Approximating Global Contact-Implicit MPC via Sampling and Local Complementarity

Sharanya Venkatesh*, Bibit Bianchini*, Alp Aydinoglu, William Yang, Michael Posa
*The first two authors contributed equally to this work.

***Abstract*—To achieve general-purpose dexterous manipulation, robots must rapidly devise and execute contact-rich behaviors. Existing model-based controllers are incapable of globally optimizing in real-time over the exponential number of possible contact sequences. Instead, recent progress in contact-implicit control has leveraged simpler models that, while still hybrid, make local approximations. However, the use of local models inherently limits the controller to only exploit nearby interactions, potentially requiring intervention to richly explore the space of possible contacts. We present a novel approach which leverages the strengths of local complementarity-based control in combination with low-dimensional, but global, sampling of possible end-effector locations. Our key insight is to consider a contact-free stage preceding a contact-rich stage at every control loop. Our algorithm, in parallel, samples end effector locations to which the contact-free stage can move the robot, then considers the cost predicted by contact-rich MPC local to each sampled location. The result is a globally-informed, contact-implicit controller capable of real-time dexterous manipulation. We demonstrate our controller on precise, non-prehensile manipulation of non-convex objects using a Franka Panda arm. Project page: https://approximating-global-ci-mpc.github.io**

***Index Terms*—Contact-rich manipulation, model predictive control, model-based, contact-implicit.**

I. Introduction

For multi-purpose robots to successfully deploy into the home and workplace, they will need to be capable of dexterous manipulation of complex objects. Progress towards this goal has been made on a number of fronts, including imitation learning from human demonstrations [1] and offline model-based planning [2], [3]. However, these approaches both require advanced knowledge of the task and can fail to generalize to even minor permutations, such as new goal configurations.

An alternative approach that has made recent strides is contact-implicit model predictive control (CI-MPC). Contact-implicit methods attempt to optimize for state and/or input trajectories and a corresponding sequence of contact modes. In the face of hybrid, nonlinear contact dynamics, CI-MPC approaches have to compromise to reach real-time rates, such as by picking the best plan out of many sampled trajectories [4], [5], or by using simplified dynamics. One such simplified model that still captures the hybrid aspect of contact-rich dynamics is a linear complementarity system (LCS) [6], a piecewise linear representation. However like many simplified models, LCS dynamics are limited in accuracy to a local neighborhood of the true underlying system (Figure 2).

Authors are with the General Robotics, Automation, Sensing, and Perception (GRASP) Laboratory at the University of Pennsylvania, Philadelphia, PA 19104 (emails: {sshastry, bibit, alpayd, yangwill, posa}@seas.upenn.edu).

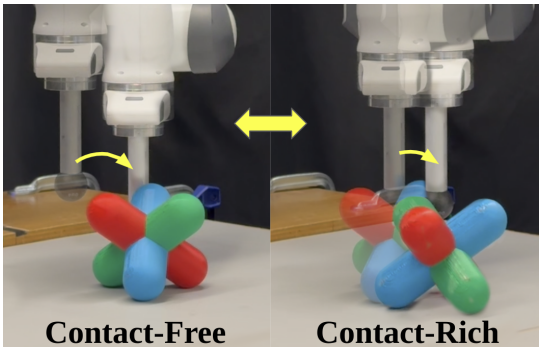

Fig. 1. Our real-time CI-MPC combines a global, explorative contact-free stage with a local contact-rich stage. At every control loop, our algorithm chooses contact-rich actions to make goal progress, or contact-free actions to pursue more amenable starting locations for future contact-rich actions.

We present a novel CI-MPC approach that inherits the local efficacy of complementarity-based control in combination with the global efficacy of sampling-based strategies, only requiring low-dimensional sampling of individual end effector locations. Our key insight is that all contact-rich control can be split into a contact-free stage in which the robot moves in a collision-free path (easier to compute than contact-rich trajectories), followed by a contact-rich stage in which the robot is able to make and break contact as it pleases (Figure 1). Existing CI-MPC methods utilizing local dynamics can be effective in the contact-rich stage, if the local neighborhood is amenable to progressing to the goal. Sampling is a natural solution: we sample end effector locations where the contact-free to contact-rich transition can occur. We quantify which location is the most advantageous to pursue by comparing their local CI-MPC optimization costs. In closed loop with cost-based and progress-based switching logic, our controller autonomously switches between contact-free and contact-rich stages, trading off future investment with immediate progress, respectively.

In this paper, we contribute:

- A novel combination of global sampling with local control for online multi-contact manipulation. This unique combination is capable of tasks that the local control alone essentially would always fail to accomplish.
- Hardware results applying our algorithm to achieve high-precision pose goals with non-convex objects with a Franka Panda robotic arm.
- A direct comparison in simulation to MuJoCo MPC (MJPC) [5], another non-local CI-MPC method our approach outperforms while being safer and more reliable.

II. Related Work

A. Offline Trajectory Optimization

While solving a global nonlinear control problem can be too time intensive to compute online, offline contact-implicit trajectory optimization can be effective in manipulation and locomotion [7], [8]. The non-smooth dynamics of contact-rich scenarios can lead to challenging loss landscapes for trajectory optimization. Mitigations for this include artificially smoothing gradients [3], [9], solving for contact mode sequences via graph search [2], sampling contact mode sequences [10] and/or control inputs [11], or restricting applications to simpler 2D problems [12]. While offline trajectory optimization is a means of discovering and optimizing complex trajectories for contact-rich motion, we seek strategies that can improvise in real time.

B. Contact-Implicit MPC

Alternative to offline methods, recent progress has been made by reformulating global tasks into more tractable online MPC problems. One approach uses local complementarity-based approximations to the global system [13], [14], encouraging consensus between local non-smooth rigid body contact constraints and local dynamics, while exploring contact modes via small-scale mixed integer optimization. Other approaches, in a distinct but similar fashion, use artificially smoothed contact dynamics during online trajectory optimization. This smooths the optimization landscape, allowing differentiable algorithms to explore different contact modes [15]–[18]. All of these methods, however, leverage local approximations and thus may fundamentally struggle to escape local minima and initiate distant, but beneficial, contact without intervention.

C. The Role of Sampling for Planning and Control

While the non-smooth dynamics of contact-rich scenarios pose serious challenges to techniques based in differentiable optimization, gradient-free methods like random sampling have shown promise. Many methods in the literature include sampling in the spaces of input trajectories [5], [19], contact mode sequences [10], [20], or both [11]. In our work, we explore a lower-dimensional form of sampling: sampling end effector configurations, using these as global seeds for the local contact-implicit controller from [14]. This does not require sampling full trajectories nor full system states, but retains the primary benefit of sampling as a way out of local minima.

III. Background

First, we review contact dynamics and its modeling choices in §III-A. We then introduce an optimal control problem that relies on contact dynamics as a constraint in §III-B. §III-C details C3 [14], an existing real-time, contact-implicit algorithm for a local approximation of the control problem.

A. Contact Dynamics

For a contact-rich system with state x and inputs u subject to contact forces λ, its dynamics can be generally written as

$$x_{k+1} = g\left(x_k, u_k, \lambda_k\right), \tag{1a}$$
$$0 \leq \lambda_k \perp \pi\left(x_k, u_k, \lambda_k\right) \geq 0, \tag{1b}$$

where the discrete-time dynamics g depend on the contact forces λ_k, which are the solution to a nonlinear complementarity problem (NCP) [21] in (1b). The NCP elegantly embeds the multi-modal nature of contact-rich systems.

In our context, the vector λ_k represents contact forces, and the complementarity constraint (1b) enforces the hybrid (or non-smooth) aspects of the dynamics; for example, it can encode the constraint that forces must only occur when in contact, and can enforce the stick-slip effects of Coulomb friction. For simulation, the solved contact forces are found to satisfy contact dynamics encoded by the NCP in (1b) and affect the system dynamics as in (1a).

A local approximation to (1) that still preserves the multi-modality of contact is a linear complementarity system (LCS) [6]. An LCS describes the state and contact force trajectories for an input sequence starting from x_0 such that

$$x_{k+1} = Ax_k + Bu_k + D\lambda_k + d, \tag{2a}$$
$$0 \leq \lambda_k \perp Ex_k + F\lambda_k + Hu_k + c \geq 0. \tag{2b}$$

where $x_k \in \mathbb{R}^{n_x}$, $\lambda_k \in \mathbb{R}^{n_\lambda}$, $u_k \in \mathbb{R}^{n_u}$, $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $D \in \mathbb{R}^{n_x \times n_\lambda}$, $d \in \mathbb{R}^{n_x}$, $E \in \mathbb{R}^{n_\lambda \times n_x}$, $F \in \mathbb{R}^{n_\lambda \times n_\lambda}$, $H \in \mathbb{R}^{n_\lambda \times n_u}$, and $c \in \mathbb{R}^{n_\lambda}$.

In computing the LCS representation of a given system, we consider a nominal $x_{\text{nom}}, u_{\text{nom}}$ and the resulting λ_{nom} from solving (1b). The LCS dynamics (2) approximate the nonlinear hybrid dynamics (1) by linearizing g and π.

Once the LCS is constructed, for a given x_k and u_k, the corresponding complementarity variable λ_k is found by solving (2b). Similarly, x_{k+1} can be computed using (2a) when x_k, u_k, and λ_k are known. Given an input u_k and state x_k, the next state is $x_{k+1} = \text{LCS}(x_k, u_k)$ defined by (2).

B. Optimal Control through Contact

We are interested in solving the nonlinear MPC problem first posed in [8] for contact-rich systems whose dynamics are well-described by an NCP,

$$\min_{x_k, u_k, \lambda_k} \sum_{k=0}^{N-1} \left(x_k^T Q_k x_k + u_k^T R_k u_k\right) + x_N^T Q_N x_N \tag{3a}$$
$$\text{s.t.} \quad x_{k+1} = g\left(x_k, u_k, \lambda_k\right), \tag{3b}$$
$$0 \leq \lambda_k \perp \pi\left(x_k, u_k, \lambda_k\right) \geq 0, \tag{3c}$$
$$x_0 = x_{\text{init}}, \tag{3d}$$
$$(x_k, u_k) \in \mathcal{C}, \tag{3e}$$
$$\text{for } k \in \{0, \ldots, N\}. \tag{3f}$$

This optimization problem solves for state $x_{0:N}$, control input $u_{0:N-1}$, and contact force $\lambda_{0:N-1}$ trajectories that satisfy the system's dynamics (3b), contact constraints (3c), initial condition (3d), and other constraints (3e), e.g. input limits, safety constraints, or goal conditions.

Solving this MPC problem for even simple multi-contact systems is intractable at real-time rates due to the complicated nature of the discontinuous contact dynamics (3c) and their impact on system dynamics (3b). An alternative is to approximate them with expressions that are faster to evaluate.

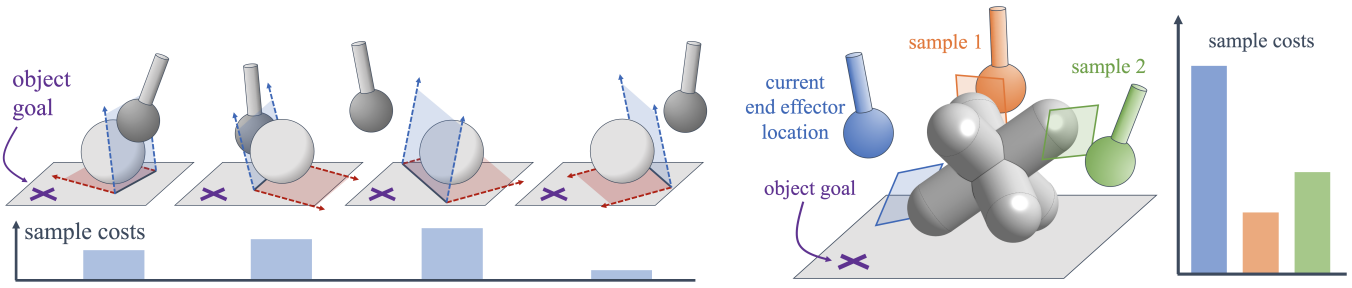

Fig. 2. Left: A spherical end effector approaches a spherical object on a flat table. Loosely speaking, the LCS approximates object geometry as a set of hyperplanes coincident with and tangent to their witness points with respect to other geometries of interest. The hyperplane for ground-object contact is in red, while the robot-object contact hyperplane is in blue. Each initial condition has its associated MPC cost. In this example, the rightmost sample's LCS approximation allows the robot to most effectively foresee progressing the object toward the goal and correspondingly has the lowest sample cost. Right: These LCS approximations are well-defined even for more complicated geometries, only requiring witness points between the object and other collision geometry.

C. Consensus Complementarity Control (C3)

To simplify (3), we pose the control problem with the dynamics expressed as an LCS (substituting (2) for (3b) and (3c)) using x_{init} (and some nominal u_0, λ_0) as the LCS linearization point. The optimization problem becomes

$$\min_{x_k,u_k,\lambda_k} \sum_{k=0}^{N-1} (x_k^T Q_k x_k + u_k^T R_k u_k) + x_N^T Q_N x_N \tag{4a}$$

$$\text{s.t.} \quad x_{k+1} = Ax_k + Bu_k + D\lambda_k + d, \tag{4b}$$

$$0 \le \lambda_k \perp Ex_k + F\lambda_k + Hu_k + c \ge 0, \tag{4c}$$

$$x_0 = x_{\text{init}}, \tag{4d}$$

$$(x_k, u_k) \in \mathcal{C}, \tag{4e}$$

$$\text{for } k \in \{0, \dots, N\}. \tag{4f}$$

This is still computationally intensive because of the complementarity constraint (4c). The method Consensus Complementarity Control (C3) [14] converges to this optimization problem's minimum. C3 uses the alternating direction of multipliers (ADMM) to iteratively solve the optimization problem while satisfying only the dynamics (4b) then only the contact constraints (4c). This first stage is a quadratic program (QP) with linear equality and inequality constraints. The second stage can crucially be parallelized across all time steps, since the contact constraints (4c) depend only on step k. Thus, the second stage can be formulated as a small-scale mixed-integer QP (MIQP), sufficiently fast when parallelized across time steps. With extra cost terms to encourage consensus between these two stages, C3 will converge to the solution to (4). In practice, using a suboptimal solution at a faster rate is more performant than a more optimal solution at a slower rate, so it is beneficial to terminate C3 after a few ADMM iterations.

IV. Methods

In this work, we extend the capabilities of C3 [14] by injecting global insights into the local problem. While using an LCS model enables C3's real-time performance, this has negative consequences that motivate our approach (§IV-A). The key is to split the problem into an initial contact-free mode and a subsequent contact-rich mode (§IV-B). The contact-free mode explores globally, setting up the contact-rich mode in a region tractably handled by C3. To approximate the new bilevel optimization problem in real-time, we sample low-dimensional end effector locations at the mode switch (§IV-C). In closed-loop, our controller uses progress and cost metrics to autonomously switch between modes (§IV-D).

A. Limitations of LCS as an MPC Modeling Choice

The LCS approximation of a system creates hyperplanes in (x, u, λ) space with respect to every contact pair of interest. While not completely accurate, one can conceptualize these hyperplanes as approximating the non-linear geometry of a scene via planes tangent to the object geometry at the witness points. Thus we can gain insights from cartoons like Figure 2 depicting $\mathbb{R}^3$ hyperplanes.[1] With these visuals in mind, it is evident the half-space boundaries can easily lock the robot into regions where it cannot apply positive normal force to move the object towards its goal (e.g. the third from the left example in Figure 2). The contact constraints in the C3 MPC problem (4) prevent planning negative normal forces, so the robot will plan to do nothing rather than make negative progress.

Thus, C3 is fundamentally local. Past C3 demonstrations [13], [14] required careful experimental design and/or additional heuristics to encourage or force the end effector out of regions where the local LCS view is antagonistically simplified. These heuristics depend on the system and the goal, and bake in prior knowledge for what regions better enable C3 to make task progress. We seek a simple way to instill the global knowledge C3 lacks into a controller that performs contact-rich manipulation at which C3 locally excels.

B. Separate Contact-Free and Contact-Rich Modes

First, consider that any optimal multi-contact control problem can be decomposed into an initial contact-free mode and a subsequent contact-rich mode. The contact-free mode has no collisions between the robot and manipulated objects, and the

[1] Precisely, the linearization of the gap function ϕ is a combination of the hyperplanes in the figure, with additional terms derived from the linearizations of the single-step dynamics.

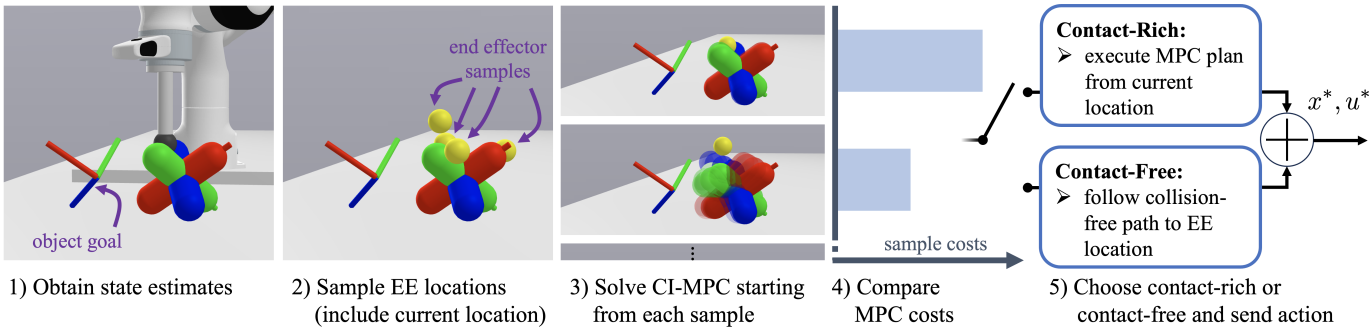

Fig. 3. The algorithm for one control loop of our sampling-based contact-implicit controller. The third step that solves a local contact-implicit MPC problem for each sample can be parallelized, since each plan is independent. In this example, the top sample's CI-MPC plan makes little progress to the object goal, and thus is associated with high cost. The second sample's CI-MPC plan shows more progress, and thus is lower cost.

contact-rich mode can make and break contact as it pleases. This suggests a new bilevel optimization problem[2] given by

$$\min_{K, x_{\text{switch}}} \left(\begin{array}{c} \min\limits_{x_k, u_k} \sum\limits_{k=0}^{K} c_{\text{free}}(x_k, u_k) \\ \text{s.t. } x_K = x_{\text{switch}} \end{array} + \begin{array}{c} \min\limits_{x_k, u_k} \sum\limits_{k=K}^{N} c_{\text{rich}}(x_k, u_k) \\ \text{s.t. } x_K = x_{\text{switch}} \end{array} \right), \tag{5}$$

whose outer problem selects a time step K and system state x_{switch} for *when and where* the mode switches from contact-free to contact-rich, and whose inner problem minimizes the contact-free and contact-rich costs (c_{free} and c_{rich}, respectively) given the intermediate state x_{switch}. This loses no generality from the original contact-rich control problem (3). To enable computational efficiency, we approximate in (5):

1) the contact-free mode as end effector repositioning (kinematic path planning);
2) the contact-rich mode via local CI-MPC (C3 [14]).

The first approximation can be justified for quasi-static systems, where the object remains stationary during contact-free motion. While no tasks are truly quasi-static, this is a common assumption in manipulation [22]–[24]. As we will see in §V, even for more dynamic tasks, this is not particularly limiting. This means the effective output of the first stage is a new position for the end effector, making the search space of the outer optimization problem low dimensional. Specifically, if a full system state x contains $[q_{\text{ee}}, q_{\text{obj}}, v]$, we impose $x_{\text{switch}} = [q_{\text{ee,switch}}, q_{\text{obj,init}}, v_{\text{init}}]$ where only $q_{\text{ee,switch}}$ differs from x_{init}. Additionally, the cost of the contact-free stage can be penalized by a simple end effector travel cost.

The second simplification is based on the efficacy of C3 for the more challenging portion of the problem, with LCS dynamics local to the starting state with x_{switch}. By making global decisions via contact-free path planning and sequencing with contact-rich motions, we are able to mitigate the limitations inherent in the locality of C3's contact-implicit control.

C. Sample Switching States

Even with simple travel cost and C3 approximations to the contact-free and contact-rich modes, respectively, the bilevel optimization problem is intractable to solve in real time because it would require solving (expensive) C3 for every candidate x_{switch}. Instead, we select a small set of x_{switch} samples, which crucially can be evaluated in parallel. This sample set always includes the current system state, i.e. our hierarchical controller always knows the contact-rich plan and can enter or remain in the contact-rich mode whenever it wants. At every control loop, our controller computes the C3 costs for the x_{switch} candidates, adding the contact-free travel costs appropriately. To sample x_{switch} candidates, we assume access to a sampling strategy $\texttt{SampleEE}(x_{\text{init}}, x_{\text{switch,prev}}, n_s)$ which draws n_s samples for potential end effector locations. The first sample is always $q_{\text{ee,init}}$, and the second is $q_{\text{ee,switch,prev}}$, if there is one (precisely, the controller's last loop was in contact-free mode and was pursuing $x_{\text{switch,prev}}$). To evaluate costs, we assume access to the C3 algorithm and the function $\texttt{C3Cost}(x_{\text{sample}}, x_{\text{goal}})$, which returns the C3 cost from (4). This sampling and evaluation step is detailed by Algorithm 1.

Algorithm 1 Sample and Evaluate

Require: $x_{\text{init}}, x_{\text{goal}}, x_{\text{switch,prev}}$ if there was one, number of samples n_s, sampling strategy `SampleEE`, travel cost weight w_{travel}, C3 algorithm and associated cost `C3Cost`

$q_{\text{sample,ee},1:n_s} \leftarrow \texttt{SampleEE}(x_{\text{init}}, x_{\text{switch,prev}}, n_s)$

Parallelizing samples, solve C3 and impose travel cost.

for $i \in \{1, \ldots, n_s\}$ **do**
  $c_{\text{travel}} \leftarrow w_{\text{travel}} \left\| q_{\text{ee,init}} - q_{\text{ee,sample},i} \right\|$
  $x_{\text{sample},i} \leftarrow [q_{\text{ee,sample},i}, q_{\text{obj,init}}, v_{\text{init}}]$
  $c_{\text{sample},i} \leftarrow \texttt{C3Cost}(x_{\text{sample},i}, x_{\text{goal}}) + c_{\text{travel}}$
end for
return $(x_{\text{sample}}, c_{\text{sample}})_{1:n_s}$

D. Switch Modes Based on Costs and Progress

Given a set of x_{switch} candidates and associated inner costs c_{sample}, we must pursue one. If we pick the current state, our controller executes the actions computed during the C3 solve. If we pick a different state, our controller commands a collision-free path to the new end effector location. A natural selection algorithm might be to choose the sample with the lowest associated cost. However, we find this results in indecisive behavior. We attribute this primarily to C3

[2]Ignoring other constraints for brevity, including dynamics, contact dynamics, initial condition, input limits, etc.

variability, as C3 is a local algorithm with no convergence guarantees. Specifically, we notice two challenges when tuning the controller:

1) Inefficiency: The controller switches from contact-rich to contact-free mode, even when the robot is already poised to manipulate the object towards the goal.
2) Ineffectiveness: The controller remains in the contact-rich mode, even though the robot is not poised to manipulate the object towards the goal.

First, we impose hysteresis on cost comparisons to make switching between modes (and between x_{switch} targets within the contact-free mode) more decisive, defining hysteresis values $h_{\text{free-to-rich}}, h_{\text{rich-to-free}}, h_{\text{free-to-free}}$ (where $h_{\text{free-to-free}}$ encourages decisiveness for where to relocate within the contact-free mode). In particular, to avoid the inefficiency issue, we choose $h_{\text{rich-to-free}}$ to be large. However, this alone risks exacerbating ineffectiveness. To address this, we force a transition to the contact-free mode if the manipulated object fails, over a specified period of time, to make progress toward the goal. After the contact-rich mode makes no progress for more than this time threshold, the controller pursues the sampled x_{switch} with the lowest cost.

To increase the likelihood of pursuing a high-quality x_{switch} sample after leaving contact-rich mode, we maintain a buffer of relevant sampled end effector positions and their associated costs. The sample buffer is pruned of samples whose associated object locations at the time of computing cost are too far from the current object location. At every control loop, we:

1) Update x_{init} from sensing.
2) Generate and evaluate samples via Algorithm 1.
3) Maintain the sample buffer: remove outdated samples based on object movement and add new ones.
4) Consider mode switching based on cost-based (with hysteresis $h_{\text{rich-to-free}}$, $h_{\text{free-to-rich}}$) and progress-based logic. If within contact-free mode, consider switching pursued end effector locations (with hysteresis $h_{\text{free-to-free}}$). If transitioning from contact-rich to contact-free, choose the lowest cost sample in the buffer.
5) Execute a plan based on the current mode. If in contact-rich mode, execute the plan from solving C3 with x_{init}. If in contact-free mode, follow a collision-free path from x_{init} to the pursued sample.

We repeat this in receding horizon fashion, which enables the control to adapt and adjust to observed system dynamics as well as to consider more samples with every control loop. One control loop of our approach is depicted in Figure 3.

V. Experiments

A. Task, State Representation, and Contact Modeling

To validate our controller, we test on multiple examples with a Franka Panda arm equipped with a spherical end effector to manipulate an object to a pose goal. When the object goal is reached within position and rotation tolerances, a new pose goal is randomly generated, demonstrating generalization over initial and goal poses. We test on two hardware examples (3D jack and planar T, both shown in Figure 4) and additionally show 3D jack results in simulation for more direct comparison

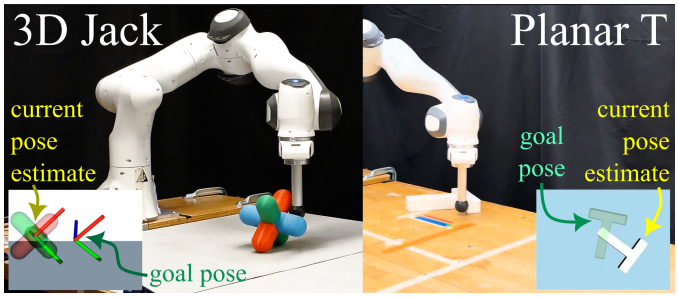

Fig. 4. We demonstrate our controller on two manipulation examples: 3D pose goals with a jack (left), and 2D planar pose goals with a T (right). The inset renderings in the bottom corners depict a view of the object's goal pose relative to the current pose estimate.

to a baseline (more detail in §V-E). Our controller operates on LCS dynamics where $x \in \mathbb{R}^{19}$ (representing end effector position, object position, object orientation, and their velocities), $u \in \mathbb{R}^3$ (representing cartesian forces applied to the end effector), and $\lambda \in \mathbb{R}^{16}$ (representing 4 contact pairs with a 4-sided polyhedral friction cone approximation [25]).

Our approach requires manual enumeration of contact *geometries*, but uses collision detection between them to identify possible contact *points* at every control loop. These points automatically change with motion, and the approach can accommodate any geometries handled by the collision detection algorithm. For both examples, one contact pair uses the closest witness point on the object to the end effector, and the remaining 3 contact pairs are for object-ground contacts. For the jack, we obtain the closest point per capsule to the ground. A visualization of all the jack contact points for a particular configuration is illustrated in Figure 5. For the T, we define small spheres at the 3 distal ends as witnesses to the ground. For this planar example, we use the full 3D states of the end effector and T but add a constraint in the contact-rich stage (C3) to enforce the end effector maintains a pre-defined height.

B. Implementation

Our controller is implemented in C++ within the Drake systems framework [26]. We take the approach in [13] and connect our controller to an operational space controller (OSC) [27], which tracks task-space commands specified by our policy at 8-12 Hz, via joint-level control at 1 KHz. We additionally integrate with Franka hardware and state estimators as in the control diagram in Figure 6. Object state estimation uses FoundationPose [28] running at 30Hz with a D455 RealSense RGBD camera. Our setup uses two computers: a computer with a 13th generation Intel Core i9-13900KF with 32 threads (for our sampling-based CI-MPC) and an NVIDIA GeForce RTX 4090 GPU (for FoundationPose), and an Intel i7-8700K processor (for our OSC and robot drivers) equipped with a real-time kernel for communicating with the Franka. Inter-computer communication occurs over LCM [29].

C. Sampling Parameters

We consider three samples (including the end effector's current location) in parallel with every control loop. We solve C3's QP with OSQP [30] and MIQP with Gurobi [31]. Our

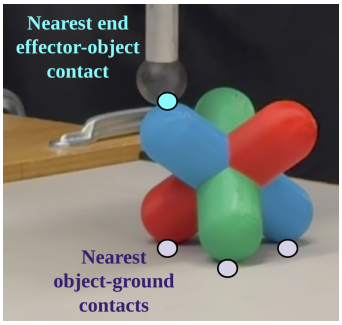

Fig. 5. Illustration of contact points considered for the jack object.

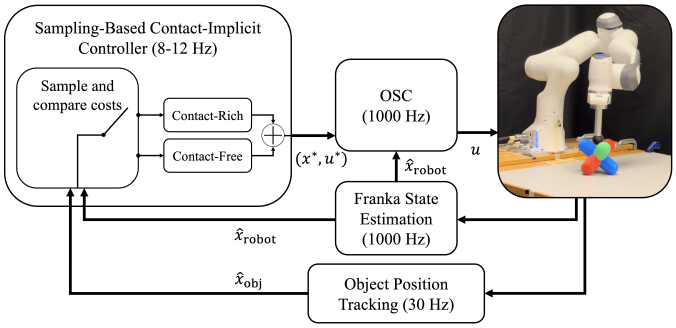

Fig. 6. A block diagram depicts our hierarchical controller (top left block), which performs real-time CI-MPC globally by sampling potential end effector locations for switching from contact-free to contact-rich mode.

samples and the C3 MIQP both utilize parallelization, so we use the maximum number of threads in our hardware setup. Because of this, additional samples slow down the control rate and decrease performance. With this setup, our controller runs at 8-12 Hz for all experiments. We randomly sample end effector locations on a sphere around the jack and on an enlarged planar perimeter for the T. While more sophisticated sampling is compatible with our approach, these simple distributions are notably effective.

D. Cost Parameters

To avoid overly aggressive behavior from high costs, we truncate the final goal's displacement from the current object location (15cm in position and 2 radians in orientation). This bounds the errors encountered by the optimization problem and mitigates indecision when orientation error approaches π radians, at which any rotation direction is equally effective. This orientation truncation requires hysteresis on the rotation axis to ensure stability of the direction of rotation.

For the Q cost matrix, a typical diagonal structure is sufficient for position and velocity errors in our experiments. However, orientation presents some challenges. The true orientation error we desire to minimize is θ^2_{error}, where θ_{error} is the scalar angle of the relative rotation between the current and goal orientations. From quaternions, this is calculated as

$$\theta_{\text{error}} = \left(\arctan\left(\frac{\left\| q^2_{\text{rel},x} + q^2_{\text{rel},y} + q^2_{\text{rel},z} \right\|}{q_{\text{rel},w}} \right) \right)^2, \quad \text{(6a)}$$

$$\text{where} \quad q_{\text{rel}} = q^{-1}_{\text{quat,curr}} \otimes q_{\text{quat,goal}}, \quad \text{(6b)}$$

for $\otimes$ as quaternion product. The $\arctan$ indicates a problematic region where its argument is zero – this occurs precisely

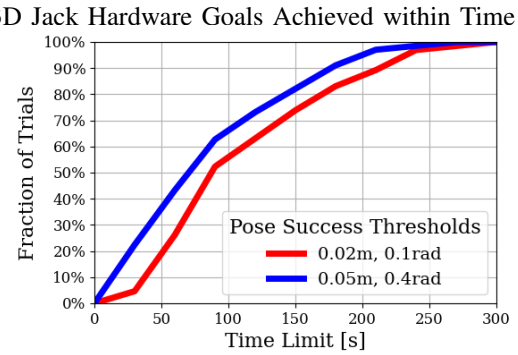

Fig. 7. Cumulative distribution for time to goal, using sets of tight and loose position and orientation tolerances.

when $q_{\text{quat,curr}} = q_{\text{quat,goal}}$. The landscape is not strictly convex at this point and is locally non-convex. Thus, the naive 2-norm error between the elements of $q_{\text{quat,curr}}$ and $q_{\text{quat,goal}}$,

$$\tilde{\theta}^2_{\text{error}} = \left\| q_{\text{quat,curr}} - q_{\text{quat,goal}} \right\|^2. \quad \text{(7)}$$

poorly captures the true θ_{error} when it is small. To address this, we set the 4x4 object quaternion portion of Q (throughout the entire MPC horizon) to be the Hessian of θ^2_{error} with respect to the elements of the current quaternion, about the $q_{\text{quat,curr}}, q_{\text{quat,goal}}$ operating point. We regularize this Hessian, adding $|\gamma| \cdot \mathbb{I}_{4\times4}$, where γ is its most negative eigenvalue to ensure positive-semi definiteness. Implementing this portion of Q is a critical step to effectively and reliably track orientation.

E. Comparisons

Due to the inability to escape geometric local minima, C3 fails essentially 100% of the time on our tasks and thus is not compared. We compare with MuJoCo MPC (MJPC) with predictive sampling [5] on the 3D jack task in simulation. As with our controller, we use MJPC as an online planner operating on a reduced model, abstracting the end effector as controlled in xyz only, whose motion is tracked via our joint-level OSC simulated in Drake. Unlike our controller, the MJPC planner models the nonlinear dynamics of the floating end effector moving in 3D with the jack and environment. Any comparison is sensitive to tuning, and we put forth a best-faith effort to tune MJPC. We found tighter control input limits to be helpful in preventing wild motions, selected a noise parameter of 0.295 to balance sample exploration and previous plan exploitation, and tuned state and input costs. MJPC performed best with control input splines of 5 knot points over a predictive horizon of 0.8 seconds, reasonably longer than our policy's 0.25 seconds, since we effectively get longer-term insight via our contact-free sampling.

VI. Results

We refer readers to our supplementary video for extended results. We evaluate performance under two sets of success thresholds: the stricter set requires under 2cm and 0.1 radians (5.7 degrees) of error, while the coarse set requires under 5cm and 0.4 radians (22.9 degrees) of error. Related works (e.g. [4], [32]) often utilize this coarse threshold. For all experiments, we execute the controller until the tight threshold is realized,

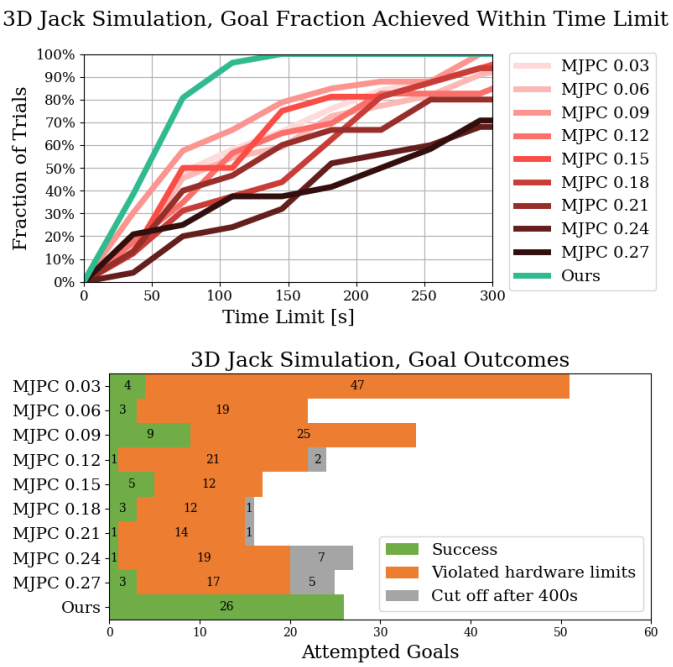

Fig. 8. Our approach outperforms MJPC in 3D jack manipulation in simulation with high precision tolerances (2cm and 0.1 radians pose error). Our controller achieves goals faster than MJPC (top) while also avoiding hardware limit violations (bottom) such as joint velocity, joint torque, and workspace limits. The MJPC lines are annotated with the values of the end effector velocity cost weight, showing decreased performance but ineffective ability to reduce hardware violations at higher end effector velocity costs.

Mean $\pm\sigma$ [Min, Max]	Time to Goal (s) within Pose Tolerances Tight: 2cm, 0.1rad	Loose: 5cm, 0.4rad
HW 3D Jack (ours) 67 trials	109.20 ± 64.24 $[17.40, 292.07]$	84.86 ± 60.54 $[5.20, 257.74]$
Sim 3D Jack (ours) 26 trials	49.31 ± 30.35 $[9.71, 124.70]$	33.84 ± 26.39 $[7.97, 97.82]$
Sim 3D Jack (MJPC) 34 trials	107.91 ± 112.38, $[3.30, 567.69]$	68.00 ± 83.50 $[1.51, 343.79]$
HW Planar Push-T (ours) 106 trials	30.45 ± 13.11 $[7.50, 79.43]$	17.43 ± 7.59 $[3.86, 42.00]$

TABLE I
PERFORMANCE METRICS OF HARDWARE (HW) AND SIMULATION (SIM) EXPERIMENTS UNDER TIGHT AND LOOSE TOLERANCES. MJPC RESULTS USE HIGHEST-PERFORMING END EFFECTOR VELOCITY COST OF 0.09.

then switch to the next goal pose. In post-processing, we back-compute the time-to-goal under the coarse success thresholds on the same experiments, presented in Table I.

A. 3D Jack

Figure 7 combines the hardware results from four continuous experiments of 21, 16, 15, and 15 successfully achieved random pose goals. All four experiments terminated due to the robot hitting workspace safety limits. Figure 7 shows the cumulative distribution for time to goal, demonstrating that while most goals are reached relatively quickly, there is a long tail with some challenging targets requiring more time.

Figure 9 depicts two single-goal trials achieved on hardware, annotated with contact-free and contact-rich modes. Both trials demonstrate cost-based and progress-based mode transitions.

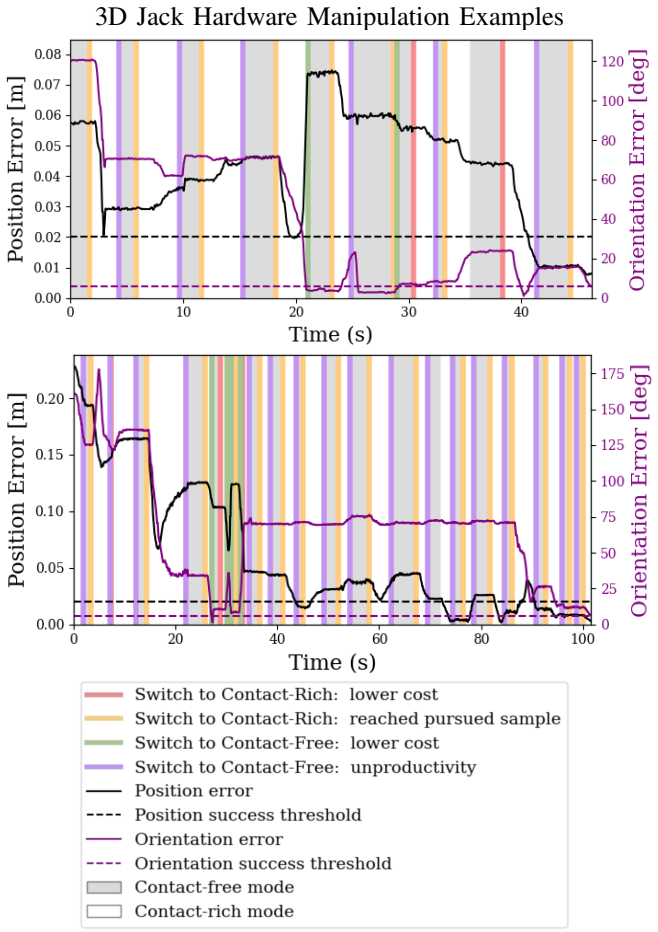

Fig. 9. Jack position and orientation errors over time, with contact-free (grey) and contact-rich (white) modes shaded and mode switching reasons labeled.

Most often, the errors do not change during the contact-free mode, since the object usually stays fixed while the robot relocates. These examples show how the controller balances position and rotation progress, occasionally sacrificing one to make progress on the other. A common cause of significant time spent per trial is simultaneous low position error and higher rotation error. However, the only terminal failure condition is when the object gets pushed to the boundary of the safe workspace, causing the robot to cross our workspace safety limits. In all other trials, our controller's persistence eventually brings the object to the goal in every test we performed.

1) Sim-to-Real Gap and Comparison to MJPC: Our controller, unsurprisingly, achieves pose goals faster in simulation than on hardware (see 49.31s simulation average compared to 109.20s hardware average in Table I). This gap, common in the literature, can be partly explained by state estimate errors, incorrect models, and FoundationPose's added computational load. We reiterate our approach is not optimal; it is a real-time CI-MPC approach that makes all its decisions on-the-fly. While not optimal, our controller is demonstrably effective on difficult tasks and outperforms MJPC [5] in simulation on our 3D jack example (Table I, Figure 8). Further, our controller satisfies Franka hardware limits (joint velocity/torque, workspace limits), while MJPC nearly always violates at least one. Given this high hardware limit violation (HWV) rate, we

did not feel safe deploying MJPC on the real robot. While impossible to compare against all possible MJPC weights, we investigate the role of end effector velocity cost weight, due to its role in the controller's speed. Increasing this cost weight makes MJPC less performant, yet does not prevent HWV.

B. Planar T Pushing

The statistics in Table I for the hardware planar T experiments combine four continuous experiments of 56, 20, 20, and 10 successfully achieved random planar pose goals in a row. With an average of 30.45s to achieve high-precision pose goals, our approach achieves a state-of-the-art time-to-goal competitive with other works on this example. Notably, while some prior works with this example use data-driven approaches (e.g. pre-training with Diffusion Policy [1]) or require offline computation (e.g. offline trajectory optimization [33]), our approach demonstrates only the object model is required, and generalization to different goal poses is a natural byproduct absent from these prior works.

VII. Limitations

While we contribute a solution to the global, 3D manipulation problem, our controller still took $\sim$1.8 minutes on average over 67 trials to achieve precise SE(3) goals. We reiterate the generality of our approach and acknowledge these tasks are challenging. However, we identify addressing inefficiencies as future work. Like all model-based methods, our controller requires dynamics models of the robot, objects it manipulates, and environment, preventing use in truly novel scenarios. Our provided demonstrations used a single spherical-shaped end effector, which we reasonably modeled as a single robot-object contact pair. If applied to a more dexterous manipulator such as a multi-fingered hand, both the contact pairs and state size would increase. The difficulty of the MPC problem scales with the number of contact pairs and the state vector size, slowing down control rates if either/both increase.

VIII. Conclusion

Our model-based controller performs generalizable, precise pose-driven tasks through multi-contact dynamics. By splitting the control problem into contact-free and contact-rich stages, we reap the benefits of global exploration when we sample new end effector locations plus local efficacy when existing CI-MPC methods take over after arriving at a desired location. In closed loop, the result is a persistent, globally-aware controller that can robustly reach precise pose goals without intervention.

Acknowledgments

This work was supported by a National Defense Science and Engineering Graduate (NDSEG) Fellowship, an NSF CAREER Award (Grant FRR-2238480), and the RAI Institute.

References

[1] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *The International Journal of Robotics Research*, p. 02783649241273668, 2023.

[2] X. Cheng, S. Patil, Z. Temel, O. Kroemer, and M. T. Mason, "Enhancing dexterity in robotic manipulation via hierarchical contact exploration," *arXiv preprint arXiv:2307.00383*, 2023.

[3] T. Pang, H. T. Suh, L. Yang, and R. Tedrake, "Global planning for contact-rich manipulation via local smoothing of quasi-dynamic contact models," *IEEE Transactions on Robotics*, 2023.

[4] A. H. Li, P. Culbertson, V. Kurtz, and A. D. Ames, "Drop: Dexterous reorientation via online planning," *arXiv preprint arXiv:2409.14562*, 2024.

[5] T. Howell, N. Gileadi, S. Tunyasuvunakool, K. Zakka, T. Erez, and Y. Tassa, "Predictive sampling: Real-time behaviour synthesis with mujoco," *arXiv preprint arXiv:2212.00541*, 2022.

[6] W. Heemels, J. M. Schumacher, and S. Weiland, "Linear complementarity systems," *SIAM journal on applied mathematics*, vol. 60, no. 4, pp. 1234–1269, 2000.

[7] A. Patel, S. L. Shield, S. Kazi, A. M. Johnson, and L. T. Biegler, "Contact-implicit trajectory optimization using orthogonal collocation," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2242–2249, 2019.

[8] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 69–81, 2014.

[9] H. J. T. Suh, T. Pang, and R. Tedrake, "Bundled gradients through contact via randomized smoothing," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4000–4007, 2022.

[10] H. Zhu, A. Meduri, and L. Righetti, "Efficient object manipulation planning with monte carlo tree search," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 10 628–10 635.

[11] L. Liu, K. Yin, M. Van de Panne, T. Shao, and W. Xu, "Sampling-based contact-rich motion control," in *ACM SIGGRAPH 2010 papers*, 2010, pp. 1–10.

[12] B. Aceituno and A. Rodriguez, "A hierarchical framework for long horizon planning of object-contact trajectories," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 189–196.

[13] W. Yang and M. Posa, "Dynamic on-palm manipulation via controlled sliding," in *Robotics: Science and Systems (RSS)*, Jul. 2024. [Online]. Available: https://roboticsconference.org/program/papers/12/

[14] A. Aydinoglu, A. Wei, W.-C. Huang, and M. Posa, "Consensus complementarity control for multi-contact mpc," *IEEE Transactions on Robotics*, 2024.

[15] G. Kim, D. Kang, J.-H. Kim, S. Hong, and H.-W. Park, "Contact-implicit mpc: Controlling diverse quadruped motions without pre-planned contact modes or trajectories," *arXiv preprint arXiv:2312.08961*, 2023.

[16] S. L. Cleac'h, T. Howell, M. Schwager, and Z. Manchester, "Fast contact-implicit model-predictive control," *arXiv preprint arXiv:2107.05616*, 2021.

[17] V. Kurtz, A. Castro, A. Ö. Önol, and H. Lin, "Inverse dynamics trajectory optimization for contact-implicit model predictive control," *arXiv preprint arXiv:2309.01813*, 2023.

[18] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 4906–4913.

[19] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Aggressive driving with model predictive path integral control," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1433–1440.

[20] H. Xue, C. Pan, Z. Yi, G. Qu, and G. Shi, "Full-order sampling-based mpc for torque-level locomotion control via diffusion-style annealing," 2024. [Online]. Available: https://arxiv.org/abs/2409.15610

[21] D. E. Stewart, *Dynamics with Inequalities: impacts and hard constraints*. SIAM, 2011.

[22] M. Halm and M. Posa, "Set-valued rigid-body dynamics for simultaneous, inelastic, frictional impacts," *The International Journal of Robotics Research*, p. 02783649241236860, 2024.

[23] M. R. Dogar and S. S. Srinivasa, "A planning framework for non-prehensile manipulation under clutter and uncertainty," *Autonomous Robots*, vol. 33, pp. 217–236, 2012.

[24] K. M. Lynch and M. T. Mason, "Stable pushing: Mechanics, controllability, and planning," *The international journal of robotics research*, vol. 15, no. 6, pp. 533–556, 1996.
[25] M. Anitescu and F. A. Potra, "Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems," *Nonlinear Dynamics*, vol. 14, no. 3, pp. 231–247, 1997.
[26] R. Tedrake *et al.*, "Drake: Model-based design and verification for robotics," 2019.
[27] P. M. Wensing and D. E. Orin, "Generation of dynamic humanoid behaviors through task-space control with conic optimization," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 3103–3109.
[28] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 868–17 879.
[29] A. S. Huang, E. Olson, and D. C. Moore, "Lcm: Lightweight communications and marshalling," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 4057–4062.
[30] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "Osqp: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020.
[31] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2024. [Online]. Available: https://www.gurobi.com
[32] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal, "Visual dexterity: In-hand reorientation of novel and complex object shapes," *Science Robotics*, vol. 8, no. 84, p. eadc9244, 2023.
[33] S. Kang, G. Liu, and H. Yang, "Global contact-rich planning with sparsity-rich semidefinite relaxations," *arXiv preprint arXiv:2502.02829*, 2025.