Robotics: Science and Systems 2026
Sydney, Australia, July 13–17, 2026

IMPACT: An Implicit Active-Set Augmented Lagrangian for Fast Contact-Implicit Trajectory Optimization

Jiayun Li[1,2] Dejian Gong[1] Georgia Chalvatzaki[1,2,3]
[1]PEARL Lab, Dept. of Computer Science, TU Darmstadt, Germany
[2]Hessian.AI, Darmstadt, Germany, [3]Robotics Institute Germany (RIG)

***Abstract*—*Contact-implicit trajectory optimization* (CITO) has attracted growing attention as a unified framework for planning and control in contact-rich robotic tasks. Recent approaches have demonstrated promising results in manipulation and locomotion without requiring a prescribed contact-mode schedule. It is well known that the underlying *mathematical programs with complementarity constraints* (MPCCs) remain numerically ill-conditioned, and systematic, scalable solution strategies for CITO remain an active area of research. More efficient and principled solvers that can handle contact constraints are therefore essential to broaden the applicability of CITO. In this work, we develop an augmented-Lagrangian approach to CITO for solving MPCC-based CITO with stationarity guarantees. The method can be interpreted as *identifying the implicit contact-mode branches on the fly* during the *trajectory optimization* (TO) iterations; we call this approach IMPACT (*IMPlicit contact ACtive-set Trajectory optimization*). We provide an efficient C++ implementation tailored to trajectory-optimization workloads and evaluate it on the open-source CITO and *contact-implicit model predictive control* (CI-MPC) benchmarks. On CITO, IMPACT achieves $2.9\times$–$70\times$ speedups over strong baselines (geometric mean $13.8\times$). On CI-MPC, we show improved control quality for contact-rich trajectories on dexterous manipulation tasks in simulation. Finally, we demonstrate the proposed method on real robotic hardware on a T-shaped object pushing task.**

I. Introduction

Optimal motion planning in contact-rich manipulation settings remains one of the major challenges in robotics. The planner must reason not only about continuous dynamics, but also about discrete events in which contacts are established, maintained, and broken to perform tasks such as underactuated manipulation and legged locomotion [1, 2]. This hybrid, nonsmooth structure makes it difficult to cast the problem as a well-defined continuous TO problem. A common workaround is to prescribe the contact mode sequence (contact schedule) a priori; conditioned on this discrete choice, the resulting TO problem can often be formulated as a smooth *nonlinear programming* (NLP) and solved efficiently using existing continuous optimization methods [3, 4]. By contrast, when the contact schedule is not fixed, the problem is typically posed as CITO. While CITO is promising for autonomously inferring contact modes in complex tasks, it is generally more challenging to solve due to nonsmooth contact dynamics and the combinatorics of mode changes [5, 6, 7].

Project page is available at https://jonaspflaume.github.io/impact_info/.

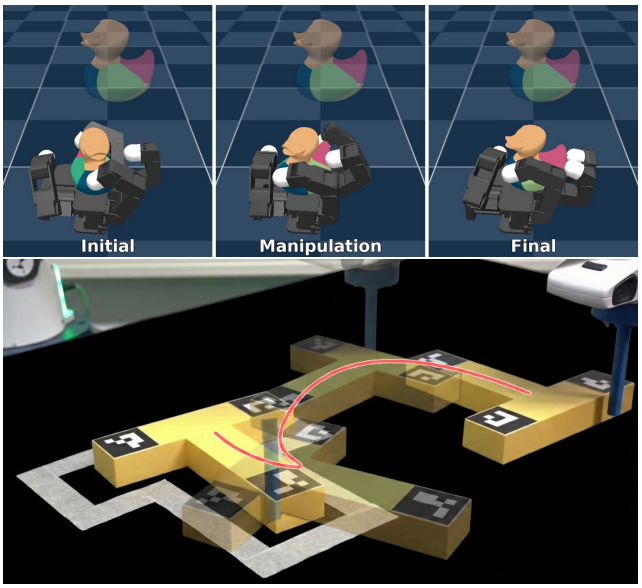

Fig. 1: IMPACT demonstrations in simulation and hardware. **Top:** Allegro Hand reorients a rubber duck in simulation. **Bottom:** a Panda robot pushes a T-shaped block to a target pose in real-robot experiments; the red curve indicates the trajectory of the block's coordinate origin during the push.

CITO commonly relies on contact models with a complementarity structure to capture the on/off nature of contact and frictional interaction, and the exact form depends on the chosen contact model (e.g., rigid vs. compliant, Coulomb vs. regularized friction) [8]. This yields a fundamentally nonsmooth optimization problem that is naturally cast as MPCCs [9]. At feasible points, complementarity induces degeneracy and violates the regularity assumptions required by standard smooth NLP constraint qualifications (CQs); consequently, standard constraint qualifications such as the *Linear Independence Constraint Qualification* and the *Mangasarian–Fromovitz Constraint Qualification* typically fail, and the associated Lagrange multipliers may become unbounded, rendering the *Karush–Kuhn–Tucker* (KKT) conditions invalid [9, 10]. Consequently, off-the-shelf NLP solvers based on standard KKT/CQ theory are often numerically brittle for MPCCs: large initial complementarity violations make convergence highly initialization-sensitive and unreliable at TO scale [11].

Due to the nonsmooth and degenerate nature of contact-implicit formulations, a variety of practical treatments have been explored to improve numerical robustness. Broadly, existing methods modify the original complementarity structure through smoothing or relaxation, or by introducing (exact) penalty terms to better fit standard solvers and their convergence assumptions [12, 13, 14]. While these treatments often improve numerical behavior, they can degrade control quality and introduce additional heuristic update parameters that strongly affect efficiency. Moreover, as these modifications are tightened to more closely approximate the original complementarity conditions, degeneracy and ill-conditioning can re-emerge, limiting reliability at TO tasks.

In this work, we propose IMPACT (*IMP*licit contact *AC*tive-set Trajectory optimization), an *implicit complementarity/contact-branch selection* method for solving MPCC-based CITO, with *stationarity guarantee* for feasible accumulation points under standard assumptions from recent augmented Lagrangian (AuLa) theory [15, 16, 17, 18]. Instead of smoothing or relaxing complementarity constraints, IMPACT retains the original nonsmooth contact constraints and handles them through the AuLa subproblems. Across these subproblems, the approximate Lagrange multipliers and penalty strength guide the selection of the active complementarity branch (contact vs. no-contact), enabling an implicit contact-mode discovery without a prescribed homotopy schedule. Fig. 2 conceptually contrasts relaxation, penalty, and IMPACT on a toy complementarity example. Moreover, this AuLa treatment prevents the multiplier blow-up typically seen in vanilla AuLa formulations for MPCCs. As a result, IMPACT provides a principled alternative to penalty-only methods. Combined with our proposed *block coordinate descent* (BCD) solver for the AuLa subproblems, IMPACT achieves substantially faster convergence than strong baselines in our experiments while maintaining competitive performance. On standard benchmarks, this translates to an order-of-magnitude speedup. Our contributions are summarized as follows.

- **Algorithm.** We introduce IMPACT, an efficient CITO method based on an AuLa scheme for MPCCs, and establish its stationarity guarantee.
- **Implementation.** We provide C++ implementations of IMPACT and the evaluation baselines, with an interface designed specifically for CITO/CI-MPC workloads.
- **Validation.** We benchmark IMPACT against representative baselines on open-source suites: (i) a CITO benchmark and (ii) a MuJoCo dexterous-manipulation benchmark for Allegro-Hand CI-MPC. We further demonstrate the method on real robotic hardware in a Push-T manipulation task.

II. Related Work

This section reviews CITO and CI-MPC methods from two perspectives: (a) approaches that *soften* contact logic to obtain smooth gradients for planning/MPC, and (b) approaches that *retain* nonsmooth contact logic and solve more directly.

A. Smooth / Relaxed Contact Models for Planning and MPC

Many CITO/CI-MPC pipelines replace rigid complementarity with *smooth* or *relaxed* surrogates to enable gradient-based optimization, which can improve numerical behavior but may affect gradient accuracy and closed-loop performance. A common approach modifies the *forward contact model* (e.g., compliant or differentiable contact), yielding smoother derivatives at the expense of strict rigid-contact fidelity [6, 2, 19, 13, 12]. Another line keeps *forward simulation* rigid but targets *differentiable optimization* by regularizing the sensitivity/KKT systems used to compute gradients through contact, enabling fast CI-MPC and relaxed-complementarity DDP variants [20, 21, 22]. Beyond local smoothing, global convex relaxations have been explored for CITO, emphasizing geometric reasoning and contact sequencing [23].

In contrast, our method adopts a *nonsmooth* AuLa viewpoint: we update relaxation via safeguarded multiplier/penalty rules and solve nonsmooth subproblems that explicitly enforce complementarity, aiming to preserve mode transitions and implicitly identify the contact mode.

B. Direct Nonsmooth Treatments of Contact Complementarity

A second class handles contact complementarity more *directly*, reflecting the inherent nonsmoothness of contact (make/break; stick/slide). Two common strategies are penalty/continuation methods and operator-splitting schemes.

Early CITO work introduces elastic (slack) variables and penalizes complementarity violations so that contacts can be "discovered" during optimization [5]. Later pipelines improve reliability via continuation strategies or automatic penalty-update rules [24, 11], and related penalty reformulations also appear in *quadratic programs with complementarity constraints* (QPCC) solvers such as LCQPow [25].

Despite their widespread use, penalty/continuation methods can become numerically stiff and brittle under large penalties, and heuristic schedules may slow convergence [11]. These issues motivate safeguarded AuLa schemes for *general* MPCCs; among penalty-based CITO planners, CRISP is closest to our approach [11].

Operator splitting / ADMM exploits CI-MPC structure by alternating trajectory updates with projections onto contact-feasible sets. The C3 family formulates CI-MPC in a consensus ADMM form with simple, parallelizable subproblems, scaling well to many-contact settings such as planar pushing and multi-contact manipulation [26, 27]. Extensions improve robustness via residual learning [28] and sampling guidance [29], while C3+ accelerates planar pushing through more efficient projections [30]. While C3+ is reminiscent of our emphasis on fast inner solves, it targets an ADMM projection setting that differs from IMPACT.

The ADMM-based methods above typically operate on linearized contact/dynamics. They often produce task-space commands, leading to QPCC-style problems and, in many implementations, an additional task-space tracking layer is required for execution [26, 30]. A recent preprint by Ménager et al. [31], released after the initial submission of IMPACT,

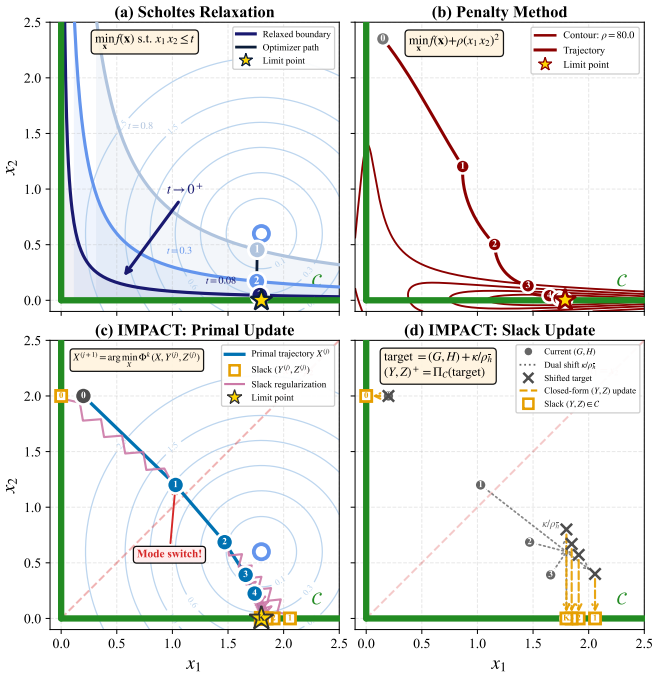

Fig. 2: Comparison of complementarity-handling methods on a 2D toy objective. Top row: Scholtes relaxation and squared-penalty method. Bottom row: IMPACT, split into the primal update and the slack update. Objective contours are shown in blue and the complementarity-feasible set is shown in green. In IMPACT, the slack variables act as anchors that regularize the next primal update, while the slack update is obtained from a dual-shifted target projected onto the complementarity set.

also uses an AuLa and alternating-minimization/BCD scheme for contact-implicit QPCCs in one-step inverse dynamics. In contrast, IMPACT targets temporally coupled TO and CI-MPC settings with general MPCC-based CITO formulations. We use an MPCC-tailored safeguarded AuLa scheme with complementarity-enforcing inner subproblems. This induces an implicit contact-mode selection behavior as slack is driven tight.

III. Problem Definition

In this work, we formulate both CITO and CI-MPC in a unified optimization framework in generalized coordinates, given in problem (1). We use $X = [x_t]_{1:T}$ to denote the trajectory of decision variables (e.g., state, control, and contact forces) over time steps $t = 1, \dots, T$. We use $r(x_t)$, $h(x_t, x_{t+1})$, and $g(x_t)$ to denote the residual function, equality constraints, and inequality constraints, respectively. $h(x_t, x_{t+1})$ consists of the standard equality constraints at time steps t and $t+1$, together with the system dynamics that couple consecutive time steps. The $G(x_t)$ and $H(x_t)$ denote the complementarity functions at time step t. The complementarity condition $0 \le G(x_t) \perp H(x_t) \ge 0$ encodes a logical exclusivity: component-wise, at least one of $G(x_t)$ or $H(x_t)$ must be zero, so they cannot be simultaneously strictly positive. Algebraically, it is equivalent to $G(x_t) \ge 0$, $H(x_t) \ge 0$, and the element-wise product constraint $G(x_t) \circ H(x_t) = 0$. Consequently, the problem is inherently nonconvex and nonsmooth. We focus on nonlinear sum-of-squares (SOS) objectives, which capture many task-space motion-planning costs. It is particularly convenient under the AuLa formulation in Section IV, since both the penalty terms and Lagrangian terms can be absorbed into the nonlinear least-squares residual, yielding an efficient Gauss-Newton scheme.

$$\min_{X} \quad \frac{1}{2}\sum_{t=1}^{T} r(x_t)^\top r(x_t) \tag{1a}$$

$$\text{s.t.} \quad h(x_t, x_{t+1}) = 0, \quad t \in 1, \dots, T-1, \tag{1b}$$

$$g(x_t) \le 0, \quad t \in 1, \dots, T, \tag{1c}$$

$$0 \le G(x_t) \perp H(x_t) \ge 0, \quad t \in 1, \dots, T. \tag{1d}$$

Problem (1) defines a general MPCC, with different discretizations and contact models recovered as special cases. For example, the template accommodates multiple friction models. With a polyhedral approximation of the friction cone, the per-step contact complementarity conditions can be written as a *linear complementarity problem* (LCP). In this case, once the contact Jacobians are fixed at each step (e.g., via a standard linearization or a "frozen Jacobian" approximation), the resulting $G(x_t)$ and $H(x_t)$ are typically affine in the decision variables. If instead friction is enforced using a second-order cone representation, the contact conditions lead to a conic complementarity formulation; this still fits within the same MPCC template, but is generally nonlinear and more challenging to solve.

In particular, our CI-MPC experiments in VI-C adopt the LCP contact model following [13] and use a time-stepping impulse-velocity formulation in the spirit of Anitescu [32]. We directly adopt the KKT system of the primal problem of the Anitescu quadratic program in [13] (Eq. (7)) and incorporate it into the IMPACT formulation:

$$\mathbf{Q}\mathbf{v}_t - \big(\mathbf{b}(\mathbf{u}_t) + \tilde{\mathbf{J}}_i^\top \boldsymbol{\beta}_t\big)/h = 0, \tag{2a}$$

$$0 \le \boldsymbol{\beta}_t \perp \tilde{\mathbf{J}}_i \mathbf{v}_t + \phi_i/h \ge 0. \tag{2b}$$

This system describes the contact dynamics at time step t, where the first line gives the discrete-time dynamics, relating the generalized velocity $\mathbf{v}_t$ through the system mass moment $\mathbf{Q}$ to the actuation/external term $\mathbf{b}(\mathbf{u}_t)$ and the contact-force variable $\boldsymbol{\beta}_t$ through the contact Jacobian $\tilde{\mathbf{J}}_i$ for the ith detected contact, including the friction-polyhedral approximation and the step size h. The second line enforces non-penetration via complementarity: the normal contact force $\boldsymbol{\beta}_t \ge 0$ can be nonzero only when the normal gap $\tilde{\mathbf{J}}_i \mathbf{v}_t + \phi_i/h$ is approximately tight, where ϕ_i denotes the normal distance for the ith contact. The generalized velocity $\mathbf{v}_t$ is then related to the generalized coordinate using time integration, possibly involving quaternion integration.

IV. A Safeguarded AuLa for MPCCs

We next describe the safeguarded AuLa outer loop used in IMPACT. This outer loop updates penalty parameters and

Lagrange multipliers while keeping complementarity enforced explicitly, and provides a stationarity-guaranteed wrapper for our BCD inner solver.

For algorithmic convenience, we rewrite Problem (1) in *vertical form* [10] by introducing auxiliary (slack) variables Y and Z. This moves nonlinearities from the complementarity set into smooth equality constraints: we enforce $G(x_t) = y_t$ and $H(x_t) = z_t$, and impose complementarity directly on the variables via $0 \le y_t \perp z_t \ge 0$. While this increases the dimension and adds equality constraints, it yields a simple complementarity set that is well-suited to our inner solver design. The resulting vertical form is:

$$\min_{X,Y,Z} \quad \frac{1}{2}\sum_{t=1}^{T} r(x_t)^\top r(x_t) \tag{3a}$$

$$\text{s.t.} \quad h(x_t, x_{t+1}) = 0, \quad t \in 1,\dots,T-1, \tag{3b}$$

$$g(x_t) \le 0, \quad t \in 1,\dots,T, \tag{3c}$$

$$G(x_t) = y_t, \quad t \in 1,\dots,T, \tag{3d}$$

$$H(x_t) = z_t, \quad t \in 1,\dots,T, \tag{3e}$$

$$0 \le y_t \perp z_t \ge 0, \quad t \in 1,\dots,T. \tag{3f}$$

Remark 1. The vertical reformulation is equivalent to the original MPCC: at any feasible point, $(y_t, z_t) = (G(x_t), H(x_t))$, so enforcing $0 \le y_t \perp z_t \ge 0$ is equivalent to $0 \le G(x_t) \perp H(x_t) \ge 0$. We adopt this split because AuLa handles the resulting smooth equalities $G(x_t) - y_t = 0$ and $H(x_t) - z_t = 0$ effectively in TO settings [33], while our inner solver exploits the resulting variable-wise complementarity structure via closed-form update (Section V).

We follow a safeguarded AuLa scheme for MPCCs in which complementarity constraints are not absorbed into the augmented Lagrangian function; instead, they are enforced explicitly in each inner solve. Concretely, we handle the smooth equality and inequality constraints via an AuLa treatment, while keeping the complementarity constraints as hard constraints in the inner subproblem. This separation preserves the MPCC structure and avoids the pathological multiplier growth that can arise when complementarity conditions are penalized directly [16, 15]. The resulting AuLa subproblem, with objective denoted by $\Phi(X, Y, Z)$, is

$$\min_{X,Y,Z} \quad \frac{1}{2}\sum_{t=1}^{T} r_t^\top r_t + \frac{\rho_{\bar{h}}}{2}\left\|\bar{h}_t + \tfrac{\kappa_t}{\rho_{\bar{h}}}\right\|^2 + \frac{\rho_g}{2}\left\|(g_t + \tfrac{\mu_t}{\rho_g})_+\right\|^2 \tag{4a}$$

$$\text{s.t.} \quad 0 \le y_t \perp z_t \ge 0, \quad t \in \{1,\dots,T\}. \tag{4b}$$

For brevity, we omit explicit dependence on the variables and stack the equality constraints in (3) at time step t as $\bar{h}_t(x_{t+1}, x_t, y_t, z_t) := [\, h(x_t, x_{t+1});\ G(x_t) - y_t;\ H(x_t) - z_t\,]$. At the terminal time step $t = T$, $h(x_t, x_{t+1})$ reduces to $h(x_t)$, as no dynamics constraint is imposed beyond the terminal state. The operator $(\cdot)_+$ denotes the element-wise positive part, $(x)_+ := \max(x, 0)$. We use $\rho_{\bar{h}}$ and ρ_g as penalty factors for the equality and inequality constraints, with corresponding multipliers κ_t and μ_t. The resulting subproblem (4) is a nonlinear least-squares problem with simple complementarity constraints.

Algorithm 1 Safeguarded AuLa outer loop for MPCCs

Require: Initial multipliers (κ^0, μ^0), penalties $(\rho_{\bar{h}}^0, \rho_g^0)$, safeguard bounds $\mu_{\max} > 0$ and $\kappa_{\min} < \kappa_{\max}$, penalty increase factor $\gamma > 1$, constraints violation factor $\eta \in [0, 1]$, inner problem tolerances $\{\varepsilon_k\} \downarrow 0$, outer problem tolerances ϵ_w, ϵ_g, $\epsilon_{\bar{h}}$.

1: Initialize w^0
2: Set $k \leftarrow 0$. Initial variable-change $||\Delta w^0||_\infty \leftarrow \infty$.
3: **while** $||\triangle w^k||_\infty \ge \epsilon_w$ or $||(g(X^k))_+||_\infty \ge \epsilon_g$ or $||\bar{h}(w_k)||_\infty \ge \epsilon_{\bar{h}}$ **do**
4: **Safeguard multipliers:**

$$\bar{\kappa}^k \leftarrow \Pi_{[\kappa_{\min}, \kappa_{\max}]}(\kappa^k), \qquad \bar{\mu}^k \leftarrow \Pi_{[0, \mu_{\max}]}(\mu^k),$$

where Π denotes component-wise clipping.
5: **Inner solve:** approximately solve the AuLa subproblem (4) with $\bar{\mu}^k$, $\bar{\kappa}^k$ and obtain $w^{k+1} = (X^{k+1}, Y^{k+1}, Z^{k+1})$ such that the inner stationary residual is below ε_k.
6: Evaluate $\bar{h}_t$, g_t using w^{k+1}, $\triangle w^k \leftarrow w^{k+1} - w^k$
7: **Multiplier update:**

$$\kappa^{k+1} \leftarrow \bar{\kappa}^k + \rho_{\bar{h}}^k\, \bar{h}_t, \qquad \mu_t^{k+1} \leftarrow \big(\bar{\mu}_t^k + \rho_g^k\, g_t\big)_+,$$

8: $\zeta_t^{k+1} \leftarrow \min\big(\mu_t^{k+1}, -g_t\big)$
9: **Penalty update:**
10: **if** $k = 0$ **or** $\max\{\|\zeta^{k+1}\|_\infty, \|\bar{h}(w^{k+1})\|_\infty\} \le \eta \max\{\|\zeta^k\|_\infty, \|\bar{h}(w^k)\|_\infty\}$ **then**
11: $\rho_{\bar{h}}^{k+1} \leftarrow \rho_{\bar{h}}^k$; $\rho_g^{k+1} \leftarrow \rho_g^k$.
12: **else**
13: $\rho_{\bar{h}}^{k+1} \leftarrow \gamma\, \rho_{\bar{h}}^k$; $\rho_g^{k+1} \leftarrow \gamma\, \rho_g^k$.
14: **end if**
15: $k \leftarrow k + 1$.
16: **end while**

To promote global convergence of the outer loop, we use the safeguarded multiplier and penalty updates in Algorithm 1. Updates are applied component-wise across all time steps, and multipliers are clipped to fixed safeguard bounds to ensure they remain bounded. Lines 8–14 implement a standard safeguard for penalty updates: we monitor the decrease of the equality-feasibility residual $\|\bar{h}(w^{k+1})\|_\infty$ and an inequality complementarity/KKT residual $\|\zeta^{k+1}\|_\infty$ computed after the multiplier update; if the combined violation does not decrease sufficiently, we increase the penalty parameters. This is one possible safeguard design, and other update rules that enforce a comparable sufficient-decrease condition can be used as well.

The accuracy of each inner solve directly affects the convergence behavior of the AuLa outer loop. To formalize this connection, we introduce a computable KKT residual for the AuLa subproblem (4) and use it as the inner-solve accuracy criterion in line 5 of Algorithm 1. This residual verifies: (i) stationarity of the augmented objective, (ii) feasibility of the complementarity set $0 \le Y \perp Z \ge 0$, and (iii) consistency of

the complementarity multipliers with the limiting normal cone of this set. The full definition of the KKT residual is given in Appendix A.

Definition 2 (ϵ-stationarity)**.** At outer iteration k, let Φ^k denote the AuLa objective in (4) with safeguarded multipliers and penalty factors. We say $w^k = (X^k, Y^k, Z^k)$ is *ϵ-stationary* for (4) if

$$r_{\text{in}}(w^k) \doteq \inf_{(u,v)\in\mathcal{M}_M(w^k)} r_{\text{in}}(w^k; u, v) \le \epsilon,$$

where r_{in} is computed with respect to Φ^k, and $\mathcal{M}_M(w^k)$ is the branch-dependent multiplier set defined in Appendix A.

The residual r_{in} is chosen so that the inner and outer accuracy criteria are consistent. It gives a computable certificate for the AuLa subproblem and is compatible with the gradient-mapping measure controlled by the inner solver introduced later in Section V. Therefore, the condition $r_{\text{in}} \le \varepsilon$ is both the accuracy requirement used by the safeguarded AuLa analysis and a natural stopping measure for the inner routine. The formal statement and assumptions are deferred to Appendix B.

We rephrase the global convergence/stationarity result of [16, Thm. 3] (and the closely related statement in [15, Cor. 4.4]) in terms of the KKT residual:

Theorem 3 (Global convergence of safeguarded AuLa under vanishing inner residual)**.** Let $\{w^k\}$ be the sequence generated by Algorithm 1. Assume that the safeguarded multipliers $(\bar\kappa^k, \bar\mu^k)$ remain bounded (e.g., line 4 of Algorithm 1). Suppose the AuLa subproblem (4) is solved inexactly in the sense that, for each k, there exist (u^k, v^k) such that

$$r_{\text{in}}(w^k; u^k, v^k) \;\le\; \varepsilon_k, \qquad \varepsilon_k \downarrow 0.$$

Let $w^\star$ be a feasible accumulation point of $\{w^k\}$ for (3) (equivalently, for (1)). If the MPCC regularity condition assumed in [16, Thm. 3] holds at $w^\star$, then $w^\star$ is a first-order stationary point of the MPCC (in the sense of [16, Thm. 3]).

Therefore, to ensure the *attainability* of stationarity, we design an inner solver to meet the Definition 2 for any prescribed tolerance $\varepsilon > 0$ within finitely many inner iterations. We present such a solver in the next section.

V. A BCD Inner Solver for AuLa Subproblems

We solve the AuLa inner subproblem (4) with a two-block coordinate descent (BCD) scheme. Each BCD iteration alternates between updating the trajectory variables X and updating the auxiliary complementarity variables (Y, Z). The X-update is carried out by a Gauss-Newton update with globalization, and the (Y, Z)-update admits a closed-form solution under the complementarity constraints.

a) Block 1: X-update (Gauss–Newton with globalization): With (Y, Z) fixed, the inner objective reduces to a smooth nonlinear least-squares problem in X. We solve it using a damped Gauss–Newton method with an Armijo backtracking line search to ensure sufficient decrease of the inner objective Φ^k.

b) Block 2: (Y, Z)-update (closed-form minimization over a union of cones): With X fixed, the (Y, Z)-subproblem decouples across time steps and complementary pairs. For ith scalar pair at time t: $(y_{t,i}, z_{t,i})$ subject to $0 \le y_{t,i} \perp z_{t,i} \ge 0$, the feasible set is the union of two convex cones, $\mathcal{C} = \{(y, 0) \mid y \ge 0\} \cup \{(0, z) \mid z \ge 0\}$. Thus we evaluate two closed-form candidates and select the one yielding the smaller quadratic penalty contribution in (4).

With the safeguarded multipliers at outer iteration k and current penalty factor $\rho_{\bar h}$:
Case 1 ($z_{t,i} = 0$):

$$y^\star_{t,i} = \max\Big(0,\; G_i(x_t) + \tfrac{1}{\rho_{\bar h}}\,\kappa_{G,t,i}\Big), \qquad z^\star_{t,i} = 0. \tag{5}$$

Case 2 ($y_{t,i} = 0$):

$$z^\star_{t,i} = \max\Big(0,\; H_i(x_t) + \tfrac{1}{\rho_{\bar h}}\,\kappa_{H,t,i}\Big), \qquad y^\star_{t,i} = 0. \tag{6}$$

We then pick between (5) and (6) by comparing their resulting objective values (equivalently, their quadratic penalty contributions). This step is an *exact* minimization over the complementarity set for fixed X and hence never increases Φ^k. We summarize the BCD approach as Algorithm 2. A graphical depiction of the algorithm is shown in Fig. 2.

Remark 4 (Why this differs from the ADMM projection in [30]). The ADMM projection in [30] is primarily feasibility-driven in consensus step: it directly projects the copied variable onto the complementarity-feasible set. In contrast, our closed-form update corresponds to a *shifted* mapping induced by the current approximate Lagrange multipliers and penalty weights. Unlike an ADMM consensus update, this step does not introduce copied variables; it is obtained by directly minimizing the AuLa subproblem with respect to the slack variables. This mapping injects the influence of the current multiplier/penalty into the slack variables, acting like a spring force that regularizes the primal update. As illustrated in Fig. 2, IMPACT therefore provides more than a pure feasibility projection and can yield substantially faster progress.

Theorem 5 (Attainability of ε-stationarity by the BCD inner solver)**.** Fix an outer iteration k and consider the BCD inner iterates $\{w^{(j)}\} = \{(X^{(j)}, Y^{(j)}, Z^{(j)})\}$ generated by Algorithm 2. Assume: (i) Φ^k is continuously differentiable, $\nabla\Phi^k$ is Lipschitz on the bounded set visited by the iterates and Φ^k is bounded below on $\{(X, Y, Z) : (Y, Z) \in \mathcal{C}\}$; (ii) the X-update is globalized and achieves sufficient decrease (e.g., damped GN with Armijo line search); and (iii) the (Y, Z)-update is an exact minimization over the complementarity set for fixed X (as in (5)–(6)). Then, for any tolerance $\varepsilon_k > 0$, there exists a finite inner iteration index j_k and multipliers $(u^{(j_k)}, v^{(j_k)})$ such that

$$r_{\text{in}}\big(w^{(j_k)}; u^{(j_k)}, v^{(j_k)}\big) \;\le\; \varepsilon_k.$$

In particular, the inner BCD loop can be terminated after finitely many iterations with an output w^k that is ε_k-stationary in the sense of Definition 2.

Algorithm 2 Inner Solver: BCD for the AuLa subproblem (4)

Require: Outer index k, safeguarded multipliers $(\bar{\kappa}^k, \bar{\mu}^k)$, penalties (ρ_h^k, ρ_g^k), initial $(X^{(0)}, Y^{(0)}, Z^{(0)})$, stagnation tolerance τ_k

Set $w^{(0)} \leftarrow (X^{(0)}, Y^{(0)}, Z^{(0)})$ and evaluate $\Phi^k(w^{(0)})$.
for $j = 0, 1, 2, \dots$ **do**
    X**-update:** compute a damped GN update on $\Phi^k(\cdot, Y^{(j)}, Z^{(j)})$ with backtracking line search; obtain $X^{(j+1)}$.
    (Y, Z)**-update:** for each (t, i), compute candidates (5) and (6) and select the one with smaller objective; obtain $(Y^{(j+1)}, Z^{(j+1)})$.
    Set $w^{(j+1)} \leftarrow (X^{(j+1)}, Y^{(j+1)}, Z^{(j+1)})$ and evaluate $\Phi^k(w^{(j+1)})$.
    Practical stopping: if $|\Phi^k(w^{(j)}) - \Phi^k(w^{(j+1)})| \le \tau_k$,
        return $w^k \leftarrow w^{(j+1)}$.
end for

The proof is given in Appendix B: sufficient decrease makes the X-block residual attainable, and exact (Y, Z)-minimization eliminates the complementarity-block part of the KKT residual.

Inner-solver guarantee and practical stopping. The inner optimizer is *not* intended to certify global optimality of the nonconvex subproblem. For the purpose of *outer-loop convergence analysis*, we adopt an *attainability* interface: for any tolerance $\varepsilon > 0$, the inner procedure can produce (in finitely many iterations) an iterate whose stationarity measure is below ε under suitable multipliers (Theorem 5). Our implementation, however, uses an *inexact/budgeted* inner solve: we do not explicitly test stationarity and instead stop when the inner augmented objective stagnates, $|\Phi^k(w^{(j)}) - \Phi^k(w^{(j+1)})| \le \tau_k$. Empirically, this criterion consistently yields sufficiently small constraint and complementarity residuals and stable outer-loop progress on all benchmarks; see Appendix B for definitions and additional diagnostics.

VI. Experiments

In this section, we evaluate our planner on two open-source benchmarks. For long-horizon planning, we use the CITO benchmark from CRISP [11]. For dexterous manipulation under multi-contact dynamics, we use the CI-MPC benchmark from [13]. All experiments are run on a desktop PC with an AMD Ryzen 9 7950X3D CPU, with no GPU acceleration. Our IMPACT implementation uses no explicit multi-threading (e.g., OpenMP/TBB). It links against BLAS/LAPACK, uses Eigen's sparse LDLT factorization for linear solves, and updates slack pairs using Eigen's SIMD-vectorized operations. We use CasADi's C++ interface for modeling and symbolic differentiation [34]. CasADi CodeGen is disabled for both CITO and CI-MPC.

A. Long Horizon CITO Benchmark

The CRISP CITO benchmark comprises six nonlinear, hybrid-dynamics, contact-rich problems for long-horizon planning. The CRISP paper compares standard NLP solvers on four tasks. We exclude the one task that requires a manually specified initial guess, which is outside our scope, and evaluate on three tasks: Push Box, Cart Transport, and Push T.

Rather than comparing against generic NLP solvers that do not explicitly handle complementarity constraints and are often brittle on MPCCs (e.g., SNOPT and PROXNLP [11]), we compare against solution strategies that are known to perform well in practice: **Scholtes relaxation (SR)**, the square **penalty method (PM)**, and **CRISP**. To handle the long planning horizons in these benchmarks, we formulate all evaluated methods using multiple shooting. We implement SR and PM using the IPOPT C++ interface [35], with symbolic automatic differentiation provided by the CasADi C++ interface. For **CRISP**, we directly use the released C++ implementation provided by the authors, which is specialized to the benchmark problems.

For **SR**, we enforce the nonnegativity constraints $G(x) \ge 0$ and $H(x) \ge 0$ explicitly as inequalities, which IPOPT handles reliably, and relax complementarity via $G(x)H(x) \le t$. We then solve a sequence of relaxed problems, initialized with $t = 1.0$ and reducing t by a factor of 10 at each outer iteration until $t = 10^{-10}$. In our experiments, this continuation schedule provides a robust trade-off between numerical stability and runtime. The outer loop terminates early when the complementarity residual falls below a prescribed tolerance. For efficiency, we warm-start IPOPT using the previous iterate's primal and dual variables.

For **PM**, we keep $G(x) \ge 0$ and $H(x) \ge 0$ as explicit inequality constraints and penalize complementarity using a squared penalty term with a relatively large penalty weight. Empirically, this configuration is stable and often more efficient than starting from a small penalty and gradually increasing it. While a large penalty can make the inner NLP more ill-conditioned, we find IPOPT remains effective in this regime and can be more efficient than an iterative NLP scheme.

The benchmark problems are shown in Fig. 3, and we provide detailed descriptions in Appendix C. We remark that these tasks require forward integration of nonlinear dynamics and include general equality and inequality constraints, so they are not directly cast as LCPs solvable by standard LCP solvers. We report problem sizes in terms of decision variables, complementarity pairs, and constraint counts: *Push Box* has 453 variables, 500 complementarity pairs, and 150 dynamics constraints. *Push T* has 1353 variables, 2150 complementarity pairs, 150 dynamics constraints, 350 equality constraints, and 200 inequality constraints. *Cart Transport* has 2404 variables, 900 complementarity pairs, 1200 dynamics constraints, 300 equality constraints, and 1200 inequality constraints.

Solver Settings and Performance Metrics: We use an all-zero initialization for every solver on all tasks, following the standardized CRISP comparison setup. This choice avoids task-specific initial guesses and makes the comparison reproducible across methods. We note, however, that warm-started initialization is common in practical TO/MPC pipelines, and the relative advantage of IMPACT may differ in such settings.

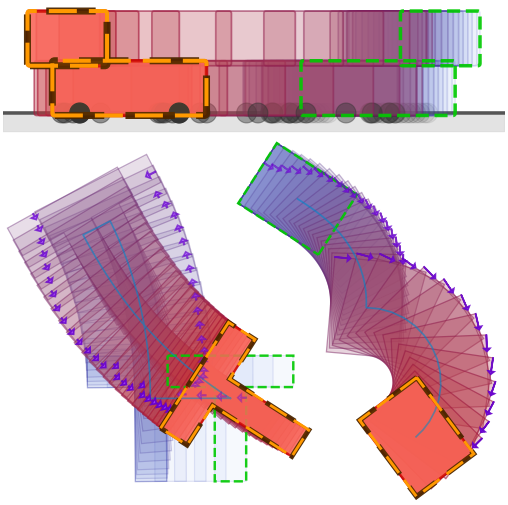

Fig. 3: IMPACT planning demos on three CITO tasks. The green dashed box marks the start pose and the orange dashed box marks the goal pose. Purple arrows visualize contact forces, and the solid green line traces the object-origin trajectory. Color indicates time from early (light blue) to late (dark orange).

For each task, all methods (SR/PM/IMPACT and CRISP) are synchronized to solve the same optimization problem with identical objective-function weights. For *Push Box* and *Push T*, we use a horizon of $T = 50$ with $\Delta t = 0.05$; for *Cart Transport*, we use $T = 300$ with $\Delta t = 0.02$. For *Push Box* and *Push T*, we randomly sample 50 goal poses, and for *Cart Transport*, we randomly sample 50 start-goal pairs.

We run each planner until it satisfies a common set of termination criteria. Specifically, we require all constraint violations to be below 10^{-5}, including (for example) the complementarity violation $\|G(x) \circ H(x)\|_\infty \leq 10^{-5}$. A run is declared a failure if (i) it reaches the maximum iteration limit (2000 in our experiments) while still violating either stopping criterion, or (ii) the resulting trajectory fails to reach the goal (e.g., becoming stuck in an intermediate configuration), a failure mode also reported in the CRISP paper. Otherwise, the run is counted as successful. We report success rate, total tracking error, iteration count, and wall-clock runtime in Table I. We report total tracking error as an end-to-end task metric; lower values typically indicate trajectories that reach the goal sooner and maintain better tracking, which often correlates with more efficient contact-mode progress. For IMPACT, we report the number of BCD sweeps, where one sweep corresponds to a full pass over all blocks. For CRISP, we report the total number of QP solves (including those from second-order corrections) as the subproblem-count metric. The MPCC formulations and all solver hyperparameters are provided in Appendix C.

Table I shows a quality-speed trade-off across solvers. In terms of *mode/trajectory quality* (total tracking error), SR is best on *Push Box* and *Push T*, while CRISP is best on *Cart Transport*. IMPACT remains competitive: it is second-best on *Push Box* and *Cart Transport*, and close to CRISP on *Push T*. In contrast, PM yields the largest tracking errors and is also less robust, with failures on *Push T* (90% success) and *Push Box* (98% success). In terms of *efficiency*, IMPACT provides a dominant runtime advantage. Despite PM having fewer iterations, the penalized subproblems can be ill-conditioned, so wall-clock time does not scale with iteration count. We find that driving the complementarity violation down to the 10^{-5} level typically requires multiple continuation iterations in CRISP, leading to increased runtime. SR also remains comparatively expensive, as we find that warm-starting does not substantially reduce the iteration count in practice. Quantitatively, IMPACT is $16.8\times$, $25.0\times$, and $34.0\times$ faster than CRISP on *Box*, *T*, and *Cart*, respectively (geometric mean $24.3\times$). Even relative to the fastest baseline in time (PM), IMPACT achieves $5.7\times$, $2.9\times$, and $25.0\times$ speedups (geometric mean $7.4\times$). Across all baseline comparisons in Table I, this corresponds to a $2.9\times$–$70\times$ speedup range with a geometric mean of $13.8\times$.

TABLE I: CITO benchmark results reported as mean $\pm$ 95% confidence interval (CI). For each task and metric, the best method is highlighted in **red** and the second best in **blue**.

Task	Solver	Success (%)	Track. Err. ↓	Iters ↓	Time (s) ↓
Box	SR	100.0	**19.46 ± 2.30**	**164 ± 12**	1.27 ± 0.10
	PM	98.0	58.00 ± 5.11	**66 ± 13**	**0.85 ± 0.17**
	CRISP	100.0	51.90 ± 4.08	641 ± 358	2.52 ± 1.34
	IMPACT	100.0	**26.99 ± 2.81**	219 ± 49	**0.15 ± 0.03**
T	SR	100.0	**1.24 ± 0.13**	**202 ± 11**	5.50 ± 0.38
	PM	90.0	21.99 ± 2.57	**117 ± 14**	**2.96 ± 0.39**
	CRISP	100.0	**8.03 ± 0.71**	1243 ± 407	25.73 ± 6.92
	IMPACT	100.0	8.48 ± 1.94	606 ± 225	**1.03 ± 0.44**
Cart	SR	100.0	433.08 ± 44.92	261 ± 22	5.63 ± 0.57
	PM	100.0	408.65 ± 42.32	**87 ± 14**	**2.00 ± 0.32**
	CRISP	100.0	**235.10 ± 21.70**	501 ± 381	2.72 ± 2.05
	IMPACT	100.0	**381.89 ± 32.35**	**101 ± 15**	**0.08 ± 0.01**

B. Real-Robot Push-T Demonstration

To evaluate IMPACT beyond simulation, we deploy it on a real robotic pushing setup for the Push-T task as a qualitative hardware demonstration (Fig. 1). We first solve a full-horizon TO problem to obtain a reference pushing trajectory, and then execute it on the robot by commanding the end-effector to emulate the pusher and track the reference. If the object deviates too much from the nominal trajectory, we replan online. To improve sim-to-real fidelity, we tune the rotational *contact-radius* constraint in the contact model based on observed rollouts. We run 10 trials with different initial object configurations; IMPACT reaches the target in all trials, achieving a 100% success rate. Additional videos are provided in the supplementary material.

C. Multi-contact High Dimensional CI-MPC Benchmark

To evaluate the scalability of IMPACT in a realistic CI-MPC setting, we use the Allegro Hand meshable-object re-orientation benchmark introduced in [13]. Following [13], we refer to the overall evaluation setup under their framework as

		Airpl	Binoc	Bowl	Bunny	Camer	Can	Cube	Cup	Eleph	Foamb	Mug	Piggy	Rubbe	Stick	Teapo	Torus	Water	Avg
Success (%)	**IMPACT**	80	**100**	90	100	**90**	100	80	100	**100**	100	100	90	**100**	30	100	100	100	**91.8±4.1**
	C-Free (0.1)	90	70	**100**	100	50	100	**100**	100	90	100	100	**100**	**100**	50	100	100	100	**91.2±4.2**
	C-Free (0.2)	**100**	80	**100**	100	60	100	90	100	90	100	100	**100**	90	**70**	100	100	100	**92.9±3.9**
Ctrl Var	**IMPACT**	**1.97**	**1.49**	**1.59**	**1.16**	**1.78**	**1.49**	**1.53**	**1.29**	**1.36**	**1.58**	**1.66**	**1.96**	**1.69**	**1.63**	**1.27**	**1.79**	**1.08**	**1.55±0.13**
	C-Free (0.1)	2.60	2.47	2.32	1.90	2.26	1.50	2.28	1.77	2.24	1.85	2.20	2.34	2.45	2.46	2.47	2.44	1.09	**2.16±0.21**
	C-Free (0.2)	5.28	4.62	5.15	3.35	5.00	3.03	4.62	4.75	4.78	3.75	3.76	5.50	5.15	5.65	4.51	4.50	2.11	**4.44±0.49**
Smooth	**IMPACT**	**0.045**	**0.027**	**0.032**	**0.018**	**0.035**	0.022	**0.028**	**0.015**	**0.024**	**0.032**	**0.034**	**0.042**	**0.034**	**0.031**	**0.022**	**0.039**	0.018	**0.029±0.004**
	C-Free (0.1)	0.060	0.046	0.045	0.045	0.036	**0.018**	0.051	0.026	0.046	0.033	0.037	0.043	0.051	0.051	0.070	0.071	**0.014**	**0.044±0.008**
	C-Free (0.2)	0.109	0.121	0.106	0.072	0.077	0.045	0.099	0.108	0.103	0.067	0.067	0.097	0.086	0.090	0.112	0.097	0.035	**0.088±0.012**
Effort	**IMPACT**	**4.8**	4.0	**1.7**	**1.7**	**2.3**	**2.6**	6.0	**1.6**	**2.0**	4.1	**2.0**	**1.9**	**1.6**	**8.7**	**1.7**	**3.5**	**1.2**	**3.0±1.0**
	C-Free (0.1)	6.3	5.7	2.0	2.1	12.5	4.5	**5.7**	2.0	2.4	**1.8**	3.4	6.4	3.6	15.2	2.1	3.7	1.4	**4.8±2.0**
	C-Free (0.2)	9.5	**3.2**	6.3	3.3	12.2	10.3	17.9	4.6	3.5	3.0	4.0	5.0	6.7	40.4	3.4	4.2	1.9	**8.2±4.8**

Fig. 4: Allegro-Hand CI-MPC benchmark results on 17 objects. We compare IMPACT against *cfree*(0.1) and *cfree*(0.2), which use control bounds $[-0.1, 0.1]$ and $[-0.2, 0.2]$, respectively. Reported metrics include **success rate** (**Success**) and control-quality measures: control variance (**Ctrl Var**), control smoothness (**Smooth**), and control effort (**Effort**). Table columns use these abbreviations. For each object, the best value for each metric is highlighted, and the final column reports the aggregate results with a 95% confidence region across all 17 objects.

cfree in the remainder of the paper. As a point of reference, the *cfree* pipeline relies on **smoothing-based** contact modeling together with the interior-point NLP solver IPOPT, and achieves a 50-60 Hz CI-MPC on our test PC. However, smoothing-based CI-MPC can exhibit undesired closed-loop behavior: smoothed contact transitions may degrade control smoothness and induce oscillations during execution. In contrast, our goal in this experiment is to demonstrate that IMPACT scales to this benchmark and achieves a practically useful control rate of around 10 Hz on the same hardware. Importantly, this rate is achieved while directly solving a *multi-contact, nonsmooth, MPCC-based* MPC problem *without* smoothing or relaxation. Such problems are widely considered challenging for real-time control due to the combinatorial mode structure and nonsmooth contact dynamics.

The *cfree* benchmark includes 17 objects spanning a wide range of surface curvatures and shapes; see Fig. 4 for the object set. The Allegro-hand setup can involve up to 20 simultaneous detected contact points; the contact complementarity conditions below are instantiated for each detected contact. We adhere closely to the cfree setup and keep all hyperparameters unchanged, including (but not limited to) the simulation time step, friction coefficients, and inertial parameters. The full list of parameters is provided in Appendix C.

In IMPACT, we optimize over the generalized velocities $\mathbf{v}_t$, contact forces $\boldsymbol{\beta}_t$ and control $\mathbf{u}_t$; given the relatively short MPC horizon, we use a single-shooting formulation and eliminate the position states by forward integration. The MPC horizon matches *cfree* and is set to 4. Following *cfree*, we compute the contact Jacobian only at the MPC initial state and hold it fixed over the prediction horizon.

We keep the same objective structure as *cfree*, but replace its quaternion inner-product term with a log-map rotation residual in $\mathbb{R}^3$, yielding a standard nonlinear least-squares form that better matches our Gauss–Newton pipeline. To keep the relative influence of the rotation terms comparable, we increase its weight accordingly. To improve closed-loop damping, we additionally include a small penalty on object velocity; in contrast, *cfree* primarily introduces damping via penalties on contact forces expressed in the contact frame. We do not perform per-object tuning: as in *cfree*, we use a single shared set of control parameters across all 17 objects. For initialization, we follow the same strategy as *cfree* by adding small random perturbations to the object's start and goal poses in hand. We also adopt the same success criteria as *cfree*: a squared position-norm error below 0.02 and a quaternion inner-product error below 0.04. Each object is evaluated over 10 trials. The total simulation horizon is 1000 time steps with a time step of 0.1 s. One key difference from the default *cfree* setup is the control bound: we use $[-0.1, 0.1]$ to promote a locally stable contact mode. For comparison, we also report *cfree* results with control bounds $[-0.1, 0.1]$ and $[-0.2, 0.2]$. Our IMPACT experiments use MuJoCo's C API via a C++ interface to match our optimizer implementation, whereas we run *cfree* using its released Python implementation. The experimental results are reported in Fig. 4. Besides the success rate, we report three control-quality metrics. Control variance is defined as $\frac{1}{d}\sum_{i=1}^{d}\mathrm{Var}(\mathbf{b}_i)$, where d is the dimension of the control signal. Control smoothness is computed as $\frac{1}{T-2}\sum_{t=1}^{T-2}\|\mathbf{u}_{t+1}-\mathbf{u}_t\|_2^2$, and control effort as $\sum_{t=1}^{T-1}\|\mathbf{u}_t\|_2^2$, where T is the trial length.

Fig. 4 summarizes performance across the 17 objects. IMPACT achieves a success rate comparable to *cfree* (avg. $91.8\%\pm4.1$ vs. $91.2\%\pm4.2$ for *cfree*(0.1) and $92.9\%\pm3.9$ for *cfree*(0.2)), while consistently improving control quality: variance 1.55 ± 0.13 (vs. 2.16 ± 0.21 and 4.44 ± 0.49), smoothness 0.029 ± 0.004 (vs. 0.044 ± 0.008 and 0.088 ± 0.012), and effort 3.0 ± 1.0 (vs. 4.8 ± 2.0 and 8.2 ± 4.8) for *cfree*(0.1/0.2) [13]. Increasing *cfree*'s control bound from $[-0.1, 0.1]$ to $[-0.2, 0.2]$ preserves success rate but substantially worsens these control-quality metrics, consistent with more oscillatory closed-loop behavior under aggressive inputs. The *stick* object is a notable outlier where IMPACT performs worse; we attribute this to the heightened sensitivity of long, slender geometries to LCP-based contact modeling, which can induce frequent mode switching. In terms of efficiency, IMPACT completes successful trials in 66.5 ± 10.9 MPC steps, comparable to *cfree*(0.1) 71.4 ± 13.5 and *cfree*(0.2) 57.8 ± 14.6, but runs

at a lower control rate of 9.53 ± 0.36 Hz (vs. 50.6 ± 0.86 and 54.0 ± 1.30 Hz).

VII. Limitations

a) Contact-mode stability under limited compute: In the Allegro-hand CI-MPC experiments, we observe a transient regime before contact is firmly established in which the optimizer may switch between nearby contact sets. This behavior is most pronounced under tight real-time compute budgets, where only coarse inner solves are feasible and the contact-branch selections can vary across iterations. Despite this, IMPACT scales well to high-dimensional MPCC-based CI-MPC and achieves strong aggregate performance on the benchmark. Incorporating additional stabilization mechanisms (e.g., trust-region [36] or hysteresis on mode updates) is a promising direction to further reduce contact-set switching in this regime.

b) Local planning and dependence on contact-encouraging costs: Under a limited compute budget, IMPACT in the Allegro-hand setup operates as a local planner. While it is effective and yields smooth behavior in many cases, it can be sensitive to local minima and depends on a contact-encouraging cost adopted from [13]. This limitation is most visible on the stick object, where slender geometry makes contact maintenance highly sensitive. Such heuristic contact terms can increase sensitivity to hyperparameters and contribute to residual oscillations; the smoother Push-T results suggest that global geometric encodings can reduce this reliance. Promising directions include incorporating offline reinforcement learning to provide global guidance and reduce reliance on hand-crafted contact costs, as well as exploring object representations beyond local parameterizations to improve robustness. We leave these directions to future work.

VIII. Conclusion

We presented IMPACT, an implicit contact active-set augmented Lagrangian method for fast contact-implicit trajectory optimization. IMPACT combines a safeguarded AuLa outer loop for MPCCs with a structured block coordinate descent inner solver, enabling efficient contact-mode discovery without a prescribed continuation schedule while retaining the original nonsmooth complementarity structure.

We implemented IMPACT in C++ and evaluated it on two open-source benchmarks. On the CRISP long-horizon CITO suite, IMPACT achieves substantial runtime gains over strong baselines, with speedups ranging from $2.9\times$ to $70\times$ (geometric mean $13.8\times$). In particular, compared with CRISP, IMPACT is $16.8\times$, $25.0\times$, and $34.0\times$ faster on Push Box, Push T, and Cart Transport, respectively (geometric mean $24.3\times$), while maintaining competitive task-level tracking performance under the same feasibility/stationarity termination criteria.

On the high-dimensional multi-contact CI-MPC benchmark for dexterous manipulation, we demonstrate that the proposed IMPACT architecture scales beyond offline planning to closed-loop control. IMPACT scales to multi-contact dynamics in generalized coordinates and, at comparable success rates to the baseline, yields improved control-quality metrics (lower control variance/smoothness/effort). Finally, we demonstrate the proposed method on real robotic hardware in a Push-T manipulation task.

Acknowledgments

This work was supported by the DFG Emmy Noether Programme (CH 2676/1-1), the EU Horizon Europe projects MANiBOT (101120823) and ARISE (101135959), the BMFTR project RIG (16ME1001), and the ERC project SIREN (101163933). We also acknowledge support from the hessian.AI Service Center (BMFTR, 16IS22091), the hessian.AI Innovation Lab (S-DIW04/0013/003), TAM, RAI, Google, and the Alfried Krupp Foundation.

References

[1] Karim Bouyarmane, Stéphane Caron, Adrien Escande, and Abderrahmane Kheddar. Multi-contact motion planning and control. In Amit Goswami and Prahlad Vadakkepat, editors, *Humanoid Robotics: A Reference*. Springer Nature, 2019. doi: 10.1007/978-94-007-6046-2_32.

[2] Tao Pang, HJ Terry Suh, Lujie Yang, and Russ Tedrake. Global planning for contact-rich manipulation via local smoothing of quasi-dynamic contact models. *IEEE Transactions on Robotics*, 39(6):4691–4711, 2023.

[3] Wilson Jallet, Antoine Bambade, Etienne Arlaud, Sarah El-Kazdadi, Nicolas Mansard, and Justin Carpentier. Proxddp: Proximal constrained trajectory optimization. *IEEE Transactions on Robotics*, 2025.

[4] Carlos Mastalli, Rohan Budhiraja, Wolfgang Merkt, Guilhem Saurel, Bilal Hammoud, Maximilien Naveau, Justin Carpentier, Ludovic Righetti, Sethu Vijayakumar, and Nicolas Mansard. Crocoddyl: An efficient and versatile framework for multi-contact optimal control. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[5] Michael Posa, Cecilia Cantu, and Russ Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2014.

[6] Aykut Özgun Önol, Philip Long, and Taşkın Padır. Contact-implicit trajectory optimization based on a variable smooth contact model and successive convexification. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2447–2453. IEEE, 2019.

[7] Jean-Pierre Sleiman, Farbod Farshidian, and Marco Hutter. Versatile multicontact planning and control for legged loco-manipulation. *Science Robotics*, 8(81):eadg5014, 2023. doi: 10.1126/scirobotics.adg5014.

[8] Quentin Le Lidec, Wilson Jallet, Louis Montaut, Ivan Laptev, Cordelia Schmid, and Justin Carpentier. Contact models in robotics: a comparative analysis. *IEEE Transactions on Robotics*, 2024.

[9] Holger Scheel and Stefan Scholtes. Mathematical programs with complementarity constraints: Stationarity,

optimality, and sensitivity. *Mathematics of Operations Research*, 25(1):1–22, 2000.

[10] Armin Nurkanović, Anton Pozharskiy, and Moritz Diehl. Solving mathematical programs with complementarity constraints arising in nonsmooth optimal control: A. nurkanović et al. *Vietnam Journal of Mathematics*, 53 (3):659–697, 2025.

[11] Yulin Li, Haoyu Han, Shucheng Kang, Jun Ma, and Heng Yang. On the Surprising Robustness of Sequential Convex Optimization for Contact-Implicit Motion Planning. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.047.

[12] Vince Kurtz, Alejandro Castro, Aykut Özgün Önol, and Hai Lin. Inverse dynamics trajectory optimization for contact-implicit model predictive control. *The International Journal of Robotics Research*, 45(1):23–40, 2026.

[13] Wanxin Jin. Complementarity-Free Multi-Contact Modeling and Optimization for Dexterous Manipulation. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.111.

[14] Sotaro Katayarna, Tatsunori Taniai, and Kazutoshi Tanaka. Quasistatic contact-rich manipulation via linear complementarity quadratic programming. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 203–210. IEEE, 2022.

[15] Xiaoxi Jia, Christian Kanzow, Patrick Mehlitz, and Gerd Wachsmuth. An augmented lagrangian method for optimization problems with structured geometric constraints. *Mathematical Programming*, 199(1):1365–1415, 2023.

[16] Lei Guo and Zhibin Deng. A new augmented lagrangian method for mpccs—theoretical and numerical comparison with existing augmented lagrangian methods. *Mathematics of Operations Research*, 47(2):1229–1246, 2022.

[17] Alberto De Marchi, Xiaoxi Jia, Christian Kanzow, and Patrick Mehlitz. Constrained composite optimization and augmented lagrangian methods. *Mathematical Programming*, 201(1):863–896, 2023.

[18] Christian Kanzow and Patrick Mehlitz. Convergence properties of monotone and nonmonotone proximal gradient methods revisited. *Journal of Optimization Theory and Applications*, 195(2):624–646, 2022.

[19] Hyung Ju Terry Suh, Tao Pang, and Russ Tedrake. Bundled gradients through contact via randomized smoothing. *IEEE Robotics and Automation Letters*, 7(2):4000–4007, 2022.

[20] Simon Le Cleac'h, Taylor A Howell, Shuo Yang, Chi-Yen Lee, John Zhang, Arun Bishop, Mac Schwager, and Zachary Manchester. Fast contact-implicit model predictive control. *IEEE Transactions on Robotics*, 40: 1617–1629, 2024.

[21] Gijeong Kim, Dongyun Kang, Joon-Ha Kim, and Hae-Won Park. Contact-implicit differential dynamic programming for model predictive control with relaxed complementarity constraints. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11978–11985. IEEE, 2022.

[22] Gijeong Kim, Dongyun Kang, Joon-Ha Kim, Seungwoo Hong, and Hae-Won Park. Contact-implicit model predictive control: Controlling diverse quadruped motions without pre-planned contact modes or trajectories. *The International Journal of Robotics Research*, 44(3):486–510, 2025.

[23] Bernhard Paus Graesdal, Shao Yuan Chew Chia, Tobia Marcucci, Savva Morozov, Alexandre Amice, Pablo Parrilo, and Russ Tedrake. Towards Tight Convex Relaxations for Contact-Rich Manipulation. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.132.

[24] Aykut Özgun Önol, Radu Corcodel, Philip Long, and Taşkın Padır. Tuning-free contact-implicit trajectory optimization. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1183–1189. IEEE, 2020.

[25] Jonas Hall, Armin Nurkanović, Florian Messerer, and Moritz Diehl. LCQPow: a solver for linear complementarity quadratic programs. *Mathematical Programming Computation*, 17(1):81–109, 2025.

[26] Alp Aydinoglu and Michael Posa. Real-time multi-contact model predictive control via admm. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 3414–3421. IEEE, 2022.

[27] Alp Aydinoglu, Adam Wei, Wei-Cheng Huang, and Michael Posa. Consensus complementarity control for multi-contact mpc. *IEEE Transactions on Robotics*, 2024.

[28] Wei-Cheng Huang, Alp Aydinoglu, Wanxin Jin, and Michael Posa. Adaptive contact-implicit model predictive control with online residual learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5822–5828. IEEE, 2024.

[29] Sharanya Venkatesh, Bibit Bianchini, Alp Aydinoglu, William Yang, and Michael Posa. Approximating global contact-implicit mpc via sampling and local complementarity. *IEEE Robotics and Automation Letters*, 2025.

[30] Hien Bui, Yufeiyang Gao, Haoran Yang, Eric Cui, Siddhant Mody, Brian Acosta, Thomas Stephen Felix, Bibit Bianchini, and Michael Posa. Push anything: Single- and multi-object pushing from first sight with contact-implicit mpc. *arXiv preprint arXiv:2510.19974*, 2025.

[31] Etienne Ménager, Pierre Fabre, Antoine Bambade, Wilson Jallet, Alberto de Marchi, and Justin Carpentier. Frictional Contact-Implicit Inverse Dynamics. working paper or preprint, August 2025. URL https://hal.science/hal-05201780.

[32] Mihai Anitescu. Optimization-based simulation of nonsmooth rigid multibody dynamics. *Mathematical Programming*, 105(1):113–143, 2006.

[33] Marc Toussaint. A novel augmented lagrangian approach for inequalities and convergent any-time non-central up-

dates. *arXiv preprint arXiv:1412.4329*, 2014.

[34] Joel AE Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[35] Lorenz T Biegler and Victor M Zavala. Large-scale nonlinear programming using ipopt: An integrating framework for enterprise-wide dynamic optimization. *Computers & Chemical Engineering*, 33(3):575–582, 2009.

[36] HJ Terry Suh, Tao Pang, Tong Zhao, and Russ Tedrake. Dexterous contact-rich manipulation via the contact trust region. *The International Journal of Robotics Research*, page 02783649251398875, 2025.