 2005).

Changxi Zheng and Doug L. James. 2012. Energy-based Self-Collision Culling for Arbitrary Mesh Deformations. *ACM Trans. on Graph. (Proc. of SIGGRAPH 2012)* 31, 4 (Aug. 2012).