An Unconstrained Convex Formulation of Compliant Contact

Alejandro M. Castro, Frank N. Permenter, and Xuchen Han

Abstract—In this article, we present a convex formulation of compliant frictional contact and a robust performant method to solve it in practice. By analytically eliminating contact constraints, we obtain an unconstrained convex problem. Our solver has proven global convergence and warm-starts effectively, enabling simulation at interactive rates. We develop compact analytical expressions of contact forces allowing us to describe our model in clear physical terms and to rigorously characterize our approximations. Moreover, this enables us not only to model point contact but also to incorporate sophisticated models of compliant contact patches. Our time stepping scheme includes the midpoint rule, which we demonstrate achieves second order accuracy even with frictional contact. We introduce a number of accuracy metrics and show that our method outperforms the existing commercial and open-source alternatives without sacrificing accuracy. Finally, we demonstrate the robust simulation of robotic manipulation tasks at interactive rates, with accurately resolved stiction and contact transitions, as required for meaningful sim-to-real transfer. Our method is implemented in the open-source robotics toolkit Drake.

Index Terms—Contact modeling, dexterous manipulation, dynamics, simulation and animation.

I. INTRODUCTION

The simulation of multibody systems with frictional contact has proven indispensable in robotics, aiding at multiple stages during the mechanical and control design, testing, and training of robotic systems. Robotic applications often require robust simulation tools that can perform at interactive rates without sacrificing accuracy, a critical prerequisite for meaningful sim-to-real transfer. However, reliable modeling and simulation for contact-rich robotic applications remains somewhat elusive.

Rigid body dynamics with frictional contact is complicated by the nonsmooth nature of the solutions. It is well known [1] that rigid contact when combined with the Coulomb model of friction can lead to paradoxical configurations for which solutions in terms of accelerations and forces do not exist. These phenomena are known as Painlevé paradoxes [2]. Theory resolves these paradoxes by allowing discrete velocity jumps and impulsive forces, formally casting the problem as a differential variational inequality [3]. In practice, event-based approaches can resolve impulsive transitions [4], though it is not clear how to reliably detect these events even for simple one-degree-of-freedom (DOF) systems [2].

Nevertheless, the problem can be solved in a weaker formulation at the velocity level using a time-stepping scheme where the next step velocities and impulses are the unknowns at each time step [5], [6]. These formulations lead in general to a nonlinear complementarity problem (NCP). A linear complementarity problem (LCP) can be formulated using a polyhedral approximation of the friction cone, though at the expense of nonphysical anisotropy [7]. Even though LCP formulations guarantee solution existence [6], [8], solving them accurately and efficiently has remained difficult in practice. This has been explained partly due to the fact that these formulations are equivalent to nonconvex problems in global optimization, which...
are generally NP-hard [9]. Indeed, popular direct methods based on Lemke’s pivoting algorithm to solve LCPs may exhibit exponential worst-case complexity [10]. Similarly, popular iterative methods based on projected Gauss–Seidel (PGS) [11], [12] have also shown exponentially slow convergence [13]. These observations are not just of theoretical value—in practice, these methods are numerically brittle and lack robustness when tasked with computing contact forces. Software typically attempts to compensate for this inherent lack of stability and robustness through nonphysical constraint relaxation and stabilization, requiring a significant amount of application-specific parameter tuning.

A. Previous Work on Convex Approximations of Contact

To improve computational tractability, Anitescu [14] introduced a convex relaxation of the contact problem. This relaxation is a convex approximation with proven convergence to the solution of a measure differential inclusion as the time step goes to zero. For sliding contacts, the convex approximation introduces a gliding artifact at a distance $\phi$ that is proportional to the time step $\delta t$, friction coefficient $\mu$, and sliding velocity $\|v_t\|$ [15], i.e., $\phi \sim \delta t \mu \|v_t\|$. This artifact can be irrelevant for problems with lubricated contacts (as in many mechanisms) and goes away for sticking contacts (dominant in robotic manipulation). For sliding contact, the approximation can be adequate for applications for which the product $\delta t \mu \|v_t\|$ is usually sufficiently small. For trajectory optimization, Todorov [16] introduces regularization into Anitescu’s formulation in order to write a strictly convex formulation with a unique, smooth, and invertible solution. For simulation, Todorov [17] uses regularization to introduce numerical compliance that provides Baumgarte-like stabilization to avoid constraint drift. As a side effect, the regularized formulation can lead to a noticeable nonzero slip velocity even during stiction [18].

B. Available Software

Even though these formulations introduce a tractable approximation of frictional contact, they have not been widely adopted in practice. We believe this is because of the lack of robust solution methods with a computational cost suitable for interactive simulation. Software such as Open Dynamics Engine [19], Dart [20], and Vortex [21] use a polyhedral approximation of the friction cone leading to an LCP formulation. Algoryx [22] uses a split solver, reminiscent of one iteration in the staggered projections method [9]. RaiSim [23] implements an iterative method with exact solution per contact, a significant improvement over the popular PGS used in computer graphics, though still with no convergence nor accuracy guarantees. Drake [24] solves compliant contact with regularized friction with its transition-aware solver TAMSI [25].

To our knowledge, Chrono [26], MuJoCo [27], and Siconos [28] are the only packages that implement the convex approximation of contact. Chrono implements a variety of solvers for Anitescu’s convex formulation [14], including projected Jacobi and Gauss–Seidel methods [29], accelerated projected gradient descent [30], spectral projected gradients [31], and more recently alternating direction method of multipliers [32]. Though these methods are first order and often exhibit slow convergence, they are amenable to parallelization and have been applied successfully in the simulation of granular flows with millions of bodies. MuJoCo targets robotic applications. Its contact model is parameterized by a daunting number of nonphysical parameters aimed at regularizing the problem, though at the expense of drift artifacts [18]. Still, it has become very popular in the reinforcement learning community given its performance. Siconos is an open-source software targeting a large-scale simulation of both the rigid and deformable objects. The authors of Siconos perform an exhaustive benchmarking campaign in [33] using the convex approximation of contact. The analysis reveals that there is no universal solver and that every solver technology suffers from accuracy, robustness, and/or efficiency problems.

C. Outline and Novel Contributions

It is not clear if these convex approximations present a real advantage when compared to approaches solving the original nonconvex NCP problem and whether the artifacts introduced by the approximation are acceptable for robotics applications. In this article, we propose a new unconstrained convex approximation and discuss techniques for its efficient implementation. We carefully quantify the artifacts introduced by the convex approximation and evaluate the robustness and accuracy of our method on a family of examples.

We introduce a two-stage time-stepping approach (see Section II-C) that allows us to incorporate both the first- and second-order schemes such as the midpoint rule. In Section VI-B, we demonstrate that the midpoint rule can achieve second-order accuracy even in problems with frictional contact. Unlike previous work [17], [34] that formulates the problem in its dual form (impulses), we write a primal formulation of compliant contact in velocities (see Section III-A). We then analytically eliminate constraints from this formulation to obtain an unconstrained convex problem (see Section III-C).

To solve this formulation, we develop the semianalytic primal (SAP) solver in Section IV and study its theoretical and practical convergence. Crucially, we show that the SAP solver globally converges from all the initial conditions (see Appendix E) and warm-starts effectively using the previous time-step velocities, enabling simulation at interactive rates.

To address accuracy and model validity, Section V-A provides compact analytic formulas for the impulses that correspond to the optimal velocities of the convex approximation. This provides intuition for the approximation, even to those without optimization expertise. Moreover, the artifacts introduced by the approximation become apparent and can be characterized precisely. We provide an exact mapping between the regularization introduced by Todorov [17] to physical parameters of compliance. Therefore, regularization is no longer treated as a tuning parameter of the algorithm but as a true physical parameterization of the contact model. This allows us to incorporate not only compliant point contact but also complex models of
compliant surface patches [35, 36], as we demonstrate in Section VI-D.

We demonstrate the effectiveness of our approach in Section VI in a number of simulation cases, including the simulation of the challenging dual-arm manipulation task shown in Fig. 1. We evaluate the accuracy, robustness, and performance of the SAP solver against available commercial and open-source optimization solvers.

We discuss extensions and variations in Section VII and limitations in Section VIII. Finally, Section IX concludes this article.

II. MULTIBODY DYNAMICS WITH CONTACT

We use generalized coordinates (in particular joint coordinates) to describe our multibody system. Therefore, the state is fully specified by the generalized positions \( q \in \mathbb{R}^{n_q} \) and the generalized velocities \( v \in \mathbb{R}^{n_v} \), where \( n_q \) and \( n_v \) denote the number of generalized positions and velocities, respectively. Time derivatives of the configurations relate to generalized velocities by the kinematic map \( N(q) \in \mathbb{R}^{n_q \times n_v} \) as

\[
\dot{q} = N(q)v.
\]

A. Contact Kinematics

Given a configuration \( q \) of the system, we assume that our geometry engine reports a set \( C(q) \) of \( n_c \) potential contacts between pairs of bodies. We characterize the \( i \)-th contact pair in \( C(q) \) by the location \( p_i \), a normal direction \( n_i \), and signed distance function \( \phi_i(q) \in \mathbb{R} \), defined negative for overlapping bodies. The kinematics of each contact is further completed with the relative velocity \( v_{c,i} \in \mathbb{R}^3 \) between these two bodies at point \( p_i \), expressed in a contact frame \( C_i \) for which we arbitrarily choose the \( z \)-axis to coincide with the contact normal \( n_i \). In this frame, the normal and tangential components of \( v_{c,i} \) are given by \( v_{n,i} = n_i \cdot v_{c,i} \) and \( v_{t,i} = v_{c,i} - v_{n,i} n_i \), respectively, so that \( v_{c,i} = [v_{t,i}, v_{n,i}] \).

We form the vector \( v_c \in \mathbb{R}^{3n_c} \) of contact velocities by stacking velocities \( v_{c,i} \) of all the contact pairs together. In general, unless otherwise specified, we use bold italics for vectors in \( \mathbb{R}^3 \) and nonitalics bold for their stacked counterparts. The generalized velocities \( v \) and contact velocities \( v_c \) satisfy the equation

\[
v_c = Jv,
\]

where \( J(q) \in \mathbb{R}^{3n_c \times n_v} \) denotes the contact Jacobian.

B. Contact Modeling

We model the normal component of the impulse \( \gamma_n \) during a time interval of size \( \delta t \) with the compliant law

\[
\gamma_n/\delta t = (-k\phi - \tau_d k v_n)_{+}
\]

where \( k \) is the stiffness parameter, \( \tau_d \) is a dissipation time scale, and \( (x)_{+} = \max(0, x) \) is the positive part operator. \( \delta t \) is needed to convert forces into impulses. This model of compliance can be written as the equivalent complementarity condition

\[
0 \leq \phi + \tau_d v_n + \frac{c}{\delta t} \gamma_n \perp \gamma_n \geq 0
\]

where \( c = k^{-1} \) is the compliance parameter and \( 0 \leq a \perp b \geq 0 \) denotes complementarity, i.e., \( a \geq 0, b \geq 0 \), and \( a b = 0 \).

The tangential component \( \gamma_t \in \mathbb{R}^2 \) of the contact impulse is modeled with Coulomb’s law of dry friction as

\[
\gamma_t = \arg \min_{\|\xi\| \leq \mu \gamma_n} v_t \cdot \xi
\]

where \( \mu > 0 \) is the coefficient of friction. Equation (4) describes the maximum dissipation principle, which states that friction impulses maximize the rate of energy dissipation. In other words, friction impulses oppose the sliding velocity direction. Moreover, (4) states that contact impulses \( \gamma \) are constrained to be in the friction cone \( F = \{ [x_t, x_n] \in \mathbb{R}^3 | \|x_t\| \leq \mu x_n \} \). The optimality conditions for (4) are [29], [39]

\[
\mu \gamma_n v_t + \lambda \gamma_t = 0
\]

\[
0 \leq \lambda \perp \mu \gamma_n - \|\gamma_t\| \geq 0
\]

where \( \lambda \) is the multiplier needed to enforce Coulomb’s law condition \( \|\gamma_t\| \leq \mu \gamma_n \). Notice that in the form we write (5), multiplier \( \lambda \) is zero during sliding and takes the value \( \lambda = \|v_t\| \) during sliding. Finally, the total contact impulse \( \gamma \in \mathbb{R}^3 \) expressed in the contact frame \( C \) is given by \( \gamma = [\gamma_t \gamma_n] \).

C. Discrete Model

We base our time-stepping scheme on the \( \theta \)-method [40, Sec. II.7]. We discretize time into intervals of fixed size \( \delta t \) and seek to advance the state of the system from time \( t^n \) to the next step at \( t^{n+1} = t^n + \delta t \). In the \( \theta \)-method, variables are evaluated at intermediate time steps \( t^\theta = t^{n+1} + (1 - \theta) t^n \), with \( \theta \in [0, 1] \). We define mid-step quantities \( q^\theta, v^\theta \), and \( v^{\theta \perp} \) in accordance with the standard \( \theta \)-method using scalar parameters \( \theta \) and \( \theta^{\perp} \):

\[
q^\theta = \theta q + (1 - \theta) q_0
\]

\[
v^\theta = \theta v + (1 - \theta) v_0
\]

\[
v^{\theta \perp} = \theta v + (1 - \theta) v_0
\]

where, to simplify notation, we use the naught subscript to denote quantities evaluated at the previous time step \( t^n \), while no additional subscript is used for quantities at the next time step \( t^{n+1} \). Using these definitions, we write the following time-stepping scheme where the unknowns are the next time-step generalized velocities \( v \in \mathbb{R}^{n_v} \), impulses \( \gamma \in \mathbb{R}^{3n_c} \), and multipliers \( \lambda \in \mathbb{R}^{n_c} \):

\[
M(q^\theta(v))(v - v_0) = \delta t k(q^\theta(v), v^\theta(v)) + J(q_0)^T \gamma
\]

\[
0 \leq \phi(q(v)) + \tau_d,i v_{n,i}(q,v) + \frac{c_i}{\delta t} \gamma_{n,i}
\]

\[
\perp \gamma_{n,i} \geq 0,
\]

\[
\mu_i \gamma_{n,i} v_{t,i}(q,v) + \lambda_i \gamma_{t,i} = 0,
\]

\[
0 \leq \lambda_i \perp \mu_i \gamma_{n,i} - \|\gamma_{t,i}\| \geq 0
\]

\[
q^{\theta \perp}(v) = N(q^\theta(v))v^{\theta \perp}(v)
\]

\[
q(v) = q_0 + \delta t q^{\theta \perp}(v)
\]
where $M(q) \in \mathbb{R}^{n \times n}$ is the mass matrix and $k(q, v) \in \mathbb{R}^n$ models external forces such as gravity, gyroscopic terms, and other smooth generalized forces such as those arising from springs and dampers.

This scheme includes some of the most popular schemes for forward dynamics:

1) explicit Euler with $\theta = \theta_{eq} = 0$;
2) symplectic Euler with $\theta = 0$ and $\theta_{eq} = 1$;
3) implicit Euler with $\theta = \theta_{eq} = 1$;
4) symplectic midpoint rule, which is second order, with $\theta = \theta_{eq} = 1/2$.

Notice that in our version of the $\theta$-method, the additional parameter $\theta_{eq}$ allows us to also incorporate the popular symplectic Euler scheme.

When only conservative forces are considered in $k(q, v)$, the symplectic Euler scheme keeps the total mechanical energy bounded, while exact energy conservations can be attained with the second-order midpoint rule; see results in Section VI-B. In addition, stability analysis in [41] and [42] shows that these implicit schemes are appropriate for the integration of stiff forces arising in multibody applications such as springs and dampers.

### D. Two-Stage Scheme

Similar to the work in [11] for the simulation of deformable objects and to projection methods used in fluid mechanics [43], we solve (7)–(12) in two stages. In the first stage, we solve for the free motion velocities $v^*$ the system would have in the absence of contact constraints, according to

$$m(v^*) = 0$$

(13)

where we define the momentum residual $m(v)$ from (7) as

$$m(v) = M(q^0(v))(v - v_0) - \delta t k(q^0(v), v^0(v)).$$

(14)

For integration schemes that are implicit in $v^*$ (e.g., the implicit Euler scheme and the midpoint rule), we solve (13) with Newton’s method. For schemes explicit in $v^*$, only the mass matrix $M$ needs to be inverted, which can be accomplished efficiently using the $O(n)$ articulated body algorithm [44].

The second stage solves a linear approximation of the balance of momentum in (7) about $v^*$ that satisfies the contact constraints, (8)–(10). To write a convex formulation of contact in Section III, our linearization uses a symmetric positive-definite (SPD) approximation of the Jacobian $\partial m / \partial v$. To achieve this, we split the noncontact forces $k$ in (7) as

$$k(q, v) = k_1(q, v) + k_2(q, v)$$

such that the Jacobians $\partial k_1 / \partial q$ and $\partial k_1 / \partial v$ are negative-definite matrices, while the same is generally not true for the Jacobians of $k_2$. The term $k_1(q, v)$ can include forces from modeling elements such as spring and dampers. The term $k_2(q, v)$ includes all other contributions that cannot guarantee negative definiteness of their Jacobians, such as Coriolis and gyroscopic forces arising in multibody dynamics with generalized coordinates. We can now define the SPD approximation of $\partial m / \partial v$ evaluated at $v^*$ as

$$A = M + \delta t^2 \theta_{eq} K + \delta t \theta D$$

(15)

$$K = -\frac{\partial k_1}{\partial q} \frac{\partial \theta_{eq}}{\partial v}$$

(16)

$$D = -\frac{\partial k_1}{\partial v}$$

(17)

where $K \succ 0$ and $D \succ 0$ are the stiffness and damping matrices of the system, respectively. For joint level spring–damper models, $K$ and $D$ are constant, diagonal, and positive-definite matrices. Section VI-B shows the performance of our scheme for a system with a linear spring.

Using this SPD approximation of $\partial m / \partial v$, the linearized balance of momentum (7) reads

$$A(v - v^*) = J^T \gamma$$

(18)

where, for convenience, we use $J$ as a shorthand to denote $J(q_0)$. The approximation in (18) and the original discrete momentum update in (7) agree to second order as shown by the following result, proved in Appendix A.

**Proposition 1:** Matrix $A$ is a first-order approximation to the Jacobian of $m$, i.e.,

$$\left. \frac{\partial m}{\partial v} \right|_{v=v^*} = A + O(\delta t).$$

Therefore, (18) is a second-order approximation of the discrete balance of momentum in (7). Moreover, $A \succ 0$.

Notice that, in the absence of constraint impulses, the velocities at the next time step are equal to the free motion velocities, i.e., $v = v^*$, and they are computed with the order of accuracy of the $\theta$-method. Furthermore, we also expect to recover the properties of the $\theta$-method when contact constraints are in stiction. As an example, for bodies in contact that are under rolling friction, the contact constraints behave as bilateral constraints that impose zero slip velocity. In this case, our two-stage method with the midpoint rule exhibits considerably less numerical dissipation than first-order methods (see results in Section VI-B).

Even after the linearization of the balance of momentum in (18), the full problem with the contact constraints (8)–(10) still consists of a nonconvex NCP. These kinds of problems are difficult to solve in practice, especially so in engineering applications for which robustness and accuracy are required. In the next section, we introduce a number of approximations to this original NCP that allow us to make the problem tractable and solve it efficiently in practice.

### III. Convex Approximation of Contact Dynamics

In this section, we build from previous work on convex approximations of contact [14], [16], [17] to write a new convex formulation of compliant contact in terms of velocities.

**A. Primal Formulation**

We introduce a new decision variable $\sigma \in \mathbb{R}^{3n_v}$ and set up our convex formulation of compliant contact as the following optimization problem:

$$\min_{v, \sigma} \ell_p(v, \sigma) = \frac{1}{2} \|v - v^*\|^2_A + \frac{1}{2} \|\sigma\|^2_R$$

s.t. $g = (Jv - \dot{v}_c + R\sigma) \in F^*$

(19)
where \( \|z\|_X^2 = z^T X z \) with \( X \succ 0 \) and \( F^* = F_1^* \times F_2^* \times \cdots \times F_{n_c}^* \) is the dual cone of the friction cone \( F = F_1 \times F_2 \times \cdots \times F_{n_c} \), with \( X \) the Cartesian product. The positive diagonal matrix \( R \in \mathbb{R}^{3n \times 3n} \) and the vector of stabilization velocities \( \hat{v}_c \) encode the problem data needed to model compliant contact. We establish a very clear physical meaning for these terms in Section V-A. We note that when \( R \) and \( \sigma \) are removed, this reformulation reduces to [15].

We refer to (19) as the primal formulation and next derive its dual. To begin, define the Lagrangian

\[
L(v, \sigma, \gamma) = \frac{1}{2} \| v - v^* \|^2_A + \frac{1}{2} \| \sigma \|^2_R - \gamma^T g
\]

(20)

where \( \gamma \in F \) is the dual variable for the constraint \( g \in F^* \). Taking gradients of the Lagrangian with respect to \( v \) and \( \sigma \) leads to the optimality conditions

\[
A(v - v^*) = J^T \gamma
\]

(21a)

\[
\sigma = \gamma.
\]

(21b)

Note that (21a) is a restatement of the balance of momentum (18) if the dual variable \( \gamma \) is interpreted as a vector of contact impulses. Using the optimality condition (21b) to eliminate \( \sigma \) from the Lagrangian yields the dual formulation

\[
\min_{\gamma \in F} \ell_\delta(\gamma) = \frac{1}{2} \gamma^T (W + R) \gamma + r^T \gamma
\]

(22)

where \( W = JA^{-1}J^T \) is the Delassus operator and \( r = v^*_c - \hat{v}_c \), with \( v^*_c = J v^* \). In summary, we have proven the following theorem.

**Theorem 1:** The dual of (19) is given by (22). Moreover, when \( \{v, \sigma\} \) is primal optimal and \( \gamma \) is dual optimal, \( \sigma = \gamma \).

Finally, we note that the dual (22) is equivalent to the formulation in [16] when \( \theta = 0 \) in (7) and \( A = M(q_0) \).

### B. Analytical Inverse Dynamics

The dual optimal impulses of (22) can be constructed from the primal optimal velocities of (19) using a simple projection operation. Following [17], we call this construction analytical inverse dynamics. Moreover, this projection decomposes into a set of individual projections for each contact impulse \( \gamma_i \), given the separable structure of the constraints. Letting \( y_i(v_{c,i}) = -R_i^{-1}(v_{c,i} - \hat{v}_{c,i}) \), these projections form the take the form

\[
\gamma_i(v_{c,i}) = P_{J_i}(y_i(v_{c,i}))
\]

(23)

\[
= \arg \min_{\gamma_i \in F_i} \frac{1}{2} (\gamma_i - y_i)^T R_i (\gamma_i - y_i)
\]

where \( R_i \in \mathbb{R}^{3 \times 3} \) is the \( i \)-th diagonal block of the regularization matrix \( R \). That is, \( \gamma_i \) is the projection \( P_{J_i} \) of \( y_i(v_{c,i}) \) onto the friction cone \( F_i \) using the norm defined by \( R_i \). Remarkably, the projection map \( P_{J_i} \) can be evaluated analytically. We provide algebraic expressions for it in Section V-A and derivations in Appendix C. The projection \( P_{J_i}(y) \) onto the full cone \( F := F_1 \times F_2 \times \cdots \times F_{n_c} \) is obtained by simply stacking together the individual projections \( P_{J_i}(y_i) \) from (23), where we form \( y \) by stacking together each \( y_i \) from all the contact pairs. In this notation, the optimal impulse \( \gamma \) of (22) and the optimal velocities \( v \) of (19) satisfy \( \gamma = P_F(y(v)) \).

### C. Unconstrained Convex Formulation

We use the analytical \( \gamma = P_F(y(v)) \) from Section III-B and the optimality condition \( \sigma = \gamma \) from Theorem 1 to eliminate both \( \sigma \) and the constraints from the primal formulation (19). In total, we obtain the following unconstrained problem in velocities only:

\[
\min_v \ell_p(v) = \frac{1}{2} \| v - v^* \|^2_A + \frac{1}{2} \| P_F(y(v)) \|^2_R.
\]

(24)

The correctness of this reformulation is asserted by the following theorem, which we prove in Appendix B.

**Theorem 2:** If \( v \) solves the unconstrained formulation (24), then \( \{v, \sigma\} \), with \( \sigma = P_F(y(v)) \), solves the primal formulation (19).

Lemma 2 in Appendix E shows that the unconstrained cost \( \ell_p(v) \) is strongly convex and differentiable with Lipschitz continuous gradients. Therefore, the unconstrained formulation (24) has a unique solution and can be efficiently solved. Section IV presents our novel SAP solver specifically designed for its solution.

We outline our time-stepping scheme in Algorithm 1. We note that Algorithm 1 is executed once per time step, with no inner iterations updating \( A \). That is, our strategy is not solving the original, possibly nonlinear, balance of momentum (7) but its (second-order accurate) linear approximation in (18). This approximation can be exact for many multibody systems encountered in practice. For instance, joint springs and dampers contribute constant stiffness and damping matrices in (15).

### IV. SAP Solver

Inspired by Newton’s method, our SAP solver seeks to solve (24) by monotonically decreasing the primal cost \( \ell_p(v) \) at each iteration, as outlined in Algorithm 2.

The SAP iterations require \( H(v) \succ 0 \) at all the iterations. At points where \( \nabla_{v,\hat{v}} P_F(y(v)) \) is differentiable, \( H(v) \) is simply set equal to the Hessian of the cost function. In general, \( H(v) \)
is evaluated using a partition of its domain. For each set in the partition, \( \nabla_{v} \ell_{p}(v) \) is differentiable on the interior, and the Hessian admits a simple formula. We globally define \( H(v) \) by adopting one of these Hessian formulas on the boundary. The partition is described in Appendix C. The Hessian formulas are given in Appendix D. Our convergence analysis in Appendix E also accounts for this definition.

As shown in Appendix E, the SAP solver globally converges at least at a linear rate. Furthermore, the SAP solver exhibits quadratic convergence when \( \nabla^{2} \ell_{p} \) exists in a neighborhood of the optimal \( v \). In practice, we initialize the SAP solver with the previous time-step velocity \( v_{0} \). The stopping criterion is discussed in Section IV-E.

A. Gradients

We provide a detailed derivation of the gradients in Appendix D. Here, we summarize the main results required for implementation. The gradient of the primal cost \( \ell_{p} \) reduces to the balance of momentum

\[
\nabla_{v} \ell_{p}(v) = A(v - v^{*}) - J^{T} \gamma(v)
\]

where \( \gamma(v) = P_{F}(y(v)) \) is given by the analytical inverse dynamics (27). We define matrix \( G \geq 0 \) that evaluates to \(-\nabla_{v} \gamma\), where \( P_{F}(y(v)) \) is differentiable. Otherwise, \( G \) extends our analytical expressions, as outlined in Appendix D. Matrix \( G \) is a block diagonal matrix, where each diagonal block for the \( i \)-th contact is a \( 3 \times 3 \) matrix.

In total, we evaluate \( H \) via

\[
H = A + J^{T} G J
\]

which is strictly positive definite since \( A > 0 \).

B. Line Search

The line search algorithm is critical to the success of the SAP solver given that \( \nabla \ell_{p}(v) \) can rapidly change during contact-mode transitions. We explore two line search algorithms: an approximate backtracking line search with Armijo’s stopping criterion and an exact (to machine epsilon) line search.

At the \( m \)-th Newton iteration, the backtracking line search starts with a maximum step length of \( \alpha_{\text{Max}} \) and progressively decreases it by a factor \( \rho \in (0, 1) \) as \( \alpha \leftarrow \rho \alpha \) until Armijo’s criterion [45, Sec. 3.1] is satisfied. We write Armijo’s criterion as \( \ell_{p}(v^{m} + \alpha \Delta v^{m}) < \ell_{p}(v^{m}) + c \alpha \Delta \ell_{p}/d\alpha(dv^{m}). \) Typical parameters we use are \( \rho = 0.8, c = 10^{-4} \), and \( \alpha_{\text{Max}} = 1.25 \).

For the exact line search, we use the method \( \text{rtsafe} \) [46, Sec. 9.4] to find the unique root of \( d\ell/d\alpha \). This is a one-dimensional root finder that uses the Newton–Raphson method and switches to bisection when an iterate falls outside a search bracket or when convergence is slow. The fast computation of derivatives we show next allows us to iterate \( \alpha \) to machine precision at a negligible impact on the computational cost. In practice, this is our preferred algorithm since it allows us to use very low regularization parameters without having to tune tolerances in the line search.

C. Efficient Analytical Derivatives for Line Search

The algorithm \( \text{rtsafe} \) requires the first and second directional derivatives of \( \ell_{p} \). We show how to compute these derivatives efficiently in \( \mathcal{O}(n) \) operations. Defining \( \ell(\alpha) = \ell_{p}(v + \alpha \Delta v) \), we compute first and second derivatives as

\[
\frac{d\ell}{d\alpha} = \Delta v^{T} \nabla_{v} \ell(\alpha) \quad \quad \text{and} \quad \quad \frac{d^{2}\ell}{d\alpha^{2}} = \Delta v^{T} H(\alpha) \Delta v.
\]

Using the gradients from Section IV-A, we can write

\[
\frac{d\ell}{d\alpha}(\alpha) = \Delta v^{T} A(v(\alpha) - v^{*}) - \Delta v^{T} J^{T} \gamma(\alpha).
\]

These are computed efficiently by first calculating the change in velocity \( \Delta v_{c} := J \Delta v \) and change of momentum \( \Delta p := A \Delta v \). The calculation is then completed via

\[
\frac{d\ell}{d\alpha}(\alpha) = \Delta p^{T}(v(\alpha) - v^{*}) - \Delta v_{c}^{T} \gamma(\alpha)
\]

which only requires dot products that can be computed in \( \mathcal{O}(n_{c}) \) and \( \mathcal{O}(n_{c}) \), respectively. Similarly for the second derivatives

\[
\frac{d^{2}\ell}{d\alpha^{2}}(\alpha) = \Delta v^{T} \Delta p + \Delta v_{c}^{T} G(\alpha) \Delta v_{c}.
\]

Notice that the first term on the right-hand side can be precomputed before the line search starts, while the second term only involves \( \mathcal{O}(n_{c}) \) operations given the block diagonal structure of \( G \).

D. Problem Sparsity

The block sparsity of \( H \) is best described with an example. We organize our multibody systems as a collection of articulated tree structures, or a forest. Consider the system in Fig. 2. In this example, a robot arm mounted on a mobile base constitutes its own tree, here labeled \( t_{1} \). The number of DOFs of the \( t \)-th tree will be denoted with \( n_{t} \). A free body is a common case of a tree with \( n_{t} = 6 \). In general, matrix \( A \) has a block diagonal structure where each diagonal block corresponds to a tree.

We define as \( \text{patches} \) a collection of contact pairs between two trees. Each contact pair corresponds to a single cone constraint in our formulation. The set of constraint indexes that belong to such a tree is denoted with \( \mathcal{T}_{p} \) of size (cardinality) \( |\mathcal{T}_{p}| = r_{p} \). Fig. 2 shows the corresponding graph where nodes correspond to trees and edges correspond to patches.

Generally, the Jacobian is sparse since the relative contact velocity only involves velocities of two trees in contact (see the line search).
Fig. 3. Each nonzero block $J_{pi}$ has size $3r_p \times n_c$. Since $A$ is block diagonal, $H$ inherits the sparsity structure of $J^T G J$.

We exploit this structure using a supernodal Cholesky factorization [47, Section 9] that can take advantage of dense algebra optimizations. Implementing this factorization requires the construction of a junction tree. For this, we apply the algorithm in [48], using cliques of $H$ as input. We use the implementation from the Conex solver [49].

The scalability of the SAP solver, like any second-order optimization method, depends on the complexity of solving the Newton system. For dense problems, this is $O(n_c^3)$ complexity, where $n_c$ denotes the number of generalized velocities. For sparse problems, the complexity can be dramatically reduced [47]. We study scalability with a number of bodies in Section VI-C. Recent work on the modeling of contact rich patches [36] studies the scalability of the SAP solver with the number of constraints.

E. Stopping Criterion

To assess convergence, we monitor the norm of the optimality condition for the unconstrained problem (24)

$$\nabla \ell_p(v) = A(v - v^*) - J^T \gamma.$$

Notice that the components of $\nabla \ell_p$ have units of generalized momentum $p = Mv$. Depending on the choice of generalized coordinates, the generalized momentum components may have different units. In order to weigh all the components equally, we define the diagonal matrix $D = \text{diag}(M)^{-1/2}$ and perform the following change of variables:

$$\tilde{v}_p = Dv$$
$$\tilde{p} = Dp$$
$$\tilde{\gamma}_c = DJ_c$$

(25)

where we define the generalized contact impulse $\tilde{\gamma}_c = J^T \gamma$.

With this scaling, all the new tilde variables have the same units, square root of Joules. Using these definitions, we write our stopping criterion as

$$\|\tilde{v}_p\| < \varepsilon_a + \varepsilon_r \max(\|\tilde{p}\|, \|\tilde{\gamma}_c\|)$$

(26)

where $\varepsilon_r$ is a dimensionless relative tolerance that we usually set in the range from $10^{-6}$ to $10^{-1}$. The absolute tolerance $\varepsilon_a$ is used to detect rare cases where the solution leads to no contact and no motion, typically due to external forces. We always set this tolerance to a small number, $\varepsilon_a = 10^{-16}$.

V. CONTACT MODELING PARAMETERS

Thus far, $R_i$ and $\hat{v}_{c,i}$ have been treated as known problem data. This section makes an explicit connection of these quantities with physical parameters to model compliant contact with regularized friction. We seek to model compliant contact as in (2), parameterized by physical parameters: stiffness $k$ (in N/m) and dissipation time scale $\tau_d$ (in seconds). Therefore, users of this model only need to provide these physical parameters, and regularization is computed from them. Notice this approach is different from the one in [17], where regularization is not used to model physical compliance but rather to introduce a user tunable Baumgarte-style stabilization to avoid constraint drift.

A. Compliant Contact, Principle of Maximum Dissipation, and Artifacts

Dropping subscript $i$ for simplicity, we solve the projection problem in (23) analytically in Appendix C for a regularization matrix of the form $R = \text{diag}(\{R_t, R_t, R_n\})$

$$\gamma = P_R(y)$$

$$= \begin{cases} y_r & \text{station, } y_r \leq \mu y_n \\ \frac{\mu y_n \hat{t}}{1 + \mu^2} (y_n + \hat{\mu} y_r) & \text{sliding, } -\hat{\mu} y_r < y_n \leq \frac{y_r}{\mu} \\ 0 & \text{no contact, } y_n < -\hat{\mu} y_r \end{cases} (27)$$

where $y_r$ and $y_n$ are the tangential and normal components of $y$, respectively, $y_r = \|\hat{y}_r\|$ is the radial component, and $\hat{t} = y_t/|y_t|$ is the unit tangent vector. We also define the coefficients $\hat{\mu} = \mu (R_t/R_n)^{1/2}$ and $\mu = \mu R_t/R_n$ that result from the warping introduced by the metric $R$.

Our compliant model of contact is defined by

$$\hat{v}_c = \begin{bmatrix} 0 \\ 0 \\ \hat{\nu}_n \end{bmatrix}$$

$$\hat{\nu}_n = -\frac{\phi_0}{\delta t + \tau_d}$$

where $\phi_0$ is the previous step signed distance reported by the geometry engine. The normal direction regularization parameters are taken as $R_n^{-1} = \delta t R_t/\tau_d^2$. To gain physical insight into our model, we substitute $y = -R^{-1}(v_c - \hat{v}_c)$ into (27) to
obtain
\[ \gamma_t = \min \left( \frac{\|v_t\|}{R_t}, \mu \gamma_n \right) \]
with \( \gamma_t \) linear with the (very small) slip velocity during stiction and with the maximum value given by \( \mu \gamma_n \), effectively modeling Coulomb’s friction. Notice that to better model stiction, we are interested in small values of \( R_t \). We discuss our parameterization of \( R_t \) in Section V-B. Moreover, since \( \ell = y_t/\|y_t\| = -v_t/\|v_t\| \), friction impulses oppose sliding and, therefore, satisfy the principle of maximum dissipation.

2) Sliding usually happens with low velocities, and therefore, the term \( \delta t \mu \parallel v_t \) is negligible.

3) For robotics applications, we are mostly interested in inelastic contact. We will see that this can be effectively modeled with \( \tau_d \approx \delta t \) in Section V-B. Therefore, in this regime, the term \( \tau_d \mu \parallel v_t \) also goes to zero as \( \delta t \to 0 \).

4) We are definitely interested in the onset of sliding. This is captured by the approximation, which properly models the Coulomb friction law.

B. Conditioning of the Problem

Regularization parameters not only determine the physical model, but also affect the robustness and performance of the SAP solver. Modeling near-rigid objects and avoiding viscous drift during stiction require very small values of \( R_t \) and \( R_n \) that can lead to ill-conditioned problems. Under these conditions, the Hessian of the system exhibits a large condition number, and round-off errors can render the search direction of Newton iterations useless. We show in this section how a judicious choice of the regularization parameters leads to much better conditioned system of equations, without sacrificing accuracy. This is demonstrated in Section VI with a variety of tests cases.

1) Near-Rigid Contact: In our formulation, rigid objects must be modeled as near-rigid using large stiffnesses. However, as mentioned above, blindly choosing large values of stiffness can lead to ill-conditioned systems of equations. Here, we propose a principled way to choose the stiffness parameter when modeling near-rigid contact.

Consider the dynamics of a mass particle \( m \) laying on the ground, with contact stiffness \( k \) and dissipation time scale \( \tau_d \). When in contact, the dynamics of this particle is described by the equations of a harmonic oscillator with natural frequency \( \omega_n^2 = k/m \) or period \( T_n = 2\pi / \omega_n \), and damping ratio \( \zeta = \tau_d \omega_n / 2 \). We say the contact is near-rigid when \( T_n \lesssim \delta t \) and the time step \( \delta t \) cannot temporally resolve the contact dynamics. In this near-rigid regime, we use compliance as a means to add a Baumgarte-like stabilization to avoid constraint drift, as similarly done in [16]. Choosing the time scale of the contact to be \( T_n = \beta \delta t \) with \( \beta \leq 1 \), we model inelastic contact with a dissipation that leads to a critically damped oscillator, or \( \zeta = 1 \). This dissipation is \( \tau_d = 2\zeta / \omega_n \), or in terms of the time step, \( \tau_d = \beta \delta t \).

Using the harmonic oscillator equations, we can estimate the value of stiffness from the frequency \( \omega_n \) as \( k = 4\pi^2 m / (\beta^2 \delta t^2) \). Since \( \tau_d \approx \delta t \), \( R_n^{-1} = \delta t k (\delta t + \tau_d) \approx \delta t^2 k \), and we estimate the regularization parameter as \( R_n = \frac{\beta^2}{4\pi^2} w \), where we define \( w = 1/m \).

It is useful to estimate the amount of penetration for a point mass resting on the ground. In this case, we have
\[ \phi = \frac{mg}{k} = \frac{\beta^2}{4\pi^2} mw g \delta t^2 = \frac{\beta^2}{4\pi^2} g \delta t^2 \]
independent of mass. Taking $\beta = 1.0$ and Earth’s gravitational constant, a typical simulation time step of $\delta t = 10^{-3}$ s leads to $\phi \approx 2.5 \times 10^{-7}$ m, and a large simulation time step of $\delta t = 10^{-2}$ s leads to $\phi \approx 2.5 \times 10^{-5}$ m, well within acceptable bounds to consider a body rigid for typical robotics applications.

For a general multi-body system, we define the per-contact effective mass as $w_i = \| W_i \|_{\text{rms}} = \| W_i \| / 3$, where $W_{ii}$ is the $3 \times 3$ diagonal block of the Delassus operator $W = J M^{-1} J^T$ for the $i$th contact. Explicitly forming the Delassus operator is an expensive operation. Instead, we use an $O(n)$ approximation. Given contact $i$ involving trees $t_1$ and $t_2$, we form the approximation $W_{ii} \approx J_{it_1} M^{-1}_{t_1} J^T_{it_1} + J_{it_2} M^{-1}_{t_2} J^T_{it_2}$. Finally, we compute the regularization parameter in the normal direction as

$$R_n = \max \left( \frac{\beta^2}{4\pi^2} \| W_{ii} \|_{\text{rms}}, \frac{1}{\delta t (\delta t + \tau_d)} \right).$$

With this strategy, our model automatically switches between modeling compliant contact with stiffness $k$ when the time step $\delta t$ can resolve the temporal dynamics of the contact and using stabilization to model near-rigid contact with the amount of stabilization controlled by parameter $\beta$. In all of our simulations, we use $\beta = 1.0$.

2) Stiction: Given that our model regularizes friction, we are interested in estimating a bound on the slip velocity at stiction. We propose the following regularization for friction:

$$R_t = \sigma w$$

where $\sigma$ is a dimensionless parameter.

To understand the effect of $\sigma$ in the approximation of stiction, we consider once again a point of mass $m$ in contact with the ground under gravity, for which $w \approx 1/m$. We push the particle with a horizontal force of magnitude $F = \mu \gamma_n$ so that friction is right at the boundary of the friction cone and the slip velocity due to regularization, $v_s$, is maximized. Then, in stiction, we have

$$\| \gamma_t \| = \frac{v_s}{R_t} = \mu mg \delta t.$$

Using our proposed regularization in (30), we find the maximum slip velocity

$$v_s \approx \mu \sigma g \delta t$$

independent of the mass and linear with the time step size. Even though the friction coefficient $\mu$ can take any nonnegative value, most often in practical applications $\mu < 1$, values on the order of 1 are in fact considered as large friction values. Therefore, for this analysis, we consider $\mu \approx 1$. In all of our simulations, we use $\sigma = 10^{-3}$. With Earth’s gravitational constant, a typical simulation with time step of $\delta t = 10^{-3}$ s leads to a stiction velocity of $v_s \approx 10^{-3}$ m/s, and with a large step of $\delta t = 10^{-2}$ s, $v_s \approx 10^{-2}$ m/s. Smaller friction coefficients lead to even tighter bounds. These values are well within acceptable bounds even for the simulation of grasping tasks, which is significantly more demanding than simulation for other robotic applications (see Section VI).

3) Sliding Soft Contact: As we discussed in Section V-A, we require $R_t / R_n \ll 1$ so that we model compliance accurately during sliding. Now, in the near-rigid contact regime, the condition $R_t / R_n \ll 1$ is no longer required since, in this regime, regularization is used for stabilization. Therefore, we only need to verify this condition in the soft contact regime, when time step $\delta t$ can properly resolve the contact dynamics, i.e., according to our criterion, when $\delta t < T_n$. In this regime, $R_n \approx \delta t^2 k$, and using (30), we have

$$\frac{R_t}{R_n} \approx \sigma \delta t^2 \omega_n = 4\pi^2 \sigma \left( \frac{\delta t}{T_n} \right)^2 \lesssim 4\pi^2$$

where, in the last inequality, we used the assumption that we are in the soft regime where $\delta t < T_n$. Since $\sigma \ll 1$ and, in particular, we use $\sigma = 10^{-3}$ in all of our simulations, we see that $R_t / R_n \ll 1$. Moreover, $R_t / R_n$ goes to zero quadratically with $\delta t / T_n$ as the time step is reduced, and the dynamics of the compliance is better resolved in time.

Summarizing, we have shown that our choice of regularization parameters enjoys the following properties.

1) Users only provide physical parameters; contact stiffness $k$, dissipation time scale $\tau_d$, and friction coefficient $\mu$. There is no need for users to tweak solver parameters.

2) In the near-rigid limit, our regularization in (29) automatically switches the method to model rigid contact with constraint stabilization to avoid excessively large stiffness parameters and the consequent ill-conditioning of the system.

3) Frictional regularization is parameterized by a single dimensionless parameter $\sigma$. We estimate a bound for the slip velocity during stiction to be $v_s \approx \mu \sigma g \delta t$. For $\sigma = 10^{-3}$, the slip during stiction is well within acceptable bounds for robotics applications.

4) We show that $R_t / R_n \ll 1$ when $\delta t$ can resolve the dynamics of the compliant contact, as required to accurately model compliance during sliding.

VI. TEST CASES

We evaluate the robustness, accuracy, and performance of our method in a number of simulation tests. All the simulations are carried out in a system with 24 2.2-GHz Intel Xeon cores (E5-2650 v4) and 128 GB of RAM, running Linux. However, all of our tests are run in a single thread.

For all of our simulations, unless otherwise specified, our model uses $\beta = 1.0$ and $\sigma = 10^{-3}$ for the regularization parameters in (29) and (30), respectively.

A. Performance Comparisons Against Other Solvers

We evaluate commercial software Gurobi, considered an industry standard, to solve our primal formulation (19). As an open-source option, we evaluate the Geodesic interior-point method (IPM) from [49]. Geodesic IPMs, in contrast with primal-dual IPMs, do not apply Newton’s method to the central-path conditions directly. Instead, they use geodesic curves that satisfy the complementarity slackness condition. Since the Geodesic IPM and the SAP solver use the same supernodal linear algebra code described in Section IV-D, it is natural to compare their performance.

For performance comparisons, we use the steady clock from the STL std::chrono library to measure wall-clock time for the SAP solver and the Geodesic IPM. For Gurobi, we access the
B. Spring–Cylinder System

We model the setup shown in Fig. 4, consisting of a cylinder of radius $R = 0.05$ m and mass $m = 0.5$ kg connected to a wall to its left by a spring of stiffness $k_s = 100$ N/m. While the cylinder is free to rotate and translate in the plane, the ground constrains the cylinder’s motion in the vertical direction. The contact stiffness is $k = 10^4$ N/m and the dissipation time scale is $\tau_d = 0.02$ s. The cylinder is initially placed with zero velocity at $x_0 = 0.1$ m to the right of the spring’s resting position, and it is then set free.

For reference, we first simulate this setup with frictionless contact, i.e., with $\mu = 0$. Without friction, the cylinder does not rotate, and we effectively have a spring–mass system with natural frequency $\omega_n = \sqrt{k_s/m}$. We use a rather coarse time step of $\delta t = 0.02$ s, discretizing each period of oscillation with about 22 steps. Fig. 5 shows the total mechanical energy as a function of time computed using three different schemes: symplectic Euler, midpoint rule, and implicit Euler. The amount of numerical dissipation introduced by the implicit Euler scheme dissipates the initial energy in just a few periods of oscillation. For the symplectic Euler scheme, we observe in Fig. 5 that, while the energy is not conserved, it stays bounded, within a band 28% peak-to-peak wide. The figure also confirms that the second-order midpoint scheme conserves energy exactly. These are well-known theoretical properties of these integration schemes when applied to the spring–mass system.

We now focus our attention to a case with frictional contact using $\mu = 1$. As we release the cylinder from its initial position at $x_0 = 0.1$ m, friction with the ground establishes a rolling contact, and the system sets into periodic motion. Since now kinetic energy is split into translational and rotational components, the rolling cylinder behaves as a spring–mass system with an effective mass $m_{\text{eff}} = m + I_o/R^2$, with $I_o$ the rotational inertia of the cylinder about its center. Therefore, the frequency of oscillation is slower, and the same time step, $\delta t = 0.02$ s, now discretizes one period of oscillation with about 27 steps.

Total energy is shown in Fig. 6. Solutions computed with the implicit Euler and the symplectic Euler scheme show similar trends to those in the frictionless case. The midpoint rule does not conserve energy exactly, but it does significantly better, with a peak-to-peak variation of only 0.16%. While the ideal rolling contact does not dissipate energy, the regularized model of friction does dissipate energy given that the slip velocity is never exactly zero, though small in the order of $\sim \sigma \mu \delta t g$, as shown in Section V-A. The symplectic Euler scheme and the midpoint rule take 10 min of simulated time and about 1000 oscillations to dissipate 10% of the total energy (see Fig. 6, right). This level of numerical dissipation is remarkably low, considering that real mechanical systems often introduce several sources of dissipation.

To study the order of accuracy of our approach, we define the $L^2$-norm position error as

$$e_q = \left( \frac{1}{T} \int_0^T (x(t) - x_e(t))^2 dt \right)^{1/2}$$

where $x_e(t)$ is the known exact solution. We simulate for $T = 5$ s about ten periods of oscillation. Fig. 7 shows the position error as a function of the time step. We see that even with frictional contact, the two-stage approach with the midpoint rule achieves second-order accuracy. Both the implicit Euler
Fig. 8. Initial conditions (top) and an intermediate configuration after 2 s of simulated time (bottom) for the clutter setup with (left) and without (right) walls. Many of the spheres in the configuration with no walls roll outside the frame in the intermediate configuration.

Fig. 9. Iterations and wall-clock time per time step for the clutter case with 40 bodies and with walls.

and the symplectic Euler scheme are first order, though the error is significantly smaller when using the symplectic Euler scheme.

C. Clutter

Objects are dropped into an 80 cm × 80 cm × 80 cm container in four different columns with the same number of objects in each (see Fig. 8). Each column consists of an arbitrary assortment of spheres of radius 5 cm and boxes with sides of 10 cm in length. With a density of 1000 kg/m$^3$, spheres have a mass of 0.524 kg and boxes have a mass of 1.0 kg. We set a very high contact stiffness of $k = 10^{12}$ N/m so that the model is in the near-rigid regime. The dissipation time scale is set to equal the time step, and the friction coefficient of all the surfaces is $\mu = 1.0$.

We first run our simulations with ten bodies per column for a total of 40 bodies. We simulate ten seconds using time steps of size $\delta t = 10$ ms. The number of solver iterations and wall-clock time per time step are reported in Fig. 9. We observe that the SAP solver needs to perform a larger number of iterations during the very energetic initial transient. As the system reaches a steady state, however, the SAP solver warm-starts very effectively, performing only about three iterations per time step. Even though the SAP solver necessitates a larger number of iterations to converge than the Geodesic IPM during this initial transient, the wall-clock time per time step is very similar. Unlike the SAP solver and the Geodesic IPM that benefit from warm start, Gurobi performs about nine iterations per time step in both the initial transient and steady states.

Fig. 10 shows two examples of convergence history. We denote with $\ell_0$ the cost evaluated at the initial guess, the previous time step velocity. With $\ell^*$, we denote the optimal cost, which we approximate with its value from the last iteration. At step 60 during the initial transient for which the SAP solver requires 21 iterations to converge, we observe that the algorithm reaches quadratic convergence after an initial linear convergence transient, matching theoretical predictions (see Appendix E). At step 520, past the initial energetic transient, the SAP solver exhibits linear convergence and satisfies the convergence criterion within five iterations.

1) Scalability: We evaluate the scalability of the SAP solver by varying the number of objects in the clutter. We study the case with and without walls (see Fig. 8) as this variation leads to very different contact configurations and sparsity patterns. The size of the problems can be appreciated in Fig. 11 showing the number of contact constraints at the end of the simulation when objects are in the steady state. We observe a larger number of contacts for the configuration without walls since, in this configuration, many of the boxes spread over the ground and lay flat on one of their faces, leading to multicontact configurations (see Fig. 8).
We define the speedup against Gurobi as the ratio of the wall-clock time spent by a solver to the wall-clock time reported by Gurobi. Fig. 12 shows the speedup for both the SAP solver and the Geodesic IPM in the configuration with and without walls. The setup with walls is particularly difficult given that objects are constrained to pile up, leading to a configuration in which almost all the objects are coupled with every other object by frictional contact (see Fig. 8). For example, the motion of an object at the bottom of the pile can lead to the motion of another object far on top of the pile. In contrast, the simulation with no walls leads to islands of objects that do not interact with each other.

In general, we observe two regimes. For problems with less than about 40 bodies, the SAP solver outperforms Gurobi significantly by up to a factor of 25 in the case with walls and up to a factor of 50 with no walls. Beyond 80 bodies, Gurobi outperforms both the SAP solver and the Geodesic IPM in the case with walls, but the SAP solver is about ten times faster for the case with no walls. Though the SAP solver shows to be about twice as fast as the Geodesic IPM for most problem sizes, it can be five times faster for small problems with eight bodies or less.

It could be argued that these speedup results depend on the accuracy settings of each solver. For a fair comparison, we define the dimensionless momentum error as

$$e_m = \frac{\|\tilde{\nabla} \ell_p\|}{\max(\|\tilde{p}\|, \|\tilde{j}\|)}$$

(32)

using the scaled generalized momentum quantities in (25). We also define the dimensionless complementarity slackness error as

$$e_\mu = \frac{1}{n_c} \sum_{i} \| g_i \cdot \gamma_i \|$$

(33)

Fig. 13 shows average values of $e_m$ and $e_\mu$ over all the time steps. Since the SAP solver satisfies the complementarity slackness exactly, $e_\mu$ is not shown. We have verified this to be true within machine precision for all the simulated cases.

The SAP solver’s momentum error is below $10^{-5}$ as expected since this is the value used for the termination condition. Similarly, the complementarity slackness is below $10^{-5}$ for the Geodesic IPM, since this is the value used for its own termination condition. Gurobi does a good job at satisfying the complementarity slackness. However, it is the solver with the largest error in the momentum equations, even though both the SAP solver and the Geodesic IPM outperform Gurobi in most of the test cases. These metrics demonstrate that when the SAP solver and the Geodesic IPM outperform Gurobi, it is not at the expense of accuracy.

2) Slip Parameter: We study the effect of the slip parameter $\sigma$ in (30). We use $\delta t = 10 \text{ ms}$ and simulate with the SAP solver 40 objects for 10 s to a steady-state configuration. At this steady state, we compute the mean slip velocity among all the contacts, shown in Fig. 14 along with the estimated slip in (31), $v_s \approx \sigma \mu \delta t g$. We see that the mean slip velocity remains below the estimated slip as expected in a static configuration with objects in stiction. In the case with walls where stiction helps to hold the steady-state static configuration, we see that the mean slip velocity closely follows the slope of the slip estimate. Without the walls, objects do not pile up in a complex static structure but simply lie on the ground, and therefore, the resulting slip velocities are significantly smaller. The sudden drop in the slip velocity for $\sigma > 10^{-3}$ is caused by the sensitivity of the final state on the value of $\sigma$. As $\sigma$ increases, so does the slip velocity bound $v_s$ and objects in the configuration without walls can slowly drift into a configuration leading to more contacts. In particular, boxes are more likely to slowly drift until one of their faces lies flat on the ground; a configuration with zero slip once steady state is reached.

We conclude by examining the effect of $\sigma$ on the conditioning of the system. Fig. 15 shows the condition number of the Hessian.
The grasp force is commanded to vary as a function of time.

in the final configuration and the mean number of Newton iterations throughout the simulation. We see that the condition number scales as $\sigma^{-1}$, while the mean number of Newton iterations is roughly proportional to $\ln(\sigma)$. Our default choice $\sigma = 10^{-3}$ is a good compromise between accurate stiction, performance, and conditioning.

D. Slip Control

While previous works on convex approximations model rigid contact [14], [15] or use regularization as a means of constraint stabilization [17], our work is novel in that we incorporate physical compliance. This allows us not only to model compliant point contact, but also to incorporate sophisticated models of surfaces patches. We incorporate the pressure field model [35] implemented as part of Drake’s [24] hydroelastic contact model. We use the discrete approximation introduced in [36] to approximate each face of the contact surface as a compliant contact point at its centroid.

To demonstrate this capability, we reproduce the test in [36] that models a Soft-bubble gripper [50], a parallel jaw WSG 50 Schunk gripper outfitted with air filled compliant surfaces (see Fig. 16). The aforementioned gripper is simulated anchored to the world holding a spatula by the handle horizontally. We use $\delta t = 5 \times 10^{-3}$ s. The grasp force is commanded to vary between 1 and 16 N with square wave having a 6-s period and a 75% duty cycle (see Fig. 17, left). This results in a periodic transition from a secure grip to a loose grip allowing the spatula to pitch in a controlled manner within grasp (see Fig. 17 and the accompanying video). These contact mode transitions are resolved by our model. We observe that stiction during the secure grip is properly resolved with the tight bounds for the slip due to regularization discussed in Section V-B. While this case only has eight DOFs, it generates about 60 contact constraints during the slip phase and about 160 contact constraints during the stiction phase.

We compare the performance of the SAP solver against both Gurobi and Geodesic IPM (see Section VI-A). For the SAP solver, we use a relative tolerance $\varepsilon_r = 10^{-3}$ (see Section IV-E). For Gurobi, we set its tolerance parameter $\text{BarQCPConvTol}$ to $10^{-8}$. For the Geodesic IPM, we set its complementary slackness tolerance to $10^{-6}$; larger values lead to failure for this task. The SAP solver bounds the momentum error, exhibiting a maximum value of $9.99 \times 10^{-2} \%$. Even with such a tight tolerance, Gurobi exhibits 2.6% maximum error. Geodesic IPM’s errors are significantly smaller, below $2 \times 10^{-4} \%$. However, its robustness is very sensitive to the specified tolerance.

Even though the SAP solver’s solutions are significantly more accurate than those from Gurobi, it performs 92 times faster than Gurobi. The SAP solver is 20 times faster than the Geodesic IPM and significantly more robust to solver tolerances. In terms of solver iterations, the SAP solver only performs 0.62 iterations on average, showcasing how effectively it warm-starts. The Geodesic IPM performs 5.6 iterations per step on average and Gurobi performs 10.1 iterations on average.

E. Dual-Arm Manipulation

We demonstrate the effectiveness of our approach with the simulation of a complex manipulation task. In this scenario, two Kuka IIWA arms (seven DOFs each) are outfitted with anthropomorphic Allegro hands (16 DOFs each) (see Fig. 1). In front of the robot, a table has a jar (with a lid, 12 DOFs) full of 16 marbles of 50 g each (96 DOFs) and a bowl (six DOFs), completing the model with a total of 160 DOFs. Contact between the jar and the lid is modeled using Drake’s hydroelastic model [35], [36] (see Section VI-D), while point contact is used for all other interactions. The time step is set to $\delta t = 5 \times 10^{-3}$ s.

The arms’ controllers track a prescribed sequence of Cartesian end-effector keyframe poses, while the hands’ controllers track prescribed open/close configurations. We use force feedback to gauge successful grasps and to know when the jar makes contact with the table. The robot is commanded to open the jar, pour its contents into the bowl, close the lid, and put the empty jar back in place (keyframes in Fig. 1 and the accompanying video).
This particular task generates hundreds of contact constraints, as shown in Fig. 18, which also labels important events during the task. We remark that our framework predicts contact mode switching as a result of the computation. For instance, the lid initially covering the jar is held by stiction, and it transitions to sliding as the robot pulls it out.

To assess accuracy, we evaluate the dimensionless momentum and complementarity slackness errors defined in (32) and (33), respectively. We perform the simulation of the same task several times using different solver tolerances. The results of these runs are shown in Figs. 19 and 20. Even though each solver uses a different tolerance parameter, it is still useful to place these tolerances in the same horizontal axis. The maximum tolerance we use for each solver corresponds to the largest value that can be used to complete the task successfully. For Gurobi and Geodesic IPM, smaller values of the tolerance parameter make the simulation impractically slow. The SAP solver, on the other hand, cannot achieve errors below $10^{-6}$ for this case due to round-off errors. Figs. 19 and 20 show both the mean and median of the errors over the entire simulation to show that errors do not follow a symmetric distribution. More interesting, however, are the minimum and maximum errors, shown as shaded areas. The SAP solver guarantees that momentum errors are below the specified tolerance, given that this is precisely its stopping criterion in (26). We see, however, that it is difficult to correlate the expected error to solver tolerance when using Gurobi or Geodesic IPM. In practice, we consistently observe that the robot does not complete the task successfully when momentum errors are larger than about 10%, regardless of the solver. Therefore, we find that being able to specify a tolerance for the momentum error directly is immensely useful. Given that the SAP solver satisfies the complementarity slackness condition exactly, the complementarity slackness error for the SAP solver is not included in Fig. 20.

Fig. 21 shows the mean number of iterations per time step for each solver. We see that the number of iterations needed by the SAP solver is consistently below the other two solvers given how effectively the SAP solver warm-starts from the previous time-step solution.

To make a fair comparison among solvers, from Fig. 19, we choose tolerances for each solver that result in similar values of the mean momentum error. For Gurobi, we set its tolerance parameter $\texttt{BarQCPConvTol}$ to $10^{-8}$. For the Geodesic IPM, we set its complementary slackness tolerance to $10^{-6}$. For the SAP solver, we set its relative tolerance to $10^{-3}$. Notice that this is not entirely fair to the SAP solver, given that the SAP solver does guarantee the maximum momentum error to be below $10^{-3}$, while this is not true for the other two solvers. Still, the SAP solver is 7.4 faster than Gurobi and 2.2 faster than the Geodesic IPM. In terms of iterations, the SAP solver performs 4 iterations on average, while the Geodesic IPM performs 8.3 iterations on average. This shows that since both the solvers use exactly the same sparse algebra, the performance gains with the SAP solver are entirely due to its ability to warm-start effectively rather than to differences in the implementation. The general-purpose solver Gurobi, on the other hand, performs 10.1 iterations on average.

In summary, the simulation of this complex robotic task demonstrates how accuracy translates directly to robustness. We observe how the maximum momentum error defined in (32) is a
good proxy for robustness in simulation; simulations with errors larger than about 10% could not complete the task successfully. In this regard, the SAP solver provides a certificate of accuracy that proves useful in practice.

VII. VARIATIONS AND EXTENSIONS

The method presented in this article can be extended in several ways.

1) Expand the family of constraints: No doubt contact constraints are the most challenging. However, our method can be extended to include bilateral constraints, proportional–derivative controllers with force limits, and even joint dry friction [17].

2) Branch induced sparsity: In this article, we exploit sparsity only at the tree level. However, branch sparsity can lead to additional performance. Consider, for instance, a standing humanoid robot with a floating hip. Since arms and the upper torso are not in contact with the ground, they can be eliminated from the computation. Additional performance gains could be attained using specialized algebra for multibody dynamics [51].

3) Parallelization: This article focuses on accuracy, robustness, and convergence properties of the algorithm executed in a single thread. The sparse algebra can be parallelized, and in particular, disjoint islands of bodies can be solved separately in different threads.

4) Deformable finite-element method (FEM) models: Using the SAP solver for the modeling of deformable objects with contact and friction is the topic of current research efforts by the authors. FEM models lead to state-dependent stiffness (16) and damping (17) matrices with a complex structure that requires specialized handling of sparsity. Moreover, modeling assumptions must be carefully analyzed in order to ensure the positive definiteness of these matrices used in our convex approximation of contact.

5) Differentiation: Since forces are a continuous function of state, the model is well suited for applications requiring gradients, such as trajectory optimization, machine learning, parameter estimation, and control. Factorizations computed during forward dynamics can be reused when computing gradients for a performant implementation.

VIII. LIMITATIONS

All the models are approximations of reality, while numerical methods can only approximate our models. We list here the limitations we identify for our method.

1) Convex approximation: The convex approximation amounts to a gliding effect during sliding at a distance \( \phi \sim \delta \mu \| v_t \| \). Regularization leads to a model of regularized friction [see (28)]. Details are provided in Section V.

2) Stiffness and dissipation: Our method requires stiffness \( K \) and damping \( D \) matrices to be SPD or SPD approximations (see Section II-D). For joint-level spring–dampers, the exact \( K \) and \( D \) can be used, but for other forces, such as those arising from a spatial arrangement of springs, an SPD approximation must be made as \( K \) might not be SPD in certain configurations.

3) Linear approximations: Algorithm 1 evaluates the SPD gradient \( A \) once at \( v^* \) at each time step. In other words, our method replaces the original balance of momentum (7) with its linear approximation (18). This is exact for many important cases and accurate to second order in the general case. See Section III-C for details.

4) Delassus operator estimation: In Section V-B, we use a diagonal approximation of the Delassus operator to estimate stiffness in the near-rigid regime. Corner cases exist. Consider a pile of books. While the inertia of a contact at the bottom of the pile is estimated solely on the mass of one book, this contact is supporting the weight of the entire stack. Stiffness is underestimated, and user intervention is needed to set proper parameters.

5) Scalability: We see no reason why the SAP solver with the direct supernodal algebra (see Section IV-D) could not scale to thousands of bodies if there is structured sparsity. However, scalability needs to be studied further, along with the usage of iterative solvers such as conjugate gradient, widely used in optimization.

IX. CONCLUSION

In this article, we presented a novel unconstrained convex formulation of compliant contact. In this formulation, constraints are eliminated using analytic formulas that we developed. Our scheme incorporates the midpoint rule into a two-stage scheme, with demonstrated second-order accuracy. We rigorously characterized our numerical approximations and the artifacts introduced by the convex approximation of contact. We reported limitations of our method and discussed extensions and areas of further research.

We showed that regularization maps to physical compliance, allowing us to eliminate algorithmic parameters and to incorporate complex models of continuous contact patches. Moreover, we studied the tradeoff between regularization and numerical conditioning for the simulation of near-rigid bodies and the accurate resolution of stiction.

We presented the SAP solver, a robust and performant solver that warm-starts very effectively in practice. The SAP solver globally converges at least at a linear rate and exhibits quadratic convergence when additional smoothness conditions are satisfied. The SAP solver can be up to 50 times faster than Gurobi in small problems with up to a dozen objects and up to ten times faster in medium-sized problems with about 100 objects. Even though the SAP solver uses the supernodal algebra implemented for the Geodesic IPM, it performs at least two times faster due to its effective warm-starts from the previous time-step solution. Moreover, the SAP solver is significantly more robust in practice given that it guarantees a hard bound on the error in momentum, effectively providing a certificate of accuracy.

We have incorporated the SAP solver into the open-source robotics toolkit Drake [24] and hope that the simulation and robotics communities can benefit from our contribution.
APPENDIX A

PROOF OF PROPOSITION 1

The Taylor expansion of $m(v)$ at $v = v^*$ reads

$$m(v) = m^* + \frac{\partial m}{\partial v}(v - v^*) + O_m(||v - v^*||^2)$$

$$= \frac{\partial m}{\partial v}(v - v^*) + O_m(||v - v^*||^2)$$

(34)

where we use the fact that by definition $m^* = m(v^*) = 0$. All the derivatives are evaluated at $v = v^*$ unless otherwise noted. We first evaluate the Jacobian of the mass matrix term in (14) as

$$\frac{\partial (M(q^0(v))(v - v_0))}{\partial v} = M(q^0(v^*)) + E$$

where we defined

$$E = \frac{\partial M(q^0)}{\partial v}(v^* - v_0).$$

Note that by combining (6) and (12), the mid-step configuration $q^0$ can be written as

$$q^0(v) = q_0 + \delta t \theta q^0_{\theta v^c}$$

$$= q_0 + \delta t \theta N(q^0_{\theta v^c})v^0(v).$$

Hence, by the chain rule, $E$ can be further calculated as

$$E = \delta t \theta \frac{\partial M(q^0)}{\partial q} \frac{\partial q^0_{\theta v^c}}{\partial v}(v^* - v_0).$$

Notice that

$$||E|| \leq \delta t \left( \frac{\partial M(q^0)}{\partial q} \right) ||q^0_{\theta v^c}|| ||v^* - v_0||$$

$$= O(\delta t^2)$$

since $||v^* - v_0|| = O(\delta t)$.

We proceed similarly to expand the Jacobian of $F_1(v) = F_1(q^0(v), v^0(v))$ as

$$\frac{\partial F_1}{\partial v} = -\delta t \theta q^0_{\theta v^c} K(q^0, v^0) - \theta D(q^0, v^0)$$

with $K$ and $D$ the stiffness and damping matrices defined by (16) and (17).

We can now write the Jacobian of $m(v)$ in (34) as

$$\frac{\partial m}{\partial v} = A + E - \delta t \frac{\partial F_2}{\partial v}$$

where we defined

$$A = M + \delta t^2 \theta q^0_{\theta v^c} K + \delta t \theta D.$$  

With these definitions, the Taylor expansion in (34) becomes

$$\frac{\partial m}{\partial v}(v - v^*) = A(v - v^*) + E(v - v^*)$$

$$- \delta t \frac{\partial F_2}{\partial v}(v - v^*) + O_m(||v - v^*||^2).$$

Since contact is compliant, forces are finite within the finite interval $\delta t$, and therefore, $||v - v^*|| = O(\delta t)$. Thus

$$E(v - v^*) = O_E(\delta t^3)$$

$$\delta t \frac{\partial F_2}{\partial v}(v - v^*) = \mathcal{O}_E(\delta t^3)$$

$$O_m(||v - v^*||^2) = O_m(\delta t^2).$$

Therefore, the positive-definite linearization

$$A(v - v^*) + \mathcal{O}_E(\delta t^3) + \mathcal{O}_E(\delta t^2) + \mathcal{O}_m(\delta t^2) = J^T \gamma$$

agrees with the original momentum balance in (7) to second order.

Finally, notice that $A$ is a linear combination of positive-definite matrices with nonnegative scalars in the linear combination, and therefore, $A > 0$.

APPENDIX B

PROOF OF THEOREM 2

Before proving this theorem, we need the following result.

Lemma 1: The conic constraint $g(v, \sigma) \in \mathcal{F}^*$ is satisfied if $\sigma$ is given by $P_f(y(v))$.

Proof: Since $\sigma$ is the projection of $y(v)$ to the cone $\mathcal{F}$ with the $R$ norm, by Moreau’s decomposition theorem, we know that $y(v) - \sigma$ is in the polar cone of $\mathcal{F}$ with the $R$ norm. That is, $(y(v) - \sigma, x)_R \leq 0$ for all $x \in \mathcal{F}$, with the inner product $\langle v, w \rangle_R = v^T R w$. Reorganizing terms, we get

$$x^T R (y(v) - \sigma) \leq 0$$

$$x^T (-R \sigma - v_c + \tilde{v}_c) \leq 0$$

$$\langle x, -R \sigma - v_c + \tilde{v}_c \rangle \leq 0$$

for all $x \in \mathcal{F}$. Therefore, it follows that $-g = -(v_c - v_c + R \sigma)$ is in the polar cone of $\mathcal{F}$, and thus, $g$ is in the dual cone of $\mathcal{F}$.

The optimality condition for the unconstrained formulation in (24) is $\nabla \ell_p(v) = 0$. It is shown in Appendix D that

$$\nabla \ell_p(v) = A(v - v^*) - J^T \gamma(v)$$

with impulses given by $\gamma(v) = P_f(y(v))$, the dual optimal. Therefore, $\nabla \ell_p(v) = 0$ implies (21a), the first optimality condition for (19).

The analytical inverse dynamics solution shows that $\gamma = P_f(y(v))$ with the primal optimal $v$. Hence, choosing $\sigma = P_f(y(v))$ with the primal optimal $v$ satisfies (21b), the second optimality condition for (19).

Finally, by Lemma 1, the cone constraint $g(v, \sigma) \in \mathcal{F}^*$ is satisfied.

APPENDIX C

ANALYTICAL INVERSE DYNAMICS

We perform the projection in (23) for a regularization of the form $R = \text{diag}([R_t, R_t, R_n])$. For simplicity, we drop contact subindex $i$. We make the change of variables $\gamma = R^{1/2} \gamma$ and $\tilde{y} = R^{1/2} y$ [17] and observe that $\gamma$ is the Euclidean projection of $\tilde{y}$ onto cone $\tilde{\mathcal{F}}$ with coefficient $\tilde{\mu} = \mu (R_t/R_n)^{1/2}$. We conclude that

$$P_f(y) = R^{-1/2} P_f(\tilde{y}).$$

We partition $\mathbb{R}^3$ into three regions (see Fig. 22): closed cone $\mathcal{F}$, denoted with $R_f$, the interior of the polar $\mathcal{F}^*$, denoted with
where the $2 \times 2$ projection matrix is
\[ P^\perp(\hat{t}) = I_2 - P(\hat{t}), \] with $P(\hat{t}) = \hat{t} \otimes \hat{t}$.

Taking the gradient of (35) results in
\[
\nabla y^\ell_R(y) = \begin{cases}
\frac{1}{1+\mu^2} s_y(y) [ \mu R \hat{t} ], & \text{stiction, } y_r \leq \mu y_n \\
-\hat{t}^T, & \text{sliding, } -\hat{t}^T < y_n \leq \frac{y_n}{\mu} \\
0, & \text{no contact, } y_n < -\hat{t}^T y_n
\end{cases}
\] (36)

with $s_y(y) = \mu y_r + y_n$ positive in the sliding region. We note that $\nabla y^\ell_R(y)$ is not differentiable at the boundaries of $\mathcal{F}$ and $\mathcal{F}^c$. At points of differentiability, the Hessian $\nabla_y^2 \ell_R(y)$ is computed by taking derivatives of (36)
\[
\nabla y^2 \ell_R(y) =
\begin{cases}
\frac{R}{1+\mu^2} \left[ \mu \hat{t} + \frac{s_y(y) P^\perp(\hat{t})/y_n}{\mu} \right] \hat{t}, & \text{stiction} \\
\frac{\hat{t}}{1+\mu^2}, & \text{sliding} \\
0, & \text{no contact}
\end{cases}
\] (37)

Clearly, in the stiction region, we have $\nabla y^2 \ell_R(y) > 0$. Since, in the stiction region, we have $s_y(y) > 0$, the linear combination of $P(\hat{t})$ and $P(\hat{t})^\perp$ in (37) is SPD (since both the projection matrices are SPD). Therefore, $\nabla_y^2 \ell_R(y) \succeq 0$.

B. Gradients With Respect to Velocities

Recall we use bold italics for vectors in $\mathbb{R}^3$ and nonitalics bold for their stacked counterpart. With $y = -R^{-1} (Jv - \hat{v}_c)$, we use the chain rule to compute the gradient in terms of velocities
\[
\nabla v \ell_R = -J^T R^\perp \nabla v \ell_R
\] (38)

which, using (36), can be shown to equal
\[
\nabla v \ell_R = -J^T \gamma.
\] (39)

At points of differentiability, we obtain the Hessian of the regularizer $\ell_R(v)$ from the gradient of $\gamma(v)$ in (39)
\[
\nabla v \ell_R(v) = -J^T \nabla v \gamma \mathbf{J}
\]
\[
\nabla v \gamma = -\nabla v \gamma R^{-1}
\]
where $\nabla v \gamma$ is a block diagonal matrix where each diagonal block is the $3 \times 3$ matrix $\nabla v_i \gamma_i$ for the $i$-th contact. Alternatively, taking the gradient of (38) leads to the equivalent result
\[
\nabla^2 v \ell_R = J^T R^{-1} \nabla^2 v \ell_R R^{-1} \mathbf{J}
\]
where we can verify indeed that $-\nabla v \gamma = R^{-1} \nabla^2 v \ell R R^{-1}$. Since $\nabla^2 v \ell_R \succeq 0$, it follows that $-\nabla v \gamma \succeq 0$.

We define $G_i \in \mathbb{R}^{3 \times 3}$ the matrix that evaluates to $-\nabla v_i \gamma_i$ within regions $\mathcal{R}_i$, $\mathcal{R}^c_i$, and $\mathcal{R}_{II}$, where the projection is differentiable. At the boundary of $\mathcal{F}$, we use the analytical expression from $\mathcal{R}_i$. At the boundary of $\mathcal{F}^c$, we use the analytical expression from $\mathcal{R}^c_i$. This extension fully specifies $G_i$ for all $y_i \in \mathbb{R}^3$. Finally, we define the $3n_c \times 3n_c$ matrix $G = \text{diag}(G_i) \succeq 0$.

APPENDIX D

COMPUTATION OF THE GRADIENTS

We will start by taking derivatives of the regularizer term $\ell_R$. First we notice that we can write this term as
\[
\ell_R = \frac{1}{2} \| \gamma \|^2_R = \sum_i \ell_{R_i}
\]
with $\ell_{R_i} = 1/2 \| \gamma_i \|^2_{R_i}$. Since $\nabla y_i \ell_{R_i} = 0$ for $i \neq j$, we only need to compute the gradients of $\ell_{R_i}(y_i)$ with respect to the contact point variable $y_i \in \mathbb{R}^3$. Dropping contact subindex $i$ for simplicity, we write the regularization as
\[
\ell_R = \frac{1}{2} \| \gamma \|^2_R = \frac{1}{2} (R^T \| \gamma \|^2 + R_n \gamma_n^2).
\]

We use (27) to write the cost in terms of $y$ as
\[
\ell_R(y) = \begin{cases}
\frac{1}{2} \| R y \|^2 + R_n y_n^2, & \text{stiction, } y_r \leq \mu y_n \\
\frac{1}{2+\mu^2} (y_n + \mu y_n)^2, & \text{sliding, } -\hat{t}^T < y_n \leq \frac{y_n}{\mu} \\
0, & \text{no contact, } y_n < -\hat{t}^T y_n
\end{cases}
\] (35)

A. Gradients per Contact Point

We use the following identities to simplify expressions:
\[
\frac{\partial y_r}{\partial y_i} = \hat{t}, \quad \frac{\partial \hat{t}}{\partial y_i} = P^\perp(\hat{t}) y_r
\]

where the $2 \times 2$ projection matrix is
\[ P^\perp(\hat{t}) = I_2 - P(\hat{t}), \] with $P(\hat{t}) = \hat{t} \otimes \hat{t}$.
C. Gradients of the Primal Cost

With these results, we can now write the gradient $\nabla \ell_p$ and weighting matrix $H$ in Algorithm 2. For the gradient, we have

$$\nabla \ell_p(v) = A(v - v^*) + \nabla \ell_R$$

which, using (39), can be written as

$$\nabla v \ell_p(v) = A(v - v^*) - J^T \gamma$$

and since the unconstrained minimization seeks to satisfy the optimality condition $\nabla v \ell_p = 0$, we recover the balance of momentum.

Finally, we define the weighting matrix $H$ as

$$H = A + J^T G J$$

which, given the definition of $G$, returns the Hessian of $\ell_p(v)$ when the gradient is differentiable and extends the analytical expressions at points of nondifferentiability. Since $A > 0$ and $G \geq 0$, we have $H > 0$.

**APPENDIX E**

**CONVERGENCE ANALYSIS OF THE SAP SOLVER**

The convergence of the SAP solver is established by first showing that the objective function $\ell_p(v) = \frac{1}{2} \|v - v^*\|^2_A + P_\gamma(y(v))\|v\|^2_B$ is strongly convex and differentiable with Lipschitz continuous gradients. The former property is inherited from the positive-definite quadratic term provided by the positive-definite matrix $A$ in (24). The latter is shown using the differentiability of the squared-distance function and the Lipschitz continuity of its gradient (see [52, Ths. 5.3-i and 6.1-i]) combined with the identity

$$d^2_{\mathcal{K}}(x) = \frac{\|P_{\mathcal{K}}(x)\|^2}{2}$$

for any closed, convex cone $\mathcal{K}$. Here, the distance and projection functions are with respect to the norm $\| \cdot \|_R$, while $\mathcal{K}^\circ$ denotes the polar cone with respect to the corresponding inner product $x^T R y$.

**Lemma 2**: The following statements hold.

1. The function $\ell_p(v)$ is strongly convex, i.e., there exists $\mu > 0$ such that

$$\ell_p(u) \geq \ell_p(v) + \nabla \ell_p(v)(u - v) + \frac{\mu}{2} \|u - v\|^2.$$ 

2. The function $\ell_p(v)$ is differentiable and has Lipschitz continuous gradients, i.e., $\nabla \ell_p(v)$ exists for all $v$, and there exists $L > 0$ satisfying

$$\|\nabla \ell_p(v) - \nabla \ell_p(u)\| \leq L \|u - v\|.$$

**Proof**: The objective $\ell_p(v)$ is a function $f : \mathbb{R}^n \to \mathbb{R}$ of the following form:

$$f(v) = \frac{1}{2} d^2_{\mathcal{K}}(Zv + b) + v^T Wv + q^T v$$

where $Z \in \mathbb{R}^{m \times n}$ and $W \in \mathbb{R}^{n \times n}$ are symmetric and positive definite, and $d_{\mathcal{K}} : \mathbb{R}^m \to \mathbb{R}$ denotes the distance function of a closed convex set $\mathcal{K} \subseteq \mathbb{R}^m$ as measured by some quadratic form $\|x\|_Q$, i.e.,

$$d_{\mathcal{K}}(x) = \inf \{\|x - z\|_Q : z \in \mathcal{K}\}.$$

The sum of a strongly convex function with a convex function is strongly convex. Since the squared distance function is convex, and the quadratic term $v^T Wv$ is strongly convex (given that $W > 0$), the first statement holds.

The second statement follows trivially if we can show that it holds for the squared distance function. Differentiability follows from [52, Ths. 5.3-i and 6.1-i, Ch. 4], which shows that

$$\nabla d^2_{\mathcal{K}}(v) = 2Q(v - P_{\mathcal{K}}(v)).$$

That the gradient of $d^2_{\mathcal{K}}(v)$ is Lipschitz follows from the Lipschitz continuity of projection maps onto closed convex sets. ■

We remark that strong convexity implies the reverse Lipschitz inequality $\|\nabla f(v) - \nabla f(u)\| \geq \mu \|v - u\|$, which, in turn, means that the parameter $\mu$ and the Lipschitz constant $L$ satisfy $\mu \leq L$.

Recall that the SAP solver (see Algorithm 2) is a special case of the following iterative method for minimizing a function $f : \mathbb{R}^n \to \mathbb{R}$ given some initial point $v_0 \in \mathbb{R}^n$,

$$d_m = -H^{-1}(v_m)\nabla f(v_m)$$

$$t_m = \arg \min_t f(v_m + td_m)$$

$$v_{m+1} = v_m + t_m d_m$$

(40)

where $H : \mathbb{R}^n \to \mathbb{R}^{n \times n}$ is a function into the set of SPD matrices, i.e., $H(v) = H(v)^T$ and $H(v) > 0$ for all $v \in \mathbb{R}^n$. It is well known that gradient descent exhibits linear convergence to the global minimum when applied to a strongly convex function with Lipschitz continuous gradient. Incorporating a condition number bound $\sigma$ for $H(v)$ into standard gradient descent analysis will prove that the iterations (40) also have linear convergence. To show this, we let $\text{cond}(H(v))$ denote the condition number of $H(v)$ and $S(v_0)$ denote the sublevel set $\{v \in \mathbb{R}^n : f(v) \leq f(v_0)\}$.

**Lemma 3**: Let $f : \mathbb{R}^n \to \mathbb{R}$ be strongly convex and differentiable with Lipschitz continuous gradients. Fix $v_0 \in \mathbb{R}^n$. If there exists $\sigma > 0$ such that $\text{cond}(H(v)) \leq \sigma$ for all $v \in S(v_0)$, then the iterations (40) converge to the global minimum $v^*$ of $f(v)$ when initialized at $v_0$. Moreover

$$f(v_{m}) - f(v_i) \leq \left(1 - \frac{\mu}{\sigma L^2}\right)^m (f(v_0) - f(v_i))$$

for all iterations $m$, where $\mu$ is the strong-convexity parameter of $f(v)$ and $L$ is the Lipschitz constant of $\nabla f(v)$.

**Proof**: Dropping the subscript $m$ from $(v_m, t_m, d_m)$, we first observe that

$$f(v + td) \leq f(v) + \langle \nabla f(v), d \rangle t + \frac{L}{2} \|d\|^2 t^2$$

by Lipschitz continuity. Substituting $d = -H^{-1}\nabla f(v)$ gives

$$f(v + td) = f(v) - \nabla f(v)^T H^{-1}\nabla f(v) t + \frac{L}{2} \|H^{-1}\nabla f(v)\|^2 t^2.$$ 

Letting $\lambda_{\max}$ and $\lambda_{\min}$ denote the maximum and minimum eigenvalues of $H$ evaluated at $v$, respectively, it also follows that

$$f(v + td) \leq f(v) - \frac{1}{\lambda_{\max}} \|\nabla f(v)\|^2 t + \frac{L}{2} \frac{1}{\lambda_{\min}} \|\nabla f(v)\|^2 t^2.$$
Letting $\tilde{t}$ denote the minimizer of the right-hand side, we conclude that
\[
f(v + t\mathbf{d}) \leq f(v + \tilde{t}\mathbf{d}) \leq f(v) - \frac{1}{2} \left( \frac{\lambda_{\min}^2}{\lambda_{\max}^2} \|\nabla f(v)\|^2 \right)
\]
where the first inequality follows from the exact line search used to select $t$. Since $\sigma^2 \geq \lambda_{\min}/\lambda_{\max}^2$, we conclude that
\[
f(v + t\mathbf{d}) \leq f(v) - \frac{1}{2\sigma^2\lambda_{\max}} \|\nabla f(v)\|^2.
\]
On the other hand, letting $f_\ast = f(v_\ast)$, we have from strong convexity that the Polyak–Lojasiewicz inequality holds
\[
\|\nabla f(v)\|^2 \geq 2\mu(f(v) - f_\ast).
\]
Hence
\[
f(v + t\mathbf{d}) \leq f(v) - \frac{\mu}{\sigma^2\lambda_{\max}} (f(v) - f_\ast).
\]
Subtracting $f_\ast$ from both sides and factoring shows
\[
f(v + t\mathbf{d}) - f_\ast \leq \left( 1 - \frac{\mu}{\sigma^2\lambda_{\max}} \right) (f(v) - f_\ast).
\]
It follows that each iteration $m$ satisfies
\[
f(v_m) - f_\ast \leq \left( 1 - \frac{\mu}{\sigma^2\lambda_{\max}} \right)^m (f(v_0) - f_\ast).
\]
Since $\sigma \geq 1$ and $L \geq \mu$, the iterations converge, and the proof is completed.

Combining these lemmas shows that the SAP solver globally converges at (at least) a linear rate. By observing that the SAP solver reduces to Newton's method when the gradient is differentiable, we can also prove local quadratic convergence assuming differentiability on a neighborhood of the optimum $v_\ast$.

**Theorem 3:** The following statements hold.

1. The SAP solver globally converges from all the initial conditions.
2. If $\nabla f(v)$ is differentiable on the ball $B(v_\ast, r) := \{ v : \|v - v_\ast\| \leq r \}$ for some $r > 0$, then the SAP solver exhibits quadratic convergence, i.e., for some finite $M$ and $\zeta > 0$
\[
\|v_{m+1} - v_\ast\| \leq \zeta \|v_m - v_\ast\|^2
\]
for all $m > M$.

**Proof:** The first statement follows from Lemmas 2 and 3. To prove the second, we show that $B(v_\ast, r)$ contains a sublevel set $\Omega_\beta = \{ v : f(v) \leq \beta \}$ for some $\beta > 0$, implying that the SAP solver reduces to Newton's method with exact line search for some $m > M$, given that sublevel sets are invariant.

To begin, we have, by strong convexity, that
\[
\beta \geq f(v) \geq f(v_\ast) + \frac{\mu}{2} \|v - v_\ast\|^2
\]
for all $v \in \Omega_\beta$. Rearranging shows that
\[
\|v - v_\ast\|^2 \leq \frac{2\beta - f(v_\ast)}{\mu}.
\]
Hence, $B(v_\ast, r)$ contains $\Omega_\beta$ for any $\beta$ satisfying $\frac{2\beta - f(v_\ast)}{\mu} < r$.

Next, we prove that Newton iterations are quadratically convergent with exact line search. Indeed, using once more the strong convexity result in (41), we obtain
\[
\|v_{m+1} - v_\ast\|^2 \leq \frac{2}{\mu} \frac{f(v_{m+1}) - f(v_\ast)}{f(v_m)}
\]
\[
= \frac{2}{\mu} \frac{f(v_m + t_m d_m) - f(v_\ast)}{f(v_m)}
\]
\[
\leq \frac{2}{\mu} \frac{f(v_m + d_m) - f(v_\ast)}{f(v_m)}
\]
\[
\leq \frac{2}{L} \|v_m + d_m - v_\ast\|^2
\]
where the first line uses strong convexity, the third exact line search, and the last Lipschitz continuity. But, for some $\kappa > 0$, we have that $\|v_m + d_m - v_\ast\|^2 \leq \kappa \|v_m - v_\ast\|^4$ by the quadratic convergence of Newton’s method with unit step size (see [45, Th. 3.5]). Hence
\[
\|v_{m+1} - v_\ast\|^2 \leq \frac{2}{\mu} L \kappa \|v_m - v_\ast\|^4
\]
and the claim is proven.

**ACKNOWLEDGMENT**

The authors would like to thank Michael Sherman for his trust on this research from day one and the Dynamics and Simulation and Dexterous Manipulation teams at the Toyota Research Institute for their continuous patience and support.

**REFERENCES**

[1] D. Baraff, “Issues in computing contact forces for non-penetrating rigid bodies,” *Algorithmica*, vol. 10, no. 2, pp. 292–352, 1993.
[2] S. Hogan and K. U. Kristiansen, “On the regularization of impact without collision: The Painlevé paradox and compliance,” *Proc. Roy. Soc. A: Math., Phys. Eng. Sci.*, vol. 473, no. 2202, 2017, Art. no. 20160773.
[3] J.-S. Pang and D. E. Stewart, “Differential variational inequalities,” *Math. Program.*, vol. 113, no. 2, pp. 345–424, 2008.
[4] E. J. Haug, S. C. Wu, and S. M. Yang, “Dynamics of mechanical systems with Coulomb friction, stick, impact and constraint addition-deletion–I theory,” *Mech. Mach. Theory*, vol. 21, no. 5, pp. 401–406, 1986.
[5] D. E. Stewart and J. C. Trinkle, “An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction,” *Int. J. Numer. Methods Eng.*, vol. 39, no. 15, pp. 2673–2691, 1996.
[6] M. Anitescu and F. A. Potra, “Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems,” *Nonlinear Dyn.*, vol. 14, no. 3, pp. 231–247, 1997.
[7] J. Li et al., “An implicit frictional contact solver for adaptive cloth simulation,” *ACM Trans. Graph.*, vol. 37, no. 4, 2018, Art. no. 52.
[8] D. E. Stewart, “Convergence of a time-stepping scheme for rigid-body dynamics and resolution of Painlevé’s problem,” *Arch. Rational Mech. Anal.*, vol. 145, no. 3, pp. 215–260, 1998.
[9] D. M. Kaufman, S. Sueda, D. L. James, and D. K. Pai, “Staggered projections for frictional contact in multibody systems,” *ACM Trans. Graph.*, vol. 27, no. 5, Dec. 2008, Art. no. 164.
[10] D. Baraff, “Fast contact force computation for nonpenetrating rigid bodies,” in *Proc. 21st Annu. Conf. Comput. Graph. Interact. Techn.*, 1994, pp. 23–34.
[11] C. Duriez, F. Dubois, A. Kheddar, and C. Andriot, “Realistic haptic rendering of interacting deformable objects in virtual environments,” *IEEE Trans. Visual. Comput. Graph.*, vol. 12, no. 1, pp. 36–47, Jan./Feb. 2006.
[12] E. Coumans and Y. Bai, “Pybullet, a python module for physics simulation for games, robotics and machine learning,” 2016–2020. [Online]. Available: http://pybullet.org
[13] K. Erleben, “Velocity-based shock propagation for multibody dynamics animation,” *ACM Trans. Graph.*, vol. 26, no. 2, pp. 12–es, 2007.
