CORE4D: A 4D Human-Object-Human Interaction Dataset for Collaborative Object REarrangement — Supplementary Material

Yun Liu[1,2,3*] Chengwen Zhang[4*] Ruofan Xing[5] Bingda Tang[1] Bowen Yang[1] Li Yi[1,2,3†]
[1]Tsinghua University [2]Shanghai Artificial Intelligence Laboratory [3]Shanghai Qi Zhi Institute
[4]Beijing University of Posts and Telecommunications [5]Northeastern University
https://core4d.github.io/

Contents:

A. Cross-dataset Evaluation

To examine the data quality of CORE4D-Real, we follow existing dataset efforts [2, 10, 25] and conduct the vision-based cross-dataset evaluation. We select an individual human-object-interaction dataset BEHAVE [1] that includes color images and select 2D human keypoint estimation as the evaluation task.

Data Preparation. For a color image from CORE4D-Real and BEHAVE [1], we first detect the bounding box for each person via ground truth human pose and obtain the image patch for the person. We then resize the image patch to get a maximal length of 256 pixels and fill it up into a 256x256 image with the black color as the background. Finally, for each 256x256 image, we automatically acquire the ground truth 2D-pixel coordinates of 22 SMPL-X [15] human body joints from 3D human poses. For data split, we follow the original train-test split for BEHAVE [1] and merge the two test sets (S1, S2) for CORE4D-Real.

Task Formulation. Given a 256x256 color image including a person, the task is to estimate the 2D-pixel coordinate for each of the 22 SMPL-X [15] human body joints.

Evaluation Metrics. P_e denotes the mean-square error of 2D coordinate estimates. Acc denotes the percentage of the coordinate estimates with the Euclidean distance to the ground truth smaller than 15 pixels.

Method, Results, and Analysis. We draw inspiration from HybrIK-X [8] and adopt their vision backbone as the solution. Table 1 shows the method performances on the two datasets under different training settings. Due to the significant domain gaps in visual patterns and human behaviors, transferring models trained on one dataset to the other would consistently encounter error increases. Despite the domain gaps, integrally training on both datasets achieves large performance gains on both CORE4D-Real and BEHAVE [1], indicating the accuracy of CORE4D-Real and the value of the dataset serving for visual perception studies.

Test \ Train	CORE4D-Real	BEHAVE [1]	CORE4D-Real +BEHAVE [1]
CORE4D-Real	152.4 / 91.2	904.9 / 35.6	**121.7 / 92.4**
BEHAVE[1]	887.9 / 37.8	146.3 / 88.9	**128.2 / 89.8**

Table 1. **Cross-dataset evaluation with BEHAVE [1] on 2D human keypoint estimation.** Results are in P_e (pixel^2, lower is better) and Acc (%, higher is better), respectively.

B. Details on Real-world Data Aquisition

In this section, we describe our system calibration (Section B.1) and time synchronization (Section B.2) in detail. Moreover, we provide detailed information on loss functions of the human mesh acquisition (Section B.3).

B.1. System Calibration

Calibrating the Inertial-optical Mocap System. Three reflective markers are fixed at known positions on a calibration rod, by which the 12 high-speed motion capture cameras calculate their relative extrinsic parameters, providing information about their spatial relationships. Additionally, three markers fixed at the world coordinate origin are employed to calibrate the motion capture system coordinate

*Equal contribution.
†Corresponding author.

with the defined world coordinate.

Calibrating Camera Intrinsic. The intrinsic parameters of allocentric and egocentric cameras are calibrated using a chessboard pattern.

Calibrating Extrinsic of the Allocentric Cameras. We place ten markers in the camera view to locate each allocentric camera. By annotating the markers' 3D positions in the world coordinate system and their 2D-pixel coordinates on allocentric images, the camera's extrinsic parameters are estimated by solving a Perspective-n-Point (PnP) problem via OpenCV.

Calibrating Extrinsic of the Egocentric Camera. We obtain the camera's pose information by fixing the camera to the head tracker of the motion capture suit. Similarly, ten markers are used to calibrate the relative extrinsic parameters of the first-person perspective cameras, allowing for determining their positions and orientations relative to the motion capture system. Additionally, to mitigate errors introduced by the integration of optical and inertial tracking systems, a purely optical tracking rigid is mounted on the motion camera.

B.2. Time Synchronization

To implement our synchronization method, we first set up a Network Time Protocol (NTP) server on the motion capture host. This server serves as the time synchronization reference for the Windows computer connected to the Kinect Azure DK. We minimize time discrepancies by connecting the Windows computer to the NTP server in high-precision mode and thus achieving precise synchronization.

Additionally, we employ a Linear Timecode (LTC) generator to encode a time signal onto the action camera's audio track. This time signal serves as a synchronization reference for aligning the first-person perspective RGB information with the motion capture data.

B.3. Loss Function Designs for Human Mesh Acquisition

To transfer the BVH [13] human skeleton to the widely-used SMPL-X [15] model. We first estimate the body shape parameters $\beta \in \mathbb{R}^{10}$ that minimize $\mathcal{L}(\beta) = \lambda_{\text{bone}}\mathcal{L}_{\text{bone}} + \lambda_{\text{shape\_reg}}\mathcal{L}_{\text{shape\_reg}}$, where $\mathcal{L}_{\text{bone}} = \sum(\bar{B}_i - B_i(\beta))^2$ computes differences between the measured human bone lengths ($\bar{B}_i$) and those decided by β ($B_i(\beta)$). The utilized bones (i) involve the upper and lower arms, upper and lower legs, and the overall human height. The regularizer $\mathcal{L}_{\text{shape\_reg}} = \|\beta\|^2$ prevents large values in β.

Given β, we then optimize the full-body pose $\theta \in \mathbb{R}^{159}$ with the following loss function:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{reg}} + \mathcal{L}_{j3\text{D}} + \mathcal{L}_{j\text{Ori}} + \mathcal{L}_{\text{smooth}} + \mathcal{L}_{h3\text{D}} + \mathcal{L}_{h\text{Ori}} + \mathcal{L}_{\text{contact}} \tag{1}$$

Regularization Loss $\mathcal{L}_{\text{reg}}$. The regularization loss term is defined as

$$\mathcal{L}_{\text{reg}} = \sum ||\theta_{\text{body}}||^2 \cdot \lambda_{\text{body}} + \left(\sum ||\theta_{l\_\text{hand}}||^2 + \sum ||\theta_{r\_\text{hand}}||^2\right) \cdot \lambda_{\text{hand}}, \tag{2}$$

where $\theta_{\text{body}} \in \mathbb{R}^{21\times3}$ represents the body pose parameters defined by 21 joints of the skeleton, $\theta_{l\_hand} \in \mathbb{R}^{12}$ and $\theta_{r\_\text{hand}} \in \mathbb{R}^{12}$ represents the hand pose parameters. For each hand, the original SMPL-X skeleton has 15 joints with parameters $\theta_{\text{hand}} \in \mathbb{R}^{15\times3}$. However, principal component analysis (PCA) is applied to the hand pose parameters. The θ_{hand} parameters are transformed into a lower-dimensional space, specifically $\mathbb{R}^{12}$. $\lambda_{\text{body}} = 10^{-3}$ and $\lambda_{\text{hand}} = 10^{-4}$ are different weights that are used to control the regularization strength for the body and hand pose parameters, respectively. This loss ensures the simplicity of the results and prevents unnatural, significant twisting of the joints.

3D Position Loss $\mathcal{L}_{j\text{3D}}$ and $\mathcal{L}_{h\text{3D}}$. The 3D position loss term is defined as

$$\mathcal{L}_{\text{3D}} = \sum ||\mathbf{T}_{\text{smplx}} - \mathbf{T}_{\text{bvh}}||^2 \cdot \lambda_{\text{3D}}, \tag{3}$$

where $\mathbf{T}_{\text{smplx}} \in \mathbb{R}^3$ represents the 3D global coordinates of the joints in the SMPL-X model and $\mathbf{T}_{\text{bvh}} \in \mathbb{R}^3$ represents the corresponding 3D global coordinates of the joints in the BVH representation. $\mathcal{L}_{j3\text{D}}$ represents the 3D position loss sum for the 21 body joints, while $\mathcal{L}_{h3\text{D}}$ represents the 3D position loss sum for the 30 hand joints (15 joints per hand). These two terms have different weights, set as $\lambda_{j3\text{D}} = 1.0$ and $\lambda_{h3\text{D}} = 2.0$, respectively.

Orientation Loss $\mathcal{L}_{j\text{Ori}}$ and $\mathcal{L}_{h\text{Ori}}$. The orientation loss term is defined as

$$\mathcal{L}_{\text{Ori}} = \sum ||\mathbf{R}_{\text{smplx}} - \mathbf{R}_{\text{bvh}}||^2 \cdot \lambda_{\text{Ori}}, \tag{4}$$

which is similar to $\mathcal{L}_{\text{3D}}$, except that $\mathcal{R}_{\text{smplx}} \in \mathbb{R}^{3\times3}$ and $\mathcal{R}_{\text{bvh}} \in \mathbb{R}^{3\times3}$ represent the rotation matrices for the adjacent joints in the SMPL-X and corresponding BVH representations, respectively. Specifically, body joints named head, spine, spine2, leftUpLeg, rightUpLeg, rightShoulder, leftShoulder, rightArm, leftArm, and neck are subjected to orientation loss, ensuring that their rotations relative to adjacent nodes are close to the BVH ground truth. λ_{Ori} is set to 0.2.

Temporal Smoothness Loss $\mathcal{L}_{\text{smooth}}$. The temporal smoothness loss term is defined as

$$\mathcal{L}_{\text{smooth}} = \sum_{i=1}^{N} \left(||\theta_i - \theta_{i-1}||^2\right) \cdot \lambda_{\text{smooth}} \tag{5}$$

where $\theta_i \in \mathbb{R}^{(21+30)\times3}$ represents the body and hand pose of the i-th frame. λ_{smooth} is set to 20.0.

Contact Loss $\mathcal{L}_{\text{contact}}$. The contact loss term is defined as

$$\mathcal{L}_{\text{contact}} = \sum \left(||\mathbf{T}_{\text{finger}} - \mathbf{T}_{\text{obj}}||^2 \cdot \mathcal{J}(\mathbf{T}_{\text{finger}}, \mathbf{T}_{\text{obj}})\right) \cdot \lambda_{\text{contact}} \tag{6}$$

where $\mathcal{T}_{\text{finger}} \in \mathbb{R}^{10\times3}$ is the global coordinates of ten fingers, and $\mathcal{T}_{\text{obj}} \in \mathbb{R}^{10\times3}$ is the corresponding global coordinates of the point closest to finger. $\mathcal{J}(\mathbf{T}_{\text{finger}}, \mathbf{T}_{\text{obj}})$ is 1 when the distance between $\mathbf{T}_{\text{finger}}$ and $\mathbf{T}_{\text{obj}}$ is less than a threshold, otherwise it is 0. And λ_{contact} is 2.0.

C. Details on CORE4D-Synthetic Data Generation

In this section, we provide details on our synthetic data generation (collaboration retargeting) method. Firstly, we clarify term definitions in Section C.1. We then explicitly introduce the whole method pipeline in detail in Section C.2. Finally, we provide implementation details in Sections C.3 and C.4.

C.1. Term Definitions

We provide definitions for the terms in our collaboration retargeting pipeline as follows.

Contact Candidate: Contact candidate is a quadruple list containing all possible contact region index (person1_leftHand, person1_rightHand, person2_leftHand, person2_rightHand) on *source*'s vertices. For each *source*, we record the contact regions of the four hands in each frame of each data sequence. At the beginning of the synthetic data generation pipeline, we sample contact candidates from these records.

Contact Constraint: Having contact candidate on *source*, we apply DeepSDF-based [14] contact retargeting to transfer the contact regions to *target*. These contact regions on *target* are the contact constraints fed into the contact-guided interaction retargeting module.

Source Interaction: During each collaboration retargeting process, we sample a human-object-human collaborative motion sequence from CORE4D-Real as the source interaction to guide temporal collaboration pattern.

Interaction Candidate: Sampling N contact candidates, we apply contact-guided interaction retargeting N times and have N human-object-human motion outputs, dubbed interaction candidates. These motions would be fed into the human-centric contact selection module to assess their naturalness.

C.2. Method Pipeline

The algorithm takes a *source-target* pair as input. First, we sample contact candidates from the whole CORE4D-Real contact knowledge on *source*. For each contact candidate, we apply object-centric contact retargeting to propagate contact candidates to contact constraints on *target*. Sampling motion from CORE4D-Real provides a high-level temporal collaboration pattern, and together with augmented low-level spatial relations, we obtain interaction candidates from the contact-guided interaction retargeting. Then, the human-centric contact selection module selects the optimal candidates, prompting a contact constraint update. After multiple iterations, the process yields augmented interactions. This iterative mechanism ensures a refined augmentation of interactions, enhancing the dataset's applicability across various scenarios.

C.3. Contact-guided Interaction Retargeting

The contact-guided interaction retargeting is a two-step optimization. We start by optimizing the motion of *target*. Then with *target* contact constraints, we optimize the poses of the two persons.

Object motion retargeting. We deliberately design temporal and spatial losses to acquire consistent and smooth *target* motion. In the concern of efficiency, we jointly optimize all frames in a single data sequence with N frames. To guarantee the fidelity of object motion, we design the fidelity loss L_f to restrict the rotation $R_{o,i}$ and the translation $T_{o,i}$ with the ground-truth rotation $R'_{o,i}$ and translation $T'_{o,i}$ in N frames:

$$\mathcal{L}_f = \lambda_f \sum_i (||R'_{o,i} - R_{o,i}||_1 + ||T'_{o,i} - T_{o,i}||_1). \quad (7)$$

We then address restriction on *target*'s spatial position to avoid penetration with the ground. The spatial loss is defined as:

$$\mathcal{L}_{\text{spat}} = \lambda_{\text{spat}} \sum_i \max(-\min(\text{height}_i), 0), \quad (8)$$

where $\min(\text{height}_i)$ represents the lowest spatial position of the objects per frame. A smoothness loss is designed to constrain the object pose difference between consecutive frames:

$$\mathcal{L}_{\text{smooth}} = \lambda_{\text{smooth}} \sum_i a^2_{R_{o,i}} + a^2_{T_{o,i}}, \quad (9)$$

where a is the acceleration of rotation and translation during N frames defined as:

$$a_{R_{o,i}} = 2R_{o,i} - R_{o,i-1} - R_{o,i+1}, \quad (10)$$

$$a_{T_{o,i}} = 2T_{o,i} - T_{o,i-1} - T_{o,i+1}, \quad (11)$$

The total object motion retargeting problem is:

$$R_o, T_o \longleftarrow \underset{R_o, T_o}{\text{argmin}} (\mathcal{L}_f + \mathcal{L}_{\text{spat}} + \mathcal{L}_{\text{smooth}}). \quad (12)$$

Human motion retargeting. We next optimize each person's motion based on the motion of *target* and the contact constraint. To acquire visually plausible motion, we design the fidelity loss $\mathcal{L}_j$ and the smoothness loss $\mathcal{L}_{\text{smooth}}$. Besides, we utilize the contact correctness loss $\mathcal{L}_c$ to acquire contact consistency in *target* interaction motion, and

leverage spatial loss L_{spat} similar to Equation 8 to avoid human-ground inter-penetration.

To enhance motion fidelity, we define two loss functions $\mathcal{L}_{\text{sr}}$ and $\mathcal{L}_{\text{wr}}$ and let $L_j = \mathcal{L}_{\text{sr}} + \mathcal{L}_{\text{wr}}$. For joints from the human arms, despite following the correct temporal collaboration pattern, their global positions would vary concerning diverse object geometries. Therefore, we utilize oriented vectors pointing to their parent body joints to obtain a relative joint fidelity:

$$\mathcal{L}_{\text{sr}} = \lambda_{\text{sr}} \sum_i \sum_{j \in \text{arm}} \|(P_{j,i} - P_{\text{parent}(j),i}) - (P'_{j,i} - P'_{\text{parent}(j),i})\|_2^2, \tag{13}$$

where $P_{j,i}$ denotes the 3D global position of joint j in frame i, and P' denotes ground-truth values. $\mathcal{L}_{\text{wr}}$ denotes constraints on the global positions of other joints:

$$\mathcal{L}_{\text{wr}} = \lambda_{\text{wr}} \sum_i \sum_{j \notin \text{arm}} \|P_{j,i} - P'_{j,i}\|_2^2. \tag{14}$$

The design of the smoothness loss is similar to Equation 9, penalizing huge acceleration of human SMPL-X parameters to avoid great motion differences between frames:

$$\mathcal{L}_{\text{smooth}} = \lambda_{\text{smooth}} \sum_i \sum_{j \in \{1,2\}} (a_{\theta_{j,i}})^2 + (a_{T_{j,i}})^2 + (a_{O_{j,i}})^2. \tag{15}$$

To leverage contact constraints, we attract human hands to the corresponding contact region on *target*. We select the positions of 20 fingertips of the two persons in the i-th frame as $\mathcal{H}_i = \{\bar{P}_{\text{tip,i}}\}_{\text{tip}\in[1,20]}$, where $\bar{P}$ are tip positions in the object's coordinate system. The contact vertices on the *target* from object-centric contact retargeting are defined as $\mathcal{C} = \{\bar{P}'_{\text{tip}}\}_{\text{tip}\in[1,20]}$. We minimize the Chamfer Distance (CD) between $\mathcal{H}_i$ and $\mathcal{C}$ to obtain contact consistency:

$$\mathcal{L}_c = \lambda_c \sum_i CD(\mathcal{H}_i, \mathcal{C}). \tag{16}$$

The total human motion retargeting problem is:

$$\theta_{1,2}, T_{1,2}, O_{1,2} \longleftarrow \underset{\theta_{1,2}, T_{1,2}, O_{1,2}}{\text{argmin}} \ (\mathcal{L}_j + \mathcal{L}_c + \mathcal{L}_{\text{spat}} + \mathcal{L}_{\text{smooth}}), \tag{17}$$

In practice, we run 1,000 and 1,500 iterations respectively for object motion retargeting and human motion retargeting. The whole pipeline is implemented in PyTorch with Adam solver. The learning rate is 0.01. In object motion retargeting, λ_f for rotation is 500, for translation is 0.005, $\lambda_{\text{spat}} = 0.01$, $\lambda_{\text{smooth}} = 1$. In human motion retargeting, $\lambda_{\text{sr}} = 0.1$, $\lambda_{\text{wr}} = 0.003$, $\lambda_c = 1,000$, $\lambda_{\text{spat}} = 0.01$, and $\lambda_{\text{smooth}} = 1$.

C.4. Human-centric contact selection

The pairwise training dataset utilized for the human pose discriminator training comprises 636,424 pairs of data. Each pair encompasses a positive human pose $S_{\text{pos}} \in \mathbb{R}^{21\times3}$ and a negative human pose $S_{\text{neg}} \in \mathbb{R}^{21\times3}$. The positive human pose is sampled from the CORE4D-Real. Conversely, the negative human pose is derived from the corresponding positive sample by introducing noise to its object pose, subsequently employing the original contact information to perform contact-guided interaction retargeting. The discriminator is trained by:

$$\mathcal{L}_{\text{ranking}} = -\log(\sigma(R_{\text{pos}} - R_{\text{neg}} - m(S_{\text{pos}}, S_{\text{neg}}))), \tag{18}$$

iterating 1,000 epochs by the Adam solver with a learning rate 2e-4.

Specifically, the noise $\Delta(\alpha, \beta, \gamma, x, y, z)$ incorporates both rotational and translational components. The rotational noise $\Delta(\alpha, \beta, \gamma)$ ranges from 20 to 60 degrees, while the translational noise $\Delta(x, y, z)$ falls within the range of 0.2 to 0.5 meters. The margin is computed by:

$$m(S_{\text{pos}}, S_{\text{neg}}) = (|\alpha| + |\beta| + |\gamma|)/10 + (|x| + |y| + |z|) * 10. \tag{19}$$

During the contact constraint update process, a penetration filtering step is performed. For each frame, the penetration volume between the human and object is calculated. If the penetration volume exceeds 10^{-4} cubic meters, it is considered a penetration case. If more than 2.5% of frames within an interaction candidate exhibit penetration, the entire candidate is discarded. Among the remaining candidates, the one with the highest score from the human pose discriminator is selected to proceed with the contact constraint update.

D. Dataset Statistics and Visualization

D.1. Collaboration Modes

CORE4D encompasses five human-human cooperation modes in collaborative object rearrangement. "Move1" refers to the scenario where two participants simultaneously rearrange objects and both are aware of the target. On the other hand, "move2" represents the scenario where objects are rearranged simultaneously, but only Person 1 knows the target. "Pass" indicates that one participant passes the object to another for relay transportation. "Join" means that Person 2 joins Person 1 in carrying the object during transportation. Lastly, "leave" signifies that Person 2 leaves during the joint transportation with Person 1.

According to the different durations of the two participants' contact with the object, "move1" and "move2" can be combined into collaborative carrying tasks. "Pass" represents the task of handover and solely moving the object.

Set	#Object						#Sequence					
	Chair	Desk	Box	Board	Barrel	Stick	Chair	Desk	Box	Board	Barrel	Stick
Real	5	6	9	5	9	4	157	213	200	128	206	58
Synthetic	418	408	376	589	602	596	1767	1344	1326	2123	1495	1961

Table 2. **Statistics on object in CORE4D.**

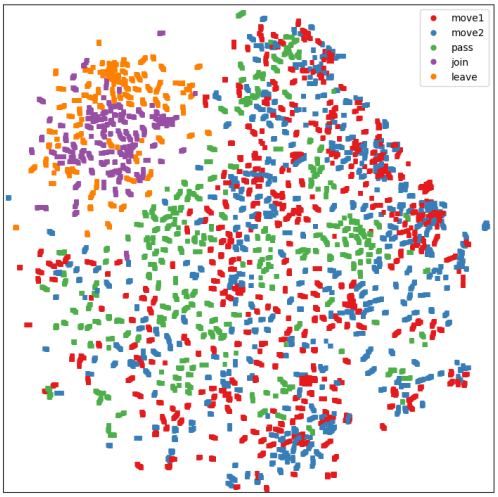

Figure 1. **T-SNE visualization of human poses for different collaboration modes.**

Incorporating the join task and the leave task, CORE4D totally comprises four different tasks (see Figure 4 in the main paper) based on the interaction between humans and objects. Fig. 5 exemplifies the motions for each task.

As depicted in Fig. 1, distinct characteristics are exhibited by different cooperation modes in high-level movements, thereby offering an innovative standpoint and potential for comprehending and investigating collaborative behaviors.

D.2. Participants

31 participants, encompassing variations in height, weight, and gender, contributed to the capturing of CORE4D-Real.

D.3. Objects

CORE4D-Real has 38 objects while CORE4D-Synthetic has about 3k objects. The objects encompass six categories, namely box, board, barrel, stick, chair, and desk, each exhibiting a rich diversity in surface shape and size. The distribution of object categories is detailed in Table 2. All the objects in CORE4D-Real are shown in Fig. 4. Fig. 3 shows samples from CORE4D-Synthetic and their interpolation process.

D.4. Camera Views

Fig. 2 shows the four allocentric and one egocentric views of our data capturing system.

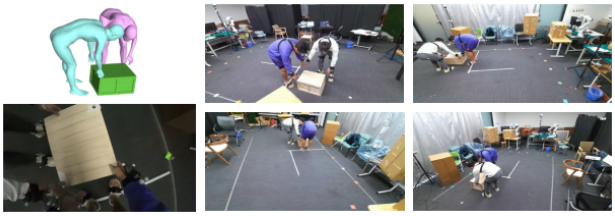

Figure 2. **Visualization of CORE4D camera views.**

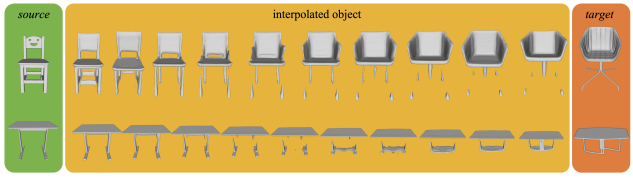

Figure 3. **Visualization of CORE4D-Synthetic objects and interpolation.**

E. Details on Data Split

Benefiting from the diverse temporal collaboration patterns from CORE4D-Real and the large data amount of CORE4D-Synthetic, we randomly select a subset of real object models and construct the training set as the combination of their real (T-Real) and synthesized (T-Synthetic) collaboration motion sequences. We formulate two test sets on CORE4D-Real supporting studies of both non-generalization and inner-category generalization. The first test set (S1) consists of interaction performed on the objects that appear in the training set, while the second one (S2) is composed of interaction from novel objects. Detailed data distribution of each object category is shown in Table 3.

F. Evaluation Metrics for Benchmarks

F.1. Human-object Motion Forecasting

Evaluation metrics include the human joints position error J_e, the object translation error T_e, the object rotation error R_e, the human-object contact accuracy C_{acc}, and the penetration rate P_r.

Set	#Object						#Sequence					
	Chair	Desk	Box	Board	Barrel	Stick	Chair	Desk	Box	Board	Barrel	Stick
T-Real	3	4	6	3	6	2	93	104	96	51	113	25
T-Synthetic	418	408	376	589	602	596	1767	1344	1326	2123	1495	1961
S1	3	4	6	3	6	2	40	62	45	21	51	6
S2	2	2	3	2	3	2	24	47	59	56	42	27

Table 3. **Train-test split on CORE4D.**

- We define J_e as the average Mean Per Joint Position Error (MPJPE) of the two persons. MPJPE represents the mean per-joint position error of the predicted human joint positions and the ground-truth values. The unit of J_e is one millimeter.
- Translation error (T_e) and rotation error (R_e) denote the average L2 difference between the predicted object translation vectors and the ground-truth ones, and the average geodesic difference between the estimated object rotation matrices and the ground-truth ones, respectively. The unit of T_e is one millimeter. The unit of R_e is one degree.
- Physical metrics: To assess contact fidelity, we detect contacts on the two hands of the two persons for each frame with an empirically designed distance threshold (5 centimeters). We then examine the contact accuracy (C_{acc}), which indicates the average percentage of contact detection errors in the predicted motions. Additionally, we examine the object penetration ratio (P_r) representing the mean percentage of object vertices inside the human meshes. The units of the two metrics are percentages.

F.2. Interaction Synthesis

Following an existing individual human-object interaction synthesis study [9], the evaluation metrics include the root-relative human joint position error $RR.J_e$, the root-relative human vertex position error $RR.V_e$, the human-object contact accuracy C_{acc}, and the FID score (*FID*).

- $RR.J_e$ denotes the average root-relative MPJPE of the two persons. The root-relative MPJPE represents the mean per-joint position error of the predicted human joint positions relative to the human root position and the ground-truth values. The unit of $RR.J_e$ is one millimeter.
- $RR.V_e$ denotes the average root-relative Mean Per Vertex Position Error (MPVPE) of the two persons. The root-relative MPVPE represents the mean per-vertex position error of the predicted human vertex positions relative to the human root position and the ground-truth values. The unit of $RR.V_e$ is one millimeter.
- C_{acc} is the same as that in Section F.1.
- The Fréchet Inception Distance (*FID*) quantitatively evaluates the naturalness of synthesized human motions. We first train a feature extractor on CORE4D-Real to encode each human-object-human motion sequence to a 256D feature vector $\bar{f}_i$ and acquire the ground-truth human motion feature distribution $\bar{D}$=$\{\bar{f}_i\}$. We then replace the motions of the two persons as synthesized ones and obtain another distribution D=$\{f_i\}$. Eventually, the *FID* denotes the 2-Wasserstein distance between $\bar{D}$ and D. Since CORE4D-Real provides action labels, the feature extractor is supervised-trained by fulfilling the action recognition task. The network structure of the feature extractor is a single-layer Transformer [21].

G. Qualitative Results on Benchmarks

Figure 6 and Figure 7 exemplify generated motions for the human-object motion forecasting task and the interaction synthesis task, respectively. Since the baseline methods do not focus on generating hand poses, we replace hand poses in ground truth with flat hands to facilitate fair comparisons. Despite diverse cooperation modes that can be generated, the baseline methods consistently encompass unsatisfactory performances including unnatural collaboration, inter-penetration, and unnatural contact.

H. Details on the Application of CORE4D-Synthetic

To evaluate the application of CORE4D-Synthetic, we use the lightweight CAHMP [4] to conduct the motion forecasting experiments. Unlike the experiments in section **Human-object Motion Forecasting** mentioned in the main paper, where 15 frames are predicted, here we predict the human-object motion for the next 10 frames given the previous 10 frames.

H.1. Task Formulation

Given the object's 3D model and human-object poses in adjacent 10 frames, the task is to predict their subsequent poses in the following 10 frames. The human pose $P_h \in \mathbb{R}^{23\times3}$ represents the joint rotations of the SMPL-X [15] model, while the object pose $P_o = \{R_o \in \mathbb{R}^3, T_o \in \mathbb{R}^3\}$ denotes 3D orientation and 3D translation of the rigid object model.

H.2. Evaluation Metrics

Following existing motion forecasting works [3, 22, 24], we evaluate human joints position error J_e, object translation error T_e, object rotation error R_e. Details of the three metrics can be found in Section F.1.

H.3. Results

Comparing the 1K real dataset with the 0.1K real dataset supplemented with synthetic data generated through retargeting, we observed that the quality of the synthetic data is comparable to the real data. Additionally, due to the increased diversity of objects and enriched spatial relations between humans and objects in the synthetic data, it exhibits better generalization performance in object motion forecasting.

Comparing the evaluation results of the 1K real dataset with the results obtained by augmenting it with additional 4K synthetic data, we observed a significant performance gain from the synthetic data. This demonstrates that the inclusion of synthetic data enhances the value of our dataset and better supports downstream tasks.

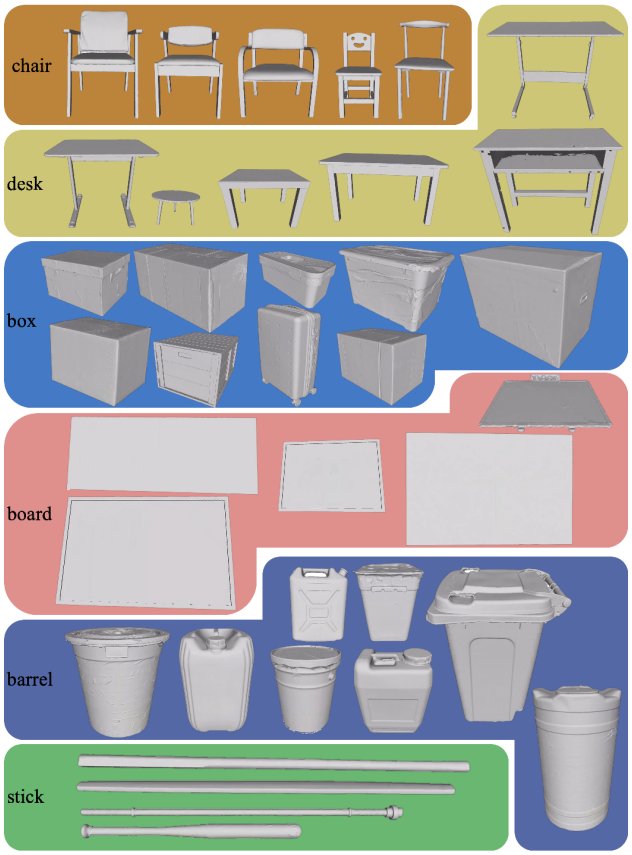

Figure 4. **Visualization of CORE4D-Real objects.**

I. Details on Humanoid Skill Learning using CORE4D

As introduced in Section 5.2 in the main paper, we use CORE4D's human-box interaction data to facilitate humanoid skill learning for box lifting. This section presents details on this application, including the simulation environment configuration (Section I.1), task formulation and evaluation (Section I.2), adapting human interaction data to humanoid (Section I.3, I.4), benchmark method designs (Section I.5), and experiments (Section I.6). An additional evaluation of policy's generalizability is shown in Section I.7.

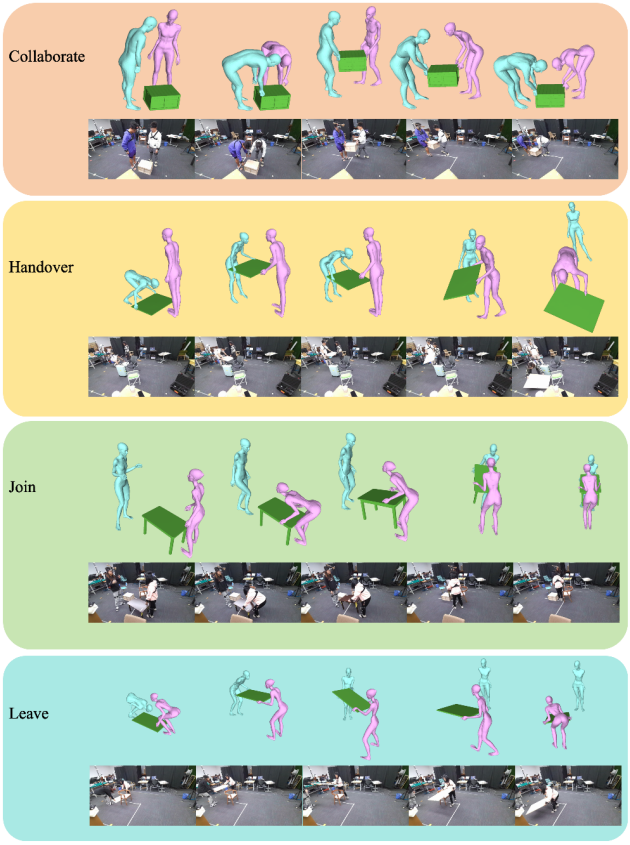

Figure 5. **Visualization of CORE4D object rearrangement tasks.**

I.1. Environment Setup

We use the popular Unitree H1 humanoid robot [20] in Isaac Gym [12] simulation environment. The H1 humanoid has 19 revolute joints with fixed limits on motion ranges. The interaction scene contains the humanoid, a box weighing 0.5kg initially posed on the floor, and a third-person-view camera providing visual signals for the skill policy.

Data clipping: The original interactions in CORE4D are collaborations of two persons. In this application, we regard each of them as two individual human-object interaction motions and obtain the box-lifting clips automatically by measuring hand-object distances and the object's height. We discard the motion if the individual is not touching the object during its lifting process.

Train-test split: As a result, we acquire 890 individual human-box lifting data clips covering 22 boxes augmented

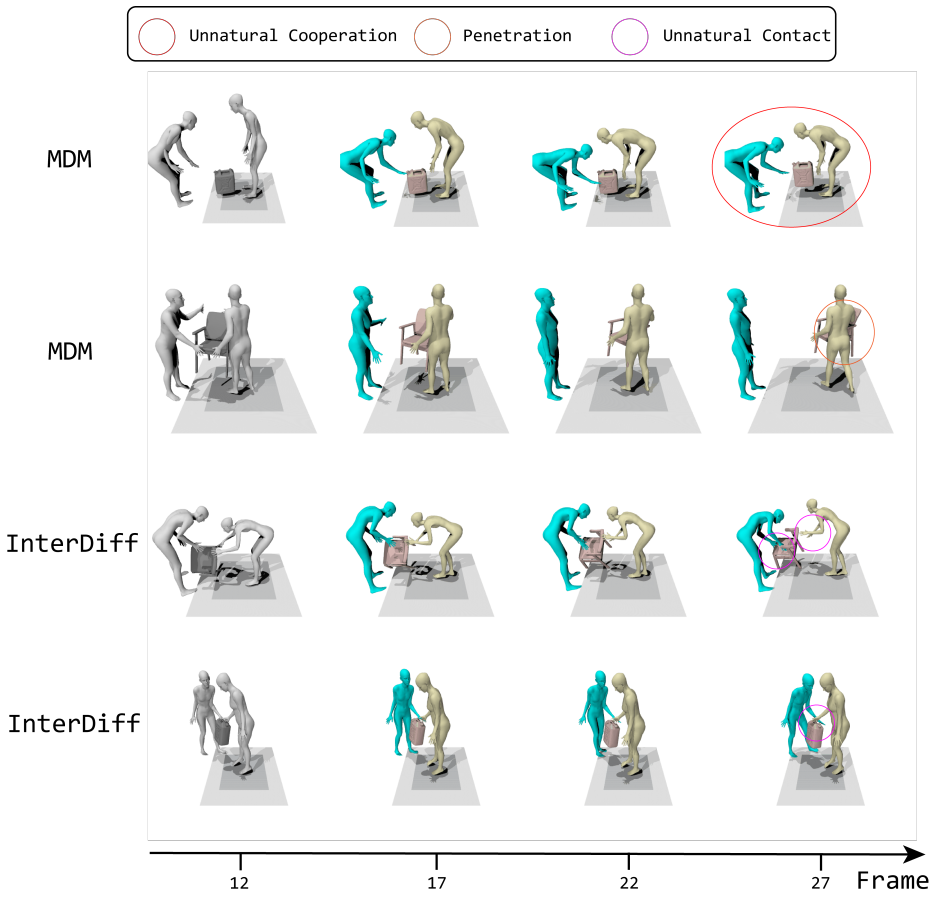

Figure 6. **Qualitative results of human-object motion forecasting.** Grey meshes are from the task inputs.

from three real-world ones. We select 16 boxes (504 clips) as the training set and the remains as the test set. The training set is used only to train the skill policy (Section I.5) and its preceding motion tracker (Section I.4), while the test set is used only to evaluate the skill policy to assess its generalizability to unseen box shapes.

Observations: The real-time input of the humanoid skill policy is $\mathcal{I} = \{s_{\text{proprio}}, r_{\text{root}}, t_{\text{root}}, \mathcal{C}, \mathcal{D}\}$, where $s_{\text{proprio}} \in \mathbb{R}^{19\times 2}$ is humanoid's proprioception, $r_{\text{root}} \in \mathbb{R}^3$ and $t_{\text{root}} \in \mathbb{R}^3$ are humanoid's root orientation and position in the world coordinate system respectively, and $\mathcal{C} \in \mathbb{R}^{360\times 480\times 3}$ and $\mathcal{D} \in \mathbb{R}^{360\times 480}$ are the color and depth image captured by the camera. Specifically, s_{proprio} is composed of the angles and velocities of each joint.

Action: Given the real-time input $\mathcal{I}$, the skill policy needs to generate an action $\mathcal{A}$ and use it to actuate the humanoid in the simulation environment. The action $\mathcal{A} \in \mathbb{R}^{19}$ is defined as 19 DoF joint angles, which are transferred to joint torques via a PD controller with pre-defined proportional-derivative gains [5]. The simulation environment uses the computed joint torques to actuate the humanoid.

I.2. Task Formulation and Evaluation Metric

Given an unseen box in the scene and a starting position, the humanoid is required to adjust its pose, touch the box, and finally lift it larger than 20 centimeters. The evaluation metric (*SR*) is the task success rate defined as whether the box reaches 20 centimeters higher than its initial position.

I.3. Retargeting Interactions from CORE4D onto H1 Humanoid Robot

Following existing humanoid skill learning advances [6, 7, 18], we use an optimization strategy to solve the retargeting problem. The optimization is finding the optimal sequence of $\{s_{\text{proprio}}, r_{\text{root}}, t_{\text{root}}\}$ that can minimize differences between motions of the human A and the humanoid B on positions of K paired joints in the world coordinate system $L_p = \sum_{k=1}^{K} \|P_{a_k} - P_{b_k}\|^2$, where $P \in \mathbb{R}^{T\times 3}$ denotes the position sequence of a joint, and T is the frame number. $< a_k, b_k >$ is a pre-defined pair of joints with similar semantics, where a_k is from the human skeleton, and b_k is from the humanoid skeleton.

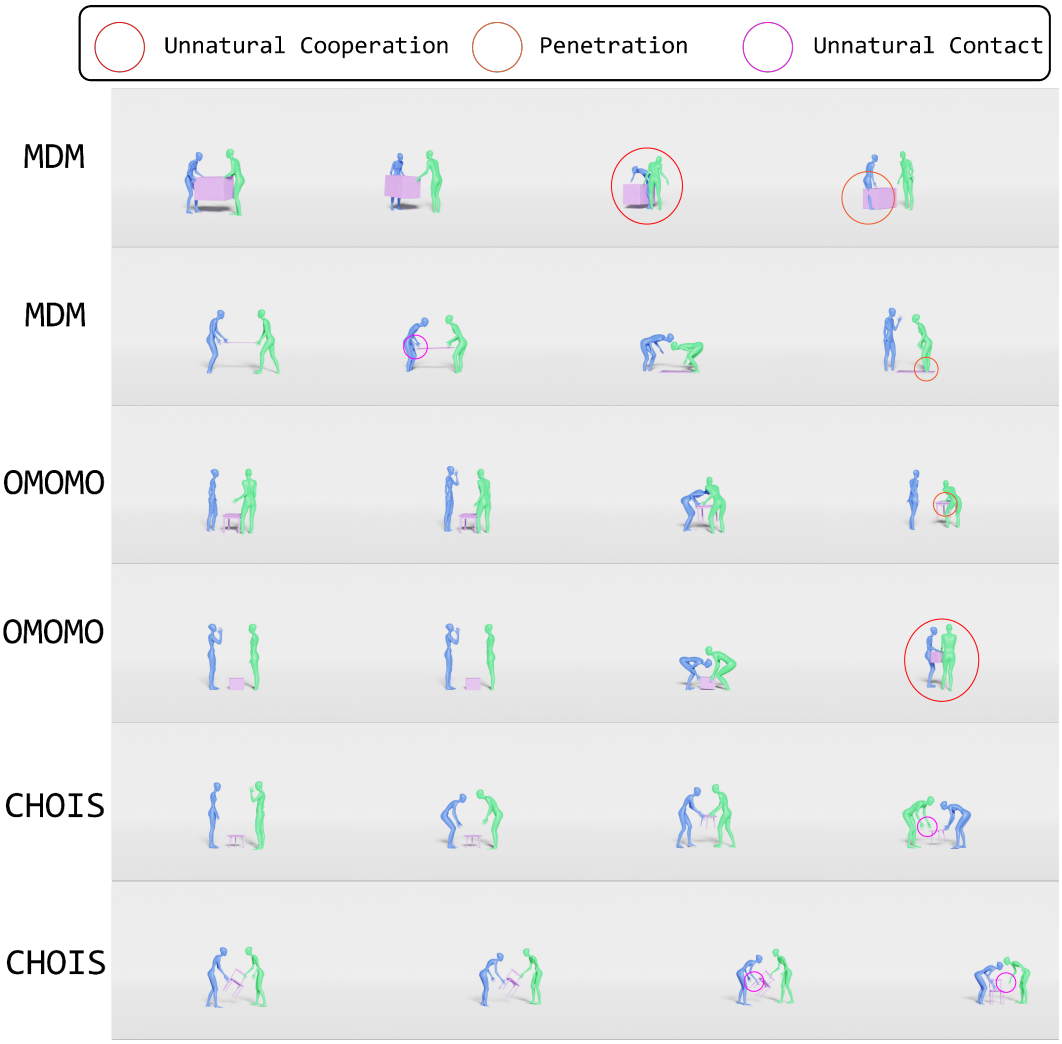

Figure 7. **Qualitative results of interaction synthesis.**

We design a temporal loss $L_t = \sum_{t=2}^{T}\sum_{i=1}^{19}(\|s_{\text{proprio},t,i} - s_{\text{proprio},t-1,i}\|^2 + \|r_{\text{root},t,i} - r_{\text{root},t-1,i}\|^2 + \|t_{\text{root},t,i} - t_{\text{root},t-1,i}\|^2)$ to further improve motion smoothness. The overall optimization target is $L_p + L_t$.

As a result, this retargeting method converts human-box interaction motions to humanoid-box interaction animations. The animations are not physically realistic, and we use the upcoming tracking method to obtain physically realistic ones following HumanPlus [5].

I.4. Tracking Humanoid-box Interaction Animations

Given a humanoid-box interaction animation, the motion tracking methods [11, 16, 19] control the humanoid in the simulation environment and generate the motion that mostly closely resembles the animation. The generated motions are both physically realistic and able to fulfill the task via mimicking animations, which are utilized as skill demonstrations in the following skill policy learning.

We select HST [5] as our tracker. The official implementation of the HST [5] tracker fails to lift the box, and we design an improved HST that can successfully track the animations and control the humanoid to lift the box physically. We describe the key designs of our tracker below.

Input: In each frame, the tracker inputs the states of the real-time humanoid S_h, the humanoid animation target $\hat{S}_h$, the real-time box S_b, and the box target $\hat{S}_b$. $S_h = \{r_{\text{root}}, \dot{r}_{\text{root}}, t_{\text{root}}, \dot{t}_{\text{root}}, J, \dot{J}, R, \dot{R}, T, \dot{T}\}$, where $r_{\text{root}} \in \mathbb{R}^3$, $\dot{r}_{\text{root}} \in \mathbb{R}^3$, $t_{\text{root}} \in \mathbb{R}^3$, $\dot{t}_{\text{root}} \in \mathbb{R}^3$, $J \in \mathbb{R}^{19}$, $\dot{J} \in \mathbb{R}^{19}$, $R \in \mathbb{R}^{(19+1)\times 3}$, $\dot{R} \in \mathbb{R}^{(19+1)\times 3}$, $T \in \mathbb{R}^{(19+1)\times 3}$, $\dot{T} \in \mathbb{R}^{(19+1)\times 3}$ denotes root's global orientation, root's global angular velocity, root's global position, root's global linear velocity, joint angle, joint velocity, global positions of the

root and joints, global linear velocities of the root and joints, global orientations of the root and joints, and global angular velocities of the root and joints. $\hat{S}_h$ is formulated the same as S_h. $S_b = \{r_b, \hat{r}_b, t_b, \hat{t}_b\}$, where r_b, $\hat{r}_b$, t_b, $\hat{t}_b$ denotes box global orientation, box global angular velocity, box global position, and box global linear velocity, respectively. $\hat{S}_b$ shares the same definition with S_b.

Output: In each frame, the tracker outputs a 19 DoF action vector representing target joint angles. The joint torques are computed via a low-level PD controller and are fed into the simulation environment to actuate the humanoid.

Reward function: The tracker is trained via a reinforcement learning (RL) method PPO [17]. The reward design is the most crucial part of the method's performance. Using original rewards from HumanPlus [5] cannot lift the box successfully due to inaccurate tracked hand positions. To handle this issue, we draw inspiration from PhysHOI [23] and use a multiplication of humanoid reward $\mathcal{R}_h$, box reward $\mathcal{R}_b$, and humanoid-box interaction reward $\mathcal{R}_i$ as the overall reward $\mathcal{R}_{\text{overall}}$: $\mathcal{R}_{\text{overall}} = \mathcal{R}_h \times \mathcal{R}_b \times \mathcal{R}_i$.

- $\mathcal{R}_h = 0.5\exp(-5\|T - \hat{T}\|_1) + 5\exp(-10\|t_{\text{root}} - \hat{t}_{\text{root}}\|_1) + 10\exp(-10\|J_l - \hat{J}_l\|_1) + 10\exp(-10\|J_r - \hat{J}_r\|_1)$, where l/r denotes the left/right hand.
- $\mathcal{R}_b = \exp(-10\|t_b - \hat{t}_b\|_1)$.
- $\mathcal{R}_i = \exp(-10\|P_{l,h\to b} - \hat{P}_{l,h\to b}\|_1 - 10\|P_{r,h\to b} - \hat{P}_{r,h\to b}\|_1)$, where $P_{h\to b}$ and $\hat{P}_{h\to b}$ denotes the humanoid joint position in the box's coordinate system, and l/r denotes the left/right hand.

Training strategies: We adopt an early-termination strategy from DeepMimic [16] that terminates the rollout when the humanoid is 0.5 meters away from its target or its root height is below 0.5 meters.

I.5. Reinforcement Learning and Imitation Learning Method Designs

Combining the retargeting method (Section I.3) with the improved HST tracker (Section I.4), we transfer CORE4D's data to physically realistic humanoid box-lifting demonstrations. The final step is to train a skill policy that mimics the demonstrations and can lift unseen boxes in test time. We select two **vision-based** imitation learning (IL) methods, HIT [5] and ACT [26], and use their official implementations.

To examine the value of the demonstrations, we compare the two IL methods with a commonly used **state-based** RL algorithm PPO [17]. The PPO is implemented with the code from HumanPlus [5], with a change on the reward design $\mathcal{R}$: $\mathcal{R} = \mathcal{R}_b + \mathcal{R}_{\textbf{success}} + \mathcal{R}_i + \mathcal{R}_{\textbf{alive}}$, where:

- $\mathcal{R}_b = \exp(\|t_b - \hat{t}_b\|_2^2)$, where $\hat{t}_b$ is the pre-defined target box center position for the task.
- $\mathcal{R}_{\textbf{success}} = [\mathcal{R}_d < 0.01]$ encouraging achieving the task.
- $\mathcal{R}_i = -0.1(\|P_{l,h\to b}\|_2^2 + \|P_{r,h\to b}\|_2^2)$, where $P_{h\to b}$ and $\hat{P}_{h\to b}$ denotes the humanoid joint position in the box's coordinate system, and l/r denotes the left/right hand. This reward encourages humanoid hands to explore near the object.
- $\mathcal{R}_{\textbf{alive}} = 0.1$ encouraging the humanoid being alive.

I.6. Experiments

The evaluation results are shown in Table 6 and Figure 6 in the main paper. Leveraging CORE4D data, the policy can achieve 21.0% (for HIT) and 26.5% (for ACT) task success rates, which are significantly larger than that of data-free PPO (0.0%), demonstrating the value of CORE4D for humanoid box-lifting skill learning.

I.7. Generalizing the Policy to New Box Weights

The boxes used in policy training weigh 0.5kg, while the boxes in CORE4D-Real are empty cartons with weights between 0.5kg and 2kg. We use the policy (HST + AST) trained with 0.5kg boxes to lift heavier boxes and report the success rates in Table 4. The results indicate that the policy can generalize to new box weights to some extent and has the potential to handle common empty cartons.

Box Weight (kg)	0.5	1.0	2.0	4.0
Success Rate (%)	26.5	11.2	7.1	3.0

Table 4. **Using the policy to lift boxes with different weights.**

J. CORE4D-Real Data Capturing Instructions and Language Annotations

This section describes the instructions given to the participants before data capturing (Section J.1) and the corresponding language annotations generated by a large language model automatically (Section J.2).

J.1. Data Capturing Instructions

Target. We divide a $4m \times 5m$ field into 20 squares and number them, and place colored labels as markers along the perimeter of the field. The following language instructs participants: *"Please collaboratively move the object to the target square. You can choose any path and orientation of the object as you like. It is not necessary to be overly precise with the final position - a rough placement is fine. Do not make unnatural motions just to achieve an exact position. Do not use verbal communication with each other."*. As for the settings when only one participant knows the target, the target square number is written on a piece of paper and shown to the participant who knows the target. And additional instructions are given as: *"If you know the target, do not use language or direct body language to inform the other party (such as pointing out the location). If you do*

System Prompt	You are a narrator skilled in summarizing the behaviors of two people collaboratively rearranging an object in one sentence and maintaining key information. Each of the behaviors is described in the following three aspects: (1) An overall task description, indicating the goal of the behavior and the constraints of the target object pose. (2) A collaboration mode instruction, specifying when should each person contact the object. (3) The obstacle situation, indicating whether the scene contains obstacles and its influence on the behavior.
User Prompt	Please clearly summarize the following behavior in a sentence. (1) Task description: Please collaboratively move the object to the target square. You can choose any path and orientation of the object as you like. It is not necessary to be overly precise with the final position – a rough placement is fine. (2) Collaboration mode instruction: Based on the target, please cooperatively transport the chair. Both participants should be in contact with the object throughout the process. (3) Obstacle situation: There are a varying number of obstacles on the field. If they get in your way, decide how to solve it using common everyday operations. If the obstacles occupy the destination, place the object near the destination.
GPT-4 Response	Two people cooperatively move a chair to a target square, maintaining contact with the chair throughout, and adjust their path as needed to navigate around obstacles, placing the chair near the target if the exact spot is obstructed.

Table 5. **The language annotation generation process by using GPT-4.**

not know the target, please assist the other participant in completing the transportation.".

Collaboration Mode. The instructions are given as follows to indicate different Collaboration Modes for the participants. For Collaborate mode: *"Based on the target, please cooperatively transport the object, or upright any overturned tables, chairs, etc. Both participants should be in contact with the object throughout the process.".* For Handover mode: *"Please decide the handover point yourselves, then have one person hand the object to the other, completing the object transfer in relay.".* For Leave and Join modes: *"One person will transport the object throughout, while the other leaves or joins to help at a time point not disclosed to the collaborator.".*

Obstacle. The instructions are given as follows to guide the participants in tackling obstacles: *"There are a varying number of obstacles on the field. If they get in your way, please decide on your own how to solve it using some common everyday operations. If the obstacles occupy the destination, please place the object near the destination.".*

J.2. Language Annotations

For each motion sequence in CORE4D-Real, given the instructions shown in Section J.1, we generate the corresponding language annotation using GPT-4, where the system prompt, the user prompt and the response from GPT-4 are shown in Table 5.

References

[1] Bharat Lal Bhatnagar, Xianghui Xie, Ilya A Petrov, Cristian Sminchisescu, Christian Theobalt, and Gerard Pons-Moll. Behave: Dataset and method for tracking human object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15935–15946, 2022. 1

[2] 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. 1

[3] Enric Corona, Albert Pumarola, Guillem Alenya, and Francesc Moreno-Noguer. Context-aware human motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6992–7001, 2020. 6

[4] Enric Corona, Albert Pumarola, Guillem Alenyà, and Francesc Moreno-Noguer. Context-aware human motion prediction, 2020. 6

[5] Zipeng Fu, Qingqing Zhao, Qi Wu, Gordon Wetzstein, and Chelsea Finn. Humanplus: Humanoid shadowing and imitation from humans. *arXiv preprint arXiv:2406.10454*, 2024. 8, 9, 10

[6] Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024. 8

[7] Tairan He, Zhengyi Luo, Wenli Xiao, Chong Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Learning human-to-humanoid real-time whole-body teleoperation. *arXiv preprint arXiv:2403.04436*, 2024. 8

[8] Jiefeng Li, Siyuan Bian, Chao Xu, Zhicun Chen, Lixin Yang, and Cewu Lu. Hybrik-x: Hybrid analytical-neural inverse kinematics for whole-body mesh recovery. *arXiv preprint arXiv:2304.05690*, 2023. 1

[9] Jiaman Li, Jiajun Wu, and C Karen Liu. Object motion guided human motion synthesis. *ACM Transactions on Graphics (TOG)*, 42(6):1–11, 2023. 6

[10] Yun Liu, Haolin Yang, Xu Si, Ling Liu, Zipeng Li, Yuxiang Zhang, Yebin Liu, and Li Yi. Taco: Benchmarking generalizable bimanual tool-action-object understanding. *arXiv preprint arXiv:2401.08399*, 2024. 1

[11] Zhengyi Luo, Jinkun Cao, Kris Kitani, Weipeng Xu, et al. Perpetual humanoid control for real-time simulated avatars. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10895–10904, 2023. 9

[12] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller,

Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. 7

[13] Maddock Meredith, Steve Maddock, et al. Motion capture file formats explained. *Department of Computer Science, University of Sheffield*, 211:241–244, 2001. 2

[14] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 3

[15] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10975–10985, 2019. 1, 2, 6

[16] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel Van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)*, 37(4):1–14, 2018. 9, 10

[17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 10

[18] Annan Tang, Takuma Hiraoka, Naoki Hiraoka, Fan Shi, Kento Kawaharazuka, Kunio Kojima, Kei Okada, and Masayuki Inaba. Humanmimic: Learning natural locomotion and transitions for humanoid robot via wasserstein adversarial imitation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13107–13114. IEEE, 2024. 8

[19] Chen Tessler, Yunrong Guo, Ofir Nabati, Gal Chechik, and Xue Bin Peng. Maskedmimic: Unified physics-based character control through masked motion inpainting. *arXiv preprint arXiv:2409.14393*, 2024. 9

[20] Unitree. Unitree's first universal humanoid robot. https://www.unitree.com/h1, 2018. 7

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 6

[22] Weilin Wan, Lei Yang, Lingjie Liu, Zhuoying Zhang, Ruixing Jia, Yi-King Choi, Jia Pan, Christian Theobalt, Taku Komura, and Wenping Wang. Learn to predict how humans manipulate large-sized objects from interactive motions. *IEEE Robotics and Automation Letters*, 7(2):4702–4709, 2022. 6

[23] Yinhuai Wang, Jing Lin, Ailing Zeng, Zhengyi Luo, Jian Zhang, and Lei Zhang. Physhoi: Physics-based imitation of dynamic human-object interaction. *arXiv preprint arXiv:2312.04393*, 2023. 10

[24] Sirui Xu, Zhengyuan Li, Yu-Xiong Wang, and Liang-Yan Gui. Interdiff: Generating 3d human-object interactions with physics-informed diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14928–14940, 2023. 6

[25] Ming Yan, Xin Wang, Yudi Dai, Siqi Shen, Chenglu Wen, Lan Xu, Yuexin Ma, and Cheng Wang. Cimi4d: A large multimodal climbing motion dataset under human-scene interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12977–12988, 2023. 1

[26] 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. 10