

## APPENDIX A RELATED WORK

### A. General Multi-Fingered In-Hand Manipulation

Without explicit constraints of contacts, the general in-hand manipulation tries to manipulate the object through any possible finger motions and contact sequences.

Recently, learning-based approaches have been successfully applied to dexterous manipulation. Reinforcement learning (RL) has shown excellent performance for in-hand manipulation that requires complex dynamic finger gaiting, with the in-hand object rotation being a representative task [10]–[13]. By training on large-scale interactions, RL-based approaches can learn complex multi-fingered behaviors that are difficult to model and plan. However, RL can be costly for in-grasp manipulation. First, it must rely on inefficient random exploration during the early training phase to learn how to maintain a constant stable grasp [14], which is a prerequisite for subsequent object movement. Second, generalization to arbitrary goals and objects is expensive to guarantee, as it can only be improved by appreciably expanding the training dataset to implicitly cover any possible test case [12]. Third, unlike in-hand rotation that allows for larger tolerance in acceptable finger motions, in-grasp object movement requires high precision for arbitrary goal poses, which is an objective at which RL does not excel.

Human skill and experience can also be leveraged in dexterous manipulation, e.g., through imitation learning (IL) [15]–[17]. Although IL performs well on daily tasks whose goals are hard to mathematically define, it is less effective for precise in-grasp object movement, as precisely controlling the object by teleoperation is difficult for humans lacking complete in-hand feedback. The generalization is also difficult to guarantee.

In parallel, considerable works have been done on model-based in-hand manipulation, which can be divided into two categories. The first is contact-explicit approaches, which first explicitly search for discrete contact sequences and then control the hand to track the planned contacts [18]–[20]. Our approach for in-grasp manipulation falls into this category, with contacts being predefined by the initial grasp. The second category is contact-implicit approaches, which use unified contact-motion models with contact smoothing to efficiently explore potential finger motions and resulting contacts [21]–[23]. These approaches excel at efficiently determining contact sequences but sacrifice physical fidelity, making them less suitable for precise in-grasp manipulation.

### B. In-Grasp Manipulation

As a typical type of in-hand manipulation, in-grasp manipulation refers to controlling the object’s pose within a stable grasp. One of the key challenges is to strictly maintain a constant stable grasp while moving the object. A series of representative works [2]–[4] on in-grasp object movement used self-designed under-actuated fingers with compliant passive joints (enabled by springs), which could naturally adapt to the in-hand object and ensure a stable grasp during random finger motion. Although such end-effectors are effective for in-grasp object movement, they may lack versatility for other

types of in-hand manipulation. In contrast, we use an open-sourced, full-actuated anthropomorphic hand, the Leap Hand [24], which is a low-cost generic end-effector widely used for various tasks. Consequently, explicitly constraining the computed joint motion to maintain a constant stable grasp is essential. Additionally, they used a local mapping from robot motion to object motion to iteratively reach local goals (usually  $\leq 2$  cm). In contrast, we optimize the full trajectory to arbitrary large-range goals within a  $5 \times 5 \times 5$  (cm) cubic space, as required by the RGMC.

Classically, the constraint of constant stable grasping is usually analyzed at the velocity level. Some works used the velocity constraint to rigorously convert the desired object velocity into finger joint velocities [25] or convert the desired object velocity and acceleration into finger joint torques [26]. These approaches were validated in simulation and not easy to deploy in the real world, since they required fully predefined object trajectories and accurate system information, including contact surface parameters or hand-object dynamics.

Our approach was initially inspired by [5], [6] that were based on purely kinematic trajectory optimization with pose constraints. To generate in-grasp trajectories, they assumed rigid contact between the thumb-tip and object (i.e., an invariant relative pose) and imposed relaxed-rigidity constraints on the other fingertips (i.e., penalizing changes in fingertip poses in the thumb-tip frame). However, the assumption of rigid thumb-tip contact is overly restrictive, as it completely forbids rolling between the thumb-tip and object. This severely limits the object’s moving space owing to the low DoFs and joint limit of the thumb. In contrast, we allow rolling contacts of all fingertips by fixing only the fingertip positions in the object frame, which enlarges the reachable space of the object.

Although we allow fingertip rolling, we do not include the rigorous rolling constraint in the optimization like a recent work [27] that required the precise geometry of the object and fingertips. Moreover, the velocity-level geometry-aware rolling constraint greatly increases the complexity of the optimization, making it impractical for real-time deployment (e.g., requiring around 8 seconds per model predictive control step in [27]).

## APPENDIX B ANALYTICAL GRADIENTS OF THE OPTIMIZATION

In this section, we introduce the key analytical gradients of the trajectory optimization problem.

### A. Preliminaries

According to the theory of Lie groups and Lie algebra for robotics [9], we denote the conversion from the axis-angle rotation vector  $\mathbf{r} \in \mathfrak{so}(3)$  to the rotation matrix  $\mathbf{R} \in \text{SO}(3)$  as  $\mathbf{R} = \exp(\mathbf{r}^\wedge)$ , and the inverse conversion is denoted as  $\mathbf{r} = \ln(\mathbf{R})^\vee$ .

According to the linear approximation of the Baker-Campbell-Hausdorff (BCH) formula [28], we have

$$\ln(\exp(\mathbf{r}_1^\wedge) \exp(\mathbf{r}_2^\wedge))^\vee \approx \begin{cases} \mathbf{J}_l(\mathbf{r}_2)^{-1} \mathbf{r}_1 + \mathbf{r}_2, & \text{when } \mathbf{r}_1 \rightarrow \mathbf{0}, \\ \mathbf{J}_r(\mathbf{r}_1)^{-1} \mathbf{r}_2 + \mathbf{r}_1, & \text{when } \mathbf{r}_2 \rightarrow \mathbf{0}. \end{cases} \quad (6)$$

Here,  $\mathbf{J}_r(\mathbf{r}) = \mathbf{J}_l(-\mathbf{r})$ , and  $\mathbf{J}_l(\mathbf{r})$  can be calculated as

$$\mathbf{J}_l(\mathbf{r}) = \frac{\sin \theta}{\theta} \mathbf{I} + \left(1 - \frac{\sin \theta}{\theta}\right) \mathbf{a} \mathbf{a}^\top + \frac{1 - \cos \theta}{\theta} \mathbf{a}^\wedge, \quad (7)$$

$$\mathbf{J}_l(\mathbf{r})^{-1} = \frac{\theta}{2} \cot \frac{\theta}{2} \mathbf{I} + \left(1 - \frac{\theta}{2} \cot \frac{\theta}{2}\right) \mathbf{a} \mathbf{a}^\top - \frac{\theta}{2} \mathbf{a}^\wedge, \quad (8)$$

where  $\theta$  and  $\mathbf{a}$  are the angle and axis of  $\mathbf{r}$ , respectively.

### B. Gradients of Orientation Distances

Here we generally introduce the gradient regarding the orientation distance. We define the weighted scalar distance of orientations  $\mathbf{R}$  and  $\mathbf{R}_d$  as

$$d_r(\mathbf{R}, \mathbf{R}_d, \mathbf{W}_r) = \frac{1}{2} \mathbf{r}_e^\top \mathbf{W}_r \mathbf{r}_e, \quad (9)$$

where  $\mathbf{r}_e$  is defined as  $\ln(\mathbf{R} \mathbf{R}_d^{-1})^\vee$ . Note that  $\mathbf{r}_e$  is defined in the world frame  $\mathcal{W}$ . In addition,  $\mathbf{W}_r$  is a semi-positive definite matrix for weighting. Here,  $\mathbf{R}$  is determined by a variable  $\mathbf{x}$ , and  $\mathbf{R}_d$  is a constant desired orientation. Note that  $\mathbf{r}_e$  is defined in the same frame as  $\mathbf{R}$  and  $\mathbf{R}_d$ .

In the following derivation, for convenience, we use both  $\mathbf{R} \in \text{SO}(3)$  and  $\mathbf{r} \in \mathfrak{so}(3)$  to represent the same rotation; e.g.,  $\mathbf{R} = \exp(\mathbf{r}^\wedge)$  and  $\mathbf{R}_d = \exp(\mathbf{r}_d^\wedge)$ .

The gradient of  $d_r$  w.r.t. the variable  $\mathbf{x}$  is derived as

$$\frac{\partial d_r}{\partial \mathbf{x}} = \frac{\partial d_r}{\partial \mathbf{r}_e} \frac{\partial \mathbf{r}_e}{\partial \mathbf{x}} = \frac{\partial d_r}{\partial \mathbf{r}_e} \frac{\partial \mathbf{r}_e}{\partial \boldsymbol{\varphi}} \frac{\partial \boldsymbol{\varphi}}{\partial \mathbf{x}} \quad (10)$$

For convenience of the following calculation, here we introduce a perturbation variable  $\boldsymbol{\varphi} \in \mathfrak{so}(3)$ , which means that we left perturb  $\mathbf{R}$  by  $\Delta \mathbf{R}$  (i.e.,  $(\Delta \mathbf{R}) \mathbf{R}$ ), where  $\Delta \mathbf{R} = \exp(\boldsymbol{\varphi}^\wedge)$ .

First, it is easy to obtain

$$\frac{\partial d_r}{\partial \mathbf{r}_e} = \mathbf{r}_e^\top \mathbf{W}_r \quad (11)$$

Second, regarding  $\frac{\partial \mathbf{r}_e}{\partial \boldsymbol{\varphi}}$ , we have

$$\begin{aligned} \frac{\partial \mathbf{r}_e}{\partial \boldsymbol{\varphi}} &= \lim_{\boldsymbol{\varphi} \rightarrow \mathbf{0}} \frac{\ln \left( \exp(\boldsymbol{\varphi}^\wedge) \exp(\mathbf{r}^\wedge) (\exp(\mathbf{r}_d^\wedge))^{-1} \right)^\vee}{\boldsymbol{\varphi}} \\ &\quad - \frac{\ln \left( \exp(\mathbf{r}^\wedge) (\exp(\mathbf{r}_d^\wedge))^{-1} \right)^\vee}{\boldsymbol{\varphi}} \\ &= \lim_{\boldsymbol{\varphi} \rightarrow \mathbf{0}} \frac{\ln(\exp(\boldsymbol{\varphi}^\wedge) \exp(\mathbf{r}_e^\wedge))^\vee - \ln(\exp(\mathbf{r}_e^\wedge))^\vee}{\boldsymbol{\varphi}} \end{aligned} \quad (12)$$

It follows from the BCH formula (6) that

$$\frac{\partial \mathbf{r}_e}{\partial \boldsymbol{\varphi}} = \lim_{\boldsymbol{\varphi} \rightarrow \mathbf{0}} \frac{\mathbf{J}_l(\mathbf{r}_e)^{-1} \boldsymbol{\varphi} + \mathbf{r}_e - \mathbf{r}_e}{\boldsymbol{\varphi}} = \mathbf{J}_l(\mathbf{r}_e)^{-1} \quad (13)$$

We thus obtain  $\frac{\partial d_r}{\partial \boldsymbol{\varphi}} = \mathbf{r}_e^\top \mathbf{W}_r \mathbf{J}_l(\mathbf{r}_e)^{-1}$ . Moreover, using (8), we can obtain  $\mathbf{r}_e^\top \mathbf{J}_l(\mathbf{r}_e)^{-1} = \mathbf{r}_e^\top$ . Thus, in special cases where the weights for each orientation dimension are the same (i.e.,  $\mathbf{W}_r = w \mathbf{I}$ ), we further have  $\frac{\partial d_r}{\partial \boldsymbol{\varphi}} = w \mathbf{r}_e^\top \mathbf{J}_l(\mathbf{r}_e)^{-1} = w \mathbf{r}_e^\top$ .

Third, as the perturbation variable  $\boldsymbol{\varphi} \rightarrow \mathbf{0}$ , we have  $\frac{\partial \boldsymbol{\varphi}}{\partial \mathbf{x}} = \mathbf{J}_a(\mathbf{x})$ , where  $\mathbf{J}_a(\mathbf{x})$  is the space Jacobian that relates the spatial angular velocity to  $\dot{\mathbf{x}}$ .

### C. Gradients of Pose Distances

Similar to the derivation of the gradients of orientation distances, the general formula of the gradients of pose distances is derived as follows.

The weighted scalar distance between poses  $\mathbf{T}$  and  $\mathbf{T}_d$  is defined as

$$d(\mathbf{T}, \mathbf{T}_d, \mathbf{W}) = \frac{1}{2} \mathbf{e}^\top \mathbf{W} \mathbf{e} \quad (14)$$

where  $\mathbf{e} = [\mathbf{p}_e; \mathbf{r}_e]$ , in which  $\mathbf{p}_e = \mathbf{p} - \mathbf{p}_d$  and  $\mathbf{r}_e = \ln \left( \exp(\mathbf{r}^\wedge) (\exp(\mathbf{r}_d^\wedge))^{-1} \right)^\vee$ . Here,  $\mathbf{T}$  is determined by a variable  $\mathbf{x}$ , and  $\mathbf{T}_d$  is a constant desired pose. Note that  $\mathbf{e}$  is defined in the same frame as  $\mathbf{T}$  and  $\mathbf{T}_d$ .

Similar to (10), we introduce a perturbation variable  $\boldsymbol{\phi} \in \mathfrak{se}(3)$ . Then, the gradient of  $d$  w.r.t.  $\mathbf{x}$  is derived as

$$\frac{\partial d}{\partial \mathbf{x}} = \frac{\partial d}{\partial \mathbf{e}} \frac{\partial \mathbf{e}}{\partial \boldsymbol{\phi}} \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{x}} \quad (15)$$

where  $\frac{\partial d}{\partial \mathbf{e}} = \mathbf{e}^\top \mathbf{W}$  and

$$\frac{\partial \mathbf{e}}{\partial \boldsymbol{\phi}} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_l(\mathbf{r}_e)^{-1} \end{bmatrix} \quad (16)$$

Additionally, we have  $\frac{\partial \boldsymbol{\phi}}{\partial \mathbf{x}} = \mathbf{J}(\mathbf{x})$ , where  $\mathbf{J}(\mathbf{x})$  is the space Jacobian that relates the spatial twist to  $\dot{\mathbf{x}}$ .

### D. Gradients of $\mathcal{J}_{\text{object}}$

It is easy to see that  $\mathcal{J}_{\text{object}}$  is only relevant to the object pose at time  $T$ . The gradient between the position distance cost and the object position variable is easy to derive. Here, we introduce the gradient regarding the orientation distance (i.e.,  $\frac{\partial d_r}{\partial \mathbf{r}_{o,T}}$ ). For brevity, we omit the subscripts  $o$  and  $T$ .

Similar to (12) and (13), we derive that

$$\frac{\partial \boldsymbol{\varphi}}{\partial \mathbf{r}} = \left( \frac{\partial \mathbf{r}}{\partial \boldsymbol{\varphi}} \right)^{-1} = \mathbf{J}_l(\mathbf{r}) \quad (17)$$

We then have

$$\frac{\partial d_r}{\partial \mathbf{r}} = \mathbf{r}_e^\top \mathbf{W}_r \mathbf{J}_l(\mathbf{r}_e)^{-1} \frac{\partial \boldsymbol{\varphi}}{\partial \mathbf{r}} = \mathbf{r}_e^\top \mathbf{W}_r \mathbf{J}_l(\mathbf{r}_e)^{-1} \mathbf{J}_l(\mathbf{r}) \quad (18)$$

### E. Gradients of $\mathcal{J}_{\text{finger}}$

We denote the Lie algebra corresponding to the object pose  $\mathbf{T}_{o,t} \in SE(3)$  as  $\boldsymbol{\xi}_{o,t} = [\mathbf{p}_{o,t}; \mathbf{r}_{o,t}] \in \mathfrak{se}(3)$ , which is defined in  $\mathcal{W}$ . The optimization variable related to  $d(\mathcal{O} \mathbf{T}_{i,t}, \mathcal{O} \mathbf{T}_{i,0}, \mathbf{W}_f)$  contains the object pose  $\boldsymbol{\xi}_{o,t}$  and finger joint angle  $\mathbf{q}_{i,t}$ . For brevity, we omit the subscripts  $o$ ,  $i$ , and  $t$ . We further denote  $\boldsymbol{\vartheta} = [\boldsymbol{\xi}; \mathbf{q}]$ .

We can use (15) to calculate the gradient, but we still need to know the space Jacobian that relates the fingertip twist in  $\mathcal{O}$  to  $\dot{\boldsymbol{\vartheta}}$ . As the object frame  $\mathcal{O}$  is moving, this Jacobian is a relative Jacobian between the finger and object. We can calculate this relative Jacobian using individual manipulator Jacobians defined in  $\mathcal{W}$  [29], in which we regard the object as a virtual manipulator. According to (2) in [29], the relative Jacobian between the fingertip twist in  $\mathcal{O}$  and  $\dot{\boldsymbol{\vartheta}}$  can be expressed as

$$\mathcal{O} \mathbf{J}_f(\boldsymbol{\vartheta}) = \begin{bmatrix} -\mathcal{O} \boldsymbol{\Psi}_f \mathcal{O} \boldsymbol{\Omega}_w \mathbf{J}_o(\boldsymbol{\xi}) & \mathcal{O} \boldsymbol{\Omega}_w \mathbf{J}_f(\mathbf{q}) \end{bmatrix}, \quad (19)$$

where  $\mathbf{J}_o(\boldsymbol{\xi})$  is the space Jacobian that relates the object's twist in  $\mathcal{W}$  to  $\boldsymbol{\xi}$ , and  $\mathbf{J}_f(\mathbf{q})$  is the space Jacobian that relates the

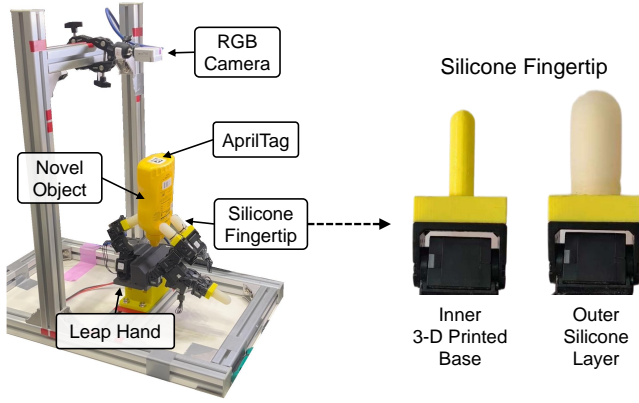


Fig. 6. Our hardware setup for the competition, comprising a Leap Hand with soft silicone fingertips, a top-view RGB camera, and an in-hand object with an AprilTag marker. Each silicone fingertip comprises an inner 3-D printed base and an outer silicone layer.

fingertip's twist in  $\mathcal{W}$  to  $\dot{q}$ . Similar to (17), it can be obtained that

$$J_o(\xi) = \begin{bmatrix} I & 0 \\ 0 & J_l(r) \end{bmatrix}, \quad (20)$$

where  $r$  refers to the object orientation in  $\mathcal{W}$ . The finger Jacobian  $J_f(\cdot)$  can be obtained from the finger's kinematics. The transformation matrices  $\Psi$  and  $\Omega$  are defined as

$${}^a\Psi_b = \begin{bmatrix} I & -S({}^ap_b) \\ 0 & I \end{bmatrix}, \quad {}^a\Omega_b = \begin{bmatrix} {}^aR_b & 0 \\ 0 & {}^aR_b \end{bmatrix}, \quad (21)$$

where  $S(p)$  refers to the skew-symmetric matrix of vector  $p$ .

#### F. Other Gradients

Other gradients, including the gradients of  $\mathcal{J}_{\text{joint}}$  and those of the constraints, can be easily derived. The details are omitted here for brevity.

### APPENDIX C IMPLEMENTATION DETAILS

#### A. Hardware Setup

The hardware setup is shown in Fig. 6. A calibrated top-view RGB camera (RealSense d405) is used to obtain the AprilTag pose. We use a Leap Hand [24], which features four fingers, each with four actuated DoFs. The control commands to the hand are the target joint angles, which are tracked by a low-level PD controller. We only use the thumb, index, and ring fingers for this task, as three fingers with soft frictional contacts are sufficient for a stable grasp. Including additional fingers increases the risk of self-collision and reduces the overall finger workspace. To enhance contact compliance and friction, we replace the original fingertips with custom-made soft silicone fingertips. The contacts between these hemispherical fingertips and the object approximate the point contacts with friction. From a hardware perspective, both the low-level PD controller and the soft fingertips provide physical compliance, facilitating stable grasping during manipulation.

#### B. Optimization Solving

We solve this non-convex constrained optimization problem using the Sequential Least Squares Programming (SLSQP) [30] algorithm implemented by the `scipy.optimize.minimize` in Python.

#### C. Initial Grasping

The quality of the initial grasp is critical for the subsequent in-grasp object movement. For the known cylinder, we manually define the target fingertip positions for the initial grasp. To efficiently obtain an initial grasp of a novel object without on-site coding, we develop a human-dragging approach, in which a human can freely move the fingers along a horizontal plane, while the algorithm constrains the fingertip heights to keep them on the same plane. This enables the human to simultaneously and conveniently adjust the three fingers to establish and record a stable grasp. Then, the target fingertip positions are set further inside the object surface with a predefined offset. This ensures that the fingertips apply appropriate grasping forces, taking advantage of the compliance provided by the low-level PD controller and soft fingertips. Finally, we use an optimization-based inverse kinematics (IK) solver [31] to compute the corresponding joint-space position command.

### APPENDIX D

#### ADDITIONAL DETAILS OF THE COMPETITION

##### A. Hyper-Parameters

The hyper-parameters we used in the competition were set as  $W_o = \text{diag}(10, 10, 10, 0.01, 0.01, 0.0)$ , and  $W_f = \text{diag}(10, 10, 10, 0.001, 0.001, 0.001)$ ; for the first trajectory optimization for each waypoint, we set  $T = 3$  and  $\lambda = 4e - 4$ ; for the re-planning, we set  $T = 1$  and  $\lambda = 5e - 3$ . These parameters are also used for the experimental evaluation in Section V.

As analyzed in Section V-B, for the competition, we set  $N_{\text{replan}} = 4$  in the first run to ensure more conservative results and  $N_{\text{replan}} = 8$  in the second run to aim for higher precision.

##### B. Performance Results

The performance details of our approach in the RGMC are provided by the competition organizers, which are summarized in the following tables. Specifically, the positions of the ten goal waypoints in the competition are listed in Table I, ranging from  $(-2.5, -2.5, -2.5)$  to  $(2.5, 2.5, 2.5)$  (cm). The average task error of the ten waypoints in each run is shown in Table II. Note that the task error of each individual waypoint is not provided separately, as the organizer did not record them in detail. From the results, it can be seen that the precision of the second run is higher than that of the first run. This improvement is attributed to the different choices of  $N_{\text{replan}}$ .

### APPENDIX E

#### ADDITIONAL RESULTS AND ANALYSIS

##### A. Comparison with Existing Approach

Given our task setup involving full-actuated hands, unknown object trajectories, and novel objects, we implement a baseline

TABLE I  
TEN GOAL WAYPOINTS IN THE COMPETITION.

Waypoint index	Position (x, y, z) (cm)
1	(2.5, 2.5, 0)
2	(2.5, 2.5, 2.5)
3	(-2.5, -2.5, -2.5)
4	(-1.3, -2.0, 0.6)
5	(-1.2, 0.7, 0.6)
6	(0.6, 0.4, 0.2)
7	(0.9, -1.2, -1.3)
8	(-2.0, 2.0, 2.0)
9	(0.0, 0.0, 2.0)
10	(0.0, 0.0, 0.0)

TABLE II  
FINAL RESULTS IN THE COMPETITION.

		Average task error (cm)
Cylinder Object	Run 1	0.080
	Run 2	0.054
Novel Object <sup>a</sup>	Run 1	0.125
	Run 2	0.063

<sup>a</sup> A mustard bottle from the YCB Dataset, as shown in Fig. 6)

similar to [5] for comparison, which performed the best in the benchmark from [32]. Regarding the specific implementation, we adopt the same framework as our approach, whereas the optimization variables include only joint angles, and the object pose is derived from the thumb-tip pose under the rigid contact assumption. We compare the planned errors, open-loop execution errors, and closed-loop execution errors in continuous five-iteration reaching of the eight corners of both  $3 \times 3 \times 3$  and  $5 \times 5 \times 5$  (cm) cubic spaces, using the cylinder object. We set  $N_{\text{replan}} = 8$  for the space with a 3-cm side length, and  $N_{\text{replan}} = 4$  for the space with a 5-cm side length.

The results are summarized in Fig. 7. It can be seen that: 1) the planned error is smaller in our approach than in the baseline, since the assumption of rigid object-thumb contact in the baseline limits the theoretical reachable space of the object, whereas our formulation allows rolling contacts; 2) both the open-loop and closed-loop execution errors in our approach are smaller than those in the baseline, demonstrating that our

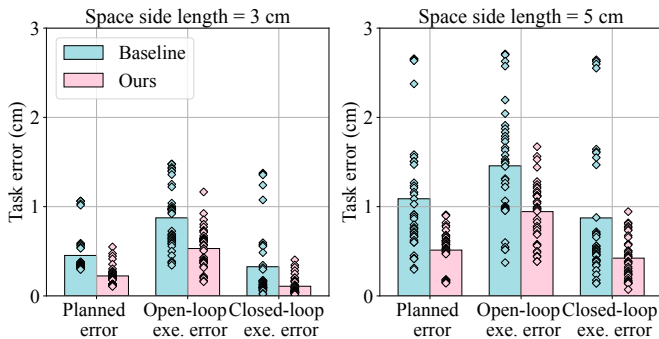


Fig. 7. Comparison between the proposed approach and the baseline [5], which assumes rigid object-thumb contact, whereas we allow for rolling. Each bar shows the average error across 40 waypoints at the corners of the cubic space with side lengths of 3 or 5 cm, and the values of each waypoint are also plotted as the scattered diamond-shaped points.

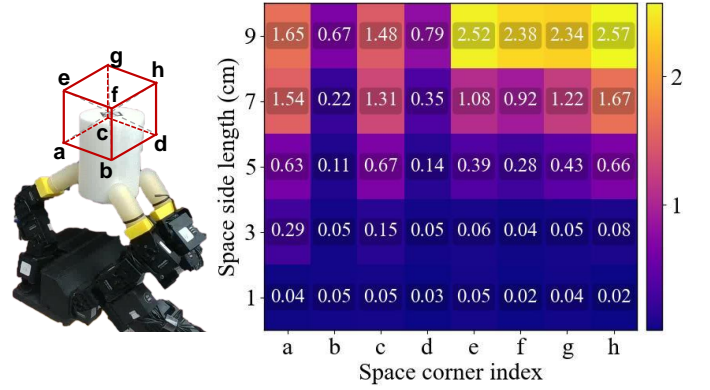


Fig. 8. Evaluation of the object's reachable space and the relationship between goal positions and task accuracy. The left figure illustrates the cubic movement space for the object, with indexed corners and centered at the initial object position. For reference, the side length of the plotted cubic is approximately 6 cm. The right figure summarizes the closed-loop execution error for each corner of spaces with side lengths ranging from 1 to 9 cm. The error is averaged over five trials for each goal.

approach also improves the actual execution accuracy in real-world scenarios; and 3) in the baseline, the execution accuracy at certain corners of the space (e.g., the waypoints with the largest errors) is not significantly improved by closed-loop execution, suggesting that the task errors for these movement directions are primarily due to the limited theoretical reachable space rather than the sim-to-real gap. These performance results motivated us to develop our proposed approach for the competition, instead of relying on the baseline.

### B. Reachable Space and Accuracy

We analyze the relationship between the distances to goals and the task accuracy, and further explore the maximum object reachable space. The cylinder object is manipulated to reach the corners of cubic spaces with side lengths ranging from 1 to 9 cm. The task becomes increasingly challenging as the distance to the goal increases. Each corner of each cubic space is reached five times. For the cubic spaces with side lengths of 7 and 9 cm, we manually adjust the grasp when necessary between goals, as slippage may occur during such large-range movements, potentially affecting subsequent manipulation. We manually choose appropriate values of  $N_{\text{replan}}$  for different goals to achieve the best performance.

The average task error of each goal position and the spatial relationship between the goals and hand is presented in Fig. 8. The results indicate that: 1) the task accuracy for local goals is very high, with the average error in reaching each corner of the  $1 \times 1 \times 1$  (cm) space being no larger than 0.5 mm; 2) the task error increases with the distance to the goal; 3) movements in certain directions are more challenging; e.g., in the  $5 \times 5 \times 5$  (cm) space, the average task errors for Corners a, c, and h are larger than those for other corners, due to the asymmetrical finger layout and the distinct mechanical configurations of the thumb and other fingers; and 4) our approach can reach the boundary of a  $9 \times 9 \times 9$  (cm) space with an average error of approximately 2 cm, demonstrating its ability to fully exploit the dexterity and workspace of the fingers to achieve goals as accurately as possible within a large in-hand space.



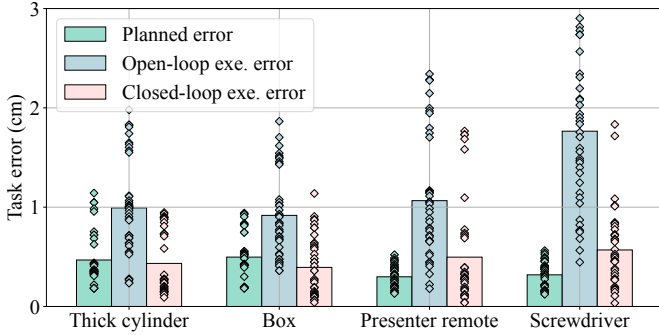


Fig. 9. Evaluation of the generalization of our approach to novel everyday objects. Each bar shows the average error over 40 waypoints on the corners of the  $5 \times 5 \times 5$  (cm) space, and the values of each waypoint are also plotted by the scattered diamond-shaped points.

### C. Novel Objects

We further evaluate the generalization of our approach to novel everyday objects, including a thick cylinder lid, box, presenter remote, and screwdriver, as shown in Fig. 3(b)-(e). We task the hand to manipulate each object to continuously reach the eight corners of the  $5 \times 5 \times 5$  (cm) space over five iterations. Only for the screwdriver do we occasionally manually adjust the grasp between different goals, due to its thin structure and highly curved surface.

The results in Fig. 9 indicate that 1) the proposed approach generalizes well to novel everyday objects, with the average task error for each object being approximately 5 mm, even for the challenging screwdriver; and 2) the curvature of the object surface affects the task accuracy, as high curvature increases the effect of fingertip rolling on contact situations, potentially leading to errors or even unstable grasps. From our experiments, we conclude that objects with lower weights and curvatures are easier to manipulate using our approach, whereas objects with higher weights and curvatures present greater challenges. This is because increased weight tends to cause slippage, and high curvature causes variations in the grasp contact during rolling.

### D. Analysis of Task Error Variance

The diamond points within Figs. 5, 6, 7 and 9 represent the task error of each waypoint along with the summary statistic. Overall, the variance in task errors can be attributed to factors such as different cubic corners, different iterations, and slight differences between initial grasps. We further investigate the variance through extensive experiments. Specifically, we apply the proposed approach to manipulate the object to the corners of the  $5 \times 5 \times 5$  (cm) cubic space over five iterations without human intervention. This process was repeated five times, with the initial grasp reset by a human before each trial. We provide a video of four initial grasps at [url<sup>1</sup>](https://rgmc-xl-team.github.io/ingrasp_manipulation/initial_grasp.mp4).

Fig. 10(a) and (b) summarize the average task errors for different runs and different cubic corners. The results indicate that 1) the average task errors of different runs are relatively consistent, with the gap between the largest and smallest closed-loop error being less than 0.08 cm, demonstrating that

the slight differences in initial grasps have little impact on the overall statistic results; and 2) certain corners (e.g., a, c, h) exhibit averagely larger errors, as discussed in Section E-B. Then, we further investigate the variance among different runs for each corner, as shown in Fig. 10(c). It can be seen that the planned errors for the same corner remains relatively consistent; in contrast, the execution errors for the same corner vary across different runs and iterations. This variability arises from minor differences in the manually established initial grasps and slight changes in the grasp (e.g., slippage) during continuous manipulation.

### E. Effect of Moving Back to Initial State

As described in Section III-B, we employ a strategy where the fingers return to the initial state (following the forward trajectory) after reaching each waypoint. This approach is adopted because the initial state typically provides a more favorable starting point for trajectory optimization toward the next goal. To experimentally evaluate the effectiveness of this strategy, we task the hand with manipulating the object to the corners of the  $5 \times 5 \times 5$  (cm) cubic space over five iterations. We compare the performance of the approach with and without this strategy, conducting five trials for each condition. An example of the manipulation process without returning to the initial state is shown in a video at [url<sup>2</sup>](https://rgmc-xl-team.github.io/ingrasp_manipulation/not_move_back.mp4).

When using this strategy, the average task error of the first iteration (8 corners) is 0.40 cm, and the average task error of all five iteration (40 waypoints) is 0.42 cm. When not using this strategy, the average task error of the first iteration is 0.47 cm; however, the object falls in the second iteration in all five tests due to low contact quality. These results indicate that employing this strategy can enhance the robustness in continuous waypoint reaching.

### F. Impact of Excessive Re-Planning

In Section IV-B, we point out that excessive re-planning may degrade contact quality and lead to larger task errors. This problem is primarily due to the “initial state” of the trajectory optimization. We assume the initial grasp is a stable and manipulable grasp, which holds true for human-designed initial grasps. However, during replanning, the “initial state” of the new optimization problem is set as the terminal state from the previous execution. Note that our simple trajectory optimization problem formulation does not explicitly constrain the quality of the finger-object contacts. Instead, it tries to implicitly maintain the contact quality by softly penalizing deviations between the terminal and initial configuration (through the cost term  $\mathcal{J}_{\text{finger}}$  and  $\mathcal{J}_{\text{joint}}$ ). Consequently, the terminal state from the previous execution may not be as good a grasp as the initial state, leading to a gradual decline in contact quality as the number of replanning iterations increases.

We provide an example to illustrate this problem in Fig. 11. It is shown that when applying too many replanning times to reach the goal waypoint, unmodeled contact occurs between the object and non-spherical parts of the fingers, which leads to significant slippage.

<sup>1</sup>[https://rgmc-xl-team.github.io/ingrasp\\_manipulation/initial\\_grasp.mp4](https://rgmc-xl-team.github.io/ingrasp_manipulation/initial_grasp.mp4)

<sup>2</sup>[https://rgmc-xl-team.github.io/ingrasp\\_manipulation/not\\_move\\_back.mp4](https://rgmc-xl-team.github.io/ingrasp_manipulation/not_move_back.mp4)

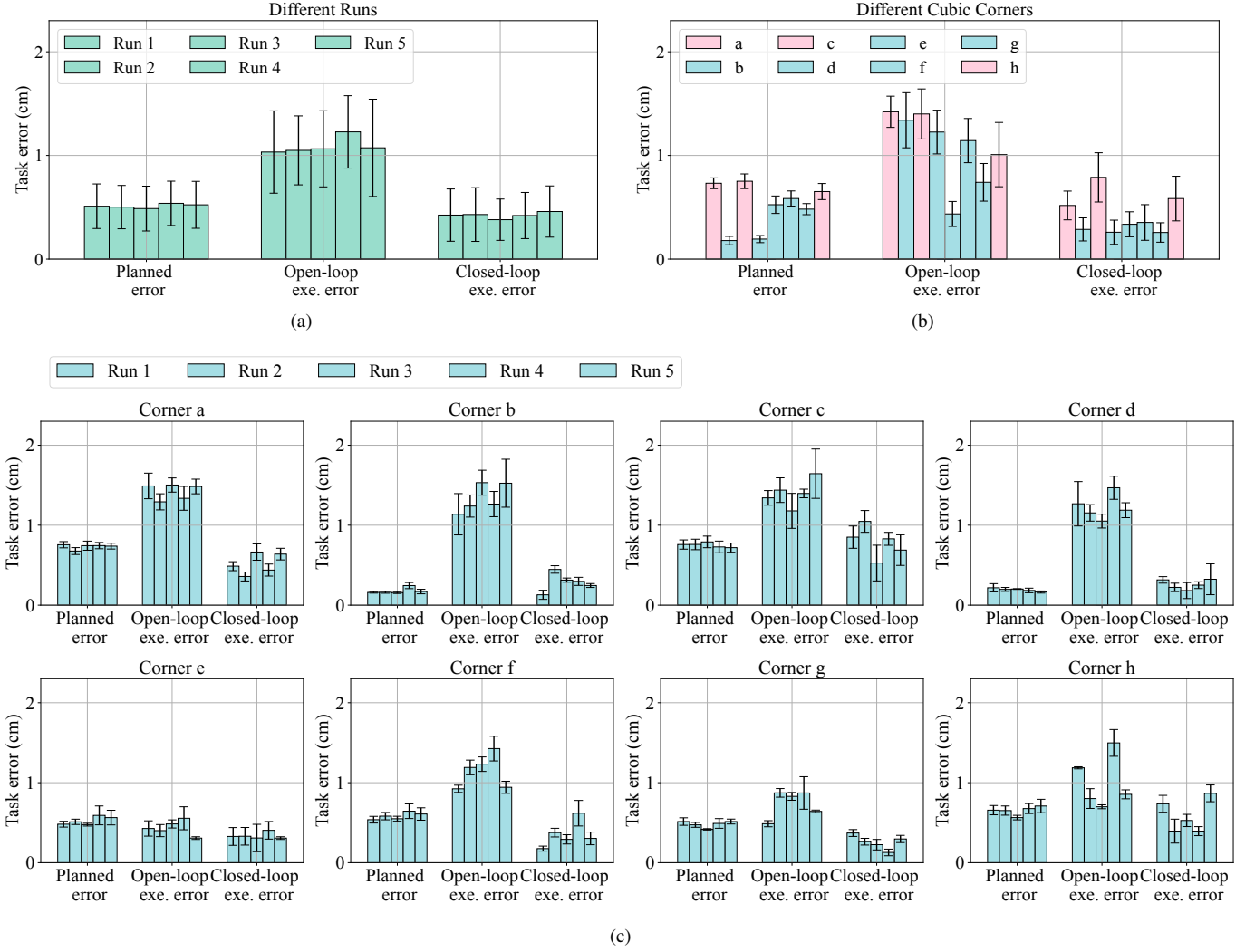


Fig. 10. Variance of task errors. (a) Variance w.r.t. different runs, where each bar represents the average error over 40 waypoints (five iterations of eight corners) in each run. (b) Variance w.r.t. different cubic corners, where each bar represents the average error over 25 attempts (five iterations in five runs) at each corner. (c) Variance w.r.t. different runs for each corner, where each bar represents the average error over five iterations for each corner in each run. The error bars represent the standard deviation.

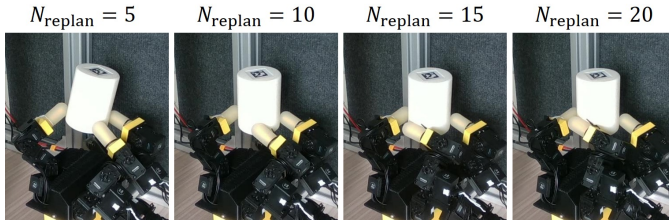


Fig. 11. Example of applying excessive re-planning.

### G. Effect of Weights for Object Pose Cost

In the competition, we empirically used  $\mathbf{W}_o = \text{diag}(10, 10, 10, 0.01, 0.01, 0.0)$  for the object pose cost  $\mathcal{J}_{\text{object}}$ . We set the goal object orientation as the current (initial) orientation, and applied the non-zero weights to slightly regulate the rotation along the X and Y axes, which improved the manipulation robustness in our experience.

We further quantitatively evaluate the effect of

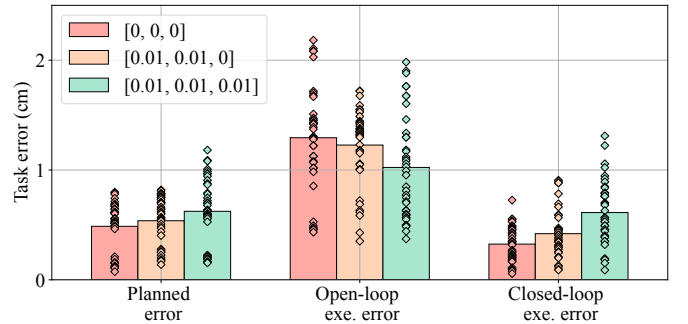


Fig. 12. Effect of different weights for the object orientation cost. Each bar shows the average error over 40 waypoints on the corners of the  $5 \times 5 \times 5$  (cm) space, and the values of each waypoint are also plotted by the scattered diamond-shaped points.

different choices of the weights for the orientation, including  $\text{diag}(0.0, 0.0, 0.0)$ ,  $\text{diag}(0.01, 0.01, 0.0)$ , and

TABLE III  
EVALUATION OF OUR APPROACH WITH GOALS OF OBJECT POSES.

Index	Goal translation (cm)	Goal rotation (degree)	Position error (cm)	Orientation error (degree)
1	(0, -1, -1)	(0, 20, 0)	0.22	1.7
2	(-2, 0, 0)	(30, 0, 0)	0.74	4.5
3	(0, 0, 2)	(0, 0, 40)	0.16	1.2

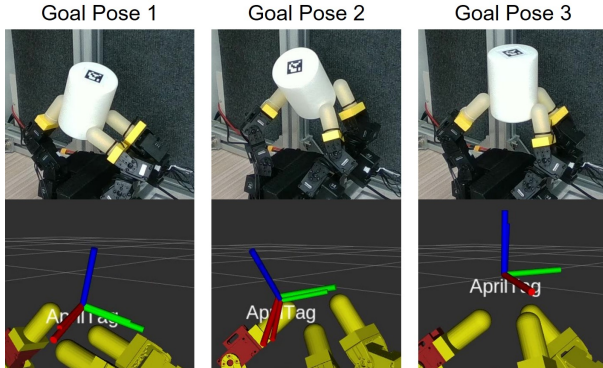


Fig. 13. Snapshots of reaching goal object poses. The figures in the second row visualize the poses, where the larger axes represent the goal poses and the smaller axes represent the poses of the AprilTag. The manipulation process is also shown in the supplementary video.

$\text{diag}(0.01, 0.01, 0.01)$ . The results of one run (40 waypoints) are shown in Fig. 12. It can be seen that 1) increasing the regulation on object orientation leads to larger planned and closed-loop errors, as it restricts the object's reachable space; however, it results in smaller open-loop errors, as smaller rotation generally reduces the risk of unexpected slippage during manipulation; and 2) regulating the Z-axis rotation further increases the closed-loop task errors, compared with regulating only the X and Y axes. Although the results of this run suggest that no regulation might yield higher precision, we find that significant slippage sometimes occur during continuous manipulation without any regulation. Consequently, we chose  $\text{diag}(0.01, 0.01, 0.0)$  for the competition as a trade-off between accuracy and robustness.

#### H. Goals of Object Poses

We further demonstrate our approach's capability to handle goals of object poses, which include both positions and orientations. The task involves continuous reaching three goal poses, for which we set  $\mathbf{W}_o = \text{diag}(10, 10, 10, 1, 1, 1)$ . The details of the goals and closed-loop manipulation results are listed in Table III. Snapshots of the manipulation process are shown in Fig. 13. The results demonstrate that our approach can be easily adapted to address goal orientations.

#### I. Implementation of Baseline

In Section E-A, we implement a baseline similar to [5] for comparison. Here we provide the details regarding the re-implementation. We adopt the same framework as our approach, whereas the optimization variables include only

joint angles, and the object pose is derived from the thumb-tip pose under the rigid contact assumption. Our implementation closely follows that in [5], with the following differences: 1) we do not include the cost of in-trajectory object poses, whose references are obtained by linear interpolation between the start and goal object poses in theirs; 2) we treat the joint velocity/movement limits as a soft constraint (penalty) instead of a hard constraint in theirs; and 3) we use the same hyperparameters as those in our approach.

#### APPENDIX F ACKNOWLEDGMENT

We sincerely thank Professor Kaiyu Hang, Professor Yu Sun, and all organizers of the RGMCM for their outstanding support and efforts in making the event a success. Their guidance and dedication were invaluable throughout the competition.