Non-Parametric Self-Identification and Model Predictive Control of Dexterous In-Hand Manipulation

Podshara Chanrungmaneekul[1], Kejia Ren[1], Joshua T. Grace[2], Aaron M. Dollar[2], and Kaiyu Hang[1]

***Abstract*— Building hand-object models for dexterous in-hand manipulation remains a crucial and open problem. Major challenges include the difficulty of obtaining the geometric and dynamical models of the hand, object, and time-varying contacts, as well as the inevitable physical and perception uncertainties. Instead of building accurate models to map between the actuation inputs and the object motions, this work proposes to enable the hand-object systems to continuously approximate their local models via a self-identification process where an underlying manipulation model is estimated through a small number of exploratory actions and non-parametric learning. With a very small number of data points, as opposed to most data-driven methods, our system self-identifies the underlying manipulation models online through exploratory actions and non-parametric learning. By integrating the self-identified hand-object model into a model predictive control framework, the proposed system closes the control loop to provide high accuracy in-hand manipulation. Furthermore, the proposed self-identification is able to adaptively trigger online updates through additional exploratory actions, as soon as the self-identified local models render large discrepancies against the observed manipulation outcomes. We implemented the proposed approach on a sensorless underactuated Yale Model O hand with a single external camera to observe the object's motion. With extensive experiments, we show that the proposed self-identification approach can enable accurate and robust dexterous manipulation without requiring an accurate system model nor a large amount of data for offline training.**

I. Introduction

Dexterous in-hand manipulation is a system-level problem consisting of an array of sub-problems, ranging from the modeling of contacts and hand-object dynamics [1], [2], to the perception, planning, and control of the task-oriented hand-object coordination [3]–[5]. At the core of all these problems, almost all existing approaches are challenged by the gaps between the required prior knowledge and online feedback, and the actual limited information available to the system [6]. For example, a very common assumption made to contact-based manipulation systems is that the object model is perfectly known. In practice, however, this is rarely possible even if many sensing modalities are available. As such, in-hand manipulation systems are often limited either in their capability of handling complex dynamics or in their generalizability across similar variations of task setups. Although learning-based approaches have been extensively investigated and shown the capabilities of acquiring complex manipulation skills [7], [8], the data, which is the enabling factor for such systems, is also a major limitation in more general and dynamic tasks.

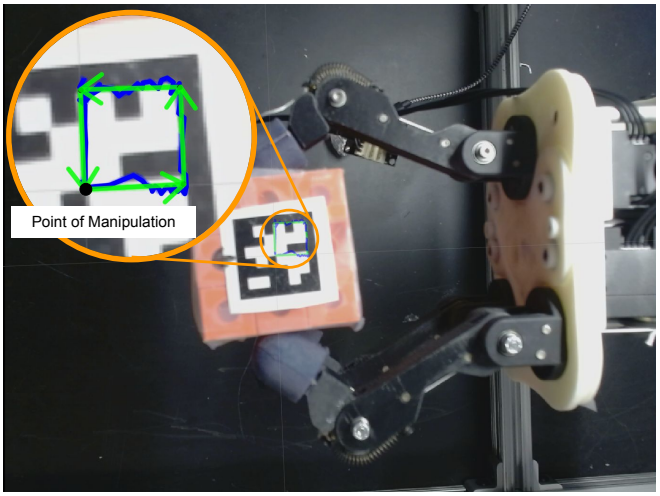

Fig. 1: An encoderless underactuated Yale Model O hand is tasked to manipulate an unknown object to trace a reference path (green trajectory). Enabled by the proposed non-parametric self-identification, and with no prior knowledge assumed, the hand is able to self-identify a system model using only 15 data points and then accurately complete the task (blue trajectory).

To bridge the aforementioned gaps while not shifting more burden to the sensing or data collection sides, we previously proposed the idea of *self-identification* [9] . For hand-object systems modeled with a number of known and missing parameters, the missing ones were iteratively self-identified by the hand-object system through exploratory actions without adding any additional sensors. The self-identified system then showed great performance in precise dexterous manipulation while tracking the real-time changes of the missing parameters. This approach was inspired by human manipulation where we do not have accurate models of everything *a priori*. Rather, humans often use a strategy that shifts the system paradigm from "sense, plan, and act" to " act, sense, and plan". However, [9] still modeled the hand-object system analytically and required a number of parameters to be self-identified and tracked in real-time, rendering the approach not easily generalizable and computationally very expensive.

To this end, this work proposes to replace the analytic parameter-based models with a non-parametric model to be self-identified. We consider a challenging setup with an encoderless underactuated robot hand, as shown in Fig. 1.

[1]Department of Computer Science, Rice University, Houston, TX 77005, USA. {pc45, kr43, kaiyu.hang}@rice.edu.

[2]Department of Mechanical Engineering and Material Science, Yale University, New Haven, CT 06511, USA. {josh.grace, aaron.dollar}@yale.edu.

This work was supported by the US National Science Foundation grant FRR-2133110 and FRR-2132823.

Given an unknown grasp on an unknown object, the proposed system first collects a small number of manipulation data points through exploratory actions, for which the grasp stability is passively secured by the hand's compliance. The system then learns a non-parametric model to map from the hand control directly to the object motion, yielding a self-identified local model of the hand-object system. In this work, the non-parametric model was learned by a Gaussian Process Regressor. By integrating the self-identified model into a Model Predictive Control (MPC) framework, we show that in-hand manipulation can be precisely achieved, while the self-identified model can be updated through additional exploratory actions as needed. A system diagram is illustrated in Fig. 2.

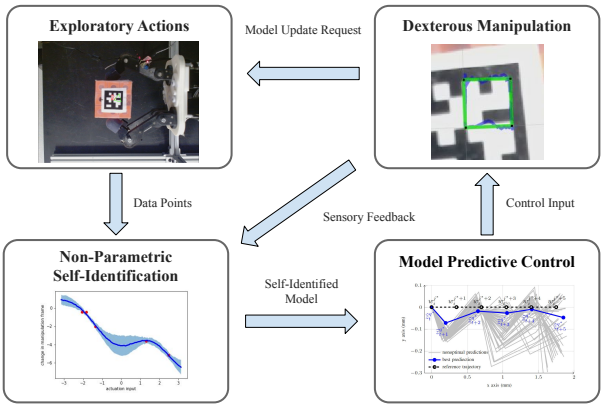

Fig. 2: System diagram of the proposed non-parametric self-identification for dexterous in-hand manipulation.

II. Related Work

Hand-Object Models: Traditional models of in-hand manipulation systems often assume that precise geometric and physical models of the hand, object, contacts, etc., are available [2]. Since such approaches are very sophisticated in modeling every detailed aspect of the system, they are limited in scalability and normally focus on specific sub-problems of hand-object systems, including contact modeling [10], force control [11], stability maintenance [3]. More importantly, model-based methods are inherently limited as the assumptions of model availability often do not hold in reality. On another hand, simplified models such as action primitives have been designed to model the manipulation mappings [12], [13]. However, as the primitives are handcrafted, they are not generalizable, nor scalable. This work proposes self-identifying a non-parametric model of the hand-object system, aiming at avoiding sophisticated modeling, model availability assumption, and unnecessary model simplification.

Data-Driven Approaches: With sufficient data and training, learning-based methods have shown unprecedented performance in acquiring complex manipulation skills [8]. In an end-to-end manner, data has filled in the gap traditionally formed by the lack of *a priori* system information and perception uncertainties [7], [14]. However, as such methods are sensitive to the amount and diversity of the training data, they are often not generalizable, even to similar task variations. Unlike those data-demanding approaches, the non-parametric Gaussian Process model employed in this work is lightweight and known to work with a minimal amount of data [15]. As such, it enables the self-identification of hand-object models online through only a few exploratory actions.

Interactive Perception: Leveraging proactive manipulation actions to unveil hidden system information can greatly improve the robot perception under limited sensing [16]. Particularly for hand-object systems, interactive perception can enable grasping and in-hand manipulation under large uncertainties [17], [18]. While most interactive perception methods focus on estimating specific parameters of a system, our non-parametric self-identification aims to directly build a mapping, approximated locally, from the hand's actuation input to the object motions to enable the MPC control of precise dexterous manipulation.

III. Hand-Object Systems and Problem Formulation

In this work, we aim to address the in-hand dexterous manipulation problem, where an object grasped by an underactuated robot hand needs to be reconfigured to certain poses. We consider the hand and the object as a whole discrete-time dynamical system. The underlying state of the hand-object system at time t is $s_t \in \mathbb{R}^N$, where N is the number of all the physical properties necessary for uniquely identifying a system state, such as hand joint configurations and hand-object contact locations. The control of the system, $u_t \in \mathbb{R}^C$, is the actuation input to the hand at time t. For an underactuated hand, the dimension of controls, C, is less than the degree of freedom of the hand. The dynamics of the system can be represented by a transition function $g : \mathbb{R}^N \times \mathbb{R}^C \mapsto \mathbb{R}^N$ such that

$$s_{t+1} = g(s_t, u_t) \tag{1}$$

Additionally, we select a fixed point on the object's surface and use this point's motion to represent the object's motion. We term this point as the Point of Manipulation (POM), whose position at time t is denoted by $z_t \in \mathbb{R}^3$. As such, the object's motion at time t can be represented by the finite difference of POM's positions, i.e., $\delta z_t = z_{t+1} - z_t$.

However, building an analytical model for such hand-object systems is impossible due to the following challenges: 1) the system state s_t is not fully observable as it contains parameters not obtainable due to the limited sensing capability of the system, such as the hand-object contact locations and the joint angles of the underactuated hand; and 2) the system dynamics g requires accurate geometric models of the hand and the object, which are in general unavailable. Moreover, even if g is solvable, it is hard to generalize across different object shapes or different types of contacts. Instead, assuming a neglectable change of the hand-object system state after applying a small enough control, we can locally approximate the system model without exactly knowing the current system state s_t. For this, we use another function

Algorithm 1 Dexterous Manipulation via Self-Id and MPC

Input: Reference trajectory X, a distance threshold α, number of initial exploratory actions d and a, number of adapting actions for model update b

$t \leftarrow 0$, $z_0 \leftarrow$ OBSERVEPOM()
$x_0 \leftarrow z_0$
$\Gamma, \Gamma^{-1} \leftarrow$ SELFIDENTIFICATION(z_0, d, a) ▷ Alg. 2
for $x_i \in X$, $i = 1, \cdots, n$ **do** ▷ Waypoints in X
  while $\|x_i - z_t\| > \alpha$ **do**
    $u_t \leftarrow$ MPC(z_t, x_{i-1}, x_i) ▷ Alg. 3
    $z_{t+1} \leftarrow$ EXECUTE(u_t) ▷ Observe POM
    if $\epsilon_t > \gamma$ **then** ▷ Sec. IV-B
      $\Gamma, \Gamma^{-1} \leftarrow$ SELFIDENTIFICATION$(z_{t+1}, 0, b)$
    end if
    $t \leftarrow t + 1$
  end while
end for

$\Gamma : \mathbb{R}^C \rightarrow \mathbb{R}^3$ to represent the locally approximated system transitions, which maps from the control input to the object's motion:

$$\Gamma(u_t) = \delta z_t \tag{2}$$

In addition, the inverse of the approximated system model is defined by $\Gamma^{-1} : \mathbb{R}^3 \rightarrow \mathbb{R}^C$. We name Γ and its inverse Γ^{-1} as *local manipulation models*. To precisely manipulate the object without prior knowledge of the system dynamics, the system needs to self-identify the local manipulation models Γ and Γ^{-1} and adapt them to different hand-object configurations when necessary.

In this work, the hand-object system is tasked to find and execute a sequence of control inputs to gradually move the object, such that the POM will trace through a reference trajectory represented by a sequence of T desired positions of POM: $X = \{x_1, \cdots, x_T\}$, where $x_1, \cdots, x_T \in \mathbb{R}^3$ are called *keypoints* of the trajectory. We formulate such dexterous in-hand manipulation as a self-identification and control problem, and approach the problem through non-parametric learning and Model Predictive Control (MPC), as illustrated in Fig. 2 and summarized in Alg. 1. The details of the approach will be described in Sec. IV and Sec. V.

Starting with the POM positioned at $z_0 \in \mathbb{R}^3$, the system identifies the models Γ and Γ^{-1} through a small number of initial exploratory actions consisting of d randomly sampled and a calculated controls, as will be detailed in Sec. IV-A and Alg. 2. Then, the self-identified models will be integrated into an MPC framework to generate real-time control u_t to move POM toward targeted keypoints of the reference trajectory. Meanwhile, the system observes the outcome position of POM z_{t+1} after each generated control u_t has been executed. If a large deviation from the desired reference trajectory has been detected, as will be described in Sec. IV-B, the system will update the models Γ and Γ^{-1} by performing b more exploratory actions. As such, being self-identified and updated in real-time, the models are used to generate controls to precisely move the POM on the object to reach each keypoint of the reference trajectory sequentially.

IV. NON-PARAMETRIC SELF-IDENTIFICATION

In this section, we present a non-parametric approach based on Gaussian Process Regression to facilitate the self-identification of the local manipulation models Γ and Γ^{-1}. Such a non-parametric learning approach does not require a parametric form of the models, which is challenging to specify and difficult to generalize. Moreover, as an efficient nonlinear function approximator that works well with a small amount of data, Gaussian Process Regression alleviates the burden of heavy online data collection, which is time-consuming for a real-world system. Specifically, as described in Sec. IV-A, with a set of data points collected by the system through online exploratory actions, the manipulation models Γ and Γ^{-1} can be learned efficiently to find the inherent relation between the control inputs and the object's motion.

A. Exploratory Actions and Self-Identification

To self-identify the manipulation models with data collected online, we dynamically maintain a training dataset, $\mathcal{D} = \{(\hat{u}_i, \delta\hat{z}_i)\}_{i=1}^{P}$, consisting of P data points the system has observed. Each data point is a pair $(\hat{u}_i, \delta\hat{z}_i)$, where $\hat{u}_i \in \mathbb{R}^C$ is a control the system has executed and $\delta\hat{z}_i \in \mathbb{R}^3$ is the object's motion observed after executing $\hat{u}_i$. The dataset $\mathcal{D}$ is initially empty but will be updated to have more data points as the system keeps executing to manipulate the object. We use *exploratory actions* to name the data points in the training dataset $\mathcal{D}$, as such actions are performed for exploring the system models. We illustrate how such exploratory actions are generated and used for self-identification in Alg. 2.

Without prior knowledge about the hand-object configuration, the system begins by randomly generating a number of d controls. Each control is randomly sampled from a C-dimensional uniform distribution within the range $[-l, l]$, where l is chosen to be arbitrarily small while not being overwhelmed by the system's physical uncertainties. The system will execute each of these d controls, observe the object's motion after each control execution, and add them to the training dataset $\mathcal{D}$.

However, certain patterns of the object's motions might not be present in $\mathcal{D}$ since the size P of the dataset is kept small in practice. Therefore, to have more representative data to effectively learn the manipulation models, we intend to increase the local density of the dataset $\mathcal{D}$ by selecting additional a controls to explore. For that, we define the local density of the dataset $\mathcal{D}$ at its i-th data point to be the reciprocal of the distance between $\delta\hat{z}_i$ and its nearest neighbor in $\mathcal{D}$:

$$\rho_{\mathcal{D}}(i) = \frac{1}{\min_{j \neq i} \|\delta\hat{z}_i - \delta\hat{z}_j\|} \tag{3}$$

The data point with the lowest local density will be used to calculate a new control $\hat{u}_s$, to be added into the training dataset $\mathcal{D}$ with its corresponding observation of the object's motion $\delta\hat{z}_s$. This new control $\hat{u}_s$ is determined by the average

Algorithm 2 Self-Identification

Input: Observed POM z_0, number of random actions d, number of extra actions a
Output: Non-parametric manipulation models Γ and Γ^{-1}

$\mathcal{D} \leftarrow$ AcquireDataset() ▷ Training Dataset
for $i = 1, \cdots, d$ **do** ▷ d Random Actions
  $\hat{u}_i \leftarrow$ Uniform$(-l, l)$ ▷ Uniform Sampling from $[-l, l]$
  $z_i \leftarrow$ Execute$(\hat{u}_i)$ ▷ Observe POM
  $\delta\hat{z}_i \leftarrow z_i - z_{i-1}$ ▷ Object's Motion
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\hat{u}_i, \delta\hat{z}_i)\}$
end for
for $i = d+1, \cdots, d+a$ **do** ▷ a Extra Actions
  $p \leftarrow \arg\min_j \rho_{\mathcal{D}}(j)$
  $p' \leftarrow \arg\min_{j \neq p} \|\delta\hat{z}_p - \delta\hat{z}_j\|$ ▷ Nearest Neighbor
  $\hat{u}_s \leftarrow (\hat{u}_p + \hat{u}_{p'})/2$ ▷ Eq. (4)
  $z_i \leftarrow$ Execute$(\hat{u}_s)$ ▷ Observe POM
  $\delta\hat{z}_s \leftarrow z_i - z_{i-1}$ ▷ Object's Motion
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\hat{u}_s, \delta\hat{z}_s)\}$
end for
$\Gamma, \Gamma^{-1} \leftarrow$ GPR$(\mathcal{D})$ ▷ Gaussian Process Regression
return Γ, Γ^{-1}

of this data point and its nearest neighbor:

$$\begin{aligned} \hat{u}_s &= \frac{\hat{u}_p + \hat{u}_{p'}}{2} \\ p &= \underset{j \in \{1, \cdots, |\mathcal{D}|\}}{\arg\min} \ \rho_{\mathcal{D}}(j) \\ p' &= \underset{j \in \{1, \cdots, |\mathcal{D}|\} \setminus \{p\}}{\arg\min} \ \|\delta\hat{z}_p - \delta\hat{z}_j\| \end{aligned} \tag{4}$$

Using this method, we would approach a uniform distribution as the number of exploratory actions increases. With the dataset $\mathcal{D}$ generated by exploratory actions, the manipulation models Γ and Γ^{-1} can be efficiently self-identified by Gaussian Process Regression (Alg. 2). It is worth noting that both models Γ and Γ^{-1} are regressed with the same dataset $\mathbf{D}$, but with a different domain and codomain of the data. As Γ and Γ^{-1} are independently learned, we cannot guarantee a closed loop between them. In other words, for the self-identified models, $\Gamma^{-1}(\Gamma(u)) \neq u$.

B. Model Update

While manipulating the object, the underlying system dynamics can vary over time due to changes in hand configuration and contacts. This can cause the failure of the self-identified models as they are locally approximated. Therefore, we introduce a mechanism in our framework that adaptively updates the model online when needed. As demonstrated with Fig. 3, if a large discrepancy between the observed system state and the prediction of self-identified models has been detected, the model will be updated with b additional actions calculated by Eq. (4). We particularly name such exploratory actions *adapting actions*, as they are used to adapt the model to a new locality.

For that, we need to define an indicator, to determine when the model update should be triggered. Consider the in-hand manipulation task defined in Sec. III. Suppose that the POM has already reached the first $i-1$ keypoints of the reference trajectory X through in-hand manipulation. In other words, the system currently targets the next keypoint $x_i \in X$. We linearly interpolate from x_{i-1} to x_i to create an intermediate trajectory $W_i = \{w_i^1, w_i^2, \cdots, w_i^M\}$ of M waypoints, where $w_i^1 = x_{i-1}$ and $w_i^M = x_i$. Given POM's position $z_t \in \mathbb{R}^3$ at the current time step, the nearest waypoint $w_i^{j^*}$ in the intermediate trajectory W_i is found by

$$j^* = \underset{j \in \{1, \cdots, M\}}{\arg\min} \ \|z_t - w_i^j\| \tag{5}$$

With this, we define the manipulation error ϵ_t at time t to be the distance between POM's position and its nearest waypoint in the intermediate trajectory:

$$\epsilon_t = \|z_t - w_i^{j^*}\| \tag{6}$$

The manipulation error ϵ_t measures how much the POM deviates from its desired trajectory. If ϵ_t is greater than a threshold γ, the framework will update the model. This mechanism can be found in lines 8 and 9 in Alg. 1.

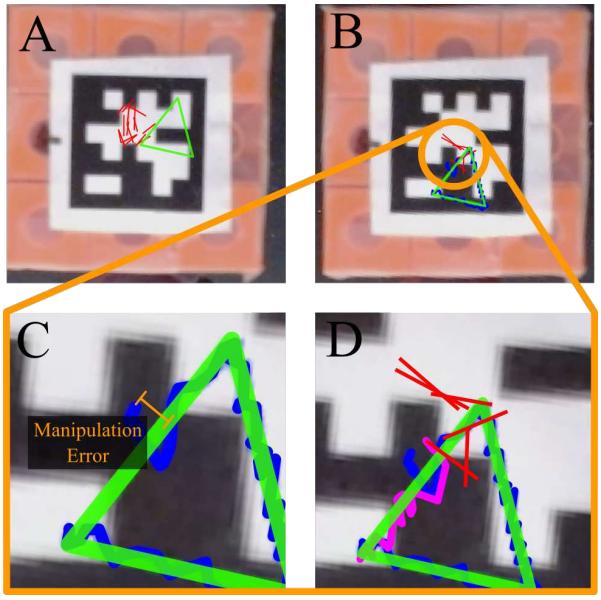

Fig. 3: An example demonstration of model update with the setup in Fig. 1, where the system is tasked to trace the reference trajectory (green triangle): (A) At the beginning, the system performs $d+a$ initial exploratory actions (red) to learn the non-parametric models. (B) The model update request is triggered during manipulation, according to the sensory feedback. (C) The manipulation error ϵ_t (yellow) exceeds the threshold γ, triggering the model update. (D) Additional b data points obtained from adapting actions (red) are used to update the models. Then, the system continues with the trajectory tracing tasks (magenta).

C. Model Transfer

The underlying system dynamics become different when the object's geometry or grasp configuration has changed. Intuitively, however, the patterns of manipulation can render some similarities across such geometric and physical variations. In other words, the self-identified manipulation models should feature generalizability on different objects or different contact locations, facilitating model transfer between different hand-object setups.

As such, when manipulating a new object, our framework has the option to initialize the manipulation models Γ and

Γ^{-1} by the models that have been learned previously with a different object. By such model transfer, the system can skip the data-gathering step for initial exploratory actions in line 3 of Alg. 1, to speed up the manipulation of a new object. We will show the benefits of using model transfer in our framework through the experiments in Sec. VI-D.

V. Model Predictive Control

As the manipulation models Γ and Γ^{-1} can be efficiently self-identified through exploratory actions in Sec. IV, we can use them as predictive models to develop a control scheme to generate controls based on the desired motion of the object. To this end, we integrate the models Γ and Γ^{-1} in a Model Predictive Control (MPC) framework to iteratively generate controls at each time step. Our MPC-based control scheme is presented in Alg. 3 and the details will be described below. Benefitting from the efficient inference of Γ and Γ^{-1}, the MPC effectively meets the requirement of real-time executions.

As some definitions in Sec. IV-B are useful for MPC, we briefly recall them here: between the last reached keypoint x_{i-1} and its next x_i in the reference trajectory X, we create an intermediate trajectory $W_i = \{w_i^1, w_i^2, \cdots, w_i^M\}$ of M waypoints by linear interpolation. In this intermediate trajectory W_i, we find the nearest $w_i^{j^*}$, with its index j^*, to POM's position z_t at the current time step.

Then, we use the intermediate trajectory W_i as a local reference to guide the MPC in searching for optimal control. Concretely, MPC uses the self-identified models Γ and Γ^{-1} to predict the behavior of the controlled hand-object system up to a prediction horizon L. By adding stochasticity into the prediction with some random ξ of control, it can simulate Q independent trajectories, as illustrated in Fig. 4. Each simulated trajectory $U^q = \{(\hat{u}_t^q, \hat{z}_t^q), \cdots, (\hat{u}_{t+L}^q, \hat{z}_{t+L}^q)\}$, where $q = 1, \cdots, Q$, is generated by the following iterative process starting with $k = 0$:

$$\begin{aligned} \hat{u}_{t+k}^q &= \Gamma^{-1}(w_i^{j^*+k} - \hat{z}_{t+k}^q) + \xi \\ \hat{z}_{t+k+1}^q &= \Gamma(\hat{u}_{t+k}^q) + \hat{z}_{t+k}^q \end{aligned} \tag{7}$$

where $\hat{u}_{t_k}^q \in \mathbb{R}^C$ and $\hat{z}_{t+k}^q \in \mathbb{R}^3$ are the predicted control and state (i.e., POM's pose) at time $t+k$ in the q-th simulated trajectory, and $\xi \sim \mathcal{N}(0, \sigma \mathbb{I}_C)$ is a multivariate Gaussian random variable. The scale σ of this Gaussian random variable is named *MPC optimization scale*.

Over the Q simulated trajectories, MPC searches for the optimal trajectory U^{q^*} (the blue one in Fig. 4) such that the accumulated distance between it and the intermediate trajectory W_i is minimized:

$$q^* = \underset{q \in \{1, \cdots, Q\}}{\arg\min} \left(\sum_{k=0}^{L} \| \hat{z}_{t+k}^q - w_i^{j^*+k} \| \right) \tag{8}$$

where $L = \min\{K, M - j^*\}$ is the prediction horizon (i.e., the length of the simulated trajectories) not greater than a hyperparameter K. The first control $\hat{u}_t^{q^*}$ in the optimal trajectory U^{q^*} is then sent to the hand actuators for execution. While the entire procedure of MPC is performed at each time step, the system will precisely control the object's motion, guided by the self-identified manipulation models.

Algorithm 3 Model Predictive Control (MPC)

Input: Observed POM z_t, last reached keypoint x_{i-1}, targeted keypoint x_i
Output: Optimized control for execution u_t

1: $\{w_i^1, \cdots, w_i^M\} \leftarrow$ LinearInterpolate(x_{i-1}, x_i)
2: $j^* \leftarrow \arg\min_j \|z_t - w_i^j\|$ ▷ Nearest Waypoint by Eq. (5)
3: $L \leftarrow \min\{K, M - j^*\}$ ▷ Prediction Horizon
4: **for** $q = 1, \cdots, Q$ **do**
5: $\quad \hat{z}_t^q \leftarrow z_t$
6: $\quad$ **for** $k = 0, \cdots, L-1$ **do**
7: $\quad\quad \xi \leftarrow \mathcal{N}(0, \sigma \mathbb{I}_C)$
8: $\quad\quad \hat{u}_{t+k}^q \leftarrow \Gamma^{-1}(w_i^{j^*+k} - \hat{z}_{t+k}^j) + \xi$ ▷ Predicted Control
9: $\quad\quad \hat{z}_{t+k+1}^q \leftarrow \Gamma(\hat{u}_{t+k}^q) + \hat{z}_{t+k}^q$ ▷ Predicted State
10: $\quad$ **end for**
11: **end for**
12: $q^* \leftarrow \arg\min_q \left(\sum_{k=0}^{L} \| \hat{z}_{t+k}^q - w_i^{j^*+k} \| \right)$ ▷ Eq. (8)
13: **return** $u_t^{q^*}$

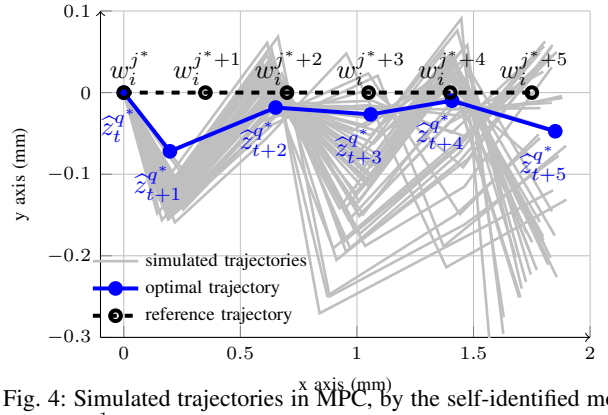

Fig. 4: Simulated trajectories in MPC, by the self-identified models Γ and Γ^{-1}. In this figure, $Q = 50$ trajectories (gray) are simulated and each one has a horizon of $L = 5$. The optimal trajectory (blue), closest to the reference trajectory (black), is selected to extract the optimal control.

VI. Experimental Evaluation

In this section, we evaluate and study the performance of the proposed framework under a real-world setting that requires precise in-hand manipulation. As shown in Fig. 5, We deployed our proposed framework on a Yale Model O underactuated hand [19]. This hand has three identical fingers, each finger of which has one motor to actuate two spring-loaded joints through the tendon. While the tendon is pulled by the motor, the joint configuration of each finger will change accordingly. The spring in each finger joint enables compliance that facilitates stable contact between the hand and the grasped object. For our experimental setup, we restricted two fingers to be parallel to each other and always take the same actuation input, while the third finger was configured to the opposite side. As such, the object's motion was physically constrained in a horizontal plane.

The POM was selected to be on the top of the object, which was tracked by a camera mounted above the object through AprilTag [20]. The camera has a resolution of

1024×512 and the tracker's frequency is 30 fps. Note that the tag-based POM tracker can be replaced by other vision-based frameworks. At the beginning of each experiment trial, the experimenter needed to hand one object in Fig. 6 to the underactuated hand by a stable grasp.

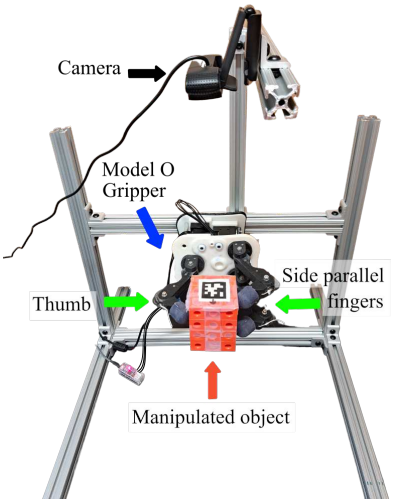

Fig. 5: The experimental setup: The Yale Model O underactuated hand is tasked to manipulate an object (an orange cube (obj #4)), whose POM is tracked by a top camera with AprilTag.

Fig. 6: The objects used in the experiments: 1) a pill bottle, 2) a mustard bottle (YCB dataset #006), 3) an apple (YCB dataset #013), 4) an orange cube, and 5) a toy airplane (YCB dataset #072a) [21].

A. Experiment Design

As defined in Sec. III, we tasked our framework to trace a reference trajectory of POM through in-hand manipulation. The reference trajectories we used in experiments are shown by the green lines in Fig. 7, including a triangle, a square, a π letter, and a spiral line. Each trajectory is represented by a sequence of desired positions of POM (i.e., keypoints, shown by the red dots in Fig. 7) in the camera's frame, therefore enabling our system to work without the necessity of hand-eye calibration. To trace a reference trajectory, the POM on the object must reach each keypoint in the correct order, with a tolerance of $\alpha = 1mm$ in Alg. 1. The blue lines in Fig. 7 showcase POM's actual trajectories while the system executes controls generated by MPC.

Besides the requirement for precise motion control, such an in-hand manipulation task challenges our framework from three other aspects: 1) With a lack of sensing capability for joint configuration and contact information, our framework needs to be effective in approximating the actual system transitions through self-identification, which is crucial for precise manipulation. 2) The real-world data collection through online manipulation is time-consuming, demanding high data efficiency of the model self-identification. 3) The contacts between the hand and the object change during manipulation, due to unpredictable sliding or rolling. This requires the adaptability of our framework to such changes. As will be shown with experiments in Sec. VI-B, VI-C, and VI-D, our framework is effective in addressing these challenges and is able to precisely control the object's motion with self-identified models under various real-world settings.

To quantitatively evaluate the performance of our framework on in-hand manipulation, we selected two metrics:

1) *Manipulation error*, as defined in Sec. IV-B, averaged over the entire trajectory. This reflects how far the actual execution deviates from the desired reference trajectory. A small manipulation error is a direct indication of accurate motion control of the object, which is highly affected by the quality of the self-identified models and the robustness of our MPC control policy.
2) *The accumulated number of adapting actions*. As described in Sec. IV-B, whenever the manipulation error exceeds a threshold $\gamma = 2mm$, our system will perform more exploratory actions to update the self-identified models. A small value of this metric means fewer times of model updates, thus reflecting the high data efficiency and good adaptability of our framework.

Note that the object was constrained to move in a horizontal plane by our setup. The reference trajectories were always given on this plane with the fixed height, and we evaluated the manipulation errors only in this plane as well.

B. Analysis on Initial Exploratory Actions

In this experiment, we study how many initial exploratory actions are needed for a decent model of self-identification. With the experiment results, we intend to show that the self-identified models, even learned with only a small amount of training data, can enable precise in-hand manipulation.

For this, we varied the number of initial exploratory actions (i.e., $d+a$ in Alg. 1) to be $10, 15, 20, 25$ and 30, and tasked the framework to manipulate all the objects in Fig. 6 and trace all the four trajectories given in Fig. 7. For each setting, we repeated the experiment 5 times. To ensure the manipulation performance is not dominated by a bad control policy, we set the MPC optimization scale not too small with $\sigma = 0.1$. The results are summarized in Fig. 8. From the results, we find fewer initial exploratory actions cause a worse quality of the self-identified system models reflected by a higher manipulation error and demand for more adapting actions. This is because the underlying system transitions can be hardly approximated via self-identification, if without sufficient exploration. However, by slightly increasing the number of initial exploratory actions, higher manipulation precision was achieved with lower manipulation errors; and

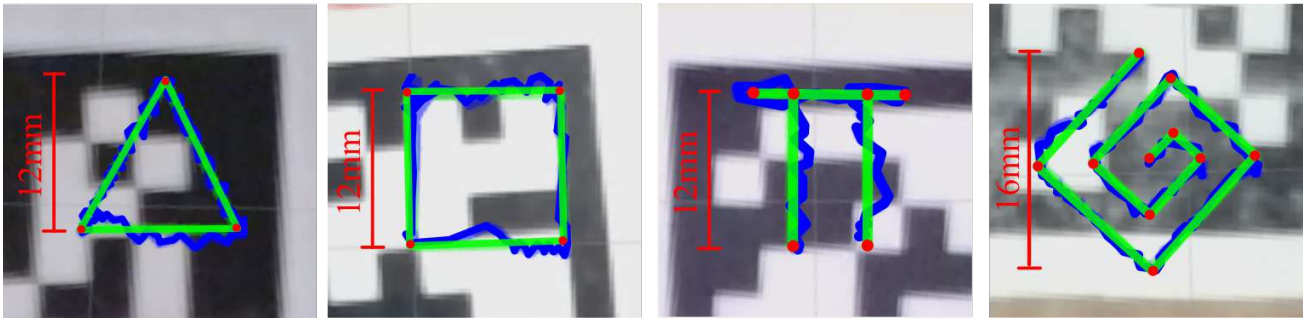

Fig. 7: Real-world trajectory tracing tasks by in-hand manipulation: a triangle, a square, a π letter, and a spiral line. Green: reference trajectories. Blue: real trajectories executed by our system. Red dots: keypoints that POM needs to sequentially go through.

the non-parametric models were better approximated via self-identification, indicated by a smaller number of adapting actions for the model update. After 20 initial exploratory actions, the manipulation performance of our framework roughly converged, and an average manipulation error of less than $0.8mm$ could be achieved for all the objects and reference trajectories. Importantly, this has demonstrated the high data efficiency of our framework, which in general only needs less than 30 data points to achieve dexterous in-hand manipulation with high precision.

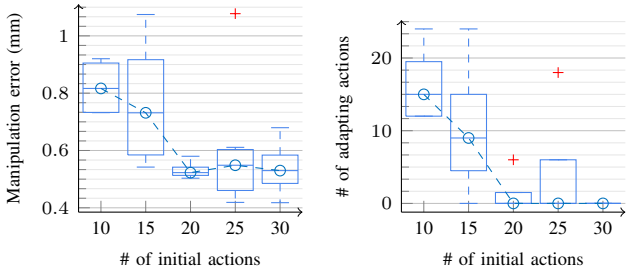

Fig. 8: Self-identification performance in terms of the number of initial exploratory actions. For different numbers of initial actions, the result is averaged over all the objects and reference trajectories.

C. MPC Performance Analysis

The self-identified models Γ and Γ^{-1} by our framework are locally approximated, thereby can never be perfect due to the lack of prior knowledge about the system and the limited sensing capability. However, for precise manipulation, we require the controls generated by MPC to be robust enough against imperfect model self-identification. In this experiment, we evaluate the robustness of the self-identified models when the control loop is closed by MPC, and analyze how it is affected by the optimization scale σ.

Intuitively, a small σ will enforce the MPC policy to be more confident about the self-identified models, thus becoming more greedy but potentially not robust to imperfect self-identifications; whereas a large σ will increase the search space of MPC for the optimal control.

In this specific experiment, we only used an orange cube (obj #4) in Fig. 6 as the object for manipulation. For MPC, the maximum prediction horizon was set to $K = 5$, and the number of simulated trajectories for optimization was set to $Q = 50$. For each different σ, we repeated the manipulation 5 times for each of the four reference trajectories in Fig. 7. The results are summarized in Fig. 9. From the results, we observed a large number of adapting actions with a small σ less than 0.02. This is expected since MPC is too greedy using the self-identified models with a small σ, resulting in fewer optimal executions and more instances of model update requests. By slightly increasing σ to introduce more stochasticity into the predictions, MPC could search more extensively for finding the optimal control and the average number of adapting actions was immediately reduced to less than 5, rendering a significant performance improvement. In general, from the experiments, we found that the self-identified models are sufficiently reliable to make predictions about control with appropriate σ; furthermore, our MPC-based control policy was able to achieve precise manipulation by closing the control loop with approximated non-parametric models.

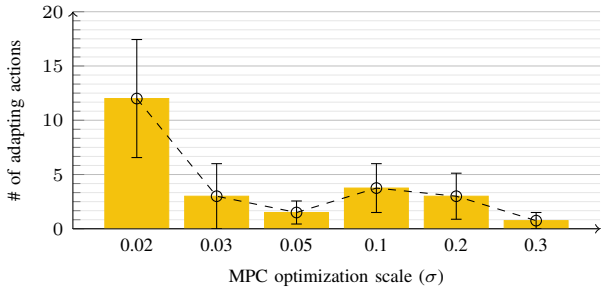

Fig. 9: MPC performance evaluation in terms of its optimization scale σ and the number of adapting actions.

D. Model Generalizability

In real-world applications, the object being manipulated and the grasp configuration are likely to be different every time. Therefore, good generalizability of the self-identified models to such variations is desirable, as it helps save the time and cost of retraining the models every time.

In this experiment, we challenged the generalizability of the self-identified models in our framework, by manipulating a new object (*target object*) with models learned through manipulating a different object (*source object*). Specifically, we saved the models Γ and Γ^{-1} learned from manipulating an orange cube (obj #4) with 25 initial exploratory actions. Similar to Sec. IV-C, we directly used this saved model to initialize the manipulation of a new object, without any initial exploratory actions on the new object. For each object, we

ran 4 trials for each reference trajectory and averaged the results in Fig. 10. As can be seen from the results, regardless of the specific object, there was no significant increase in manipulation errors even though the manipulation models were learned using a different object. When manipulating a new object, although we observed that more adapting actions were performed for the model update, the total number of exploratory actions was reduced from 25 to 12 since no exploratory actions were needed initially. In summary, the experiment has shown good generalizability of the self-identified models, and efficient utilization of model transfer.

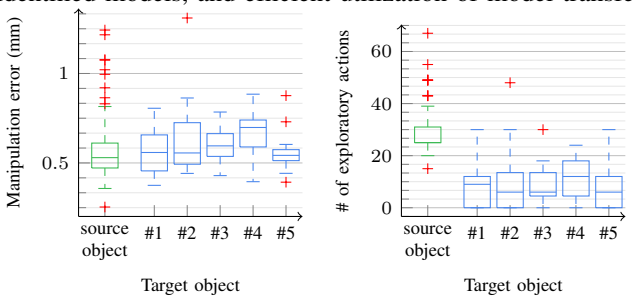

Fig. 10: Results of model transfer experiments. The model was self-identified on a source object (green) and then used for manipulating different other target objects (blue).

VII. Conclusion

In this paper, we approached the in-hand dexterous manipulation problem with non-parametric self-identification and Model Predictive Control. With a small number of exploratory actions, our proposed framework efficiently self-identifies the underlying manipulation models through Gaussian Process Regression. By integrating the self-identified manipulation models into an MPC-based framework, a robust control method can be developed for precise manipulation. Furthermore, when the self-identified local models become unreliable for generating effective controls, our framework can adaptively update the models by performing more exploratory actions.

With extensive real-world experiments on an underactuated Yale Model O hand, we show that 1) our proposed framework can achieve millimeter-level in-hand manipulation without requiring a large amount of data nor sophisticated sensing systems; 2) the MPC-based control is robust when integrated with imperfect manipulation models, which are locally approximated; 3) the self-identified manipulation models can well generalize on similar setup variations.

In future work, we aim to implement the proposed framework on a more complex or higher-dimensional system, for example, an underactuated hand with more motors or a compliant robot arm with higher degrees of freedom. Furthermore, we plan to improve the current framework, to achieve higher precision and dexterity of in-hand manipulation by a more capable control policy; or to enhance the adaptability of self-identification to different types of motion constraints.

References

[1] A. M. Okamura, N. Smaby, and M. R. Cutkosky, "An overview of dexterous manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 1, 2000, pp. 255–262.

[2] I. Kao, K. M. Lynch, and J. W. Burdick, "Contact modeling and manipulation," in *Springer Handbook of Robotics*. Springer, 2016, pp. 931–954.

[3] K. Hang, M. Li, J. A. Stork, Y. Bekiroglu, F. T. Pokorny, A. Billard, and D. Kragic, "Hierarchical fingertip space: A unified framework for grasp planning and in-hand grasp adaptation," *IEEE Transactions on Robotics*, vol. 32, no. 4, pp. 960–972, 2016.

[4] V. Kumar, E. Todorov, and S. Levine, "Optimal control with learned local models: Application to dexterous manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 378–383.

[5] W. Yuan, S. Dong, and E. Adelson, "Gelsight: High-resolution robot tactile sensors for estimating geometry and force," *Sensors*, vol. 17, no. 12, p. 2762, 2017.

[6] J. Bütepage, S. Cruciani, M. Kokic, M. Welle, and D. Kragic, "From visual understanding to complex object manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, no. 1, 2019.

[7] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with large-scale data collection," in *International Symposium on Experimental Robotics*. Springer, 2016, pp. 173–184.

[8] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4238–4245.

[9] K. Hang, W. G. Bircher, A. S. Morgan, and A. M. Dollar, "Manipulation for self-identification, and self-identification for better manipulation," *Science Robotics*, vol. 6, no. 54, 2021.

[10] B. Sundaralingam and T. Hermans, "Geometric in-hand regrasp planning: Alternating optimization of finger gaits and in-grasp manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 231–238.

[11] M. Li, K. Tahara, and A. Billard, "Learning task manifolds for constrained object manipulation," *Autonomous Robots*, vol. 42, no. 1, pp. 159–174, 2018.

[12] B. Calli and A. M. Dollar, "Robust precision manipulation with simple process models using visual servoing techniques with disturbance rejection," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 1, pp. 406–419, 2019.

[13] M. V. Liarokapis and A. M. Dollar, "Post-contact, in-hand object motion compensation with adaptive hands," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 456–467, 2016.

[14] L. P. Kaelbling, "The foundation of efficient robot learning," *Science*, vol. 369, no. 6506, pp. 915–916, 2020.

[15] N. Fazeli, S. Zapolsky, E. Drumwright, and A. Rodriguez, "Learning data-efficient rigid-body contact models: Case study of planar impact," in *Conference on Robot Learning*. PMLR, 2017, pp. 388–397.

[16] J. Bohg, K. Hausman, B. Sankaran, O. Brock, D. Kragic, S. Schaal, and G. S. Sukhatme, "Interactive perception: Leveraging action in perception and perception in action," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1273–1291, 2017.

[17] M. C. Koval, N. S. Pollard, and S. S. Srinivasa, "Pose estimation for planar contact manipulation with manifold particle filters," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 922–945, 2015.

[18] R. Platt, L. Kaelbling, T. Lozano-Perez, and R. Tedrake, "Efficient planning in non-gaussian belief spaces and its application to robot grasping," in *International Symposium on Robotics Research*, 2017, pp. 253–269.

[19] L. U. Odhner, L. P. Jentoft, M. R. Claffee, N. Corson, Y. Tenzer, R. R. Ma, M. Buehler, R. Kohout, R. D. Howe, and A. M. Dollar, "A compliant, underactuated hand for robust manipulation," *The International Journal of Robotics Research*, vol. 33, no. 5, pp. 736–752, 2014.

[20] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 3400–3407.

[21] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *2015 International Conference on Advanced Robotics (ICAR)*, 2015, pp. 510–517.