Adaptive Fingers Coordination for Robust Grasp and In-Hand Manipulation under Disturbances and Unknown Dynamics

Farshad Khadivar[1] and Aude Billard[1]

Abstract—We present a control framework for achieving a robust object grasp and manipulation in hand. In-hand manipulation remains a demanding task as the object is never stable and task success relies on carefully synchronizing the fingers' dynamics. Indeed, fingers must simultaneously generate motion while maintaining contact with the object and, by staying within the hand's frame, ensuring that the object remains manipulable. These challenges are exacerbated once the hand gets disturbed or when the internal dynamics of the manipulated object are unknown, such as when it is filled with liquid moving during manipulation. We present a control strategy based on coupled dynamical systems, whereby the fingers move in synchronization using an intermediate dynamics responsible for coordinating fingers. To adapt to changes in forces due to model uncertainties and unexpected disturbances, we employ an adaptive torque-controller combined with a joint impedance regulator that guarantees high tracking accuracy while adapting to dynamic changes. We validate the approach in multiple experiments on a 16 degrees-of-freedom robotic hand grasping and manipulating objects with different mass properties, e.g., uneven or varying mass distribution in a glass half-filled with water. We show that the robot can compensate for disturbances generated by internal dynamics and external perturbations. Additionally, we showcase how our controller, in conjunction with learning from human demonstration, provides a robust solution for more complicated manipulations such as finger gaiting.

Index Terms—Dexterous Manipulation, Grasping, Coupled Dynamical Systems, Finger Gating, Robust/Adaptive Control of Robotic Systems, Object with uneven mass distribution.

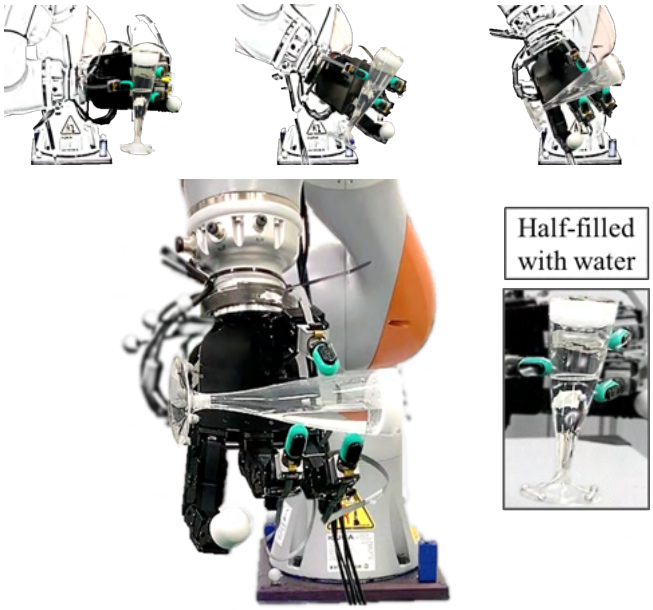

Fig. 1: Rotating a half-full champagne glass, an object with varying mass distribution. The glass is first grasped in a vertical pose (top left) then rotated to reach a stable horizontal pose (central). The glass is stabilized by optimizing the object-level grasp-impedance and by extracting contact frames from tactile sensors at contact points. Contact forces of all fingers are synchronously modulated based on a coupled dynamical system and an adaptive control scheme.

I. Introduction

IN-HAND manipulation is the ability to re-position an object within a hand. Grasping and manipulating objects of more variety and complexity is best explored when using a robotic hand with multiple degrees of freedom [1]. Although securely grasping the object is an essential ingredient required prior to in-hand manipulation [2], the main challenges arise after the object is grasped. The hand must carefully re-position the object by synchronously modulating the object-finger contacts/interactions [3]. Performing this in a robust fashion is never easy and becomes even more convoluted in presence of uncertain and inaccurate estimation of the system's mechanical properties [4]. In this paper, we address the problem of ensuring robust in-hand manipulation when faced with a poor model of the object's dynamics, model imperfections, and external disturbances. Next, we describe (i) our approach to addressing this problem, (ii) the main parts of our control framework and their novelty over existing ones, (iii) the experimental validation procedure, and (iv) summarize our contributions.

Coordination of fingers and adaptation of torques for robust manipulation: We hypothesize that robustness relies on carefully synchronizing the fingers' dynamics and offer a novel control strategy based on coupled dynamical systems (DSs), combined with an adaptive torque-controller for providing live adaptation of the position and force generated by the fingers to stabilize the object. Consider the example of tilting a glass half-filled with

[1]Farshad Khadivar and Aude Billard are both with Learning Algorithms and Systems Laboratory (LASA), École Polytechnique Fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland. farshad.khadivar@epfl.ch and aude.billard@epfl.ch

This article has supplementary videos providing a summary of the paper's experiments and applications

This work was supported by the European Research Council (ERC), Advanced Grant agreement No 741945, Skill Acquisition in Humans and Robots

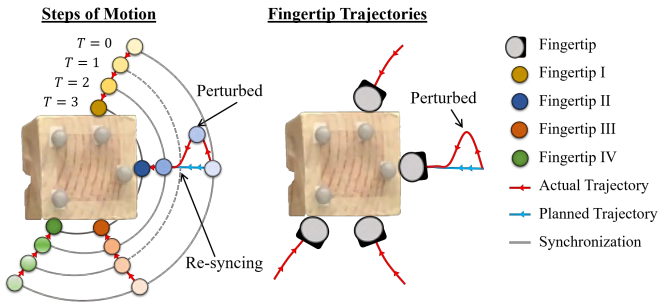

Fig. 2: An example of synchronized grasp. All fingers are coordinated to move together. If one finger (here fingertip II) is perturbed anytime during motion, the others wait for it to re-synchronize ($T = 0$ to $T = 2$), rather than retracting or deviating from the intended trajectory. The perturbed finger, also, accelerates to synchronize while recovering the planned trajectory and approaching the object with correct orientation.

water, Figure 1. Here, moving fingers in coordination is crucial. If one finger moves faster than the others, i.e., with asynchronous displacement, the water in the glass consequently moves faster, and the object gains momentum, thus rendering the grasp hard to control. Furthermore, adaptation here is equally critical. As the liquid spreads in the glass, the hand must re-balance the force to compensate for the changes in the mass distribution. Assume further that the hand dynamics is only partially known; in that case, the controller must compensate for the unknown dynamics and prevent the object from veering away from the desired trajectory.

Coordinated Task Planning: the use of coupled DSs to synchronize multiple robots' dynamics is not new [5, 6, 7, 8]. Here, we extend this concept to in-hand manipulation. The challenge is using coupled DS to enable the hand to produce the desired motion on the object. We achieve such planning for fingers' synchronization by introducing a novel approach that controls the coupling of DSs of multi-limbs. More precisely, we design an intermediate dynamics that regulates the relative speed of the fingers, so as to accelerate or decelerate their motion. Through this intermediate DS we ensure that fingers apply, in coordination, the desired motion and force on the object. We obtain a closed-form expression that enables automatic and synchronous re-planning and repositioning of the fingers in response to changes in the object's pose. Compared to coupling DS with a virtual reference trajectory [7] or using temporal scaling dynamic [8], our intermediate DS coupling method ensures, in addition to being robust to disturbances to each finger, that all the fingers follow a specific desired trajectory, leading the object to the desired pose. For instance, though all fingers are synchronized to move together, if one finger is disturbed, the others wait for the perturbed finger to re-synchronize, rather than retracting or deviating from the intended trajectory; see Figure 2 for an illustration.

Control Effort Adaptation: We propose a joint-level adaptive torque controller to track the fingers' desired trajectory generated by the DS and to regulate joint-impedance gains. The adaptation of the torques online is key to mitigating uncertainties arising from poorly modeled robot dynamics, object-finger interactions, and the object's physical properties (change in mass distribution). The concept of adaptive controllers [9] per se is well known in robotics. From the rich pool of adaptive controllers, we borrow a model reference adaptive control (MRAC) [10] and tailor it for our coordinated task-planning method. To the best of our knowledge, our work is the first to utilize MRAC as low-level torque control for grasp and manipulation, in a real robotic setup. Furthermore, to better suit the controller for the task at hand, we embed an impedance regulation to the adaptive control to avoid the well-known issue of gains saturation.

Stability and Robotic Evaluation: We analyze the closed-loop system and show that: i) the coupled DS is asymptotically stable to the desired object's pose; ii) convergence of the error of the adaptive torque-control; iii) under no disturbances, the system provides torques to ensure force-closure for the grasp.

We evaluate the developed controller in four real-world robotic experiments: (a) comparing with control baseline and testing the hand controller in tracking tasks, (b) assessing the grasp adaptation with multiple objects of different dynamic properties, (c) for the same object, performing rotational and translational in-hand manipulation, and (d) rolling a cuboid in-hand by finger-gaiting. We perform the latter to demonstrate the strength of our approach in a challenging in-hand manipulation task. Rotating an object in hand is referred to as in-hand rolling that requires continuous finger gaiting [11]. For the most recent attempt at this task, researchers relied on learning the sequences of finger motion through reinforcement learning closing the loop in vision and haptic feedback [12, 13, 14]. In this work, we use the concept of learning human demonstrations to extract the coupling and the sequence across fingers motion. We show that embedding this into our controller offers immediate robustness to the above disturbances. Note that we perform various experiments, yet only the planning part of our framework differs among all the tests. Due to the adaptation properties, the low-level control is fixed across all experiments, reducing substantially the amount of engineering.

Contributions: In this paper, we focus on the problem of robustness in grasp and in-hand manipulation. Our main contributions are

(i) introducing a novel intermediate DS coupling method to achieve coordinated finger motions.
(ii) tracking fingers' desired velocity using a joint-level adaptive torque controller. We regulate joint impedances to realize real-world grasp and manipulation tasks.

Additionally, we show that this approach can be combined with learning from human demonstration to learn a desired trajectory for the object and identify the corresponding sequence of desired motion for the fingers. We further show that learning can be used to identify roles for

each group of fingers and embed this in the intermediary coupling.

Paper Structure: The core of the paper consists of four parts: In Section IV, we describe in detail the proposed method for coordinating fingers. In Section V, we close our control approach by presenting the adaptive control system with impedance regulation. In Section VI, we explain how this controller is used in grasp and manipulation problems. In Section VII, we extensively evaluate and discuss the strength of our approach in real-world robotic experiments.

II. Related Work

A variety of dexterous manipulation problems have received attention in the past, including regrasping [15, 16], slide-rolling [17, 18], pivoting [19, 20], caging [21] , and finger gaiting [22, 23, 24]. In in-hand manipulation, the challenges with real-world execution, the complexity of task planning, and the inefficiency of feedback control law are primarily due to imprecise knowledge of object properties, contact mechanics, and the robot's dynamics [25]. In this respect, considerable efforts have been put into designing control systems based on either hybrid force/position scheme or varying-impedance modulation; see a thorough review by Ozawa and Tahara [26]. Still, synchronizing fingers and adapting to uncertainties are among the unexplored aspects of control design.

Planning Grasp and Manipulation: Among experimentally validated methods, Shi et al. [27] proposed to control the fingers' acceleration in four time-steps to control slippage time and relative sliding displacement. Deriving acceleration profiles for non-laminar objects, however, is not trivial due to angular kinetics and dynamics. Sundaralingamand and Hermans [28] focused on high-level task planning for in-hand manipulation. They combined the concept of finger gaiting and incremental manipulation, i.e., after each increment of manipulation, one or two fingers update their position to prepare for the next interval. Planning is mainly built on knowledge of an object's shape, which challenges this approach for real-world implementation. Inspired by previous efforts on bimanual manipulation [7, 29], we propose a coupled dynamical system for synchronizing object and finger motions. Our coupling is designed to best benefit grasp and manipulation tasks in that it remains robust and reactive in the face of disturbances.

Control for Grasp and Manipulation: Modulating the finger impedance based on object kinetics is another method for realizing the desired manipulation task [30]. Pfanne et al. [31] control the grasp impedance and plan the manipulation in the null space of the grasp, where the existence of null space needs to be ensured by optimizing the grasp configuration [32]. Besides, accurate grasp matrix estimation [33] becomes critical in this type of control. Alternatively, tactile sensing is used to estimate the local geometry and to provide feedback for adapting contact impedance [34]. For instance, Sodhi et al. [35] used a factor graph with a digit tactile sensor for estimating the states of the object. Calandra et al. [36] trained a deep convolutional network for predicting the quality of a grasp from GelSight tactile sensors. Likewise, in our study, we learned a model to estimate contact frames from BioTac tactile sensors and used this online frame estimation for grasp optimization under various objects and shape uncertainties.

Data Driven Methods: Inaccuracy in sensing and the complexity in modeling object properties are other challenges. This has encouraged researchers to move to data-driven algorithms that, as an alternative, use approximate models, such as model-based [37] or model-free [13, 14, 38] reinforcement learning. However, for such learning algorithms, where the need for a perfect simulated environment is substantial, extracting appropriate features of physical interaction is a significant challenge that leads to all the issues related to sim-to-reality and vice versa. Furthermore, data-driven approaches cannot cope easily with perturbations unless they have seen these as examples during training. In this work, a combination of model-based and learning approaches enables us (a) to estimate the dynamics of the object-finger forces, which might not be easy to determine, and (b) to control explicitly for different perturbations, e.g., changes in the position of the object.

Limited Work-Space of a Hand: One common issue with in-hand manipulation is to ensure that the object remains within the reach of each finger. Dafle et al. [1] took advantage of gravity by throwing and re-grasping objects, and they extended their work later in [39] and [40]. Likewise, Bai and Liu [41] control the tilt angle of the palm, based on energy conservation, to let the object roll. Dynamic forces on the object, if known precisely, can be utilized to manipulate the object as, for instance, in an impressive demo by Furukawa et al. [42], in which the hand re-grasps a foam cylinder object by tossing it in the air and catching it. This type of manipulation largely relies on the dynamic properties and, in particular, on the object's moment of inertia. Regrasping becomes nearly impossible, once the distribution of the object's mass is non-uniform or varying. The majority of the previous studies considered one-step in-hand manipulation [27, 30, 31]. Here, we learn manipulation steps from human demonstrations and extract finger roles accordingly.

III. Approach

We perform trajectory planning in task space and compute control commands in joint space. For planning, we devise our controller on the basis of dynamical systems (DSs) as DS-based controllers are robust to perturbations [43]. More precisely, in our high-level planning, we express the task as a desired velocity $\dot{\boldsymbol{x}}_i^d \in \mathbb{R}^{d_x}$, with d_x being the dimension of position in task space, for each i-th fingertip following a DS of the form

$$\dot{\boldsymbol{x}}_i^d \;=\; f_i(\boldsymbol{x}_i) \tag{1}$$

$f_i(\cdot)$ is a continuous and asymptotically stable DS at an attractor $\boldsymbol{x}_i^* \in \mathbb{R}^{d_x}$ meaning that $\|\boldsymbol{x}_i - \boldsymbol{x}_i^*\| \to 0$ as $t \to \infty$.

The function f can be linear, nonlinear, or cyclic, and it can hold and generate all possible trajectories to reach the target state [44].

In low-level control of a robot with n_j joints, we can set the joint positions, $\boldsymbol{q} \in \mathbb{R}^{n_j}$, or send joint torques, $\boldsymbol{\tau} \in \mathbb{R}^{n_j}$, as the control command. In practice, although position control is intuitive to implement, it relies on high gains that can, when contact arrives earlier than planned, damage both the robot's joints and the object. In grasp/manipulation applications, a precise knowledge of object shape and robot model (i.e., an accurate inverse dynamics models [45]) is needed when using position control. Controlling joint torques is more suited to establishing compliant object-finger interactions, which leaves room for adaptation to underlying imperfections or model imprecision. As a result, we follow a torque-based control approach for the low-level robot control:

$$\boldsymbol{\tau}_i \;=\; u_i(\dot{\boldsymbol{x}}_i^d, \boldsymbol{q}_i, \dot{\boldsymbol{q}}_i) \tag{2}$$

where $\boldsymbol{\tau}_i \in \mathbb{R}^{n_j}$ is the joint torques computed to realize the desired fingertip velocity $\dot{\boldsymbol{x}}_i^d$ from Eq.1. Our objective in low-level control is to define the function $u_i(\cdot)$ for each i-th finger. With $u_i(\cdot)$, we adapt explicitly the joint compliance to allow handling uncertainties about the object's shape, mass distribution, and contact mechanics.

IV. Finger Synchronization based on DS

We begin with a measure of relative distance variable and its intermediate DS. We derive the task-space planner for each fingertip and describe how a single robot can be controlled with this intermediate DS. Next, we explain how multiple fingers are coupled through intermediate DSs, and how this results in coordination of fingers suitable for grasp and manipulation.

A. Intermediate Dynamic

Let's consider a single robot end-effector (a single fingertip in our case). The task is to control this end-effector to reach a desired position $\boldsymbol{x}^d \in \mathbb{R}^{d_x}$. An additional requirement is that $\boldsymbol{x}^d$ needs to be on the connecting line between two specific points in the task space: $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)} \in \mathbb{R}^{d_x}$. These two points (see Section IV-C2) are used for formulating both grasp and manipulation tasks. But, for now, let's assume that they are externally defined. Given this, the desired position of the robot, $\boldsymbol{x}^d$, can be expressed as a form of a relative distance measure:

$$\boldsymbol{x}^d = \boldsymbol{x}^{(2)} - z^d(\boldsymbol{x}^{(2)} - \boldsymbol{x}^{(1)}) \tag{3}$$

where $z^d \in \mathbb{R}_{[0,\,1]}$ is a variable representing the relative distance of the desired position, $\boldsymbol{x}^d$, from the second point $\boldsymbol{x}^{(2)}$. Instead of directly fixing z^d for the robot to track, we set an intermediate target, $\boldsymbol{x}^* \in \mathbb{R}^{d_x}$, defined by $z \in \mathbb{R}_{[0,\,1]}$:

$$\boldsymbol{x}^* = \boldsymbol{x}^{(2)} - z(\boldsymbol{x}^{(2)} - \boldsymbol{x}^{(1)}) \tag{4}$$

then regulating z to converge to z^d. This intermediate variable will later serve as the coupling variable. Determining and controlling z is the core of our coordinated planning problem. Regardless of the actual robot position, z is needed to evolve based on a dynamics that ensures $\|z^d - z\| \to 0$ as $t \to \infty$, hence we propose the following DS:

$$\dot{z} = g(z, z^d) = -\beta_1 \frac{1 - e^{-\beta_2(z - z^d)}}{1 + e^{-\beta_2(z - z^d)}} \tag{5}$$

both $\beta_1 \in \mathbb{R}^+$ and $\beta_2 \in \mathbb{R}^+$ are positive scalar values gearing the convergence behavior.

Theorem 1: The DS given by (5) asymptotically converges to z^d i.e., $\lim_{t\to\infty} \|z^d - z\| = 0$.

Proof: See Appendix.A-B.

Remark 1: For any input value $z' \in \mathbb{R}_{[0,\,1]}$ to DS (5), we can find z by integrating for one time step δ, i.e., $z = g(z', z^d)\delta + z'$, which sets the next intermediate target point, $\boldsymbol{x}^*$ (Eq.4) for the robot to follow.

B. Task-Space Dynamical System

1) State-Dependent Target Point: Altering the variable z is equivalent to changing the target position, $\boldsymbol{x}^*$, for the robot. Accordingly, the attractor of the task-space DS is state-dependent, which we need to consider in DS derivation for the robot end-effector. Given the robot end-effector position $\boldsymbol{x} \in \mathbb{R}^{d_x}$, variable z, and the intermediate DS (5), we propose the following task-space DS to control a robot end-effector:

$$\begin{aligned}\dot{\boldsymbol{x}}^d = f(\boldsymbol{x}, z) = &-\left(z\mathbf{A}_x + \dot{z}\mathbf{I}_{d_x}\right)\left(\boldsymbol{x}^{(2)} - \boldsymbol{x}^{(1)}\right) \\ &- \mathbf{A}_x\left(\boldsymbol{x} - \boldsymbol{x}^{(2)}\right)\end{aligned} \tag{6}$$

$\mathbf{A}_x \in \mathbb{R}^{d_x \times d_x} \succ 0$ is the state matrix, and $\mathbf{I}_{d_x}$ is the identity matrix.

Theorem 2: The task-space DS given by (6) is asymptotically stable at the target point $\boldsymbol{x}^*$, i.e., $\lim_{t\to\infty} \|\boldsymbol{x}^* - \boldsymbol{x}\| = 0$.

Proof: See Appendix.A-C.

DS (5) is asymptotically stable at z^d meaning that $\lim_{t\to\infty} \|\boldsymbol{x}^d - \boldsymbol{x}^*\| = 0$, and DS (6) asymptotically converges to $\boldsymbol{x}^*$. Therefore, given Theorem 1 and Theorem 2, DS given by (6) is stable at the desired point $\boldsymbol{x}^d$, i.e., $\lim_{t\to\infty} \|\boldsymbol{x}^d - \boldsymbol{x}\| = 0$. Note that the stability of DS (6) at $\boldsymbol{x}^d$ is not necessarily asymptotic, see Section IV-B3.

2) Feedback for Intermediate Variable: We obtain z by providing the DS (5) with the state feedback from the robot (see Remark 1). First, we compute the actual relative distance from the current position of the robot. To better distinguish the actual relative distance from the DS variable z, let's use α instead of z'. Let α be a measure of the current relative distance:

$$\alpha = \frac{\left(\boldsymbol{x}^{(2)} - \boldsymbol{x}\right)^T \left(\boldsymbol{x}^{(2)} - \boldsymbol{x}^{(1)}\right)}{\|\boldsymbol{x}^{(2)} - \boldsymbol{x}^{(1)}\|^2} \tag{7}$$

then we can estimate z by one-step integration of the intermediate DS (5), $z = g(\alpha, z^d)\delta + \alpha$, for one time step δ. In this way, we use the intermediate DS in closed-loop control.

3) Overall DS for One Fingertip: The overall control loop of one fingertip finds the desired velocity, $\dot{\boldsymbol{x}}^d$, using the following DSs combined:

$$\begin{cases} \dot{z} = g(\alpha, z^d) \\ \dot{\boldsymbol{x}}^d = f(\boldsymbol{x}, z) \end{cases} \tag{8}$$

Different combination of these two DSs leads to different linear and nonlinear convergence behavior. Figure 3 illustrates $2D$ examples of the vector field for $\dot{\boldsymbol{x}}^d$ by using (8) where the intermediate dynamic, $g(\alpha, z^d)$, has a lower, an equal, and a higher convergence speed, compared to the robot task-space dynamic, $f(\boldsymbol{x}, z)$. If a fingertip is perturbed while traveling from $\boldsymbol{x}^{(1)}$ to $\boldsymbol{x}^d$, from Figure 3, the slower intermediate DS will bring the finger back to the connecting trajectory of $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$. We take advantage of this property (by choosing $\mathbf{A}_x = 5\mathbf{I}_{d_x}$, $\beta_1 = 1$, and $\beta_2 = 5$), both in grasp and manipulation planning: during grasping phase to ensure that the fingertips approach the object in the correct direction and, during manipulation, to reduce the deviation from the desired contact velocities.

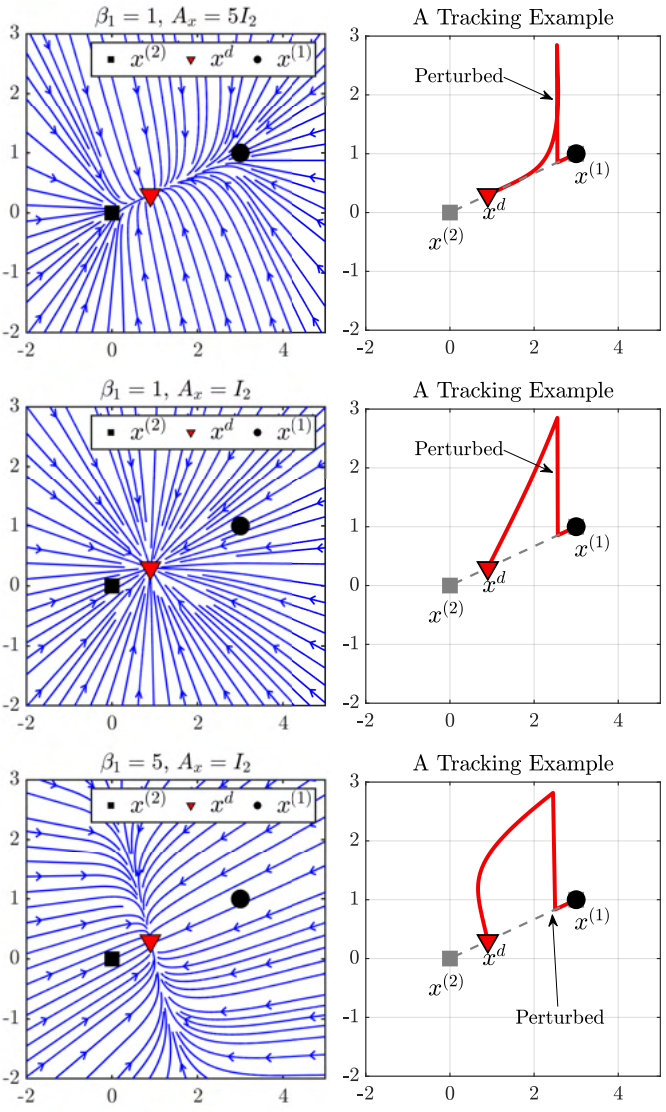

Fig. 3: Examples of the vector field of the desired velocity, $\dot{\boldsymbol{x}}^d$,(right column) generated by DS (8), and examples of a perturbed tracking (left column). For each row of examples respectively, the cases illustrate the intermediate dynamic, $g(\alpha, z^d)$, that has a lower (top row), an equal (middle row), and a higher (bottom row) convergence speed, compared to the robot task-space dynamic, $f(\boldsymbol{x}, z)$. $\beta_2 = 5$ in all cases.

C. DS Coupling and Fingers Coordination

1) Coupling: To couple n_f fingertips, each controlled by DS (8), we define the intermediate DS for the i-th fingertip, $i = 1, \cdots, n_f$:

$$\begin{cases} \dot{z}_i = g(\alpha_i, z^c) \\ \dot{\boldsymbol{x}}_i = f(\boldsymbol{x}_i, z_i) \end{cases} \tag{9}$$

where z^c acts as the coupling variable, and is:

$$z^c(\alpha_1, ..., \alpha_{n_f}, z^d) = \frac{1}{n_f + 1}(z^d + \sum_{i=1}^{n_f} \alpha_i). \tag{10}$$

z^c couples the entire system and captures the status of all fingers conveyed through the measurement of their actual relative distances $\{\alpha_i\}_{i=1}^{n_f}$.

Theorem 3: The coupled system (9) for n_f fingers converges to a stable state at $\boldsymbol{x}_i^d$ for $i = 1, \cdots, n_f$, i.e., for each i-th fingertip as $t \to \infty$, $\|z^d - z_i\| \to 0$, and consequently, $\|\boldsymbol{x}_i^d - \boldsymbol{x}_i\| \to 0$.

Proof: See Appendix.A-D.

Figure 4 illustrates an example of three coupled dynamical systems (9), where the robot end-effectors are initiated from the same position, $\boldsymbol{x}_1(t = 0) = \boldsymbol{x}_2(t = 0) = \boldsymbol{x}_3(t = 0)$, and synchronously converge to their respective desired positions. From Figure 4 we also observe that in $1k$ random cases of initial positions and z^d, eventually $\|\alpha_i - \alpha_j\|_{i \neq j} \to 0$, $\|\alpha_i - z^c\| \to 0$, and $\|z^c - z^d\| \to 0$.

2) Defining $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$: The position of states $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ shapes the desired vector field and determines the behavior of the system in face of perturbation (see Figure 3). Defining where to place these two points is the core to our planning approach for grasp (see Section.VI-A) and manipulation (see Section.VI-B2). $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ set the positions, respectively, where a task begins and terminates. For instance, consider an object manipulated by three fingers. The task is to rotate the object on a plane, see Figure 5. In such a case, fingers are required to move synchronously throughout the entire trajectory. For all fingers, $\{\boldsymbol{x}_i^{(1)}\}_{i=1}^{n_f}$ is the initial fingertip configuration, and $\{\boldsymbol{x}_i^{(2)}\}_{i=1}^{n_f}$ is the final one, after a full rotation.

3) Robustness of DS: A key concern in our approach, which we assess in several parts, is robustness to uncertainties and perturbations. Our coupling variable (Eq.10) is formulated in such a way that, if a finger is perturbed, the other fingers will wait instead of retracting, see Figure 6. This feature of coupling is critical for grasp and manipulation, as a perturbation on one finger should not cause other fingers to break contact and leave the object's surface. We also show that reaction to perturbation is not arbitrary and that a fingertip will restore the intended trajectory

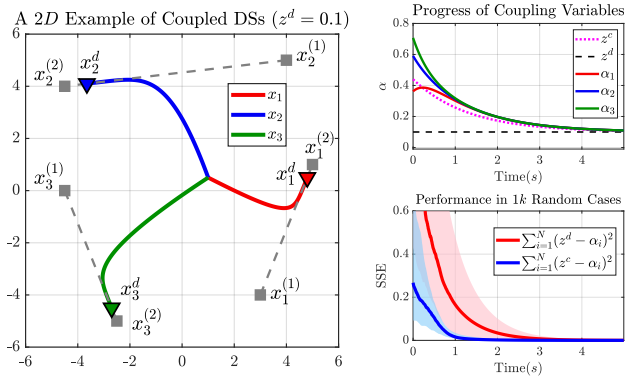

Fig. 4: A coupled system of three robot end-effectors controlled by DS (9). The left figure demonstrates a tracking example, where all fingers start at the same initial position and are needed to reach their respective desired position $\boldsymbol{x}_i^d$, and where the top right figure is the progress of the corresponding coupling variables for this example. The bottom right figure shows the sum of square errors (SSE) in tracking the coupling relative variable z^c and the desired relative variable z^d in $1k$ cases of random initial positions and z^d. The solid lines are the median, and the shaded area displays 25-75 percentiles over $1k$ cases.

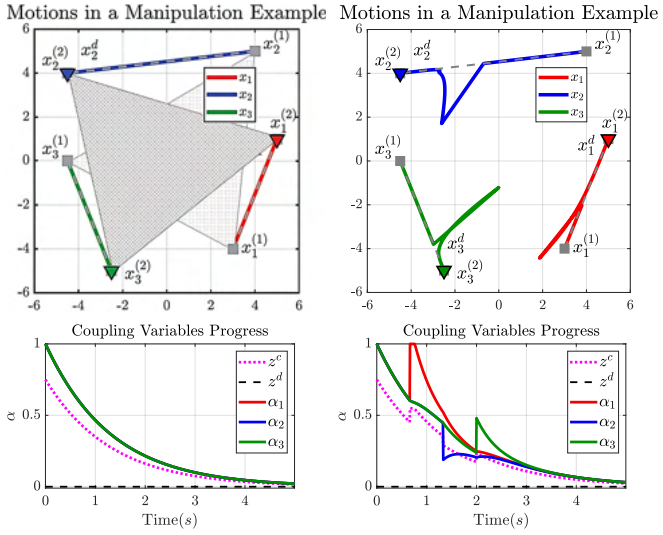

Fig. 5: An example of planning for manipulation where an object is first grasped in positions $\boldsymbol{x}_1^{(1)}$, $\boldsymbol{x}_2^{(1)}$, and $\boldsymbol{x}_3^{(1)}$. Then the fingers are tasked to rotate the object within the hand frame by simultaneously moving to $\boldsymbol{x}_1^{(2)}$, $\boldsymbol{x}_2^{(2)}$, and $\boldsymbol{x}_3^{(3)}$. Figures on the left show the ideal case, and figures on the right depict the same examples where the fingers react to perturbation at different times.

(Figure 4), which is critical in grasp and manipulation.

V. Joint-Space Adaptive Controller

A. Low-Level Control

We estimate online the function $u_i(\cdot)$ in Eq.2. The objective of control law $u_i(\cdot)$ is to compute a set of joint torques $\{\boldsymbol{\tau}_i\}_{i=1}^{n_f}$ that realizes the desired fingertip velocities $\{\dot{\boldsymbol{x}}_i^d\}_{i=1}^{n_f}$ given by DSs (9). As the control laws for all fingers are derived in a similar fashion, for better readability, we drop the finger-dependent indices i for each i-th finger in this section.

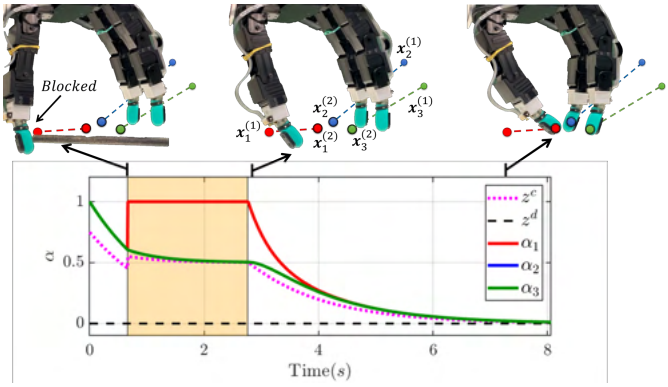

Fig. 6: Response of the coupled multi-fingered system to a disturbance in grasp planning. Whereas the first fingertip (thumb) is blocked for a couple of seconds, the other fingers wait (the shaded area) for the thumb to rejoin them.

We perform control and adaptation in the joint space, as each finger joint differs in its mechanical imperfections and physical properties hence requires different adaptation laws. We propose a group of adaptive laws for learning and controlling the system's dynamics under uncertainties. We first introduce a nominal dynamics to be the reference model. Then, the controller is tasked to ensure that the actual dynamics of joints perfectly follows this reference model.

B. Nominal Joint-Space Dynamics

Let's express the controllable states of the system, $\boldsymbol{\zeta} = [\boldsymbol{q}^T, \dot{\boldsymbol{q}}^T]^T \in \mathbb{R}^{2n_j}$, consisting of the joint position, $\boldsymbol{q} \in \mathbb{R}^{n_j}$, and joint velocity, $\dot{\boldsymbol{q}} \in \mathbb{R}^{n_j}$ for each i-th finger with n_j number of joints. $\boldsymbol{\zeta}$ is needed to follow a reference vector $\boldsymbol{\zeta}_r$ that evolves based on a nominal dynamics of the form:

$$\dot{\boldsymbol{\zeta}}_r = \mathbf{A}_r \boldsymbol{\zeta}_r + \mathbf{B}_r \boldsymbol{r}(\dot{\boldsymbol{x}}^d), \tag{11}$$

in which $\mathbf{A}_r \in \mathbb{R}^{2n_j \times 2n_j} \succ 0$, and $\mathbf{B}_r \in \mathbb{R}^{2n_j \times n_j}$, $rank(\mathbf{B}_r) = n_j$, are designed to shape the reference model. They are set such that Eq.11 is asymptotically stable at a desired state $\boldsymbol{\zeta}^d$. Matrix $\mathbf{A}_r$ controls how fast $\boldsymbol{\zeta}_r$ converges to $\boldsymbol{\zeta}_i^d$ while $\mathbf{B}_r$ regulates $\boldsymbol{\zeta}_r$ to track $\boldsymbol{\zeta}^d$. Moreover, $\boldsymbol{r}(\dot{\boldsymbol{x}}^d) \in \mathbb{R}^{n_j}$ is a bounded regulation signal.

Remark 2: $\boldsymbol{r}(\dot{\boldsymbol{x}}^d)$ links the desired fingertip velocity $\dot{\boldsymbol{x}}^d$ (Eq.1 and Eq.9) to the joint-space desired state $\boldsymbol{\zeta}^d$. Appendix.B-A provides further details on how to compute $\boldsymbol{r}(\dot{\boldsymbol{x}}^d)$ from $\dot{\boldsymbol{x}}^d$.

For brevity, henceforth we drop the dependency of $\boldsymbol{r}$. The nominal dynamics takes the desired velocity from DS (Eq.9) as input; however, note that the concept of nominal dynamics and actual dynamics differ from DSs explained in task planning, Section IV. Here, we shape the dynamic behavior of each joint, regardless of the desired task (target) at hand. As a result, the controller that we develop in this section can be used with other planning approaches as well.

C. Control Rule and Adaptive Laws

The governing dynamics of each finger can be represented by a DS [46] of the form:

$$\dot{\boldsymbol{\zeta}} = \mathbf{A}\boldsymbol{\zeta} + \mathbf{B}\boldsymbol{\nu}. \tag{12}$$

The control effort $\boldsymbol{\nu}$ corresponds to the joint torques of the i-th finger ($\boldsymbol{\tau}_i$ in Eq.2). Let's assume that $\mathbf{A} \in \mathbb{R}^{2n_j \times 2n_j}$, and $\mathbf{B} \in \mathbb{R}^{2n_j \times n_j}$ are unknown and can vary with time as the evolution of the fingers' states is subjected to uncertainties.

The controller's objective is to ensure that $\boldsymbol{\zeta}$ tracks the reference dynamics model (Eq.11) in presence of nonlinearities and uncertainties. To achieve this, we propose an adaptive mechanism to modulate ν:

$$\boldsymbol{\nu} = \boldsymbol{\Psi}_{\zeta}\boldsymbol{\zeta} + \boldsymbol{\Psi}_{\mathrm{r}}\boldsymbol{r}. \tag{13}$$

The adaptive control gains $\boldsymbol{\Psi}_{\zeta} \in \mathbb{R}^{n_j \times 2n_j}$, and $\boldsymbol{\Psi}_{\mathrm{r}} \in \mathbb{R}^{n_j \times n_j}$ are estimated/trained online to compensate for changes in the dynamics of the fingers and the tracking error $\boldsymbol{e} = \boldsymbol{\zeta} - \boldsymbol{\zeta}_i^d$, using the following update rules:

$$\begin{cases} \dot{\bar{\boldsymbol{\Psi}}}_{\boldsymbol{\zeta}} &= -\Lambda_{\zeta}\, \mathbf{B}_r^T\, \mathbf{P}\, \boldsymbol{e}\, \boldsymbol{\zeta}^T - \mathbf{S}\, \Upsilon_{\boldsymbol{\zeta}}\, \bar{\boldsymbol{\Psi}}_{\boldsymbol{\zeta}} \\ \dot{\bar{\boldsymbol{\Psi}}}_{\mathrm{r}} &= -\Lambda_r\, \mathbf{B}_r^T\, \mathbf{P}\, \boldsymbol{e}\, \boldsymbol{r}^T(t) - \mathbf{S}\, \Upsilon_{\mathbf{r}}\, \bar{\boldsymbol{\Psi}}_{\mathbf{r}} \end{cases} \tag{14}$$

wherein:

$$\mathbf{S}_{jj} = \max\big(0, \boldsymbol{\epsilon}_j - e^{-\frac{e_j^2}{2\sigma^2}}\big)$$
$$\Upsilon_{\boldsymbol{\zeta}jj} = \boldsymbol{\kappa}_j^* \big(1 - \frac{1}{(\bar{\boldsymbol{\Psi}}_{\boldsymbol{\zeta}} \bar{\boldsymbol{\Psi}}_{\boldsymbol{\zeta}}^T)_{jj} + \iota}\big)$$
$$\Upsilon_{\mathbf{r}jj} = \boldsymbol{\kappa}_j^* \big(1 - \frac{1}{(\bar{\boldsymbol{\Psi}}_{\mathbf{r}} \bar{\boldsymbol{\Psi}}_{\mathbf{r}}^T)_{jj} + \iota}\big)$$

where $\Lambda_{\zeta} \in \mathbb{R}^{n_j \times n_j}$, and $\Lambda_r \in \mathbb{R}^{n_j \times n_j}$ are positive definite matrices that tune convergence rate of the adaptive gains, whereas $\mathbf{P} \in \mathbb{R}^{2n_j \times 2n_j} \succ 0$ forms a quadratic Lyapunov function, see Appendix A-A. $\boldsymbol{\kappa}_j^*$ and $\boldsymbol{\epsilon}_j$ are the desired stiffness and the allowed error-tolerance for j-th joint of each finger. $\boldsymbol{\kappa}_j^*$ and $\boldsymbol{\epsilon}_j$ are fixed values defined by the user (we set $\boldsymbol{\kappa}^* = [0.15, 0.12, 0.1, 0.05]$ for each finger, and $\boldsymbol{\epsilon}_j = 0.02$ for all joints in our experiments), and they avoid saturation in control signal $\boldsymbol{\nu}$, as well as in the adaptive gains $\boldsymbol{\Psi}_{\zeta}$, and $\boldsymbol{\Psi}_{\mathrm{r}}$.

Theorem 4: Using the control law given by (13) and the adaptive laws (14), the system (12) asymptotically converges to the nominal dynamics (11), and $\lim_{t\to\infty} \|\boldsymbol{\zeta}^d - \boldsymbol{\zeta}\| \to 0$ if there exist $\mathbf{P} \succ 0$ and $\mathbf{Q} \succ 0$ such that $\mathbf{P}\mathbf{A}_r + \mathbf{A}_r^T\mathbf{P} = -\mathbf{Q}$

Proof: See Appendix B-B

The diagonal matrix $\mathbf{S}$ acts as an activation function for stiffness regulation inside the error bound $\boldsymbol{\epsilon}$; for instance, setting $\boldsymbol{\epsilon}_j = 0$ deactivates this feature of the controller for the j-th joint of the finger. Appendix A-A provides insight into how to design $\mathbf{P}$, $\mathbf{A}_r$, and $\mathbf{B}_r$.

Note that in the first time-step, we initialize adaptive gains, $\boldsymbol{\Psi}_{\zeta}$ and $\boldsymbol{\Psi}_{\mathrm{r}}$, with a guessed value. Then, at the end of each iteration, we update their values via adaptive law in Eq.14. The adaptation rate has to be faster than the actual dynamics. Once a task is completed, we can store the trained $\boldsymbol{\Psi}_{\zeta}$, and $\boldsymbol{\Psi}_{\mathrm{r}}$, and use them as the initial guess for new trials. Algorithm 1 summarizes our control framework that will be later used in grasp and manipulation tasks.

Algorithm 1: Coordinated adaptive torque control for each i-th finger.

Input : $\boldsymbol{x}_i$, $\dot{\boldsymbol{q}}_i$, $\boldsymbol{q}_i$, $\boldsymbol{\eta}_i$, $\boldsymbol{x}_i^{(2)}$
Output : $\boldsymbol{\tau}_i$
Main Parameters : β_1, β_2 (Eq.5), and Λ_{ζ}, Λ_r (Eq.14)

Defining DS Models:
- $\mathbf{A}_x \succ 0$ for DS (Eq.9)
- $\mathbf{A}_r \succ 0$, $\mathbf{B}_r$ $(rank(\mathbf{B}_r) = n_j)$
- $\mathbf{P} \succ 0$ that $\mathbf{P}\mathbf{A}_r + \mathbf{A}_r^T\mathbf{P} = -\mathbf{Q}$ for DS (Eq.11)

Initialization : $\bar{\boldsymbol{\Psi}}_{\boldsymbol{\zeta}}$, $\bar{\boldsymbol{\Psi}}_r$

for $t \leftarrow 0$ to T do
1) $\boldsymbol{x}_i$, $\dot{\boldsymbol{q}}_i$, $\boldsymbol{q}_i \leftarrow$ robot's feedback
2) $\alpha_i \leftarrow$ Eq.7 $\leftarrow \boldsymbol{x}_i$
3) $z^c \leftarrow$ Eq.10 $\leftarrow \{\alpha_i\}_{i=1}^{n_f}, z^d$
4) $\dot{z}_i \leftarrow$ Eq.5 $\leftarrow z^c, \alpha_i$
5) $z_i \leftarrow \max\big(0, \min(1, \dot{z}_i\delta + \alpha_i)\big)$
6) $\dot{\boldsymbol{x}}_i^d \leftarrow$ Eq.6
7) $\boldsymbol{r}$, $\boldsymbol{\zeta}_i^d \leftarrow$ Eq.23
8) $\boldsymbol{\zeta} \leftarrow [\dot{\boldsymbol{q}}_i^T, \boldsymbol{q}_i^T]^T$
9) compute $\boldsymbol{\tau}_i$ from Eq.13
10) update $\bar{\boldsymbol{\Psi}}_{\boldsymbol{\zeta}}$, $\bar{\boldsymbol{\Psi}}_r$ by Eq.14

VI. Grasp and Manipulation

Thus far, we have formulated a robust, stable, and finger-synchronized controller for a robotic hand. Here, we discuss in detail how this control framework is used for grasping and manipulating objects; see Figure 7 for an illustration. First, we optimize contact wrenches to stabilize the object in the current configuration. Then, we build upon previous work [47] using object-level impedance in a virtual frame to find a set of attractors $\boldsymbol{x}_i^{(1)}$ and $\boldsymbol{x}_i^{(2)}$ for each i-th finger (e.g., as shown Figure 5). These attractors are coordinated in the grasping process in order to insert force to the object in a synchronized fashion.

A. Contact Wrench Optimization

1) Grasp Configuration: When grasping an object, we can heuristically estimate a stable grasp configuration, i.e., the combination of finger placements on the object surface. This estimation might be sub-optimal compared to solutions in interesting studies such as [47, 48, 49]. However, an optimal grasp configuration is never stable if contact forces are erroneously applied [50]. In our experiments, grasp configuration is heuristically set (the thumb opposes the index and middle finger). Then, we focus on force adaptation at contact points, whether or not a grasp configuration is optimal.

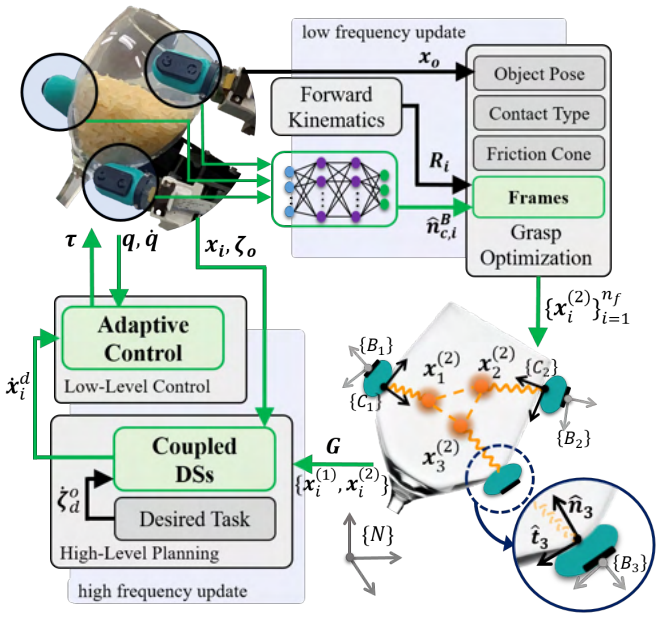

Fig. 7: Block diagram of our method for grasp and manipulation using a coordinated adaptive torque-control. Contact frames are estimated with low frequency by using tactile sensors and the learned neural-network model. These frames are then sent to the grasp optimization algorithm to construct an appropriate object-level impedance (orange elements). Finally, the extracted attractors are fed to the devised controller that runs with high frequency. Components with a green color signify the focus of this study and our contribution to the state of the art.

2) Object-Level Impedance: Once the finger-object contacts are established, each finger must apply a contact force, denoted as $\lambda_i \in \mathbb{R}^3$, to stabilize the object. We utilize contact wrenches to construct an object-level impedance. The following section describes one method for formulating the problem and optimizing the contact wrenches. Since we define the desired grasp stiffness (Section VI-A6), the essential factor for us is the direction of the contact wrenches and their relative magnitude with respect to each other (rather than their absolute values). Notably, alternative methods to formulate the optimization of contact wrenches could also be employed in conjunction with the method described in Section VI-A6.

3) Optimization: We adopt the force closure linear programming-based optimization in [3] and assume planar motion for the object, determining the desired contact wrenches $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1^T, ..., \boldsymbol{\lambda}_{n_f}^T]^T$:

$$\begin{aligned} \text{maximize} \quad & \eta \\ \text{s.t.} \quad & \mathbf{G}\boldsymbol{\lambda} = 0 \\ & \mathbf{F}\boldsymbol{\lambda} - \mathbf{1}_n \eta \geq 0 \\ & \eta \geq 0 \\ & \boldsymbol{e}\boldsymbol{\lambda} \leq \boldsymbol{n}_f \end{aligned} \tag{15}$$

$\eta \in \mathbb{R}^+$ is the optimization variable. $\boldsymbol{e} = [\boldsymbol{e}_1, ..., \boldsymbol{e}_{n_f}]$, and $\boldsymbol{e}_i = [1, 0, 0]$. $\mathbf{F} = diag(\mathbf{F}_1, ..., \mathbf{F}_{n_f})$ is the friction cone matrix, and $\mathbf{G}$ represents the grasp matrix where $\mathbf{F}_i \in \mathbb{R}^{3\times3}$ and $\mathbf{G}_i \in \mathbb{R}^{3\times3}$ are, respectively, the block of $\mathbf{F}$ and $\mathbf{G}$ corresponding to the i-th contact point.

4) Grasp Matrix: The grasp matrix maps contact wrenches to the net wrench (force and moment) on an object. Specifically, in our implementation, it first projects the contact wrenches onto a virtual frame (as described in [47]), and then maps the projected wrenches to the net wrench on the object. The grasp matrix is computed at each time step using fingers' position, contact frames, contact properties, the object's center of mass (see [3]), and the notion of virtual frame [47].

5) Contact Model: For each contact point, the friction law puts constraints on the contact wrench $\boldsymbol{\lambda}_i$, defined by the type of contact (soft/hard finger) and by the friction coefficient μ. Here we consider the hard finger (HF) contact type (no twist/moment at contact point) and the friction to be $\mu = \mu_n - \tan(e_{nn})$ where μ_n is the nominal friction and e_{nn} is the absolute error in contact normal ($\hat{\boldsymbol{n}}_i$) estimation discussed in Section VI-A6 and provided in Table IV.

6) Contact-Frame Estimation: In optimization (15), each contact wrench is expressed in a frame $\{C_i\} = \{\hat{\boldsymbol{n}}_i, \hat{\boldsymbol{t}}_i, \hat{\boldsymbol{o}}_i\}$, which encapsulates the object-shape information at i-th contact point. The unit vector $\hat{\boldsymbol{n}}_i \in \mathbb{R}^3$ is the contact normal, directed toward the object. The other two vectors are orthogonal and lie in the tangent plane of the contact. Moreover, the computation of grasp matrix $\mathbf{G}$ and friction cone matrix $\mathbf{F}$ are based on the contact frames and their positions. We use tactile observation to estimate vector $\hat{\boldsymbol{n}}_i$ of each contact. In essence, we use a mapping from electrodes impedance $\boldsymbol{\rho}_i \in \mathbb{R}^{19}$ of tactile sensor to vector $\hat{\boldsymbol{n}}_i$ at the contact point: $\hat{\boldsymbol{n}}_i = f_{ANN}(\boldsymbol{\rho}_i)$, see Appendix C for details on learning the mapping.

B. Determining Attractors

1) Grasp: With contact wrenches from optimization (15), we set the attractors of the DS (9) as

$$\boldsymbol{x}_i^{(2)} = \boldsymbol{K}_x^{-1}\mathbf{G}_i\boldsymbol{\lambda}_i + \boldsymbol{x}_i^{(1)} \tag{16}$$

$\boldsymbol{x}_i^{(1)}$ is the contact position on the object surface. $\boldsymbol{K}_x$ is the stiffness coefficient chosen according to the desired finger-object compliance [47]. Therefore, given the analysis in Section IV-C3, applying the wrench force will be in a synchronized fashion, which will guarantee a robust grasp within the fingers.

2) In-Hand Manipulation: Once the object is securely grasped, in-hand manipulation is generated by repositioning the set of attractors $\{\boldsymbol{x}_i^{(2)}\}_{i=1}^{i=n_f}$. Attractor repositioning for DS (9) is done using the grasp matrix and the desired object velocity:

$$\boldsymbol{x}_i^{(2)} = \mathbf{G}_i^T\dot{\boldsymbol{\zeta}}_o^d\delta + \boldsymbol{K}_x^{-1}\mathbf{G}_i\boldsymbol{\lambda}_i + \boldsymbol{x}_i^{(1)} \tag{17}$$

here $\boldsymbol{x}_i^{(1)}$ is the contact point, similar to Eq.16. $\boldsymbol{\zeta}_o \in \mathbb{R}^{d_x}$ represents the states of the object; for instance in a planar motion, $\boldsymbol{\zeta}_o = [x_o, y_o, \theta_z]$. $\dot{\boldsymbol{\zeta}}_o^d \in \mathbb{R}^{d_x}$ is the object's desired velocity at each time step which can be also computed

from a DS. Note that the process of defining attractors in manipulation is compatible with grasp, hence the system will remain coupled throughout the entire procedure of first grasping and then manipulating the object. In our control approach, the fingers are dynamically coupled and the manipulation will be executed by all fingers synchronously contributing to the task (unless a finger has been exempted or has a different role, see Section VII-D). Therefore, similar to grasping, the coupling DS (Eq.5) has to be computed and updated online.

VII. Experiments and Evaluations

We evaluate our approach qualitatively and quantitatively in four experiments with a real robotic system. We perform the following assessments:

(i) We evaluate the quality of control, adaptation, and coordination of fingers, in Section VII-A.
(ii) In Section VII-B, we evaluate the robustness of the grasped object with different shapes and mass properties Figure 10. This experiment is based on the approach summarized in Figure 7 and we study grasp adapting in an uncertain environment.
(iii) We perform in-hand manipulation (translation and rotation) of four objects of different shapes, and mass properties, in Section VII-C.
(iv) In Section VII-D, we augment our control approach with learning from demonstration to execute a complex manipulation task: in-hand object rolling. We devise this experiment to give an instance of a dynamic disturbance during in-hand manipulation.

The hardware for experiments consists of an Allegro V4 left hand[1] mounted on a KUKA LBR iiwa 14[2] robotic arm, OptiTrack motion capture system[3], and BioTac tactile sensors[4]. The Allegro hand has four fingers (thumb included), each with four actuated DoFs (16 DoFs in total). The KUKA arm has 7 DoFs. Both robots are operated in torque control mode at 200 Hz. Note that the robotic hand needs to be fully actuated, and this is the only hardware requirement for the efficient implementation of our controller. A set of three BioTac sensors are used as fingertips to obtain tactile information at 60 Hz. BioTac sensor is filled with an elastic liquid-filled skin that provides compliance similar to the human fingertip. To track the object's 3D pose within the hand frame, we used the Optitrack motion capture system with a 250 Hz running frequency.

A. Evaluation of Coordinated Finger-Control

We test the strength of our coordinated fingers control approach in two scenarios:

(i) We fix the base of Allegro hand in a specific pose. The fingertips need to first move from rest positions to ready positions and, then, to track a specific sequence of attractors, see Figure 8.
(ii) We compare our control algorithm against an impedance-based controller with a state-varying impedance called "Passive-DS" [51]. The task here is similar to the first scenario, except that it is repeated with the base of the hand being fixed in three different poses.

1) Tracking Performance and Dynamic Learning: In this experiment, all adaptive gains $\mathbf{\Psi}_\zeta$ and $\mathbf{\Psi}_\mathrm{r}$ for all fingers are initiated with zero value matrices and $\boldsymbol{\kappa}^* = [0.15, 0.12, 0.1, 0.05]$ for each finger and $\boldsymbol{\epsilon}_j = 0.02$ for all joints. Figure 8 shows the performance of our controller in the first experiment where the tracking error decreases simultaneously for all fingers. We observe, in displacement from rest positions to ready positions, higher oscillations in error compared to the transitions in other task segments (for instance from $\{\boldsymbol{x}_i^{(1)}\}_{i=1}^{i=n_f}$ to $\{\boldsymbol{x}_i^{(2)}\}_{i=1}^{i=n_f}$). In the first part, adaptive gains are initialized as zero matrices and the oscillation is due to controller exploration to find the proper range of gains. Whereas, in the other parts, gains are only fine-tuned in that specific range, see the right figure in Figure 8. Furthermore, using coupled dynamical system when moving from $\{\boldsymbol{x}_i^{(m)}\}_{i=1}^{i=n_f}$ to $\{\boldsymbol{x}_i^{(m+1)}\}_{i=1}^{i=n_f}$ results in synchronous convergence with lower oscillation in tracking. While converging to their attractors, all fingers will simultaneously regulate the joints' stiffness to desired values $\{\boldsymbol{\kappa}_j\}_{j=1}^{j=n_j}$. The regulation of joint impedance not only enables us to better control finger-object interactions later but also avoids joint saturation due to large initial errors, thus resulting in higher adaptation efficiency.

2) Comparison with Impedance-Based Controllers: The purpose of this experiment is to compare the performance of our low-level adaptive torque controller with impedance-based controllers as the baseline. Considering related studies with real robot evaluation, we find that impedance-based controllers are primarily employed to control the fingertip positions of the robotic hand [30, 33, 47]. We compare our method with the passive-DS control used in [47] for grasping. Passive-DS control [51] is one variation of impedance-based controllers, and it has an improved structure compared to the PD control employed in [30]. Passive-DS is compatible with the dynamical systems. Thus by using this controller, we constrain the experiment such that the only feature to be tested is the performance of the low-level controller.

We fix the base of the hand in three different poses. Then, we perform separately a tracking task by using two controllers: (a) our adaptive control algorithm, and (b) a PD controller with a state varying impedance. In the first, second, and third pose the gravity force is parallel to X, Y, and Z coordinates of the hand frame, respectively. The consistency of control performance, given different hand poses, is the key requirement to obtaining a robust and secure grasp. Here, we showcase the strength of our

[1]http://wiki.wonikrobotics.com/AllegroHandWiki/index.php/Allegro_Hand
[2]https://kuka.com/products/robotics-systems/industrial-robots/lbr-iiwa
[3]https://optitrack.com/applications/robotics
[4]https://syntouchinc.com/robotics

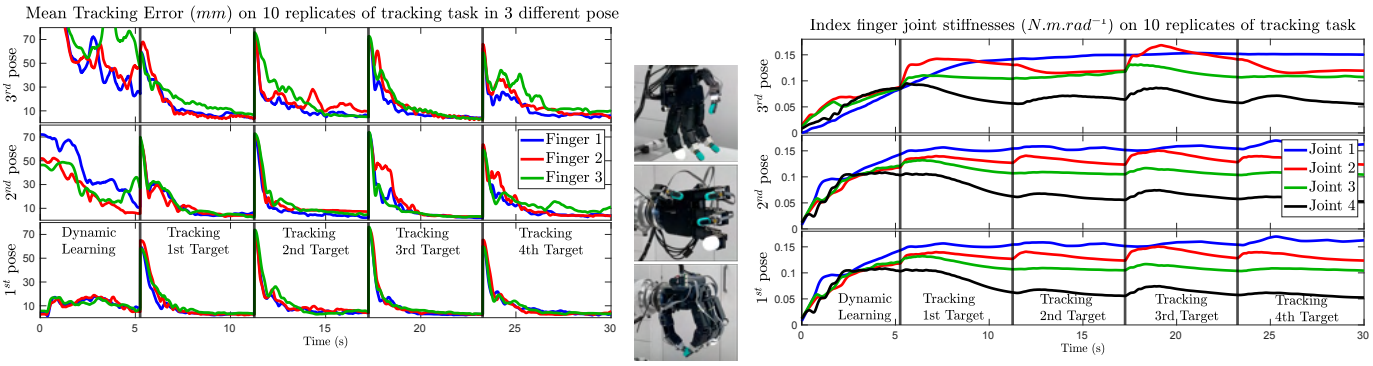

Fig. 8: Evaluation of the coordinated adaptive torque control on a robotic hand in three different wrist poses (different gravity and dynamics effect). In the first phase of the task, the fingers move from "rest" positions to "ready" positions, wherein first the unknown DS (Eq.12) is learned with adaptive gains converging. Then, the task is to track a specific sequence of different target positions. From left to right: (left): Mean tracking error (mm) over 10 replicates, (middle): robotic hand in three different poses, and (right): index finger joints' stiffness ($\boldsymbol{\kappa}$ in Eq.14) which are required to converge to the desired values, $\boldsymbol{\kappa}^* = [0.15, 0.12, 0.1, 0.05]$.

controller for situations where the pose of the hand is not fixed.

After implementing the task using the same control parameters for all poses, we recorded average steady-state errors over 20 replicates for each pose. To make a fair comparison, we pushed the passive-DS control gains to maximum achievable performance, without destabilizing the robot's joints. Figure 9 illustrates the results of the second scenario. In all three hand base poses, our controller consistently exhibits a higher accuracy compared to passive-DS (even with large PD gains). This is because we establish our adaptation at joint level to adapt for individual joint uncertainties and specifications, which is a key factor in object manipulation by a robotic hand. Therefore, for the cases where the uncertainty level and the gravity effect are considerably higher than the experiment in Figure 9 (e.g., when an object is grasped), our low-level adaptive controller significantly outperforms the impedance-based controllers.

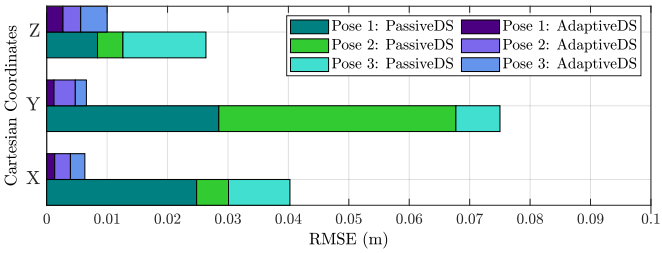

Fig. 9: Tracking RMSE of each Cartesian coordinate for our adaptive controller and a passive-DS with large gains. This experiment is similar to the one in Figure 8 and it is replicated 20 times for each specific pose.

B. Grasp Adaptation in an Uncertain Environment

In this part, our focus is on evaluating the grasp robustness for different objects of different properties, see Figure 10. In other words, we assess the performance of our controller in adapting the grasp where the robotic hand

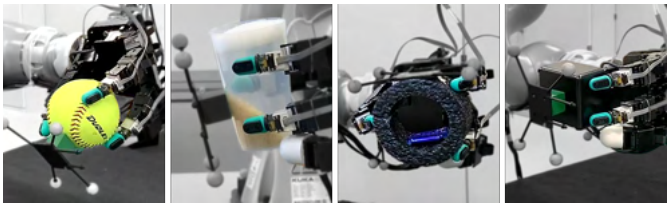

Fig. 10: Objects used in both grasp and manipulation experiments. From left to right: (1) Baseball (heavy object), (2) Tumbler half full of rice (varying mass distribution), (3) Annulus ring with a mass attached to inner part (uneven mass distribution), and (4) cuboid (normal object, easy to grasp).

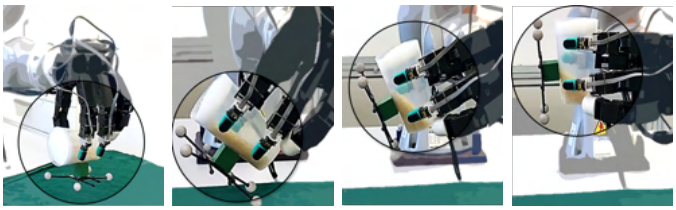

Fig. 11: Adaptation in grasp experiment when rotating a tumbler half-filled with rice. In this particular case, the mass distribution changes, thus imposing disturbances on fingers stability. Object relative displacement to the hand is recorded via OptiTrack motion capture system.

is needed to displace the object, e.g., hand-over task, and pick and place task. To do so, we devise the following scenarios: Using the same control framework, Figure 7, the Allegro hand first grasps the object in a predefined grasp configuration. While the object is held within the hand, the wrist (KUKA's last joint) rotates 90^o clockwise, thereby manipulating the position and the orientation of the object. This task is executed for ten replicates per object. Figure 11 depicts an example of it where the tumbler is half-filled with rice (object with varying mass distribution); see also supplementary materials for the videos of the experiment.

1) Results on Grasp Adaptation during Wrist Rotation: In this experiment, the controller has to adapt joint torques to face not only the changes in the gravity

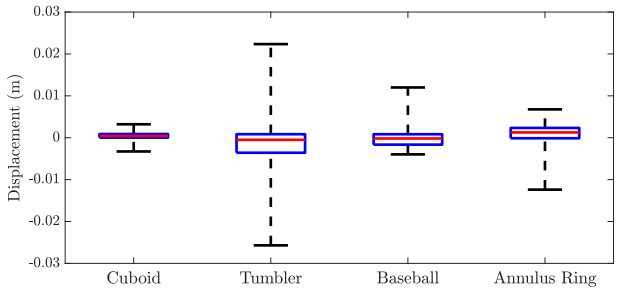

Fig. 12: Results of the adaptive grasp robustness experiment and rotation experiment replicated ten times (see Figure 11 as one example). Box-plots indicate objects relative displacement with respect to the hand's frame during wrist rotation.

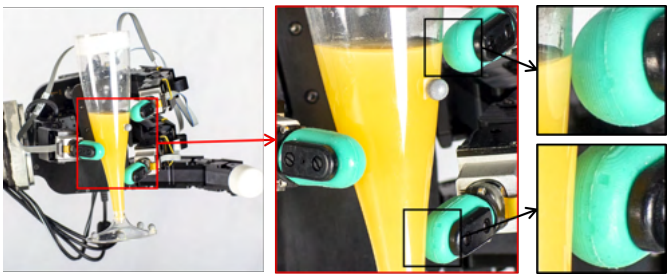

Fig. 13: An example of contact force adaptation. From the prior knowledge the object is assumed to be of a cylinder shape. Hence fingers arrive at contact each with a different orientation. Given the tactile sensor and the trained NN model (Eq.28), the normal vector $\hat{\boldsymbol{n}}_i$ at contact point is estimated for each i-th finger. Approximated contact frames are then fed to grasp optimization process (15) in order to extract contact wrenches.

direction but also the variation of object inertia. During this task, using OptiTrack motion capture system, we recorded the motion of object's center of mass relative to the hand frame. Figure 12 shows the boxplot of ten replicates of the experiment for each object. Given this figure, and Table I, our grasp algorithm is able to hold the object stable with very low relative displacement, and it provides proper robustness despite changes in the inertia of the objects. Slippage, partial occultation of the object from the OptiTrack system in the hand frame (jerk in position feedback), and rapid changes in inertia (faster than the adaptation rate in tumbler case) were the main causes of failed grasps in Table I. In the previous section, we show that our adaptive controller is capable of facing uncertainties at the joint level. Here, as long as the finger joints sense the perturbation from object uncertainties (i.e., fingertips are in contact with objects), our controller is capable of adapting to these uncertainties and remains robust enough to hold the object stable.

2) Force and Contact Normal-Vector Adaptation: Figure 13 depicts an example of contact-force adaptation. For each finger in contact, the vector normal to object surface $\hat{\boldsymbol{n}}_i$ is re-estimated through the trained NN model (Eq.28). These estimations are used to improve our prior knowledge of the object's shape, especially at contact points under the fingertips and tactile sensors. $\hat{\boldsymbol{n}}_i$ of fingers in contact is then fed to the grasp optimization process, Figure 7,

TABLE I: The number of trials performed for 10 successful executions of the adaptive grasp robustness experiment.

Cuboid	Tumbler	Baseball	Annulus Ring
10	14	12	12

TABLE II: Summary of results on three in-hand manipulations tasks ($20mm$ of translation, 30^o, and 45^o of rotation) on four objects in Figure 10. "Attempts" numbers indicate the number of trials performed for 10 successful executions, and "Error" values shows the steady-state tracking error of successful trials.

Task	$20\,mm$			
Objects	Cuboid	Tumbler	Baseball	Ring
Error (%)	−1±10	.2±10	.4±13	.1±11
Attempts	10	11	11	10
Task	30^o			
Objects	Cuboid	Tumbler	Baseball	Ring
Error (%)	8.7±3.8	−.5±5.5	−.3±11	−1.3±4.7
Attempts	11	12	11	10
Task	45^o			
Objects	Cuboid	Tumbler	Baseball	Ring
Error (%)	5.6±4.0	−.5±5.4	−1.0±5.0	.7±4.9
Attempts	12	13	14	11

where via shaping the object-level impedance, the force direction for each fingertip is extracted. In Figure 13, we observe that the robotic finger updates the force direction (pushing more toward the center of the ring and increasing the contact area), thereby securing grasp stability.

C. In-Hand Manipulation, Accuracy and Robustness

The experiments in this section are devised to manipulate the objects listed in Figure 10. Assuming that the objects are grasped perfectly, in order to realize a desired pose, the controller is needed to apply force to the object through the fingers by using Eq.17 and Eq.9; see Figure 14 as an example and see supplementary videos for the rest of objects. For comparison purposes, the desired pose manipulations are selected such that within the workspace of the hand, they are feasible for all objects. We also use in-hand manipulation in a gearbox assembly task (presented in supplementary videos) as an application of in-hand manipulation.

1) Results: Table II summarizes the results of ten replicates of a specific manipulation, which reveals a high tracking accuracy of our control approach. Failed attempts in Table II similar to Table I were mainly due to slippage, or partial occultation of the object from OptiTrack system. Figure 15 presents the tracking error profiles of 30^o rotation for all objects of the experiment. Apart from accurate tracking, one observation from Figure 15 is that, unlike grasping, objects with more regular shape or mass are not necessarily easier to manipulate. For instance, the mean tracking error for the cuboid is higher than the other objects in Figure 15, but it has the highest stable grasp

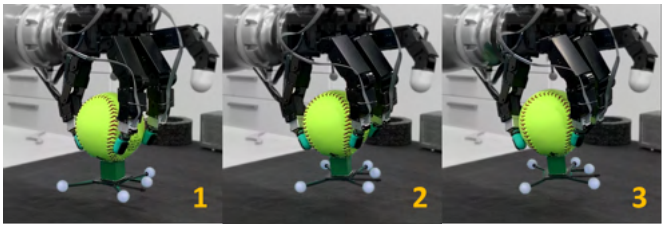

Fig. 14: A baseball, as an example of a heavy object, is rotated 30^o. The task is performed using DS (Eq.9) and the control approach in Figure 7. The mean rotation error for 10 replicates of this experiment is $-2 \times 10^{-3} rad$, see Table II.

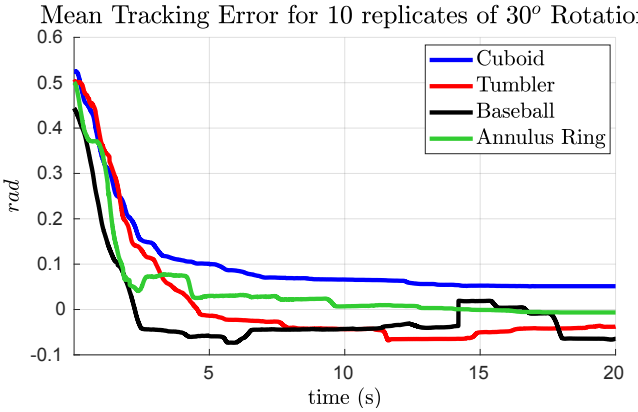

Fig. 15: Mean tracking error for one of the manipulation experiments. In this task, we rotate different objects listed in Figure 10 for 30^o in hand.

(Table I and Figure 12). This is otherwise unexpected if we do not consider the importance of robotic hand's dynamics.

2) Comparing with the State of the Art: Using Allegro hand in a similar experiment, Sundaralingam et al. [30] performed multiple in-hand manipulations on 10 objects from the YCB dataset. To move an object, they regulated the impedance of the grasp in the direction of the objective pose via kinematic trajectory optimization. Comparing Table II with the reported manipulation results in [30], we observe that the highest margin of position error (13% for the baseball) in Table II is smaller than the lowest reported median error for an object ($\approx$ 20% for the Lego object) in [30]. Similarly for the orientation, the reported median error (9.86%) in [30] is higher than the error margin in Table II. This difference can be explained by the fact that our low-level control adapts to dynamic uncertainties that results in tracking errors, contrary to the PD controller used in [30], see Section.VII-A2.

D. Finger-Gaiting, Rolling Object In Hand

In this experiment, we use our control approach to accomplish a complex manipulation task: in-hand object rolling, otherwise known as finger-gaiting. In this task, the hand rotates the object through a combination of contact/non-contact movements, see Figure 16 and 17. Human demonstrations (see Appendix.D) are used to extract the role of each finger and to identify the segments of the task through hidden Markov models (HMM) [52]. We show that task planning in this fashion in conjunction with our control framework enables us to encode a robust solution for in-hand rolling.

1) Task Segmentation: We employed the sticky hierarchical Dirichlet process (HDP)-HMM in [53] to find the optimum number of states and their sequence. HDP-HMM is applied on the recorded $\mathcal{D}_o = \{{}^o\boldsymbol{\psi}_t^{(i)}\}_{t=0..T_i}^{i=1,n_e}$ of the object's rotations (finger gaiting), where we stored the fingers' relative angular velocities ω and pressures p as observations with ${}^o\boldsymbol{\psi}_t^{(i)} = [\omega_1^T, p_1, \cdots, \omega_4^T, p_4]^T$. The HMM model extracted 3 hidden states each corresponding to a task segment, see Appendix.D for further detail.

2) Planning: According to task segmentation with HMM, this finger-gaiting manipulation consists of several fingers attaching to and detaching from the object's surface. In the phases of manipulation, where a finger does not interact with the object, it has to be controlled to preserve or increase its manipulability. Fingers that reach workspace limits will update their position in order to increase manipulability; and the other fingers will provide support to stabilize the object. In each task segment and for each finger, we detect the finger role, based on relative displacement:

$$\ell_i(s_t) = \begin{cases} 1, & \text{if } \bar{\dot{\boldsymbol{x}}}_i(s_t) < \epsilon \\ 0, & \text{if } \bar{\dot{\boldsymbol{x}}}_i(s_t) \geq \epsilon \end{cases}$$

where $\bar{\dot{\boldsymbol{x}}}_i(s_t) = \sum_{k=1}^{n_{s_t}} \|\dot{\boldsymbol{x}}_i^k\|^2 / n_{s_t}$ is relative mean velocity of i-th finger in task segment s_t, and $\epsilon > 0$ is a small positive value. Thus, $\ell_i(s_t) = 1$ indicates that i-th finger is contributing to rotation by inserting force on the object and $\ell_i(s_t) = 0$ denotes that for the given s_t finger i is detached. In case there is no interaction with the object, meaning that $\ell_i(s_t) = 0$, we compute the finger's desired attractor $\boldsymbol{x}_i^{(2)}$ from $\boldsymbol{\Psi}_o$:

$$\boldsymbol{x}_i^{(2)}|_{s_t=0} = \frac{1}{n_{s_t}} \sum_{k=1}^{n_{s_t}} \Big(u(\|\dot{\boldsymbol{x}}_i^k\|^2 + \delta) - u(\|\dot{\boldsymbol{x}}_i^k\|^2 - \delta) \Big) \boldsymbol{x}_i^k$$

here, $u(.)$ is the unit step function and $0 < \delta \ll 1$ is a small positive value. The extracted attractors for each task segment, and the desired object velocity are used for control and planning identical to Section VI-B2.

3) Evaluation: We realize a human like in-hand object rolling (see Figure 16). Figure 17 shows an implementation of this approach on a real robotic hand. The object is grasped in a configuration suitable for both stability and for generating the desired motion (see Section VI-A). The hand then begins rotating the object through a combination of several contact/non-contact movements; see also videos in the supplementary materials that explain a full cycle of rolling manipulation in different phases of finger motions. For this experiment, we could not use tactile sensors due to hardware limits and tactile sensor shape restrictions. Hence, to extract contact frames, we relied on prior knowledge of the object's shape and the robot's forward kinematics.

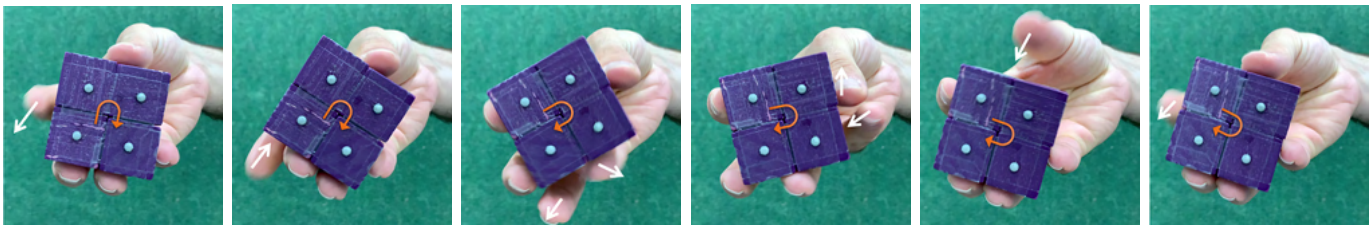

Fig. 16: From left to right, a human performing finger gaiting in-hand manipulation: the object is rotated clockwise and motion direction are depicted by arrows. The fingers have different roles and contributions to the task. The kinesthetic demonstrations, including object and finger motions, are later used in high-level planning and learning task segmentation through HMM.

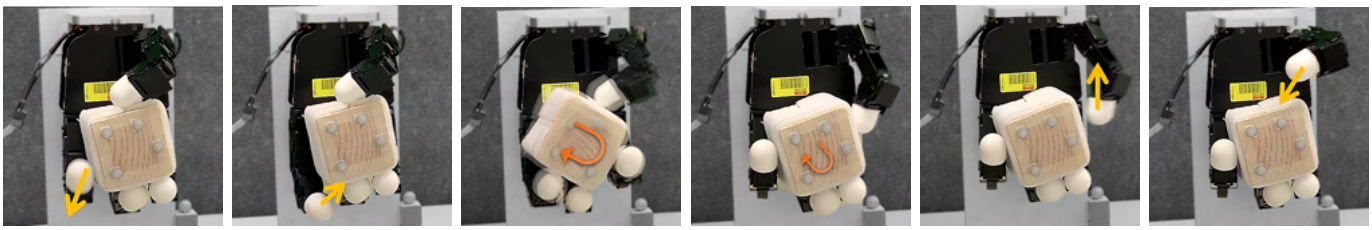

Fig. 17: From left to right, the robotic hand performs in-hand rolling (finger gaiting) manipulation: task planning and finger roles distributions are learned from human demonstrations. Object-finger interactions in contact and non-contact displacement are torque controlled using our adaptive controller to stably manipulate the object within the hand workspace.

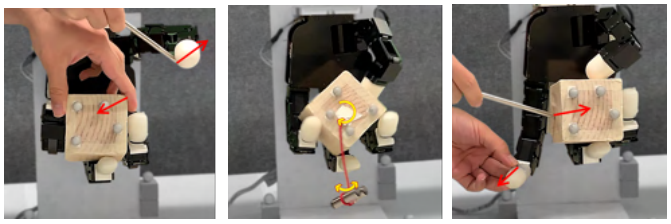

Fig. 18: Testing the robustness of our control approach by applying various perturbations to the fingers and the object during the finger-gaiting task: (left) perturbing thumb, (middle) attached pendulum that weighs as 20% of the cuboid, generates dynamic perturbations during manipulation, and (right) perturbing object and pinky finger.

Table III summarizes the results of 10 replicates of this experiment for a full cycle of rolling object. Each cycle consists of 4 sequential repetition of the task shown in Figure 17. Our results in Table III show 85% success rate in task execution and 60% success rate in the full-cycle rotation (accomplishing 4 tasks in succession). Failures were in the last rolling executions and were mainly due object blocking a finger (not allowing for finger workspace update) and object slipping during rotation.

TABLE III: Number of successful trials in 10 replicates of full object rolling experiment. Each cycle consists of 4 repetition of the rolling task in succession; see Figure 17 as an example of one rolling task.

Rolling Order:	I	II	III	IV
Successful Trials:	10	10	8	6

4) Robustness: This complex manipulation shows that with appropriate high-level planning, our control framework enables the encoding of a robust solution for task execution, even in face of different types of disturbances; see Figure 18. Disturbances were tested in three scenarios: (a) we add an additional unknown mass dangling from a wire, hence modifying significantly the mass and mass distribution, (b) we move the object horizontally, to showcase the adaptation of the fingers, and coordination and the immediate re-positioning of the fingers relative to the object's location, and (c) we pull one finger away and subject the object to an external force to validate the adaptation of the forces at the fingertips that re-balance the object in the absence of one finger. See also a supplementary video of the experiment that shows examples of rolling in-hand manipulation and efficient adaptation of our controller to various perturbations.

VIII. Discussion

In-hand manipulation and dexterous grasp are among the most challenging tasks for real-world robot applications. It is not straightforward, if not impossible, to acquire a universal framework for grasping different objects and for performing various manipulations. Due to the diverse complexities listed in the Section I, most real-world implementations of in-hand manipulation are engineered for a specific task. The choice of tasks, objects, and the manipulation scheme are constrained due to hardware limits. Nevertheless, we achieved several dexterous grasp and manipulation focusing on post-grasp stability and manipulation accuracy. In this study, we center our attention on the robustness and adaptation, accordingly. Hence, all of our evaluations are on a real-world robotic setup, where the uncertainties and modeling challenges are significantly higher than the simulated environment.

We first begin with robotic-hand control. During any dexterous manipulation, we will not be able to hold an object securely if the robot hand system itself is not stable. Therefore, learning and compensating for the

dynamics of the robotic hand is of primary importance. However, precise knowledge of the physical properties of the object and the robot is never available. Therefore, we developed a compliant torque-controller rather than using stiff joint positions. Our proposed coordinated adaptive torque-controller was designed to learn the robot dynamics for precision and compliance. We assumed that the control system states are fully observable and controllable; hence, to entirely benefit from our control approach, the robotic hand (whether or not anthropomorphic) has to be fully actuated.

Regarding grasp and manipulation, our main intuition is to acquire a coordinated multi-finger system and to develop a method for increasing robustness under uncertainties, such as varying or uneven mass distribution. We assume that, for each object, the grasp configuration is given and is not necessarily optimal or well-calibrated. We consequently improve this prior knowledge through tactile sensing and an online re-estimation of the contact frames. This enhanced knowledge, in addition to the robot's kinematic feedback, facilitates modulating object-level impedance for a more secure grasp. The effective frequency of the tactile sensor, however, restricts the update rate of the object-level impedance. As a result, the controller can tackle only the disturbances with a lower rate than this frequency. Besides, the object-level grasp impedance is adopted form [47], and it is based on static grasp assumption. We treated this assumption as one source of unmodeled dynamics addressed by our adaptation. However, such an assumption is restrictive, and modeling object dynamics, even inaccurate, into the grasp impedance would help the adaption at the joint level.

To improve the modeling of DSs of our approach, nonlinear systems could be used, especially for planning couple DSs, as some types of manipulation may not involve linearly coupled fingers. Additionally, our optimization (15) assumed a planar motion for the object, which sufficed for our real-world scenarios. While extending the planner to 3D motion planning would allow for more complex manipulations, it would require a different contact model and optimization with the inclusion of the fourth finger of the hand. However, in practice, this approach proved to be restrictive due to finger collisions and limited space for object movement.

To further improve the application of our approach, there are several limitations that we would like to address. In most of the experiments, slippage, especially for heavier objects, caused some trials to fail. Slippage detection and appropriate adaptation, accordingly, could improve the success rate. Although the object pose estimation within the hand by the OptiTrack system was accurate, it was never easy to obtain. Marker occlusion during finger motions occurs frequently, thus rendering pose tracking rough/unsteady. Seeking more reliable feedback, we would like to estimate the object's pose by using tactile observations within the hand frame.

IX. Conclusion

In this article, we have introduced a new control framework for grasping and manipulating objects. The approach of this study is based on synchronizing finger (task planning) and adapting joint torque (low-level control). We have compared our torque controller against a widely utilized low-level controller (PD) and have shown that our approach is consistently more accurate while maintaining finger joints compliant and coupled. Our proposed controller for grasp and manipulation uses contact-frame estimation (grasp matrix computation) and grasp optimization to construct an object-level impedance. We then use the devised coordinated adaptive torque controller to realize the desired impedance and perform grasp/manipulation tasks. Our main focus in this study was the robustness of grasp and manipulation. Hence, the pipeline has been extensively tested in various real-world robotic experiments and performed with high accuracy and robustness, under different uncertain conditions. Finally, we have shown that our framework, along with learning from demonstration, can successfully perform complicated manipulations, such as finger gaiting.

Acknowledgments

This work was supported by the European Research Council (ERC), Advanced Grant agreement No 741945, Skill Acquisition in Humans and Robots. Also, the authors would like to thank Salman Faraji, Konstantinos Chatzilygeroudis, and Harshit Khurana for their help. The authors thank Ms. Holly Cogliati-Bauereis for assistance in grammar and word choice for this article.

References

[1] N. C. Dafle et al., "Extrinsic dexterity: In-hand manipulation with external forces," in 2014 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2014, pp. 1578–1585.

[2] T. Feix et al., "The grasp taxonomy of human grasp types," IEEE Transactions on Human-Machine Systems, vol. 46, no. 1, pp. 66–77, 2016.

[3] D. Prattichizzo and J. C. Trinkle, "Grasping," in Springer handbook of robotics. Springer, 2016, pp. 955–988.

[4] A. M. Okamura et al., "An overview of dexterous manipulation," in Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065), vol. 1. IEEE, 2000, pp. 255–262.

[5] S.-J. Chung and J.-J. E. Slotine, "Cooperative robot control and concurrent synchronization of lagrangian systems," IEEE transactions on Robotics, vol. 25, no. 3, pp. 686–700, 2009.

[6] F. Caccavale and M. Uchiyama, "Cooperative manipulation," in Springer Handbook of Robotics. Springer, 2016, pp. 989–1006.

[7] S. S. Mirrazavi Salehian et al., "A unified framework for coordinated multi-arm motion planning," The International Journal of Robotics Research, vol. 37, no. 10, pp. 1205–1232, 2018.

[8] T. Kastritsi et al., "Progressive automation with dmp synchronization and variable stiffness control," IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 3789–3796, 2018.

[9] G. C. Goodwin and K. S. Sin, Adaptive filtering prediction and control. Courier Corporation, 2014.

[10] R. Ortega and M. W. Spong, "Adaptive motion control of rigid robots: A tutorial," Automatica, vol. 25, no. 6, pp. 877–888, 1989.

[11] L. Han and J. C. Trinkle, "Dextrous manipulation by rolling and finger gaiting," in Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146), vol. 1. IEEE, 1998, pp. 730–735.

[12] J. Xu et al., "Sampling-based finger gaits planning for multifingered robotic hand," Autonomous Robots, vol. 28, no. 4, pp. 385–402, 2010.

[13] A. Rajeswaran et al., "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," arXiv preprint arXiv:1709.10087, 2017.

[14] Andrychowicz et al., "Learning dexterous in-hand manipulation," The International Journal of Robotics Research, vol. 39, no. 1, pp. 3–20, 2020.

[15] K. Tahara et al., "Externally sensorless dynamic regrasping and manipulation by a triple-fingered robotic hand with torsional fingertip joints," in 2012 IEEE International Conference on Robotics and Automation. IEEE, 2012, pp. 3252–3257.

[16] K. Hang et al., "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.

[17] X.-Z. Zheng et al., "On dynamic control of finger sliding and object motion in manipulation with multi-fingered hands," IEEE Transactions on Robotics and Automation, vol. 16, no. 5, pp. 469–481, 2000.

[18] M. Cherif and K. K. Gupta, "Global planning for dexterous reorientation of rigid objects: Finger tracking with rolling and sliding," The International Journal of Robotics Research, vol. 20, no. 1, pp. 57–84, 2001.

[19] A. Holladay et al., "A general framework for open-loop pivoting," in 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015, pp. 3675–3681.

[20] Y. Karayiannidis et al., "Adaptive control for pivoting with visual and tactile feedback," in 2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2016, pp. 399–406.

[21] R. R. Ma et al., "Modeling and evaluation of robust whole-hand caging manipulation," IEEE Transactions on Robotics, vol. 35, no. 3, pp. 549–563, 2019.

[22] R. Platt et al., "Manipulation gaits: Sequences of grasp control tasks," in IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004, vol. 1. IEEE, 2004, pp. 801–806.

[23] R. R. Ma and A. M. Dollar, "An underactuated hand for efficient finger-gaiting-based dexterous manipulation," in 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014). IEEE, 2014, pp. 2214–2219.

[24] J.-A. Seon et al., "Enhance in-hand dexterous micromanipulation by exploiting adhesion forces," IEEE Transactions on Robotics, vol. 34, no. 1, pp. 113–125, 2017.

[25] A. Bicchi, "Hands for dexterous manipulation and robust grasping: A difficult road toward simplicity," IEEE Transactions on robotics and automation, vol. 16, no. 6, pp. 652–662, 2000.

[26] R. Ozawa and K. Tahara, "Grasp and dexterous manipulation of multi-fingered robotic hands: a review from a control view point," Advanced Robotics, vol. 31, no. 19-20, pp. 1030–1050, 2017.

[27] J. Shi et al., "Dynamic in-hand sliding manipulation," IEEE Transactions on Robotics, vol. 33, no. 4, pp. 778–795, 2017.

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

[29] N. Vahrenkamp et al., "Integrated grasp and motion planning," in 2010 IEEE International Conference on Robotics and Automation. IEEE, 2010, pp. 2883–2888.

[30] B. Sundaralingam and T. Hermans, "Relaxed-rigidity constraints: kinematic trajectory optimization and collision avoidance for in-grasp manipulation," Autonomous Robots, vol. 43, no. 2, pp. 469–483, 2019.

[31] M. Pfanne et al., "Object-level impedance control for dexterous in-hand manipulation," IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 2987–2994, 2020.

[32] K. Hertkorn et al., "Planning in-hand object manipulation with multifingered hands considering task constraints," in 2013 IEEE International Conference on Robotics and Automation. IEEE, 2013, pp. 617–624.

[33] M. Li et al., "Learning of grasp adaptation through experience and tactile sensing," in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. Ieee, 2014, pp. 3339–3346.

[34] Q. Li et al., "A review of tactile information: Perception and action through touch," IEEE Transactions on Robotics, vol. 36, no. 6, pp. 1619–1634, 2020.

[35] P. Sodhi et al., "Learning tactile models for factor graph-based estimation," arXiv preprint arXiv:2012.03768, 2020.

[36] R. Calandra et al., "More than a feeling: Learning to grasp and regrasp using vision and touch," IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 3300–3307, 2018.

[37] V. Kumar et al., "Learning dexterous manipulation policies from experience and imitation," arXiv preprint arXiv:1611.05095, 2016.
[38] H. Zhu et al., "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost," in 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019, pp. 3651–3657.
[39] N. Chavan-Dafle and A. Rodriguez, "Prehensile pushing: In-hand manipulation with push-primitives," in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2015, pp. 6215–6222.
[40] N. Chavan Dafle and A. Rodriguez, "Stable prehensile pushing: In-hand manipulation with alternating sticking contacts," in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 254–261.
[41] Y. Bai and C. K. Liu, "Dexterous manipulation using both palm and fingers," in 2014 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2014, pp. 1560–1565.
[42] N. Furukawa et al., "Dynamic regrasping using a high-speed multifingered hand and a high-speed vision system," in Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006. IEEE, 2006, pp. 181–187.
[43] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," IEEE Transactions on Robotics, vol. 27, no. 5, pp. 943–957, 2011.
[44] F. Khadivar et al., "Learning dynamical systems with bifurcations," Robotics and Autonomous Systems, vol. 136, p. 103700, 2021.
[45] K. Farshad et al., "Efficient configuration exploration in inverse dynamics acquisition of robotic manipulators," in 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 1934–1941.
[46] P. Culbertson and M. Schwager, "Decentralized adaptive control for collaborative manipulation," in 2018 IEEE international conference on robotics and automation (ICRA). IEEE, 2018, pp. 278–285.
[47] M. Li et al., "Dexterous grasping under shape uncertainty," Robotics and Autonomous Systems, vol. 75, pp. 352–364, 2016.
[48] S. Dragiev et al., "Gaussian process implicit surfaces for shape estimation and grasping," in 2011 IEEE International Conference on Robotics and Automation. IEEE, 2011, pp. 2845–2850.
[49] K. Hang et al., "Hand–object configuration estimation using particle filters for dexterous in-hand manipulation," The International Journal of Robotics Research, vol. 39, no. 14, pp. 1760–1774, 2020.
[50] M. A. Roa and R. Suárez, "Grasp quality measures: review and performance," Autonomous robots, vol. 38, no. 1, pp. 65–88, 2015.
[51] K. Kronander and A. Billard, "Passive interaction control with dynamical systems," IEEE Robotics and Automation Letters, vol. 1, no. 1, pp. 106–113, 2015.
[52] S. Niekum et al., "Learning and generalization of complex tasks from unstructured demonstrations," in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2012, pp. 5239–5246.
[53] E. B. Fox et al., "An hdp-hmm for systems with state persistence," in Proceedings of the 25th international conference on Machine learning. ACM, 2008, pp. 312–319.
[54] S. Feng et al., "Optimization based full body control for the atlas robot," in 2014 IEEE-RAS International Conference on Humanoid Robots. IEEE, 2014, pp. 120–127.
[55] H. K. Khalil and J. W. Grizzle, Nonlinear systems. Prentice hall Upper Saddle River, NJ, 2002, vol. 3.
[56] C. M. Bishop, Neural Networks for Pattern Recognition. USA: Oxford University Press, Inc., 1995.
[57] A. Paszke, Gross et al., "Automatic differentiation in pytorch," in NIPS-W, 2017.
[58] S. H. Lee et al., "Autonomous framework for segmenting robot trajectories of manipulation task," Autonomous robots, vol. 38, no. 2, pp. 107–141, 2015.

Appendix A
Stability of Dynamical Systems

A. Stable Dynamical Systems

Stable DS is a globally stable velocity field that provides the system's controller with desired velocities. A DS function $f(\boldsymbol{\xi})$ is continuous, autonomous, state-dependent, and converges to an attractor $\boldsymbol{\xi}^*$. This attractor can represent either a fixed state or a specific limit cycle [44]. A DS function can also be expressed as a nonlinear combination of linear DSs to be $f(\boldsymbol{\xi}) = \sum_{k=1}^{K} h^k(\boldsymbol{\xi})\big(\mathbf{A}^k\boldsymbol{\xi} + \boldsymbol{b}^k\big)$ with $\boldsymbol{\xi} \in \mathbb{R}^n$ being the states of the system. $\mathbf{A}^k \in \mathbb{R}^{n\times n}$, as state matrix, and $\boldsymbol{b}^k \in \mathbb{R}^n$ construct the k-th linear system. The state-dependent mixing function, $h^k(\boldsymbol{\xi})$, represents a relative effect of the k-th system on shaping the overall nonlinear system. Given a desired robot motion, DS parameters $\mathbf{A}^k$, $\boldsymbol{b}^k$ and $h^k(\boldsymbol{\xi})$ can be either designed or learned from a demonstration [43]. Regardless of the acquisition method, these parameters must satisfy the following sufficient conditions for stability in the sense of Lyapunov :

$$\begin{cases} (\mathbf{A^k})^T\mathbf{P} + \mathbf{P}\mathbf{A^k} \prec -\mathbf{Q}^k \\ (\mathbf{Q}^k)^T = \mathbf{Q}^k \succ 0,\ (\mathbf{P}^k)^T = \mathbf{P}^k \succ 0 \\ \boldsymbol{b}^k = -\mathbf{A}^k\boldsymbol{\xi}^* \end{cases} \tag{18}$$

where $\mathbf{Q}^k \in \mathbb{R}^{n\times n}$ determines the convergence rate of k-th linear system to $\boldsymbol{\xi}^*$ and $\mathbf{P}^k \in \mathbb{R}^{n\times n}$ forms the potential function of the desired velocity field. DS-based controllers –i.e., controllers driven by a DS– are robust to perturbations because DS nests all possible trajectories to reach the target point, i.e., attractor.

B. Stability of the Intermediate-DS

If we consider the Lyapunov function $V(z) = \frac{1}{2}(z-z^d)^2$, taking the time derivative results in $\dot{V} = (z - z^d)\dot{z}$. By replacing $\dot{z}$ with the proposed DS (5) we will have

$$\dot{V} = -\beta_1 \frac{1 - e^{-\beta_2(z-z^d)}}{1 + e^{-\beta_2(z-z^d)}}(z - z^d)$$

for $z - z^d$ the derivative is $\dot{V}(z) = 0$, and for $z \neq z^d$, $\dot{V} = -\beta_1\gamma(z)$ where $\gamma(z) = \frac{1-e^{-\beta_2(z-z^d)}}{1+e^{-\beta_2(z-z^d)}}(z-z^d)$. It is easy to check that for $z \neq z^d$, $\gamma(z) > 0$. If we take the derivative of γ:

$$\gamma^{'} = \frac{1 - e^{-2\beta_2(z-z^d)} + 2\beta_2(z - z^d)e^{-\beta_2(z-z^d)}}{(1 + e^{-\beta_2(z-z^d)})^2}$$

then:

$$\begin{cases} \gamma^{'} > 0 & z > z^d \\ \gamma^{'} < 0 & z < z^d \end{cases}$$

hence for $z \neq z^d$, $\gamma(z) > 0$ and $\dot{V}(z) < 0$. As a result, the DS (5) is asymptotically stable and $\|z^d - z\| \to 0$ as $t \to \infty$.

C. Stability of the Task-Space DS

We establish, based on the tracking error $\boldsymbol{x}_i - \boldsymbol{x}_i^*$, the Lyapunov function for the taskspace DS. Let's assume the Lyapunov function $V(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_i^*)^T\mathbf{P}(\boldsymbol{x}_i - \boldsymbol{x}_i^*)$ and take it's time derivative:

$$\dot{V} = \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_i^*)^T\mathbf{P}(\dot{\boldsymbol{x}}_i - \dot{\boldsymbol{x}}_i^*) + \frac{1}{2}(\dot{\boldsymbol{x}}_i - \dot{\boldsymbol{x}}_i^*)^T\mathbf{P}(\boldsymbol{x}_i - \boldsymbol{x}_i^*)^T$$

By using DS (6) and the derivative of Eq.4 into the time derivative of the Lyapunov function, we will have

$$\dot{V} = -\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x}_i^*)^T(\mathbf{A}_x^T\mathbf{P} + \mathbf{P}\mathbf{A}_x)(\boldsymbol{x}_i - \boldsymbol{x}_i^*)$$

which by satisfying similar conditions, such as Eq.18, the stability of the DS (6) is guaranteed.

D. Stability of Coupled DS

For each i-th finger, proving that $\lim_{t\to\infty}\|z^c - z_i\| = 0$ is similar to stability proof in the Appendix.A-B. Here, we want to show that if $\lim_{t\to\infty}\|z^c - z_i\| = 0$ then $\|z^d - z_i\| \to 0$, as well. From $z^c = \frac{1}{n_f+1}(z^d + \sum_{i=1}^{n_f}\alpha_i)$ we can rewrite the error term $z^c - z_i$ in the following form:

$$z^c - z_i = (z^d - z_i) - \frac{\sum_1^{n_f}(z^d - z_i)}{n_f + 1} \tag{19}$$

which means :

$$\sum_1^{n_f}(z^c - z_i) = \frac{1}{n_f + 1}\sum_1^{n_f}(z^d - z_i) \tag{20}$$

by having the square of Eq.19 and then the sum over the entire systems we will have:

$$\sum_1^{n_f}(z^c - z_i)^2 = \sum_1^{n_f}(z^d - z_i)^2 - \frac{n_f + 2}{(n_f + 1)^2}\Big(\sum_1^{n_f}(z^d - z_i)\Big)^2 \tag{21}$$

next we use the square Eq.20 in the right hand side of Eq.20 such that we acquire the following equality:

$$\sum_1^{n_f}(z^d - z_i)^2 = \sum_1^{n_f}(z^c - z_i)^2 + (n_f + 2)\Big(\sum_1^{n_f}(z^c - z_i)\Big)^2 \tag{22}$$

which can also be observed in Figure 4. From Eq.22, we conclude that when $\lim_{t\to\infty}\|z^c - z_i\| = 0$ for all $i = 1$ to n_f then $\lim_{t\to\infty}\sum_1^{n_f}(z^d - z_i)^2 = 0$, and therefore, $\|z^d - z_i\| \to 0$ as $t \to \infty$.

Appendix B
Joint-Space Control

A. Computing the Regulation Signal

We can use any inverse kinematic (IK) solver [54] or more advanced probabilistic IK models [47] for mapping the desired velocity (Eq.9) to the desired state $\boldsymbol{\zeta}_i^d$, then we can use $\boldsymbol{r} = -\mathbf{B}_r^\dagger\mathbf{A}_r\boldsymbol{\zeta}_i^d$ to compute the regulation signal ($\mathbf{B}_r^\dagger$ is the pseudo inverse of $\mathbf{B}_r$). Although this computation coordinates well with the rest of our control derivation, we offer the following quadratic optimization:

$$\min_{\boldsymbol{r},\boldsymbol{q}_i',\dot{\boldsymbol{q}}_i'} -\frac{\mathrm{w}_1}{2}\|\boldsymbol{\delta}\|^2 - \frac{\mathrm{w}_2}{2}\|\boldsymbol{q}_i'\|^2$$
$$\text{s.t.} \begin{cases} \mathbf{B}_r\boldsymbol{r} + \mathbf{A}_r[\boldsymbol{q}_i'^T, \dot{\boldsymbol{q}}_i'^T]^T = 0 \\ \mathbf{J}\dot{\boldsymbol{q}}_i' = \dot{\boldsymbol{x}}_i^d + \boldsymbol{\delta} \\ \boldsymbol{q}_i' - \dot{\boldsymbol{q}}_i'\,dt = \boldsymbol{q}_i \\ \boldsymbol{q}_i' \in [\boldsymbol{q}_{min}, \boldsymbol{q}_{max}] \\ \dot{\boldsymbol{q}}_i' \in [\dot{\boldsymbol{q}}_{min}, \dot{\boldsymbol{q}}_{max}] \end{cases} \tag{23}$$

with $\boldsymbol{\zeta}_i^d = [\boldsymbol{q}_i'^T, \dot{\boldsymbol{q}}_i'^T]^T$, and weights $\mathrm{w}_1 \in \mathbb{R}$, and $\mathrm{w}_2 \in \mathbb{R}$ that tune the importance of reaching the desired velocity and penalizing non-uniform joint angles. This optimization scheme will be solved at every time step and works similar to other IK solvers; however, it favors a more uniform joint configuration and smoother velocity transition, suitable for the problem at hand. It should be noted that we assumed the solution of IK for a given $\dot{\boldsymbol{x}}_i^d$ exists, which relies on appropriate task planning and hardware limits. The stability of DS (9) and IK optimization in (23) guarantee that $\boldsymbol{r}$ remains bounded.

B. Stability Proof of Adaptive Control

We define $\mathbf{e} = \boldsymbol{\zeta} - \boldsymbol{\zeta}_i^d$ to be the tracking error. From Eq.11 and Eq.12 with the controller in Eq.13, we can construct the error dynamics as

$$\dot{\boldsymbol{e}} = (\mathbf{A} + \mathbf{B}\boldsymbol{\Psi}_\zeta)\boldsymbol{\zeta} - \mathbf{A}_r\boldsymbol{\zeta}_r + (\mathbf{B}\boldsymbol{\Psi}_r - \mathbf{B}_r)\boldsymbol{r})$$

and if we assume that there exist $\boldsymbol{\Psi}_\zeta^*$,and $\boldsymbol{\Psi}_\mathrm{r}^*$ such that

$$\begin{aligned} \mathbf{A} + \mathbf{B}\boldsymbol{\Psi}_\zeta^* &= \mathbf{A}_r \\ \mathbf{B}\boldsymbol{\Psi}_r^* &= \mathbf{B}_r \end{aligned} \tag{24}$$

then the error dynamics will be summarized as

$$\dot{\boldsymbol{e}} = \mathbf{A}_r \boldsymbol{e} + \mathbf{B}\widetilde{\boldsymbol{\Psi}}_\zeta \boldsymbol{\zeta} + \mathbf{B}\widetilde{\boldsymbol{\Psi}}_r \boldsymbol{r} \tag{25}$$

where $\widetilde{\boldsymbol{\Psi}}_\zeta = \boldsymbol{\Psi}_\zeta - \boldsymbol{\Psi}_\zeta^*$ and $\widetilde{\boldsymbol{\Psi}}_\mathrm{r} = \boldsymbol{\Psi}_\mathrm{r} - \boldsymbol{\Psi}_\mathrm{r}^*$ are gain prediction errors. The goal is to have adaptation laws that stabilize this system and regulate prediction errors. For this, we consider the following Lyapunov function:

$$\mathbf{V}(\boldsymbol{e}, \widetilde{\boldsymbol{\Psi}}_\zeta, \widetilde{\boldsymbol{\Psi}}_r) = \frac{1}{2}\boldsymbol{e}^T\mathbf{P}\boldsymbol{e} + \frac{1}{2}\mathbf{tr}\big(\widetilde{\boldsymbol{\Psi}}_\zeta^T \boldsymbol{\Theta}_\zeta \widetilde{\boldsymbol{\Psi}}_\zeta + \widetilde{\boldsymbol{\Psi}}_r^T \boldsymbol{\Theta}_r \widetilde{\boldsymbol{\Psi}}_r\big)$$

where $\boldsymbol{\Theta}_\zeta$ and $\boldsymbol{\Theta}_r \succ 0$. Differentiating in time, we have:

$$\begin{aligned}\dot{\mathbf{V}} = &\frac{1}{2}\boldsymbol{e}^T(\mathbf{P}\mathbf{A}_r + \mathbf{A}_r^T\mathbf{P})\boldsymbol{e} + \boldsymbol{e}^T\mathbf{P}\mathbf{B}\big(\widetilde{\boldsymbol{\Psi}}_\zeta\boldsymbol{\zeta} + \widetilde{\boldsymbol{\Psi}}_r\boldsymbol{r}\big) \\ &+ \mathbf{tr}\big(\widetilde{\boldsymbol{\Psi}}_\zeta^T\boldsymbol{\Theta}_\zeta\dot{\widetilde{\boldsymbol{\Psi}}}_\zeta + \widetilde{\boldsymbol{\Psi}}_r^T\boldsymbol{\Theta}_r\dot{\widetilde{\boldsymbol{\Psi}}}_r\big)\end{aligned} \tag{26}$$

To make the Lyapunov derivative independent of the unknown matrix $\mathbf{B}$, from Eq.24, we assume $\mathbf{B} = \mathbf{B}_r\boldsymbol{\Psi}_r^{*-1}$, for which the existence of $\boldsymbol{\Psi}_r^{*-1}$ relies on $\mathbf{B}$ being full rank or the system (Eq.12) being controllable. As we place the adaptive control on joint torques, the system (Eq.12) is fully controllable. Thus, this assumption is not restrictive for our controller. Given this, we can replace $\boldsymbol{\Theta}_\zeta$ and $\boldsymbol{\Theta}_r$ with:

$$\begin{aligned}\boldsymbol{\Theta}_\zeta &= \boldsymbol{\Psi}_r^{*-T}\Lambda_\zeta^{-1} \\ \boldsymbol{\Theta}_r &= \boldsymbol{\Psi}_r^{*-T}\Lambda_r^{-1}\end{aligned}$$

that converts Eq.26 to:

$$\begin{aligned}\dot{\mathbf{V}} = &-\frac{1}{2}\boldsymbol{e}^T\mathbf{Q}\boldsymbol{e} + \boldsymbol{e}^T\mathbf{P}\mathbf{B}_r\boldsymbol{\Psi}_r^{*-1}\big(\widetilde{\boldsymbol{\Psi}}_\zeta\boldsymbol{\zeta} + \widetilde{\boldsymbol{\Psi}}_r\boldsymbol{r}_t\big) \\ &+ \mathbf{tr}\big(\widetilde{\boldsymbol{\Psi}}_\zeta^T\boldsymbol{\Psi}_r^{*-T}\Lambda_\zeta^{-1}\dot{\widetilde{\boldsymbol{\Psi}}}_\zeta + \widetilde{\boldsymbol{\Psi}}_r^T\boldsymbol{\Psi}_r^{*-T}\Lambda_r^{-1}\dot{\widetilde{\boldsymbol{\Psi}}}_r\big)\end{aligned} \tag{27}$$

with this and, taking advantage of trace properties for square matrices $\mathbf{tr}(\mathbf{AB}) = \mathbf{tr}(\mathbf{BA})$, we can introduce the following adaptation laws:

$$\begin{aligned}\dot{\boldsymbol{\Psi}}_\zeta = \dot{\widetilde{\boldsymbol{\Psi}}}_\zeta &= -\Lambda_\zeta\,\mathbf{B}_r^T\,\mathbf{P}\,\boldsymbol{e}\,\boldsymbol{\zeta}^T - \mathbf{S}\,\Upsilon_\zeta\,\bar{\boldsymbol{\Psi}}_{\boldsymbol{\zeta}} \\ \dot{\boldsymbol{\Psi}}_r = \dot{\widetilde{\boldsymbol{\Psi}}}_r &= -\Lambda_r\,\mathbf{B}_r^T\,\mathbf{P}\,\boldsymbol{e}\,\boldsymbol{r}^T - \mathbf{S}\,\Upsilon_\mathbf{r}\,\bar{\boldsymbol{\Psi}}_\mathbf{r}\end{aligned}$$

where Λ_ζ and Λ_r are positive definite matrices designed to tune convergence rate of the adaptive gains. Diagonal matrices $\mathbf{S}$, $\boldsymbol{\Psi}_{\boldsymbol{\xi}}$, $\boldsymbol{\Psi}_\mathrm{r}$ are activation and regulation matrices for Eq.14 as explained in Section V. $\mathbf{P}$ with $\mathbf{A}_r$ satisfy $\mathbf{P}\mathbf{A}_r + \mathbf{A}_r^T\mathbf{P} = -\mathbf{Q}$ as the necessary and sufficient stability condition for the reference model (Eq.11). Putting these laws in Eq.27, the Lyapunov derivative simplifies to:

$$\dot{\mathbf{V}} = -\frac{1}{2}\boldsymbol{e}^T\mathbf{Q}\boldsymbol{e}$$

since $\mathbf{V} > 0$ and $\dot{\mathbf{V}} \leq 0$ the system in Eq.25 is stable from Lyapunov perspective, thus $\boldsymbol{e}, \widetilde{\boldsymbol{\Psi}}_\zeta$, and $\widetilde{\boldsymbol{\Psi}}_r$ are bounded. As $\boldsymbol{r}$ is bounded by definition, the system (Eq.25) is bounded as well. Therefore, $\ddot{\mathbf{V}} = -\boldsymbol{e}^T\mathbf{Q}\dot{\boldsymbol{e}}$ will be bounded always, and from Barbalat's lemma [55], this system is asymptotically stable.

Appendix C
Contact-Frame Estimation

For a secure grasp, addressing the uncertainties stemming from an object's shape is the missing piece of our puzzle. In many real world experiments, the object shape is not accurately known. We train a model to estimate, online from tactile feedback, the frame of the contact, $\{C_i\} = \{\hat{\boldsymbol{n}}_i, \hat{\boldsymbol{t}}_i, \hat{\boldsymbol{o}}_i\}$. The tactile sensor consists of a rigid core surrounded by an elastic liquid-filled skin that provides compliance similar to the human fingertip. When forces are applied to the contact, the skin and fluid deforms, which is detected by an array of electrodes on the surface of the sensor core.

1) Model Training: We first need to collect the ground truth, consisting of (a) BioTac electrodes impedance, $\boldsymbol{\rho}_i \in \mathbb{R}^{19}$, (b) the orientation of fingertip base (sensor base), and (c) the orientation of the contact plane. We use the OptiTrack motion-capture system to get both fingers, $\boldsymbol{q}_{f,i}^O$, and contact plane, $\boldsymbol{q}_{c,i}^O$, quaternions: $\hat{\boldsymbol{n}}_{c,i}^B = \begin{bmatrix} -2(q'_{i,x}q'_{i,z} + q'_{i,w}q'_{i,y}) \\ -2(q'_{i,y}q'_{i,z} - q'_{i,w}q'_{i,x}) \\ 1 - 2(q'^2_{i,w} + q'^2_{i,z}) \end{bmatrix}$ in which $\boldsymbol{q}'_i = \boldsymbol{q}_{f,i}^O\bar{\boldsymbol{q}}_{c,i}^O = [q'_{i,w}, q'_{i,x}, q'_{i,y}, q'_{i,z}]$. The collected dataset is represented by $\mathcal{D} = \{\hat{\boldsymbol{n}}_{c,i}^{B\,(m)}, \boldsymbol{\rho}_i^{(m)}\}_{m=0}^{m=M}$ where M is the total number of samples(here $40,000$); see an example of data collection and the recording setup in supplementary videos. The desired mapping, $\hat{\boldsymbol{n}}_{c,i}^B = f_{NN}(\boldsymbol{\rho}_i)$ is modeled and learned by an artificial neural network (ANN) [56], and by using PyTorch library [57]. The data is split into training (75%) and testing (25%) sets for 20 folds. The key hyperparameters associated with the shallow neural networks are (a) activation function, (b) number of hidden layers, and (c) neurons per each hidden layer. The activation function used in the networks is ReLU, and the other hyperparameters were determined using grid search. The model accuracy on their respective datasets are provided in Table IV.

TABLE IV: Results on training and testing the model for contact normal estimation. This model is selected after a 20-fold cross validation with training (75%) and testing (25%) sets.

Hidden layers	Neurons per layer	Training set absolute error	Test set absolute error
2	200	$1.81° \pm 1.40°$	$5.70° \pm 1.90°$

2) Using the ANN Model: The learned mapping $\hat{\boldsymbol{n}}_{c,i}^B = f_{NN}(\boldsymbol{\rho}_i)$ outputs the vector $\hat{\boldsymbol{n}}_{c,i}^B$ presented in the fingertip's base frame, which is needed to be transformed $\hat{\boldsymbol{n}}_{c,i}^B$ to the inertia frame $\{N\}$, fixed to the root frame of the robotic hand. We can use the forward kinematic of fingertips given the joint configuration $\boldsymbol{q}_i$ of the i-th fingertip :

$$\hat{\boldsymbol{n}}_i = \boldsymbol{R}_i^T(\boldsymbol{q}_i)\, f_{NN}(\boldsymbol{\rho}_i) \tag{28}$$

matrix $\boldsymbol{R}_i(\boldsymbol{q}_i)$ is the rotation matrix from frame $\{B\}_i$ to $\{N\}$. Finally, by using Eq.28, contact frame $\{C_i\} =$

$\{\hat{\boldsymbol{n}}_i, \hat{\boldsymbol{t}}_i, \hat{\boldsymbol{o}}_i\}$ at i-th contact point are updated with the provided frequency of tactile sensors (60Hz for BioTac).

Appendix D
Learning Rolling in Hand from Demonstration

1) Data Collection: We record the object's pose, the finger positions, and contact forces from human demonstrations. The position information from expert demonstrations is recorded at 250 Hz by using the Opti-Track Motion Capture system, see Figure 19. The contact wrenches are recorded with a set of TPS tactile sensors, thus providing a measure of exerted pressure at 50 Hz, at the fingertips. Next, each demonstration is expressed in sets of data represented in the object frame at center of mass : $\mathcal{D}_o = \{{}^o\boldsymbol{\psi}_t^{(i)}\}_{t=0..T_i}^{i=1,n_e}$ where ${}^o\boldsymbol{\psi}_t^{(i)} = [{}^o\boldsymbol{x}_1(t)^T, p_1(t), \cdots, {}^o\boldsymbol{x}_{n_f}(t)^T, p_{n_f}(t)]^T$, and n_e is the number of recorded demonstrations. Recorded data $\mathcal{D}_o$ explains how the fingertips move relative to the object. This data is then used to identify the task segments, grasp positions, and the finger roles.

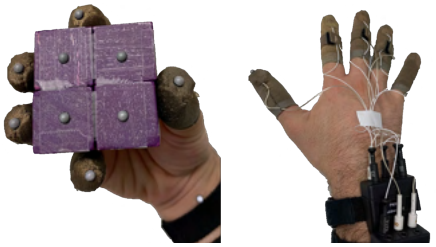

Fig. 19: Setup for recording data to learn from human demonstration. Markers are attached to the object and fingertips to track positions in the motion-capture system (OptiTrack), and TPS sensors for recording exerted pressure on fingertips.

2) Model Selection: Two probabilistic approaches are prevalent for decomposing the human motions into meaningful actions or states: (i) Gaussian Mixture Models (GMM) [58], where segments are extracted by fitting a GMM to the set of demonstrations, considering each GMM component as one segment, and (ii) Hidden Markov models (HMM) [52] that use a Markov process to explain the probabilities of sequences of hidden events (states) from possible observations (emissions). The latter is more suitable for the problem at hand, because the segmentation is not only based on a spatial sense but also on the transition dynamics across segments, and because HMM addresses transition dynamics handled through Markovian states. Given the dataset $\mathcal{D}_o$, we independently segment each trajectory while discovering the sequence of observations over T_i discrete time steps. In our case, the number of underlying hidden states ,$\mathbf{S} = \{s_t\}_{t=0..T_i}$, is unknown and the objective is to find the number of hidden states and then the sequence of them. Hence, we employed the sticky hierarchical Dirichlet process (HDP)-HMM in [53]. HDP-HMM uses Bayesian non-parametric methods in HMM to find the optimum number of states and their state transition matrix.

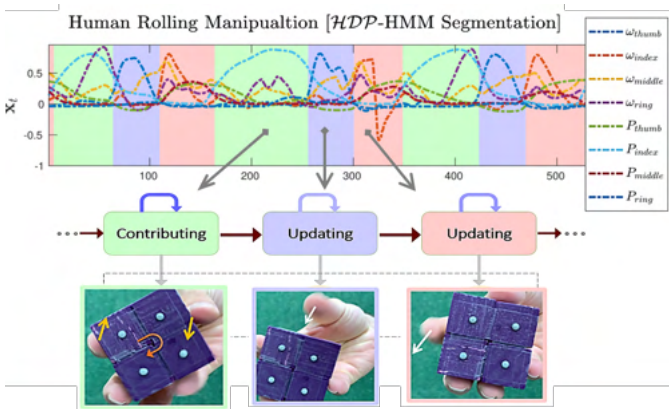

Fig. 20: Task segments extracted from HDP-HMM using relative angular velocities and pressure data. The green segment corresponds to a state where some fingers, here index and thumb, "contribute" to object rotation. They remain in contact with the object, with increased contact forces and no relative angular velocity. The next segment is an "updating" state for a finger, here the thumb. The finger moves to get to the next position as its relative angular velocity increases, whereas the other fingers remain in contact to support the object. Similarly, the other segment is an "updating" state for other "contributor finger," index.

3) Model Training: We applied HDP-HMM on the recorded $\mathcal{D}_o$ of object's rotations (finger gaiting), where we stored fingers' relative angular velocities ω and pressures p as observations with ${}^o\boldsymbol{\psi}_t^{(i)} = [\omega_1^T, p_1, \cdots, \omega_4^T, p_4]^T$. Subsequently, an HMM model is derived over 100 computation trials of 2000 iteration step. The HMM model extracted 3 hidden states (see Figure 20), each corresponding to a task segment. From Figure 20, we observe that some fingers (index and thumb) contribute to task of object rotation, mostly in one segment, "contributing" state, and that, in the other segments, these fingers update their position, "updating" state.

Farshad Khadivar received the M.Sc. degree in control systems from Sharif University of Technology, Iran, in 2017 and a Ph.D. degree in the field of robotics and artificial intelligence from the Swiss Federal Institute of Technology in Lausanne (EPFL), Lausanne, Switzerland, in 2022. He is currently a senior researcher with the Learning Algorithms and Systems Laboratory (LASA) at EPFL. His research interest is mainly in developing machine learning solutions and control theories for robot manipulation.

Aude Billard received the M.Sc. degree in physics from the Swiss Federal Institute of Technology in Lausanne (EPFL), Lausanne, Switzerland, in 1995, and an M.Sc. degree in knowledge-based systems and a Ph.D. degree in artificial intelligence from the University of Edinburgh, Edinburgh, U.K., in 1996 and 1998, respectively. She is currently a full professor at the Institutes of Micro and Mechanical Engineering and the head of the Learning Algorithms and Systems Laboratory, School of Engineering, EPFL. Her research interests are in machine-learning methods for making robots adaptive and able to learn through human guidance and practice.