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.

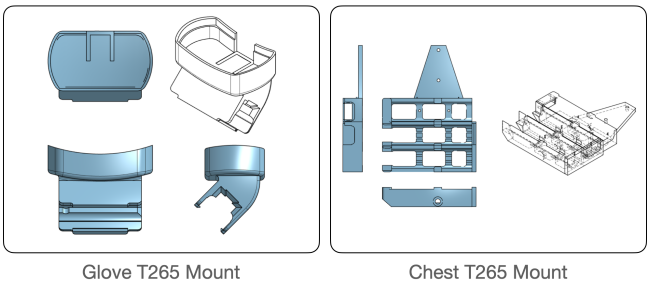

Fig. 11: **Detailed view of chest mount and glove mount** The glove mount follows the contour of the hump on the top of the Rokoko glove, and an opening is added to route the USB-C cable to the glove. The angle of the camera is set to 45 degrees facing upwards so that the camera view is less obstructed 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) [46], 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 [36] as the image encoder. For models based on diffusion policy, we use Denoising Diffusion Implicit Models (DDIM) [90] 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 object and the other to move the box to the table's center.

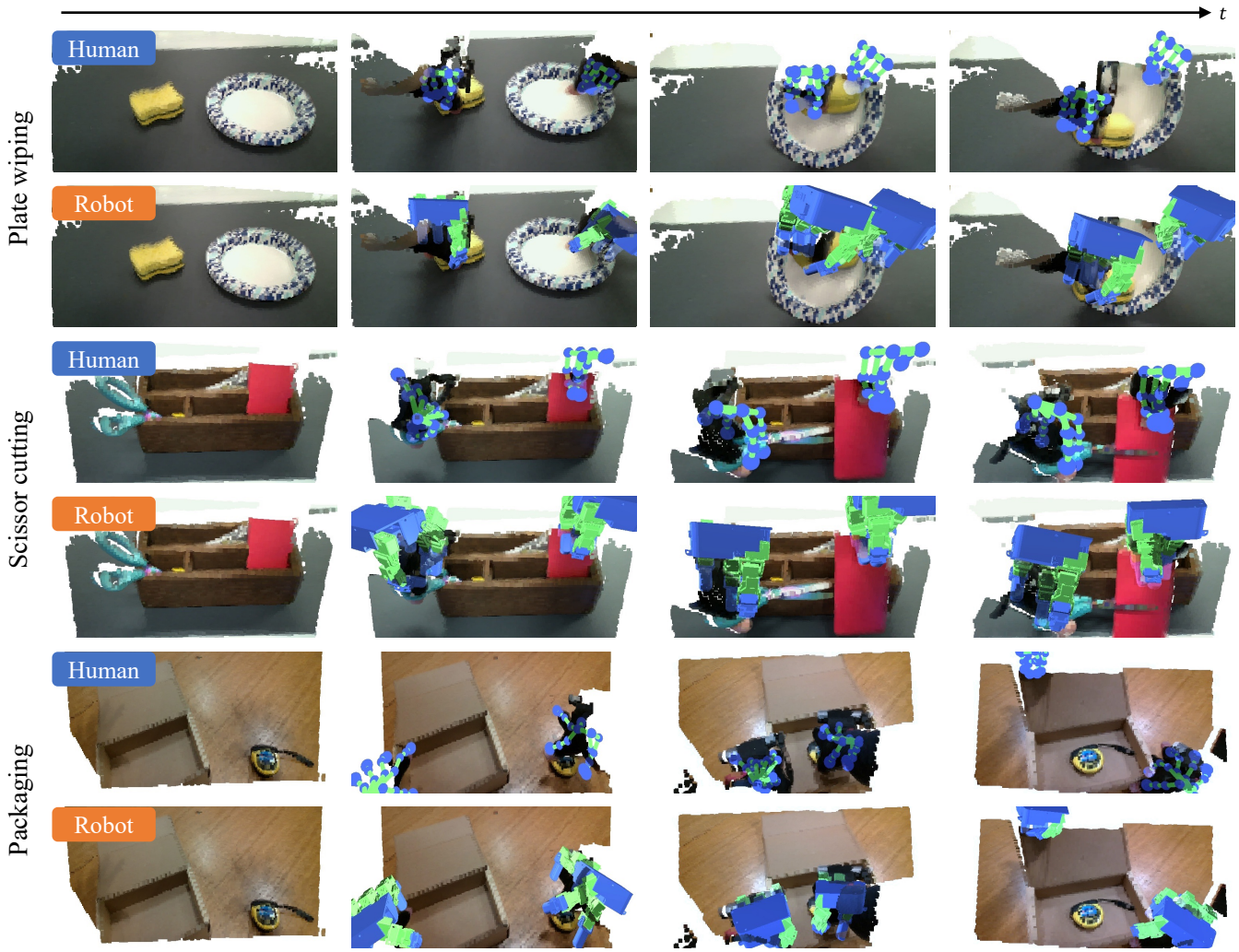

Fig. 12: **Visualization of collected human data and retargeted robot data.** DEXIL successfully adapts human motion capture data for tasks such as plate wiping, scissor cutting, and packaging. We demonstrate the entire workflow of executing these tasks.

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)

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,

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

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

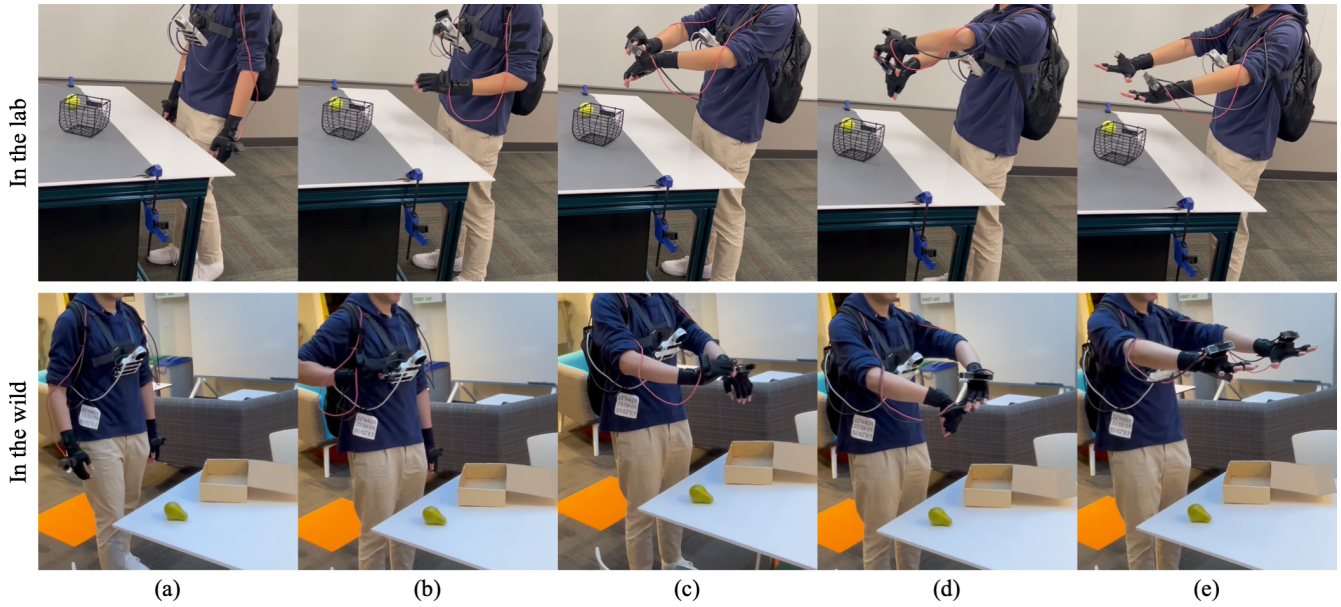

Fig. 13: **Prepration of data collection in the wild.** The first row illustrates data collection conducted in a laboratory setting, and the second row depicts in-the-wild data collection. (a) Initially, the human data collector moves around in the environment to track 6-DoF wrist poses with SLAM. (b)-(d) 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.

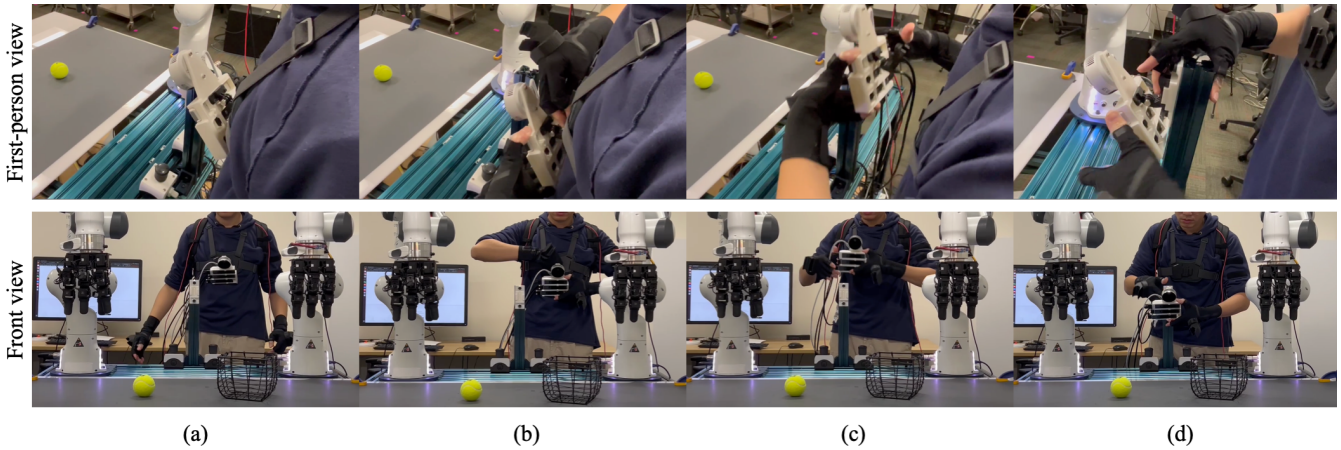

Fig. 14: **Switching DexCap from the human to the robot.** We illustrate, from both first-person and front views, the seamless transition of DexCap from a human data collector to a bimanual dexterous robot system. This process involves effortlessly detaching the cameras from the chest mount and inserting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

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 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 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.