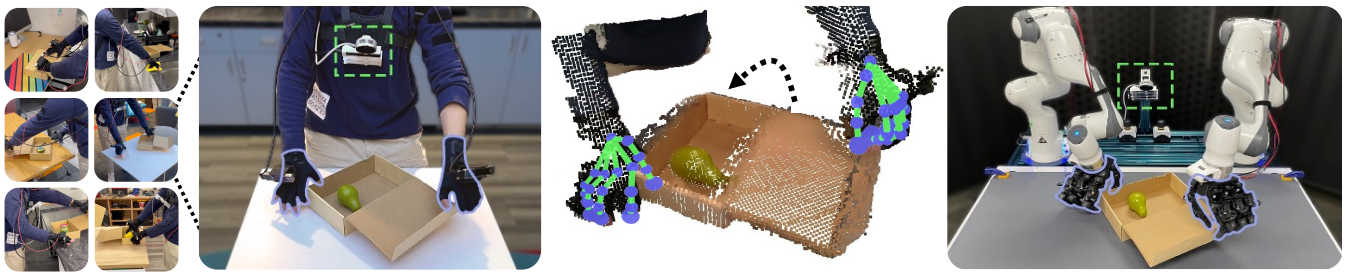
DexCap: Scalable and Portable Mocap Data Collection System for Dexterous Manipulation

Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, C. Karen Liu
Stanford University
https://dex-cap.github.io



(a) DexCap: Portable motion capture system (b) Mocap data and 3D scene (c) DexIL: Dexterous imitation learning

Fig. 1: **DexCap** facilitates the in-the-wild collection of high-quality human hand motion capture data and 3D observations. Leveraging this data, **DexIL** adapts it to the robot embodiment and trains control policy to perform the same task.

***Abstract*—Imitation learning from human hand motion data presents a promising avenue for imbuing robots with human-like dexterity in real-world manipulation tasks. Despite this potential, substantial challenges persist, particularly with the portability of existing hand motion capture (mocap) systems and the complexity of translating mocap data into effective robotic policies. To tackle these issues, we introduce DexCap, a portable hand motion capture system, alongside DexIL, a novel imitation algorithm for training dexterous robot skills directly from human hand mocap data. DexCap offers precise, occlusion-resistant tracking of wrist and finger motions based on SLAM and electromagnetic field together with 3D observations of the environment. Utilizing this rich dataset, DexIL employs inverse kinematics and point cloud-based imitation learning to seamlessly replicate human actions with robot hands. Beyond direct learning from human motion, DexCap also offers an optional human-in-the-loop correction mechanism during policy rollouts to refine and further improve task performance. Through extensive evaluation across six challenging dexterous manipulation tasks, our approach not only demonstrates superior performance but also showcases the system's capability to effectively learn from in-the-wild mocap data, paving the way for future data collection methods in the pursuit of human-level robot dexterity.**

I. Introduction

Building robotic systems to perform everyday manipulation tasks is a long-standing challenge. Our living environments and daily objects are designed with human hand functionality in mind, posing a substantial challenge for developing future home robots. Recent breakthroughs in robotic dexterity, especially in the control of multi-fingered mechanical hands with a high degree of freedom, have shown remarkable potential [1–3]. However, enabling robotic hands to emulate human-level dexterity in manipulation tasks remains unsolved, due to both hardware and algorithmic challenges.

Imitation Learning (IL) [4, 5] has recently made considerable strides toward this goal [6, 7], especially through supervised training using human demonstration data. One commonly used way to collect data is to teleoperate robot hands to perform the tasks. However, due to the requirement of a real robot system and slow robot motion, this approach is expensive to scale up. An alternative way is to directly track human hand motions during manipulation without controlling the robot. Current system is primarily vision-based with a single-view camera. However, besides the question of whether the tracking algorithm can provide accurate 3D information which is critical for robot policy learning, these systems are vulnerable to visual occlusions that frequently occur during hand-object interactions.

A better alternative to vision-based methods for gathering dexterous manipulation data is through motion capture (mocap). Mocap systems provides accurate 3D information and are robust to visual occlusions. Hence human operators can directly interact with the environment with their hands, which is fast and easier to scale up since no robot hardware is required. To scale up hand mocap systems to data collection in everyday tasks and environments for robot learning, a suitable system should ideally be portable and robust for long capture sessions, provide accurate finger and wrist poses, as well as 3D environment information. Most hand mocap systems are not portable and rely on well-calibrated third-view cameras. While electromagnetic field (EMF) gloves overcome this issue, they cannot track the 6-DoF wrist pose in the world frame, which is important for end-effectors policy learning. Devices like IMU-based whole-body suits can monitor wrist position but are prone to drift over time.

In addition to hardware challenges, there are also algorithmic challenges to use motion capture data for robot imitation learning. While dexterous robot hands enable the possibility of learning directly from human hand data, the inherent dif-

ferences in size, proportion, and kinematic structure between the robot hand and human hand call for innovative algorithms to overcome these embodiment gaps. Towards solving these challenges, our work simultaneously introduces a new portable hand mocap system, DEXCAP, and an imitation algorithm, DEXIL, that allows the robot to learn dexterous manipulation policies directly from the human hand mocap data.

DEXCAP (Fig. 1) is a portable hand mocap system that tracks the 6-DoF poses of the wrist and the finger motions in real-time (60Hz). The system includes a mocap glove to track finger joints, a camera mounted on top of each glove to track the 6-DoF poses of the wrists with SLAM, and an RGB-D LiDAR camera on the chest to observe the 3D environments.

Besides the hardware challenges, research efforts on developing algorithms to utilize mocap data for robot learning have been missing due to the lack of such a data collection system and collected data. Prior algorithms that learn from human motion focus on learning the rewards [8, 9], high-level plans [10, 11], and visual representations [12, 13], which often require additional robot data and cannot be directly used for low-level control. In this work, we argue that the main challenge of learning low-level control from human motion is that the data is missing precise 3D information of the hand motion (e.g., 6-DoF hand pose, 3D finger positioning), which are exactly what DEXCAP can provide.

To leverage data collected by DEXCAP for learning dexterous robot policies, we propose imitation learning from mocap data, DEXIL, which consists of two major steps — data retargeting and training generative-based behavior cloning policy with point cloud inputs, with an optional human-in-the-loop motion correction step. For retargeting, we use inverse kinematics (IK) to retarget the robotic hand's fingertips to the same 3D location as the human's fingertips. The 6-DoF pose of the wrist is used to initialize the IK to ensure the same wrist motion between the human and the robots. Then we convert RGB-D observations to point cloud-based representations. We then use a point cloud-based behavior cloning algorithm based on Diffusion Policy [14]. In more challenging tasks when IK is insufficient to fulfill the embodiment gap between human and robot hands, we propose a human-in-the-loop motion correction mechanism. During policy rollouts, humans can wear the DEXCAP and interrupt the robot's motion when unexpected behavior occurs, and such interruption data can be further used for policy finetuning.

In summary, the main contributions of this work include:

- DEXCAP: a novel portable human hand mocap system, enabling real-time tracking of wrist and finger movements for dexterous manipulation tasks.
- DEXIL: an imitation learning framework leveraging hand mocap data for directly learning dexterous manipulation skills from human hand motions.
- Human-in-the-Loop Correction: a human-in-the-loop correction mechanism with DEXCAP, significantly enhancing robot performance in complex tasks.

II. RELATED WORKS

A. Dexterous manipulation

Dexterous manipulation has been a long-standing research area in robotics [15–19], posing significant challenges to planning and control due to the high degrees-of-freedom. The traditional optimal control methods [17–19] often necessitate simplification of the contacts, which is usually not tenable in more complex tasks. Recently, reinforcement learning has been explored to learn dexterous policies in simulation with minimal assumptions about the task or the environment [2, 20–29]. The learned policies can solve complex tasks, including in-hand object re-orientatation [2, 20, 23–25, 28], bimanual manipulation [26, 30], and long-horizon manipulation [22, 27]. However, due to the sim-to-real gap, deploying the learned policy on a real-world robot remains challenging. Imitation learning, on the other hand, focuses on learning directly from real-world demonstration data, which is obtained through either teleportation [1, 6, 31, 32] or human videos [3, 33, 34]. DIME [31] uses VR to teleoperate a dexterous hand for data collection; Qin et al. [35] uses an RGB camera to track hand pose for teleoperation; DexTransfer [36] uses human mocap data to guide dexterous grasping; DexMV [33], DexVIP [34] and VideoDex [3] leverages human video data for learning the motion priors but often require additional training in simulation or real robot teleoperation data. Our work focuses on dexterous imitation learning, which relies on DEXCAP to collect high-quality hand mocap data grounded in 3D point cloud observation, which can be directly used to train low-level positional control on robots with single or dual hands.

B. Hand motion capture system

Human hand mocap is an important technique for applications in computer vision and graphics. Most previous systems are camera-based, IMU-based, or electromagnet(EMF)-based. Camera-based systems utilize monocular camera [37–39], RGB-D camera [40–42], VR headset [43], or multi-view camera with markers [44, 45]. However, the quality of hand motion tracking quickly deteriorates in scenarios involving heavy occlusions, which happen frequently in hand-object interactions. Some of these systems also require third-view calibrated cameras which are not portable or scalable. More recently, Inertia Measurement Unit (IMU) has been used for in-the-wild human mocap [46–50]. Nevertheless, most of them focus on whole-body motion capture and miss fine-grained finger motions. EMF-based mocap gloves are designed for capturing finger motion, which is widely used for dexterous teleoperation [51–53]. However, the glove does not track the 6-DoF palm poses grounded in the environment and misses visual observations for training robot policies. DEXCAP is a mocap glove system that is designed to collect data for training visuomotor manipulation policies. Through novel engineering designs, our system stays robust to occlusions, captures fine-grained finger motion, tracks palm poses using SLAM, and records RGB-D images to reconstruct the scene with a wearable camera vest.

(a) DexCap front view (b) DexCap back view (c) Details of the camera setup

Fig. 2: **Details of the human system.** (a) Our setup includes a 3D-printed rack on a chest harness, featuring a Realsense L515 LiDAR camera on top and three Realsense T265 tracking cameras below. (b) An Intel NUC and power bank in a backpack power the system for approximately 40 minutes of data collection. (c) The T265 cameras, initially in a known pose for calibration, are relocated to hand mounts during data collection to monitor palm positions, ensuring consistency through a click-in design. Finger motions are captured by Rokoko gloves, accurately tracking the finger joint positions.

C. Robot learning with human demonstration

Imitation Learning (IL) has enabled robots to successfully perform various manipulation tasks [4, 54–60]. Traditional IL algorithms such as DMP and PrMP [61–64] enjoy high learning sample efficiency but are limited in their ability to handle high-dimensional observations. In contrast, recent IL methods built upon deep neural networks can learn policies with raw image observation inputs [65, 66], even for high-degree robot systems with bimanual arms [67, 68]. Despite their effectiveness, one key challenge for imitation learning is how to scale up the training data. Prior works focus on teleoperation data [66, 69–77] which is expensive to collect due to the requirement of the robot hardware. More recently, learning from human motion data has started to receive more attention because it allows collecting data without robot hardware [78]. By leveraging human videos [11, 79], hand trajectories [10, 80–82], promising results have been shown to train policies with less manual human effort. However, these human motions are in 2D image space [80, 83, 84], which fails to directly train 6-DoF manipulation policies in 3D environments and usually requires additional teleoperation data to bridge the gap [10, 11, 79]. Recently, human-in-the-loop correction algorithms have also shown promising results in robot learning [85–87]. Our DexCap provides tracking of 6-DoF hand poses together with finger motions grounded in 3D point cloud observations, which is portable for data collection without a robot. Based on the data collected with DexCap, we introduce DexIL which is a point cloud-based imitation learning algorithm for learning fine-grained dexterous manipulation policies, with an optional human-in-the-loop correction step for more challenging tasks.

D. Portable data collection systems for manipulation

Recently advancements in low-cost hand-held grippers have shown promising results in collecting robot manipulation data without robot hardware [88–94]. All of these systems are designed and used for the parallel-gripper data collection process, while in this work we aim to collect multi-finger hand motion data for dexterous manipulation tasks (e.g., using scissors and unscrewing bottle caps).

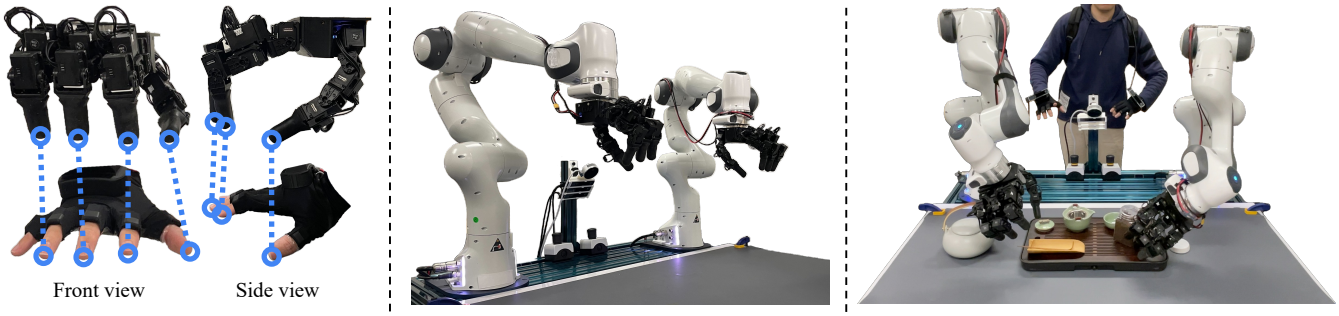
III. Hardware System: DexCap

In this section, we introduce the system design including (1) a portable human hand motion capture system DexCap that is used for data collection (Sec. III-A) and (2) a bimanual robot system equipped with dexterous hands for testing the policies learned from the collected data (Sec. III-B).

A. DexCap

To capture the fine-grained hand motion data suitable to train dexterous robot policies, DexCap is designed with four key objectives in mind: (1) detailed finger motion tracking, (2) accurate 6-DoF wrist pose estimation, (3) aligned 3D observations recording in a unified coordinate frame with hands, and (4) outstanding portability for data collection in various real-world environments. We achieved these objectives with zero compromise on *scalability*—DexCap must be simple to calibrate, inexpensive to build, and robust for data collection of daily activities in the wild.

Tracking finger motions. Our system uses electromagnetic field (EMF) gloves, offering a significant advantage over vision-based finger tracking systems, particularly in the robustness to visual occlusions that frequently occur in hand-object interactions. In our system, finger motions are tracked using Rokoko motion capture gloves as illustrated in Figure 2. Each glove's fingertip is embedded with a tiny magnetic sensor, while a signal receiver hub is placed on the glove's dorsal side. The 3D location of each fingertip is measured as the relative 3D translation from the hub to the sensors. In appendix we included a qualitative comparison between our EMF glove system and state-of-the-art vision-based hand-tracking methods across different manipulation scenarios.



(a) Retargeting with fingertip IK (b) Bimanual dexterous robot setup (c) Human-in-the-loop correction setup

Fig. 3: **Details of the robot system.** Mirroring the human system, the robot system reuses the same chest cameras and mount. (a) Once the motion is captured by DexCap, it's retargeted to LEAP hand through discarding pinky finger and IK to match fingertip location. (c) An optional human-in-the-loop correction step can be performed to further refine the motions transferred. Specifically, the human will provide the delta input in real time when the robot system is carrying out the task. Note the hand T265 is only used at correction time, as the robot arm already knows the exact location of fingers.

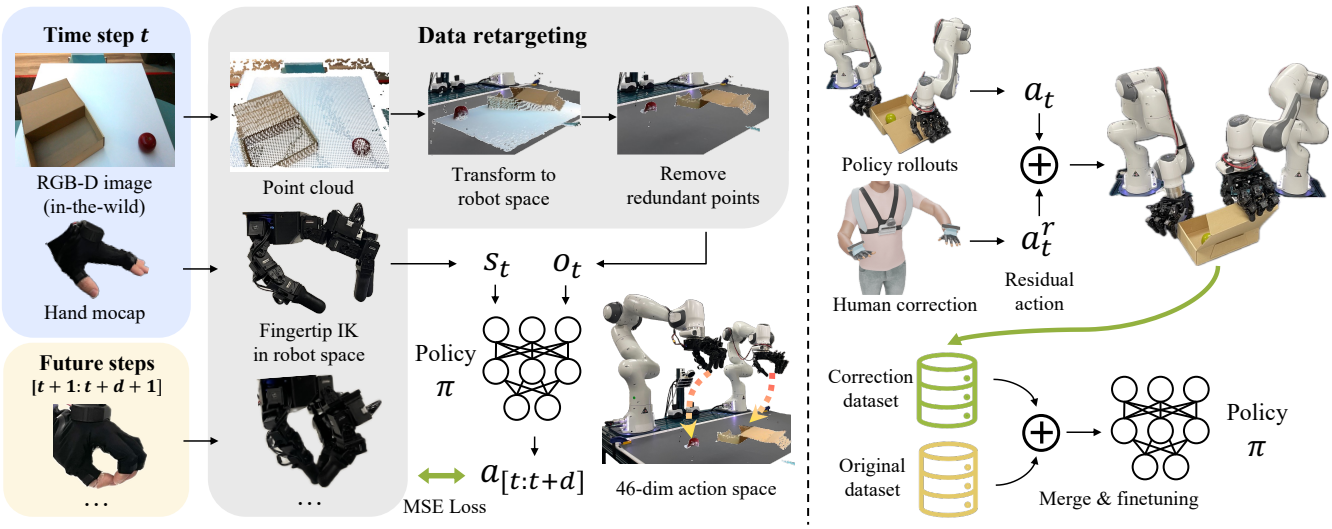
Tracking 6-DoF wrist pose. Beyond finger motion, knowing the precise positioning of a robot's end-effector in a 3D space is crucial for robot manipulation. This necessitates DexCap to estimate and record the 6-DoF pose trajectories of human hands during data collection. While camera-based and IMU-based methods are commonly used, each has its limitations. Camera-based systems, often non-portable and limited in their ability to estimate wrist orientation, are less suited for data collection in manipulation tasks. IMU-based systems, although wearable, tend to suffer from position drifting when used for long recording sessions. To address these challenges, we develop a 6-DoF wrist tracking system based on the SLAM algorithm, as shown in Figure 2(c). This system uses an Intel Realsense T265 camera, mounted on each glove's dorsal side. It combines images from two fisheye cameras and IMU sensor signals to construct an environment map using the SLAM algorithm, enabling consistent tracking of the wrist's 6-DoF pose. This design has three key advantages: it is portable, allowing for wrist pose tracking without the need for hands to be visible in third-person camera frames; SLAM can autonomously correct position drift with the built map for long-time use; and the IMU sensor provides crucial wrist orientation information to train the robot policy in the subsequent pipeline.

Recording 3D observations and calibration. Capturing the data necessary for training robot policies requires not only the tracking of hand movement but also recording observations of the 3D environment as the policy input. As depicted in Figure 2(a), we design a wearable camera vest for this purpose. It incorporates an Intel Realsense L515 RGB-D LiDAR camera, mounted on the top of the chest, to capture the observations during human data collection. The next critical question then becomes how to effectively integrate the tracked hand motion data with the 3D observations. To simplify the calibration process, we designed a 3D-printed camera rack underneath the chest camera mount as illustrated in Figure 2(c). At the beginning of the data collection, all tracking cameras are placed in the rack slots, which secures a constant transformation between the camera frames. Then, we take off the tracking cameras from the rack and insert them into the camera slot attached to each glove. In this way, we can easily transform the hand pose tracking results into the observation frame of the chest camera with the constant initial transformation. The full calibration process is demonstrated in Appendix Figure 13 and supplementary videos, which takes around 10 seconds. To further ensure stable observations amidst human movement, another fisheye tracking camera (marked red in Fig. 2(c)) is mounted under the LiDAR camera, which provides a more robust SLAM performance than the LiDAR camera with its wide field of view. We define the initial pose frame of this tracking camera as the world frame for all stream data. Figure 6 is the visualization of the collected data by transforming the observations into colored point clouds in the world frame alongside the captured hand motions.

System Portability. Central to the portability of DexCap is a compact mini-PC (Intel NUC 13 Pro), carried in a backpack, which serves as the primary computation unit for data recording. This PC is powered by a portable power bank with a 40000mAh battery, enabling approximately 40 minutes of continuous data collection (Fig. 2(b)). The total weight of the backpack is 3.96 pounds. The supplementary video shows that donning and calibrating DexCap is fast and simple, taking less than 10 seconds. Additionally, DexCap's hardware design is modular and inexpensive to build — no restriction to brands or models of cameras, motion capture gloves, and mini-PCs. We will open-source the code and instruction videos for builders, along with a range of hardware options. The overall cost of the DexCap is kept within a $4k USD budget.

B. Bimanual dexterous robot

To validate the robot policy trained by the data from DexCap, we establish a bimanual dexterous robot setup. This setup comprises two Franka Emika robot arms, each equipped with a LEAP dexterous robotic hand (a four-fingered hand with 16 joints) [95], as depicted in Figure 3(b). For policy evaluation, the chest LiDAR camera used in human data collection is detached from the vest and mounted on



(a) DexIL overview (b) Human-in-the-loop correction

Fig. 4: **Algorithm overview.** (a) DexIL first retargets the DexCap data to the robot embodiment by first constructing 3D point clouds from RGB-D observations and transforming it into robot operation space. Meanwhile, the hand motion capture data is retargeted to the dexterous hand and robot arm with fingertip IK. Based on the data, a robot policy is learned to output a sequence of future goal positions as the robot actions. (b). DexCap also offers an optional human-in-the-loop correction mechanism, where humans apply delta residual action to the policy-generated actions to correct robot behavior. The corrections are stored in a new dataset and uniformly sampled with the original dataset for fine-tuning the robot policy.

a stand positioned between the robot arms. To simplify the process of switching the camera system between the human and robot, a quick-release buckle has been integrated into the back of the camera rack, allowing for swift camera swaps – in less than 5 seconds. In this way, the robot utilizes the same observation camera employed during human data collection. Note that, for robot setups, only the LiDAR camera is used and wrist cameras are not needed. Both the robot arms and the LEAP hands operate at a control frequency of 20Hz. We use end-effector position control for both robot arms and joint position control for both LEAP hands.

IV. Learning Algorithm: DexIL

Our goal is to use the human hand motion capture data recorded by DexCap to train dexterous robot policies. There are several research questions along the way - (1) How can we re-target the human hand motion to the robotic hand? (2) What algorithm can learn dexterous policies, especially when the action space is high-dimensional in the bimanual setup? (3) In addition, we would like to investigate the failure cases for learning directly from human motion capture data and their potential solutions.

To tackle these challenges, we introduce DexIL, a three-step framework to train dexterous robots using human hand motion capture data. The first step is to re-target the DexCap data into the action and observation spaces of the robot embodiment (Sec. IV-A). Second step trains a point-cloud-based diffusion policy using the re-targeted data (Sec. IV-B). The final step involves an optional human-in-the-loop correction mechanism, designed to address unexpected behaviors that emerge during the policy execution (Sec. IV-C).

A. Data re-targeting

Action re-targeting. As illustrated in Figure 3(a), a notable challenge emerges due to the size disparity between the human hand and the LEAP hand, with the latter about 50% larger [95]. This size difference makes it hard to directly transfer the finger motions to the robotic hardware. The first step is to retarget the human hand motion capture data into the robot embodiment, which requires mapping the finger position and 6-DoF palm pose with inverse kinematics (IK).

One critical finding in prior research is that fingertips are the most frequently contacted areas on a hand when interacting with objects (as evidenced in studies like HO-3D [41], GRAB [44], ARCTIC [45]). Motivated by this, we re-target finger motion by matching fingertip positions using inverse kinematics (IK). Specifically, we deploy an IK algorithm that generates smooth and accurate fingertip motion in real time [96–98] to determine the 16-dimensional joint positions for the robotic hand. This ensures the alignment between robot fingertips and the human fingertips in the DexCap data. Considering the design of the LEAP hand, which features four fingers, we adapt our process by excluding little finger information during IK computations. Additionally, the 6-DoF wrist pose captured in the mocap data serves as an initial reference for wrist pose in the IK algorithm. Figure 6 demonstrates the final result of re-targeting. The 6-DoF pose of the wrist $\boldsymbol{p}_t = [\boldsymbol{R}_t|\boldsymbol{T}_t]$ and the finger joint positions $\boldsymbol{J}_t$ of the LEAP hands are then used as the robot's proprioception state $\boldsymbol{s}_t = (\boldsymbol{p}_t, \boldsymbol{J}_t)$. We use position control in our setup and the robot's action labels are defined as next future states $\boldsymbol{a}_t = \boldsymbol{s}_{t+1}$.

Observation post-processing. Observation and state rep-

resentation choice are critical for training robot policies. We convert the RGB-D images captured by the LiDAR camera in the DexCap data into point clouds using the camera parameters. This additional conversion offers two significant benefits compared to RGB-D input. First, because DexCap allows the human torso to move naturally during data acquisition, directly using RGB-D input would need to account for the moving camera frame. By transforming point cloud observations into a consistent world frame—defined as the coordinate frame of the main SLAM camera at the start of the mocap (the main camera is marked in red in Fig. 2(c))—we isolate and remove torso movements, resulting in a stable robot observation. Second, point clouds provide flexibility in editing and alignment with the robot's operational space. Given that some motions captured in the wild may extend beyond the robot's reachability, adjusting the placement of point cloud observations and motion trajectories ensures their feasibility within the robot's operational range. Based on these findings, all RGB-D frames from the mocap data are processed into point clouds aligned with the robot's space, and the task-irrelevant elements, such as the table surface points, are excluded. This refined point cloud data thus becomes the observation inputs $\boldsymbol{o}_t$ fed into the robot policy π.

B. Point cloud-based diffusion policy

With the transformed robot's state $\boldsymbol{s}_t$, action $\boldsymbol{a}_t$ and corresponding 3D point cloud observation $\boldsymbol{o}_t$, we formalize the robot policy learning process as a trajectory generation task. More specifically, a policy model π, processes the point cloud observations $\boldsymbol{o}_t$ and the robot's current proprioception state $\boldsymbol{s}_t$ into an action trajectory $(\boldsymbol{a}_t, \boldsymbol{a}_{t+1}, \ldots, \boldsymbol{a}_{t+d})$ (as in Fig. 4). Given point cloud observation with N points $\boldsymbol{o}_t$ in $\mathbb{R}^{N\times 3}$, we uniformly down-sample it into K points and concatenate the RGB pixel color corresponding to each point into the final policy input in $\mathbb{R}^{K\times 6}$. To bridge the visual gap between human hands and the robot's hand, we use forward kinematics to transform the links of the robot model with the proprioception state $\boldsymbol{s}_t$ and merge the point clouds of the transformed links into the observation $\boldsymbol{o}_t$. During training, we also use data augmentation over the inputs by applying random 2D translations to the point clouds and motion trajectories within the robot's operational space.

One challenge of learning dexterous robot policies, especially for bimanual dexterous robots, is handling the large dimensional action outputs. In our setup, the action output includes two 7-DoF robot arms and two 16-DoF dexterous hands for d steps, which forms a high-dimensional regression problem. Similar challenges have also been studied in image generation tasks, which aim to regress all pixel values in a high-resolution frame. Recently, diffusion model [99, 100], with its step-by-step diffusion process, has shown success in modeling complex data distributions with high-dimensional data. For robotics, diffusion policy [14] follows the same idea and formalizes the control problem into an action generation task. Thus we use a diffusion policy as the action decoder, where we empirically find it outperforms traditional MLP-based architecture for learning dexterous robot policies.

C. Human-in-the-loop correction

With the design presented above, DexIL can learn challenging dexterous manipulation skills (e.g., pick-and-place and bimanual coordination) directly from DexCap data without the need for on-robot data. However, our simple retargeting method does not address all aspects of the human-robot embodiment gap. For example, when using a pair of scissors, a stable hold of scissors requires inserting the fingers deep into the handle. Due to the differences in finger length proportion, directly matching the fingertips and the joint motion does not guarantee the same force exerted on the scissors.

To address this issue, we offer a human-in-the-loop motion correction mechanism, which consists of two modes - residual correction and teleoperation. During policy execution, we allow humans to provide corrective actions to robots in real-time by wearing DexCap. In residual mode, DexCap measures the delta position changes of human hands $(\Delta\boldsymbol{p}_t^H, \Delta\boldsymbol{J}_t^H)$ relative to hands' initial states $(\boldsymbol{p}_0^H, \boldsymbol{J}_0^H)$ at the beginning of the policy roll-out. The delta position is applied as a residual action $\boldsymbol{a}_t^r = (\Delta\boldsymbol{p}_t^H, \Delta\boldsymbol{J}_t^H)$ to the robot policy action $\boldsymbol{a}_t = (\boldsymbol{p}_{t+1}, \boldsymbol{J}_{t+1})$, scaled by α and β. The corrected robot action can then be formalized as $\boldsymbol{a}_t' = (\boldsymbol{p}_{t+1} \bigoplus \alpha \cdot \Delta\boldsymbol{p}_t^H, \boldsymbol{J}_{t+1} + \beta \cdot \Delta\boldsymbol{J}_t^H)$. We empirically find that setting β with a small scale (< 0.1) offers the best user experience, which avoids fingers moving too fast.

In the case when a large position change is desired, a press on the foot pedal will switch the system to teleoperation mode. DexCap now ignores the policy rollout and applies human wrist delta directly to the robot wrist pose. The robot fingertips are now directly following human fingertips. In other words, the robot fingertip will track the human fingertip in their respective wrist frame through IK. Users can also switch back to the residual mode after correcting the robot's mistake by pressing the foot pedal again.

Since the robot has already learned an initial policy, typically the correction happens in a small portion of the rollout, greatly reducing the human effort. The corrected actions and observations are stored in a new dataset $\mathcal{D}'$. Training data is sampled with equal probability from $\mathcal{D}'$ and the original dataset $\mathcal{D}$ to fine-tune the policy model, similar to IWR [101].

V. Experiments

We aim to answer the following research questions:

- $\mathcal{Q}1$: What is the quality of DexCap data?
- $\mathcal{Q}2$: Can DexIL directly learn dexterous robot policies from DexCap data without any on-robot data?
- $\mathcal{Q}3$: What model architecture choices are critical to improving the performance?
- $\mathcal{Q}4$: Can DexIL learn from in-the-wild DexCap data?
- $\mathcal{Q}5$: How does human-in-the-loop correction help when DexCap data is insufficient?

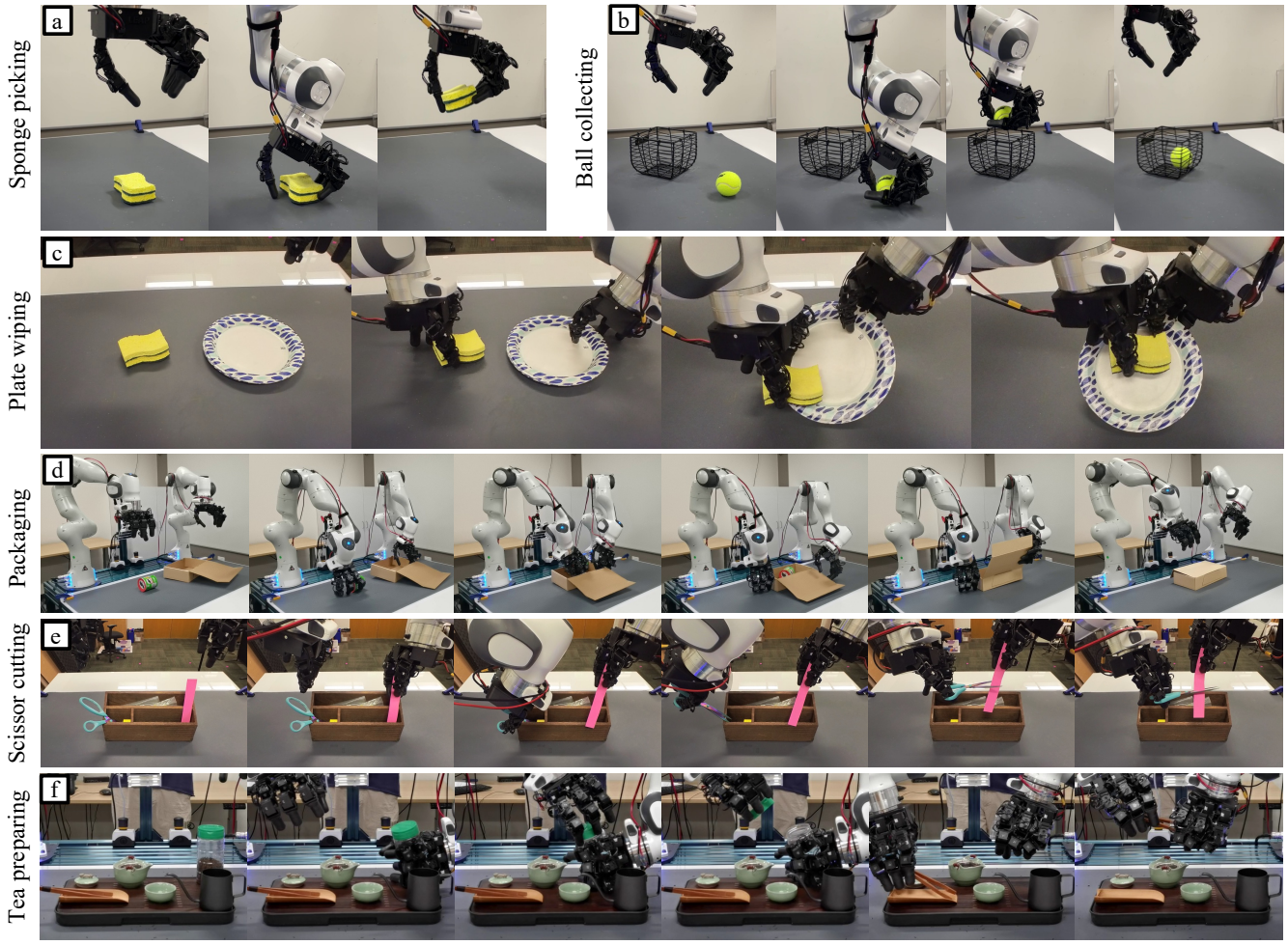


Fig. 5: **Experiment Tasks.** (a) *Sponge Picking*: Pick and lift the sponge. (b) *Ball Collecting*: Pick up a ball and drop it into a basket. (c) *Plate Wiping*: Use both hands to pick up a plate and sponge, then wipe the plate vertically twice. (d) *Packaging*: Place items into a box with one hand while using the other to either push or stabilize them, before securely closing the box lid. (e) *Scissor Cutting*: Secure paper with one hand and use scissors in the other to cut through the paper. (f) *Tea Preparing*: Grasp the tea bottle with one hand, use the other hand to uncap, then pick up tweezers to extract tea and pour it into the pot.

$\mathcal{Q}$**6**: Can the whole framework handle extremely challenging bimanual dexterous manipulation tasks (e.g., using scissors and preparing tea)?

A. Experiment setups

a) Tasks: we evaluate DexIL using six tasks of varying difficulty to assess its performance with DexCap data. These tasks range from basic, such as *Sponge picking*, *Ball collecting*, and *Plate wiping*, which test single-handed and dual-handed coordination, to more complex ones like *Packaging*, which looks at bimanual tasks and generalization using both familiar and new objects. *Scissor cutting* focuses on the effectiveness of the human-in-the-loop correction mechanism in precise tool use, whereas *Tea preparing* challenges the system with a long-horizon task requiring intricate actions. To further analyze performance, we introduce the **Subtask** metric for multi-step tasks, indicating the completion of task subgoals, such as placing an object inside a box in *Packaging*, or picking up scissors in *Scissor Cutting*.

b) Data: We utilize two data types: (1) DexCap *data* capturing human hand motion (In-the-wild data refers to a mixture of data collected in more than 10 scenes) and (2) *human-in-the-loop correction data* for adjusting robot actions or enabling teleoperation to correct errors, collected using a foot pedal. Data were initially recorded at 60Hz and then downsampled to 20Hz to match the robot's control speed, except for correction data, which was collected directly at 20Hz. For data collection, we gathered 30 minutes of DexCap *data* across the first three tasks, resulting in 251, 179, and 102 demos respectively. An hour of *in-the-wild* DexCap *data* provided 104 demos for *Packaging*. *Scissor Cutting* and *Tea Preparing* tasks each received an hour of DexCap data, yielding 96 and 55 demos respectively.

c) Baselines: We evaluate multiple baselines to determine the model architecture with the best performance, focusing on three key aspects using DexCap data: identifying the best imitation learning framework for bimanual dexterous

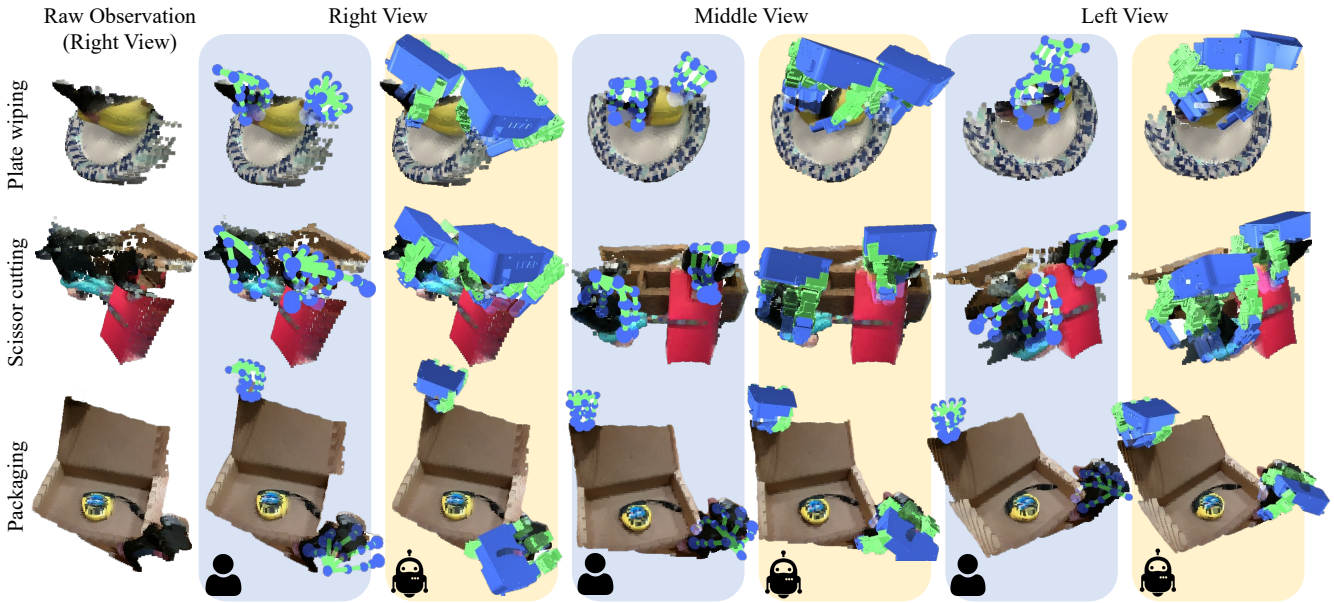


Fig. 6: **Data Retargeting for Tasks.** DEXIL effectively retargets human mocap data for activities like plate wiping, scissor cutting, and packaging. The initial column displays the raw point cloud scene. Columns 2-7 offer three views—right, middle, left—with blue background columns depicting human data and yellow for robot hand retargeting. This side-by-side arrangement highlights the precision of our fingertip IK in translating human to robot hand motions.

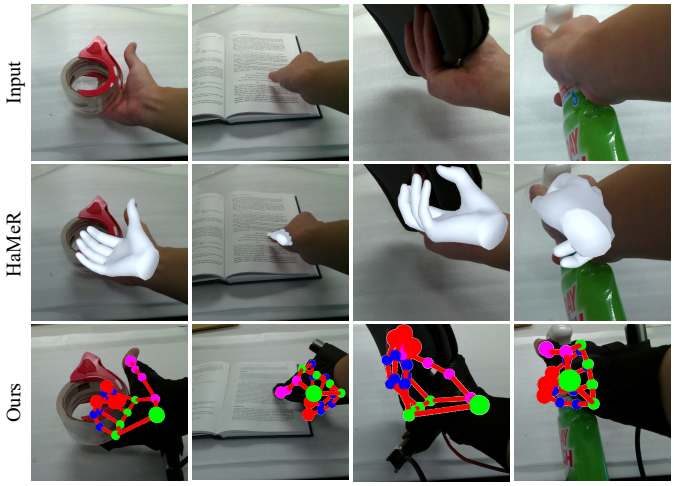


Fig. 7: **Compare with vision-based method.** We demonstrate that motion capture gloves provide more stable hand pose estimation results compared to vision-based methods and are not affected by visual occlusion.

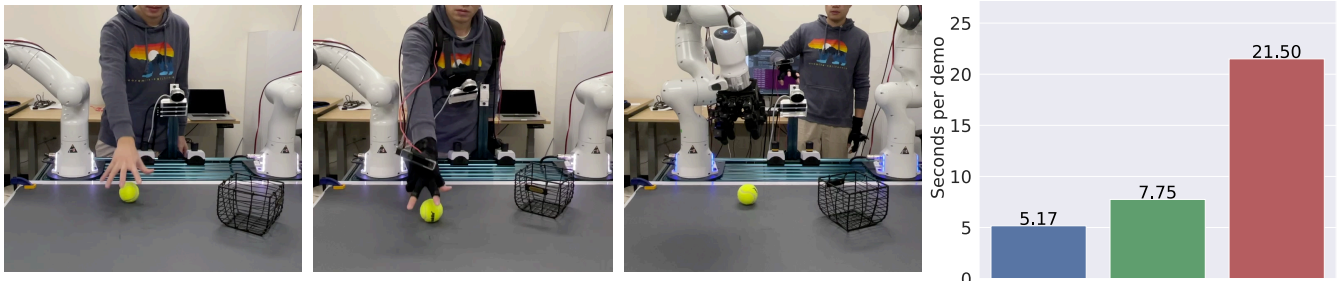
manipulation between BC-RNN [102] and diffusion policy (DP)[14], assessing the most effective observation type to bridge the visual gap between human and robot hands (comparing image inputs [14, 65] and a point cloud method [103]), and determining the most suitable encoder for point cloud inputs by comparing PointNet[104] and Perceiver [105, 106] encoders. Implementation details are included in the appendix.

d) Metric: Each model variant is tested for 20 trials in each task with randomized initial placements. The task success rate is reported in Table I II III. For the multi-object *Packaging* task, each object is tested with 5 trials - 6 trained objects (30 total trials) and 9 unseen objects (45 total trials).

B. Results

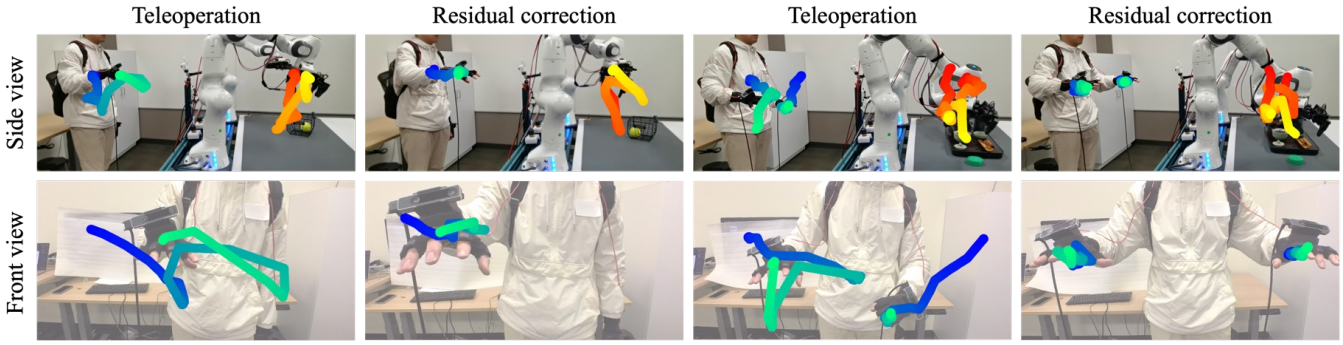
DEXCAP delivers high-quality 3D mocap data ($\mathcal{Q}1$). Figure 6 showcases DEXCAP's ability to capture detailed hand motion in 3D, aligning human actions with object point clouds across all views, such as in *Plate wiping* and *Scissor cutting* tasks (blue columns). The retargeted robot hand motions, depicted in the yellow columns, demonstrate precise alignment in the same 3D space. In Figure 7, we compare DEXCAP with the state-of-the-art vision-based hand pose estimation method HaMeR [39], observing their performance from similar viewpoints. We find that the vision-based approach is vulnerable to self-occlusion, particularly when the fingers are obscured. As depicted in Figure 7, HaMeR struggles in instances of significant occlusion, either failing to detect the hand (as seen in the second column) or inaccurately estimating fingertip positions (noted in the first, third, and fourth columns). In contrast, DEXCAP demonstrates good robustness under these conditions. Beyond the challenge of occlusion, most vision-based methods rely on 2D hand estimation, predicated on learning from 2D image projection losses. Consequently, these methods are inherently limited in their ability to discern the precise 3D hand positioning, as they are trained based on presumed, fixed camera intrinsic parameters, which do not necessarily match the actual camera used for experiments. In Figure 8, we showcase the data collection throughput of DEXCAP, which is three times faster than traditional teleoperation.

DEXCAP data can directly train dexterous robot policies ($\mathcal{Q}2$). Table I is the experiment result of training robot policies only using DEXCAP data. Within 30-minute hand motion capture demonstrations collected by DEXCAP, the learned policies



(a). Human motion (b). DexCap (c). Teleoperation

Fig. 8: **Data collection throughput comparison.** DEXCAP's data collection speed in the *Ball collecting* task is close to natural human motion and is three times faster than traditional teleoperation.



(a) Ball collecting (b) Tea preparing

Fig. 9: **Visualization of human-in-the-loop corrections.** DEXCAP supports teleoperation and residual correction for human-in-the-loop adjustments. Teleoperation directly translates human hand movements to the robot end-effector actions, indicated by color-fading trajectories from blue to green (human) and red to yellow (robot) over 20 timesteps. Residual correction adjusts the robot's end-effector based on changes from the human hand's initial pose, enabling minimal movement but requiring more precise control. Users can switch between correction modes with a foot pedal.

achieve up to 72% average task success rate in single-hand pick-and-place (*Sponge picking*, *Ball collecting*) and bimanual coordination (*Plate wiping*) tasks. This result highlights the effectiveness of DEXCAP data on training dexterous robot policies without on-robot data, which introduces a new way for training robot dexterous manipulation.

Generative-based algorithm with point cloud inputs shows advantages ($\mathcal{Q}$3). In Table I, we compare the performance of multiple model architectures. We first observe that, due to the visual appearance gap between human and robot hands, the policies with full image inputs fail completely (BC-RNN-img, DP-img). We then try masking out human and robot hands with white circles in training and evaluation. This setting brings improvements, where DP-img-mask achieves more than 30% success rate in all tasks. Meanwhile, diffusion policy works better than MLP-based BC-RNN policies (25% higher in averaged task success rate). This result verifies our hypothesis that generative-based policy is more suitable for learning dexterous policies. Although getting promising results, masking out the end-effector loses details for in-hand manipulation. This hypothesis is verified by the low success rate in the *Plate wiping* task, which requires the robot to use fine-grained finger motion to grab the plate from the edge. Our point cloud-based learning algorithms (DP-point-raw, DP-point, DP-prec), on the other hand, do not require masking over observations and achieve more than 60% task success rate. This result highlights the advantage of using point cloud inputs, which allow us to add robot hand points to the observation without losing the details in the original inputs. We also observe that, even without adding robot hand points, DP-point-raw achieves close performance to DP-point. This might because the downsampling process of the point cloud inputs lowers the appearance gap between human gloves and robot hands. Furthermore, compared to the PointNet, the model with Perceiver encoder has higher performance, especially in bimanual tasks with multiple task objects (20% improvement on task success rate in *Plate wiping*). Based on these findings, we use DP-perc as the default model architecture for DEXIL.

DEXIL can purely learn from in-the-wild DEXCAP data ($\mathcal{Q}$4). The first three columns of Table II are the results of training policies using in-the-wild DEXCAP data. We first notice that image-input baselines (BC-RNN-img-mask, DP-img-mask) have close to zero performance when learning with in-the-wild data. This observation verifies our hypothesis that the viewpoint changes caused by human body movements during in-the-wild data collection bring challenges to learning image-based policies. Our DEXIL transforms the point cloud inputs into a consistent world frame, resulting in stable

	DexCap Data Only			
	Sponge picking	Ball collecting	Plate wiping	Overall
BC-RNN-img	0.00	0.00	0.00	0.00
BC-RNN-img-mask [65]	0.25	0.10	0.10	0.15
BC-RNN-point [103]	0.45	0.30	0.25	0.33
BC-RNN-prec [105]	0.50	0.30	0.35	0.38
DP-img	0.00	0.00	0.00	0.00
DP-img-mask [14]	0.55	0.40	0.30	0.42
DP-point-raw	0.70	**0.70**	0.40	0.60
DP-point	0.75	0.65	0.50	0.63
Ours (DP-perc)	**0.85**	0.60	**0.70**	**0.72**

TABLE I: Quantitative results for learning with DexCap data.

Packaging	In-the-wild DexCap			30 human corrections		
	Subtask	All	Unseen	Subtask	All	Unseen
BC-RNN-img-mask [65]	0.00	0.00	0.00	0.23	0.07	0.00
BC-RNN-point [103]	0.33	0.23	0.16	0.40	0.27	0.22
DP-img-mask [14]	0.17	0.00	0.00	0.47	0.33	0.00
Ours	**0.70**	**0.47**	**0.40**	**0.83**	**0.57**	**0.42**

TABLE II: Quantitative results for the *Packaging* task.

Trained objects Unseen objects

Fig. 10: **Objects used in the *Packaging* task**

observations and thus getting better results (70% in Subtask and 47% in full task setup). Please refer to our video results for more visualization of the stabilized input point clouds. By training the policy with multiple task objects using in-the-wild (Fig. 10), our model can already generalize to unseen object instances, with a 40% success rate. During evaluation, we identified two primary issues with the policy learned from in-the-wild DexCap data: firstly, the absence of force information in DexCap data causes the right hand to struggle with stabilizing the box during box closure attempts by the left hand. Secondly, the box lid occasionally moves out of the chest camera's view due to human movements, hindering the robot's ability to learn precise lid grasping. These challenges prompt us to seek improvement strategies.

Human-in-the-loop correction greatly help when DexCap data is insufficient ($\mathcal{Q}5$). Figure 9 illustrates two types of human-in-the-loop correction mode with DexCap. Users can switch between the two modes by stepping on the foot pedal and the whole trajectory is stored and used for fine-tuning the policy. The last three columns of Table II showcase the effectiveness of using human-in-the-loop correction together with policy fine-tuning to improve the model performance. With just 30 human correction trials during policy rollout, the fine-tuned policy with image inputs (DP-img-mask) achieves a 33% improvement in the full task success rate for trained objects. This significant boost is mainly because the human correction data is collected using a fixed camera - the same setup used for the evaluations. This result further supports our conclusion: image-based approaches are more effective in learning with fixed third-view cameras compared to the in-the-wild scenarios with moving cameras. Human corrections also result in a 10% improvement in our approach that utilizes point cloud inputs. However, we've observed that fine-tuning with human corrections has a minor effect on the results for unseen objects, primarily due to the limited amount of correction data (30 trials in total).

Scissor cutting	DexCap Data Only		30 human corrections	
	Subtask	All	Subtask	All
BC-RNN-point [103]	0.00	0.00	0.10	0.00
Ours	0.00	0.00	**0.45**	**0.20**

TABLE III: Quantitative results for the *Scissor cutting* task.

Tea preparing	DexCap Data Only		30 human corrections	
	Subtask	All	Subtask	All
Ours	0.30	0.00	0.65	0.25

TABLE IV: Quantitative results for the *Tea preparing* task.

Our whole framework can handle extremely challenging tasks ($\mathcal{Q}6$). DexIL together with human-in-the-loop correction is able to solve extremely challenging tasks such as *Scissor cutting* and *Tea preparing*. In Table III, we showcase that our system can achieve a 45% success rate on picking up the scissor from the container and 20% in cutting a piece of paper tape. In our supplementary video, we also showcase how the robot performs the long-horizon *Tea preparing* task which includes unscrewing a bottle cap and pouring tea into the pot. Table IV presents the evaluation results of our approach (DP-perc) in the *Tea preparing* task. The subtask is defined as successfully unscrewing the cap of the tea bottle. We found that even with human mocap data only (DexCap Data Only), our model can achieve a 30% success rate in uncapping. Most of the failures occur during the task of picking up the tweezers, which requires high-precision control over the fingertip. In such cases, human-in-the-loop correction significantly improves performance. With 30 human corrections, we achieve a 35% improvement in the uncapping success rate and attain a 25% success rate for the entire task. Please refer to our video submission for more qualitative results of this task. These tasks showcase the high potential of our framework in learning extremely challenging dexterous manipulation tasks.

VI. Conclusion and Limitations

We present DexCap, a portable hand motion capture system, and DexIL, an imitation algorithm enabling robots to learn dexterous manipulation directly from human mocap data. DexCap, designed to overcome occlusions, capture fine-grained 3D hand motion, record RGB-D observations, and allow data collection outside the lab. DexIL applies this data to teach robots complex dexterous manipulation tasks, with an optional human-in-the-loop correction mechanism to further improve performance. Demonstrating proficiency in tasks like scissor cutting and tea preparation, DexCap and DexIL significantly advance robotic dexterity. We hope DexCap can

pave the path for future research on scaling up dexterous manipulation data with portable devices. All hardware designs and code will be open-source.

While DexCap collects high-quality mocap data in-the-wild for learning challenging dexterous manipulation tasks, it has several limitations that need future research: (1) The system's power consumption currently restricts the collection time to be at most 40 minutes. Future improvements will focus on enhancing power efficiency to extend the collection time. (2) Our learning algorithm DexIL utilizes fingertip inverse kinematics to retarget human hand motion to various robotic hands. However, the size difference between human and robotic hands (with some robotic fingers being thicker) can make some tasks difficult to perform, such as playing the piano. Future developments will aim to integrate advancements in robotic hand design to minimize these size differences and fully demonstrate the system's potential. (3) Current DexCap collects only 3D observations and motion capture data, lacking force sensing. One promising direction we plan to explore involves the use of conformal tactile textiles, as introduced in [107], to gather tactile information during data collection.

Acknowledgments

This research was supported by National Science Foundation NSF-FRR-2153854 and Stanford Institute for Human-Centered Artificial Intelligence, SUHAI. This work is partially supported by ONR MURI N00014-21-1-2801. We would like to thank Yunfan Jiang, Albert Wu, Paul de La Sayette, Ruocheng Wang, Sirui Chen, Josiah Wong, Wenlong Huang, Yanjie Ze, Christopher Agia, Jingyun Yang and the SVL PAIR group for providing help and feedback. We also thank Zhenjia Xu, Cheng Chi, Yifeng Zhu for their suggestions on the robot controller. We especially thank Kenneth Shaw, Ananye Agrawal, Deepak Pathak for open-sourcing the LEAP Hand.

References

[1] Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter Fox. Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9164–9170. IEEE, 2020.

[2] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand dexterous manipulation from depth. *arXiv preprint arXiv:2211.11744*, 2022.

[3] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. Videodex: Learning dexterity from internet videos. *CoRL*, 2022.

[4] Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, 3(6):233–242, 1999.

[5] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

[6] Irmak Guzey, Ben Evans, Soumith Chintala, and Lerrel Pinto. Dexterity from touch: Self-supervised pre-training of tactile representations with robotic play. *arXiv preprint arXiv:2303.12076*, 2023.

[7] Aravind Sivakumar, Kenneth Shaw, and Deepak Pathak. Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube. *arXiv preprint arXiv:2202.10448*, 2022.

[8] Laura Smith, Nikita Dhawan, Marvin Zhang, Pieter Abbeel, and Sergey Levine. Avid: Learning multi-stage tasks via pixel-level translation of human videos. *arXiv preprint arXiv:1912.04443*, 2019.

[9] Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022.

[10] Chen Wang, Linxi Fan, Jiankai Sun, Ruohan Zhang, Li Fei-Fei, Danfei Xu, Yuke Zhu, and Anima Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. *arXiv preprint arXiv:2302.12422*, 2023.

[11] Homanga Bharadhwaj, Abhinav Gupta, Vikash Kumar, and Shubham Tulsiani. Towards generalizable zero-shot manipulation via translating human interaction plans. *arXiv preprint arXiv:2312.00775*, 2023.

[12] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[13] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

[14] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[15] J Kenneth Salisbury and John J Craig. Articulated hands: Force control and kinematic issues. *The International journal of Robotics research*, 1(1):4–17, 1982.

[16] Matthew T Mason and J Kenneth Salisbury Jr. Robot hands and the mechanics of manipulation. 1985.

[17] Igor Mordatch, Zoran Popović, and Emanuel Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pages 137–144, 2012.

[18] Yunfei Bai and C Karen Liu. Dexterous manipulation using both palm and fingers. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1560–1565. IEEE, 2014.

[19] Vikash Kumar, Yuval Tassa, Tom Erez, and Emanuel Todorov. Real-time behaviour synthesis for dynamic hand-manipulation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6808–6815. IEEE, 2014.

[20] Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys

Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. *arXiv preprint arXiv:2210.13702*, 2022.

[21] Tao Chen, Jie Xu, and Pulkit Agrawal. A system for general in-hand object re-orientation. *Conference on Robot Learning*, 2021.

[22] Abhishek Gupta, Justin Yu, Tony Z. Zhao, Vikash Kumar, Aaron Rovinsky, Kelvin Xu, Thomas Devlin, and Sergey Levine. Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention. In *ICRA*, pages 6664–6671. IEEE, 2021.

[23] Zhao-Heng Yin, Binghao Huang, Yuzhe Qin, Qifeng Chen, and Xiaolong Wang. Rotating without seeing: Towards in-hand dexterity through touch. *arXiv preprint arXiv:2303.10880*, 2023.

[24] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-Hand Object Rotation via Rapid Motor Adaptation. In *Conference on Robot Learning (CoRL)*, 2022.

[25] Gagan Khandate, Siqi Shang, Eric T Chang, Tristan L Saidi, Johnson Adams, and Matei Ciocarlie. Sampling-based Exploration for Reinforcement Learning of Dexterous Manipulation. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.020.

[26] Binghao Huang, Yuanpei Chen, Tianyu Wang, Yuzhe Qin, Yaodong Yang, Nikolay Atanasov, and Xiaolong Wang. Dynamic handover: Throw and catch with bimanual hands. *arXiv preprint arXiv:2309.05655*, 2023.

[27] Yuanpei Chen, Chen Wang, Li Fei-Fei, and C Karen Liu. Sequential dexterity: Chaining dexterous policies for long-horizon manipulation. *arXiv preprint arXiv:2309.00987*, 2023.

[28] Johannes Pitz, Lennart Röstel, Leon Sievers, and Berthold Bäuml. Dextrous tactile in-hand manipulation using a modular reinforcement learning architecture. *arXiv preprint arXiv:2303.04705*, 2023.

[29] Kelvin Xu, Zheyuan Hu, Ria Doshi, Aaron Rovinsky, Vikash Kumar, Abhishek Gupta, and Sergey Levine. Dexterous manipulation from images: Autonomous real-world rl via substep guidance. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5938–5945. IEEE, 2023.

[30] Toru Lin, Zhao-Heng Yin, Haozhi Qi, Pieter Abbeel, and Jitendra Malik. Twisting lids off with two hands. *arXiv:2403.02338*, 2024.

[31] Sridhar Pandian Arunachalam, Sneha Silwal, Ben Evans, and Lerrel Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. In *2023 ieee international conference on robotics and automation (icra)*, pages 5954–5961. IEEE, 2023.

[32] Sridhar Pandian Arunachalam, Irmak Güzey, Soumith Chintala, and Lerrel Pinto. Holo-dex: Teaching dexterity with immersive mixed reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5962–5969. IEEE, 2023.

[33] Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *European Conference on Computer Vision*, pages 570–587. Springer, 2022.

[34] Priyanka Mandikal and Kristen Grauman. Dexvip: Learning dexterous grasping with human hand pose priors from video. In *Conference on Robot Learning*, pages 651–661. PMLR, 2022.

[35] Yuzhe Qin, Hao Su, and Xiaolong Wang. From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation. *IEEE Robotics and Automation Letters*, 7(4):10873–10881, 2022.

[36] Zoey Qiuyu Chen, Karl Van Wyk, Yu-Wei Chao, Wei Yang, Arsalan Mousavian, Abhishek Gupta, and Dieter Fox. Dextransfer: Real world multi-fingered dexterous grasping with minimal human demonstrations. *arXiv preprint arXiv:2209.14284*, 2022.

[37] Christian Zimmermann, Duygu Ceylan, Jimei Yang, Bryan Russell, Max Argus, and Thomas Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 813–822, 2019.

[38] Gyeongsik Moon, Shoou-I Yu, He Wen, Takaaki Shiratori, and Kyoung Mu Lee. Interhand2. 6m: A dataset and baseline for 3d interacting hand pose estimation from a single rgb image. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 548–564. Springer, 2020.

[39] Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3d with transformers. *arXiv preprint arXiv:2312.05251*, 2023.

[40] Tanner Schmidt, Richard A Newcombe, and Dieter Fox. Dart: Dense articulated real-time tracking. In *Robotics: Science and systems*, volume 2, pages 1–9. Berkeley, CA, 2014.

[41] Shreyas Hampali, Mahdi Rad, Markus Oberweger, and Vincent Lepetit. Honnotate: A method for 3d annotation of hand and object poses. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3196–3206, 2020.

[42] Yu-Wei Chao, Wei Yang, Yu Xiang, Pavlo Molchanov, Ankur Handa, Jonathan Tremblay, Yashraj S Narang, Karl Van Wyk, Umar Iqbal, Stan Birchfield, et al. Dexycb: A benchmark for capturing hand grasping of objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9044–9053, 2021.

[43] Shangchen Han, Beibei Liu, Randi Cabezas, Christo-

pher D Twigg, Peizhao Zhang, Jeff Petkau, Tsz-Ho Yu, Chun-Jung Tai, Muzaffer Akbay, Zheng Wang, et al. Megatrack: monochrome egocentric articulated hand-tracking for virtual reality. *ACM Transactions on Graphics (ToG)*, 39(4):87–1, 2020.

[44] Omid Taheri, Nima Ghorbani, Michael J Black, and Dimitrios Tzionas. Grab: A dataset of whole-body human grasping of objects. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 581–600. Springer, 2020.

[45] Zicong Fan, Omid Taheri, Dimitrios Tzionas, Muhammed Kocabas, Manuel Kaufmann, Michael J Black, and Otmar Hilliges. Arctic: A dataset for dexterous bimanual hand-object manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12943–12954, 2023.

[46] Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J Black, Otmar Hilliges, and Gerard Pons-Moll. Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018.

[47] Yifeng Jiang, Yuting Ye, Deepak Gopinath, Jungdam Won, Alexander W Winkler, and C Karen Liu. Transformer inertial poser: Real-time human motion reconstruction from sparse imus with simultaneous terrain generation. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022.

[48] Xinyu Yi, Yuxiao Zhou, Marc Habermann, Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Feng Xu. Physical inertial poser (pip): Physics-aware real-time human motion tracking from sparse inertial sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13167–13178, 2022.

[49] Tom Van Wouwe, Seunghwan Lee, Antoine Falisse, Scott Delp, and C Karen Liu. Diffusion inertial poser: Human motion reconstruction from arbitrary sparse imu configurations. *arXiv preprint arXiv:2308.16682*, 2023.

[50] Fabian C Weigend, Xiao Liu, and Heni Ben Amor. Probabilistic differentiable filters enable ubiquitous robot control with smartwatches. *arXiv preprint arXiv:2309.06606*, 2023.

[51] Lars Fritsche, Felix Unverzag, Jan Peters, and Roberto Calandra. First-person tele-operation of a humanoid robot. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 997–1002. IEEE, 2015.

[52] Bin Fang, Di Guo, Fuchun Sun, Huaping Liu, and Yupei Wu. A robotic hand-arm teleoperation system using human arm/hand with a novel data glove. In *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 2483–2488. IEEE, 2015.

[53] Chengxu Zhou, Christopher Peers, Yuhui Wan, Robert Richardson, and Dimitrios Kanoulas. Teleman: Teleoperation for legged robot loco-manipulation using wearable imu-based motion capture. *arXiv preprint arXiv:2209.10314*, 2022.

[54] Sylvain Calinon, Florent D'halluin, Eric L Sauser, Darwin G Caldwell, and Aude G Billard. Learning and reproduction of gestures by imitation. *IEEE Robotics & Automation Magazine*, 17(2):44–54, 2010.

[55] A.J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 2, pages 1398–1403 vol.2, 2002. doi: 10.1109/ROBOT.2002.1014739.

[56] Jens Kober and Jan Peters. Imitation and reinforcement learning. *IEEE Robotics & Automation Magazine*, 17 (2):55–62, 2010.

[57] Peter Englert and Marc Toussaint. Learning manipulation skills from a single demonstration. *The International Journal of Robotics Research*, 37(1):137–154, 2018.

[58] Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, pages 357–368. PMLR, 2017.

[59] Aude Billard, Sylvain Calinon, Ruediger Dillmann, and Stefan Schaal. Robot programming by demonstration. In *Springer handbook of robotics*, pages 1371–1394. Springer, 2008.

[60] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57 (5):469–483, 2009.

[61] Stefan Schaal. Dynamic movement primitives-a framework for motor control in humans and humanoid robotics. In *Adaptive motion of animals and machines*, pages 261–280. Springer, 2006.

[62] Jens Kober and Jan Peters. Learning motor primitives for robotics. In *2009 IEEE International Conference on Robotics and Automation*, pages 2112–2118. IEEE, 2009.

[63] Alexandros Paraschos, Christian Daniel, Jan R Peters, and Gerhard Neumann. Probabilistic movement primitives. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper/2013/file/e53a0a2978c28872a4505bdb51db06dc-Paper.pdf.

[64] Alexandros Paraschos, Christian Daniel, Jan Peters, and Gerhard Neumann. Using probabilistic movement primitives in robotics. *Autonomous Robots*, 42(3):529–551, 2018.

[65] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstra-

tions for robot manipulation. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=JrsfBJtDFdI.

[66] Peter Florence, Lucas Manuelli, and Russ Tedrake. Self-supervised correspondence in visuomotor policy learning. *IEEE Robotics and Automation Letters*, 5(2): 492–499, 2019.

[67] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[68] Jennifer Grannen, Yilin Wu, Brandon Vu, and Dorsa Sadigh. Stabilize to act: Learning to coordinate for bimanual manipulation. In *Conference on Robot Learning*, pages 563–576. PMLR, 2023.

[69] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 5628–5635. IEEE, 2018.

[70] Ajay Mandlekar, Jonathan Booher, Max Spero, Albert Tung, Anchit Gupta, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1048–1055. IEEE, 2019.

[71] Liyiming Ke, Ajinkya Kamat, Jingqiang Wang, Tapomayukh Bhattacharjee, Christoforos Mavrogiannis, and Siddhartha S Srinivasa. Telemanipulation with chopsticks: Analyzing human factors in user demonstrations. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11539–11546. IEEE, 2020.

[72] Chen Wang, Rui Wang, Ajay Mandlekar, Li Fei-Fei, Silvio Savarese, and Danfei Xu. Generalization through hand-eye coordination: An action space for learning spatially-invariant visuomotor control. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8913–8920. IEEE, 2021.

[73] Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. Viola: Imitation learning for vision-based manipulation with object proposal priors. *arXiv preprint arXiv:2210.11339*, 2022.

[74] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[75] Philipp Wu, Yide Shentu, Zhongke Yi, Xingyu Lin, and Pieter Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. *arXiv preprint arXiv:2309.13037*, 2023.

[76] Jensen Gao, Annie Xie, Ted Xiao, Chelsea Finn, and Dorsa Sadigh. Efficient data collection for robotic manipulation via compositional generalization. *arXiv preprint arXiv:2403.05110*, 2024.

[77] Toru Lin, Yu Zhang, Qiyang Li, Haozhi Qi, Brent Yi, Sergey Levine, and Jitendra Malik. Learning visuotactile skills with two multifingered hands. *arXiv:2404.16823*, 2024.

[78] Jiafei Duan, Yi Ru Wang, Mohit Shridhar, Dieter Fox, and Ranjay Krishna. Ar2-d2: Training a robot without a robot. *arXiv preprint arXiv:2306.13818*, 2023.

[79] Mengda Xu, Zhenjia Xu, Cheng Chi, Manuela Veloso, and Shuran Song. Xskill: Cross embodiment skill discovery. In *Conference on Robot Learning*, pages 3536–3555. PMLR, 2023.

[80] Jingyun Yang, Junwu Zhang, Connor Settle, Akshara Rai, Rika Antonova, and Jeannette Bohg. Learning periodic tasks from human demonstrations. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8658–8665. IEEE, 2022.

[81] Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. *arXiv preprint arXiv:2311.01977*, 2023.

[82] Haoyu Xiong, Haoyuan Fu, Jieyi Zhang, Chen Bao, Qiang Zhang, Yongxi Huang, Wenqiang Xu, Animesh Garg, and Cewu Lu. Robotube: Learning household manipulation from human videos with simulated twin environments. In *Conference on Robot Learning*, pages 1–10. PMLR, 2023.

[83] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European conference on computer vision (ECCV)*, pages 720–736, 2018.

[84] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[85] Huihan Liu, Soroush Nasiriany, Lance Zhang, Zhiyao Bao, and Yuke Zhu. Robot learning on the job: Human-in-the-loop autonomy and learning during deployment. *arXiv preprint arXiv:2211.08416*, 2022.

[86] Zhenghao Peng, Wenjie Mo, Chenda Duan, Quanyi Li, and Bolei Zhou. Learning from active human involvement through proxy value propagation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[87] Jonathan Spencer, Sanjiban Choudhury, Matthew

Barnes, Matthew Schmittle, Mung Chiang, Peter Ramadge, and Siddhartha Srinivasa. Learning from interventions. In *Robotics: Science and Systems (RSS)*, 2020.

[88] Shuran Song, Andy Zeng, Johnny Lee, and Thomas Funkhouser. Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. *IEEE Robotics and Automation Letters*, 5(3):4978–4985, 2020.

[89] Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. Visual imitation made easy. In *Conference on Robot Learning*, pages 1992–2005. PMLR, 2021.

[90] Kiran Doshi, Yijiang Huang, and Stelian Coros. On hand-held grippers and the morphological gap in human manipulation demonstration. *arXiv preprint arXiv:2311.01832*, 2023.

[91] Felipe Sanches, Geng Gao, Nathan Elangovan, Ricardo V Godoy, Jayden Chapman, Ke Wang, Patrick Jarvis, and Minas Liarokapis. Scalable. intuitive human to robot skill transfer with wearable human machine interfaces: On complex, dexterous tasks. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6318–6325. IEEE, 2023.

[92] Hongjie Fang, Hao-Shu Fang, Yiming Wang, Jieji Ren, Jingjing Chen, Ruo Zhang, Weiming Wang, and Cewu Lu. Low-cost exoskeletons for learning whole-arm manipulation in the wild. *arXiv preprint arXiv:2309.14975*, 2023.

[93] Nur Muhammad Mahi Shafiullah, Anant Rai, Haritheja Etukuru, Yiqian Liu, Ishan Misra, Soumith Chintala, and Lerrel Pinto. On bringing robots home. *arXiv preprint arXiv:2311.16098*, 2023.

[94] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

[95] Kenneth Shaw, Ananye Agarwal, and Deepak Pathak. LEAP Hand: Low-Cost, Efficient, and Anthropomorphic Hand for Robot Learning. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.089.

[96] Daniel Rakita, Bilge Mutlu, and Michael Gleicher. RelaxedIK: Real-time Synthesis of Accurate and Feasible Robot Arm Motion. In *Proceedings of Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018. doi: 10.15607/RSS.2018.XIV.043.

[97] Daniel Rakita, Haochen Shi, Bilge Mutlu, and Michael Gleicher. Collisionik: A per-instant pose optimization method for generating robot motions with environment collision avoidance. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9995–10001. IEEE, 2021.

[98] Yeping Wang, Pragathi Praveena, Daniel Rakita, and Michael Gleicher. Rangedik: An optimization-based robot motion generation method for ranged-goal tasks. pages 9700–9706, 2023.

[99] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[100] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[101] Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Human-in-the-loop imitation learning using remote teleoperation. *arXiv preprint arXiv:2012.06733*, 2020.

[102] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[103] Yuzhe Qin, Binghao Huang, Zhao-Heng Yin, Hao Su, and Xiaolong Wang. Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation. In *Conference on Robot Learning*, pages 594–605. PMLR, 2023.

[104] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *arXiv preprint arXiv: Arxiv-1612.00593*, 2016.

[105] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver: General perception with iterative attention. *arXiv preprint arXiv: Arxiv-2103.03206*, 2021.

[106] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. *arXiv preprint arXiv: Arxiv-1810.00825*, 2018.

[107] Yiyue Luo, Yunzhu Li, Pratyusha Sharma, Wan Shou, Kui Wu, Michael Foshey, Beichen Li, Tomás Palacios, Antonio Torralba, and Wojciech Matusik. Learning human–environment interactions using conformal tactile textiles. *Nature Electronics*, 4(3):193–201, 2021.

[108] Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[109] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[110] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

APPENDIX A
IMPLEMENTATION DETAILS

A. DEXCAP *hardware implementations*

Figure 11 illustrates the hardware design of DEXCAP. All models are 3D-printed with PLA material. The chest camera mount is equipped with four slots for cameras: at the top, an L515 RGB-D LiDAR camera, followed by three T265 fisheye SLAM tracking cameras. The LiDAR camera and the uppermost T265 camera are securely fixed to the camera rack, while the two lower T265 cameras are designed to be detachable and can be affixed to the glove's back for hand 6-DoF pose tracking. The design features of the camera mounts on both the chest and gloves include a locking mechanism to prevent the cameras from accidentally slipping out. On the glove, the camera mount is positioned over the magnetic hub on its dorsal side, ensuring a firm attachment between the hub and the mount. For powering and data storage, the user wears a backpack containing a 40000mAh portable power bank and a mini-PC with 64GB RAM and 2TB SSD. The system's total weight is 3.96 pounds, optimized for ease of mobility, supporting up to 40 minutes of continuous data collection. The power bank's rapid recharge capability, requiring only 30 minutes for a full charge, enables extensive data collection sessions over several hours.

B. *Data collection details*

Figure 13 and the supplementary video illustrate the beginning steps of a data collection session. Initially, all cameras are mounted on the chest. Upon initiating the program, the participant moves within the environment for several seconds, allowing the SLAM algorithm to build the map of the surroundings. Subsequently, the bottom T265 cameras are relocated to the glove mounts, initiating the data collection phase. This preparatory phase is completed in approximately 15 seconds, as demonstrated in the video submission.

The data collection encompasses four data types, recorded at 60 frames per second: (1) the 6-DoF pose of the chest-mounted LiDAR camera, as tracked by the top T265 camera; (2) the 6-DoF wrist poses, as captured by the two lower T265 cameras attached to the gloves; (3) the positions of finger joints within each glove's reference frame, detected by the motion capture gloves; and (4) RGB-D image frames from the LiDAR camera. The initial pose of the top T265 camera establishes the world frame for all data, allowing for the integration of all streamed data—RGB-D point clouds, hand 6-DoF poses, and finger joint locations—into a unified world frame. This configuration permits unrestricted movement by the participant, enabling easy isolation and removal of body movements from the dataset.

Data are initially buffered in the mini-PC's RAM, supporting a 15-minute collection at peak frame rate (60 fps). Once the RAM is full, data capture slows to 20 fps due to storage shifting to the SSD. We empirically find that this reduction in frame rate may affect SLAM tracking accuracy, potentially leading to jumping tracking results. Thus, we use the first 10 minutes of each session prioritized for high-quality data capture. After collection, transferring the data from RAM to SSD is efficiently completed within 3-5 minutes using multi-threading.

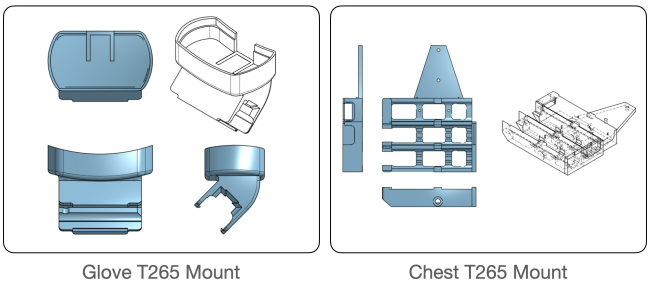


Fig. 11: **Detailed view of chest mount and glove mount** The glove mount follows the contour of the hump on the top of the Rokoko glove, and an opening is added to route the USB-C cable to the glove. The angle of the camera is set to 45 degrees facing upwards so that the camera view is less obstructed from the back of the hand. The slide guide has an indentation matching the position of the back plate to ensure the same insertion position across experiments. The chest mount houses 3 identical slots following the contour of the T265. An additional slot is added to fit in the slide plate of the T265.

In this study, we primarily investigate two types of DEXCAP data: (1) data captured in the robot space and (2) data collected in the wild. For the first category, we position the chest camera setup on a stand between two robot arms. The robots are then adjusted to a resting position, clearing the operational space for human interaction. This arrangement allows for the direct use of DEXCAP to collect data within the robot's operational area. Such data underpins basic experiments for tasks like *Sponge picking*, *Ball collecting*, and *Plate wiping*, alongside more complex challenges, including *Scissor cutting* and *Tea preparing*. For the second category, individuals don DEXCAP to gather data outside the lab setting, focusing on the system's zero-shot learning performance with *in-the-wild* DEXCAP data and its ability to generalize to unseen objects, particularly in the *Packaging* task.

C. *Data retargeting details*

To adapt the collected raw DEXCAP data for training robot policies (commonly known as retargeting). This involves two key steps: (1) retargeting the observations and (2) retargeting the actions.

For observation retargeting, the initial step is to convert the RGB-D inputs into 3D point clouds, ensuring each pixel's color is preserved. These point clouds are then aligned with the world frame, defined by the initial pose of the main T265 camera. Subsequently, a point cloud visualization UI is launched, displaying the aligned input point clouds alongside the robot operation space's point clouds within a unified coordinate frame. Through this UI, users can adjust the point cloud's position within the robot operation space using the keyboard's

directional keys. This adjustment process is required only once for all data collected in the same location and is completed in under a minute. After aligning the point clouds with the robot space, points below the robot's table surface are eliminated, refining the observation data for policy development.

Action retargeting begins with applying a consistent transformation between the T265 cameras on the chest mount to translate the hand joint locations into the world frame. Then, we use the previously calculated point cloud transformation matrix to transform the hand joints to the robot operation space. The results of this process are visualized in Figure 12 by depicting the transformed hand joints together with the point cloud as a skeletal model of the hand. The final phase employs inverse kinematics to map the fingertip positions between the robot hand (LEAP hand) and the human hand. We use the hand's 6-DoF pose to initialize the LEAP hand's orientation for IK calculation. Figure 12 illustrates the IK results, showing the robot hand model integrated with the observational point clouds, thereby generating the actions required for training the robot policy.

All of the point cloud observations are downsampled uniformly to 5000 points and stored together with robot proprioception states and actions into an hdf5 file. We manually annotate the start and end frames of each task demonstration from the entire recording session (10 minutes each). The motion for resetting the task environment is not included in the training dataset.

D. Robot controller details

Position control is employed throughout our experiments, structured hierarchically: (1) At the high level, the learned policy generates the goal position for the next step, which encompasses the 6-DoF pose of the end-effector for both robot arms and a 16-dimensional finger joint position for both hands. (2) At the low level, an Operational Space Controller (OSC) [108], continuously interpolates the arm's trajectory towards the high-level specified goal position and relays interpolated OSC actions to the robot for execution. Meanwhile, finger movements are directly managed by a joint impedance controller. Following each robot action, we calculate the distance between the robot's current proprioception and the target pose. If the distance between them is smaller than a threshold, we regard that the robot has reached the goal position and will query the policy for the next action. To prevent the robot from becoming idle, if it fails to reach the goal pose within h steps, the policy is queried anew for the subsequent action. We designate $h = 10$ in our experiments. We empirically find that for tasks that consist of physical contact with objects or applying force, this situation happens more often and a smaller h will have a smoother robot motion.

E. Policy model and training details

For all image-input methods, we use ResNet-18 [109] as the image encoder. For models based on diffusion policy, we use Denoising Diffusion Implicit Models (DDIM) [110] for the denoising iterations. For all baselines, the time horizon of the inputs is set to three. For pointcloud-based methods, the input point cloud is uniformly downsampled to 1000 points. We list the hyperparameters for each architecture in Table V, VI, VII.

Hyperparameter	Default
Batch Size	16
Learning Rate (LR)	1e-4
Num Epoch	3000
LR Decay	None
Image Encoder	ResNet-18
Image Feature Dim	64
RNN Type	LSTM
RNN Horizon	3
GMM	None

TABLE V: Hyperparameters - BC-RNN-img

Hyperparameter	Default
Batch Size	16
Learning Rate (LR)	1e-4
Num Epoch	3000
LR Decay	None
Point Cloud Encoder	PointNet
Point Cloud Downsample	1000
Pooling Type	MaxPooling
UNet Embed Dim	256
UNet Down dims	[256, 512, 1024]
UNet Kernel Size	5
Diffusion Type	DDIM
Diffusion Num Train	100
Diffusion Num Infer	10
Input Horizon	3

TABLE VI: Hyperparameters - DP-point

F. Task implementations

In this section, we introduce the details of each task design

- *Sponge Picking*: A sponge is randomly placed on the table within a 40×70 centimeter area. The objective is to grasp the sponge and lift it upwards by more than 30 centimeters.
- *Ball Collecting*: A ball is randomly positioned on the right side of the table within a 40×30 centimeter area, while a basket is similarly placed randomly on the left side within the same dimensions. The task is completed when the ball is grasped and then dropped into the basket.
- *Plate Wiping*: In a setup akin to the *Ball Collecting* task, a plate and a sponge are randomly placed on the right and left sides of the table, respectively, each within a 40×30 centimeter area. The goal involves using both hands to pick up the plate and sponge separately, then utilizing the sponge to wipe the plate twice. This task demands coordination between the two hands, positioning the plate in the table's middle area to facilitate the wiping action.
- *Packaging*: An empty paper box and a target object are randomly positioned on the table, with the object within a 40×30 centimeter area on the right and the box within a 10×10 centimeter area on the left. This task aims to assess the model's ability to generalize across various objects, including unseen ones not present in the training dataset. Success involves using one hand to pick up the

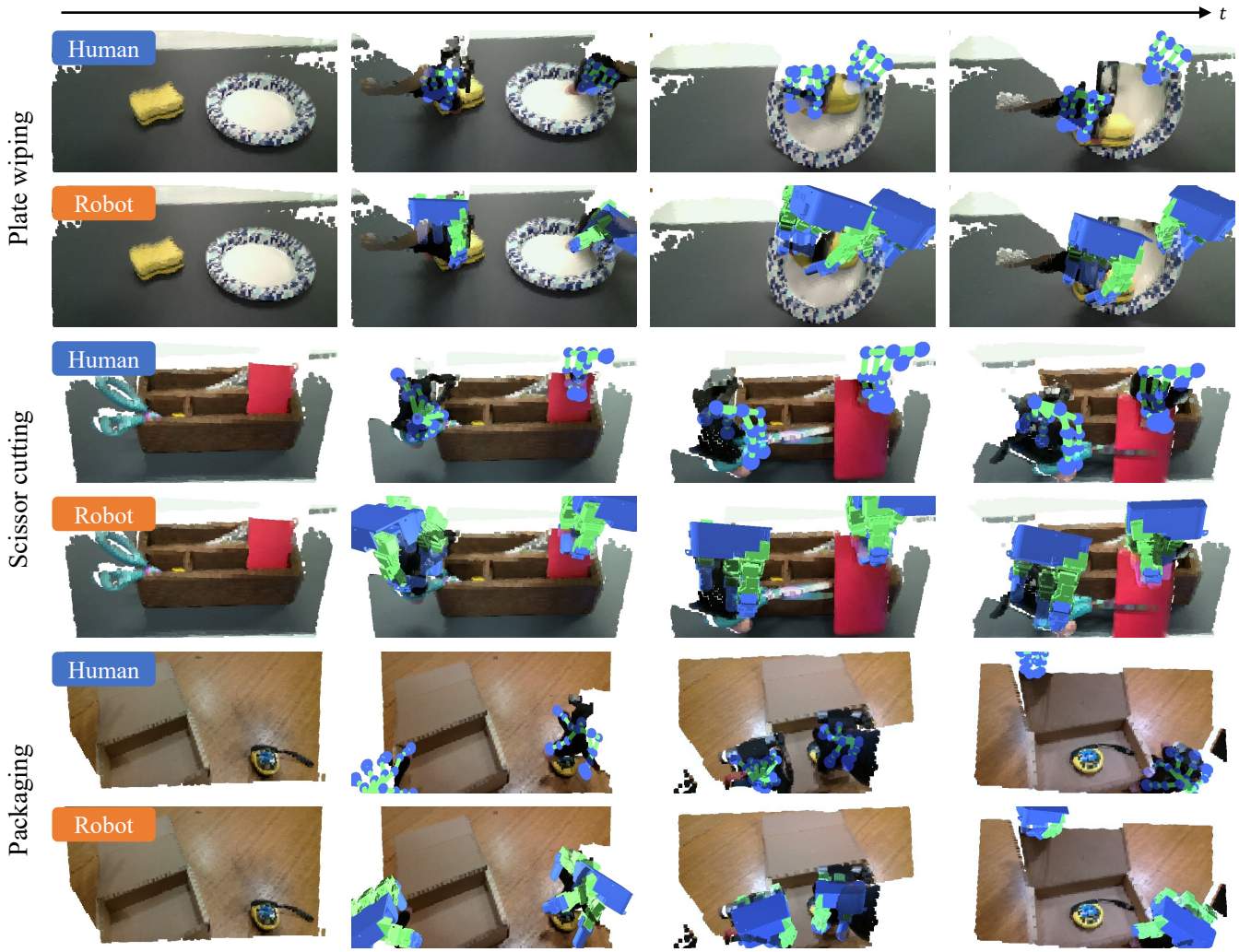


Fig. 12: **Visualization of collected human data and retargeted robot data.** DEXIL successfully adapts human motion capture data for tasks such as plate wiping, scissor cutting, and packaging. We demonstrate the entire workflow of executing these tasks.

Hyperparameter	Default
Batch Size	16
Learning Rate (LR)	1e-4
Num Epoch	3000
LR Decay	None
Point Cloud Encoder	Perceiver
Point Cloud Downsample	1000
Pooling Type	MaxPooling
UNet Embed Dim	256
UNet Down dims	[256, 512, 1024]
UNet Kernel Size	5
Diffusion Type	DDIM
Diffusion Num Train	100
Diffusion Num Infer	10
Input Horizon	3

TABLE VII: Hyperparameters - Ours (DP-prec)

object and the other to move the box to the table's center. The object is then placed into the box, followed by stabilizing the box with one hand while the other closes it by grasping and moving the lid.

- *Scissor Cutting*: A container is fixed at the table's center, with scissors on the left and a strip of paper tape on the right. The task begins with the left hand functionally grasping the scissors—inserting the thumb into one handle and the index and middle fingers into the other. Simultaneously, the right hand grasps the paper tape. Both scissors and tape are then lifted and moved towards the center, with the left hand operating the scissors to cut the tape. A cut exceeding 3 millimeters deems the task successful.
- *Tea Preparing*: A tea table is centrally placed with a fixed orientation, accompanied by a tea bottle, tweezers, and a teapot. The robot must first grasp the tea bottle with the left hand and unscrew the cap with the right hand, completing two rotations. The cap is then taken off and placed on the right side of the tea table. Subsequently, the right hand picks up the tweezers from the top right corner of the tea table. The robot then attempts to pour tea from

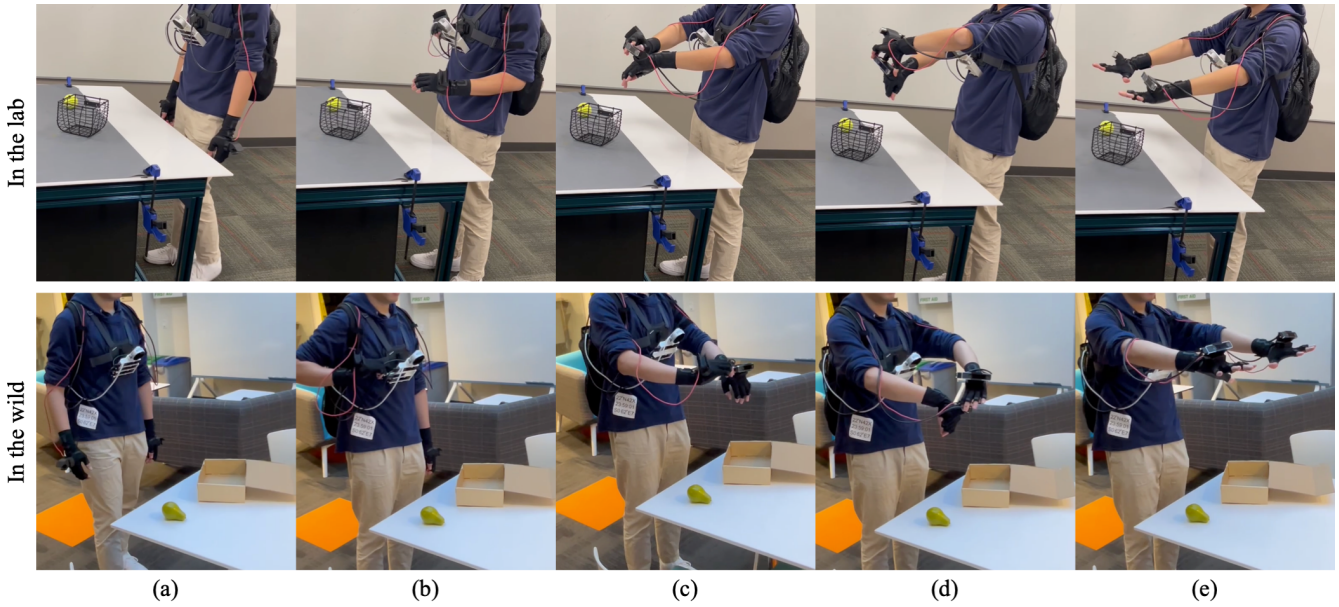


Fig. 13: **Prepration of data collection in the wild.** The first row illustrates data collection conducted in a laboratory setting, and the second row depicts in-the-wild data collection. (a) Initially, the human data collector moves around in the environment to track 6-DoF wrist poses with SLAM. (b)-(d) Subsequently, the data collector detaches the two cameras from the chest mount and secures them onto the glove mount. (e) With this setup, the human is prepared to begin data collection.

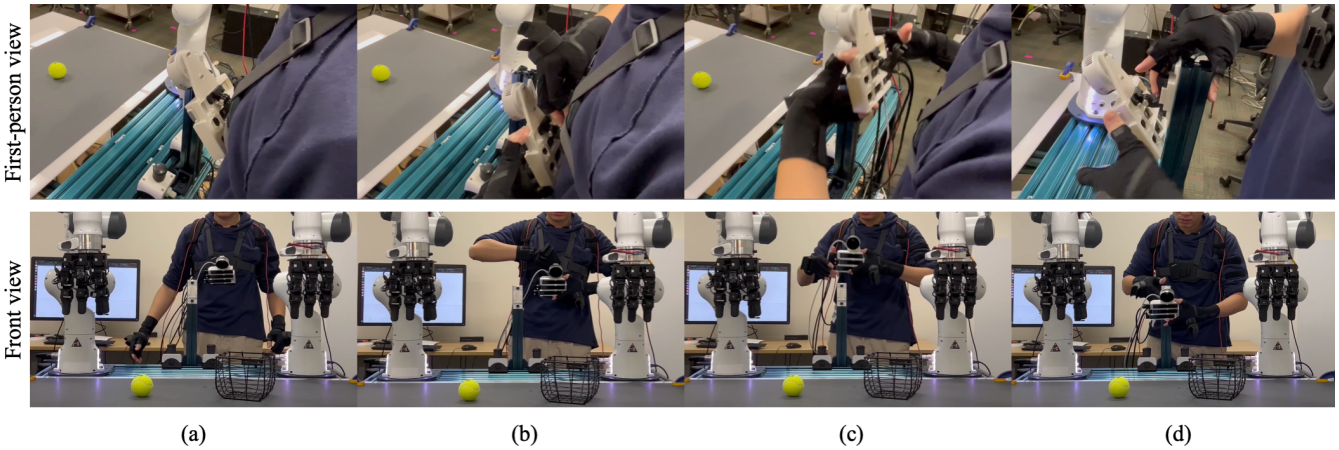


Fig. 14: **Switching DexCap from the human to the robot.** We illustrate, from both first-person and front views, the seamless transition of DexCap from a human data collector to a bimanual dexterous robot system. This process involves effortlessly detaching the cameras from the chest mount and inserting them into a stationary mount on the robot's table.

the bottle into the teapot with the left hand, while the right hand uses the tweezers to aid the pouring process. Finally, the robot returns the tweezers and the tea bottle to their corresponding positions on the table. The task is deemed successful if tea makes it into the teapot and both the tea bottle and tweezers are returned to their respective places. For the task to be considered fully successful, the tea bottle must be completely released from the left hand.

G. Human-in-the-loop implementations

DexCap incorporates two human-in-the-loop correction methodologies: teleoperation and residual correction. Both methods can be utilized during policy rollouts to gather additional correction data, which is used in further refining the policy for enhanced task performance. Detailed descriptions of these algorithms and their implementation are provided in the main paper. In the human-in-the-loop process, we employ the mini-PC to live stream data from all T265 tracking cameras. This tracking information is then transmitted to a Redis server configured on the local network. Concurrently, the robot, operating the learned policy on a workstation, receives delta movements of the human hands from the Redis server. These deltas serve as residual corrections and are integrated into each robot action. The RGB-D LiDAR camera, positioned on the central bar between the robot arms, connects to the workstation

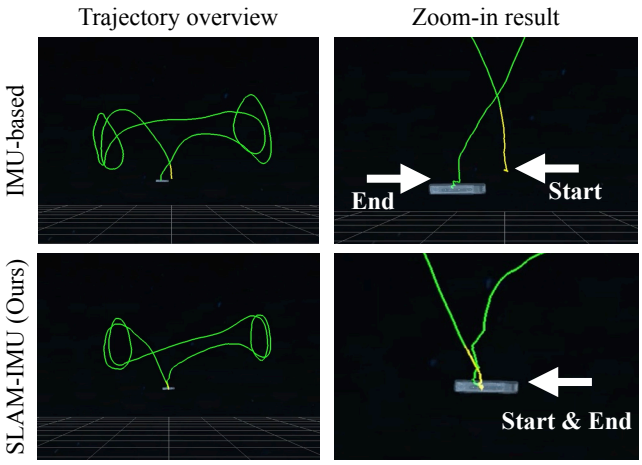


Fig. 15: **Compare with IMU-based mocap system.** We disable the SLAM mapping and pose-correction features of the T265 tracking camera, forcing it to rely on IMU information to track the pose. The human operator held the camera, started from a fixed location, moved it along a predefined trajectory, and then returned to the starting position. IMU-based method (first row) fails to match the endpoint with the start point, which indicates that there is pose drift during tracking. Our SLAM-IMU method (second row) doesn't drift and captures smooth trajectory during the tracking.

Drifting error (cm)	Trajectory 1	Trajectory 2
IMU-based	8.0 ± 3.1	11.3 ± 4.7
SLAM-IMU (Ours)	0.4 ± 0.2	0.8 ± 0.3

TABLE VIII: Drifting error of different tracking methods.

to capture observation data. Instead of recording the robot's actual positional changes, we log the action commands dispatched to the robot controller. This design is crucial for tasks involving physical contact with the environment and objects.

APPENDIX B
SUPPLEMENTARY EXPERIMENT RESULTS

A. *Tracking accuracy*

Figure 15 and Table VIII present qualitative and quantitative results, respectively. We observe that the IMU-based method suffers from pose drifting during tracking, while our SLAM-IMU approach more accurately tracks hand poses, with an average error of 0.8 cm compared to the 11.3 cm error of the IMU-based method.