HOI4D: A 4D Egocentric Dataset for Category-Level Human-Object Interaction —— Supplementary Material

Yunze Liu*[1,3], Yun Liu*[1], Che Jiang[1], Kangbo Lyu[1], Weikang Wan[2], Hao Shen[2], Boqiang Liang[2], Zhoujie Fu[1], He Wang[2], Li Yi†[1,3]
[1] Tsinghua University, [2] Peking University, [3] Shanghai Qi Zhi Institute
https://hoi4d.github.io

Supplementary Material

A. Details of HOI4D

A.1. CAD Model Visualization

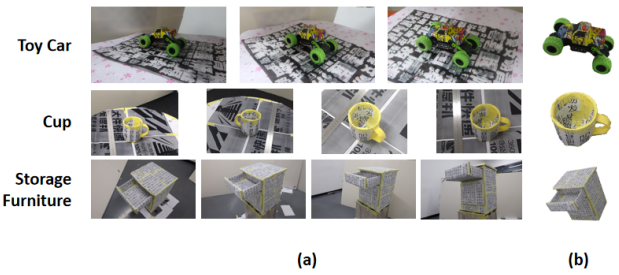

Figure 1. We show some examples of **(a)** our multi-view high-resolution color images with various depression angles, **(b)** corresponding CAD models.

Figure 1 shows some examples of CAD models in HOI4D. We first manually decorate objects with stickers for some categories to enrich the object texture and hide some highly specular areas. We then use off-the-shelf software packages [**?**, **?**] to reconstruct the CAD model from multi-view high-resolution color images. Various camera depression angles shown in Figure 1(a) provide realistic geometry of the CAD model with the finest detail and reconstruct both inner and outer surfaces of some specific categories such as cup and storage furniture. More visualization of CAD models in HOI4D are shown in Figure 2.

A.2. Dataset Statistics

To better support various research fields with different requirements for scene complexity, we manually divide all scenes into simple and complex scenarios. In a simple scene, the object interacted with is not obscured by surrounding objects, and the variety of camera views in the video is small to improve consistency across frames. Complex scenes have no such restrictions. Figure 3 and Table 1 report the number of instances in different scenarios. Moreover, Table 2 reports the proportion of object points in the point cloud of scenes in diverse categories.

Scene	Mean (std)
Simple	1.67 (0.79)
Complex	6.02 (2.93)

Table 1. The average number of objects in the simple and complex scenes.

Scene	3D static scene	4D dynamic scene
Simple	6.3	10.7
Complex	13.9	17.0

Table 2. The percentage of objects points in the point cloud of simple and complex scenes.

A.3. Task Definitions

Table 3 shows the detailed tasks defined by each category. There are 76 tasks in all 16 categories. Pick-and-place tasks are included for each object, as well as functionality-based tasks that can be used to perceive object mobility and functionality in interactive scenarios.

A.4. Hand Pose Loss Terms in Hand Pose Annotation

Joint angle loss. The joint angle loss term $\mathcal{L}_j$ is defined following [**?**] as:

$$\mathcal{L}_j = \sum_{i=1}^{45} \max(\underline{\theta_i} - \theta_m[i], 0) + \max(\theta_m[i] - \overline{\theta_i}, 0) \quad (1)$$

where $\underline{\theta_i}$ and $\overline{\theta_i}$ correspond to the lower and upper limits of the i^{th} joint angle parameter $\theta_m[i]$.

2D joint loss. The 2D joint loss of MANO joints is defined

*Equal contribution.
†Corresponding author.

Figure 2. Some examples of CAD models among all 16 categories.

as:

$$\mathcal{L}_{2D} = \sum_{i=1}^{21} c[i] \left\| proj(J_{\theta_m}[i]) - J_{anno}[i] \right\|_2^2 \quad (2)$$

where $proj(J_{\theta_m}[i])$ denotes the 2D projection of the i^{th} 3D joint location J_{θ_m}, $J_{anno}[i]$ is the corresponding 2D annotation, and $c[i]$ is the confidence coefficient. In practice, we set confidence coefficient $c[i]$ to 1 if joint i is visible, and 0.8 otherwise.

Mask loss. The mask loss is defined following [**?**] as:

$$\mathcal{L}_m = 1 - \frac{\|\hat{I}_m \otimes I_m\|_1}{\|\hat{I}_m \oplus I_m - \hat{I}_m \otimes I_m\|_1} \quad (3)$$

where $\hat{I}_m$ and I_m denote the rendered and the ground-truth 2D mask respectively. $\oplus$ and $\otimes$ are the pixel-wise product and sum operators respectively.

Depth loss. The depth loss is defined following [**?**] as:

$$\mathcal{L}_d = \sum_{p \in \hat{I}_m \otimes I_m} \left\| \hat{D}_p - D_p \right\|_1 \quad (4)$$

where $\hat{D}_p$ and D_p denote the rendered and the ground-truth depth at pixel p respectively. The depth loss calculate l_1 loss of depth in pixels where $\hat{I}_m$ and I_m intersect.

Point cloud loss. We use chamfer-distance from the rendered mano hand vertices to the ground truth point cloud which refers to the depth image cropped by 2D hand mask. This term compensates the depth loss, giving supervision to all mano hand vertices.

Contact loss. The contact loss term $\mathcal{L}_{Contact}$ is defined following [**?**] as:

$$\mathcal{L}_{Contact} = \lambda_R \mathcal{L}_R + (1 - \lambda_R)\mathcal{L}_A \quad (5)$$

where attraction loss $\mathcal{L}_A$ penalize hand vertices near the object's surface but are not in contact, repulsion loss $\mathcal{L}_R$ penalizes hand and object interpenetration. The contact weighting coefficient $\lambda_R \in [0, 1]$ balances between $\mathcal{L}_A$ and $\mathcal{L}_R$. $\lambda_R = 1$ when the action segmentation label indicates the hands are not interacting with the object. These terms is manually tuned for different contact modes.

Temporal consistency loss. The temporal consistency loss

Bowl	Bottle	Mug	Toy Car
Pick and place Put it in the drawer Take it out of the drawer Take the ball out of the bowl Put the ball in the bowl Pick and place(with ball)	Pick and place Pour all the water into a mug Put it in the drawer Take it out of the drawer Reposition the bottle Pick and place(with water)	Pick and place Put it in the drawer Take it out of the drawer Fill with water by a kettle Pour water into another mug Pick and place(with water)	Pick and place Push toy car Put it in the drawer Take it out of the drawer
Bucket	**Knife**	**Kettle**	**Chair**
Pick and place Pour water into another bucket	Pick and place Put it in the drawer Take it out of the drawer Cut apple	Pick and place Pour water into a mug	Pick and place with both hands Pick and place with one hand
Storage Furniture	**Pliers**	**Laptop**	**Lamp**
Open and close the drawer Open and close the door Put the drink in the door Put the drink in the drawer	Pick and place Put it in the drawer Take it out of the drawer Clamp something	Pick and place Open and close the display	Pick and place Turn and fold Turn on and turn off
Safe	**Garbage Can**	**Scissors**	**Stapler**
Open and close the door Put something in it Take something out of it	Open and close Throw something in it	Pick and place Cut something	Pick and place Bind the paper

Table 3. Task Definitions of 16 Categories.

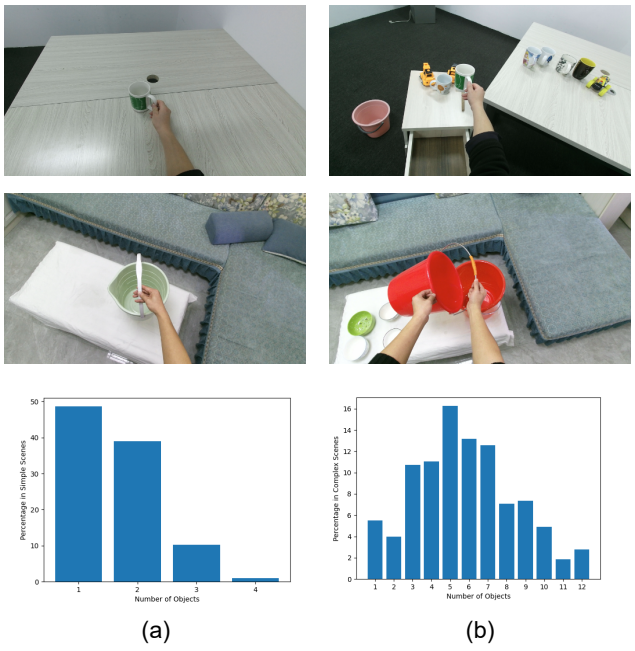

Figure 3. Sample scenes and statistics on the number of objects in a scene. **(a)** Simple scenes. **(b)** Complex Scenes.

term $\mathcal{L}_{tc}$ is defined following [**?**] as:

$$\mathcal{L}_{tc} = \sum_{t \in \mathcal{T}} (\|\Delta_h^t\|^2 + \|\Delta_h^t - \Delta_h^{t-1}\|^2) \tag{6}$$

where $\Delta_h^t = \theta_h^t - \theta_h^{t-1}$. $\mathcal{T}$ is an optimization batch consists of 6 to 11 consecutive frames.

B. Qualitative analysis of Category-Level Pose Tracking

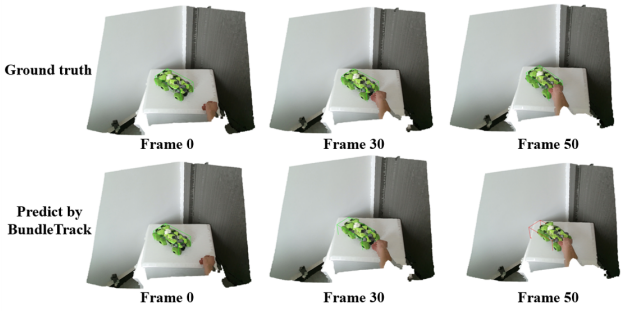

Figure 4. Qualitative evaluation on the toy car trajectory.

Figure 4 below illustrates some failure cases of pose tracking experiments. We found that human objects interaction is mainly responsible for failures for existing methods. When the camera moves, but the human is not interacting with the object, the existing methods are capable of this task; When the human interacts with the object, the occlusion of the hand and the rapid movement of the object

greatly improves the difficulty of pose tracking. It is expected that more research will focus on category-level pose tracking in real-world human-object interaction scenarios.

C. Categories of Action Segmentation

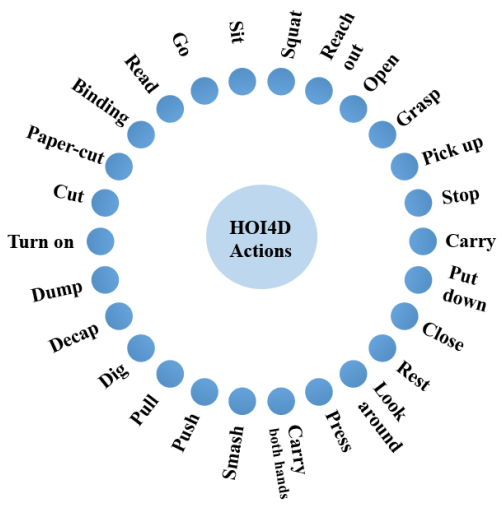

Figure 5. Categories of Action Segmentation.

Figure 5 shows the action categories defined in the action segmentation task. Different from the action definition in the existing action segmentation dataset, our action category is a finer-grained label in the scene of human-object interaction.

D. Imitation Learning for Robot Dexterous Manipulation

HOI4D not only can serve for vision benchmarks regarding category-level human-object interaction but also provides rich knowledge for robot learning. The human-object interaction trajectories combined with the pose states of objects and hands can be naturally treated as demonstrations for robot imitation learning. This allows robots to accomplish complex interactions with various objects, which are still very challenging in the robotics community. In this section, we consider a new robotics task named category-level dexterous manipulation, where a dexterous hand needs to manipulate novel object instances from a known object category in a predefined way. Specifically, we train a robotic agent to execute an object manipulation task in a simulation environment following the design of ManiSkill [**?**], but with the actuator being a dexterous hand. We validate that existing state-of-the-art RL algorithms could hardly achieve satisfactory results on this challenging task, while significant progress can be made through imitating the rich demonstrations in HOI4D. This shows the value HOI4D brings to the robot learning community. Below we present the experiment in detail.

D.1. Environment Setup

Inspired by ManiSkill [**?**], we build an environment based on SAPIEN [**?**] simulator.

Environment design: the environment use the SAPIEN simulator with timestep set to 0.05. The environment supports various interactions between the models presented in HOI4D and PartNet and the SAPIEN model of robot Adroit Hand. In this section, we choose toy car models for imitation learning research.

Task definition: we define a *Pick Up* task that requires the Adroit Hand to pick up a toy car on the table and leave it a certain height off the table. The task is successful if the toy car is close enough to the target location and is kept static for a period of time afterwards. The time limit for each episode is 200, and an episode will be evaluated as unsuccessful if it goes beyond the time limit.

Observation: the observation of the task is composed of three components: (i) joint angles and joint velocities of the Adroit Hand; (ii) global position and velocities of the Adroit Hand root; (iii) point cloud of the scene(4 dimensions: 3 for xyz position, 1 for segmentation that discriminates Adroit Hand and car). The Observation is suitable for studying category-level generalization. The point cloud is captured from eight cameras mounted on the table to provide a panoramic and object-centric view. To avoid losing too much visual information when the Adroit Hand is in contact with the object, four of these cameras are close to the object and 90° apart from each other, and the other four cameras are farther away from the object. In addition, we downsample the point cloud to 1200 points to increase training speed.

Action: the action space of the task is the motor command of the 30 joints of the Adroit Hand. We use PID controllers to control the joints of the Adroit Hand. We use velocity controllers for all the joints. During training, the range of the velocities is normalized to (-1, 1). The action space corresponds to the normalized target velocities of all controllers.

Reward: the reward function of the environment is set upon three stages: (i) the first-stage reward gives punishment to the distance between Adroit Hand's palm and the center of the toy car. It also contains a positive proportion of the dot product of the vector from the Adroit Hand to the object and the vector with palm orientation; (ii) the second-stage reward discourages the relative velocity between the Adroit Hand and the object. It is also negatively related to the total distance between finger tips and the object mesh; (iii) the final-stage reward is defined based on the height of the Adroit Hand and the object. The reward is positively correlated with these heights.

D.2. Demonstration Collection

Our demonstrations for imitation learning consists of observations that represent states of the Adroit Hand and the object being interacted with and actions that represent the motion of the Adroit Hand, which are called state-action demonstrations. We transform real hand-object interaction videos in HOI4D to the state-action demonstrations in SAPIEN [**?**], and the main challenges of this process are three-fold. First, since the manipulation tasks in the simulation environment only focus on the Adroit Hand and the object interacted with, the complicated scenarios in real videos may interfere with the imitation learning process and become noise. Second, the human hand and robot Adroit Hand are intrinsically different. Third, compared with the third-person view, egocentric videos in HOI4D suffer from more severe occlusion of both human hand and object, which brings more inaccurate human hand and object pose annotations for demonstration collection.

Inspired by DexMV [**?**], we divide the demonstration collection process into three steps named hand joint retargeting, state-only demonstration collection and state-action demonstration collection. We transform the human hand pose represented as 51 DoF MANO model [**?**] to 30 DoF Adroit Hand pose in the hand joint retargeting step. We then combine poses of Adroit Hand and object to generate state-only demonstrations, and finally compute the Adroit Hand action vectors to obtain the state-action demonstrations and use them as inputs of imitation learning.

Hand joint retargeting. Our hand joint retargeting method is inspired by DexMV [**?**]. Given the original 3D positions of human hand keypoints represented by MANO model [**?**], our goal is to find the optimum robot Adroit Hand pose that minimizes the distances between corresponding keypoints from human hand and Adroit Hand. We use the task space vectors [**?**, **?**] and accordingly define the objective function [**?**] of the optimization problem. The 15 task space vectors used in our method are designed based on DexPilot [**?**]: vectors between five fingertips and vectors from wrist to fingertips for both human hand and Adroit Hand.

State-only demonstration collection. While human hand-object interaction videos are all provided from simple scenes in HOI4D, a variety of backgrounds that are unrelated to the interaction process, such as tables and sofas with diverse geometries and positions, may increase the variance among videos and impede the learning of the hand-object interaction process. Thus we obtain the human hand and object poses using methods mentioned in the main paper, and then other information in the videos are all discarded. We use CAD models consistent with actual objects to build the bridge between real-world and simulation environments and place the CAD model and the Adroit Hand in corresponding positions and orientations according to the poses from videos. Figure 6 shows some examples of our state-only demonstrations.

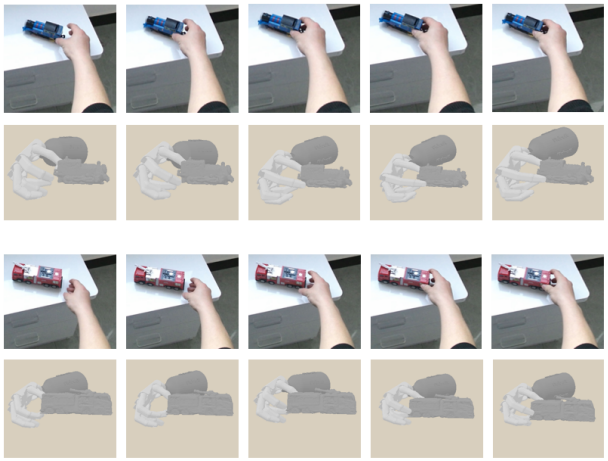

Figure 6. Examples of state-only demonstrations.

State-action demonstration collection. In our manipulation task settings, the action vector is directly obtained from the linear or angular velocity of each joint of the Adroit Hand. For each joint of the Adroit Hand. We fit the sequence of its angles with a cubic spline interpolation model in the time dimension to improve the smoothness of the motion trajectory, and the action can compute from the derivative of the fitting function. Combining the state-only demonstrations with the action vectors, we obtain the state-action demonstrations for imitation learning.

D.3. Imitation Learning Approaches

We perform imitation learning algorithms by augmenting the Reinforcement Learning algorithms using our demonstrations generated by the above methods. We compare such approaches with the RL algorithms. For RL algorithms, we adopt the commonly used Soft Actor-Critic(SAC) [**?**]. For imitation learning algorithms, we adopt the Generative Adversarial Imitation Learning(GAIL) [**?**], which is the SOTA IL method in robotic manipulation. For a fair comparison, we use SAC algorithm in the RL part of GAIL with the same hyper-parameters. (Other RL and IL algorithms will be studied in our future work).

We use point cloud-based vision architecture as our feature extractor since the input of our network contains point cloud. The point cloud features include position(3 dimensions: xyz)and segmentation masks(1 dimension for Adroit Hand)and we concatenate the robot state(joint angles and joint velocity of the Adroit Hand, global position and velocities of Adroit Hand root)to the features of each point.

For the RL algorithm, we parameterized the policy net-

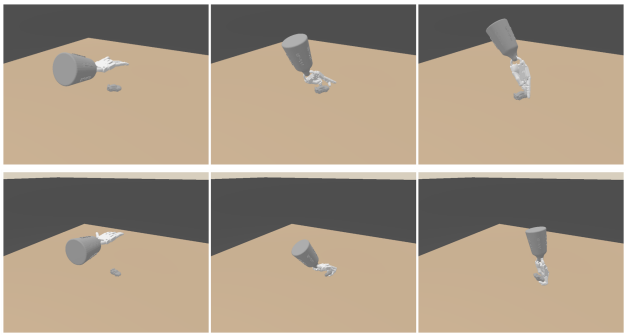

Figure 7. Comparison of RL agent and IL agent on the *Pick Up* task. **(Top Row)** RL agent. **(Bottom Row)** IL agent.

work and the value network with the *PointNet+Transformer* architecture that is the baseline backbone in ManiSkill [**?**].

For the IL algorithm, we use the same architecture in the policy network and value network as the RL algorithm. For the discriminator network in GAIL, we also use the *PointNet+Transformer* architecture. While the expert trajectories are crucial for GAIL, we carefully select 12 demonstrations with high quality from different toy cars as our expert trajectories.

D.4. Results and Analysis

We evaluate the RL and IL methods on the *Pick Up* task. In the experiments, the success rate is evaluated with three random seeds. The results is presented in Table 4.

Figure 7 shows a comparison of RL agent and IL agent on the *Pick Up* task.

Table 4 shows that the IL algorithm can outperform the RL algorithm. While category-level dexterous manipulation is still a challenging task in robotics, the success rate of RL and IL algorithms are both low. Since our input observation for RL and IL is the point cloud of the scene, it is harder for RL and IL agents to learn to pick up the toy car than the previous works that always use the ground truth states of the environment as input observation. We use only 12 demonstrations in IL that are generated from different objects in the toy car category. These demonstrations can greatly improve the success rate of the task, which shows that our demonstrations in HOI4D can greatly help RL learn better policy.

Method	Mean success rate (std)
RL	3.5 (3.2)
GAIL	17.4 (7.9)

Table 4. The average success rate of different methods.