OG-VLA: Orthographic Image Generation for 3D-Aware Vision-Language Action Model

APPENDIX

A. TASKS

Task Types	Goal States	Success Ranges	Training Data
Pickup Object	10, 20, 30, 40 (cm)	±5 cm	623
ReorientObject	0, 45, 135, 180 (°)	±20°	355
Open Drawer	25, 50, 75, 100 (%)	±10%	554
Close Drawer	0, 25, 50, 75 (%)	±10%	671
Open Cabinet	25, 50, 75, 100 (%)	±10%	319
Close Cabinet	0, 25, 50, 75 (%)	±10%	478
Pour Water	25, 50, 75, 100 (%)	±10%	312
Transfer Water	20, 40, 60, 80 (%)	±10%	259

TABLE I: Overview of the 8 tasks in ARNOLD. Each task features 4 goal states specified by human language, one of which is reserved for novel state evaluation and the other three are seen in the training dataset. The task is considered successful when the object state remains in the success range for two seconds. For Transfer Water, an additional condition of only less than 10% spillage of the original amount of water in the cup is imposed.

Tasks	Examples of Templates
Pickup Object	*Raise* [value_object] [value_height] *above the ground*
Reorient Object	*Reorient* [value_object] [value_degree] *away from the up axis*
Open Drawer	*Open the* [value_position] [value_object] [value_percent]
Close Drawer	*Close the* [value_position] [value_object] [value_percent]
Open Cabinet	*Open the* [value_position] [value_object] [value_percent]
Close Cabinet	*Close the* [value_position] [value_object] [value_percent]
Pour Water	*Pour* [value_percent] *water out of* [value_object]
Transfer Water	*Transfer* [value_percent] *water to* [value_object]

TABLE II: Examples of instruction templates used for the tasks.

B. PROMPTS USED IN MODEL VARIANTS

A. OG-VLA Prompt and Response

Prompt: Task: “the bottle should be twenty centimeters from the ground.". Where should the robot move next? Show the gripper’s next pose as translation and rotation heatmaps on the input orthographic views. Translation should be represented as red heatmap on all 4 views. Follow the provided instruction to compute correct translation points in the images. Rotation should be represented as yellow, blue, and green heatmaps for the front, top, and left views, corresponding to the x, z, and y axes respectively.

Response: The next gripper pose for the given task ’the bottle should be twenty centimeters from the ground.’ for timestep 1 is shown in the generated images.

B. Text Action model Prompt and Response

Prompt: Task: “pull the top drawer 50% open". Where should the robot move next? Format the robot’s gripper action as a relative 3D coordinate, an Euler rotation, and a binary gripper open/close state. All numbers are floats with two decimal places, each in relative coordinates.

Response: pos: [0.54, 0.42, 0.62], rot: [-1.57, -0.0, -1.57]

C. DETAILED ABLATION RESULTS AND ANALYSIS

A. Action Prediction Ablations

We experimented with two common alternative architectural choices for action generation in VLAs, both of which failed to learn a working policy under similar training and evaluation settings as for OG-VLA.

1) Text Action: is an architecture variant where the LLM also produces gripper’s next pose in the form of text akin to prior VLA models [1]. In general, the raw output sequence from the LLM can be of slightly different format and contain additional text (e.g. text like *the next robot action is:*), so long as it contains the information above. We apply regex parsing to extract from the text the gripper position p^{text}, orientation ω^{text}. The prompt used for this ablation is shown in Appendix B-B.

Predicting actions as text without any visual reasoning on the output end of the model results in an imprecise action prediction model. Li et al. [1] have shown this architecture to work when training with large-scale robot datasets. However, we find that with a small robotics dataset, it is hard for VLAs to learn precise control via direct text token prediction.

2) Action Tokens: is an architecture variant where the LLM produces special action tokens added to the LLM vocabulary, such as $[trans0]$ or $[rot0]$ for translation and rotation modalities, akin to that for the image modality. This architecture is closer to works that perform action tokenization [2]. We decode the hidden state vectors for these tokens using additional MLP decoders to predict 3D translation and rotation vectors.

The model still struggles to predict sufficiently precise actions to perform the tasks. This may be due to insufficient visual reasoning available for action decoding, as the decoders use the hidden states corresponding to the special token produced by the LLM. We also observed degeneration of LLM’s ability to produce coherent and grammatical text as the training progressed for this model design. This may have been caused by the training of new tokens and additional decoders failing to preserve original reasoning capabilities of the model, an issue that might be addressed by co-training with the pretraining datasets. We do not observe degeneration when

Model	Pickup Object	Reorient Object	Open Drawer	Close Drawer	Open Cabinet	Close Cabinet	Pour Water	Transfer Water	Overall
(1) No Reconstruction	76.7 ± 6.2	10.0 ± 10.8	3.3 ± 2.4	20.0 ± 0.0	1.7 ± 2.4	10.0 ± 7.1	28.3 ± 11.8	13.3 ± 12.5	20.4 ± 4.7
-Novel Object	56.7 ± 6.2	10.0 ± 0.0	3.3 ± 2.4	23.3 ± 15.5	0.0 ± 0.0	10.0 ± 8.2	13.3 ± 8.5	6.7 ± 2.4	15.4 ± 5.1
-Novel Scene	51.7 ± 8.5	6.7 ± 6.2	10.0 ± 4.1	10.0 ± 8.2	0.0 ± 0.0	5.0 ± 4.1	11.7 ± 6.2	6.7 ± 6.2	12.7 ± 3.0
-Novel State	0.0 ± 0.0	6.7 ± 6.2	6.7 ± 2.4	5.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	13.3 ± 8.5	0.0 ± 0.0	4.0 ± 2.1
(2) Reconstruction	86.7 ± 2.4	15.0 ± 7.1	38.3 ± 2.4	51.7 ± 2.4	0.0 ± 0.0	16.7 ± 2.4	25.0 ± 4.1	16.7 ± 6.2	31.2 ± 2.3
-Novel Object	85.0 ± 4.1	0.0 ± 0.0	1.7 ± 2.4	55.0 ± 10.8	0.0 ± 0.0	5.0 ± 4.1	18.3 ± 2.4	6.7 ± 6.2	21.5 ± 0.3
-Novel Scene	73.3 ± 2.4	1.7 ± 2.4	26.7 ± 9.4	36.7 ± 4.7	1.7 ± 2.4	1.7 ± 2.4	16.7 ± 9.4	8.3 ± 2.4	20.8 ± 1.1
-Novel State	0.0 ± 0.0	13.3 ± 6.2	13.3 ± 2.4	20.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	8.3 ± 6.2	13.3 ± 2.4	8.5 ± 1.6
(3) Faded Reconstruction	93.3 ± 2.4	5.0 ± 4.1	23.3 ± 2.4	36.7 ± 8.5	0.0 ± 0.0	5.0 ± 4.1	30.0 ± 0.0	20.0 ± 10.8	26.7 ± 2.3
-Novel Object	75.0 ± 4.1	18.3 ± 2.4	0.0 ± 0.0	55.0 ± 4.1	0.0 ± 0.0	8.3 ± 4.7	36.7 ± 11.8	5.0 ± 4.1	24.8 ± 2.3
-Novel Scene	76.7 ± 6.2	11.7 ± 6.2	28.3 ± 4.7	28.3 ± 2.4	5.0 ± 4.1	8.3 ± 2.4	20.0 ± 8.2	11.7 ± 6.2	23.8 ± 3.1
-Novel State	0.0 ± 0.0	18.3 ± 10.3	6.7 ± 2.4	10.0 ± 4.1	1.7 ± 2.4	0.0 ± 0.0	1.7 ± 2.4	11.7 ± 6.2	6.2 ± 0.5

TABLE III: Success rates for image generation modes for each task. We show that generating actions with reconstruction or with faded reconstructions work better than that without reconstruction.

Model	Pickup Object	Reorient Object	Open Drawer	Close Drawer	Open Cabinet	Close Cabinet	Pour Water	Transfer Water	Overall
OG-VLA	86.7 ± 2.4	15.0 ± 7.1	38.3 ± 2.4	51.7 ± 2.4	0.0 ± 0.0	16.7 ± 2.4	25.0 ± 4.1	16.7 ± 6.2	31.2 ± 2.3
-Novel Object	85.0 ± 4.1	0.0 ± 0.0	1.7 ± 2.4	55.0 ± 10.8	0.0 ± 0.0	5.0 ± 4.1	18.3 ± 2.4	6.7 ± 6.2	21.5 ± 0.3
-Novel Scene	73.3 ± 2.4	1.7 ± 2.4	26.7 ± 9.4	36.7 ± 4.7	1.7 ± 2.4	1.7 ± 2.4	16.7 ± 9.4	8.3 ± 2.4	20.8 ± 1.1
-Novel State	0.0 ± 0.0	13.3 ± 6.2	13.3 ± 2.4	20.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	8.3 ± 6.2	13.3 ± 2.4	8.5 ± 1.6
+Tiled Views	75.0 ± 7.1	6.7 ± 4.7	33.3 ± 2.4	28.3 ± 6.2	1.7 ± 2.4	20.0 ± 4.1	15.0 ± 10.8	18.3 ± 6.2	24.8 ± 0.3
-Novel Object	58.3 ± 8.5	15.0 ± 7.1	1.7 ± 2.4	30.0 ± 0.0	0.0 ± 0.0	10.0 ± 4.1	40.0 ± 4.1	6.7 ± 6.2	20.2 ± 1.3
-Novel Scene	60.0 ± 7.1	15.0 ± 10.8	45.0 ± 7.1	21.7 ± 6.2	5.0 ± 4.1	5.0 ± 4.1	26.7 ± 6.2	1.7 ± 2.4	22.5 ± 0.5
-Novel State	0.0 ± 0.0	10.0 ± 4.1	10.0 ± 7.1	18.3 ± 4.7	1.7 ± 2.4	1.7 ± 2.4	1.7 ± 2.4	16.7 ± 11.8	7.5 ± 2.6
-LLM	86.7 ± 2.4	5.0 ± 7.1	6.7 ± 4.7	33.3 ± 4.7	0.0 ± 0.0	6.7 ± 4.7	15.0 ± 0.0	6.7 ± 2.4	20.0 ± 1.5
-Novel Object	68.3 ± 6.2	1.7 ± 2.4	6.7 ± 9.4	40.0 ± 4.1	0.0 ± 0.0	3.3 ± 2.4	10.0 ± 4.1	8.3 ± 2.4	17.3 ± 1.6
-Novel Scene	71.7 ± 6.2	8.3 ± 6.2	18.3 ± 2.4	21.7 ± 8.5	3.3 ± 2.4	0.0 ± 0.0	15.0 ± 4.1	5.0 ± 4.1	17.9 ± 3.3
-Novel State	0.0 ± 0.0	0.0 ± 0.0	6.7 ± 2.4	16.7 ± 2.4	0.0 ± 0.0	1.7 ± 2.4	3.3 ± 2.4	10.0 ± 4.1	4.8 ± 0.8
+Tiled Views -LLM	71.7 ± 10.3	1.7 ± 2.4	13.3 ± 8.5	16.7 ± 4.7	0.0 ± 0.0	8.3 ± 2.4	15.0 ± 8.2	10.0 ± 0.0	17.1 ± 3.4
-Novel Object	56.7 ± 8.5	8.3 ± 2.4	1.7 ± 2.4	16.7 ± 8.5	0.0 ± 0.0	1.7 ± 2.4	15.0 ± 4.1	6.7 ± 2.4	13.3 ± 1.6
-Novel Scene	61.7 ± 6.2	5.0 ± 4.1	20.0 ± 4.1	11.7 ± 6.2	0.0 ± 0.0	3.3 ± 4.7	10.0 ± 0.0	10.0 ± 0.0	15.2 ± 1.2
-Novel State	0.0 ± 0.0	6.7 ± 2.4	10.0 ± 0.0	30.0 ± 4.1	1.7 ± 2.4	0.0 ± 0.0	6.7 ± 6.2	3.3 ± 4.7	7.3 ± 0.8
-Instruction to IG	71.7 ± 4.7	8.3 ± 2.4	20.0 ± 10.8	40.0 ± 12.2	1.7 ± 2.4	11.7 ± 9.4	5.0 ± 4.1	15.0 ± 4.1	21.7 ± 2.6
-Novel Object	66.7 ± 6.2	0.0 ± 0.0	1.7 ± 2.4	45.0 ± 4.1	0.0 ± 0.0	8.3 ± 2.4	20.0 ± 4.1	1.7 ± 2.4	17.9 ± 0.8
-Novel Scene	50.0 ± 8.2	11.7 ± 2.4	25.0 ± 0.0	26.7 ± 2.4	10.0 ± 0.0	11.7 ± 2.4	13.3 ± 4.7	5.0 ± 4.1	19.2 ± 2.1
-Novel State	0.0 ± 0.0	3.3 ± 4.7	8.3 ± 6.2	13.3 ± 8.5	0.0 ± 0.0	0.0 ± 0.0	1.7 ± 2.4	8.3 ± 2.4	4.4 ± 1.8

TABLE IV: Model ablation results for each task. We ablate components of OG-VLA to justify our design choices such as using tiled views, and the contribution of the LLM in the pipeline.

finetuning the model with original architectural components in OG-VLA without any co-training with other datasets. We leave the study with co-training for above architectural variants to future works.

B. Image Generation Mode

We study three image generation modes for action prediction: (1) Generation without reconstruction: an all black image background (2) Generation with reconstruction: an RGB image that is a reconstruction of the input image, and (3) Generation with faded reconstruction: a shifted RGB image between the range $[0, 128]$ that is a reconstruction of the input image. For each image mode, the action Gaussian distributions are overlaid on these backgrounds. These three choices carry tradeoffs. The first method is the purest way to represent actions, but it appears challenging for image generators pre-trained on generating color images to learn to generate uniform black backgrounds. The second method does not require un-learning generation of color images, but burdens the generator with the additional reconstruction task, which has the potential to take model capacity away from action generation. On the flipside, it has the potential for some positive cross-task transfer, and better scene understanding and visual reasoning. The third method is a middle ground between the other two methods, which we include in our study.

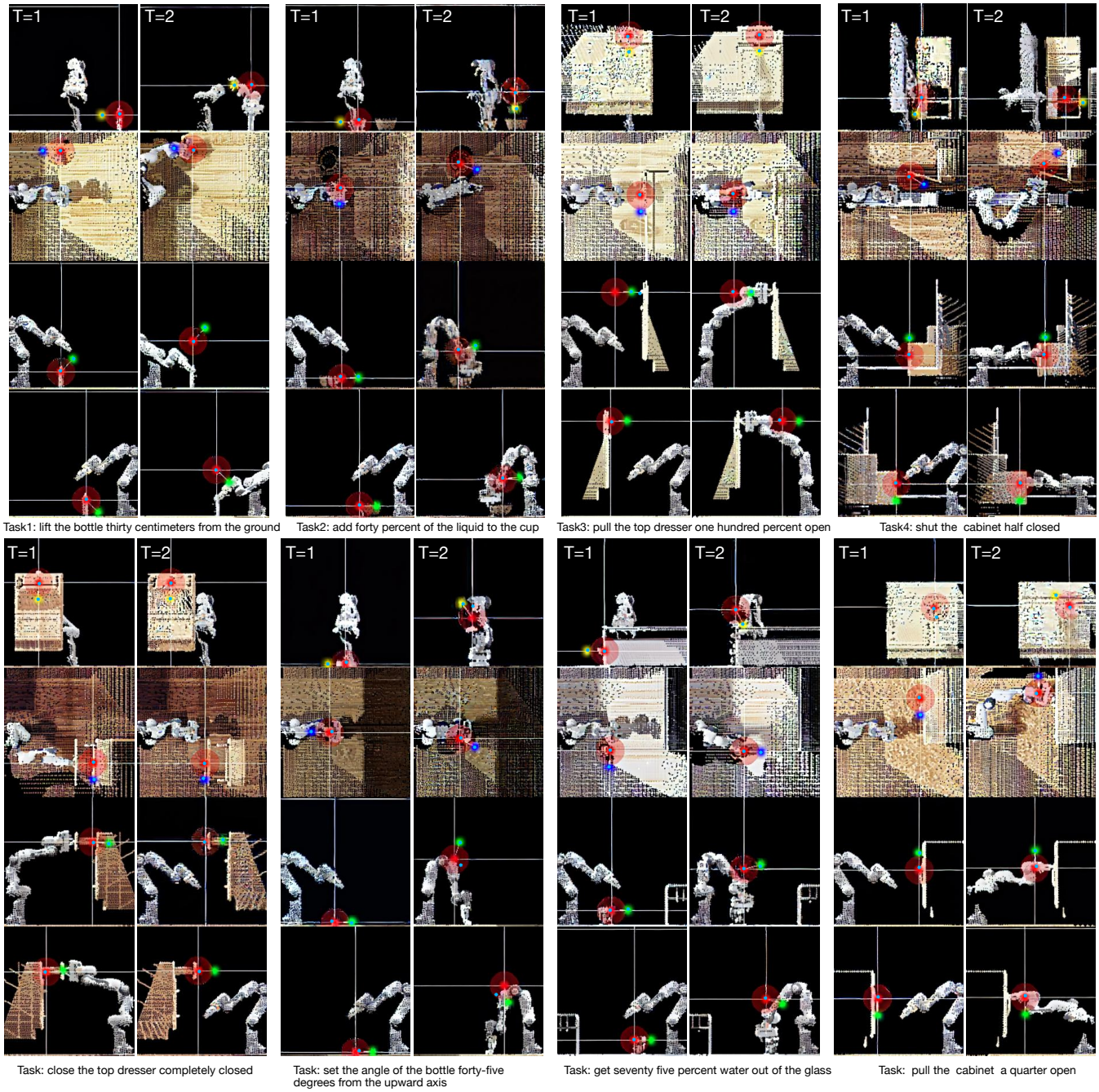


Fig. 1: Example gripper position and rotation outputs from OG-VLA for eight different tasks. The rows are the different views: front, top, left, and right. For each task, the two columns are two timesteps required to solve the task. The red Gaussian is the predicted position. The yellow, blue, and green Gaussians are predicted rotation angles along x, z, and y-axis respectively. The blue dot is our model's output gripper position, back-projected to each view. The dots on rotation Gaussians are showing the extracted pixel for computing the rotation angle in reference to the horizontal right axis in each view.

It eases action decoding by keeping values in the $(128, 255]$ range exclusively for action annotations. For methods (2) and (3), we apply an additional filtering step to identify the Gaussian and recover a grayscale heatmap.

We report success rates across tasks and test splits in Table III. Generation without reconstruction of the scene performs the worst of all generation modes. This may be due to forcing the IMAGEGENERATOR to unlearn it's prior of generating natural images and forcing it to only predict Gaussian distributions. We also observe instability in training this version, as sometimes the IMAGEGENERATOR would collapse to produce completely black or noisy images and stop generating actions on them. Generation with reconstruction and that with faded reconstruction did not show consistent difference over all test splits. Generation with reconstruction learns a policy that's better by 4.4% on Novel Pose split and 2.3% Novel State split. Generation with faded reconstruction performs better on Novel Object and Novel State splits by

	Pickup Object	Reorient Object	Open Drawer	Close Drawer	Open Cabinet	Close Cabinet	Pour Water	Transfer Water	Overall
Training set evaluation	76.7 $\pm$ 4.7	11.7 $\pm$ 2.4	35.0 $\pm$ 0.0	58.3 $\pm$ 4.7	0.0 $\pm$ 0.0	10.0 $\pm$ 4.1	18.3 $\pm$ 6.2	20.0 $\pm$ 4.1	28.8 $\pm$ 0.5
+ Ground Truth Translation	91.7 $\pm$ 2.4	21.7 $\pm$ 14.3	76.7 $\pm$ 6.2	81.7 $\pm$ 2.4	15.0 $\pm$ 7.1	21.7 $\pm$ 4.7	28.3 $\pm$ 2.4	33.3 $\pm$ 2.4	46.2 $\pm$ 0.9
+ Ground Truth Rotation	96.7 $\pm$ 4.7	60.0 $\pm$ 4.1	43.3 $\pm$ 6.2	75.0 $\pm$ 8.2	5.0 $\pm$ 0.0	40.0 $\pm$ 7.1	40.0 $\pm$ 8.2	11.7 $\pm$ 2.4	46.5 $\pm$ 3.8

TABLE V: Ablated evaluation of the model's translation and rotation prediction capabilities on a sampled training set of 20 episodes, similar to the test sets. In each evaluation, we individually ablate either the translation or rotation predictions to assess the model's prediction capabilities for each.

3.3% and 3%. Therefore, we conclude that these model variants have similar performance, and generation with faded reconstruction did not work better as we had hypothesized due to its balance between forcing the model to focus less on scene reconstruction and more on action prediction. Therefore, we report generation with reconstruction as our final method.

C. Model Ablations

We present ablations on OG-VLA's design choices in Table IV. The first result shows the effect of tiling the 4 orthographic views instead of feeding them to the IMAGEGENERATOR in batch of 4 as also explored in Genima [3]. For tiling, we stack the 4 views as in 2D array, following prior work. Tiling did not improve the performance for our model as opposed to results reported in the prior works. This may have occurred because the prior work did not have an LLM in the pipeline, so tiling becomes necessary for modeling interactions between views to generate multi-view consistent predictions. However, due to the LLM in OG-VLAconditioning generation in all views, there's sufficient interaction and reasoning between views through the predicted image tokens $\left(t_a^i, \quad i \in \{1, \ldots, 4\}\right)$.

The second experiment ablates the LLM to study its contribution to the overall performance. We remove the LLM from the system, directly passing image and text representations to the IMAGEGENERATOR. This result shows a drop in performance, highlighting the importance of the LLM in our pipeline.

Next, we study adding tiling to the previous LLM ablation to ensure that interaction between views, which was earlier happening through the LLM, can now take place in the IMAGEGENERATOR. Tiling further leads to a drop in performance, perhaps because of the reduction in the number of tokens used to represent each view. Without tiling, each image is represented by 256 patch tokens, but with tiling all 4 images are represented by 256 tokens in total, potentially creating too tight a representational bottleneck. This version is also similar to Genima with only an IMAGEGENERATOR and tiled camera input views in the pipeline.

Finally, we study the effect of bypassing the prompt and instruction into the IMAGEGENERATOR, shown in the last section of Table IV. The performance drop suggests that some instruction information may have been lost in image tokens output from the LLM, therefore it is beneficial to provide that information separately to the IMAGEGENERATOR for more accurate action prediction.

D. Qualitative results

We present qualitative results in Figure 1 of the output of OG-VLA, showing the generated actions, inferred gripper positions and rotations. We show predictions for the tasks: Pickup Object, Transfer Water, Open Drawer, and Close Cabinet. We add remaining task prediction examples in the Appendix. These visualizations show that OG-VLA can do complex translation and rotation tasks, that require both object and free space reasoning. We observe that the translation predictions are quite consistent for most cases, except for Task3 at T=1. The prediction in the Left view is incorrect, however, due to correct predictions in other views, the most likely 3D point (blue dot) is still correctly extracted at the handle of the drawer using the optimization in Equation **??**. In Task4, we show a failure case where the task is to half-close the cabinet, however the predicted point in T=2 is less than the half-way point.

E. Ablation with ground truth translation and rotation

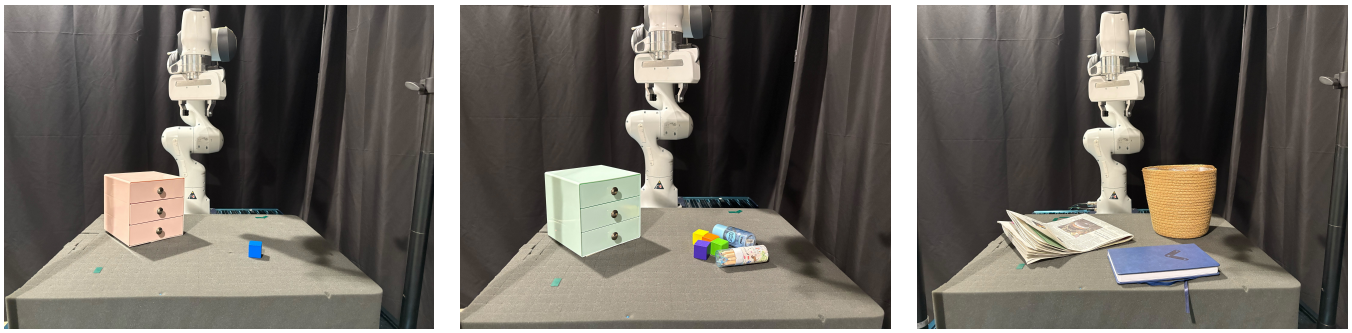
We present another set of ablations in Table V for studying our model's translation and rotation prediction abilities in an ablated setting on a sampled training set. First, we note that the overall performance of our model on Training and Novel Pose split is quite similar, which indicates that the model has not overfit and is generalizing well w.r.t the training performance. We observe that there is huge scope for improvement, both in predicting more precise translations and rotations from the last two rows of the table. We believe that this gap can be closed with training techniques like co-training with other datasets for better reasoning [4] and leveraging existing large robotics datasets like that in other VLA models [2]. We believe leveraging these datasets with a 3D-aware VLA framework can unlock better learning signals from these large-scale datasets and significantly improve results for our proposed VLA architecture.

D. REAL WORLD DETAILS

We calibrate the camera using MoveIt hand-eye calibration using an ArUco board [1]. We record the trajectory at 30 Hz frame rate. For training OG-VLA, we only use the first 1/5-th of the trajectory before each annotated keyframe action in the trajectory, for augmenting states prior to each keyframe.

Figure 2 shows the different training and test scenes. We provide evaluation videos of successful and failed trajectories for these test scenes on our website.

[1] https://github.com/moveit/moveit$_calibration$



(a) Training demonstration objects **(b)** Novel objects used at test time **(c)** Novel scene distractors used at test time

Fig. 2: Real-world setup: objects used at training and test-time. For novel scene, we additional vary the lighting and background by switching on/off external lighting source, and removing curtains.

REFERENCES

[1] X. Li, C. Mata, J. Park, K. Kahatapitiya, Y. S. Jang, et al., "LLaRA: Supercharging robot learning data for vision-language policy," in *International Conference on Learning Representations*, 2025.

[2] M. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, et al., "OpenVLA: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[3] M. Shridhar, Y. L. Lo, and S. James, "Generative image as action models," in *Proceedings of the 8th Conference on Robot Learning (CoRL)*, 2024.

[4] W. Yuan, J. Duan, V. Blukis, W. Pumacay, R. Krishna, A. Murali, A. Mousavian, and D. Fox, "RoboPoint: A vision-language model for spatial affordance prediction for robotics," *arXiv preprint arXiv:2406.10721*, 2024.