

A Implementation Details

Architecture. In this section, we detail the architecture of the reconstructive model, predictive model, and render head. Both the reconstructive and predictive models share the same Transformer architecture, which includes recent techniques such as SwiGLU [5], RoPE [7], and QK Norm [2]. To handle the 3D triplane representations, we utilize rotary position encoding inspired by M-RoPE [8]. The feature dimensions are split into three parts, corresponding to the three spatial dimensions of the triplane (*i.e.*, $x - y$, $x - z$, and $y - z$ planes). For each dimension, we apply RoPE encoding separately, which allows the model to effectively learn relative positional relationships across the three triplane planes.

The render head takes the triplane features, interpolated according to the sampled points, as input and outputs the corresponding density, RGB values, and semantic features. The render head is implemented as two MLP layers with residual connections. Each layer consists of a linear layer, followed by layer normalization and a ReLU activation followed by a linear layer. This architecture enables efficient feature processing and representation learning for both the reconstruction and prediction tasks, with the same render head shared across both tasks.

Additionally, we follow GNFactor [9] by adopting a coarse-to-fine hierarchical structure for rendering. In the "fine" network, we apply depth-guided sampling to refine the predictions, improving the model's ability to reconstruct and predict scene features, especially at finer levels of detail.

Hyperparameter	Value
triplane resolution	$16 \times 16 \times 16$
transformer depth	8
transformer width	768
attention heads	12
MLP ratio	4.0
render head width	768
render head layers	2
batch size	256
ray batch size	32
optimizer	AdamW
learning rate	0.0001
coarse sampled points	128
fine depth sampled points	64
fine uniform sampled points	64
λ_{pred}	1.0
λ_{recon}	1.0
λ_{rgb}	1.0
λ_{sem}	0.1
λ_{depth}	0.01

Table A: Hyperparameters.

Target view augmentation. In real-world setups, the limited and often fixed camera viewpoints pose a challenge for rendering-based pretraining. To address this limitation, we leverage a pre-trained visual-conditioned multi-view diffusion model to generate novel target views as additional supervision. Specifically, we utilize See3D [3], a large-scale pretrained multi-view video diffusion model.

Given input RGB-D images for a target frame, we first reconstruct the scene's point cloud. We then select an input viewpoint as the base view, from which we sample a new viewpoint with a random angular offset on a sphere surrounding the center of the scene. The angular offset is restricted within a maximum range of ± 30 degrees, as we observe that the quality of the synthesized novel view images decreases as the offset angle becomes larger. We generate a camera trajectory consisting of 25 frames, where each frame corresponds to a camera pose interpolated between the original and new viewpoints. We then use point cloud rendering to project the reconstructed point cloud onto each frame of the trajectory, producing warped images and corresponding masks. These warped images and masks, along with the original view, are fed into See3D, which generates high-fidelity novel-view images. Due to the time required for generating these trajectories, we augment only the keyframe viewpoints. For each keyframe, we randomly sample four distinct camera trajectories, generating a total of 100 novel view images per keyframe. In both simulation and real-world experiments, we maintain fixed camera setups and viewpoints, with the view augmentation process described above applied consistently.

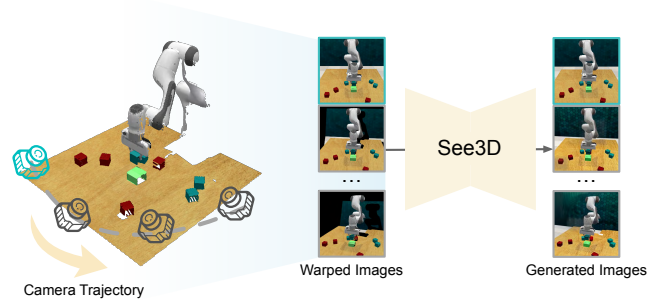


Figure A: Target view augmentation pipeline.

Hyperparameters. We present the hyperparameters used in DynaRend as shown in Tab. A.

55 B Simulation Experiment Details

56 **RLBench-18.** For the 18-task setting, we follow PerAct [6] and select 18 RLBench tasks that
 57 involve at least two or more variations in object types, spatial arrangements, or instructions. This
 58 curated subset is designed to evaluate the multi-task generalization capabilities of different agents
 59 under diverse and realistic conditions. Detailed descriptions of each task can be found in Tab. B.

Task	Variation Type	# Variations	Avg. Keyframes	Language Template
open drawer	placement	3	3.0	"open the __ drawer"
slide block	color	4	4.7	"slide the block to __ target"
sweep to dustpan	size	2	4.6	"sweep dirt to the __ dustpan"
meat off grill	category	2	5.0	"take the __ off the grill"
turn tap	placement	2	2.0	"turn __ tap"
put in drawer	placement	3	12.0	"put the item in the __ drawer"
close jar	color	20	6.0	"close the __ jar"
drag stick	color	20	6.0	"use the stick to drag the cube onto the __ target"
stack blocks	color,count	60	14.6	"stack __ blocks"
screw bulb	color	20	7.0	"screw in the __ light bulb"
put in safe	placement	3	5.0	"put the money away in the safe on the __ shelf"
place wine	placement	3	5.0	"stack the wine bottle to the __ of the rack"
put in cupboard	category	9	5.0	"put the __ in the cupboard"
sort shape	shape	5	5.0	"put the __ in the shape sorter"
push buttons	color	50	3.8	"push the __ button, [then the __ button]"
insert peg	color	20	5.0	"put the ring on the __ spoke"
stack cups	color	20	10.0	"stack the other cups on top of the __ cup"
place cups	count	3	11.5	"place __ cups on the cup holder"

Table B: Details on 18 RLBench tasks.

60 **RLBench-71.** For large-scale multi-task evaluation, we follow SPA [10] and divide all executable
 61 tasks in RLBench into two groups, consisting of 35 and 36 tasks respectively. For each group, we
 62 train a language-conditioned multi-task agent. All RVT-2 [1] baselines are implemented using the
 63 same two-stage architecture, differing only in the choice of pretrained vision encoder. All other com-
 64 ponents and training hyperparameters remain consistent with the original RVT-2 setup. Reported
 65 results are taken directly from SPA [10].

66 The 35 tasks in Group 1 include: basketball in hoop, put rubbish in bin, meat off grill,
 67 meat on grill, slide block to target, reach and drag, take frame off hanger, water
 68 plants, hang frame on hanger, wipe desk, stack blocks, reach target, push button, lamp
 69 on, toilet seat down, close laptop lid, open box, open drawer, pick up cup, turn tap,
 70 take usb out of computer, play jenga, insert onto square peg, take umbrella out of
 71 umbrella stand, insert usb in computer, straighten rope, turn oven on, change clock,
 72 close microwave, close fridge, close grill, open grill, unplug charger, press switch,
 73 take money out safe.

74 The 36 tasks in Group 2 include: The 36 tasks in Group 2 include: change channel, tv on, push
 75 buttons, stack wine, scoop with spatula, place hanger on rack, move hanger, sweep to
 76 dustpan, take plate off colored dish rack, screw nail, take shoes out of box, slide
 77 cabinet open and place cups, lamp off, pick and lift, take lid off saucepan, close
 78 drawer, close box, phone on base, toilet seat up, put books on bookshelf, beat the buzz,
 79 stack cups, put knife on chopping board, place shape in shape sorter, take toilet
 80 roll off stand, put umbrella in umbrella stand, setup checkers, open window, open
 81 wine bottle, open microwave, put money in safe, open door, close door, open fridge,
 82 open oven, plug charger in power supply.

83 **Colosseum.** Colosseum [4] is a recently proposed benchmark designed to evaluate the general-
 84 ization capabilities of robotic manipulation policies under diverse domain shifts. It consists of 20
 85 manipulation tasks drawn from the RLBench suite, including: basketball in hoop, close box,
 86 close laptop lid, empty dishwasher, get ice from fridge, hockey, meat on grill, move
 87 hanger, wipe desk, open drawer, slide block to target, reach and drag, put money in
 88 safe, place wine at rack location, insert onto square peg, stack cups, turn oven on,
 89 straighten rope, setup chess, scoop with spatula.

90 Colosseum defines 14 types of perturbation factors to simulate real-world domain shifts. In our
 91 evaluation, we follow the standard setting and apply 12 of these perturbation factors to all 20 tasks

to systematically assess the robustness and generalization ability of learned policies under unseen conditions. The 12 perturbations can be categorized into three groups:

- **Manipulation Object (MO) Perturbation:** The manipulation object is the item directly manipulated by the robot, such as the wine bottle in place wine at rack location. Manipulation object perturbations include variations in color, texture, and size.
- **Receiver Object (RO) Perturbation:** The receiver object is task-relevant but not directly manipulated, such as the rack in the same task. Receiver object perturbations also include changes in color, texture, and size.
- **Background Perturbation:** These factors affect the overall scene without modifying task-relevant objects. They include changes to light color, table color, table texture, background texture, camera pose, and the addition of distractor objects.

Tab. C summarizes the applied perturbations for each task. For more implementation details and perturbation configurations, please refer to Colosseum [4].

Variation	MO	RO	MO Color	MO Size	MO Texture	RO Color	RO Size	RO Texture	Object Mass
	-	-	discrete	continuous	discrete	discrete	continuous	discrete	continuous
basketball in hoop	ball	hoop	20	[0.75,1.25]	213	20	[0.75,1.15]	-	-
close box	box	-	20	[0.75,1.15]	-	-	-	-	-
close laptop lid	laptop	-	20	[0.75,1.00]	-	-	-	-	-
empty dishwasher	dishwasher	plate	20	[0.80,1.00]	-	20	[0.80,1.00]	213	-
get ice from fridge	cup	fridge	20	[0.75,1.25]	213	20	[0.75,1.00]	-	-
hockey	stick	ball	20	[0.95,1.05]	-	20	[0.75,1.25]	213	[0.1,0.5]
meat on grill	meat	grill	20	[0.65,1.15]	-	20	-	-	-
move hanger	hanger	pole	20	-	-	20	-	-	-
wipe desk	sponge	beans	20	[0.75,1.25]	213	20	-	-	[1.0,5.0]
open drawer	drawer	-	20	[0.75,1.00]	-	-	-	-	-
slide block to target	block	-	20	-	213	-	-	-	[1.0,15.0]
reach and drag	stick	block	20	[0.80,1.10]	213	20	[0.50,1.00]	213	[0.5,2.5]
put money in safe	money	safe	20	[0.50,1.00]	213	20	-	213	-
place wine at rack location	bottle	shelve	20	[0.85,1.15]	-	20	[0.85,1.15]	213	-
insert onto square peg	peg	spokes	20	[1.00,1.50]	-	20	[0.85,1.15]	213	-
stack cups	cups	-	20	[0.75,1.25]	213	-	-	-	-
turn oven on	knobs	-	20	[0.50,1.50]	-	-	-	-	-
straighten rope	rope	-	20	-	213	-	-	-	-
setup chess	chess pieces	board	20	[0.75,1.25]	213	20	-	-	-
scoop with spatula	spatula	block	20	[0.75,1.25]	213	20	[0.75,1.50]	213	[1.0,5.0]

Table C: Summary of tasks and their perturbation factors. For more details, check Colosseum [4].

C Real-world Experiment Details

Hardware Setup. For the hardware setup, we use a Franka Research 3 robot arm equipped with a parallel gripper. Visual input is provided by two Orbbec Femto Bolt RGB-D cameras, positioned on either side of the robot arm. The cameras are calibrated using the `kalibr` package to determine the extrinsics between them, while `easy_handeye` package is used to calibrate the extrinsics between the camera and the robot base-frame. The cameras provide RGB-D images at a resolution of 1280x720 at 30Hz. To prepare the images for model input, we resize the shortest edge to 256 and crop them to a resolution of 256x256.

Task	Variation Type	# Variations	Avg. Keyframes	Language Template
stack cups	color	5	11.5	"stack __ cups on __ cup"
stack blocks	color	5	12.1	"stack __ blocks on __ block"
sort shape	placement	1	5.4	"put yellow circle in shape sorter"
close pot	placement	1	5.7	"put lid on pot"
put item in drawer	category	4	8.3	"put __ in drawer"

Table D: Details on 5 real-world tasks.

Data Collection. The real-world dataset is collected through human demonstrations. Specifically, for each task and scenario, a human demonstrator kinesthetically moves the robot arm to specify a series of end-effector poses. Afterward, the robot arm is reset to its initial position, and the demonstrator sequentially moves it to the specified poses. During this process, the camera streams and

Task Name	No Variations	MO Color	RO Color	MO Texture	RO Texture	MO Size	RO Size	Light Color	Table Color	Table Texture	Distractor	Background Texture	Camera Pose
basketball in hoop	100	96	100	96	-	100	100	100	100	100	40	100	100
close box	96	84	-	-	-	92	-	92	88	88	88	96	88
close laptop lid	100	100	-	-	-	0	-	100	100	96	68	100	96
empty dishwasher	0	0	0	-	0	0	0	0	0	0	0	0	0
get ice from fridge	92	88	96	88	-	80	72	96	100	96	64	92	100
hockey	32	36	36	-	20	44	32	28	36	20	28	28	32
meat on grill	96	100	100	-	-	100	-	100	100	100	96	96	100
move hanger	0	0	0	-	-	-	-	0	0	0	8	0	0
wipe desk	0	0	-	0	-	0	-	0	0	0	0	0	0
open drawer	84	68	-	-	-	68	-	68	72	60	64	68	68
slide block to target	100	100	-	100	-	-	-	100	100	100	100	100	100
reach and drag	100	96	96	100	100	52	100	96	96	96	60	100	96
put money in safe	56	80	4	60	64	68	-	64	60	68	60	76	68
place wine at rack location	88	80	92	-	80	52	84	80	84	88	96	88	88
insert onto square peg	8	4	12	-	28	16	12	16	0	16	0	8	8
stack cups	72	68	-	52	-	36	-	48	52	60	36	72	56
turn oven on	76	100	-	-	-	56	-	92	92	96	100	100	88
straighten rope	80	64	-	84	-	-	-	64	72	68	24	60	80
setup chess	20	0	4	16	-	16	-	16	24	24	12	8	12
scoop with spatula	84	56	96	80	88	96	60	16	88	96	56	88	88
average	64.2	61.0	53.0	67.6	54.2	54.3	51.5	59.0	63.2	63.6	50.0	64.0	63.4

Table E: All results on Colosseum benchmark.

robot arm states are recorded, including end-effector position, joint positions, and joint velocities. For each task, we collect 30 human demonstrations to train the model.

Keyframe Discovery. Following the approach of PerAct [6], we use heuristic rules to identify keyframes from the collected demonstrations. A frame is considered a keyframe if: 1) The joint velocities are near zero, and 2) The gripper open state has not changed.

Execution. For robot control and motion planning, we use the Franka ROS and MoveIt with RRT-Connect as default planner.

D Detailed Results

We present the detailed results of all tasks in Tab. E and Tab. F.

E Visualizations

In Fig. C and Fig. E, we present the results of target view augmentation in both the simulator and real-world environments. Fig. B and Fig. D showcase the rendering results in the simulator and real-world settings, respectively.

F Additional Qualitative Analysis

In the supplementary materials, we provide episode rollouts of several representative tasks in the simulator and real-world demos in the attached videos.

Method	RVT	MoCov3	MAE	DINOv2	CLIP	EVA	InternViT	MVP	VC-1	SPA	DynaRend
<i>Group 1</i>											
basket ball in hoop	100	100	100	100	100	100	100	100	100	100	100
put rubbish in bin	96	100	100	96	96	96	100	96	100	100	100
meat off grill	100	100	100	100	100	100	100	100	100	100	100
meat on grill	88	80	76	76	68	80	72	68	76	80	92
slide block to target	8	0	84	96	24	4	0	100	100	4	100
reach and drag	100	100	96	88	100	96	100	96	100	100	100
take frame off hanger	96	88	88	92	88	84	84	88	88	96	96
water plants	48	64	60	28	64	60	44	52	60	68	56
hang frame on hanger	0	8	4	0	4	8	8	12	4	4	24
wipe desk	0	0	0	0	0	0	0	0	0	0	0
stack blocks	56	60	72	72	68	56	60	84	68	68	80
reach target	96	60	96	88	100	96	80	92	96	92	100
push button	92	100	100	100	100	100	100	100	100	100	100
lamp on	60	88	68	84	88	52	80	28	88	64	100
toilet seat down	100	100	100	100	100	100	100	96	96	100	96
close laptop lid	84	96	96	96	96	84	80	80	96	100	96
open drwaer	92	88	96	92	100	88	88	92	96	96	96
pick up cup	88	92	92	88	96	96	88	96	96	96	96
turn tap	88	88	84	84	96	88	92	96	100	100	96
take usb out of computer	100	100	100	100	100	100	100	100	88	100	92
play jenga	100	96	96	96	100	96	100	96	96	96	100
insert onto square peg	8	28	84	80	44	88	40	64	92	84	24
take umbrella out of umbrella stand	92	92	100	100	92	100	96	100	100	100	100
insert usb in computer	8	12	20	20	24	24	20	16	8	64	12
straighten rope	56	56	44	72	80	48	72	52	60	84	56
turn oven on	92	96	96	96	96	96	96	100	100	100	96
change clock	64	64	68	48	68	64	72	64	60	68	68
close microwave	100	100	100	100	100	100	100	100	100	100	100
close fridge	92	80	92	92	88	92	96	88	92	100	96
close grill	92	96	96	96	96	96	96	100	100	96	96
open grill	76	100	100	100	100	100	100	96	100	100	96
unplug charger	48	44	32	48	36	48	40	40	44	44	96
press switch	84	92	92	88	72	76	84	76	88	92	76
take money out safe	80	100	96	100	100	100	100	100	100	100	100
<i>Group 2</i>											
change channel	0	0	8	4	0	0	4	0	0	4	100
tv on	0	4	8	0	4	4	8	4	4	8	100
push buttons	0	12	4	4	0	0	0	0	12	4	96
stack wine	12	12	16	40	4	12	0	28	8	28	20
scoop with spatula	0	0	0	0	0	0	0	0	0	0	92
place hanger on rack	0	0	0	0	0	0	0	0	0	0	100
move hanger	72	0	0	0	0	0	0	0	0	0	88
sweep to dustpan	48	92	96	96	96	92	100	100	88	96	100
take plate off colored dish rack	92	96	100	96	92	84	96	88	92	96	84
screw nail	32	52	36	36	36	36	52	32	32	48	56
take shoes out of box	4	20	28	24	36	40	12	32	36	36	8
slide cabinet open and place cups	0	0	0	0	0	0	4	0	0	4	0
lamp off	96	100	96	96	100	96	96	100	100	100	100
pick and lift	72	88	96	92	96	92	80	96	96	96	88
take lid off saucepan	100	100	100	100	100	100	100	100	100	100	100
close drawer	100	100	100	100	100	96	100	100	100	100	100
close box	96	92	92	96	96	100	96	100	96	100	88
phone on base	100	100	100	100	100	100	96	100	100	100	100
toilet seat up	72	80	88	100	88	88	80	88	92	96	24
put books on bookshelf	28	12	24	24	28	28	20	20	28	16	28
beat the buzz	88	88	92	96	88	92	84	88	88	100	100
stack cups	24	40	56	52	52	48	56	64	68	64	68
put knife on chopping board	80	72	76	68	72	80	88	80	76	80	72
place shape in shape sorter	12	20	36	32	28	36	20	44	36	56	48
take toilet roll off stand	100	100	92	76	96	92	88	84	92	96	84
put umbrella in umbrella stand	20	8	0	12	12	0	4	12	8	12	12
setup checkers	44	76	80	68	68	88	92	92	80	80	92
open window	92	96	96	100	100	96	100	96	100	100	92
open wine bottle	96	80	100	88	92	92	88	96	88	88	100
open microwave	72	100	100	88	96	100	80	96	100	100	84
put money in safe	100	96	100	88	92	100	96	100	100	100	96
open door	88	100	96	96	96	96	96	84	96	96	92
close door	12	32	68	56	60	80	20	24	20	60	0
open fridge	24	44	52	48	44	36	64	52	32	64	68
open oven	0	8	4	12	8	4	20	4	4	16	80
plug charger in power supply	40	32	36	32	24	44	36	24	32	60	28

Table F: All results on 71 RL Bench tasks.

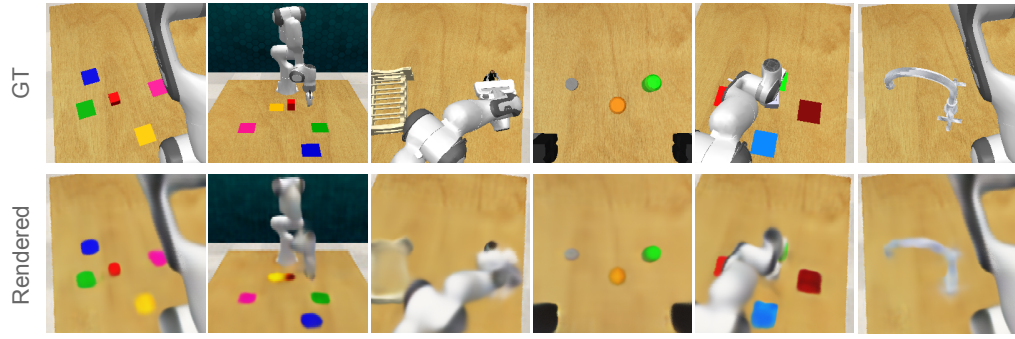


Figure B: Visualization of rendered results in simulation.

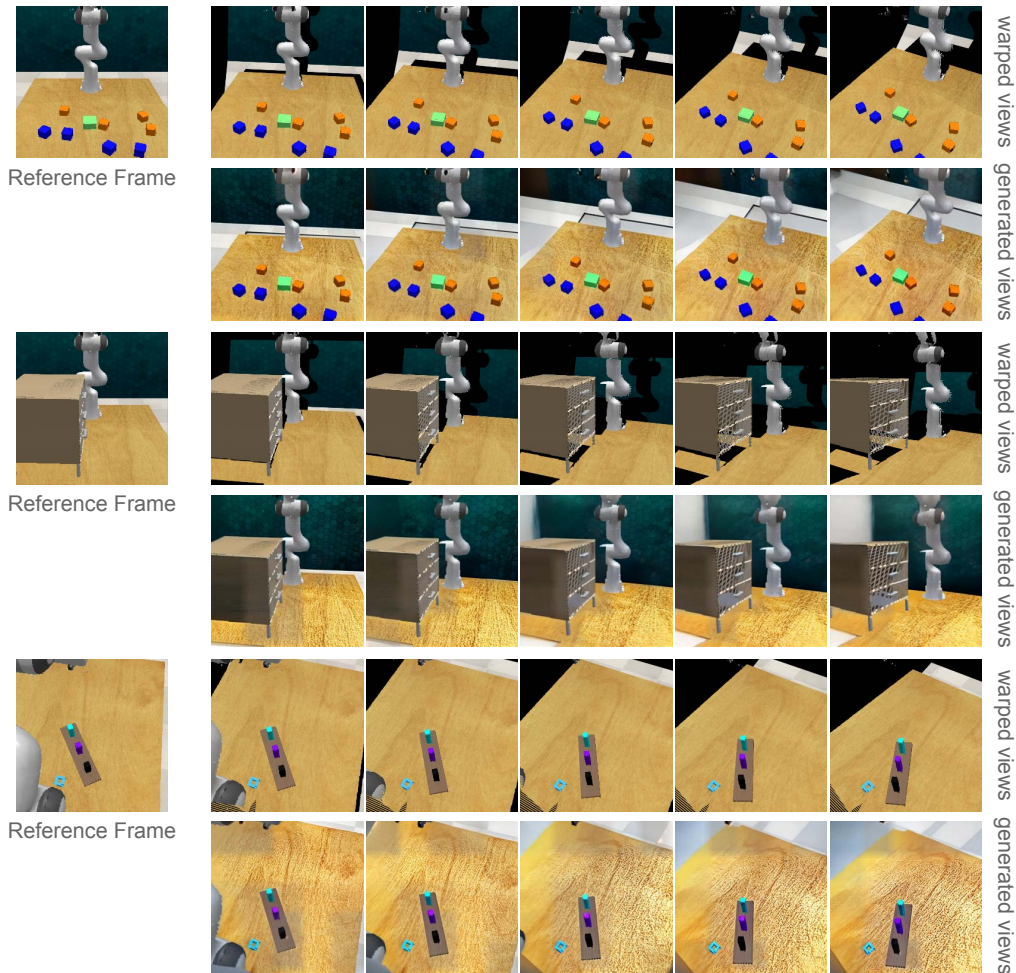


Figure C: Visualization of target view synthesis in simulation.

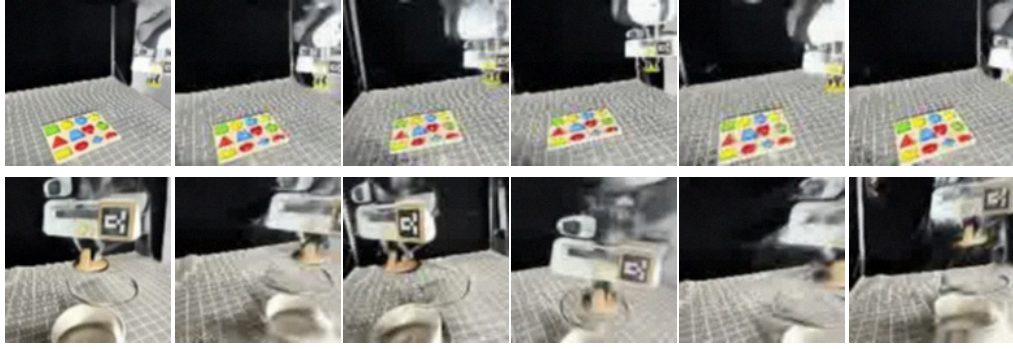


Figure D: Visualization of rendered results in the real world.

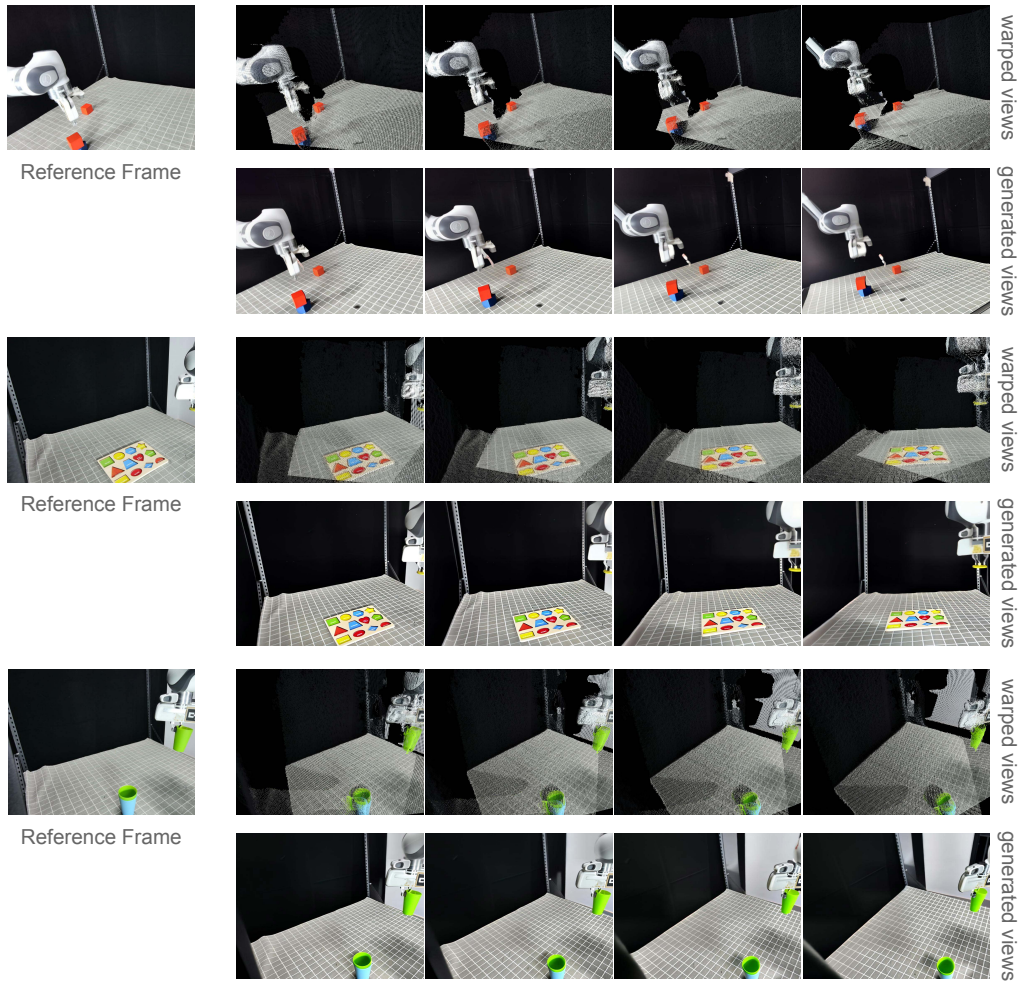


Figure E: Visualization of target view synthesis in the real world.

References

- [1] Goyal, A., Blukis, V., Xu, J., Guo, Y., Chao, Y.W., Fox, D., 2024. Rvt-2: Learning precise manipulation from few demonstrations. arXiv preprint arXiv:2406.08545 .
- [2] Henry, A., Dachapally, P.R., Pawar, S., Chen, Y., 2020. Query-key normalization for transformers. arXiv preprint arXiv:2010.04245 .
- [3] Ma, B., Gao, H., Deng, H., Luo, Z., Huang, T., Tang, L., Wang, X., 2024. You see it, you got it: Learning 3d creation on pose-free videos at scale. arXiv preprint arXiv:2412.06699 .
- [4] Pumacay, W., Singh, I., Duan, J., Krishna, R., Thomason, J., Fox, D., 2024. The colosseum: A benchmark for evaluating generalization for robotic manipulation, in: RSS 2024 Workshop: Data Generation for Robotics.
- [5] Shazeer, N., 2020. Glu variants improve transformer. arXiv preprint arXiv:2002.05202 .
- [6] Shridhar, M., Manuelli, L., Fox, D., 2023. Perceiver-actor: A multi-task transformer for robotic manipulation, in: Conference on Robot Learning, PMLR. pp. 785–799.
- [7] Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., Liu, Y., 2024. Roformer: Enhanced transformer with rotary position embedding. Neurocomputing 568, 127063.
- [8] Wang, P., Bai, S., Tan, S., Wang, S., Fan, Z., Bai, J., Chen, K., Liu, X., Wang, J., Ge, W., et al., 2024. Qwen2-vl: Enhancing vision-language model’s perception of the world at any resolution. arXiv preprint arXiv:2409.12191 .
- [9] Ze, Y., Yan, G., Wu, Y.H., Macaluso, A., Ge, Y., Ye, J., Hansen, N., Li, L.E., Wang, X., 2023. Gnfactor: Multi-task real robot learning with generalizable neural feature fields, in: Conference on Robot Learning, PMLR. pp. 284–301.
- [10] Zhu, H., Yang, H., Wang, Y., Yang, J., Wang, L., He, T., 2024. Spa: 3d spatial-awareness enables effective embodied representation. arXiv preprint arXiv:2410.08208 .