Real2Edit2Real: Generating Robotic Demonstrations via a 3D Control Interface

Supplementary Material

Method	Source		Generation		
	No Simulation	RGB Only	VLA Compatible	Novel Texture	Novel Trajectory
MimicGen [10]	✗	✗	✓	✓	✓
SkillMimicGen [3]	✗	✗	✓	✓	✓
RoboSplat [18]	✓	✗	✓	✓	✓
Real2Render2Real [19]	✓	✗	✓	✓	✓
DemoGen [17]	✓	✗	✗	✗	✓
R2RGen [16]	✓	✗	✗	✗	✓
UMIGen [5]	✓	✗	✗	✗	✓
RoboTransfer [8]	✓	✓	✓	✓	✗
MVAug [13]	✓	✓	✓	✓	✗
Real2Edit2Real (ours)	✓	✓	✓	✓	✓

Table 1. Comparison with Other One-to-many Demonstration Generation Methods.

1. Contribution Clarification

To better clarify our contribution, we provide a detailed comparison between our method and other one-to-many demonstration generation approaches, as shown in Table 1. Simulation-based methods like MimicGen [10] and SkillMimicGen [3] rely on simulators and require scene and object assets, which not only introduce a significant sim-to-real gap but also make it difficult to perform data augmentation directly on real-world data. Methods such as RoboSplat [18] and Real2Render2Real [19] are built on 3D Gaussian Splatting. Although they do not require a simulation engine, they still rely on dense scanning to reconstruct the objects or scenes. This means that they cannot perform data generation using only the RGB observations from the original demonstrations, which significantly limits their scalability. Another line of research, including DemoGen [17], R2RGen [16], and UMIGen [5] , generates new 3D point-cloud demonstrations through point-cloud editing. However, their reliance on depth sensors limits their compatibility with the current mainstream VLA paradigm that uses multi-view RGB inputs, and also prevents them from performing texture-level augmentation. Methods based on video generation, such as RoboTransfer [8] and MVAug [13], can directly augment multi-view 2D demonstrations, but they only enhance visual aspects such as texture, without increasing the diversity of object spatial distributions or robot trajectories.

In contrast, our method requires no simulator and directly augments the original RGB observations, significantly improving scalability. It simultaneously generates new textures and trajectories for VLA training, highlighting its unified and flexible design.

2. Real2Edit2Real Implementation Details

In this section, we provide more details of the proposed framework, Real2Edit2Real:

- In Section 2.1, we provide additional information for the hybrid training paradigm.
- In Section 2.2, we explain the full pipeline of depth-reliable spatial editing in detail.
- In Section 2.3, we discuss more about 3D-controlled video generation model.

2.1. Metric-scale Geometry Reconstruction

Data Visualization. Fig. 1 shows the visualization of the training data. We can see that real-world depth maps are often noisy and contain large invalid regions, whereas synthetic depth is clean and accurate. By training with our proposed hybrid training paradigm, our model learns to reconstruct geometry in metric scale in the real world, effectively compensating for the limitations of depth sensors.

Training Details. In Table 2, we provide the details of fine-tuning VGGT [14] to Metric-VGGT.

2.2. Depth-reliable Spatial Editing

Background Depth Completion. As we mentioned in the manuscript, projecting edited point clouds to depth maps may cause missing regions in the background due to the object moving and novel robot motion. To mitigate this artifact, we first inpaint the background, which deletes the foreground objects and robot in the multi-view first frames with an image-edit model [15]. Figure 2 provides the prompt we

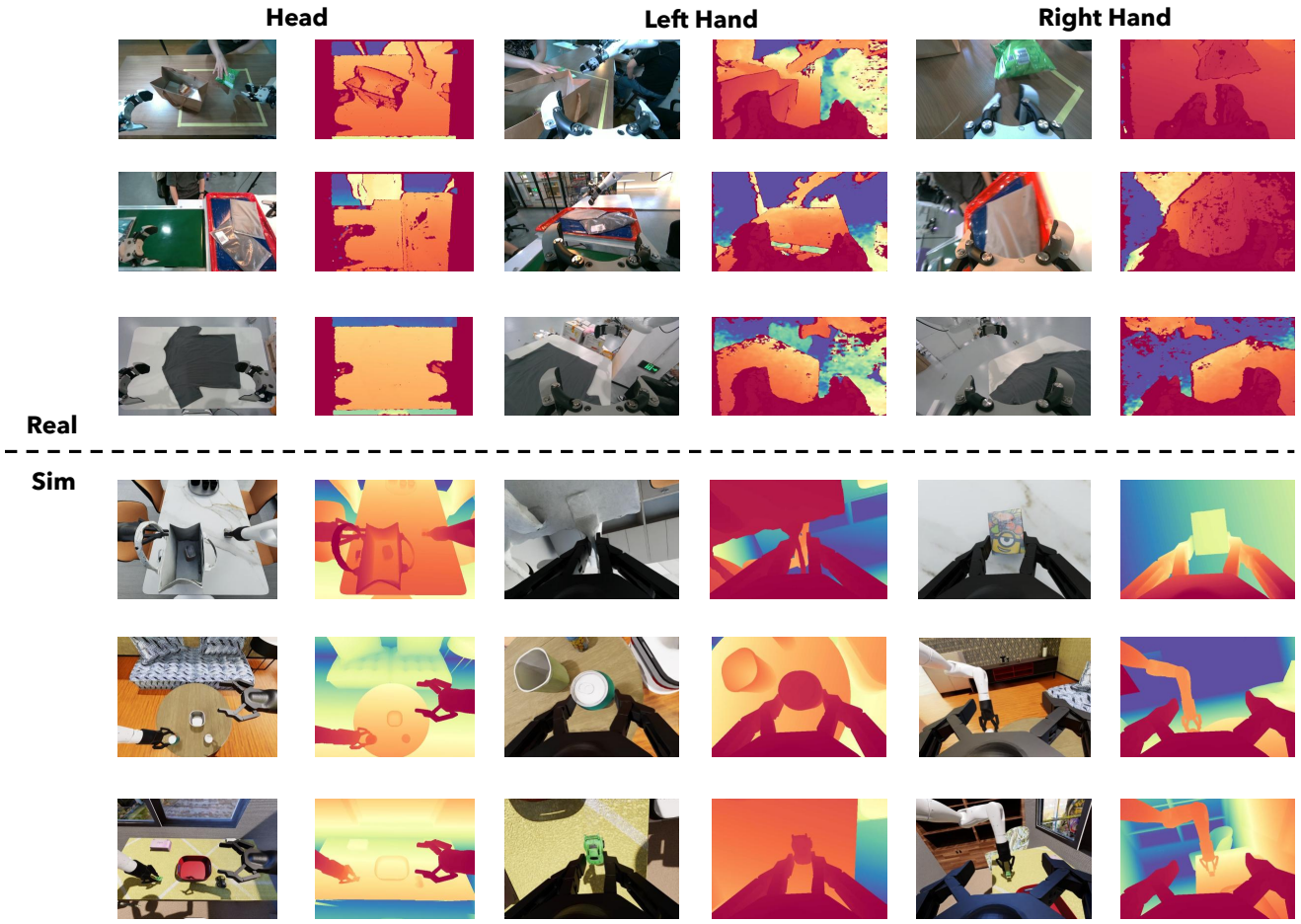

Figure 1. Training Data Visualization of Metric-VGGT. Depth visualization: red is the nearest, blue is the farthest.

```
"While keeping everything else in the image
    unchanged, remove the black gripper and the
    black wire."
"While keeping everything else in the image
    unchanged, remove the white robotic arm."
"While keeping everything else in the image
    unchanged, remove xxx on the table."
```

Figure 2. Prompt Used for Background Inpainting. In the prompt, xxx means the manipulated objects.

Figure 3. Example of the Inpainted Background.

used for image editing, and Figure 3 shows an example of the inpainted background. Then, we reconstruct the metric geometry of the background with Metric-VGGT. To correct the metric-scale inconsistencies introduced by image editing, we incorporate an additional point cloud alignment procedure, as shown in Algorithm 1.

Spatial Editing Pipeline. After getting a completed background point cloud, we separate foreground objects through Grounded-SAM [6, 9, 11] and robotic dual arms through forward kinematics. Following, we provide an example to detail the full spatial editing pipeline. Algorithm 2, 3 shows the spatial editing pipeline of the Mug to Basket task. The Object Relocation Segment produces the depth sequence for Smooth Object Relocation described in the manuscript.

Config	Value
Base Model	VGGT-1B
Training Real Data	100,000
Training Sim Data	40,000
Fine-Tuning Scheme	Full Parameter
Total Training Steps	150,000
Learning Rate	2e-4
Backbone Learning Rate	2e-5
LR Scheduler	Cosine Annealing Scheduler
ETA Minimum	1e-6
Weight Decay	1e-2
View Num	3
Global Batch Size	16
Gradient Accumulation Steps	4
Mixed Precision	bf16
Optimizer	AdamW
Training Image Size	518

Table 2. Training Details of Metric-VGGT.

Algorithm 1 Background Point-Cloud Alignment

Require: Origin first frame point cloud $\mathcal{P}^o$, unaligned background point cloud $\mathcal{P}^{edit}$, table mask M^{table}.
Ensure: Metric-aligned background point cloud $\mathcal{P}^{bg}$.
function ESTIMATEPLANE($\mathcal{P}$)
$plane \leftarrow$ RansacPlaneSegment($\mathcal{P}$)
// $plane : [a, b, c, d], ax + by + cz + d = 0$
return $plane$
end function
$plane^o \leftarrow$ EstimatePlane($\mathcal{P}^o[M^{table}]$)
$plane^{edit} \leftarrow$ EstimatePlane($\mathcal{P}^{edit}[M^{table}]$)
$scale \leftarrow plane^o[3]/plane^{edit}[3]$
$\mathcal{P}^{bg} \leftarrow scale \times \mathcal{P}^{edit}$
return $\mathcal{P}^{bg}$

2.3. 3D-Controlled Video Generation

Training Data. For training the 3D-controlled multi-view video generation model, we sample 7K episodes of 64 tasks from the Agibot-World datasets [1]. To get the control conditions of the training data, we used the Metric-VGGT to predict the depth maps and compute the Canny Edges from depth. To ensure the 3D control condition remains consistent across multi-view and temporal, we perform global normalization on the depth sequences of all three views within a training chunk, rather than normalizing each depth map individually.

Condition Dropout. In the training stage, we fine-tune the backbone of GE-Sim [7] (based on Cosmos-Predict-2B [20]) with sampling data from the Agibot-World Dataset [1]. In multi-condition compositional generation, intensity-based conditions such as depth maps and Canny

Algorithm 2 Pipeline of Mug to Basket

Require: Source point clouds $\mathcal{P}^l$, $\mathcal{P}^r$, $\mathcal{P}^{mug}$, $\mathcal{P}^{basket}$, background point cloud $\mathcal{P}^{bg}$, joint states $\mathcal{Q}$, action trajectory $\mathcal{A}$, camera poses $\mathcal{T}^h$, $\mathcal{T}^l$, $\mathcal{T}^r$, skill-1 start timestep t_1, skill-1 end timestep t_2, skill-2 start timestep t_3, skill-2 end timestep t_4.
Ensure: Novel depth sequence $\mathcal{D}^{h\star}$, $\mathcal{D}^{l\star}$, $\mathcal{D}^{r\star}$, joint states $\mathcal{Q}^\star$, action trajectory $\mathcal{A}^\star$, camera poses $\mathcal{T}^\star$.
function RENDERDEPTH($\mathcal{P}, \mathcal{T}, \mathcal{Q}$)
$D_1 \leftarrow$ ProjectPointCloud($\mathcal{P}, \mathcal{T}$)
$D_2 \leftarrow$ RenderLinkDepth($\mathcal{Q}, \mathcal{T}$)
return Merge(D_1, D_2)
end function
Sample a Object Transform Pair $\mathbf{T}^{mug}, \mathbf{T}^{basket} \in \mathbb{R}^{4\times4}$
$\mathcal{D}^{h\star} \leftarrow list(), \mathcal{D}^{l\star} \leftarrow list(), \mathcal{D}^{r\star} \leftarrow list()$
$\mathcal{Q}^\star \leftarrow list(), \mathcal{A}^\star \leftarrow list(), \mathcal{T}^\star \leftarrow list()$
// Object Relocation Segment
for t in range(0, 30) **do**
$\mathbf{T}_t^{mug}, \mathbf{T}_t^{basket} \leftarrow$ Interpolate($\mathbf{T}^{mug}, \mathbf{T}^{basket}, 30, t$)
$\mathcal{P}_t^\star \leftarrow \mathbf{T}_t^{mug}\mathcal{P}_0^{mug} \cup \mathbf{T}_t^{basket}\mathcal{P}_0^{basket} \cup \mathcal{P}_0^l \cup \mathcal{P}_0^r \cup \mathcal{P}^{bg}$
$\mathcal{D}^{h*} \leftarrow \mathcal{D}^{h*} \cup$ RenderDepth($\mathcal{P}_t^\star, \mathcal{T}_0^h, \mathcal{Q}_0$)
$\mathcal{D}^{l*} \leftarrow \mathcal{D}^{l*} \cup$ RenderDepth($\mathcal{P}_t^\star, \mathcal{T}_0^l, \mathcal{Q}_0$)
$\mathcal{D}^{r*} \leftarrow \mathcal{D}^{r*} \cup$ RenderDepth($\mathcal{P}_t^\star, \mathcal{T}_0^r, \mathcal{Q}_0$)
$\mathcal{Q}^\star \leftarrow \mathcal{Q}^\star \cup \mathcal{Q}_0, \mathcal{A}^\star \leftarrow \mathcal{A}^\star \cup \mathcal{A}_0$
$\mathcal{T}^\star \leftarrow \mathcal{T}^\star \cup (\mathcal{T}_0^h, \mathcal{T}_0^l, \mathcal{T}_0^r)$
end for
// Motion-1 Segment
$\mathcal{A}_{start}^\star \leftarrow \mathcal{A}_0, \mathcal{A}_{end}^\star \leftarrow \mathbf{T}^{mug}A_{t_1},$
for t in range(0, t_1) **do**
$\mathbf{T}_t, \mathcal{A}_t^\star, \mathcal{Q}_t^\star \leftarrow$ MotionPlan($\mathcal{A}_{start}^\star, \mathcal{A}_{end}^\star, t$)
$\mathcal{P}_t^{ree} \leftarrow \mathcal{P}_t^r \setminus$ FK($\mathcal{P}_t^r, \mathcal{Q}_t$)
$\mathcal{P}_t^\star \leftarrow \mathbf{T}_t\mathcal{P}_t^{ree} \cup \mathcal{P}_t^l \cup \mathbf{T}^{mug}\mathcal{P}_t^{mug} \cup \mathbf{T}^{basket}\mathcal{P}_t^{basket} \cup \mathcal{P}_{bg}$
$\mathcal{D}^{h\star} \leftarrow \mathcal{D}^{h\star} \cup$ RenderDepth($\mathcal{P}_t^\star, \mathcal{T}_t^h, \mathcal{Q}_t^\star$)
$\mathcal{D}^{l\star} \leftarrow \mathcal{D}^{l\star} \cup$ RenderDepth($\mathcal{P}_t^\star, \mathcal{T}_t^l, \mathcal{Q}_t^\star$)
$\mathcal{D}^{r\star} \leftarrow \mathcal{D}^{r\star} \cup$ RenderDepth($\mathcal{P}_t^\star, \mathbf{T}_t\mathcal{T}_t^r, \mathcal{Q}_t^\star$)
$\mathcal{Q}^\star \leftarrow \mathcal{Q}^\star \cup \mathcal{Q}_t^\star, \mathcal{A}^\star \leftarrow \mathcal{A}^\star \cup \mathcal{A}_t^\star$
$\mathcal{T}^\star \leftarrow \mathcal{T}^\star \cup (\mathcal{T}_t^h, \mathcal{T}_t^l, \mathbf{T}_t\mathcal{T}_t^r)$
end for
// Skill-1 Segment
for t in range(t_1, t_2) **do**
$\mathcal{Q}_t^\star \leftarrow$ IK($\mathbf{T}^{mug}\mathcal{A}_t$)
$\mathcal{P}_t^{ree} \leftarrow \mathcal{P}_t^r \setminus$ FK($\mathcal{P}_t^r, \mathcal{Q}_t$)
$\mathcal{P}_t^\star \leftarrow \mathbf{T}^{mug}(\mathcal{P}_t^{ree} \cup \mathcal{P}_t^{mug}) \cup \mathcal{P}_t^l \cup \mathbf{T}^{mug}\mathcal{P}_t^{mug} \cup \mathcal{P}_{bg}$
$\mathcal{D}^{h\star} \leftarrow \mathcal{D}^{h\star} \cup$ RenderDepth($\mathcal{P}_t^\star, \mathcal{T}_t^h, \mathcal{Q}_t^\star$)
$\mathcal{D}^{l\star} \leftarrow \mathcal{D}^{l\star} \cup$ RenderDepth($\mathcal{P}_t^\star, \mathcal{T}_t^l, \mathcal{Q}_t^\star$)
$\mathcal{D}^{r\star} \leftarrow \mathcal{D}^{r\star} \cup$ RenderDepth($\mathcal{P}_t^\star, \mathbf{T}^{mug}\mathcal{T}_t^r, \mathcal{Q}_t^\star$)
$\mathcal{Q}^\star \leftarrow \mathcal{Q}^\star \cup \mathcal{Q}_t^\star, \mathcal{A}^\star \leftarrow \mathcal{A}^\star \cup \mathbf{T}^{mug}\mathcal{A}_t$
$\mathcal{T}^\star \leftarrow \mathcal{T}^\star \cup (\mathcal{T}_t^h, \mathcal{T}_t^l, \mathbf{T}^{mug}\mathcal{T}_t^r)$
end for

Algorithm 3 Continued to Pipeline of Mug to Basket

// Motion-2 Segment
$\mathcal{A}^{\star}_{start} \leftarrow \mathbf{T}^{mug}\mathcal{A}_{t_2}$, $\mathcal{A}^{\star}_{end} \leftarrow \mathbf{T}^{basket}A_{t_3}$,
for t in range(t_2, t_3) **do**
  $\mathbf{T}_t, \mathcal{A}^{\star}_t, \mathcal{Q}^{\star}_t \leftarrow \text{MotionPlan}(\mathcal{A}^{\star}_{start}, \mathcal{A}^{\star}_{end}, t)$
  $\mathcal{P}^{ree}_t \leftarrow \mathcal{P}^{r}_t \setminus \text{FK}(\mathcal{P}^{r}_t, \mathcal{Q}_t)$
  $\mathcal{P}^{\star}_t \leftarrow \mathbf{T}_t(\mathcal{P}^{ree}_t \cup \mathcal{P}^{mug}_t) \cup \mathcal{P}^{l}_t \cup \mathbf{T}^{basket}\mathcal{P}^{basket}_t \cup \mathcal{P}_{bg}$
  $\mathcal{D}^{h\star} \leftarrow \mathcal{D}^{h\star} \cup \text{RenderDepth}(\mathcal{P}^{\star}_t, \mathcal{T}^{h}_t, \mathcal{Q}^{\star}_t)$
  $\mathcal{D}^{l\star} \leftarrow \mathcal{D}^{l\star} \cup \text{RenderDepth}(\mathcal{P}^{\star}_t, \mathcal{T}^{l}_t, \mathcal{Q}^{\star}_t)$
  $\mathcal{D}^{r\star} \leftarrow \mathcal{D}^{r\star} \cup \text{RenderDepth}(\mathcal{P}^{\star}_t, \mathbf{T}_t\mathcal{T}^{r}_t, \mathcal{Q}^{\star}_t)$
  $\mathcal{Q}^{\star} \leftarrow \mathcal{Q}^{\star} \cup \mathcal{Q}^{\star}_t$, $\mathcal{A}^{\star} \leftarrow \mathcal{A}^{\star} \cup \mathcal{A}^{\star}_t$
  $\mathcal{T}^{\star} \leftarrow \mathcal{T}^{\star} \cup (\mathcal{T}^{h}_t, \mathcal{T}^{l}_t, \mathbf{T}_t\mathcal{T}^{r}_t)$
end for
// Skill-2 Segment
for t in range(t_3, t_4) **do**
  $\mathcal{Q}^{\star}_t \leftarrow \text{IK}(\mathbf{T}^{basket}\mathcal{A}_t)$
  $\mathcal{P}^{ree}_t \leftarrow \mathcal{P}^{r}_t \setminus \text{FK}(\mathcal{P}^{r}_t, \mathcal{Q}_t)$
  $\mathcal{P}^{\star}_t \leftarrow \mathbf{T}^{basket}(\mathcal{P}^{ree}_t \cup \mathcal{P}^{mug}_t \cup \mathcal{P}^{basket}_t) \cup \mathcal{P}^{l}_t \cup \mathcal{P}_{bg}$
  $\mathcal{D}^{h\star} \leftarrow \mathcal{D}^{h\star} \cup \text{RenderDepth}(\mathcal{P}^{\star}_t, \mathcal{T}^{h}_t, \mathcal{Q}^{\star}_t)$
  $\mathcal{D}^{l\star} \leftarrow \mathcal{D}^{l\star} \cup \text{RenderDepth}(\mathcal{P}^{\star}_t, \mathcal{T}^{l}_t, \mathcal{Q}^{\star}_t)$
  $\mathcal{D}^{r\star} \leftarrow \mathcal{D}^{r\star} \cup \text{RenderDepth}(\mathcal{P}^{\star}_t, \mathbf{T}^{basket}\mathcal{T}^{r}_t, \mathcal{Q}^{\star}_t)$
  $\mathcal{Q}^{\star} \leftarrow \mathcal{Q}^{\star} \cup \mathcal{Q}^{\star}_t$, $\mathcal{A}^{\star} \leftarrow \mathcal{A}^{\star} \cup \mathbf{T}^{basket}\mathcal{A}_t$
  $\mathcal{T}^{\star} \leftarrow \mathcal{T}^{\star} \cup (\mathcal{T}^{h}_t, \mathcal{T}^{l}_t, \mathbf{T}^{baseket}\mathcal{T}^{r}_t)$
end for
return $\mathcal{D}^{\star}$, $\mathcal{Q}^{\star}$, $\mathcal{A}^{\star}$, $\mathcal{T}^{\star}$

edges tend to dominate the visual information, potentially diminishing the influence of other control signals during training [4]. However, these two conditions always introduce noise after spatial editing. To improve robustness against imperfect control signals, we apply random dropout to the depth and Canny edge conditions during training, where they are independently dropped with a probability of 0.5, and jointly dropped with a probability of 0.1. By randomly masking portions of these inputs, the model is encouraged to rely on complementary visual evidence, rather than depending solely on the intensity conditions, ultimately improving the realism of the generated videos under noisy conditions.

Training Details. In Table 3, we provide the details of training the 3D-controlled video generation model.

Config	Value
Base Model	GE-Sim-2B
Training Data	7000 Episodes 64 Tasks
Fine-Tuning Scheme	Full Parameter
Total Training Steps	20,000
Learning Rate	1e-4
LR Scheduler	Constant with Warmup
LR Warmup Steps	1000
Weight Decay	5e-5
Global Batch Size	16
Gradient Accumulation Steps	1
Max Gradient Norm	1.0
Mixed Precision	bf16
Optimizer	AdamW
Training Resolution	384×512
Video Chunk Length	25
Memory Frames	4

Table 3. Training Details of 3D-controlled Video Generation Model.

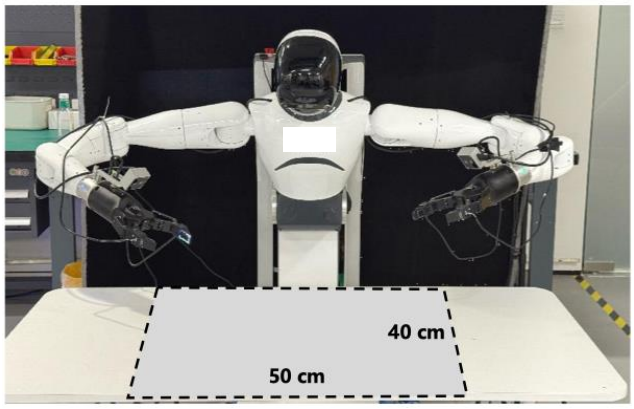

Figure 4. Visualization of Manipulation Workspace.

3. Experiment Details

Workspace. Figure 4 shows the workspace of four real-robot manipulation tasks in the manuscript. The workspace is determined by the maximal range in which the robot's kinematic configuration can perform the intended tasks.

Object Set. Figure 5 shows all the objects we used in the manipulation tasks. Because it is impractical to verify every object in the training set, all objects in the figure are newly purchased to minimize any potential overlap with the dataset. This setup enables a more reliable assessment of our framework's generalization to unseen objects. In addition, the real-world testing laboratory is also absent from the training data used to develop our framework.

Data Generation. To generate demonstrations, we first reconstruct the source demonstrations and apply a confidence threshold between 30% and 50% to remove spurious points. During spatial editing, we define an augmentation region around the object's original location, typically a 40cm×40cm square, and augment object rotations within a 30°–60° range. For video generation, we set the diffusion step to 6 and the memory length to 4, for which we uniformly sample from the already generated frames, including both the first and last frames.

Figure 5. Visualization of Manipulation Objects.

4. Additional Experiments

4.1. Quantitative Results of Video Generation

To evaluate the performance of our video generation model, we conduct experiments using data collected from the four real-world robot tasks to prevent data contamination. Table 4 shows the quantitative results compared to GE-Sim [7] with the conditional I2V setting. It demonstrates that our video generation module produces robot demonstrations with significantly enhanced visual realism.

Method	FVD ↓	LPIPS ↓	SSIM ↑	PSNR ↑
GE-Sim [7]	663.4	0.2038	0.7491	20.41
Ours	**352.9**	**0.1252**	**0.8647**	**22.95**

Table 4. Quantitative Results of Video Generation. We compare our method with GE-Sim across several standard metrics on conditional I2V. Bold numbers indicate the best performance.

4.2. Diffusion Policy on Mug to Basket

To further validate the quality of the data generated by Real2Edit2Real, we conduct additional Diffusion Policy [2] experiments on the Mug to Basket task. We use a ViT-S encoder initialized with DINO-v3 [12] weights and train the Diffusion Policy in a full-parameter manner on different training data. Table 5 shows the success rate of diffusion polices trained with real demonstrations and generated demonstrations, which indicates that generating data from only a few source demonstrations, like 1-5, can make DP surpass that trained with 50 real demonstrations on this task.

4.3. Generation Time Analysis

Figure 6 presents the generation time analysis of the Real2Edit2Real framework. While the integration of video generation modules introduces a computational bottleneck at lower GPU counts, our approach exhibits parallel scalability. By leveraging multi-GPU acceleration, the data generation time is significantly reduced, allowing Real2Edit2Real to surpass the success rates of manual teleoperation in a short time.

R10	R20	R50	R1G200	R2G200	R5G200
0/20	9/20	11/20	13/20	14/20	17/20

Table 5. Performance of Diffusion Policy on the Mug to Basket task. R means the number of real demonstrations, and G means the number of generated demonstrations.

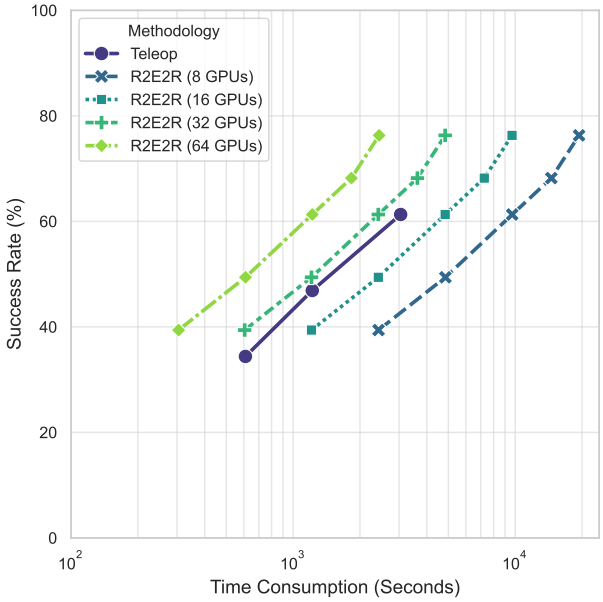

Figure 6. Time Analysis of Real2Edit2Real. We report the success rate (%) relative to the total time consumption (seconds, log-scale) for data generation.

4.4. Generation Data Scaling Analysis

In the manuscript, we investigate how increasing the number of source demonstrations affects policy performances. Here, we additionally examine, within our proposed Real2Edit2Real framework, the impact of generating more demonstrations. To this end, we produce varying numbers of demonstrations from a single source demonstration and evaluate the resulting policies, using the same training and evaluation protocols as in the manuscript. Experimental results are shown in Figure 7 and Table 6. The results indicate: (1) Both policies exhibit consistently improved success rates when scaling up generated demonstrations. (2) When we generate more than 300 demonstrations from only one demo, the average success rates surpass that of 50 real demonstrations.

# Demo	Mug to Basket		Pour Water		Lift Box		Scan Barcode		Total	
	Go-1	$\pi_{0.5}$	Go-1	$\pi_{0.5}$	Go-1	$\pi_{0.5}$	Go-1	$\pi_{0.5}$	Go-1	$\pi_{0.5}$
Real 10	8 / 20	8 / 20	5 / 20	1 / 20	11 / 20	13 / 20	5 / 20	4 / 20	36.3%	32.5%
Real 20	12 / 20	14 / 20	7 / 20	2 / 20	12 / 20	15 / 20	8 / 20	5 / 20	48.8%	45.0%
Real 50	14 / 20	13 / 20	8 / 20	8 / 20	15 / 20	17 / 20	12 / 20	11 / 20	61.3%	61.3%
Real 1 Gen 50	8 / 20	9 / 20	11 / 20	4 / 20	10 / 20	6 / 20	11 / 20	4 / 20	50.0%	28.8%
Real 1 Gen 100	12 / 20	11 / 20	12 / 20	6 / 20	10 / 20	7 / 20	12 / 20	9 / 20	57.5%	41.3%
Real 1 Gen 200	14 / 20	15 / 20	12 / 20	10 / 20	12 / 20	10 / 20	14 / 20	11 / 20	65.0%	57.5%
Real 1 Gen 300	15 / 20	16 / 20	12 / 20	12 / 20	14 / 20	11 / 20	18 / 20	11 / 20	73.8%	62.5%
Real 1 Gen 400	15 / 20	19 / 20	14 / 20	15 / 20	15 / 20	13 / 20	18 / 20	13 / 20	77.5%	75.0%

Table 6. Scaling Analysis of Generated Demonstrations. This compares the performance of polices trained with different numbers of demonstrations generated from only one source demonstration. We can see that increasing the number of generated demonstrations leads to improved success rates for both policies. When we generate more than 300 demonstrations from only one demo, the average success rates even surpass that of 50 real demonstrations.

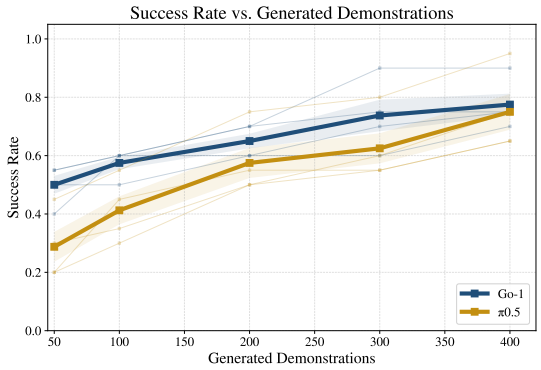

Figure 7. Scaling Analysis of Generated Demonstrations. Bold curves denote the task-averaged performance, while the faint translucent curves visualize the trajectories of individual tasks. Both policies exhibit consistently improved success rates when scaling up generated demonstrations.

4.5. Ablation Study of Control Conditions

In the manuscript, we introduced our 3D-controlled video generation model, which uses depth as the 3D control interface and incorporates Canny edges computed from the depth map as an auxiliary condition. To investigate the roles of depth and Canny edges in video generation, we conduct qualitative ablation studies by removing each condition individually. Fig. 9, 10, 11, 12 show the results on four tasks, respectively. The results demonstrate that removing either the depth control or the Canny edge constraint leads to issues such as object blurring and incorrect interactions, which substantially degrade the quality of the generated demonstrations.

5. Additional Visualizations

Figure 13, 14, 15, 16 show more visualizations of the four real-world manipulation tasks.

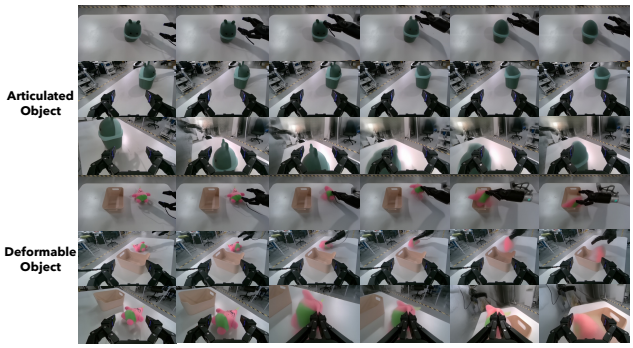

Figure 8. Visualization of Failure Cases.

6. Limitation and Discussion

Despite the advantages of our proposed Real2Edit2Real framework, which enables scalable multi-view demonstration augmentation, it still has certain limitations.

Video Generation Time. As analyzed in Section 4.3, the video generation module currently poses a computational bottleneck within our framework, particularly in resource-constrained scenarios. Future research could explore the integration of acceleration techniques from the generative modeling community, such as KV caching and model distillation, to further enhance the throughput of our data generation pipeline.

Object Generalization. As illustrated in Figure 8, our generative model exhibits limitations in object generalization, particularly when handling articulated or deformable objects. This stems primarily from the lack of these object categories in our training distribution, which can lead to visual artifacts such as motion blurring or structural inconsistency during video synthesis. To mitigate this, future work could focus on scaling up the diversity and volume of training data to enhance the model's robustness across a broader spectrum of object geometries and physical properties.

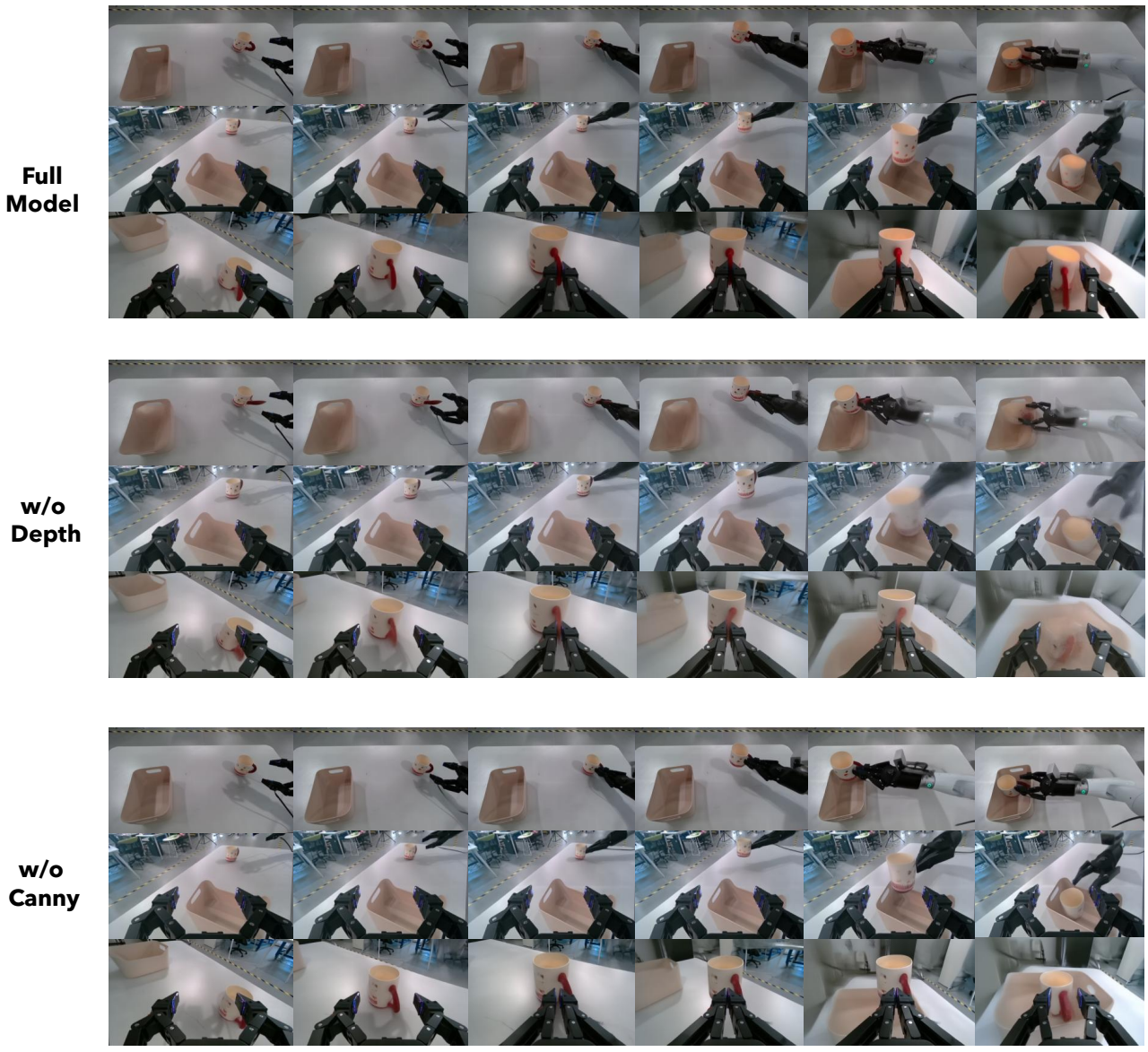

Figure 9. Ablation Study of Control Conditions on Mug to Basket.

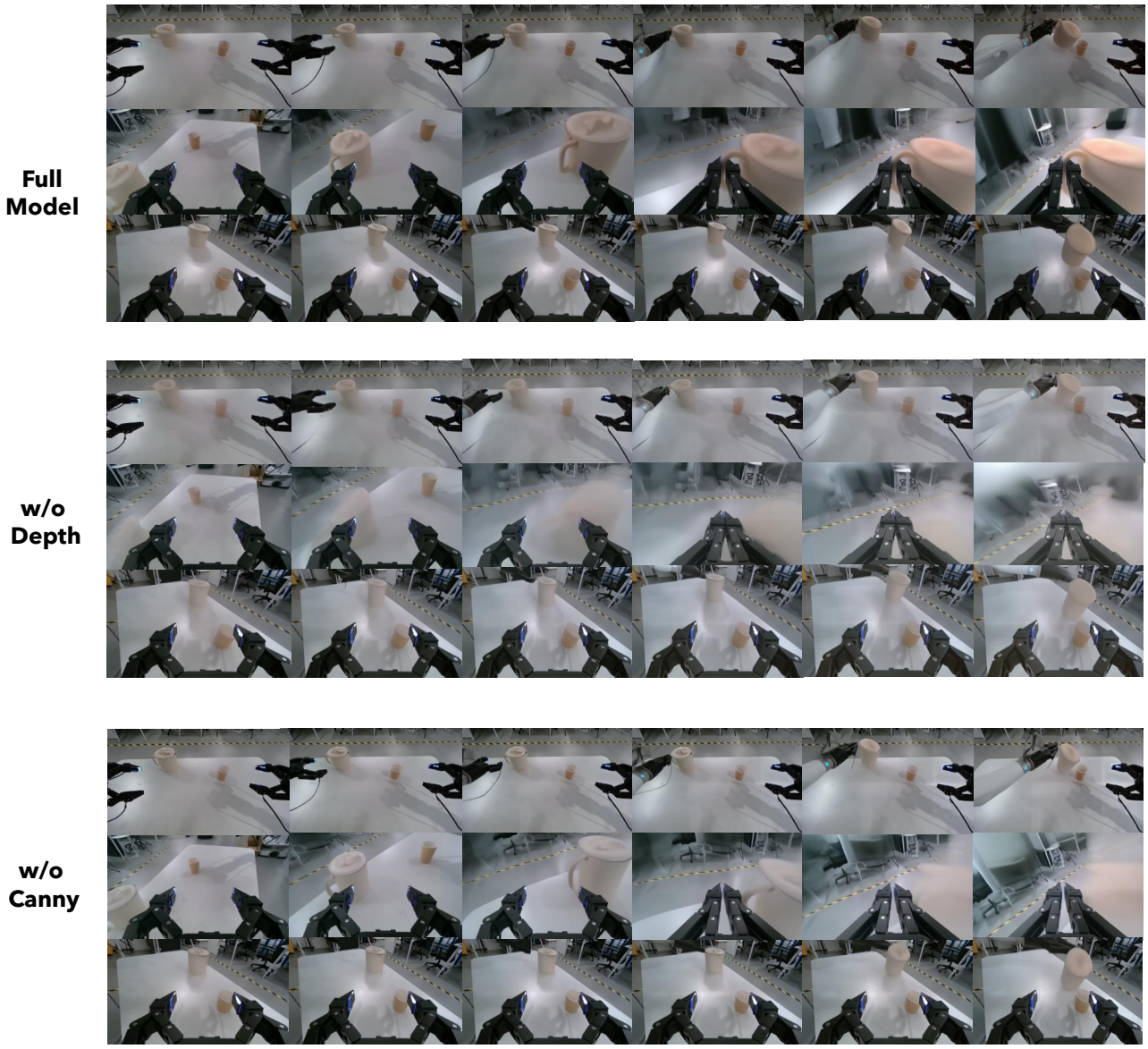

Figure 10. Ablation Study of Control Conditions on Pour Water.

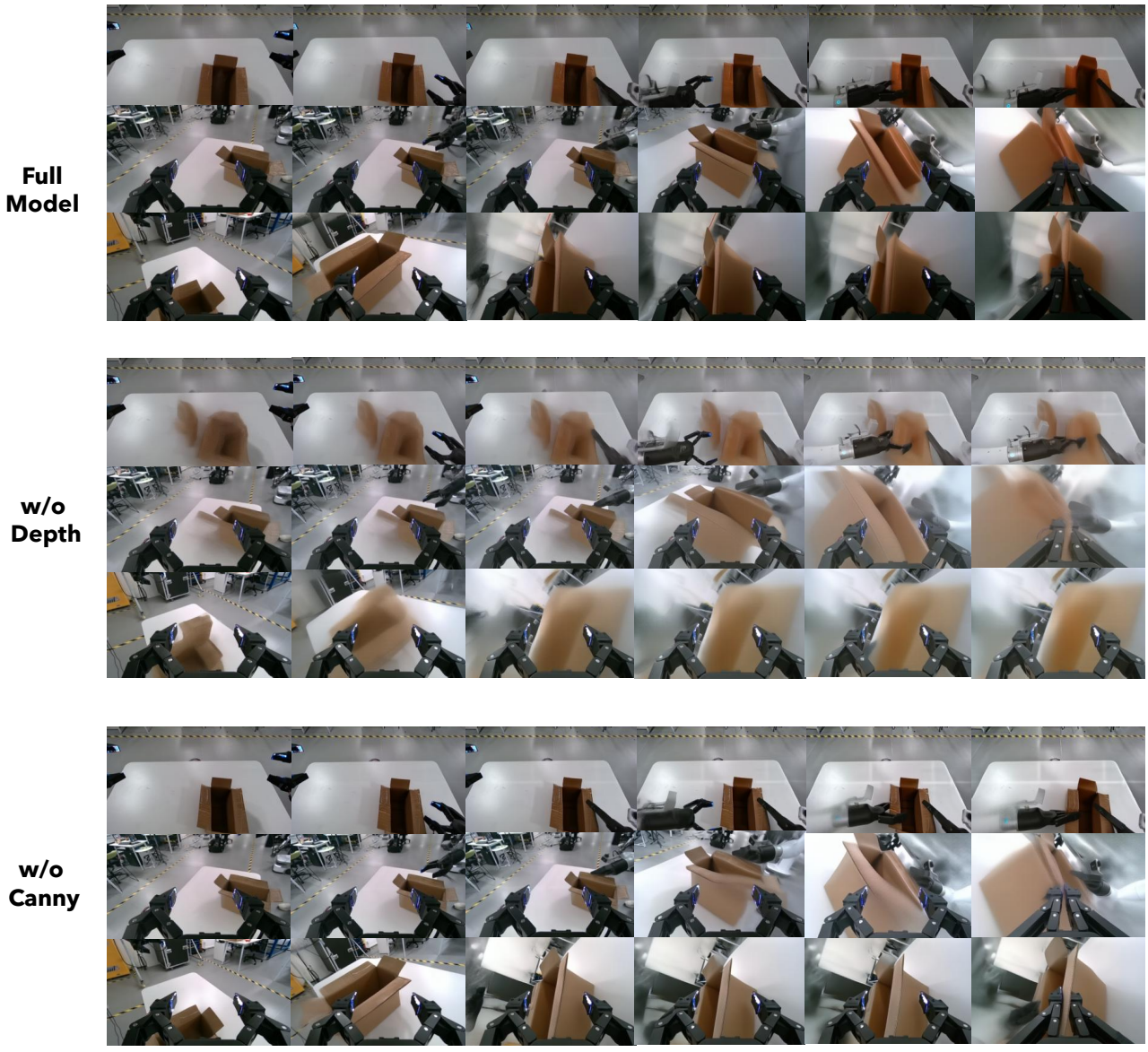

Figure 11. Ablation Study of Control Conditions on Lift Box.

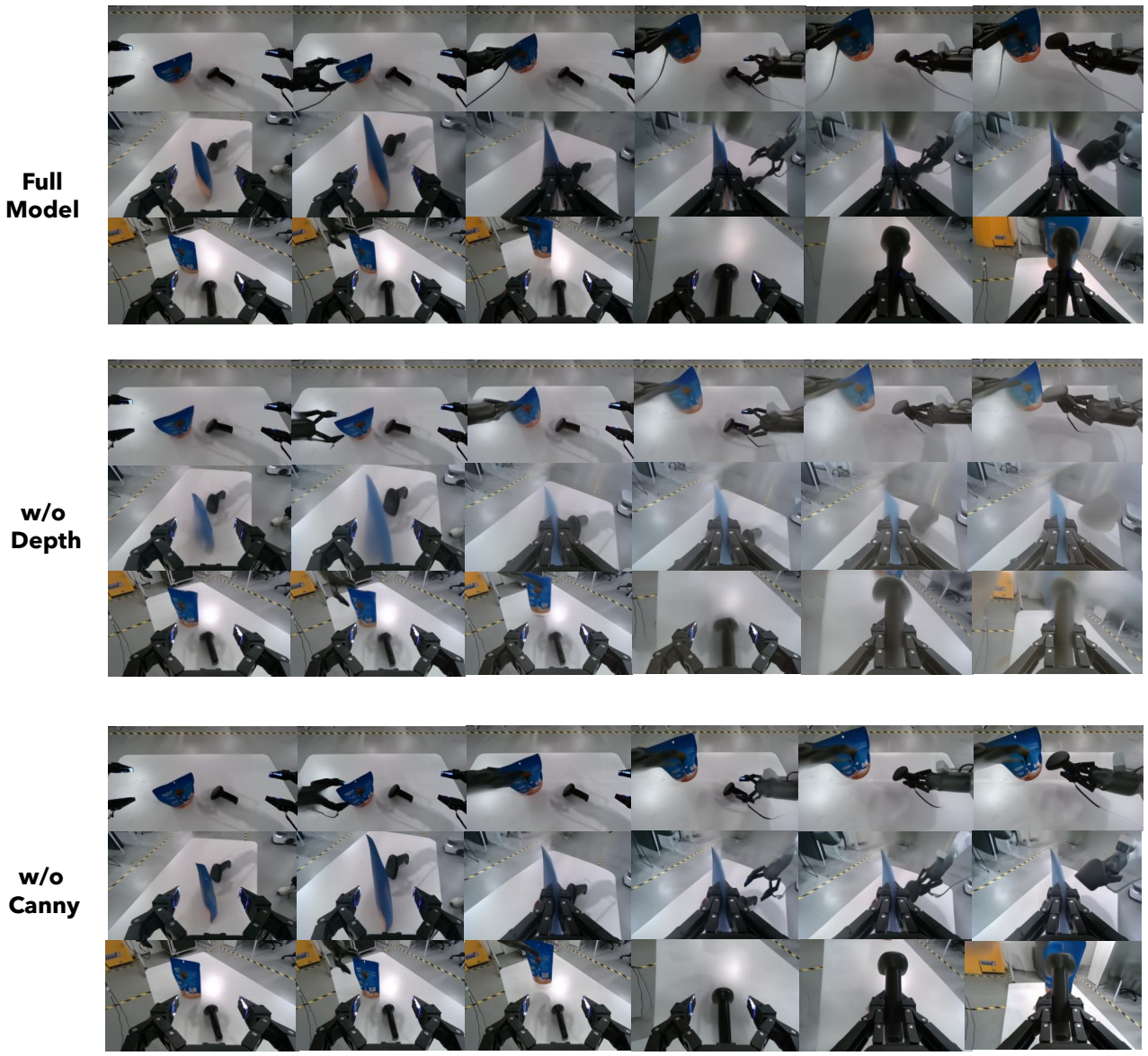

Figure 12. Ablation Study of Control Conditions on Scan Barcode.

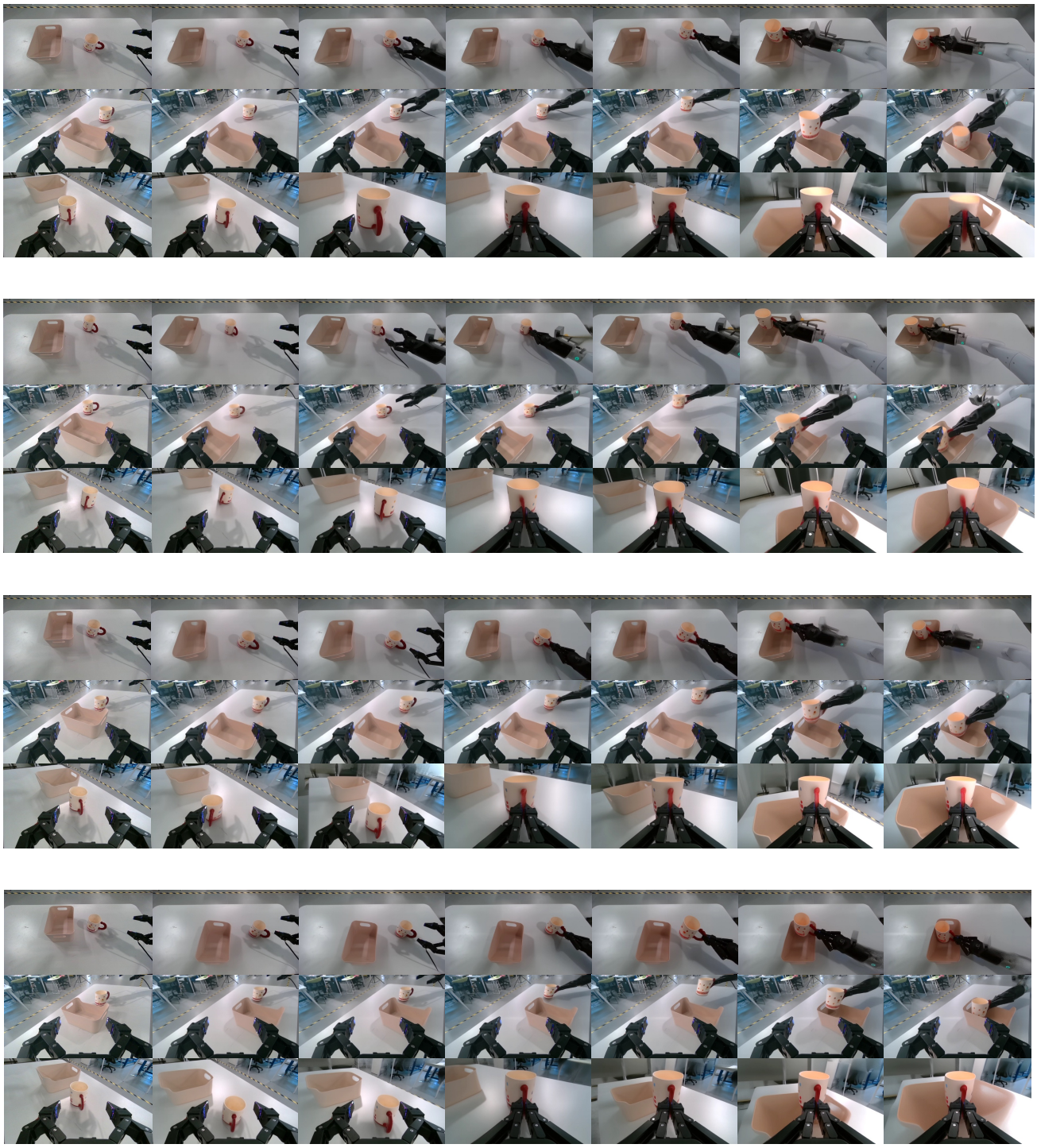

Figure 13. Visualization of Generated Videos on Mug to Basket.

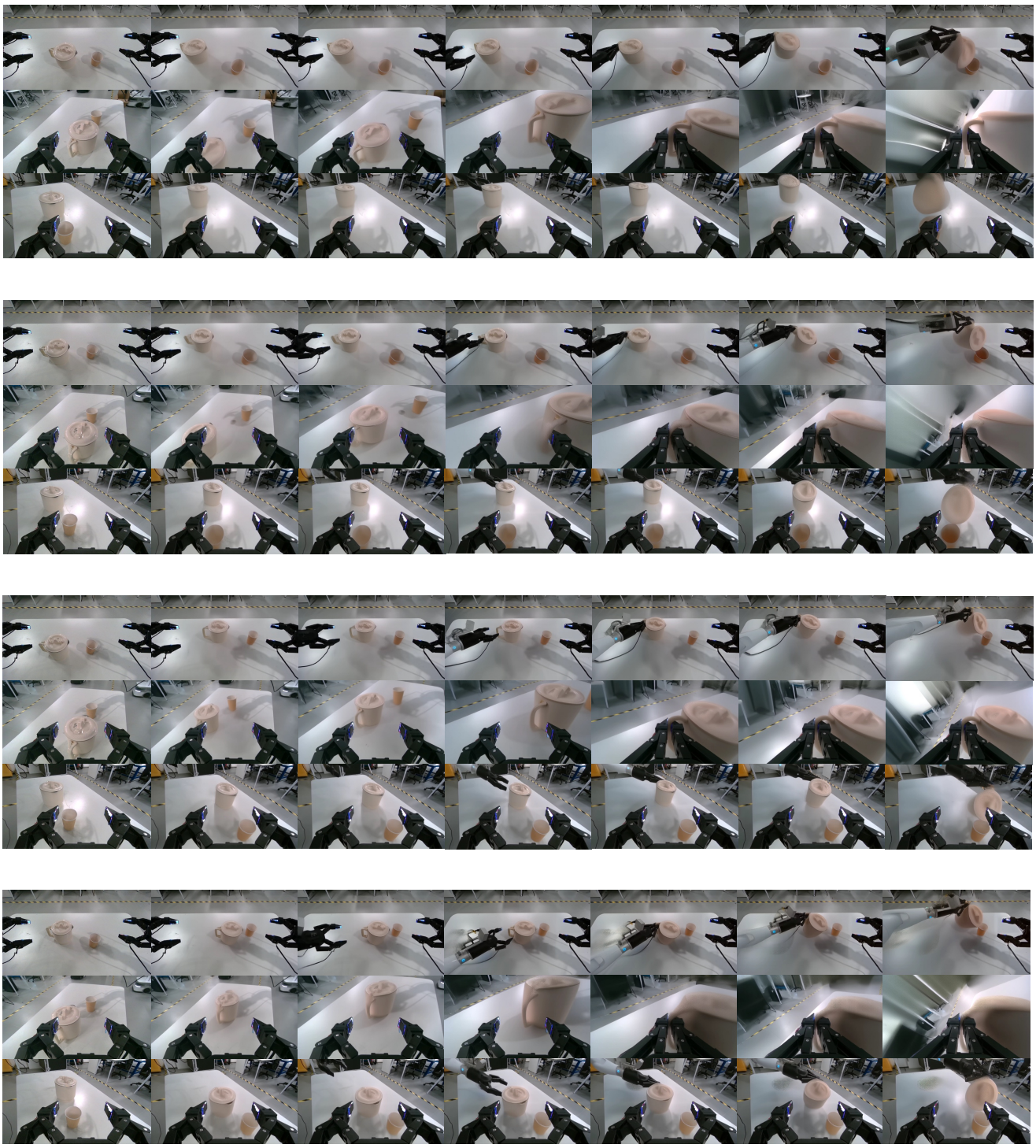

Figure 14. Visualization of Generated Videos on Pour Water.

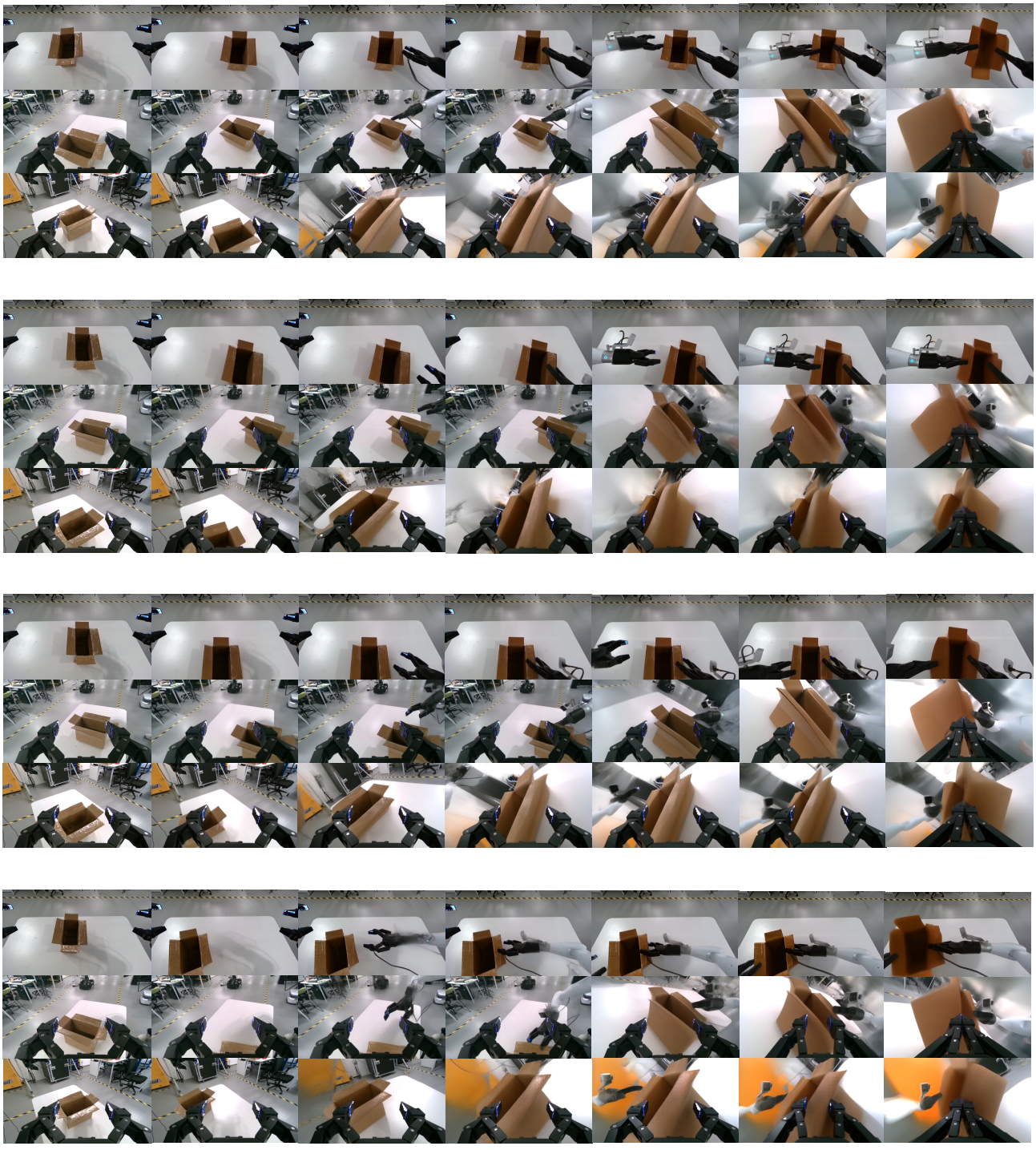

Figure 15. Visualization of Generated Videos on Lift Box.

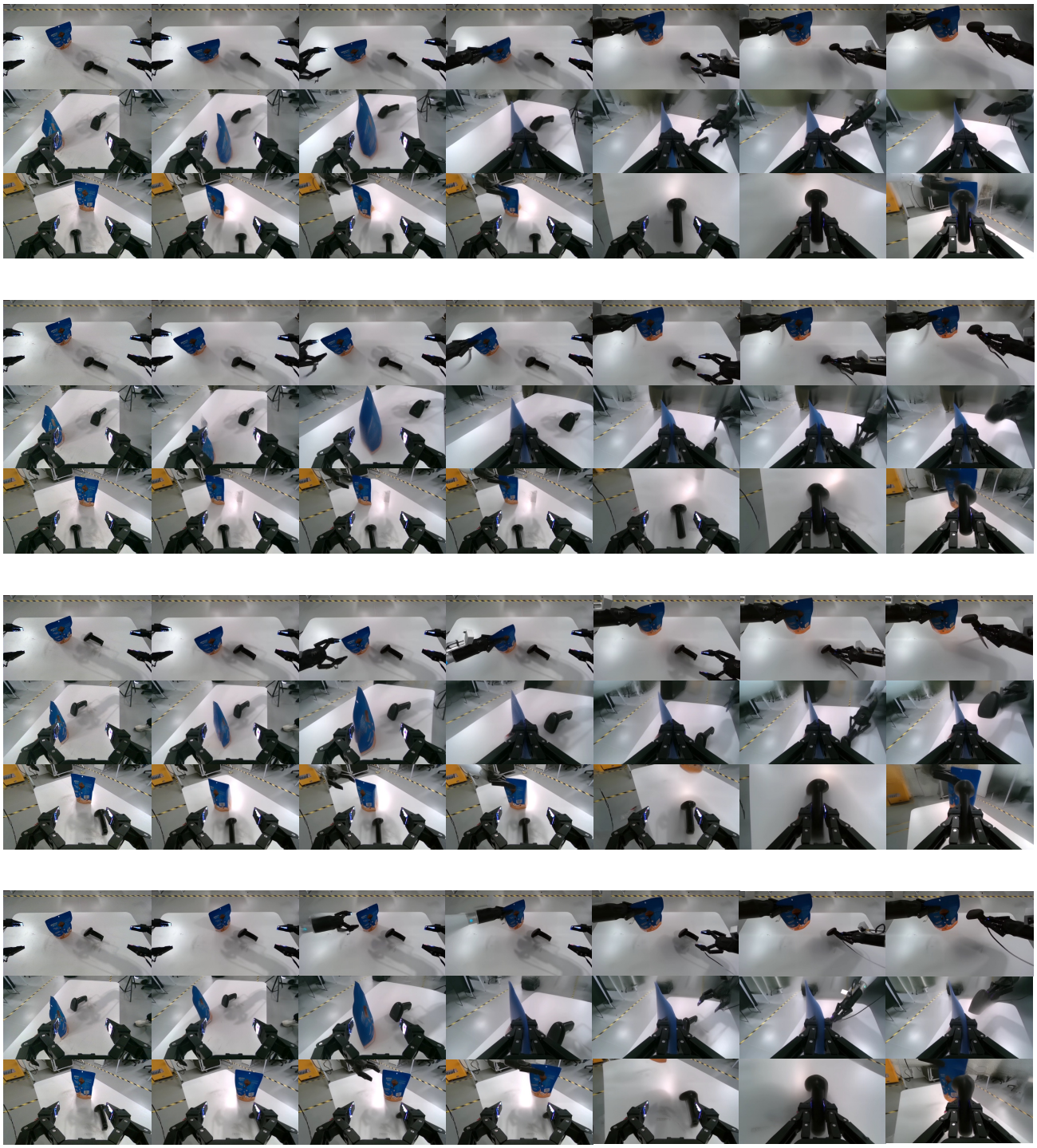

Figure 16. Visualization of Generated Videos on Scan Barcode.

References

[1] Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, Shu Jiang, Yuxin Jiang, Cheng Jing, Hongyang Li, Jialu Li, Chiming Liu, Yi Liu, Yuxiang Lu, Jianlan Luo, Ping Luo, Yao Mu, Yuehan Niu, Yixuan Pan, Jiangmiao Pang, Yu Qiao, Guanghui Ren, Cheng Ruan, Jiaqi Shan, Yongjian Shen, Chengshi Shi, Mingkang Shi, Modi Shi, Chonghao Sima, Jianheng Song, Huijie Wang, Wenhao Wang, Dafeng Wei, Chengen Xie, Guo Xu, Junchi Yan, Cunbiao Yang, Lei Yang, Shukai Yang, Maoqing Yao, Jia Zeng, Chi Zhang, Qinglin Zhang, Bin Zhao, Chengyue Zhao, Jiaqi Zhao, and Jianchao Zhu. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems, 2025. 3

[2] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023. 5

[3] Caelan Garrett, Ajay Mandlekar, Bowen Wen, and Dieter Fox. Skillmimicgen: Automated demonstration generation for efficient skill learning and deployment. In *8th Annual Conference on Robot Learning*, 2024. 1

[4] Lianghua Huang, Di Chen, Yu Liu, Yujun Shen, Deli Zhao, and Jingren Zhou. Composer: Creative and controllable image synthesis with composable conditions. *arXiv preprint arXiv:2302.09778*, 2023. 4

[5] Yan Huang, Shoujie Li, Xingting Li, and Wenbo Ding. Umigen: A unified framework for egocentric point cloud generation and cross-embodiment robotic imitation learning. *arXiv preprint arXiv:2511.09302*, 2025. 1

[6] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 2

[7] Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, et al. Genie envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025. 3, 5

[8] Liu Liu, Xiaofeng Wang, Guosheng Zhao, Keyu Li, Wenkang Qin, Jiaxiong Qiu, Zheng Zhu, Guan Huang, and Zhizhong Su. Robotransfer: Geometry-consistent video diffusion for robotic visual policy transfer. *arXiv preprint arXiv:2505.23171*, 2025. 1

[9] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. 2

[10] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. In *7th Annual Conference on Robot Learning*, 2023. 1

[11] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024. 2

[12] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025. 5

[13] Zizhao Tong, Di Chen, Sicheng Hu, Hongwei Fan, Liliang Chen, Guanghui Ren, Hao Tang, Hao Dong, and Ling Shao. Fidelity-aware data composition for robust robot generalization. *arXiv preprint arXiv:2509.24797*, 2025. 1

[14] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. 1

[15] Peng Wang, Yichun Shi, Xiaochen Lian, Zhonghua Zhai, Xin Xia, Xuefeng Xiao, Weilin Huang, and Jianchao Yang. Seededit 3.0: Fast and high-quality generative image editing. *arXiv preprint arXiv:2506.05083*, 2025. 1

[16] Xiuwei Xu, Angyuan Ma, Hankun Li, Bingyao Yu, Zheng Zhu, Jie Zhou, and Jiwen Lu. R2rgen: Real-to-real 3d data generation for spatially generalized manipulation. *arXiv preprint arXiv:2510.08547*, 2025. 1

[17] Zhengrong Xue, Shuying Deng, Zhenyang Chen, Yixuan Wang, Zhecheng Yuan, and Huazhe Xu. DemoGen: Synthetic Demonstration Generation for Data-Efficient Visuomotor Policy Learning. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, 2025. 1

[18] Sizhe Yang, Wenye Yu, Jia Zeng, Jun Lv, Kerui Ren, Cewu Lu, Dahua Lin, and Jiangmiao Pang. Novel demonstration generation with gaussian splatting enables robust one-shot manipulation. *arXiv preprint arXiv:2504.13175*, 2025. 1

[19] Justin Yu, Letian Fu, Huang Huang, Karim El-Refai, Rares Andrei Ambrus, Richard Cheng, Muhammad Zubair Irshad, and Ken Goldberg. Real2render2real: Scaling robot data without dynamics simulation or robot hardware. In *9th Annual Conference on Robot Learning*, 2025. 1

[20] Jiyao Zhang, Mingjie Pan, Baifeng Xie, Yinghao Zhao, Wenlong Gao, Guangte Xiang, Jiawei Zhang, Dong Li, Zhijun Li, Sheng Zhang, Hongwei Fan, Chengyue Zhao, Shukai Yang, Maoqing Yao, Chuanzhe Suo, and Hao Dong. Agibot digitalworld. https://agibot-digitalworld.com/, 2025. 3