DexMachina: Functional Retargeting for Bimanual Dexterous Manipulation

Zhao Mandi[1], **Yifan Hou**[1], **Dieter Fox**[2], **Yashraj Narang**[2], **Ajay Mandlekar**[2,†], **Shuran Song**[1,†]

[1]Stanford University [2]NVIDIA [†]equal advising

Abstract: We study the problem of functional retargeting: learning dexterous manipulation policies to track object states from human hand-object demonstrations. We focus on long-horizon, bimanual tasks with articulated objects, which is challenging due to large action space, spatiotemporal discontinuities, and embodiment gap between human and robot hands. We propose DexMachina, a novel curriculum-based algorithm: the key idea is to use virtual object controllers with decaying strength: an object is first driven automatically towards its target states, such that the policy can gradually learn to take over under motion and contact guidance. We release a simulation benchmark with a diverse set of tasks and dexterous hands, and show that DexMachina significantly outperforms baseline methods. Our algorithm and benchmark enable a functional comparison for hardware designs, and we present key findings informed by quantitative and qualitative results. With the recent surge in dexterous hand development, we hope this work will provide a useful platform for identifying desirable hardware capabilities and lower the barrier for contributing to future research. Videos and more at `project-dexmachina.github.io`

Keywords: Dexterous Manipulation, Reinforcement Learning, Simulation-based Learning

Figure 1: **Functional Retargeting.** We study the problem of functional retargeting, where the goal is to retarget human hand demonstrations into functional dexterous robot policies that manipulate an object to follow the demonstrated trajectory. Our proposed algorithm, DexMachina, achieves functional retargeting from one human demonstration to a variety of existing dexterous hand embodiments over a range of articulated objects.

1 Introduction

Dexterous robot hands, with their resemblance to human hands, spark the expectation for achieving human-level dexterity. Yet the reality presents many hardware and algorithmic challenges that bottleneck the progress in dexterous manipulation. Prior learning-based methods have seen success in relatively simple and short-horizon tasks, but are often limited by manual reward-engineering [1, 2] or costly data-collection [3, 1] due to the embodiment gap between human hands and dexterous hands.

Human hands are hence a natural source for learning guidance. In this work, we formulate learning from human with an emphasis on task capability. We denote the problem as *functional retargeting*:

given a human demonstration, the goal is to learn dexterous hand policies that can manipulate the object to follow the demonstrated trajectory (see Fig. 1). This is distinguished from *kinematic* retargeting [3], which produces human-like motions without ensuring feasibility. The problem is even more compelling for long-horizon, bimanual demonstrations with articulated objects, which encompass a significant portion of daily human activities, but pose several key challenges: exploration is difficult under the high-dimensional action space, the intricate contact sequences demand stable and precise hand movements; due to the embodiment gap, human hand motion cannot be directly mapped to feasible robot actions, which limits the scalability of imitation data collection.

To address these challenges, we propose DexMachina[1], a novel curriculum-based RL algorithm for functional retargeting. Precise bimanual coordination is often required to manipulate an object successfully (e.g. opening a waffle iron mid-air, see Fig. 1), but naive approaches often get stuck in early failures or suboptimal actions. This motivates us to design a curriculum to allow the policy to explore in a less fragile setting. Our key idea is to use *virtual object controllers*—they apply control forces that drive an object towards its demonstrated trajectory—and *auxiliary motion and contact rewards*, which guide the policy to learn task strategies as the virtual controller strength decays. The policy first learns to mimic the human motion without worrying about failing the task, then learns to take over manipulation as the virtual controllers fade away.

Despite continuous effort in developing new hands and sensing capabilities [4, 5, 6, 7, 8, 9], there is a lack for standardized and accessible evaluation benchmarks. To address this, we build ChiralLabs , a simulation benchmark with a diverse set of 6 dexterous hands and 5 articulated objects [10], and provides a unified testbed where new hands and tasks can be easily added and quickly evaluated. On this benchmark, we empirically show that DexMachina significantly outperforms baseline methods, and applies successfully to a wide variety of hands, articulated objects, and long-horizon demonstrations.

With an effective algorithm and evaluation benchmark for functional retargeting, it is now possible to make functional comparisons across different hardware: informed by the policy learning performance on ChiralLabs, we obtain a meaningful measure for both the hands' functionality and readiness to learn from human guidance. This comparison is generalizable and accessible: our algorithm requires no hand-specific adaptations, and our task environments are fast to run and easy to customize. With the recent surge in the development of robotic hand hardware, we hope this functional comparison will be helpful for making informed decisions for both acquiring and designing new hands.

Our contributions are summarized as follows:
• We study **Functional Retargeting**, where we learn feasible dexterous manipulation policies from human hand-object demonstrations. We propose **DexMachina**, a novel algorithm for functional retargeting based on a curriculum over virtual object controllers and motion and contact guidance.
• We introduce **ChiralLabs**, a benchmark with 6 curated dexterous hand assets and 5 articulated objects, for evaluating both different functional retargeting algorithms and robotic hand designs.
• We use ChiralLabs to demonstrate DexMachina achieves state-of-the-art learning performance across a variety of robotic hands and tasks. Our simulation environments and learning algorithms will be open-sourced to facilitate future research.

2 Related Work

Reinforcement Learning for Dexterous Manipulation. Reinforcement Learning (RL) has been used for dexterous manipulation tasks such as in-hand object orientation [1, 11, 12, 13, 14] and single-hand grasping [2, 15, 16, 17, 18, 19], but achieving more complex, longer-horizon manipulation remains challenging due to the burden of designing rewards to guide exploration for such tasks. Model-based methods have been applied to tasks such as ball dribbling [8] and Rubik's cube turning [9], but they require careful engineering for each object and task. In our work, we seek to study bimanual long-horizon tasks where it can be difficult to specify concrete goals or design RL rewards to guide exploration. This motivates our use of human demonstrations, which both act as a goal specification

[1]Deus ex machina ("god from the machine"), is when a seemingly unsolvable problem is conveniently solved by an external force — much like how our algorithm moves an object by itself before the policy gradually learns to take over, hence the name DexMachina.

and provide guidance for how to solve the task. Simulation is a common tool to train dexterous hand policies [20] due to the high exploration cost of running RL on real hardware [21]. Our ChiralLabs simulation benchmark supports evaluation across several dexterous hands and diverse tasks defined by human demonstration data, in contrast to existing RL benchmarks [22, 23] for dexterous manipulation.

Imitation Learning for Dexterous Manipulation. Imitation learning (IL) is a compelling alternative to RL, since the the use of demonstrations can mitigate or eliminate the burden of exploration, but it can require accurate on-robot action data that is challenging to capture for dexterous hands. Most existing approaches [3, 24, 25, 26, 27, 28] require setting up a teleoperation system customized for a particular robot hand embodiment. Human hand data (such as videos) are another source of data. Prior work has used human hand data for learning rough grasp affordances [29], improved retargeting [30], or co-training with human hand data and teleoperation data [31, 32], but these approaches have been limited for short-horizon manipulation (mainly grasping). Instead, our work assumes access to a single tracked hand-object demonstration per task and uses the demonstration to guide RL training. Similar approaches have been used for humanoid locomotion [33], simple hand manipulation [34], and dexterous manipulation on short-horizon tasks [35, 36].

Curriculum Learning. It is common practice in optimization-based motion planning to warmstart an optimization from relaxed physical constraints, resulting in a better solution at convergence [37, 38, 39]. This idea of learning using a curriculum, which moves from easier to more difficult problems, has been adopted by RL methods [40, 41]. Some prior work uses this approach to relax physical constraints, such as allowing force before making contact [42] or relaxing gravity, friction and constraint-solver parameters [36]. Our approach uses a curriculum over object dynamics, allowing the agent to gradually learn how to manipulate the object over time (Fig. 2).

3 Functional Retargeting Formulation

We define the *functional retargeting* problem as follows: given one object η, one human hand-object demonstration sequence $\mathcal{D}^\eta$, and a pair of dexterous robot hands ζ, the goal is to learn a robot policy that can manipulate the object to track the demonstrated object states. More formally, one human demonstration $\mathcal{D}^\eta = \{G, H\}$ contains T timesteps of densely tracked object states G and hand poses H. We focus on articulated objects, hence the object states include both part pose and revolute joint angle values. At any timestep t, given an achieved object state $\hat{g}_t$ (position, rotation, and articulation) and the target object state from the demonstration $g_t = \{g_t^P, g_t^R, g_t^J\}$, we denote the distance function as F (computes both rotation, position, and articulation joint error). The learned policy for (η, ζ) should minimize the accumulated tracking error across all timesteps: $\pi_\theta^{\eta,\zeta} = \text{argmin}_\theta \sum_{t=1}^{T}(F(\hat{g}_t, g_t))$.

4 Method

Overview. We propose DexMachina, a curriculum-based RL algorithm for functional retargeting. In §4.1, we begin by introducing the task reward, which encourages object tracking in but is insufficient for effective policy learning. In §4.2, we extract motion and contact information from demonstrations, which we use to define residual actions and auxiliary rewards. While these components improve learning, they still fall short in complex long-horizon tasks. This motivates our curriculum strategy, presented in §4.3, where we introduce an auto-curriculum based on virtual object controllers to achieve efficient functional retargeting across different dexterous hands.

4.1 RL Environment and Task Reward

We train reinforcement learning (RL) policy to achieve the functional retargeting task. An RL environment is constructed by pairing one demonstration $\mathcal{D}^\eta$ and one set of bimanual dexterous robot hands ζ. At each timestep t, write $G_t = \{g_t^P, g_t^R, g_t^J\}$ for the recorded object position, rotation and joint angles at timestep t, and $\hat{G}_t = \{\hat{g}_t^P, \hat{g}_t^R, \hat{g}_t^J\}$ for the object's achieved states corresponding to each term. The task reward r_{task} is the product of three terms measuring accuracy in each state

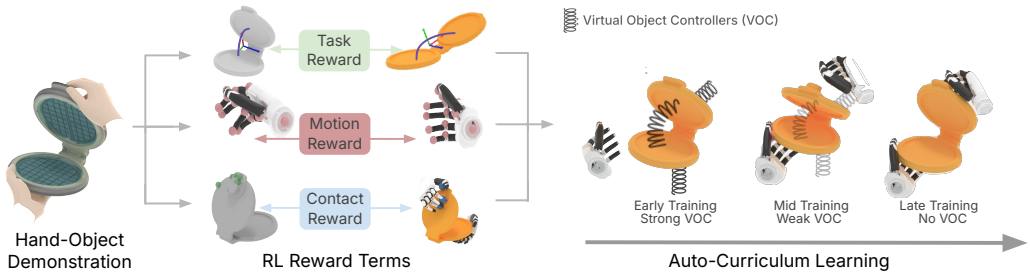

Figure 2: **DexMachina Overview.** DexMachina is a curriculum-based RL algorithm for functional retargeting. We process densely-tracked human hand demonstration to extract reference robot joints and keypoints (pink spheres) and approximated contact positions on object mesh vertices (green spheres), which we use to define auxiliary rewards in addition to the task reward. We then introduce an auto-curriculum using virtual object controllers, which initially moves the object on its own to follow the demonstration, and are then decayed over the course of RL training as the policy learns to take over manipulation.

component, encouraging balanced learning [35]. Formally:

$$d_{\text{pos}} = ||\hat{g}_t^{\,T} - g_t^T||_2;\ d_{\text{rot}} = 2\cos^{-1}(|\langle \hat{g}_t^{\,R}, g_t^R \rangle|);\ d_{\text{ang}} = ||\hat{g}_t^{\,J} - g_t^J||_2$$

$$r_{\text{task}} = r_{\text{pos}} * r_{\text{rot}} * r_{\text{angle}} = \exp(-\beta_{\text{pos}} d_{\text{pos}}) \exp(-\beta_{\text{rot}} d_{\text{rot}}) \exp(-\beta_{\text{ang}} d_{\text{ang}})$$

where β_{pos}, β_{rot}, and β_{ang} are scalar weights that control the desirable error scale for each component.

4.2 Action Formulation and Auxiliary Rewards

Although task reward specifies desired object states, it does not provide useful information for *how* to achieve them. To address this, we (1) propose a hybrid action formulation, which constrains the wrist action space to align more with the human demonstrators; and (2) define auxiliary rewards, which guide the policy to follow the human's hand-object interaction strategy. As a preliminary, we first apply pre-processing on the demonstration data $\mathcal{D}^\eta$ to extract relevant motion and contact information.

Data Pre-Processing. Given $\mathcal{D}^\eta$ with T timesteps, N object parts, and a dexterous hand ζ with J actuated joints and K collision links, we first run a kinematics-only retargeting algorithm [3] that matches dexterous hand poses with human hand motion. Then we obtain:

1. **Collision-aware kinematic retargeted joints** $\mathcal{Q} \in \mathbb{R}^{T\times J}$ and **reference keypoints** $\mathcal{X} \in \mathbb{R}^{T\times K\times 3}$ by replaying the retargeting results in simulation and recording (1) the achieved joint values and (2) 3D keypoint positions of the dexterous hand links. To eliminate object penetrations, we replay the retargeted joint values as soft control targets in simulation while keeping the object fixed — See Appendix A.2 for more details.
2. **Approximated hand-object contact**. Although kinematic retargeting produces human-like dexterous hand poses, the motions often fail to manipulate the object. Hence we extract contact information as additional guidance for object interaction. We use a distance-based approximation to obtain exactly when and where a specific dexterous hand link should be in contact with a specific object part (detailed in Appendix A.4). The results are approximated contact positions $C \in \mathbb{R}^{(T\times N\times K\times 3)}$ and a mask $M \in \mathbb{R}^{(T\times N\times K)}$ that indicates whether a pair of object part and hand link has valid contacts.

Hybrid Action Outputs. Given the retargeted joint results $\mathcal{Q}$, we use the joint values for 6-DoF wrist joints as base actions, which are added to the policy's output residual actions. The remaining finger joints use absolute actions that are normalized by their joint limits (see Appendix A.3 for full details). This formulation effectively constrains the policy's action space, and we empirically find it to significantly improve the learning efficiency.

Motion Imitation Reward. To encourage human-like hand motions, we take the motion reference keypoints $\mathcal{K}$ and retargeted joint values $\mathcal{Q}$, and define (1) motion imitation reward r_{imi} based on

keypoint matching, (2) behavior-cloning reward r_{bc} based on joint angle distances to the reference. Formally:

$$r_{\text{imi}} = \frac{1}{K}\sum_{i=1}^{K}\exp(-\beta_{\text{imi}}||\hat{x}_i - x_i||_2);\ r_{\text{bc}} = \frac{1}{J}\sum_{i=1}^{J}\exp(-\beta_{\text{bc}}||\hat{q}_i - q_i||_2);$$

where each $(\hat{x}_i, x_i)$ denotes the achieved and reference positions for the ith keypoint and $(\hat{q}_i, q_i)$ denotes the achieved and retargeted values for the ith joint.

Contact Reward. We read contact positions between each hand link and each object part, and compute contact reward by matching the policy contacts with the corresponding demonstration contacts. For each side of the hand, we denote the policy's and demonstration's contact positions and validity masks as $C, \hat{C} \in \mathbb{R}^{N\times K\times 3}$, $M, \hat{M} \in \mathbb{R}^{N\times K\times 1}$, respectively. We compute L_2 contact distance masked by validity masks and use it to define contact reward r_{con}:

$$D = \|C - \hat{C}\|_2 \in \mathbb{R}^{N\times K};\ \text{set } D^{(i,j)} = \begin{cases} d_{\max}, & \text{if } M^{(i,j)}_{\text{demo}} \neq M^{(i,j)}_{\text{policy}} \\ 0, & \text{if } M^{(i,j)}_{\text{demo}} = M^{(i,j)}_{\text{policy}} = 0 \end{cases} \tag{1}$$

$$r_{\text{con}} = \frac{1}{2NK}\Big(\sum_{i=1}^{N}\sum_{j=1}^{K}\exp(-\beta_{\text{con}} D^{(i,j)}_{\text{left}}) + \sum_{i=1}^{N}\sum_{j=1}^{K}\exp(-\beta_{\text{con}} D^{(i,j)}_{\text{right}})\Big) \tag{2}$$

The final RL reward is a weighted sum of the above terms: $r_t = \lambda_{\text{task}} r_{\text{task}} + \lambda_{\text{imi}} r_{\text{imi}} + \lambda_{\text{bc}} r_{\text{bc}} + \lambda_{\text{con}} r_{\text{con}}$. See Appendix A.4 for precise weights and additional reward details.

4.3 Auto-Curriculum with Virtual Object Controllers

Motivation. The above reward terms and action constraints are sometimes sufficient short and simple tasks, but struggle on long-horizon clips with complex contacts. The policy often experiences catastrophic early-stage failures: e.g. after lifting a box with both hands, it might fail to anticipate that one hand will need to reposition mid-air to open the lid while the other hand adjusts for single-handed grasping. The policy would attempt different actions, most of which would drop the box and terminate the episode.

This motivates us to propose our curriculum approach, to let the policy explore different strategies in a less fragile setting. Our core idea is using *virtual object controllers*: they drive the object to follow the targets on its own, such that the policy can learn through the entire sequence and be discouraged from myopic strategies.

Algorithm 1 DexMachina Curriculum

Require: Reward thresholds $\sigma_{\text{task}}, \sigma_{\text{imi}}, \sigma_{\text{bc}}, \sigma_{\text{con}}$
Require: Reward deques $D_{\text{task}}, D_{\text{imi}}, D_{\text{bc}}, D_{\text{con}}$
Require: Initial gains k_p, k_v, decay ratios ϕ_p, ϕ_v, max episode length $L_{\max}$

for each PPO iteration **do**
  for each environment where episode is done **do**
    Get achieved episode length L
    Get cumulative rewards $R_{\text{task}}, R_{\text{imi}}, R_{\text{bc}}, R_{\text{con}}$
    for each term $z \in \{\text{task}, \text{imi}, \text{bc}, \text{con}\}$ **do**
      Compute normalized reward: $\bar{r}_z = \frac{R_z}{L_{\max}}$
      Append $\bar{r}_z$ to deque D_z
    end for
  end for
  for each reward type $z \in \{\text{task}, \text{imi}, \text{bc}, \text{con}\}$ **do**
    Compute mean: $\mu_z = \text{mean}(D_z)$
  end for
  if $k_p = 0$ **then**
    continue // no need to decay
  end if
  if $\mu_z > \sigma_z\ \forall z \in \{\text{task}, \text{imi}, \text{bc}, \text{con}\}$ **then**
    // Learning is stable, applying gain decay
    $k_p \leftarrow k_p \cdot \phi_p$
    if $k_p \leq 0.01$ **then**
      $k_p \leftarrow 0; \quad k_v \leftarrow 0$
    end if
    $k_v \leftarrow k_v \cdot \phi_v$
  end if
end for

Virtual Object Controllers. We treat the demonstration states G as control goals and apply virtual spring-damper constraints that move the object along its target trajectory. Initially, the virtual controllers handle most of the object movement; over time, the controller's influence is gradually reduced, requiring the policy to assume greater control to complete the task. They controllers are implemented using privileged information in simulation. Each object is equipped with six virtual 1-DoF joints for its base pose and a 1-DoF joint for articulation, and all joints are actuated

by PD controllers [43]. At every timestep, these controllers apply virtual forces based on the error between the current object state and control targets from the demonstration. The control strength is parametrized by gain parameters (k_p, k_v), which are decayed over time to enable a structured hand-off to the learned policy.

Curriculum scheduling. Algorithm 1 describes our proposed curriculum. At the beginning of curriculum training, we set high virtual controller gains with critical damping; then we exponentially decay the gains based on the policy's learning progress, which is tracked with a history of past rewards. As a result, the policy initially will consistently achieve high task reward; because it receives a weighted sum of task and auxiliary rewards, the policy learns actions that improve motion and contact rewards while avoiding disrupting the object trajectory. Later, as the object controllers weaken, the policy gradually learns to adjust its motions to maintain high task reward. Because the auxiliary rewards use a much smaller weight, the policy can deviate from the reference hand motions learned at the earlier stages in order to prioritize optimizing for high task rewards.

5 Experiments

Experiment Setup. We use hand-object data from ARCTIC [10] (see §A), which includes 5 articulated objects [44] and 7 demonstrations consisting of diverse motion sequences (picking up and reorienting objects, opening/closing lids, etc.) We evaluate our algorithm on both short- (used in prior work [35]) and long-horizon demonstrations. We curate assets for 6 open-source dexterous robot hand models, with varying sizes and kinematic designs. We use Genesis [45] for physics simulation, and PPO [46, 47] as the base RL algorithm. The policies share the same structure for state-based input observation spaces for all hands and tasks, and control both hands at once. See Appendix for details on RL training (§B.1) and evaluation setup §B.4.

Baseline Methods. Due to various differences in physics simulation and training configurations, we re-implement baseline methods in our training framework and make several adaptations to ensure a fair comparison — see § B.3 for implementation details. We compare against the following methods:

1. **Kinematics Only.** Directly use kinematic retargeting [3] results as policy controller targets.
2. **ObjDex [35].** learns a high-level wrist planner for wrist base actions, and a low-level policy with task reward and the same hybrid actions as ours. We validate our re-implementation by showing improved performance on the same demonstrations used in the original results.
3. **Task + Auxiliary Rewards without curriculum.** To evaluate the effect of our proposed curriculum, we run RL training with only our proposed motion imitation and contact rewards. For a fair comparison, all training hyper-parameters are identical with our curriculum setting.
4. **ManipTrans [36].** A concurrent work that fine-tunes a motion imitation model with RL using contact force rewards and a curriculum on error thresholds and physics parameters. Since the original method is evaluated on rigid objects in a different physics simulator, we re-implement their proposed curriculum while using our hybrid actions and auxiliary reward terms.

Overview of Experiments. We present empirical results that (1) evaluate the effectiveness of DexMachina against baselines and no-curriculum settings (§ 5.1); (2) ablate key components of our method (§ 5.2); (3) demonstrate DexMachina's applicability across various dexterous hand embodiments and utility as an evaluation framework for comparing different hand designs (§5.3).

5.1 DexMachina Main Results

We evaluate DexMachina and baseline methods on four representative dexterous hands (Inspire [48], Allegro [49], Xhand [50], and Schunk [51]) and seven demonstration clips (see C for visualization). Averaged success rates for each task are reported in Figure 3. The key takeaways are the following:

DexMachina consistently improves performance on all hands and tasks. We highlight the rightmost four columns in Fig. 3, which correspond to long-horizon demonstrations with complex

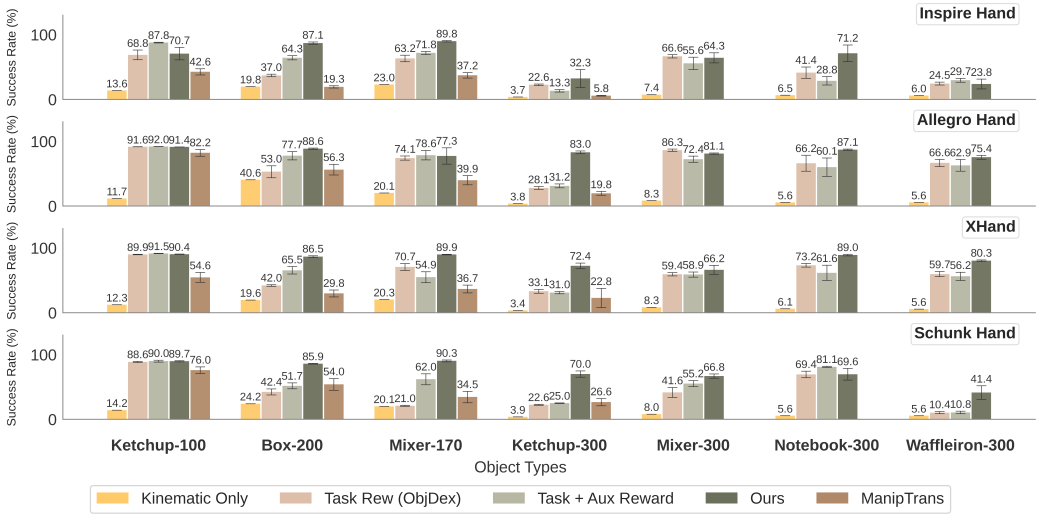

Figure 3: **DexMachina Core Results.** We evaluate DexMachina on four representative dexterous hands paired with seven demonstrations with diverse objects and motion sequences. We compare between direct replay of kinematic retargeting results ("Kinematic Only"), training with only a task reward ("Task Rew (ObjDex)", i.e., our re-implementation of ObjDex [35]), training with both task and auxiliary rewards ("Task + Aux Reward"), and with our proposed auxiliary rewards and curriculum ("Ours"). With rare exceptions, DexMachina demonstrates clear improvements over baseline methods, especially on long-horizon tasks with more complex motions.

motion sequences[2]. Task reward alone falls short on these clips; incorporating auxiliary rewards ('Task + Aux Rewards') improves performance on some tasks, but the gains are inconsistent. In contrast, DexMachina significantly outperforms no-curriculum setting despite using the same rewards.

Task reward and hybrid actions can achieve reasonable performance on short-horizon tasks: our re-implementation of ObjDex [35] ('Task Rew (ObjDex)' in Fig. 3) performs better than their original reporting on the same demonstrations (left three columns in Fig. 3, detailed in §B.3). The kinematic retargeting results alone cannot complete the task ('Kinematics Only') — our videos qualitatively show that they visually align well with human hands, but the actions cannot achieve more than slightly lifting up each object.

DexMachina lets the policy learn task strategies that adapt to hardware constraints. The auxiliary rewards do not always align with the best task strategy, but instead act as soft guidance to serve the curriculum, giving the policy flexibility to explore. Qualitatively, we observe that the policies may deviate from the motion and contact guidance and learn different strategies: as shown in Fig. 4: on Notebook-300, the XHand policy follows the human demonstrator to use the left hand to hold up the object and the right hand to close the cover; however, for the smaller, less-actuated Inspire Hand, the policy learns to use both hands to stabilize the object and close the cover. On Mixer-300, the Allegro Hand fingers are long enough to close the lid easily, but the Schunk Hand policy shows more wrist movements to achieve the same effect.

5.2 DexMachina Ablations

Action Ablations. We compare our hybrid action formulation with: (1) absolute actions on all joints and (2) less-constrained residual actions on wrist joints, in which the wrist joint limits are set to cover the maximum motion range in the entire demonstration clip. We train in the no-curriculum setting, use a subset of tasks and hands and average over three seeds for each method. Results are shown in Fig. 8. While all methods benefit from using auxiliary rewards, using more restrictive bounds on wrist motion results in the best overall performance.

[2]For instance, 'Waffleiron-300' requires the policy to pick up the object, open and close the lid, flip it back and forth, then open and close the lid again, all mid-air (see Appendix §C for a visualization)

Figure 4: **DexMachina Hand-Specific Strategies.** DexMachina enables the policy to learn task strategies that adapt to their hardware constraints. We show snapshots of trained policy rollouts for different hands on the same task: left side shows XHand and Inspire Hand for Notebook-300 task; right side shows Schunk Hand and Allegro Hand for Mixer-300 task.

Curriculum Ablations. In Fig. 3, we compare DexMachina with ManipTrans [36], which uses a curriculum over error thresholds for motion and object poses plus gravity and friction parameters. We observe that it achieves no clear improvements over the no-curriculum setting, and training is less stable: given the same budget of RL iterations, the ManipTrans policy initially achieves high task reward, but performance drops as the curriculum progresses and cannot recover. This indicates that merely decaying physics parameters is not sufficient for long-horizon tasks with articulated objects, which needs a stronger guidance to completely solve the task until the policy gradually takes over.

5.3 Hand Embodiment Analysis

After validating that DexMachina achieves functional retargeting across various tasks and hands, we now use our algorithm and benchmark for a functional comparison between different dexterous hands. We focus on the four long-horizon tasks from §5.1, and evaluate DexMachina on two additional hands, Ability [52] and DexRobot Hand (see Fig. 5). We discuss the following key findings:

Larger, fully-actuated hands achieve both higher final performance and better learning efficiency. The Allegro Hand, despite being less anthropomorphic in appearance, is surprisingly capable due to its long finger length providing stability for in-hand / in-air manipulations. **Similarity in size is less important than degrees of freedom.** For instance, the Inspire, Ability, and Schunk Hand all have similar sizes, but Schunk has actuated fingertips and a foldable palm, and achieves on average better performance than Inspire and Ability. **Although less-actuated hands are more similar to human hands in appearance, learned strategies are less human-like than the bigger but more capable hands.** Because all hands use the same set of human hand motion references (both as base wrist actions and motion rewards), the extent to which a policy deviates from human guidance is determined by their size and kinematic constraints. As a result, hands like Inspire and Ability often need different strategies to complete the task.

Naturally, our conclusions are limited by the objects and tasks that we test on: for instance, the larger hands will not perform well for smaller objects (e.g., tweezers). However, our evaluation framework can be easily extended to add new dexterous hands and test tasks or objects.

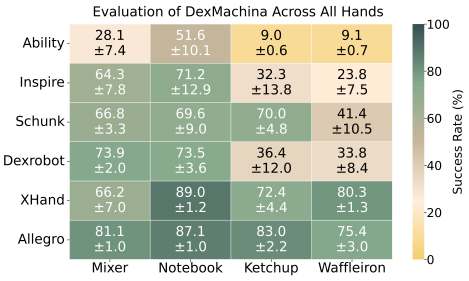

Figure 5: Full evaluation of all six hands using DexMachina on long-horizon tasks.

6 Conclusion

We present DexMachina, a curriculum-based RL algorithm for functional retargeting, where the key idea is to use virtual object controllers that let the policy easily explore task strategies under motion and contact guidance. In our simulation benchmark with a diverse set of tasks and dexterous hands, we show DexMachina significantly outperforms baseline methods and enables functional comparison across different dexterous hand designs. We hope our algorithm and benchmark environments will provide useful platform for identifying desirable dexterous hand capabilities and lower the barrier for contributing to future research.

7 Limitations

DexMachina has a few key limitations. First, our policy uses state-based input that relies on privileged information from the physics simulator; this information can be challenging to acquire in the real world. This limitation can be addressed with either vision-based RL policy training [2], or more practically, a distillation setup that trains visuomotor policies using demonstration data generated from state-based policies, as seen in prior work [35]. Second, our problem formulation assumes access to high-quality human hand-object demonstration data, which requires object reconstruction and accurate pose tracking for both human hands and articulated objects. Such data can be expensive to collect and requires careful curation (e.g. ARCTIC [10] uses a motion-capture system with dense manual annotations and post-processing). Future work could investigate alternative methods to scale up data collection: one direction is leveraging recent progress in 3D generative models and reconstruction methods. Third, because we use open-source assets for the dexterous hands and estimate physical properties (such as mass, inertia, and collision shapes), the simulated hands might fail to capture some of the dynamics and capabilities of the real hardware. To address this, more careful tuning with real reference hardware will be needed; ideally, accurate simulation models would be provided directly by manufacturers. Lastly, our learned RL policies have not yet been evaluated in real-world settings on the examined range of dexterous hands due to lack of hardware access. With inputs from the community, we aim for our simulation benchmark to enable accessible research and evaluation of hand designs without requiring physical hardware; furthermore, our learned policy can be used as teacher policies to be distilled for sim-to-real transfer, which prior work in similar settings has demonstrated [35, 36].

Acknowledgments

This work was supported in part by NVIDIA, and by NSF Awards #2143601, #2037101, and #2132519. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors. The authors would like to thank current and former colleagues at NVIDIA: Kelly Guo, Milad Raksha, David Hoeller, Bingjie Tang for their help with physics simulation environments and insightful discussions during algorithm development; and all the members of REALab at Stanford University for providing useful feedback on initial drafts of the paper manuscript.

References

[1] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020. 1, 2

[2] T. G. W. Lum, M. Matak, V. Makoviychuk, A. Handa, A. Allshire, T. Hermans, N. D. Ratliff, and K. V. Wyk. Dextrah-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics, 2024. URL https://arxiv.org/abs/2407.02274. 1, 2, 9

[3] Y. Qin, W. Yang, B. Huang, K. Van Wyk, H. Su, X. Wang, Y.-W. Chao, and D. Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. In *Robotics: Science and Systems*, 2023. 1, 2, 3, 4, 6

[4] M. Rakić. Paper 11: The 'belgrade hand prosthesis'. *Proceedings of the Institution of Mechanical Engineers, Conference Proceedings*, 183(10):60–67, 1968. doi:10.1243/PIME_CONF_1968_183_179_02. URL https://doi.org/10.1243/PIME_CONF_1968_183_179_02. 2

[5] C. Loucks, V. Johnson, P. Boissiere, G. Starr, and J. Steele. Modeling and control of the stanford/jpl hand. In *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, volume 4, pages 573–578, 1987. doi:10.1109/ROBOT.1987.1088031. 2

[6] S. Jacobsen, E. Iversen, D. Knutti, R. Johnson, and K. Biggers. Design of the utah/m.i.t. dextrous hand. In *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, volume 3, pages 1520–1532, 1986. doi:10.1109/ROBOT.1986.1087395. 2

[7] J. Butterfass, G. Hirzinger, S. Knoch, and H. Liu. Dlr's multisensory articulated hand. i. hard- and software architecture. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, volume 3, pages 2081–2086 vol.3, 1998. doi:10.1109/ROBOT.1998.680625. 2

[8] D. Shiokata, A. Namiki, and M. Ishikawa. Robot dribbling using a high-speed multifingered hand and a high-speed vision system. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2097–2102. IEEE, 2005. 2

[9] R. Higo, Y. Yamakawa, T. Senoo, and M. Ishikawa. Rubik's cube handling using a high-speed multi-fingered hand and a high-speed vision system. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6609–6614. IEEE, 2018. 2

[10] Z. Fan, O. Taheri, D. Tzionas, M. Kocabas, M. Kaufmann, M. J. Black, and O. Hilliges. ARCTIC: A dataset for dexterous bimanual hand-object manipulation. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 6, 9, 14, 15

[11] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5977–5984. IEEE, 2023. 2

[12] H. Qi, A. Kumar, R. Calandra, Y. Ma, and J. Malik. In-hand object rotation via rapid motor adaptation. In *Conference on Robot Learning*, pages 1722–1732. PMLR, 2023. 2

[13] Z.-H. Yin, B. Huang, Y. Qin, Q. Chen, and X. Wang. Rotating without seeing: Towards in-hand dexterity through touch. *arXiv preprint arXiv:2303.10880*, 2023. 2

[14] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84):eadc9244, 2023. doi:10.1126/scirobotics.adc9244. URL https://www.science.org/doi/abs/10.1126/scirobotics.adc9244. 2

[15] V. Caggiano, S. Dasari, and V. Kumar. Myodex: A generalizable prior for dexterous manipulation. *ArXiv*, abs/2309.03130, 2023. URL https://api.semanticscholar.org/CorpusID:260927595. 2

[16] Z. Luo, J. Cao, S. J. Christen, A. Winkler, K. Kitani, and W. Xu. Grasping diverse objects with simulated humanoids. *ArXiv*, abs/2407.11385, 2024. URL https://api.semanticscholar.org/CorpusID:271217823. 2

[17] P. Mandikal and K. Grauman. Dexterous robotic grasping with object-centric visual affordances. In *International Conference on Robotics and Automation (ICRA)*, 2021. 2

[18] T. Zhu, R. Wu, J. Hang, X. Lin, and Y. Sun. Toward human-like grasp: Functional grasp by dexterous robotic hand via object-hand semantic representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45:12521–12534, 2023. URL https://api.semanticscholar.org/CorpusID:258462924. 2

[19] H. Yuan, B. Zhou, Y. Fu, and Z. Lu. Cross-embodiment dexterous grasping with reinforcement learning. *ArXiv*, abs/2410.02479, 2024. URL https://api.semanticscholar.org/CorpusID:273098035. 2

[20] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations, 2018. URL https://arxiv.org/abs/1709.10087. 3

[21] K. Xu, Z. Hu, R. Doshi, A. Rovinsky, V. Kumar, A. Gupta, and S. Levine. Dexterous manipulation from images: Autonomous real-world rl via substep guidance. *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5938–5945, 2022. URL https://api.semanticscholar.org/CorpusID:254877506. 3

[22] C. Bao, H. Xu, Y. Qin, and X. Wang. Dexart: Benchmarking generalizable dexterous manipulation with articulated objects. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21190–21200, 2023. doi:10.1109/CVPR52729.2023.02030. 3

[23] S. R. Company. Shadow hand official website. https://www.shadowrobot.com/dexterous-hand-series/, 2025. Accessed: 2025-03-17. 3

[24] C. Wang, H. Shi, W. Wang, R. Zhang, L. Fei-Fei, and C. K. Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024. 3

[25] S. Yang, M. Liu, Y. Qin, D. Runyu, L. Jialong, X. Cheng, R. Yang, S. Yi, and X. Wang. Ace: A cross-platfrom visual-exoskeletons for low-cost dexterous teleoperation. *arXiv preprint arXiv:240*, 2024. 3

[26] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang. Open-television: Teleoperation with immersive active visual feedback. *arXiv preprint arXiv:2407.01512*, 2024. 3, 14

[27] K. Shaw, Y. Li, J. Yang, M. K. Srirama, R. Liu, H. Xiong, R. Mendonca, and D. Pathak. Bimanual dexterity for complex tasks. In *8th Annual Conference on Robot Learning*, 2024. 3

[28] H. Zhang, S. Hu, Z. Yuan, and H. Xu. Doglove: Dexterous manipulation with a low-cost open-source haptic force feedback glove. *arXiv preprint arXiv:2502.07730*, 2025. 3

[29] P. Mandikal and K. Grauman. Learning dexterous grasping with object-centric visual affordances. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6169–6176, 2020. URL https://api.semanticscholar.org/CorpusID:233439776. 3

[30] S. Park, S. Lee, M. Choi, J. Lee, J. Kim, J. Kim, and H. Joo. Learning to transfer human hand skills for robot manipulations, 2025. URL https://arxiv.org/abs/2501.04169. 3

[31] K. Shaw, S. Bahl, and D. Pathak. Videodex: Learning dexterity from internet videos, 2022. URL https://arxiv.org/abs/2212.04498. 3

[32] M. Xu, Z. Xu, C. Chi, M. Veloso, and S. Song. Xskill: Cross embodiment skill discovery. In *Conference on robot learning*, pages 3536–3555. PMLR, 2023. 3

[33] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graph.*, 37(4):143:1–143:14, July 2018. ISSN 0730-0301. doi:10.1145/3197517.3201311. URL http://doi.acm.org/10.1145/3197517.3201311. 3

[34] Y. Wang, J. Lin, A. Zeng, Z. Luo, J. Zhang, and L. Zhang. Physhoi: Physics-based imitation of dynamic human-object interaction, 2023. URL https://arxiv.org/abs/2312.04393. 3

[35] Y. Chen, C. Wang, Y. Yang, and C. K. Liu. Object-centric dexterous manipulation from human motion data. *ArXiv*, abs/2411.04005, 2024. URL https://api.semanticscholar.org/CorpusID:273850263. 3, 4, 6, 7, 9, 16, 17

[36] K. Li, P. Li, T. Liu, Y. Li, and S. Huang. Maniptrans: Efficient dexterous bimanual manipulation transfer via residual learning. *arXiv preprint arXiv:2503.21860*, 2025. 3, 6, 8, 9, 16, 17

[37] I. Mordatch, E. Todorov, and Z. Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (ToG)*, 31(4):1–8, 2012. 3

[38] T. Pang and R. Tedrake. A convex quasistatic time-stepping scheme for rigid multibody systems with contact and friction. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6614–6620. IEEE, 2021. 3

[39] T. Pang, H. T. Suh, L. Yang, and R. Tedrake. Global planning for contact-rich manipulation via local smoothing of quasi-dynamic contact models. *IEEE Transactions on robotics*, 39(6): 4691–4711, 2023. 3

[40] A. S. Chiappa, P. Tano, N. Patel, A. Ingster, A. Pouget, and A. Mathis. Acquiring musculoskeletal skills with curriculum-based reinforcement learning. *bioRxiv*, 2024. doi: 10.1101/2024.01.24.577123. URL https://www.biorxiv.org/content/early/2024/01/25/2024.01.24.577123. 3

[41] H. Zhang, S. Christen, Z. Fan, L. Zheng, J. Hwangbo, J. Song, and O. Hilliges. ArtiGrasp: Physically plausible synthesis of bi-manual dexterous grasping and articulation. In *International Conference on 3D Vision (3DV)*, 2024. 3

[42] X. Mao, Y. Xu, Z. Sun, E. Miller, D. Layeghi, and M. Mistry. Learning long-horizon robot manipulation skills via privileged action. *arXiv preprint arXiv:2502.15442*, 2025. 3

[43] G. F. Franklin, J. D. Powell, A. Emami-Naeini, and J. D. Powell. *Feedback control of dynamic systems*, volume 4. Prentice hall Upper Saddle River, 2002. 6

[44] X. Xu, D. Bauer, and S. Song. Robopanoptes: The all-seeing robot with whole-body dexterity. *arXiv preprint arXiv:2501.05420*, 2025. 6

[45] G. Authors. Genesis: A universal and generative physics engine for robotics and beyond, December 2024. URL https://github.com/Genesis-Embodied-AI/Genesis. 6, 16

[46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347. 6

[47] D. Makoviichuk and V. Makoviychuk. rl-games: A high-performance framework for reinforcement learning. https://github.com/Denys88/rl_games, May 2021. 6, 16

[48] L. Beijing Inspire-Robots Technology Co. Inspire hand official website. https://inspire-robots.store/collections/the-dexterous-hands, 2025. Accessed: 2025-03-17. 6

[49] W. Robotics. Allegro hand official website. https://www.allegrohand.com/, 2025. Accessed: 2025-03-17. 6

[50] ROBOTERA. Xhand1 official website. https://www.robotera.com/en/goods1/4.html, 2025. Accessed: 2025-03-17. 6

[51] S. S. . C. KG. Schunk 5-finger hand official website. https://schunk.com/us/en/gripping-systems/special-gripper/svh/c/PGR_3161, 2025. Accessed: 2025-03-17. 6

[52] PSYONIC. Ability hand official website. https://www.psyonic.io/ability-hand, 2025. Accessed: 2025-04-30. 8

[53] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), Nov. 2017. 14

[54] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021. 16

[55] B. Wen, W. Yang, J. Kautz, and S. Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *CVPR*, 2024. 17

[56] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes, 2018. URL https://arxiv.org/abs/1711.00199. 17

[57] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Computer Vision, ACCV 2012 - 11th Asian Conference on Computer Vision, Revised Selected Papers*, number PART 1 in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 548–562, 2013. ISBN 9783642373305. doi:10.1007/978-3-642-37331-2_42. 11th Asian Conference on Computer Vision, ACCV 2012 ; Conference date: 05-11-2012 Through 09-11-2012. 17

Please see our submission website: dexmachina-submission.github.io

A Demonstration Data Processing Details

A.1 ARCTIC Demonstration Selection and Curation

We use a subset of hand-object interaction clips from the ARCTIC dataset [10], which contains articulated object scans and interaction sequences with tracked MANO [53] hand poses and object states. Each selected clip is defined by an object (e.g. 'box'), a subject tag (e.g. 's01-u01') identifying the human demonstrator, and a (start, end) tuple to trim the sequence to a fixed length, hence the number of used frames T is defined as $T = (\text{end} - \text{start})$

Dexterous Hand Asset Processing. All of the dexterous hands in our experiments are curated from open-source URDF models and manually edited to add 6-DoF wrist joints that achieve a 'floating hand' style wrist actuation. Some of the dexterous hand models require additional processing for stable simulation, such as manually changing mass or inertia values, running convex-decomposition to improve collision mesh quality, and adding dummy links to fingertips to record and track keypoint positions. For each dexterous hand, we manually specify which finger links should match with which MANO [53] hand joints(e.g. thumb to human thumb), which is required by the kinematic retargeting [26] algorithm. The kinematic retargeting results are also used for controller gain tuning, which ensures the dexterous hand controllers are stable and fast enough to match the desired human hand movement and speed within reasonable error.

A.2 Object-aware Retargeting Post-processing

Because we use densely-tracked human hand and object interaction as demonstration, a purely kinematic retargeting algorithm [26] on fingertip positions results in frequent penetration with the object, which leads to damaging base-actions during policy learning, and infeasible keypoint positions which we use for imitation reward computation. To address this, we run the simulation for each pair of dexterous hands, and for each demonstrated timestep, we fixate the object to its target state (both root pose and object joint angle), and set retargeted joint values as control targets. This process lets the simulation to resolve collision and

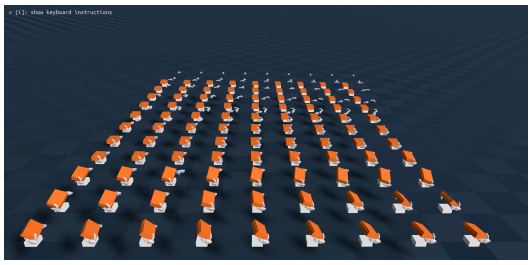

Figure 6: We perform an improved retargeting scheme over pure kinematic retargeting

Then we record the achieved joint values and keypoints to use for policy learning. In implementation, this process can be easily parallelized in simulation, which we illustrate in Figure 6.

A.3 Hybrid Action Outputs.

Formally, we use the following notations:

- $\text{clip}(x, a, b)$: elementwise clamp of input value x between a and b
- $a_t \in \mathbb{R}^J$: the policy's joint action output at time t clipped to $[-1, 1]$, i.e. $a_t = \text{clip}(\pi_\theta(o_t), -1, 1)$
- $q_t^{(i)}$: the target position for the i-th joint at time t
- $\mathcal{I}_f \subset \{1, \dots, J\}$: indices corresponding to the finger DOFs
- $\mathcal{I}_w^{\text{T}} \subset \{1, \dots, J\}$: indices of the three wrist translation DOFs, $|\mathcal{I}_w^{\text{T}}| = 3$
- $\mathcal{I}_w^{\text{R}} \subset \{1, \dots, J\}$: indices of the three wrist rotation DOFs, $|\mathcal{I}_w^{\text{R}}| = 3$

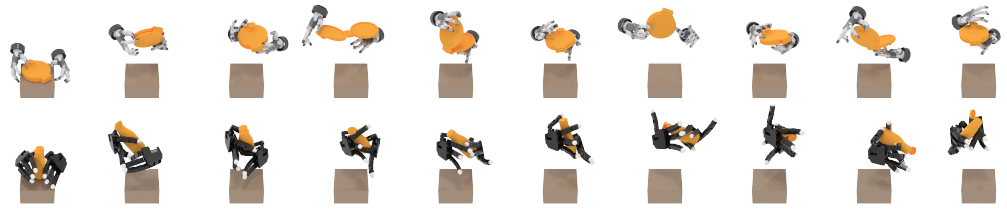

Figure 7: Visualization of the long-horizon tasks achieved by our trained RL policy. The brown box is used as platform surface, which follows the original ARCTIC data collection setup where objects are placed on a square cardboard box on a table surface [10].

- $\mathbf{q}_t \in \mathbb{R}^J$: the **retargeted** joint values at time t
- $s_{\text{T}}, s_{\text{R}}$: scaling factors for translation and rotation actions respectively
- $\ell, u \in \mathbb{R}^J$: vectors of lower and upper joint limits
- $\hat{q}_t \in \mathbb{R}^J$: the joint target values sent to the policy's controller

Then, the joint target computation is defined as:

$$
\begin{aligned}
a_t^{\text{wrist-T}} &= a_t[\mathcal{I}_w^{\text{T}}] \in \mathbb{R}^3, \quad q_t^{\text{wrist-T}} = \mathbf{q}_t[\mathcal{I}_w^{\text{T}}] + s_{\text{T}} \cdot a_t^{\text{wrist-T}} \\
a_t^{\text{wrist-R}} &= a_t[\mathcal{I}_w^{\text{R}}] \in \mathbb{R}^3, \quad q_t^{\text{wrist-R}} = \mathbf{q}_t[\mathcal{I}_w^{\text{R}}] + s_{\text{R}} \cdot a_t^{\text{wrist-R}} \\
a_t^{\text{fingers}} &= a_t[\mathcal{I}_f], \quad q_t^{\text{fingers}} = \ell_{\mathcal{I}_f} + \frac{u[\mathcal{I}_f] - \ell[\mathcal{I}_f]}{2} \cdot (a_t^{\text{fingers}} + 1) \\
\hat{q}_t &= \text{concat}(q_t^{\text{wrist-T}},\ q_t^{\text{wrist-R}},\ q_t^{\text{fingers}})
\end{aligned}
$$

A.4 Contact Approximation

Let: $V_o = \{v_i^o\}_{i=1}^{N_o}$ be the vertices of one object part mesh, $V_h = \{v_j^h\}_{j=1}^{N_h}$ be the vertices of one MANO hand mesh, γ be the contact distance threshold, N_c be the maximum number of raw contact approximations (we use $\gamma = 0.01, N_c = 50$), and K be the number of collision links on a dexterous robot hand.

First, we do contact approximation by finding object mesh vertices that, their L_2 distance to the nearest neighbor on the MANO mesh is within γ: for each v_i^o, we get $v_j^* = \arg\min_j \|v_i^o - v_j^h\|_2$, and mark v_i^o as an approximate contact point if $\|v_i^o - v_{j^*}^h\|_2 < \gamma$. When there's more than N_c vertices within this threshold, we use farthest sub-sampling to get the final set of N_c contacts, denoted as $C = \{v_k^c\}_{k=1}^{N_c} \subset V_o$.

Next, we "retarget" the raw approximate contacts to the dexterous robot hand: let $L = \{\ell_m\}_{m=1}^K$ be the center positions of the dexterous hand links, for each contact point $v_k^c \in C$, assign it to the nearest link: $m^* = \arg\min_m \|v_k^c - \ell_m\|_2$ For each link ℓ_m, compute the average position of the assigned contacts: $\bar{v}_m = \frac{1}{|C_m|} \sum_{v_k^c \in C_m} v_k^c$ where $C_m \subset C$ is the subset of contacts assigned to link ℓ_m. If $|C_m| = 0$, then $\bar{v}_m$ is marked as invalid.

The final outputs are:

- A contact tensor $\mathcal{C} \in \mathbb{R}^{T \times N \times K \times 3}$
- A validity mask $\mathcal{M} \in \{0, 1\}^{T \times N \times K}$

where T is the number of time-steps in the demonstration clip, N is the number of object parts ($N = 2$ for all our articulated object assets). The exact same procedure is repeated for each dexterous hand, hence each bi-manual RL task environment has two copies of contact information with the same shapes.

B Experiment Details

B.1 RL Training and Evaluation Details

We use Genesis for physics simulation [45] and PPO as the base RL algorithm implemented by the rl-games [47] package. In the reported results for both ours and baseline methods, we use $12,000$ parallel environments for RL training on all the dexterous hands, except for Dex Hand which uses $10,000$ environments due to memory constraints. Each training run occupies either one single NVIDIA L40s or H100 GPU, and we run 5 random seeds for each demonstration and each pair of dexterous hands for all compared methods, except for action ablation experiments in §5.2 which use 3 random seeds.

B.2 RL Policy Observation and Action Space

We use state-based input for policy observation space: this include object states, joint targets, finger-to-object distances, and normalized hand-object contact forces.

B.3 Details on Baseline Reimplementation

Our most relevant baseline methods [35, 36] are built with Isaac-Gym [54] with various simulation-specific implementation details. To ensure a fair comparison, we have dedicated effort to ensure a faithful reimplementation using our training framework and RL environments. Some of our modifications can result in better performance for the baselines than their original reports: for example, Genesis [45] uses a more stable simulation contact modeling and is more memory efficient, which enables training up to $12,000$ parallel environments with much higher learning efficiency than the Isaac Gym environments used by baselines (i.e. 2048 for ObjDex [35] and 4096 environments for ManipTrans [36]). We describe further details our reimplementation for each baseline below:

ObjDex [35] Reimplementation Details. To ensure a fair comparison, we have contacted the original authors to obtain their setup details that were not available publicly, including: 1. A good estimate for the exact frame start- and end- parameters for the ARCTIC clips used for training; 2. A frame interpolation multiplier that effectively extends the episode length for RL training to be longer than the original demonstration (e.g. an ARCTIC clip with T timesteps requires training RL on $4T$ or $7T$ episode steps). We reuse their clip range but choose not to use the interpolation after empirically finding it to increase training time without improving task performance.

Moreover, the original ObjDex [35] method uses a two-level framework, where a high-level wrist planner is first learned across all ARCTIC demonstration clips, and a low-level RL policy outputs wrist residual actions. We instead directly use the kinematic retargeting results for the wrist base actions. The high-level wrist planner design assumes access to a bigger dataset and makes the low-level RL policy sensitive to the learned planner outputs — **we hypothesize this is the main reason for why our reimplementation can achieve better performance than the original results** (e.g. On Ketchup-100, our re-implementation achieves $> 90\%$ success rate for all hands, whereas the original paper reports 41.2%; on Mixer-170, ours achieves $> 70\%$ success rate for three out of four hands, whereas the paper reports 57.6%).

ManipTrans [36] Reimplementation Details. Because the original method did not directly evaluate on ARCTIC demonstrations (albeit the paper's appendix Section A.1 [36] reported *qualitative* results for some ARCTIC objects), we reimplement their proposed curriculum while keeping everything else aligned with our best-performing setup (this includes hybrid action formulation, training with both task and auxiliary rewards and the same RL hyper-parameters, etc.). We follow the original method [36] to decay four parameters during training, namely the thresholds for object pose errors and hand keypoint error, the z-axis gravity value, and the friction parameter, which we write as $\epsilon^P_{\text{object}}, \epsilon^R_{\text{object}}, \epsilon_{\text{finger}}, g_{\text{gravity}}, \mu$, respectively. We modified Genesis [45]'s rigid solver to support modifying the gravity vector during RL training. ManipTrans [36] does not disclose the exact decaying

schedule for these parameters or the range for gravity and friction parameters, hence we choose the same exponential scheduler to stay consistent with our virtual object controller curriculum, and choose a range of $g_{\text{gravity}} \in [0, -9.81], \mu \in [4.0, 1.0]$. More specifically, given a max iteration I, desired range of parameters, and a decay interval v, the parameter is decayed every v iterations; after the parameters reach their final values, training proceeds for another fixed number of iterations (this is also aligned with our method). The exponential schedule depends on the given max iterations $\mathcal{I}$, for each parameter $\omega \in \{\epsilon^R_{\text{object}}, \epsilon_{\text{finger}}, g_{\text{gravity}}, \mu\}$: its value at a given training iteration can be written as $\omega_{\text{current}} = \omega_{\text{init}} \cdot \left(\frac{\omega_{\text{final}}}{\omega_{\text{init}}}\right)^{t/I}$. Note that we use a pseudo value $\bar{g}_{\text{gravity}} \in [9.81, 0]$ because the decay computation assumes positive bounds, and the actual applied gravity is $g_{\text{gravity}} = 9.81 - \bar{g}_{\text{gravity}}$.

B.4 Policy Evaluation Setup

Evaluation Across Random Seeds. For each method and task, we run 5 random seeds; each seed run saves a best policy checkpoint based on cumulative task reward, and each checkpoint is evaluated for 20 episodes. For each evaluation episode, we record the achieved object states (both pose and revolute joint angle) and compare against the demonstration trajectory.

Performance Metrics. Our functional retargeting task requires a manipulation policy to achieve articulated object tracking, which involves balancing both pose and joint angle errors over long time sequences. For performance reporting, prior work has explored per-step success rate [35] or tracking error [36], but both have clear shortcomings: success rate reporting is based on how many timesteps out of the entire demonstration a policy can move the object to track within given error thresholds, hence the results are highly sensitive to the threshold values (this case requires 3 different thresholds for position, rotation, and joint angle errors), which also depend on the object size and geometries. Reporting tracking errors can be accurate, but it shows three different errors for every task, hence making it difficult to derive high-level comparisons and takeaways from experiment results. To address these limitations, we propose to follow prior work in object pose tracking [55, 56, 57] and use a similar ADD-AUC[3] metric with the key difference that we compute ADD for each object part separately (to accommodate articulated objects) and average ADD results before computing AUC. We found this to be a less-sensitive metric that still reports one success rate value for each method while reflective of the qualitative results from policy roll-outs.

C Additional Experiment Results

We visualize keyframes of our long-horizon manipulation tasks in Fig. . Please see supplementary videos for additional qualitative results for policy roll-outs. The figure in the next page shows results for our action ablation experiments described in §5.2.

[3] ADD stands for Average Distance, AUC stands for Area Under Curve, we don't use ADD-S because we have the exact matching targets from the demonstration

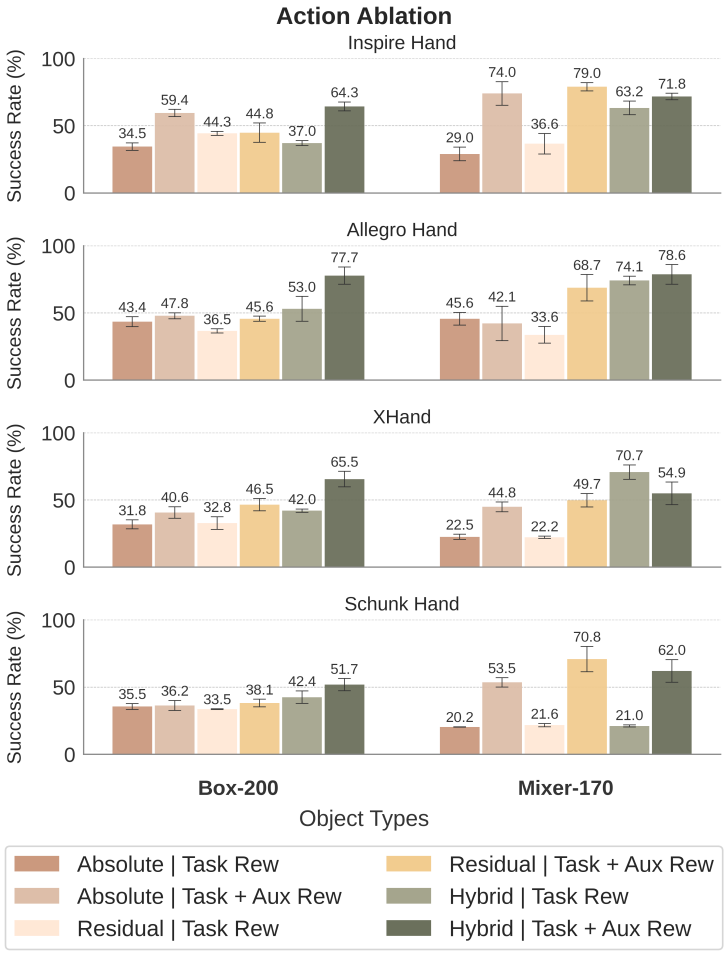

Figure 8: **Hand Action Ablation.** We ablate on action output formulations on a subset of dexterous hands and objects and trained *without* curriculum. Hybrid actions with more restrictive bounds (light and dark green bars) shows better learning performance than absolute actions and full residual actions with less wrist constraints, both in training with task rewards or with both task plus auxiliary rewards settings