TASTE-Rob: Advancing Video Generation of Task-Oriented Hand-Object Interaction for Generalizable Robotic Manipulation

Supplementary Material

8. Details of TASTE-Rob dataset

8.1. Comparisons between Ego4D and TASTE-Rob

Ego4D [16] differs from TASTE-Rob in two key aspects: the camera perspective and the recording methodology. **Camera Perspective**: Ego4D utilized seven distinct head-mounted cameras during data collection, resulting in diverse camera perspectives throughout the dataset. In contrast, a consistent camera perspective is essential for our task. As demonstrated in Figure 12, videos in Ego4D show perspective variations in the positioning of static objects, as evidenced by the blue box in the first case. **Recording Methodology**: In Ego4D's data collection process, participants were provided with cameras to record their daily activities. These recordings typically exceeded eight minutes in duration, and final short clips are extracted from them. Consequently, these clips often contain actions from preceding or subsequent activities, creating misalignment with the provided language instructions. This misalignment is exemplified in Figure 12 (second case, highlighted by red bounding boxes): the participant is mixing food on a plate, which contradicts the instruction "puts down the bowl".

8.2. Design of Language Instruction in TASTE-Rob

For object manipulation tasks, it is essential to establish a one-to-one correspondence between actions and language instructions. Specifically, each language instruction should uniquely identify both the target object to be manipulated and the specific action to be performed. However, Ego4D's [16] language instructions are insufficiently detailed to satisfy this criterion.

Identifying Unique Target Objects. The language instructions in Ego4D [16] only specify the object types without providing distinctive determiners. For instance, given the instruction "C picks up a bowl and puts down it", when multiple bowls are present on the table, any of them could potentially be considered as the target object. In TASTE-Rob, we incorporate distinctive determiners to uniquely identify objects, such as "red bowl", "bowl with a tomato", and "bowl next to the kettle". Figure 13 illustrates examples of video frames and their corresponding instructions from TASTE-Rob.

Identifying Unique Actions. Regarding object manipulation tasks, Ego4D's [16] language instructions merely describe the action without specifying the target location. For instance, the instruction "C puts down the bowl" indicates the action "puts down" but fails to specify the intended placement location. In TASTE-Rob, we provide comprehensive instructions that include detailed spatial information, such as "take the solid black pen and put it in the pen holder" and "take the orange eraser and put it on the pencil case".

Figure 11. **Distribution of objects in videos from six scenes.** The diverse range of objects across different scenarios demonstrate the richness of activities captured in TASTE-Rob

(a) Examples of HOI videos in Ego4D [16].

(b) Examples of HOI videos in TASTE-Rob.

Figure 12. **Comparison between Ego4D [16] and TASTE-Rob**. Ego4D videos lack fixed camera perspective and precise action-language alignment, as shown in the red boxes, while TASTE-Rob solve these limitations.

Figure 13. **Examples of video frame and corresponding language instruction.** In TASTE-Rob, we enhance the language instructions with more specific details to ensure accurate identification of unique target objects and their corresponding actions.

8.3. Diversity of Manipulated Objects in TASTE-Rob

The distribution of manipulated objects in TASTE-Rob across different scenarios is visualized through word clouds in Figure 11. These visualizations emphasize context-dependent objects, such as kitchen utensils (bowls, spoons, plates) in cooking activities and Clothing (skirts, shorts, hats) in bed environment. Additionally, some objects appear universally across scenarios, including cups, tissues, and mobile devices. The broad spectrum of objects encountered in TASTE-Rob reflects its comprehensive coverage of diverse activities.

9. Analyzing Details of hand poses

9.1. MANO parametric hand model

Given a RGB image of hands, HaMeR [34] is able to reconstruct 3D hand meshes by predicting MANO [39] parameters (pose parameters and shape parameters). Specifically, the pose parameters are decomposed into two components: global orientation and hand pose. Global orientation parameters are represented by $\theta^g \in \mathbb{R}^{1\times3\times3}$, defining the overall 3D orientation of the hand in space. Hand pose parameters control the articulation of 15 hand joints, which are represented by $\theta^p \in \mathbb{R}^{15\times3\times3}$, defining the full hand articulation. We perform a comprehensive analysis of hand pose distribution by examining both global orientation and hand pose parameters from 16 sampled frames across TASTE-Rob videos.

9.2. Calculating Details of analyzing Distribution

Orientation Distribution. To analyze palm orientation, we first extract the normal vector of the palm plane from θ^g. This normal vector is then projected onto the frame plane (X-Y plane), and we calculate the projected angle, which tells us how much the palm's direction deviates from the positive X-axis in the frame plane. Specifically, 0 degree points along the positive positive X-axis. We also normalize this projected angle to the 0-360 degree range. Finally, we divide the range into four 90-degree intervals to analyze the distribution of palm orientations across different directions.

Inter-finger Angles Distribution. To analyze the inter-finger angles, we calculate the angles between thumb-index and index-middle finger pairs. For each frame, we first extract the normalized direction vectors of fingers ($\boldsymbol{d}_t$ for thumb, $\boldsymbol{d}_i$ for index, and $\boldsymbol{d}_m$ for middle finger) using their joint positions. The inter-finger angles are computed as the inverse cosine of the dot product between corresponding vectors: $\arccos(\boldsymbol{d}_t \cdot \boldsymbol{d}_i)$ and $\arccos(\boldsymbol{d}_i \cdot \boldsymbol{d}_m)$. After obtaining these angles, we plot their distributions as probability density curves, and these curves illustrate how finger spreads are distributed in TASTE-Rob.

Finger curvature Distribution. To analyze finger curvature patterns, we compute the bending angle for each finger. Each finger has three joints: metacarpophalangeal (MCP) near the palm, proximal interphalangeal (PIP) in the middle, and distal interphalangeal (DIP) near the fingertip. We calculate the bending angle by obtaining two normalized direction vectors: one from MCP to PIP joint $\boldsymbol{j}_{M2P}$, and an-

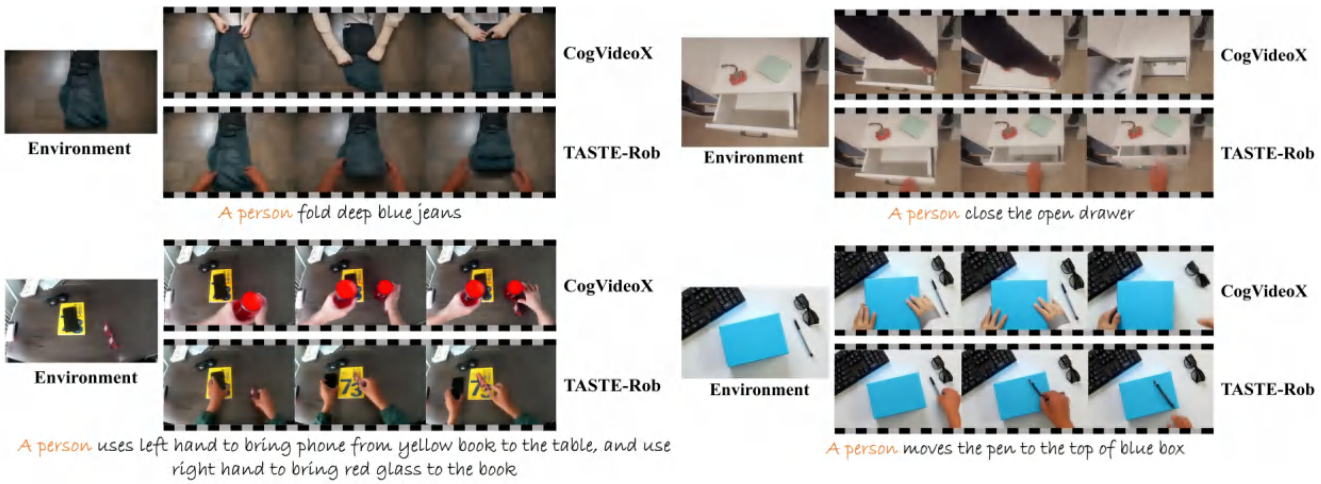

Figure 14. **More results of video generation comparison between TASTE-Rob and CogVideoX [60].**

other from PIP to DIP joint $\boldsymbol{j}_{P2D}$. The bending angle is then computed as the inverse cosine of the dot product between these vectors: $\arccos(\boldsymbol{j}_{M2P} \cdot \boldsymbol{j}_{P2D})$. We plot these angles' distributions as probability density curves, which reveal the common curvature patterns and articulation preferences of different finger joints in TASTE-Rob.

10. Our Introduced Metrics

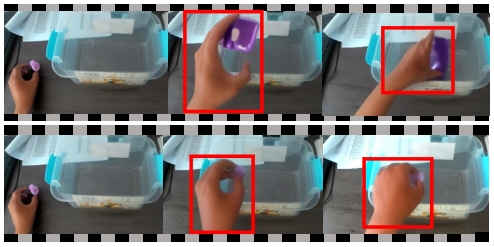

Figure 15. **Examples of generated videos and corresponding GPV value.** A higher GPV value indicates less consistency in grasping poses. .

Grasping Pose Variance. As illustrated in Section 9, we employ HaMeR [34] to predict accurate global orientation parameters θ^g and hand pose parameters θ^p. During grasping, variations in global orientation parameters are natural as hands need to move objects. However, hand pose parameters should remain consistent during grasping, as variations in grasping pose may lead to unstable grasps. To evaluate this consistency, we introduce Grasping Pose Variance (GPV), which computes the variance of hand pose parameters θ^p across frames containing grasping poses within a single video. Firstly, we utilize an off-the-shelf hand-object detector [40] to extract $L^{'}$ frames containing grasping poses from a video, where the hand-object detector can detect whether hands are interacting with objects. Then, we utilize HaMeR [34] to predict hand pose parameters $\{\theta_i^p\}_{i=1}^{L^{'}}$ of these frames, and calculate GPV as $\frac{1}{L^{'}}\sum_{i=1}^{N}(\theta_i^p - \bar{\theta^p})^2$, where $\bar{\theta^p} = \frac{1}{L^{'}}\sum_{i=1}^{N}\theta_i^p$. Besides, for videos containing both hands, we compute GPV values for each hand independently and take their mean value.

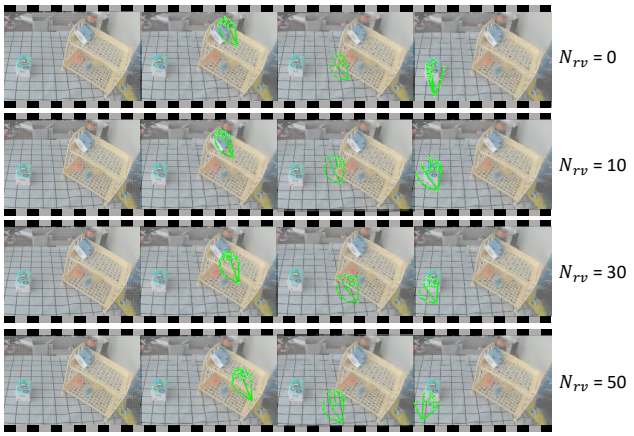

Figure 16. **Visualization of different denoising steps of pose refinement.** When $N_{rv} = 0$, no pose refinement is applied, whereas $N_{rv} = 50$ indicates that our learnable MDM generates pose sequences directly from Gaussian noise.

Visualization of Grasping Pose Variance. As illustrated in Figure 15, we compare the GPV values of two generated videos. The first case displays a video generated by our coarse action planner, while the second case presents a video generated by our complete pose-refinement pipeline. The results show that the second case maintains consistent grasping poses, whereas in the first case, the grasping poses vary significantly from pinching to holding, leading to a higher GPV value. In detail, each generated video consists of $L = 16$ frames, with the corresponding grasping

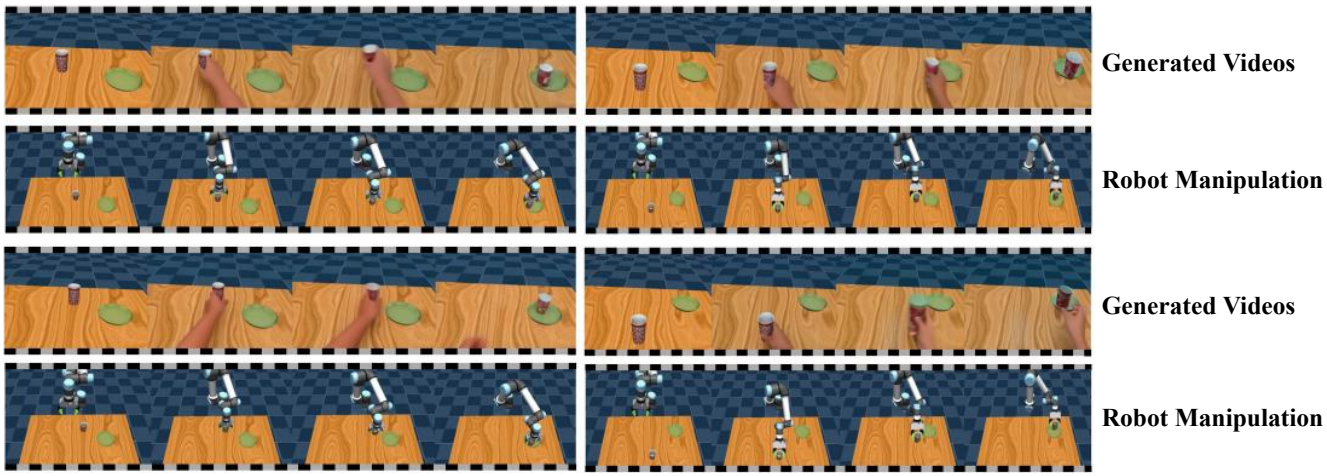

Figure 17. **Examples of video generation and robot manipulation in simulation platform.**

sequence length $L^{'}$ ranging from 6 to 8 frames. For visualization clarity, we only display a few frames in Figure 15.

Grasping Type Classification Error (GTCE). First, we extract 8 frames from both the ground truth and generated videos during the manipulation process. Then, in order to detect and classify grasping types, we use HaMeR [34] to predict the corresponding pose parameters for these frames. Finally, we classify hand poses into 17 grasping types as in [14] and evaluate the generated poses by calculating the classification error against the ground truth.

Hand Movement Direction Accuracy (HMDA). HMDA assesses the quality of generated poses by quantifying the disparity in hand movement directions between generated and ground truth videos. Initially, hand keypoints are extracted from each frame of both videos via MediaPipe [31]. Subsequently, movement vectors are derived by computing the differences between consecutive keypoints. The angle differences between these vectors are then calculated, which effectively measures the divergence in hand movement directions. Ultimately, the metric value is obtained by computing the average of these angle differences.

11. More Qualitative Results

Different Denoising Steps of Pose Refinement. As illustrated in Figure 16, we refine the inconsistent grasping poses while maintaining spatial awareness through a N_{rv}-step denoising process, starting from pose sequences extracted from our coarse generated videos ($N_{rv} = 0$). During inference, we set N_{rv} as 10.

Comparisons on Video Generation. As shown in Figure 14, TASTE-Rob is able to generate high-quality HOI videos on various environment and tasks. We also provide corresponding videos in the supplementary materials.

N_{rv}	Generation Quality		
	KVD↓	FVD↓	PIC↑
0	0.04	10.85	0.88
10	**0.03**	**9.43**	**0.90**
20	0.03	9.97	0.89
30	0.04	10.04	0.90
40	0.04	10.70	0.87
50	0.06	11.27	0.88

Table 6. **Quantitative comparison among different N_{rv}.**

Results on Robot Manipulation. As shown in Figure 17, TASTE-Rob is able to generate high-quality HOI videos on robot simulation platform and achieve accurate robot manipulation.

12. More Implementation Details

12.1. Stage I: A Coarse Action Planner

In Stage I, we fine-tune DynamiCrafter [54] on TASTE-Rob from its released checkpoint at 512 resolution. The fine-tuning process is conducted with a total batch size of 16 and a learning rate of 5×10^{-5} for 30K steps. For inference, we use a 50-step DDIM sampler and adopt the same multi-condition classifier-free guidance as in DynamiCrafter [54].

Video Sampling Strategy. Before fine-tuning, existing general I2V latent diffusion models [49, 54, 60, 63] obtain video frames $\boldsymbol{v}$ by sampling from a short clip within a longer video, with limited focus on maintaining action integrity in the generated videos. However, our HOI video generation requires strict action integrity, as it directly

impacts the effectiveness of robot manipulation learning. Therefore, we sample frames from the entire video to ensure complete action preservation. For clarity, we denote $\boldsymbol{v}$ as $\boldsymbol{v}^{0:L-1}$, and $\boldsymbol{v}^i \in \mathbb{R}^{3\times H\times W}$ as the i-th frame of $\boldsymbol{v}$. In our work, we set $\boldsymbol{v}^0$ as the start frame of the entire video, $\boldsymbol{v}^{L-1}$ as the end frame, and obtain the intermediate frames $\{\boldsymbol{v}^1, \boldsymbol{v}^2, ..., \boldsymbol{v}^{L-2}\}$ through uniform sampling across the entire video, ensuring continuity and completeness in action representation. Specifically, we set L as 16, and generate videos consisting of 16 frames.

12.2. Stage II: Revising Hand Pose Sequences

In Stage II, we train our learnable MDM [45] on TASTE-Rob. We set the 60-frame hand pose key-points normalized coordinates (extracted from videos by MediaPipe [31]) as motion parameters and train MDM to generate these coordinates conditioned on environment images and language instructions. The training is conducted with a batch size of 64 and a learning rate of 1×10^{-4} for 500K steps. During inference, MDM employs a 50-step DDIM sampler when generating from Gaussian noise. For pose refinement, we apply a 10-step denoising process from the extracted coarse hand pose sequences and adopt the same classifier-free guidance as in [45].

12.3. Stage III: Regenerating with Refined Pose

In Stage III, we fine-tune our learnable frame-wise adapter from Stable Diffusion 2. The training is conducted with a batch size of 32 and a learning rate of 5×10^{-5} for 30K steps. For training, the frame-wise adapter take the 16-frame visualized hand pose images as input. Specifically, we extract the 16-frame hand pose key-point coordinates and then draw their single-channel visualized images. For inference, we first draw visualized images from the refined hand pose sequences. Then, we generate videos conditioned on these visualized images, environment images and language instructions, applying a 50-step denoising process and the same multi-condition classifier-free guidance as ToonCrafter [55].