Abstract — The hybrid nature of multi-contact robotic systems, due to making and breaking contact with the environment, creates significant challenges for high-quality control. Existing model-based methods typically rely on either good prior knowledge of the multi-contact model or require significant offline model tuning effort, thus resulting in low adaptability and robustness. In this paper, we propose a real-time adaptive multi-contact model predictive control framework, which enables online adaption of the hybrid multi-contact model and continuous improvement of the control performance for contact-rich tasks. This framework includes an adaption module, which continuously learns a residual of the hybrid model to minimize the gap between the prior model and reality, and a real-time multi-contact MPC controller. We demonstrated the effectiveness of the framework in synthetic examples, and applied it on hardware to solve contact-rich manipulation tasks, where a robot uses its end-effector to roll different unknown objects on a table to track given paths. The hardware experiments show that with a rough prior model, the multi-contact MPC controller adapts itself on-the-fly with an adaption rate around 20 Hz and successfully manipulates previously unknown objects with non-smooth surface geometries. Accompanying media can be found at: https://sites.google.com/view/adaptive-contact-implicit-mpc/home

I. INTRODUCTION

The primary hurdles in achieving high-performance real-time control for multi-contact robotic tasks, such as dexterous manipulation, can be attributed to two key aspects. First, sequencing and positioning of effective contact locations, corresponding to combinatoric number of discrete contact choices and continuous inputs of robot actuation, typically make the problem computationally intractable. Second, variations of extrinsic (e.g., geometry and friction of objects) and intrinsic characteristics (e.g., robot physical parameters) makes it extremely challenging to acquire high-accuracy system models. In model-based multi-contact control [1], [2], the performance and speed of a controller are ultimately limited by the quality and complexity of the obtained model. Thus, current research efforts are predominantly centered on these two challenges.

Significant progress has been made recently in developing fast multi-contact model predictive control (MPC) [2], [3]. Those methods typically require an accurate multi-contact models, leading to large amount of model tuning in practice. The most recent works [4], [5] start using the collected data to learn a model [6]–[9] to improve the control performance for contact-rich tasks. Despite the success, the offline model learning in those methods requires a significant effort of data collection and training to obtain an effective model, limiting the flexibility of these methods when deployed in practice. To address those limitations, in this work we take a perspective from classic adaptive control paradigm [10] but try to extend classic adaptive control into multi-contact systems: we aim to develop an adaptive multi-contact MPC controller such that it can adjust multi-contact hybrid model in real time to account for variations of unknown objects/environments and improve its control performance for multi-contact tasks.

In this work, we present a novel adaptive contact-implicit MPC framework. It enables continuously adjusting the hybrid model from real-time in-stream data meanwhile the MPC controller uses the most recent model for high-quality control on multi-contact tasks. Compared to the existing work, we emphasize the two key contributions below.

(i) We present an adaptation law for multi-contact systems that continuously minimizes the gap between the prior model and reality by learning a hybrid residual model from in-stream data. It is inspired by our offline learning method [6] but unlike the previous work it focuses on online learning and utilizes physics-based prior models to speed up the process. Furthermore, the hybrid model adaption module is integrated into our fast multi-contact MPC controller [11] which allows real-time adaptive contact-implicit control on robots with on-the-fly performance improvement for contact-rich tasks.

(ii) We validate the proposed approach with two different hardware experiments, accomplishing challenging robot arm manipulation tasks. Our method adapts itself on-the-fly with an adaption rate around 20 Hz and can consistently be employed to manipulate objects with uneven or irregular surface

Fig. 1. Given an initial guess that the object is a rigid sphere, the controller adapts its model of the governing contact dynamics to roll and push real fruits (orange, lime) with a Franka Emika Panda arm, tracking a desired motion.
geometries. The findings also indicate that our approach can achieve success in tasks that purely model-based methods fail.

II. RELATED WORK

1) Learning Multi-Contact Dynamics Models: The hybrid nature of multi-contact dynamics poses challenges for gradient-based learning. One line of work, termed differentiable simulation, focuses on smoothing hybrid mode boundaries, although this can lead to approximation errors [12]–[14]. Recently, however, results have shown that conventional neural networks are limited in capturing the multi-modality and high-stiffness of multi-contact systems [7], [15], [16]. To address this, researchers [6], [7], [16] explicitly exploit the hybrid structure (complementarity formulation) of contact dynamics to develop learning algorithms, achieving state-of-the-art performance. In our adaptive contact-implicit MPC framework, the adaption module focuses on learning a hybrid residual model on top of prior dynamics, which is based on our previous work [6]. The benefits of [6] is that it simultaneously performs the mode partitioning and linear regression by proposing a novel training loss. The quadratic formulation in the loss enables fast updates using in-stream data, which can run up to 20 Hz.

2) Fast Multi-Contact Model Predictive Control: To handle the combinatoric complexity of choosing hybrid contact modes, previous work [17], [18] use pre-defined sequence of modes to achieve real-time multi-contact control on legged locomotion [19] and manipulation [20]. To achieve contact-implicit control, the work [3] proposed to relax contact mode boundaries with smoothing approximation. Concurrently, our work [11] introduced an approach that preserves the hybrid structures while decoupling the combinatorial complexity from planning depth. However, rather than operating with an accessible ground-truth multi-contact model, we incorporate a hybrid model adaptation module into the MPC controller. This module operates continuously, updating the hybrid model with data collected from real-world interactions. Consequently, the MPC controller can consistently improve its performance for multi-contact tasks.

3) Task-Oriented Multi-Contact Modeling Learning: A relevant body of recent research is task-driven multi-contact model learning, which aims to find a model that can be used to (derive a policy and) accomplish given tasks. This line of work shares the same idea as deep model-based reinforcement learning [21], although the latter typically suffers from huge data demand. The most recent work focuses on learning a task-driven computationally affordable models, which have been shown requiring a small amount of data to successfully solve deleterious manipulation [4] and bipedal locomotion [5]. It is important to note that the above existing methods predominantly rely on an offline model learning. This means that the model undergoes episodic updates after accumulating a buffer of historical policy data. In contrast, the approach presented in this paper emphasizes continuous and online updates to the hybrid residual model. This innovative method facilitates real-time improvements in the multi-contact hybrid model, consequently improving performance during the deployment of the MPC policy. Additionally, this adaptation framework holds the potential to reduce data consumption when compared to offline model learning methodologies.

4) Adaptive MPC: The most relevant theme of this work is adaptive control, which focuses on control of uncertain systems through real-time model adaptation and learning [22] and has long been an ongoing research direction [23]. Researchers have developed robust (adaptive) MPC methods [24]–[27], which are successfully applied to electric vehicles [28], climate control [29], quadrotors [30]. Similarly, there are multiple recent methods that perform adaptive MPC under unknown noise distributions [31] or incorporate components from adaptive control into learning-based MPC [32]. Also, the recent work [33] presents an adaptive MPC variant that automatically estimates control and model parameters by leveraging ideas from Bayesian optimization performing manipulation tasks. Unlike our approach which focuses on real-time adaptive MPC for systems that make and break contact, the authors focus on reaching tasks avoiding any contact interaction.

III. BACKGROUND

A. Multi-Contact Dynamics

The multi-contact dynamics of a manipulator is

\[ M(q)\dot{v} + C(q, v) = Bu + J(q)^T\lambda, \]

where \( q \) is the generalized positions vector, \( v \) is the generalized velocities, \( \lambda \) represents the contact forces. The contact forces can be represented by the following complementarity equation:

\[ 0 \leq \lambda \perp \psi(q, v, u, \lambda) \geq 0. \]

In this context, \( \psi \) signifies the connection between the contact forces and the generalized positions, velocities, and inputs, as discussed in references [34]–[36]. We denote a multi-contact model with \( \mathcal{M}_c \), with \( \mathcal{M}_c(q, v, \dot{v}, u, \lambda) : [0, 2) \) holds and with a slight abuse of notation denote the dynamics \( \dot{x} = \mathcal{M}_c(x, \lambda, u) \) with \( x = [q^T, v^T]^T \).

B. Linear Complementarity Problem

In this study, we employ linear complementarity problems (LCP’s) as a means to depict contact forces [35]–[37].

Definition 1: Given a vector \( q \in \mathbb{R}^m \), and a matrix \( F \in \mathbb{R}^{m \times m} \), the LCP \( (q, F) \) describes the following program:

\[
\begin{align*}
\text{find} & \quad \lambda \in \mathbb{R}^m \\
\text{subject to} & \quad y = F\lambda + q, \quad 0 \leq \lambda \perp y \geq 0.
\end{align*}
\]

C. Linear Complementarity Systems

We employ linear complementarity systems (LCS) as localized models for multi-contact systems [11], [38], [39].
Definition 2: An LCS describes the trajectories \( (x_k)_{k \in \mathbb{N}_0} \) and \( (\lambda_k)_{k \in \mathbb{N}_0} \) for an input sequence \( (u_k)_{k \in \mathbb{N}_0} \) such that

\[
x_{k+1} = Ax_k + Bu_k + D\lambda_k + d,
\]

\[
0 \leq \lambda_k \perp Ex_k + F\lambda_k + Hu_k + c \geq 0,
\]

for a given \( x_0 \) where \( x_k \in \mathbb{R}^{n_x} \), \( \lambda_k \in \mathbb{R}^{n_\lambda} \), \( u_k \in \mathbb{R}^{n_u} \).

Given \( x_k \) and \( u_k \), you can determine the associated complementary variable \( \lambda_k \) by solving the linear complementarity problem LCP \((E x_k + H u_k + c, F)\) (Definition 1). Moving forward, we describe the set of matrices in the LCS model \(^3\) as \( \theta = \{A, B, D, E, F, H, c\} \). If the elements depend on a given state-input pair \((x^*, u^*)\), we represent them as \( \Lambda^* = \Lambda(x^*, u^*) \) and the associated LCS model as \( \Lambda(\theta^*) \). Similarly, with a slight abuse of notation, we denote the state-contact force pair an LCS generates as \( (x_{k+1}, \lambda_k) = \Lambda(x_k, u_k, \theta^*) \).

D. Consensus Complementarity Control (C3)

In our previous work, we presented a framework to solve model predictive control (MPC) problems with local hybrid models (details on how this non-convex problem can be solved at real-time rates can be found in the manuscript [11]):

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

s.t.

\[
x_{k+1} = Ax_k + Bu_k + D\lambda_k + d,
\]

\[
0 \leq \lambda_k \perp Ex_k + F\lambda_k + Hu_k + c \geq 0,
\]

where \( N \) is the planning horizon, \( Q_k, Q_N \) are positive semidefinite matrices and \( R_k \) are positive definite matrices. Given an LCS model and an initial state, C3 returns an input, i.e. \( u_{C3} = C3(x_0, \Lambda(\theta^*)) \).

IV. Problem Formulation

In this work, we are interested in solving the following MPC problem at real-time rates:

\[
\min_{x_k, \lambda_k, u_k} \sum_{k} g_c(x_k, u_k)
\]

s.t.

\[
x_{k+1} = \mathcal{M}(x_k, \lambda_k, u_k)
\]

where \( g_c \) is the cost, often quadratic and \( \mathcal{M} \) is the discretization of \( \mathcal{M} \) (for details [40]). As in MPC frameworks, one can utilize the first input after solving \( [\Theta] \), denoted as \( u_0(\mathcal{M}) \) while repeatedly solving the problem in a receding horizon manner. Here, we operate under the assumption that we do not have access to the ground-truth multi-contact model \( \mathcal{M} \) but possess an approximated version of it referred to as \( \mathcal{M}^\text{prior} \).

In this setting, our aim is to collect data online and adaptively learn a residual \( \mathcal{R} \) from scratch while simultaneously solving the following MPC problem, both at real-time rates:

\[
\min_{q_k, v_k, u_k, \lambda_k} \sum_{k} g_c(q_k, v_k, u_k)
\]

s.t.

\[
x_{k+1} = \mathcal{M}^\text{prior} (x_k, \lambda_k, u_k)
\]

where \( \mathcal{M}^\text{prior} \) represents our adaptive model. Because our adaptation acts jointly on the hybrid mode boundaries and their dynamics, \( \mathcal{M}^\text{prior} \) is not the result of summation of prior dynamics and residual as in classical residual learning frameworks, and will be described in Section VI. Our goal is that, over time, our adaptive model, \( \mathcal{M}^\text{prior} \), accurately captures the behavior of the true dynamics \( \mathcal{M} \), and that \( u_0(\mathcal{M}^\text{prior}) \) well-approximates \( u_0(\mathcal{M}) \) in [5]. We highlight that we want to achieve both goals at real-time rates.

V. Physics-based LCS

This section describes a process of converting Anitescu’s approach for simulating contact dynamics [36] into an LCS approximation around a given state-input pair \((x^*, u^*)\). We choose Anitescu’s formulation over other approaches, such as the Stewart-Trinkle formulation [41], because it is a convex contact model (i.e. \( F^* \) as in Section III-C) is positive semi-definite for any \( x^* \) and \( u^* \) and our proposed learning algorithm (discussed in Section VI) relies on this assumption. Next, we describe our physics based model, \( \mathcal{M}^\text{prior} \), which is a discrete-time approximation of \([1], [2]\):

\[
q_{k+1} = q_k + \Delta t v_{k+1},
\]

\[
v_{k+1} = v_k + M^{-1} \left( \Delta t Bu_k - \Delta t C(q_k, v_k) + J_c(q_k)^T \lambda_k \right),
\]

with the complementarity constraints:

\[
0 \leq \lambda_k \perp \frac{E^T}{\Delta t} \left( \phi + J_n(q_k)(q_{k+1} - q_k) \right) + \mu J_I(q_k)v_{k+1} \geq 0.
\]

Here \( \phi \) represents the distance between rigid body pairs, \( J_n, J_I \) are Jacobians for normal and tangential directions, the contact Jacobian is defined as \( J_c = E^T J_n + \mu J_I \), \( \mu \) represents the coefficient of friction and \( E_l = \text{blkdiag}(e, \ldots, e) \) with \( e = [1, \ldots, 1] \in \mathbb{R}^{1 \times n_c} \) (where \( n_c \) represents the number of edges of the polyhedral approximation of the friction cone).

Given state \((x^*)^T = [(q^*)^T, (v^*)^T]\) and input \( u^* \), we can approximate \([7]\) as:

\[
q_{k+1} = q_k + \Delta t v_{k+1},
\]

\[
v_{k+1} = v_k + \Delta t J_f \left[ q_k^T \quad v_k^T \quad u_k^T \right]^T + D\lambda_k + d_v,
\]

where \( J_f = J_f(q^*, v^*, u^*) \) is the Jacobian of \( f(q, v, u) = M^{-1}(q)Bu - M^{-1}(q)C(q, v) \) evaluated at \((q^*, v^*, u^*)\), \( d_v = f(q^*, v^*, u^*) - J_f \left[ (q^*)^T \quad (v^*)^T \quad (u^*)^T \right]^T \) is a constant vector and \( D = M^{-1}(q)J_c(q^*)^T \). Similarly the equation \([8]\) can be approximated as:

\[
0 \leq \lambda_k \perp \frac{1}{\Delta t} E_l^T \left( \phi(q^*) + J_n(q^*)q_k - J_n(q^*)q_k^* \right) + J_c(q^*)v_{k+1} + \epsilon \lambda_k \geq 0.
\]

We note that because simulation considers single-step predictions, and MPC requires multi-step predictions, \([9]-[10]\) differs slightly from the LCS presented in previous work [36]. These equations can be written in the LCS format \([8]\) and represented as \( \Lambda(\theta^*) \). We have also introduced a regularizing term \( \epsilon \lambda_k \) in \([10]\) to ensure that \( F^* \) is positive definite as Anitescu’s formulation produces a positive semi-definite \( F^* \).
VI. ADAPTIVE MPC WITH RESIDUAL LEARNING

In this section, we describe our adaptive MPC framework shown in Figure 2. First, we describe the residual learning module (shown in red) in detail. Then we give details about the interaction of the residual learning module with C3.

Standard residual learning [43] focuses on models:

\begin{equation}
    x_{k+1} = M^{\text{prior}}(x_k, u_k) + R_{\text{traditional}}(x_k, u_k),
\end{equation}

where the prior model and the learned residual are combined in an additive manner. If we consider our physics based model, \(M^{\text{prior}}\), to be an LCS (as in Section V), (11) is equivalent to:

\begin{equation}
    x_{k+1} = A^* x_k + B^* u_k + D^* \lambda_k + d^* + R_{\text{traditional}}(x_k, u_k),
\end{equation}

\begin{equation}
    0 \leq \lambda_k \perp E^* x_k + F^* \lambda_k + H^* u_k + c^* \geq 0,
\end{equation}

and one can use state-of-the-art residual learning frameworks [44], [45] to learn \(R_{\text{traditional}}\). This form of adaptation is well studied, and not the focus of this paper. Furthermore, because (12) does not adapt the hybrid structure encoded in the complementarity constraints, it is doomed to be data inefficient and will struggle to adapt [15], [16]. Unlike in prior methods, our focus is on learning both the hybrid boundary and the contact dynamics:

\begin{equation}
    x_{k+1} = A^* x_k + B^* u_k + D^* \lambda_k + d^*,
\end{equation}

\begin{equation}
    0 \leq \lambda_k \perp E^* x_k + F^* \lambda_k + H^* u_k + c^* + R(x_k, u_k, \lambda_k) \geq 0,
\end{equation}

where \(R\) represents the model error in contact equation, e.g. (10) and this effect implicitly appears in the dynamics equation. We will focus on learning a residual that is of the form \(c_r \in \mathbb{R}^{n_c}\) as we have observed high performance in our experiments with this choice, but our framework is easily adaptable to the more general setting.\(^1\) This residual, denoted as \(c_r\), possesses the ability to capture inaccuracies such as those in normal distance, tangential friction directions and coefficients of friction. Hence, we consider:

\begin{equation}
    x_{k+1} = A^* x_k + B^* u_k + D^* \lambda_k + d^*,
\end{equation}

\begin{equation}
    0 \leq \lambda_k \perp E^* x_k + F^* \lambda_k + H^* u_k + c^* + c_r \geq 0,
\end{equation}

where we learn the vector \(c_r\) adaptively at real-time rates to compensate the error. We note that, as in traditional residual learning, our learned residual term \(c_r\) might be time-varying, and thus also capture state-dependent variations in the complementarity constraints. Equation (14) consists of physics based model parameters \(\theta^*\), derived (via Anitescu formulation) from our prior knowledge of the system and also the residual vector \(c_r\) that is learned by collecting data during the experiment. We denote (14) as \(L(\theta^*, c_r)\) that combines the prior model \(M^{\text{prior}}\) with the learned residual \(R\).

Once an experiment starts, we collect data of the form \(B = \{x_{k+1}, x_k, u_k\}_{k=n+1}^{n+n_b}\) into our buffer where \(n_{ct}\) represents the current time-step and \(n_b\) is the number of data points stored. For each \((x_k, u_k)\), we also calculate the corresponding LCS matrices \(\theta^*_k\) (consists of \(A^*_k, B^*_k, \ldots, F^*_k\)) via the method described in Section V and define the buffer appended with those matrices as \(B^A = \{x_{k+1}, x_k, u_k, \theta^*_k\}_{k=n+1}^{n+n_{ct}}\). Next, we introduce a variant of the implicit loss function that was defined in [6]:

\begin{equation}
    L_c(B^A, c_r) = \sum_{k=n+1}^{n+n_{ct}} l_c(x_{k+1}, x_k, u_k, \theta^*_k, c_r),
\end{equation}

where the function \(l_c(x_{k+1}, x_k, u_k, \theta^*_k, c_r)\) is defined as

\begin{equation}
    l_c(\cdot) = \min_{\lambda_k \geq 0, n_c > 0} \frac{1}{2} \left( D^T k \lambda_k + z \right)^T Q_d(D^T k \lambda_k + z) + \frac{1}{\epsilon} \left( \lambda_k^T \eta_k + \frac{1}{2} \gamma \right) \theta^* k - c_r - \eta_k \theta^* k \right)^2,
\end{equation}

where \(z = A^*_k x_k + B^*_k u_k + d^*_k - x_{k+1}\) and \(q = \frac{E^T k x_k + F^T k u_k + c^*_k}{E^T k u_k + c^*_k}\). Here, \(\gamma\) is a constant such that \(0 < \gamma < \sigma_{\min}(F^T k F^T k)\) for all \(k\) where \(\sigma_{\min}()\) denotes the smallest singular value. It is important to note that based on our specific formulation using Anitescu’s method, all \(F^*_k\) are positive definite and we can always find a \(\gamma\) that satisfies the given equality. Similarly, \(\epsilon\) is a constant such that \(\epsilon > 0\). Under these conditions, we can calculate the gradient of the loss function with respect to the residual parameter \(c_r\), i.e. \(\nabla_{c_r} L_c(\cdot)\) following our previous work [6].

This approach requires solving a single quadratic program per data point in the batch (hence is relatively fast). After gradient calculations, residual parameters are updated with a simple gradient step (via learning rate \(\xi > 0\)). Following this discussion, Algorithm 1 summarizes how the proposed learning method works.

Both our residual learning module and C3 run at real-time rates (Figure 2). The MPC algorithm uses the latest residual

\(^1\)Proposed framework can handle residuals of the form \(R(x_k, \lambda_k, u_k) = E_k x_k + F_k \lambda_k + H_k u_k + c_r\).

\(^2\)As in Section III-C, \(A^*_k = A(x^*, u^*)\) where \((x^*, u^*) = (x_k, u_k)\).
value \( c_r \), as well as the current state \( x^* \) and computes the optimal input \( u_{C3} = C3(x^*, L(\theta^*, c_r)) \). Then, the desired next state-contact force pair is computed as \( (x_d, \lambda_d) = L(x^*, u_{C3}, \theta^*, c_r) \). An impedance controller is used to track the desired values \([46],[47]\).

VII. EXAMPLES

A. Synthetic Example: Cart-pole with Soft Walls

We consider a classical cart-pole underactuated system which has been augmented with two soft walls (Figure 3). The pole can contact these walls, requiring contact-aware control to stabilize the system (for further details, see [38]):

\[
\dot{x}_k = Ax_k + Bu_k + D\lambda_k + d,
\]

\[
0 \leq \lambda_k \perp Ex_k + F\lambda_k + c \geq 0,
\]

where we have perfect knowledge of the parameters except for \( c \). We write \( c = \hat{c} + \Delta \phi \), where \( \hat{c} \) is our initial guess of the parameter. For the purposes of illustration, we begin with an initial error of \( \Delta \phi = [-1.15, 1.15] \). Notice that the model error that is related to each contact (\( \lambda_k \)) has a different sign. Our deliberate choice of this setup was made to create a range of scenarios, allowing the contact model to predict both situations: contact and no contact, as well as the presence or absence of actual contact events during those situations. In Figure 3, it can be seen that our method successfully returns useful gradient information in all of those scenarios. We highlight that our approach is capable of adapting even when there is no actual contact events (Figure 3). We also show that our framework successfully stabilizes the system as well as learning the true residual values (Figure 4).

B. Hardware Experiment: Adaptive Trajectory Tracking

Here, we show that our real-time adaptive MPC framework can reliably be used for multi-contact manipulation tasks that require high-speed reasoning about contact events. Our goal is to roll a rigid ball along a circular trajectory using a Franka Emika Panda Arm (Figure 1). We perform vision-based estimation for the ball using Hough transform \([48]\) and utilize an impedance controller \([46],[47]\) to track high-level commands that our adaptive MPC produces (Figure 2). For adaptive MPC, we simplify the arm as a point contact. For full details please check the manuscript \([11]\).

In previous work, careful manual identification of model parameters was required to achieve success \([11]\). Here, via our adaptive MPC approach, we do not need to assume access to an accurate model. For this experiment, the actual radius of the ball (\( M \)) is 5 mm smaller than our parameter estimation (\( M_{\text{prior}} \)). The state estimation for the ball is noisy (vision-based, 80 Hz) and we do not have accurate estimation of many model parameters such as coefficient of friction. As a result, MPC with this incorrect model fails to interact with the ball, and therefore fails to accomplish the task. Due to the stiff, hybrid nature of contact dynamics, we have found MPC to be highly sensitive to modeling errors that effect the contact/no-contact transition.

Even though the purely model-based approach fails, our adaptive MPC framework successfully accomplishes the task. The residual learning module (Algorithm 1, 20 Hz), in combination with the C3 module (80 Hz), identifies the discrepancy at real-time rates. In the supplementary video, it can be observed that at the beginning the end-effector misses the ball due to inaccurate radius estimation. The C3 module then tries, and fails, to initiate contact. As motivated earlier, this discrepancy creates a non-zero gradient and therefore the residual values that correspond to end-effector ball contact pair gradually increase (shown in Figure 6).

In the end, with the learned residual compensating for the inaccurate model, our approach successfully rolls the ball for 4 successive circles. Figure 6 shows the loss, \( \mathcal{L}_e \), as
well as the residual, $c_r$, for this experiment. The residual for the end-effector and ball contact shows convergence as the rolling proceeds and the loss curve shows a decreasing trend, proving the effectiveness of our method. We note that in this experiment, the LCP violation rate is $\epsilon = 10^{-7}$, the terms in $Q_d$ related to balls translational velocity are set to $10^5$ (others are set to zero), learning rate is $\xi = 10^{-3}$, stiffness parameter is $\gamma = 10^{-2}$ and we have a batch data size of $n_b = 10$. We demonstrate the trajectories of the ball with respect to the desired path in Figure 5.

C. Hardware Experiment: Objects of Non-smooth Surface

We repeat the previous hardware experiment (Section VII-B) with slightly deformable objects that have non-smooth surface geometries (Figure 1). We do not have accurate initial estimates of the geometry as our estimate of a fruit’s geometry is simply a sphere. For each experiment experiments, we assign an approximate radius for the given fruit and set the sphere radius in our prior model, $M^{\text{prior}}$, accordingly.

Due to the bulges and dents on the fruit’s surface, initiating rolling can be hard since the fruit can roll back even after a push in the correct direction. MPC without adaptation gets stuck in the beginning and fails to even initiate rolling. In contrast, our method quickly adapts and starts making stronger pushes to initiate rolling (as shown in the supplementary video). In 90% of the 20 experiments, our method successfully initiated rolling and started tracking the circular path, while MPC without adaptation was never successful (0%).

Fruits tend to produce unpredictable motions at times due to its non-smooth surface geometry, but our method can adapt and accomplish the task. Our approach has managed to track at least one circle for 70% of the trials with orange and 50% of trials with lime (20 experiments for each case). In Figure 5, we report the long-term tracking (4 successive circles) performance of our method with multiple different fruits. The videos of experiments are in the supplementary material.

VIII. Conclusion

We presented an adaptive model predictive control framework for multi-contact systems. The approach uses online residual updates and adaptively compensates model errors and uncertainties at real-time rates. The effectiveness of the method has been shown on multiple hardware experiments, including high-dimensional manipulation problems that include objects with non-smooth surface geometries (fruits).

We have further shown examples where pure model-based control fails but our adaptive strategy leads to success. We have also demonstrated that our learning module is capable of learning the contact model accurately even without actual contact interactions. Also, our parameter estimation can converge to the true parameter values at real-time rates.

We are interested in exploring a wider range of residual models in the future. For example, we might extend our results to include state-dependent terms, or simultaneously learn the non-contact and contact residuals. Integrating our framework with tactile sensors is also an interesting future direction which could speed up our learning process as well as increase the performance of the low-level controller [38].