GROUNDING FORCE: IMAGE-ALIGNED CONTACT AND FORCE PREDICTION FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) models inherit semantic understanding from pretrained vision-language models and map observations and instructions directly to robot actions. The success of an action, however, also depends on the physical interaction, where the robot makes contact with an object and how it applies force. A VLA trained only to imitate actions never represents this physical interaction explicitly. Prior work trains the VLA to predict targets that describe the scene rather than the interaction, or measures the interaction with force or tactile sensors, which most robots and datasets lack. We present Grounding Force (GF), which trains a VLA to predict, alongside its next action, the contact and force that action will produce, as a contact map and force maps on the wrist-camera image. This supervision is used only in training, so the policy needs no force or tactile sensor at test time. Across four backbones (π_0, $\pi_{0.5}$, GR00T N1.6, and OpenVLA-OFT) on RoboCasa and RoboTwin 2.0, Grounding Force consistently improves the base VLAs, by up to 7.3% in success rate, and helps more than predicting future images, depth, or semantics. With pseudo-labels from a labeler trained only in simulation, it also improves $\pi_{0.5}$ by 13.0% on both real single-arm and bimanual robots. These results indicate that predicting the physical interaction in the image frame, without any sensor at test time, is an effective supervision signal for VLAs. Code and videos: `https://groundingforce.github.io`.

1 INTRODUCTION

Vision-language-action (VLA) models have emerged as a leading approach to general-purpose robotic manipulation, integrating visual perception, language understanding, and action generation in a single network (Brohan et al., 2023; Kim et al., 2025b; Black et al., 2024; Bjorck et al., 2025). Most VLAs fine-tune a vision-language model (VLM) pretrained on web-scale image–text data to predict robot actions from camera images, conditioned on a language instruction. This pretraining gives them a strong semantic understanding of the scene. Knowing which object to act on and where it is, however, is not always enough. Success also depends on the physical interaction an action produces, where the robot makes contact and how it applies force (Mason, 2001; Lynch & Mason, 1996; Bicchi & Kumar, 2000).

Standard VLAs do not represent this interaction explicitly. They learn by imitating demonstrated actions (Brohan et al., 2023; Kim et al., 2025b; Black et al., 2024), and these labels specify where the end-effector should move, not where it should make contact or how much force it should apply (Liu et al., 2025b; Yu et al., 2026). Fig. 1a shows such a failure, where a base VLA places a bowl into a stack of two bowls. The held bowl is slightly off center, so its rim hits the edge of the stack instead of going inside. The policy keeps pressing down, and this force pushes a bowl out of the stack and flips it over. A shift of only a few millimeters in the gripper's position decides whether the bowl goes inside or hits the edge. An imitation loss penalizes such a small error only slightly, whether or not it leads to failure. Nothing in the objective tells the policy that this is the difference that matters.

Predicting an intermediate target alongside the actions is known to help VLAs without any additional sensing, with targets such as keypoints, future images, or depth (Nasiriany et al., 2025; Zhao et al., 2025; Zhang et al., 2026). These targets describe the scene, where to act or how it will look, but

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

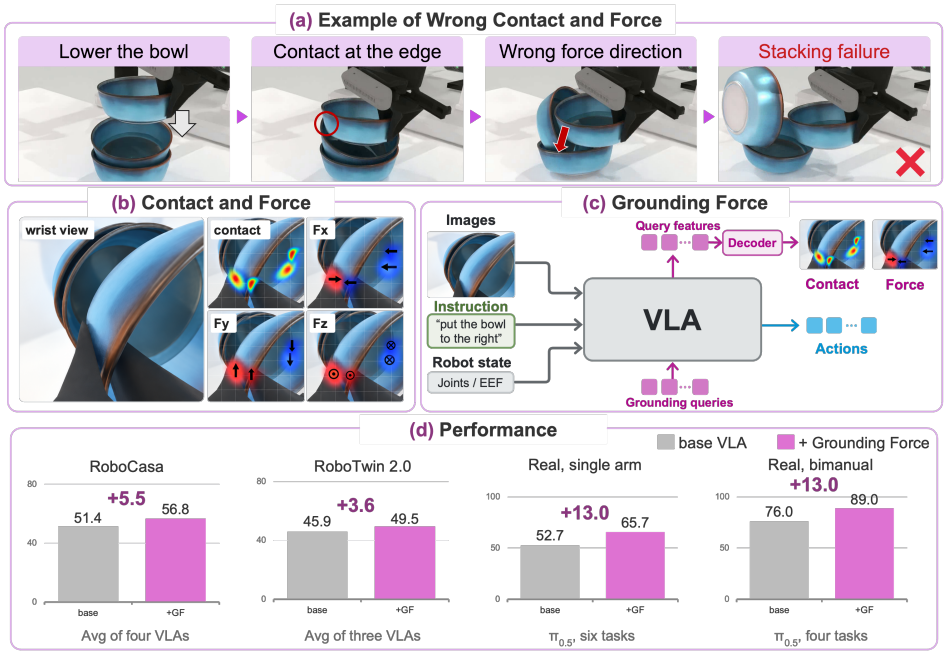


Figure 1: **Grounding Force (GF).** (a) A base VLA stacking bowls in RoboTwin 2.0. The rim of the held bowl hits the edge of the stack, and the policy keeps pressing down until a bowl is pushed out of the stack and flips over. (b) Grounding Force learns to predict where contact occurs and how force acts, as a contact map and one signed force map per axis over the wrist image. (c) Grounding query tokens pass through the VLA with its inputs, and a decoder turns their features into the maps during training only. (d) Success rate (%) with and without Grounding Force.

not the contact and force of the interaction (Sec. 2). Work on force and tactile sensing shows that this information is valuable. Policies that receive force or tactile signals improve on contact-rich manipulation (Liu et al., 2025b; Hou et al., 2025; Xue et al., 2025; He et al., 2025; Yu et al., 2026; Huang et al., 2025; Bi et al., 2026; Jones et al., 2025), and force can also be estimated at deployment by a model trained on sensor data (Zhao et al., 2026). These methods rely on sensors, at least when the training data are collected, and most robots and datasets do not have them.

We therefore investigate whether a VLA can learn contact and force as a prediction target, inferred from the inputs it already has. Such a target should be compact, describing only what the action must realize, and aligned with the observation the policy conditions on. We propose Grounding Force (GF), a method that trains a VLA to predict, jointly with its actions, the contact and force that determine the outcome of an interaction—between the robot and objects, and between a held object and its surroundings. The key design choice is the coordinate frame in which contact and force are predicted. We ground contact and force in the image coordinates of the policy's wrist-camera observation, as a contact map that marks the region of interaction and three signed force maps that give the x, y, and z components of the force (Fig. 1b). A set of grounding query tokens, processed by the VLM backbone alongside the visual and language tokens, predicts these maps and is read by the action head, so that actions are decoded from a representation trained to encode the interaction (Fig. 1c). The supervision is privileged, obtained from simulation and used only in training, so no sensor is required. At deployment the policy adds only nine query tokens and negligible overhead.

Our contributions are threefold:

- We propose contact and force as a prediction target for VLAs, expressed as maps aligned with the wrist-camera image, labeled in simulation and pseudo-labeled on real robots without any force or tactile sensor.
- We introduce Grounding Force, which learns this target through grounding query tokens in the VLM backbone that the action head also reads. It applies to four VLAs with different action heads, with negligible inference overhead.

- Extensive experiments with four VLAs (π_0, $\pi_{0.5}$, GR00T N1.6, and OpenVLA-OFT) (Black et al., 2024; 2025; Bjorck et al., 2025; Kim et al., 2025a) on RoboCasa (Nasiriany et al., 2024) and RoboTwin 2.0 (Chen et al., 2025) and on real single-arm and bimanual robots show that Grounding Force consistently improves the base VLAs, most on tasks where contact decides the outcome (Fig. 1d). Ablations further show that it outperforms future-frame targets such as pixels, image features, depth, and semantics, and that both the force maps and their alignment with the wrist image are needed for the gain (Sec. 4.3).

2 Related Work

Vision-Language-Action Models. VLAs build on VLMs pretrained on web-scale image–text data and fine-tune them on robot data to map observations and language to actions. They differ primarily in how actions are decoded. RT-2 and OpenVLA discretize actions into tokens and decode them autoregressively (Brohan et al., 2023; Kim et al., 2025b). π_0 and $\pi_{0.5}$ add an action expert trained with flow matching (Black et al., 2024; 2025), GR00T N1.6 conditions a diffusion transformer on the VLM's features (Bjorck et al., 2025), and OpenVLA-OFT predicts a chunk of continuous actions in parallel (Kim et al., 2025a). Regardless of the action decoder, these models are trained mainly to imitate demonstrated actions, which provide no direct supervision on where objects are, how the scene will evolve, or what comes next. To address this, recent studies have trained VLAs to predict intermediate targets alongside actions, such as affordances and keypoints (Nasiriany et al., 2025; Yuan et al., 2025; Tang et al., 2025; Fang et al., 2024; Li et al., 2025), 2D or 3D point tracks (Gu et al., 2024; Wen et al., 2023; Bharadhwaj et al., 2024; Kim et al., 2026), future images or their features (Wu et al., 2024; Zhao et al., 2025; Wang et al., 2026; Cen et al., 2025; Chen et al., 2026; Zheng et al., 2025), depth and semantics (Zhang et al., 2026), or language subtasks (Zawalski et al., 2024; Lin et al., 2026), or to align their features with those of a 3D foundation model (Li et al., 2026). Grounding Force extends these approaches with a target that describes the interaction itself—where the action makes contact and how it applies force.

Contact and Force in VLAs. Force and tactile signals have been brought into manipulation policies through force-informed imitation, compliance, and reactive control (Liu et al., 2025b; Hou et al., 2025; Xue et al., 2025; He et al., 2025), and into VLAs through force-aware experts, tactile encoders, and language-grounded sensor fusion (Yu et al., 2026; Huang et al., 2025; Bi et al., 2026; Jones et al., 2025). These methods feed force or tactile signals to the policy as separate inputs fused with its visual features, so these signals are not tied to locations in the camera image, and the sensor is needed at deployment. Prior work also infers force from the observations, but learns it from sensor measurements and outputs a global quantity. FD-VLA predicts the reading of a force–torque sensor from the observations, a single reading that aggregates all contacts (Zhao et al., 2026). ForceSight predicts one grip force and one applied force for a goal end-effector pose, defined by a wrist-mounted force–torque sensor (Collins et al., 2024). In contrast, Grounding Force learns contact and force from simulation, so it needs no sensor for data collection or deployment, and predicts the force at each contact as maps on the wrist-camera image, at the location where the contact occurs in the policy's own view.

3 Method

3.1 Problem Formulation

VLA policies. We consider a VLA policy that maps an observation $o_t = (\{I_t^m\}_{m=1}^M, q_t)$, consisting of M camera images and the proprioceptive state, together with a language instruction ℓ, to an action chunk $A_t \in \mathbb{R}^{H \times d_a}$ of horizon H. We write any such policy $\pi_\theta(A_t \mid o_t, \ell)$ in terms of a VLM backbone f_θ and an action head g_ϕ,

$$\mathbf{x}_t = \mathcal{T}(o_t, \ell), \qquad \mathbf{h}_t = f_\theta(\mathbf{x}_t), \qquad \hat{A}_t = g_\phi(\mathbf{h}_t), \tag{1}$$

where $\mathcal{T}$ tokenizes the observation and the instruction into visual, language, and proprioceptive tokens, and g_ϕ is trained by imitation on demonstrations $\{(o_t, \ell, A_t)\}$ with a backbone-specific loss $\mathcal{L}_{\text{act}}$ — flow matching for π_0, $\pi_{0.5}$, and GR00T N1.6, and L_1 regression for OpenVLA-OFT. Eq. (1) writes the four backbones in a common form, in which $\mathbf{h}_t$ denotes the representations the backbone

computes over the input tokens and g_ϕ the module that maps them to actions. The design of this module differs across the backbones (Sec. 4.1).

The prediction target. In addition to the action, Grounding Force supervises the policy to predict the contact and force its next action will produce. Both are expressed as maps over the wrist-camera image. Writing I_t^{w} for that image among the M views, and $p = (i, j) \in \mathcal{P}$ for the integer coordinates of its $P = |\mathcal{P}|$ patches, the target is

$$Y_{t+1} = \big(C_{t+1},\, F^x_{t+1},\, F^y_{t+1},\, F^z_{t+1}\big) \in \mathbb{R}^{P\times 4}, \tag{2}$$

where the contact map $C_{t+1} \in [0,1]^P$ marks where the interaction occurs and the force maps $F^x_{t+1}, F^y_{t+1}, F^z_{t+1} \in [-1,1]^P$ give the three signed components of the contact force there.

3.2 Image-Aligned Contact and Force Maps

Predicting in the image plane. Contact and force could also be predicted as 3D quantities in the world frame, as contact points and force vectors. We instead express them as maps over the wrist-camera image, the view among the policy's inputs that shows the gripper and the objects it manipulates. Two considerations favor image-plane maps over world-frame quantities. First, a pretrained VLM represents each image as a grid of patch tokens (Dosovitskiy et al., 2021), and aligning contact and force with this grid lets the maps build on the VLM's visual representation. Second, an image-plane target is defined in the camera's own view. A world-frame target also needs the depth of each contact and the current camera pose. The model is given neither, so it would have to learn to map its view to world coordinates from the world-frame labels alone.

Contacts and forces. Labels are computed automatically by replaying each demonstration in the simulator. At step $t{+}1$ we label every contact between the robot and an object it is grasping or pushing, which we call a manipulated object, and every contact between a manipulated object and its surroundings, as when a held object is placed or inserted. Writing $\mathcal{S}_{t+1}$ for this set of contacts, each contact $k \in \mathcal{S}_{t+1}$ has a 3D position $\mathbf{r}_k$ and a contact force $\mathbf{f}_k \in \mathbb{R}^3$ in newtons acting on the manipulated object, both in the world frame. Contacts are projected with no occlusion test, so a contact inside the field of view is labeled even when hidden, such as the underside of a held object, while contacts outside it are dropped.

Aligning contact and force with the image. We represent contacts as heatmaps, the standard choice in keypoint localization (Tompson et al., 2014; Newell et al., 2016), because they handle a number of contacts that varies from step to step. Each contact $k \in \mathcal{S}_{t+1}$ is therefore rendered as a 2D Gaussian $w_k(p)$ on the patch grid, centered at the contact's image position in patch units, $\mathbf{u}_k = \Pi(\mathbf{r}_k)/s$, where Π is the wrist camera's projection from world coordinates to pixels and s is the patch width in pixels, so $\mathbf{u}_k$ and p share the same coordinates. At every patch $p \in \mathcal{P}$, the contact map takes the maximum over the Gaussians of all contacts, so it stays in $[0,1]$ even where the Gaussians of nearby contacts overlap:

$$w_k(p) = \exp\!\left(-\frac{\|p - \mathbf{u}_k\|^2}{2\sigma^2}\right), \qquad C_{t+1}(p) = \max_{k\in\mathcal{S}_{t+1}} w_k(p), \tag{3}$$

where $\sigma = 0.8$ patch widths. To keep the direction of the force, each axis has its own signed map, with x pointing right and y up in the image and z toward the camera, and each contact's force is transformed from the world frame into these axes. At every patch p, the force maps sum the Gaussians of all contacts weighted by their force vectors, giving the net force at that patch. The sum is then compressed by the symmetric logarithm (symlog), which keeps light contacts visible beside heavy ones, and normalized by the largest force expected along each axis in each dataset (Appendix A):

$$\mathbf{F}_{t+1}(p) = \frac{\mathrm{symlog}\big(\sum_{k\in\mathcal{S}_{t+1}} w_k(p)\,\mathbf{f}_k\big)}{\mathrm{symlog}\big(\mathbf{F}^{\max}\big)}, \qquad \mathrm{symlog}(v) = \mathrm{sign}(v)\log(1+|v|) \text{ for } v \in \mathbb{R}, \tag{4}$$

where $\mathbf{F}_{t+1} = (F^x_{t+1}, F^y_{t+1}, F^z_{t+1})$, $\mathbf{F}^{\max}$ collects these per-axis values, symlog and the division act on each axis separately, and the result is clipped to $[-1,1]$.

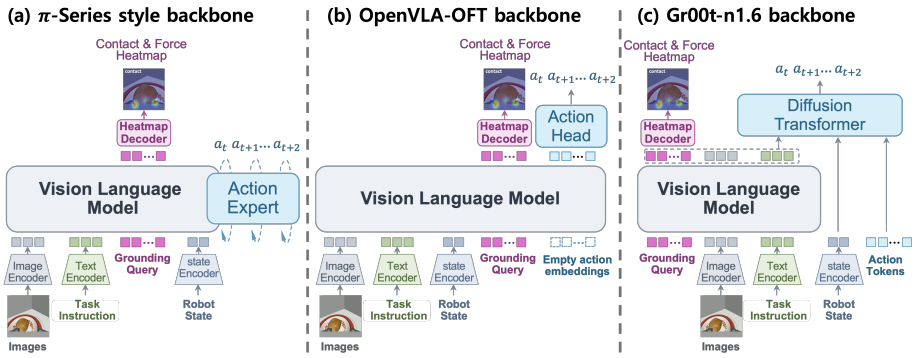


Figure 2: **How the grounding queries are integrated into each base VLA.** The queries (magenta) join the token sequence of the VLM, and a lightweight decoder turns their outputs into the maps during training only. (a) In π_0 and $\pi_{0.5}$ the action expert attends to the VLM prefix, queries included. (b) In OpenVLA-OFT the action tokens pass through the same VLM, and the action head decodes them. (c) In GR00T N1.6 the query outputs join the features the diffusion transformer conditions on. Four queries are drawn, and we use nine.

Pseudo-labels for real robots. Real demonstrations come without contact and force labels. We therefore train a labeler on the simulated RoboCasa demonstrations to predict Y_{t+1} from (o_t, ℓ). The labeler is a $\pi_{0.5}$ model trained with Grounding Force, with the action head removed. The labeler is applied to the collected real demonstrations without training on real data, and its predictions serve as pseudo-labels for fine-tuning the real-robot policy. This transfer needs no world frame shared between simulation and the real robot, since the maps are defined in the wrist-camera image.

3.3 Grounding Force

Grounding queries. To infer Y_{t+1} from the inputs the policy already has, we append Q learnable *grounding query* tokens $Z \in \mathbb{R}^{Q\times d}$ to the input of the VLM backbone. Through attention, the queries gather the information needed to predict the contact and force maps into a few tokens, as learnable queries do in DETR, BLIP-2 and DreamVLA (Carion et al., 2020; Li et al., 2023; Zhang et al., 2026). Fig. 2 shows how the queries are integrated into each of the four base VLAs. Their output representation $\bar{Z}_t$ is read both by a lightweight decoder D_ψ, which produces the maps, and by the action head:

$$[\,\mathbf{h}_t\,;\,\bar{Z}_t\,] = f_\theta\big([\,\mathbf{x}_t\,;\,Z\,]\big), \qquad \hat{Y}_{t+1} = D_\psi(\bar{Z}_t), \qquad \hat{A}_t = g_\phi(\mathbf{h}_t,\,\bar{Z}_t). \tag{5}$$

Here $[\,\cdot\,;\,\cdot\,]$ denotes concatenation along the token dimension. The queries never attend to action tokens, so $\bar{Z}_t$ depends only on the inputs of the VLM backbone, never on the actions. The decoder D_ψ receives only $\bar{Z}_t$, not the image tokens. The maps can therefore be predicted only if $\bar{Z}_t$ encodes the contact and force, and the action head reads this same $\bar{Z}_t$.

Heatmap decoder. The decoder D_ψ is a lightweight two-layer transformer that decodes the four maps of Eq. (2) from the Q query outputs. It is identical for all four base VLAs, and on a bimanual robot it produces one set of maps per arm (Appendices B, C and G).

3.4 Training and Inference

Objective. The model is trained end-to-end with

$$\mathcal{L} = \lambda_a\,\mathcal{L}_{\text{act}} + \lambda_m\,\mathcal{L}_{\text{map}}, \qquad \mathcal{L}_{\text{map}} = \tfrac{1}{4P}\big\|\hat{Y}_{t+1} - Y_{t+1}\big\|_2^2, \tag{6}$$

where $\mathcal{L}_{\text{act}}$ is the base VLA's own action loss and $\mathcal{L}_{\text{map}}$ is the mean squared error of the maps, averaged over both arms on a bimanual robot. The weights λ_a and λ_m, and a small regularizer on the decoder, are given in Appendix B.

Inference. At deployment the heatmap decoder is discarded, and the policy runs with no force or tactile input. The Q query tokens, the only addition to the base VLA, are computed in the backbone's existing forward pass and add about 1% to the inference time of π_0 (Appendix D).

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate on two simulated benchmarks, RoboCasa (Nasiriany et al., 2024) for single-arm and RoboTwin 2.0 (Chen et al., 2025) for bimanual manipulation. RoboCasa comprises 24 atomic kitchen tasks in seven categories defined by the benchmark, from pick-and-place to insertion. We train on its Human-50 split of 1,184 human demonstrations and evaluate in the in-distribution setting, from the initial states of the benchmark's demonstrations (Appendix E). RoboTwin 2.0 comprises 50 tasks, of which we use 18 that require both arms, grouped into four categories (Appendix E). We train on 50 scripted demonstrations per task and evaluate 100 rollouts per task on held-out scenes in its clean setting, without domain randomization. On both benchmarks we report the mean success rate over tasks and the success rate of each category.

Base VLAs. We apply Grounding Force to four base VLAs that cover three ways of connecting an action head to its VLM. In π_0 (Black et al., 2024) and $\pi_{0.5}$ (Black et al., 2025) the action expert shares self-attention with the VLM, in GR00T N1.6 (Bjorck et al., 2025) a diffusion transformer cross-attends to it, and OpenVLA-OFT (Kim et al., 2025a) decodes actions in parallel inside one bidirectional network. Each is fine-tuned from its released weights with its own released recipe, and the model with Grounding Force is trained identically except for the added queries and decoder. Each base VLA and its Grounding Force counterpart are evaluated at the same training step, chosen in advance. Appendix D gives the training details of each, and Appendices B and A the hyperparameters of Grounding Force.

We do not report GR00T N1.6 on RoboTwin, where the released model did not reach a usable success rate in our runs.

4.2 MAIN RESULTS IN SIMULATION

Grounding Force improves the average success rate of every base VLA on both benchmarks (Tables 1 and 2), by 2.17% to 7.26% absolute on RoboCasa and 1.34% to 5.39% on RoboTwin. The gain does not depend on a favorable checkpoint, since on OpenVLA-OFT every checkpoint with Grounding Force from 60k steps on outperforms the best checkpoint of the base VLA (Appendix F).

RoboCasa. As shown in Table 1, the gains are largest in the categories where contact decides the outcome. For every base VLA, the largest category gain falls on Pressing buttons, Insertion or Open/Close doors, reaching 14.77% for π_0 on Pressing buttons. Success in these categories depends on where the gripper or the held object makes contact and how much force it applies, which is what the maps supervise. Strong base VLAs benefit as well. $\pi_{0.5}$, which has the highest base success rate, gains as much as OpenVLA-OFT, which has the lowest. GR00T N1.6 gains least and is the only base VLA that loses in any category, on Open/Close doors and drawers. Its smaller gain is consistent with its released recipe, which fine-tunes only a few layers of the VLM.

RoboTwin. As shown in Table 2, the gains carry over to RoboTwin, which is evaluated on held-out scenes, so they do not come only from fitting the training scenes. The largest category gain of $\pi_{0.5}$ and OpenVLA-OFT is on Object arrangement, 11.83% for $\pi_{0.5}$, where blocks and bowls must be stacked or set down in place. The only category loss is OpenVLA-OFT on Bimanual lifting, 3.50% over its two tasks, or 7 of 200 rollouts.

4.3 ABLATION STUDIES

We ablate Grounding Force on π_0 and RoboCasa. We first investigate what the queries should predict (Fig. 3), and then how force is represented, where the queries are placed and how many are needed (Fig. 4).

Table 1: **Results on RoboCasa.** Average success rate (%) over the 24 tasks and per category (names shortened). Bold marks the better of each base VLA and its GF version, and parentheses give the change. GR00T and OFT denote GR00T N1.6 and OpenVLA-OFT, and Appendix E gives per-task results.

	Average	Pick & place	Doors	Drawers	Knobs	Levers	Buttons	Insertion
π_0	53.78	36.64	54.68	87.00	45.00	82.07	67.93	32.40
π_0 **+ GF**	**59.09** (+5.31)	**39.11** (+2.47)	**61.20** (+6.52)	**90.70** (+3.70)	**51.55** (+6.55)	**82.10** (+0.03)	**82.70** (+14.77)	**40.75** (+8.35)
$\pi_{0.5}$	66.49	59.41	66.38	85.20	49.75	85.17	81.47	42.60
$\pi_{0.5}$ **+ GF**	**73.59** (+7.10)	**68.45** (+9.04)	**68.23** (+1.85)	**94.45** (+9.25)	**55.35** (+5.60)	**92.57** (+7.40)	**85.17** (+3.70)	**56.45** (+13.85)
GR00T	55.41	41.30	**52.83**	**93.50**	49.60	75.93	71.63	29.65
GR00T **+ GF**	**57.58** (+2.17)	**44.54** (+3.24)	52.35 (−0.48)	89.80 (−3.70)	**51.55** (+1.95)	**79.03** (+3.10)	**77.17** (+5.54)	**32.40** (+2.75)
OFT	29.83	7.89	41.30	52.75	28.10	47.53	43.83	25.90
OFT **+ GF**	**37.09** (+7.26)	**16.94** (+9.05)	**53.33** (+12.03)	**63.85** (+11.10)	**29.05** (+0.95)	**50.63** (+3.10)	**44.43** (+0.60)	**35.15** (+9.25)

Table 2: **Results on RoboTwin 2.0.** Average success rate (%) over the 18 bimanual tasks and per category (our grouping, Appendix E), on held-out scenes, with the notation of Table 1. GR00T N1.6 is not reported (Sec. 4.1).

	Average	Arrangement	Lifting	Coordination	Pick & place
π_0	50.83	31.50	46.00	48.33	69.86
π_0 **+ GF**	**52.17** (+1.34)	**34.00** (+2.50)	**47.50** (+1.50)	**49.33** (+1.00)	**70.29** (+0.43)
$\pi_{0.5}$	63.72	39.50	60.50	65.33	84.71
$\pi_{0.5}$ **+ GF**	**69.11** (+5.39)	**51.33** (+11.83)	**66.00** (+5.50)	**67.00** (+1.67)	**86.14** (+1.43)
OFT	23.11	11.17	**44.50**	37.00	21.29
OFT **+ GF**	**27.11** (+4.00)	**18.83** (+7.66)	41.00 (−3.50)	**39.67** (+2.67)	**24.86** (+3.57)

What to predict. As shown in Fig. 3, predicting contact and force gains 5.31%, 1.6 times as much as the best future-frame target. The future-frame targets follow those of prior VLAs, each predicting the next third-person frame, as pixels (Zhao et al., 2025), as the SigLIP features the VLM reads (Zheng et al., 2025), or as depth and semantics (Zhang et al., 2026). Among them the SigLIP features gain the most and pixels the least, likely because the third-person view is also an input and its next frame is close to a copy of the current one. All targets are read from the same grounding queries by the same decoder, use simulator labels where they need them, and use a tuned loss weight (Appendix D), so they differ only in what they predict and on which view (Sec. 3.2). Contact alone gains nothing, so the added queries do not by themselves explain the gain. The gain appears only once force is added. We hypothesize that the image already shows where contact occurs, but not the direction and magnitude of the force, which decide whether a press or an insertion succeeds. Together with the result for pixels, this suggests that a target helps most when it adds what the image does not show, as force does, rather than repeating what it shows, as pixels and contact locations do.

How to represent force. As shown in Fig. 4a, predicting the same contacts and forces as 3D points and vectors in the world frame, instead of as wrist-image maps, falls 1.06% below the base model. This is likely because such a target requires the depth of each contact and the camera pose, which the model is not given (Sec. 3.2). Keeping the direction of the force also helps, since one signed map per axis gains 1.91% more than a single merged force map. These results suggest that contact and force are most useful as maps on the wrist-camera image that keep the direction of the force.

Where to place the queries, and how many. Fig. 4b shows that our placement of the grounding queries in the VLM backbone gains 5.31%, about twice as much as placing them in the action expert (2.80%). In both placements the queries attend to the image and instruction tokens, but only in the VLM backbone are they processed by layers with vision–language pretraining. This suggests that the grounding queries benefit from the pretrained knowledge of the VLM, which the action expert lacks. Fig. 4c shows that nine queries give the best result among the numbers we tried. Both fewer and more queries gain less, and 36 queries lose the gain entirely. Like the size of any network trained on limited data, the number of queries likely has a suitable range. Too few queries may lack the capacity to encode the maps, while too many may add redundant tokens that compete for attention.

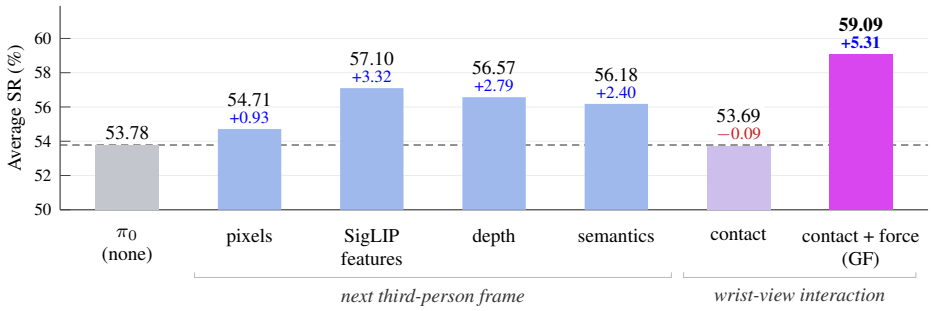


Figure 3: **What to predict.** Average success rate (%) on RoboCasa over the 24 tasks, with the change from π_0 (dashed line) under each value. The future-frame targets predict the next third-person frame as pixels, as SigLIP features (the image tokens the VLM reads, as in FLARE), or as depth or semantics. The interaction targets are maps on the wrist view.

(a) How to represent force

Force	Frame	SR	Δ_{π_0}
π_0		53.78	–
merged	image	57.18	+3.40
per-axis	3D world	52.72	−1.06
per-axis	image	**59.09**	+5.31

(b) Where to place the queries

Placement	SR	Δ_{π_0}
π_0	53.78	–
action expert	56.58	+2.80
VLM backbone	**59.09**	+5.31

(c) How many queries

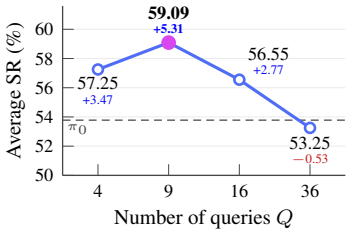


Figure 4: **Design choices of Grounding Force** on π_0 and RoboCasa. Average success rate (%) over the 24 tasks, and its change from π_0. In (a), force is one merged map or one signed map per axis, on the wrist image or as 3D world-frame vectors. Grounding Force is highlighted.

4.4 Real-World Experiments

Setup. We evaluate Grounding Force on a real Franka Panda and on a bimanual setup of two Franka Pandas. The single-arm setup uses one wrist camera and two scene cameras, and the bimanual setup one wrist camera per arm and one scene camera (Appendix H). We evaluate six single-arm and four bimanual tasks, shown in Fig. 5a and described in Appendix H, which range from inserting a gear and closing a zipper to lifting a box with both arms. Each task has 50 demonstrations, and each policy is evaluated 50 times per task with randomized object placement. The contact and force maps are pseudo-labels from the labeler of Sec. 3.2. On the bimanual setup, each arm's wrist view is labeled separately. We fine-tune $\pi_{0.5}$ on each task with and without these maps and otherwise identically.

Results. Fig. 5b shows that Grounding Force carries over to real robots without any force or tactile sensor. With pseudo-labels from a labeler trained only in simulation, it improves $\pi_{0.5}$ on all ten tasks and raises the average success rate by 13.0% on both setups. As in simulation, the largest gains are on tasks where the held object must be pressed against another, such as inserting the gear (20.0%), setting up the coffee machine and opening the beer bottle. The labels also transfer from one simulated arm to two real ones, since each arm's wrist view is labeled separately, and the bimanual tasks gain as much as the single-arm ones. The smallest gain, 2.0% on closing the zipper, coincides with the weakest pseudo-labels of the ten tasks.

Pseudo-labels. Without ground truth on the real robot, we examine the pseudo-labels on the wrist images (Fig. 5c). For the gear, the beer bottle and the Lego box, the contact map lies on the two fingertips, and the predicted forces point inward as in a grip. For the bag zipper, the contact map is not active while the fingers hold the slider, so the pseudo-labels miss the grasp. This suggests that the real-world gain depends on the quality of the pseudo-labels, which Appendix H evaluates against the gripper state on all ten tasks. In every episode we inspected, however, the labels stay at

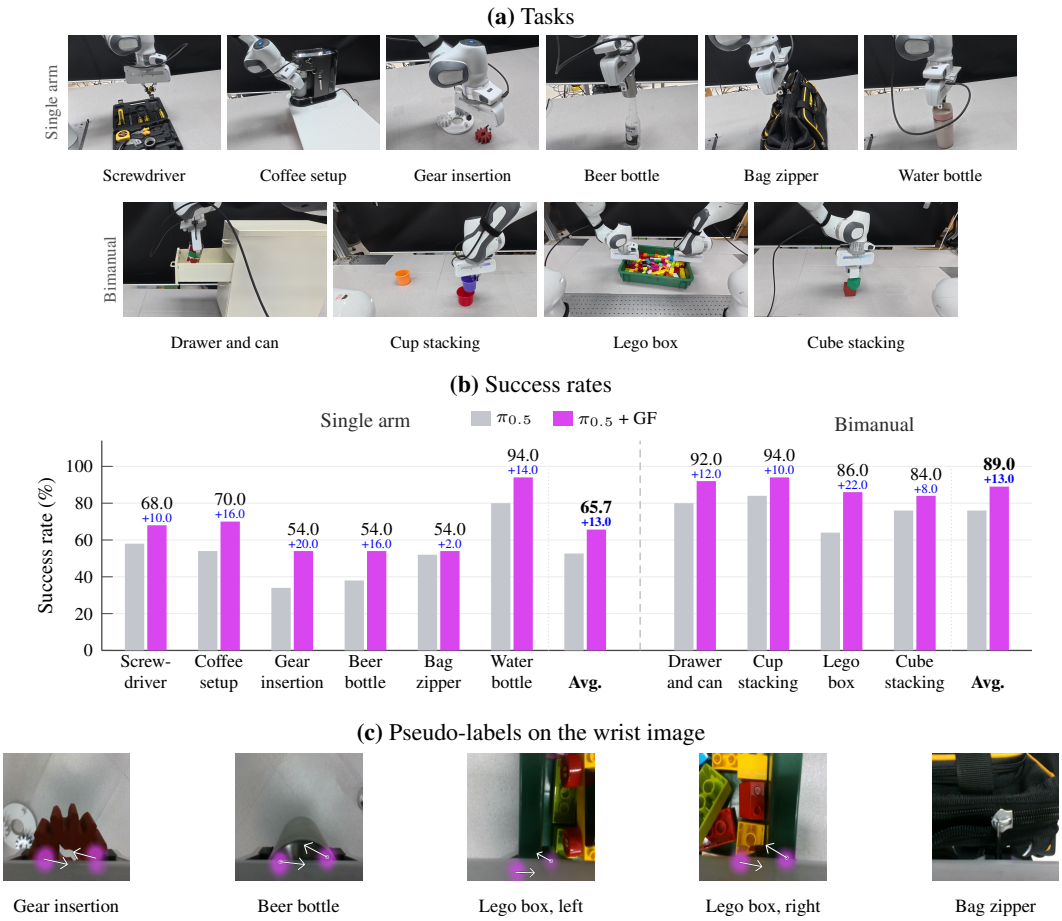


Figure 5: **Real-world experiments** on one Franka Panda (single arm) and on two (bimanual), without any force or tactile sensor. (a) The ten tasks, one representative frame each. (b) Success rate (%) of $\pi_{0.5}$ with and without Grounding Force, over 50 trials per task. (c) Pseudo-labels of the RoboCasa-trained labeler at a typical closed-finger frame, with the contact map in magenta and the in-plane force as arrows (a dot marks a component toward the camera).

the fingertips, so on the real robot they mark only the grasp, not the contact between the held object and the scene (Appendix H). Even so, these grasp-only pseudo-labels improve $\pi_{0.5}$ on both setups, which suggests that contact with the held object is already a useful target on its own.

5 Conclusion and Limitations

We propose Grounding Force, a method that trains a VLA to predict contact and force as maps aligned with its wrist-camera image. At deployment the policy needs no force or tactile sensor and adds negligible overhead. Across four backbones and two simulated benchmarks, Grounding Force consistently improves the base VLAs and, in our ablations, helps more than predicting future images, depth, or semantics. With pseudo-labels from a labeler trained only in simulation, it also improves $\pi_{0.5}$ on all ten single-arm and bimanual tasks on real robots. Our work has two limitations. First, on the real robot the pseudo-labels mark only the grasp, not the contact between a held object and the scene. Second, they fail on deformable objects, which do not appear in RoboCasa. Both could be addressed by training the labeler on more varied contacts, including deformable objects, or by fine-tuning it on a few real demonstrations recorded with a force sensor.

AI USE STATEMENT

We used generative AI tools in this work. An LLM-based coding agent assisted in writing the code of our method and experiments, and all AI-generated code was reviewed, tested and revised by the authors. An LLM assisted in drafting and editing the text, in preparing figures and tables, and in searching for related work. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The main paper describes the method, the labels and the objective in Sec. 3 and the benchmarks, base VLAs and evaluation protocol in Sec. 4.1. The appendix gives the details needed to reproduce the results: Appendix A the rules that select and normalize the contact and force labels, Appendix B the decoder architecture and the full objective with its weights, Appendix C the bimanual setting, Appendix D the optimization settings and the batch size, training steps and action chunk of every run, Appendix E the training data, evaluation protocols and per-task results on both benchmarks, and Appendix H the real-robot hardware, data collection, task definitions and pseudo-label checks. We will release the code, the training and evaluation configurations, and the label-generation scripts, together with rollout videos, on the project page linked in the abstract.

REFERENCES

Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision (ECCV)*, 2024.

Jianxin Bi, Kevin Yuchen Ma, Ce Hao, Mike Shou Zheng, and Harold Soh. VLA-Touch: Enhancing vision-language-action model with dual-level tactile feedback. *IEEE Robotics and Automation Letters*, 2026.

Antonio Bicchi and Vijay Kumar. Robotic grasping and contact: A review. In *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, volume 1, pp. 348–353. IEEE, 2000.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/black25a.html.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, pp. 213–229, 2020.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, et al. WorldVLA: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

Jiayi Chen, Wenxuan Song, Pengxiang Ding, Ziyang Zhou, Han Zhao, Barrett Tang, Donglin Wang, and Haoang Li. Unified Diffusion VLA: Vision-language-action model via joint discrete denosing diffusion process. In *International Conference on Learning Representations (ICLR)*, 2026.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

Jeremy A. Collins, Cody Houff, You Liang Tan, and Charles C. Kemp. ForceSight: Text-guided mobile manipulation with visual-force goals. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10874–10880, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

Kuan Fang, Fangchen Liu, Pieter Abbeel, and Sergey Levine. MOKA: Open-world robotic manipulation through mark-based visual prompting. *Robotics: Science and Systems (RSS)*, 2024.

Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, Priya Sundaresan, Peng Xu, Hao Su, Karol Hausman, Chelsea Finn, Quan Vuong, and Ted Xiao. RT-trajectory: Robotic task generalization via hindsight trajectory sketches. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F1TKzG8LJO.

Zihao He, Hongjie Fang, Jingjing Chen, Hao-Shu Fang, and Cewu Lu. Foar: Force-aware reactive policy for contact-rich robotic manipulation. *IEEE Robotics and Automation Letters*, 10(6):5625–5632, 2025. doi: 10.1109/LRA.2025.3560871.

Yifan Hou, Zeyi Liu, Cheng Chi, Eric Cousineau, Naveen Kuppuswamy, Siyuan Feng, Benjamin Burchfiel, and Shuran Song. Adaptive compliance policy: Learning approximate compliance for diffusion guided control. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4829–4836, 2025.

Jialei Huang, Shuo Wang, Fanqi Lin, Yihang Hu, Chuan Wen, and Yang Gao. Tactile-VLA: unlocking vision-language-action model's physical knowledge for tactile generalization. *arXiv preprint arXiv:2507.09160*, 2025.

Joshua Jones, Oier Mees, Carmelo Sferrazza, Kyle Stachowicz, Pieter Abbeel, and Sergey Levine. Beyond sight: Finetuning generalist robot policies with heterogeneous sensors via language grounding. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5961–5968. IEEE, 2025.

Jisoo Kim, Jungbin Cho, Sanghyeok Chu, Ananya Bal, Jinhyung Kim, Gunhee Lee, Sihaeng Lee, Seung Hwan Kim, Bohyung Han, Hyunmin Lee, et al. PRI4R: Learning world dynamics for vision-language-action models with privileged 4D representation. *arXiv preprint arXiv:2603.01549*, 2026.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025a.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. In *International Conference on Learning Representations*, volume 2026, pp. 132324–132345, 2026.

Jinming Li, Yichen Zhu, Zhibin Tang, Junjie Wen, Minjie Zhu, Xiaoyu Liu, Chengmeng Li, Ran Cheng, Yaxin Peng, Yan Peng, and Feifei Feng. CoA-VLA: Improving vision-language-action models via visual-text chain-of-affordance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9759–9769, October 2025.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning (ICML)*, pp. 19730–19742, 2023.

Fanqi Lin, Ruiqian Nai, Yingdong Hu, Jiacheng You, Junming Zhao, and Yang Gao. OnetwoVLA: A unified vision-language-action model with adaptive reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=tWMfhoP3as.

Jason Jingzhou Liu, Yulong Li, Kenneth Shaw, Tony Tao, Ruslan Salakhutdinov, and Deepak Pathak. FACTR: Force-attending curriculum training for contact-rich policy learning. *arXiv preprint arXiv:2502.17432*, 2025a.

Wenhai Liu, Junbo Wang, Yiming Wang, Weiming Wang, and Cewu Lu. ForceMimic: Force-centric imitation learning with force-motion capture system for contact-rich manipulation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1105–1112. IEEE, 2025b.

Kevin M Lynch and Matthew T Mason. Stable pushing: Mechanics, controllability, and planning. *The international journal of robotics research*, 15(6):533–556, 1996.

Matthew T Mason. *Mechanics of robotic manipulation*. MIT press, 2001.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems (RSS)*, 2024.

Soroush Nasiriany, Sean Kirmani, Tianli Ding, Laura Smith, Yuke Zhu, Danny Driess, Dorsa Sadigh, and Ted Xiao. RT-Affordance: Affordances are versatile intermediate representations for robot manipulation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8249–8257. IEEE, 2025.

Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision (ECCV)*, pp. 483–499, 2016.

Grace Tang, Swetha Rajkumar, Yifei Zhou, Homer Rich Walke, Sergey Levine, and Kuan Fang. KALIE: Fine-tuning vision-language models for open-world manipulation without robot data. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9507–9515. IEEE, 2025.

Jonathan Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1799–1807, 2014.

Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. In *International Conference on Learning Representations*, volume 2026, pp. 80929–80944, 2026.

Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.

Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *International Conference on Learning Representations*, volume 2024, pp. 10641–10662, 2024.

Han Xue, Jieji Ren, Wendi Chen, Gu Zhang, Yuan Fang, Guoying Gu, Huazhe Xu, and Cewu Lu. Reactive diffusion policy: Slow-fast visual-tactile policy learning for contact-rich manipulation. In *Proceedings of Robotics: Science and Systems (RSS)*, 2025.

Jiawen Yu, Hairuo Liu, Qiaojun Yu, Jieji Ren, Ce Hao, Haitong Ding, Guangyu Huang, Guofan Huang, Yan Song, Panpan Cai, et al. Forcevla: Enhancing vla models with a force-aware moe for contact-rich manipulation. *Advances in Neural Information Processing Systems*, 38:93409–93439, 2026.

Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction in robotics. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4005–4020. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/yuan25c.html.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=S70MgnIA0v.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *Advances in Neural Information Processing Systems*, 38: 24195–24228, 2026.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1702–1713. IEEE, 2025.

Ruiteng Zhao, Wenshuo Wang, Yicheng Ma, Xiaocong Li, Francis EH Tay, Marcelo H Ang Jr, and Haiyue Zhu. Fd-vla: Force-distilled vision-language-action model for contact-rich manipulation. *arXiv preprint arXiv:2602.02142*, 2026.

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loïc Magne, Avnish Narayan, You Liang Tan, Guanzhi Wang, Qi Wang, Jiannan Xiang, Yinzhen Xu, Seonghyeon Ye, Jan Kautz, Furong Huang, Yuke Zhu, and Linxi Fan. FLARE: Robot learning with implicit world modeling. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 3952–3971. PMLR, 2025.

A Contact and Force Labels

Which contacts are labeled. Sec. 3.2 labels the contacts between the robot and the manipulated object and between the manipulated object and its surroundings. Three further rules apply. Contacts between the robot and its surroundings are also labeled while the robot is in contact with an object. Contacts of the robot with itself are dropped. In tasks that open or close a fixture, the fixture is excluded from the robot's contact set. For a contact without a manipulated object, the force is the one the robot applies. Fig. 6 shows examples of these labels.

Patch grids and axes. The patch grid of the wrist image is 16×16 ($P = 256$) for the 128×128 images of RoboCasa, 12×16 ($P = 192$) for the 96×128 images of RoboTwin 2.0, and 16×16 for the 224×224 images of the real robots. Each contact force is transformed into the image axes with the wrist camera pose of that step. The clipping in Eq. (4) keeps the rare force spikes of the simulator from dominating the loss.

Normalization. $F^{\max}$ is set for each dataset near the 95th percentile of $|F|$ along each axis, so that nearly all forces fall inside $[-1, 1]$ and only the rare spikes above it are clipped. We use $F^{\max} = (25, 12, 25)$ N for (x, y, z) on RoboCasa and $(52.6, 9.4, 25.2)$ N on RoboTwin 2.0, whose bimanual tasks involve larger horizontal forces. The real-robot pseudo-labels follow the RoboCasa normalization, since they are produced by the labeler trained on RoboCasa.

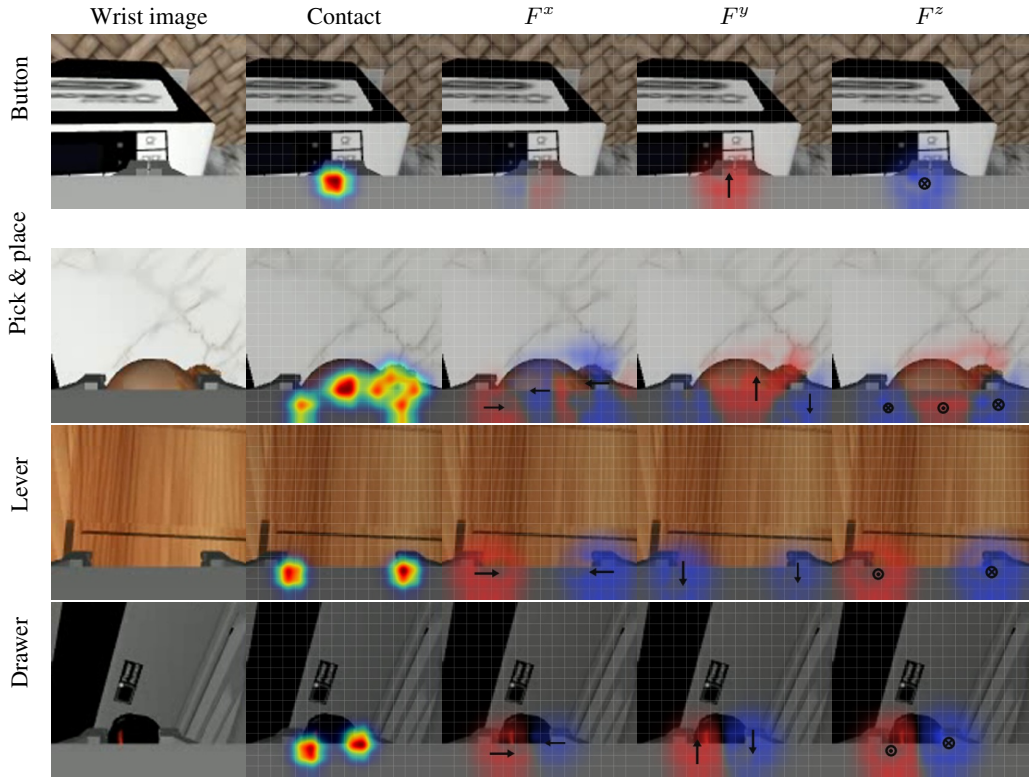


Figure 6: **Examples of contact and force labels** on RoboCasa, drawn as in Fig. 1b: pressing a button, picking and placing an object, turning a lever and opening a drawer. The labels mark the contacts between the fingertips and the object and, when a held object is placed, the contacts between the object and the counter, as in the pick-and-place example.

B Heatmap Decoder and Objective

Two details of the decoder matter for the objective. The first is how the four maps are computed from its axis tokens (Eq. (7)). The second is how those axis tokens are regularized (Eq. (8)).

Architecture. The decoder D_ψ is a transformer encoder of width $D = 2048$ with two layers, eight heads and an MLP ratio of four. Its input sequence has three parts, all of width D. The first part is the $Q = 9$ query outputs $\bar{Z}_t$ of Eq. (5), projected to width D and given 2D sine–cosine position embeddings on a 3×3 grid. The second part is $4K = 16$ learnable axis tokens, $K = 4$ for each of the four channels (contact, F^x, F^y, F^z). The third part is P learnable patch tokens with 2D sine–cosine position embeddings on the patch grid, which gives 281 tokens in total for $P = 256$. The two layers attend across all three parts, so the axis and patch tokens gather information from the query outputs. After the two layers, the decoder keeps the output vectors at the axis-token positions, $\mathbf{e}_{k,a} \in \mathbb{R}^D$, and at the patch positions, $\mathbf{h}_p \in \mathbb{R}^D$, and discards those at the query positions. The map value of channel a at patch p is the average cosine similarity between $\mathbf{h}_p$ and the K axis tokens of that channel, scaled and passed through a $\tanh$:

$$\hat{Y}_a(p) = \tanh\Big(s \cdot \frac{1}{K} \sum_{k=1}^{K} \cos\left(\mathbf{h}_p, \mathbf{e}_{k,a}\right)\Big), \tag{7}$$

where s is a learnable scalar. Without it the average similarity, which lies in $[-1, 1]$, would give outputs only in $[-\tanh(1), \tanh(1)] \approx [-0.76, 0.76]$. The scale lets the output span the full label range $[-1, 1]$. In other words, each channel has K learned reference vectors. A patch takes a positive value where its output points the same way as these references, a negative value where it points the opposite way, and zero where it is orthogonal to them.

Objective. The objective used in all runs is Eq. (6) with one added term, a regularizer on the decoder's axis tokens,

$$\mathcal{L} = \lambda_a \mathcal{L}_{\text{act}} + \lambda_m \mathcal{L}_{\text{map}} + \lambda_d \mathcal{L}_{\text{dir}}, \qquad \mathcal{L}_{\text{dir}} = \frac{1}{4K} \sum_{a=1}^{4} \sum_{k=1}^{K} \left\| W \mathbf{e}_{k,a} - \mathbf{d}_a \right\|_2^2. \tag{8}$$

$\mathcal{L}_{\text{act}}$ and $\mathcal{L}_{\text{map}}$ are those of Eq. (6). The added term $\mathcal{L}_{\text{dir}}$ trains neither the maps nor the policy. It acts only on the axis tokens. A learnable linear layer $W \in \mathbb{R}^{3\times D}$, shared by all axis tokens, projects each axis-token output $\mathbf{e}_{k,a}$ of Eq. (7) to three dimensions. The projection is pulled toward one target per channel, shared by its K tokens. The target $\mathbf{d}_a$ is the unit vector of the channel's axis in the image frame of Sec. 3.2, in which x points right, y up and z toward the camera. It is $(1, 0, 0)$ for F^x, $(0, 1, 0)$ for F^y and $(0, 0, 1)$ for F^z, and $(0, 0, 0)$ for the contact channel, which has no direction. This keeps the reference vectors of the three force channels tied to distinct axes while leaving the other $D - 3$ dimensions free. With $\lambda_d = 0.05$ the term is a weak regularizer. We use $\lambda_a = 5.0$ and $\lambda_m = 1.0$ in every run.

C Bimanual Robots

On a bimanual robot, each arm has its own wrist camera and its own set of maps. The maps of an arm are labeled in its wrist view from the contacts of that arm and of the object it manipulates. On both RoboTwin 2.0 and the real bimanual robot, the two arms share the same nine grounding queries, as in the single-arm setting, and the decoder produces the maps of each arm from their outputs. $\mathcal{L}_{\text{map}}$ is computed for each arm and averaged. On the real bimanual robot the pseudo-labels are produced for each wrist view separately by the single-arm labeler of Sec. 3.2.

D Training and Inference Details

Fine-tuning. We follow the released fine-tuning recipe of each base VLA (Black et al., 2024; 2025; Bjorck et al., 2025; Kim et al., 2025a) and add only the grounding queries and the decoder. Table 3 lists the optimizer, learning-rate schedule, precision and denoising steps of each base VLA, and Table 4 the batch size, training steps and action chunk of each run.

Table 3: Optimization settings of each base VLA, shared by the base model and the model with Grounding Force.

	π_0 / $\pi_{0.5}$	GR00T N1.6	OpenVLA-OFT
Initialization	π_0 / $\pi_{0.5}$ base	GR00T-N1.6-3B	OpenVLA-7B
Tuned parameters	all	top LLM layers	LoRA, rank 32
Optimizer	AdamW (0.9, 0.95)	AdamW	AdamW (0.9, 0.999)
Weight decay	10^{-10}	10^{-5}	0.01
Peak learning rate	2.5×10^{-5}	10^{-4}	10^{-4} / 2×10^{-4} *
Schedule	cosine†	cosine over 100k	$\times 0.1$ at 50k
Warm-up	1,000 steps	5% of steps	none
Precision	bfloat16	bfloat16‡	bfloat16
Denoising steps	10	4	– (L1 regression)

* RoboCasa / RoboTwin 2.0. † To 2.5×10^{-6} at 30k steps, then constant. ‡ The heatmap decoder runs in fp32.

Ablation targets. The future-frame targets of Sec. 4.3 predict the next third-person frame. Each is predicted per patch of the 16×16 grid by the decoder of Appendix B, which is kept identical, including its sequence of 281 tokens, and only the output head and the target change. The pixel target is the $8 \times 8 \times 3$ block of each patch (192 values). The SigLIP target is the 2048-dimensional image token that the VLM itself computes for the frame after its projection layer. The target is detached, so no gradient flows into the image encoder through it. Both targets are normalized to zero mean and unit variance within each patch and regressed with a mean squared error. The depth target is the

Table 4: Settings of each run. The training steps are those of the evaluated checkpoint, and the action chunk is the number of actions predicted per inference.

Benchmark	Model	Batch size	Training steps	Action chunk
RoboCasa	π_0	32	40k	50
	$\pi_{0.5}$	32	40k	50
	GR00T N1.6	64	50k	50
	OpenVLA-OFT	32	100k	16
RoboTwin 2.0	π_0	64	40k	20
	$\pi_{0.5}$	64	40k	20
	OpenVLA-OFT	32	100k	50
Real robots	$\pi_{0.5}$	32	40k	50

8×8 block of each patch of the 128×128 depth image, regressed with a mean squared error on normalized log-depth, with far-clipped pixels masked. The semantic target predicts the proportion of seven simulator classes in each patch with a soft cross-entropy. These variants use no direction loss ($\lambda_d = 0$), and each uses its best auxiliary loss weight among 0.05, 0.2 and 1.0.

Inference cost. Table 5 measures the cost of the queries on π_0. The nine queries are appended to the prefix, which the backbone encodes once per inference, so across the ten denoising steps they add only nine cached key-value entries. The inference time is averaged over 300 inferences of each model on one B200 GPU.

Table 5: Inference cost of the grounding queries on π_0. One inference produces one action chunk.

	Prefix tokens	Inference time
π_0	816	33.55 ms
π_0 + GF	825	33.93 ms
Overhead	+9	+0.38 ms (1.1%)

E Benchmark Details and Per-Task Results

Training data. On RoboCasa we train on the human demonstrations selected by the benchmark's 50-demonstration training split. Each task provides 53 or 54 demonstrations, of which the split selects 50. In 21 tasks, one or two of the selected demonstrations are missing from the released data, and two tasks (`TurnOffSinkFaucet`, `TurnOnStove`) have no split and use all of their demonstrations. The training set therefore contains 1,184 demonstrations, between 48 and 54 per task. On RoboTwin 2.0 we train on 50 scripted demonstrations per task. Images are 128×128 on RoboCasa and 96×128 on RoboTwin, from three cameras on each.

Evaluation. On RoboCasa each policy is evaluated from the initial state of every released demonstration, 53 or 54 per task and 1,293 in total, with a limit of 700 steps, and success is determined by the benchmark's success criterion for each task. These initial states include those of the training demonstrations. On RoboTwin 2.0 each policy is evaluated for 100 episodes per task with the `aloha-agilex` embodiment. The evaluation uses the clean setting without domain randomization and the seen-instruction setting. The evaluation scenes are generated by the benchmark's seed search and differ from the training scenes. GR00T N1.6 is not evaluated on RoboTwin (Sec. 4.1), since a gain over a base VLA that does not solve the tasks would not isolate the effect of Grounding Force.

Categories. RoboCasa's 24 tasks follow the benchmark's seven categories. RoboTwin 2.0 does not group its tasks. We assign each of the 18 bimanual tasks to one of four categories, as listed in Table 7.

Per-task results. Table 6 gives the per-task results on RoboCasa. Grounding Force improves 70 of the 96 task results of the four base VLAs, ties 9 and falls behind on 17. Table 7 gives the per-task results on RoboTwin 2.0, where Grounding Force improves 26 of the 54 task results of the three base VLAs, ties 11 and lowers 17. The losses are small, 11 of them within 3 points and none above 10, while 8 of the gains exceed 10 points, up to 66 points on stack_blocks_two with $\pi_{0.5}$. Several ties come from tasks that no model solves, such as blocks_ranking_rgb and stack_blocks_three.

F Robustness across Training Checkpoints

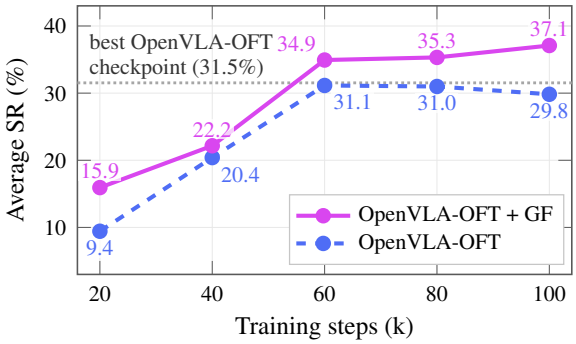


Figure 7: **Success rate across training.** OpenVLA-OFT on RoboCasa, averaged over the 24 tasks, evaluated every 10k steps and shown every 20k steps. The 100k points are those of Table 1, and the dotted line marks the best of all evaluated checkpoints of the base VLA, at 70k.

Fig. 7 evaluates OpenVLA-OFT on RoboCasa every 10k steps. Once both runs have plateaued, from 60k steps on, every checkpoint with Grounding Force (34.93% to 37.48%) is above the best checkpoint of the base VLA (31.53%, at 70k). Even with each run at its own best checkpoint, Grounding Force leads by 5.95%.

G Predicted Maps in Simulation

Figs. 8 and 9 show the maps that the grounding queries predict during simulated rollouts of the base VLAs with Grounding Force, at a moment of contact. In all four models with Grounding Force, the predicted contact map concentrates on the fingertips in contact with the object, and the predicted F^x at the two fingertips mostly points inward, as in a grip. On RoboTwin 2.0 the maps of each arm follow its own wrist view, and each arm's contact appears when that arm grasps its block. Figs. 10 to 12 show the rollouts these frames come from, next to a rollout of the base VLA on the same task, in the scene camera and the wrist camera.

H Real-World Setup and Pseudo-Labels

Hardware. Fig. 13 shows the two setups. The single-arm setup is a Franka Panda with an Intel RealSense D435 on the wrist and two RealSense D455 scene cameras. The bimanual setup has two Franka Pandas, with a D435 on each wrist and one RealSense L515 scene camera. All cameras are recorded at 224×224 and 30 Hz, and the robot state, recorded at 143 Hz, is resampled to the camera clock. Demonstrations are collected by leader–follower teleoperation with FACTR (Liu et al., 2025a). Actions are absolute joint positions, converted to deltas during training, and the gripper action is binary.

Tasks. The ten tasks are defined as follows.

- **Screwdriver** (single arm): pick up a screwdriver lying on the table and place it into the toolbox.
- **Coffee setup** (single arm): pick up a mug and place it under the spout of a coffee machine.
- **Gear insertion** (single arm): pick up a red gear and insert it onto the free pin of a gear board, which requires aligning the gear's bore with the pin.

Table 6: Per-task success rate (%) on RoboCasa. Bold marks the better of each base VLA and its GF version.

		π_0		$\pi_{0.5}$		GR00T N1.6		OpenVLA-OFT	
Category	Task	base	+ GF	base	+ GF	base	+ GF	base	+ GF
Pick and place	PnPCabToCounter	22.2	**25.9**	**57.4**	48.1	29.6	**31.5**	7.4	**16.7**
	PnPCounterToCab	48.1	**50.0**	64.8	**72.2**	57.4	**59.3**	3.7	**14.8**
	PnPCounterToSink	40.7	**46.3**	**66.7**	61.1	25.9	**29.6**	5.6	**13.0**
	PnPMicrowaveToCounter	25.9	**38.9**	40.7	**75.9**	29.6	**33.3**	5.6	**13.0**
	PnPCounterToStove	32.1	**33.3**	67.9	**75.5**	**43.4**	39.6	3.8	**9.4**
	PnPStoveToCounter	46.3	**59.3**	81.5	**90.7**	63.0	**64.8**	14.8	**31.5**
	PnPCounterToMicrowave	**35.2**	22.2	35.2	**55.6**	35.2	**42.6**	3.7	**13.0**
	PnPSinkToCounter	**42.6**	37.0	61.1	**68.5**	46.3	**55.6**	18.5	**24.1**
Open/close doors	CloseSingleDoor	90.7	90.7	**90.7**	85.2	87.0	87.0	70.4	**85.2**
	CloseDoubleDoor	55.6	**61.1**	55.6	**68.5**	**48.1**	44.4	37.0	**48.1**
	OpenSingleDoor	66.7	**70.4**	85.2	85.2	63.0	**66.7**	33.3	**57.4**
	OpenDoubleDoor	5.7	**22.6**	34.0	34.0	**13.2**	11.3	**24.5**	22.6
Open/close drawers	CloseDrawer	98.1	98.1	100.0	100.0	**100.0**	98.1	79.6	**87.0**
	OpenDrawer	75.9	**83.3**	70.4	**88.9**	**87.0**	81.5	25.9	**40.7**
Twisting knobs	TurnOffStove	31.5	**38.9**	25.9	**33.3**	**40.7**	38.9	**22.2**	20.4
	TurnOnStove	58.5	**64.2**	73.6	**77.4**	58.5	**64.2**	34.0	**37.7**
Turning levers	TurnOnSinkFaucet	64.8	**66.7**	75.9	**88.9**	66.7	**70.4**	46.3	46.3
	TurnOffSinkFaucet	**87.0**	83.3	85.2	**94.4**	72.2	**74.1**	61.1	**77.8**
	TurnSinkSpout	94.4	**96.3**	94.4	94.4	88.9	**92.6**	**35.2**	27.8
Pressing buttons	TurnOffMicrowave	77.8	**87.0**	90.7	90.7	70.4	**75.9**	**66.7**	37.0
	TurnOnMicrowave	63.0	**81.5**	74.1	**79.6**	66.7	**74.1**	18.5	**37.0**
	CoffeePressButton	63.0	**79.6**	79.6	**85.2**	77.8	**81.5**	46.3	**59.3**
Insertion	CoffeeSetupMug	22.2	**27.8**	27.8	**48.1**	16.7	**27.8**	14.8	**22.2**
	CoffeeServeMug	42.6	**53.7**	57.4	**64.8**	**42.6**	37.0	37.0	**48.1**
Average		53.78	**59.09**	66.49	**73.59**	55.41	**57.58**	29.83	**37.09**

- **Beer bottle** (single arm): pick up a bottle opener, fit its opening over the cap of a beer bottle and press down to pop the cap off.
- **Bag zipper** (single arm): grasp the slider of a bag's zipper and pull it along the track to close the bag.
- **Water bottle** (single arm): grasp the lid of a water tumbler, twist it open and lift it off.
- **Drawer and can** (bimanual): open a drawer and place a can inside it.
- **Cup stacking** (bimanual): stack three cups in the center in a given color order (red, orange, purple).
- **Lego box** (bimanual): pull a box of Lego pieces toward the center with one arm, place a Lego piece into it with the other, and lift the box with both arms.
- **Cube stacking** (bimanual): stack three cubes in the center in a given color order (red, green, white).

Figs. 14 and 15 show one rollout of each task with its instruction, edited here for grammar. The policy receives the original wording.

Pseudo-label checks. The real robots have no contact or force ground truth, so Fig. 16 checks the pseudo-labels of Sec. 4.4 against the gripper. A frame counts as having contact when the peak of its contact map exceeds 0.5. The fingers count as closed when the gripper is at least 15 mm narrower than the 95th percentile of its width in that episode, and as open within 5 mm of it, excluding 1 s around each transition. The threshold is set per episode because the closed width depends on the object, 61 mm on the can and 3 mm on the zipper slider. The margin is the fraction of closed-finger frames with contact minus that of open-finger frames. It measures how specifically the pseudo-labels respond to a grasp, since contact should be present while the fingers hold an object and absent while they are open. A margin of 1 means that contact is predicted in every closed-finger frame and in no

Table 7: Per-task success rate (%) on RoboTwin 2.0, over 100 rollouts per task on held-out scenes, grouped into the categories we assign. Bold marks the better of each base VLA and its GF version.

		π_0		$\pi_{0.5}$		OpenVLA-OFT	
Category	Task	base	+ GF	base	+ GF	base	+ GF
Object arrangement	blocks_ranking_rgb	0	0	0	0	0	0
	blocks_ranking_size	34	**41**	67	**73**	**26**	23
	stack_blocks_three	0	0	0	0	0	0
	stack_blocks_two	**1**	0	8	**74**	0	**1**
	stack_bowls_three	59	**70**	64	64	6	**28**
	stack_bowls_two	**95**	93	**98**	97	35	**61**
Bimanual lifting	grab_roller	91	**95**	98	98	**89**	79
	lift_pot	**1**	0	23	**34**	0	**3**
Coordinated manipulation	handover_mic	**89**	88	97	**98**	**80**	77
	hanging_mug	13	**16**	22	**29**	7	**26**
	scan_object	43	**44**	**77**	74	**24**	16
Pick and place	place_bread_basket	57	**69**	83	**88**	19	**25**
	place_bread_skillet	61	**62**	82	**85**	**19**	14
	place_burger_fries	98	**100**	100	100	8	**41**
	place_can_basket	**58**	49	72	**74**	27	**36**
	place_cans_plasticbox	93	**95**	**100**	99	**27**	17
	place_dual_shoes	**49**	47	73	73	**10**	2
	place_object_basket	**73**	70	83	**84**	39	39
Average		50.83	**52.17**	63.72	**69.11**	23.11	**27.11**

open-finger frame, and a margin of 0 means that the prediction does not depend on the gripper. On nine tasks the margin lies between 0.36 and 0.83, while on the bag zipper it is 0.12, with contact in only 17% of the closed-finger frames. The bag zipper has the lowest margin of the ten tasks and also by far the smallest gain. This suggests that weak pseudo-labels limit the gain. The other tasks show no clear relation between the margin and the size of the gain. In every episode we inspected, the contact maps stay at the fingertips rather than at the contact between the held object and the scene. These checks therefore describe the pseudo-labels as a coarse signal that marks only the grasp and varies in quality across tasks. That Grounding Force improves all ten tasks with labels of this quality, obtained without any sensor on the robot, is the main result of the real-world experiments. The map target does not require accurate labels to help, and the only task with clearly weak labels is also the only one with a small gain, which points to better labels as a direct route to larger gains.

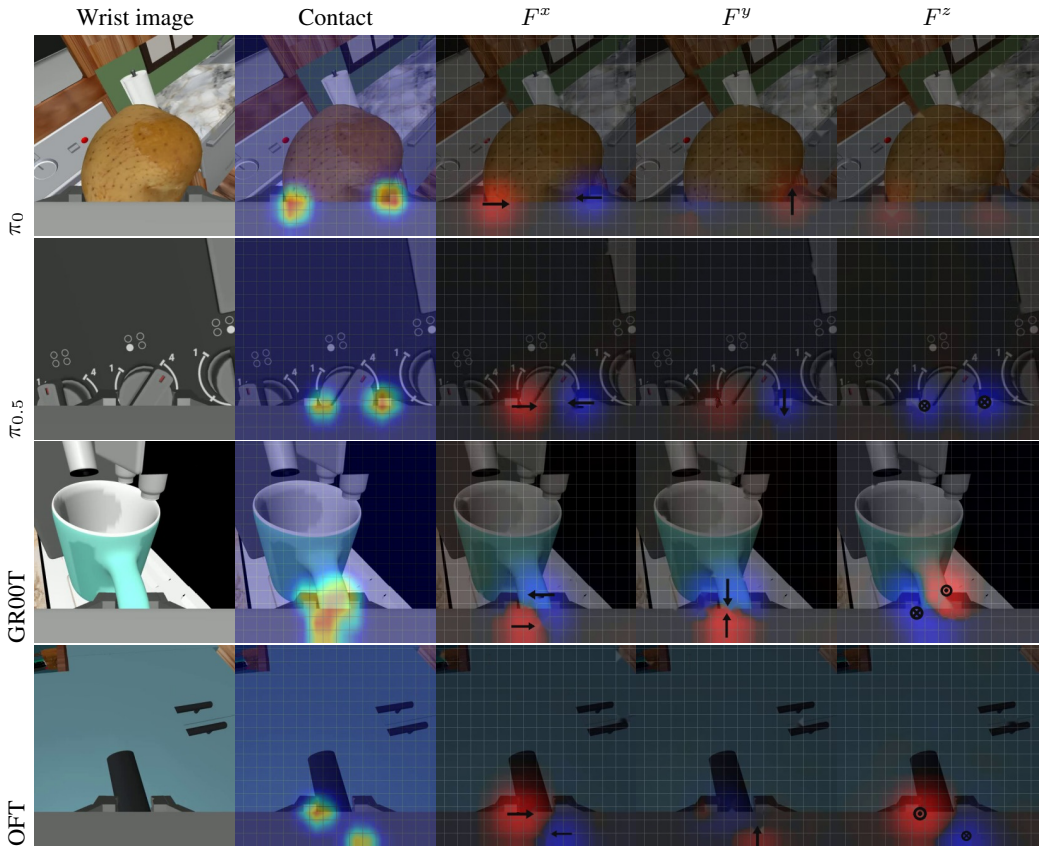


Figure 8: **Maps predicted during RoboCasa rollouts** of the four base VLAs with Grounding Force, each at a moment of contact: π_0 on PnPMicrowaveToCounter, $\pi_{0.5}$ on TurnOnStove, GR00T N1.6 on CoffeeSetupMug and OpenVLA-OFT on OpenSingleDoor. The maps are drawn as in Fig. 1b: the contact map in a jet color scale over the wrist image, and each force map in red for positive and blue for negative values, with arrows for the in-plane direction and a dot (toward the camera) or a cross (away from it) for F^z; x points to the right and y up.

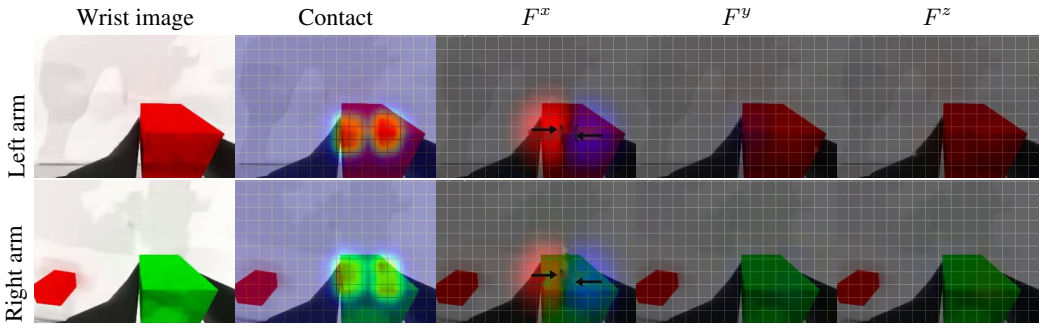


Figure 9: **Maps predicted for each arm during a RoboTwin 2.0 rollout** of $\pi_{0.5}$ with Grounding Force on stack_blocks_two, at the moment each arm grasps its block (left arm at 11.0 s, right arm at 24.5 s). Colors and symbols as in Fig. 8.

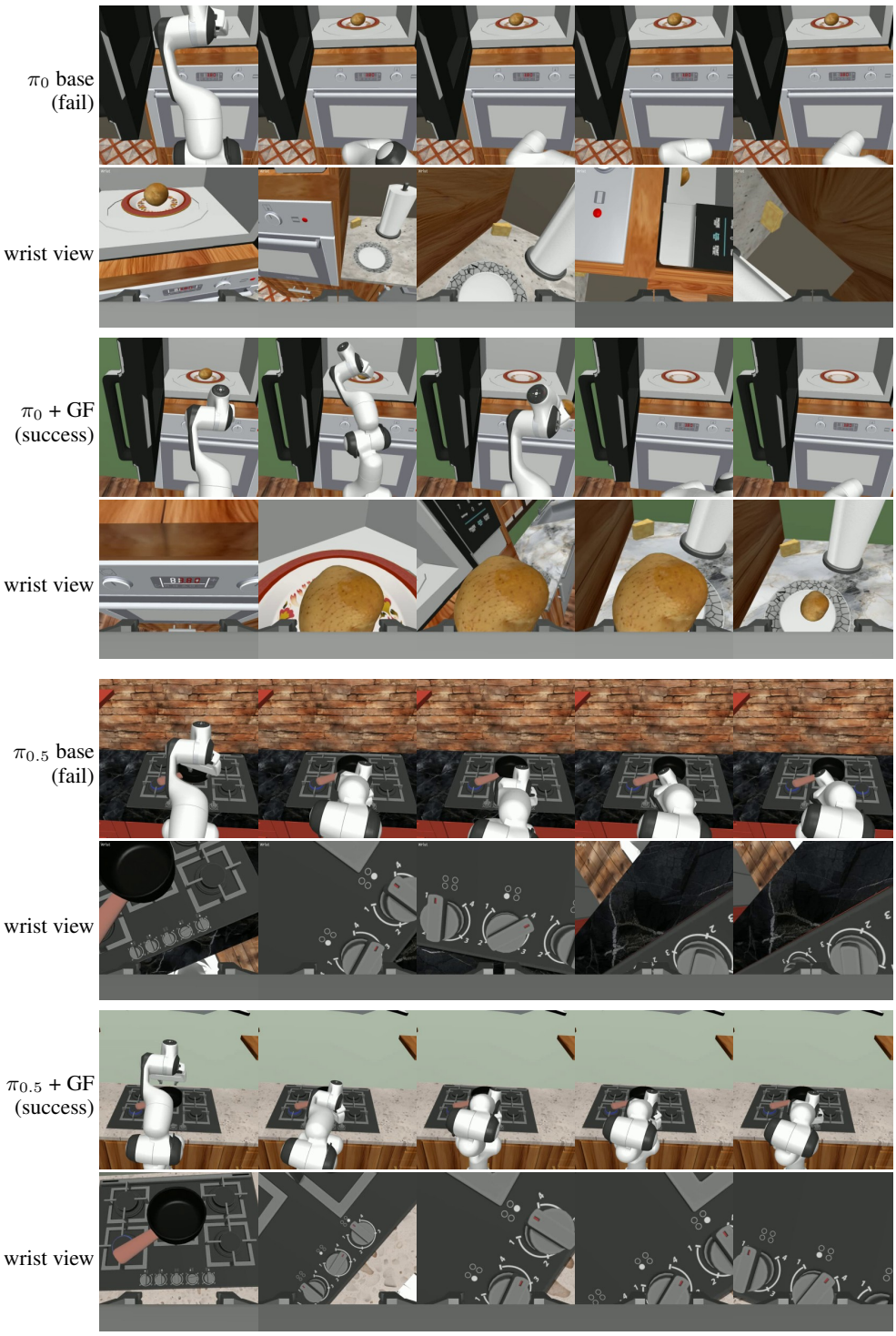


Figure 10: **Example rollouts on RoboCasa** of π_0 (PnPMicrowaveToCounter) and $\pi_{0.5}$ (TurnOnStove), with and without Grounding Force. For each rollout, the top row is the center camera and the bottom row the wrist camera, at five moments spaced evenly over the rollout, the last one at its end. The base rollouts reach the step limit without completing the task, and the rollouts with Grounding Force succeed. Each pair shows one episode of each model, not the same initial state.

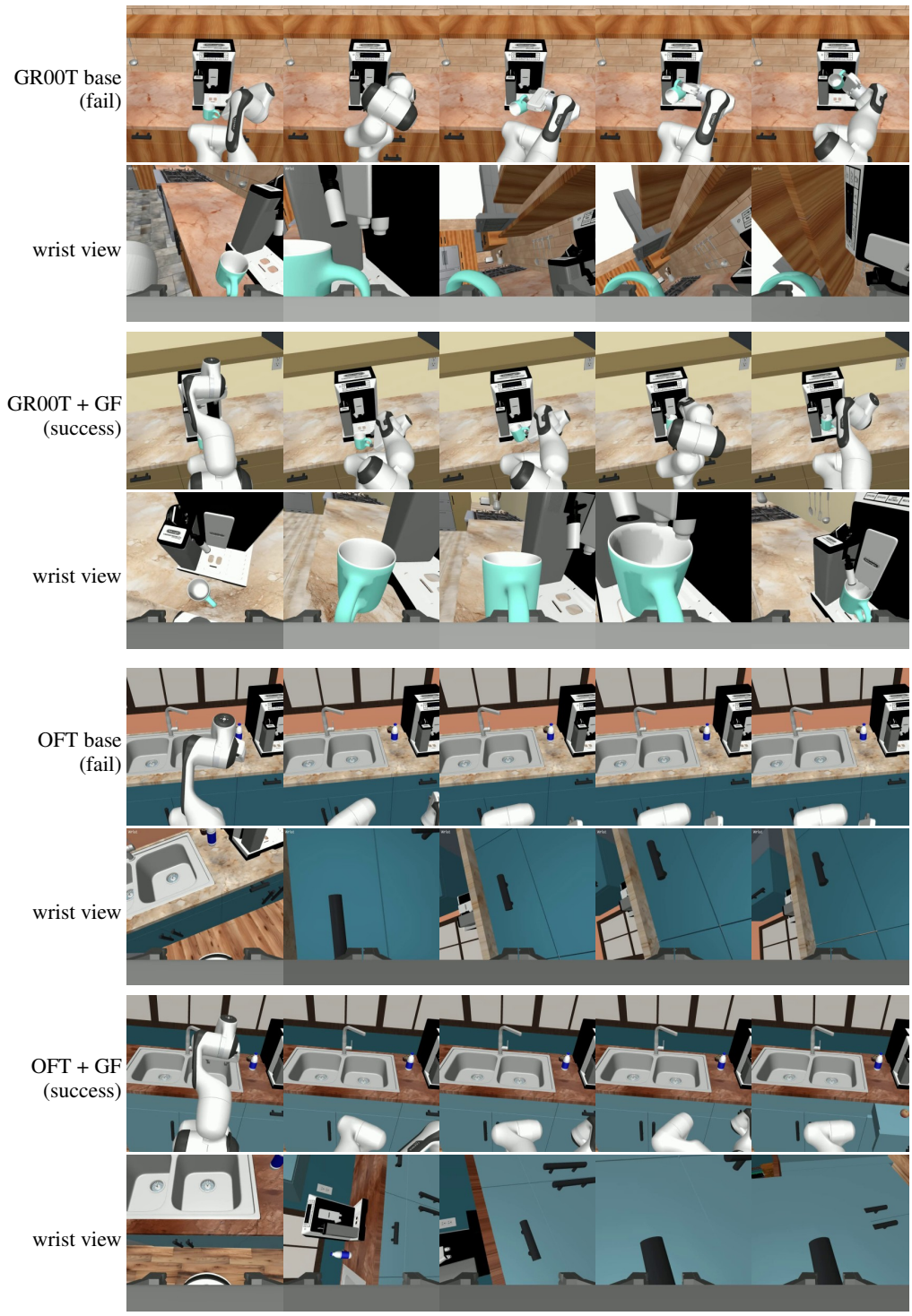


Figure 11: **Example rollouts on RoboCasa** of GR00T N1.6 (CoffeeSetupMug) and OpenVLA-OFT (OpenSingleDoor), with and without Grounding Force, laid out as in Fig. 10.

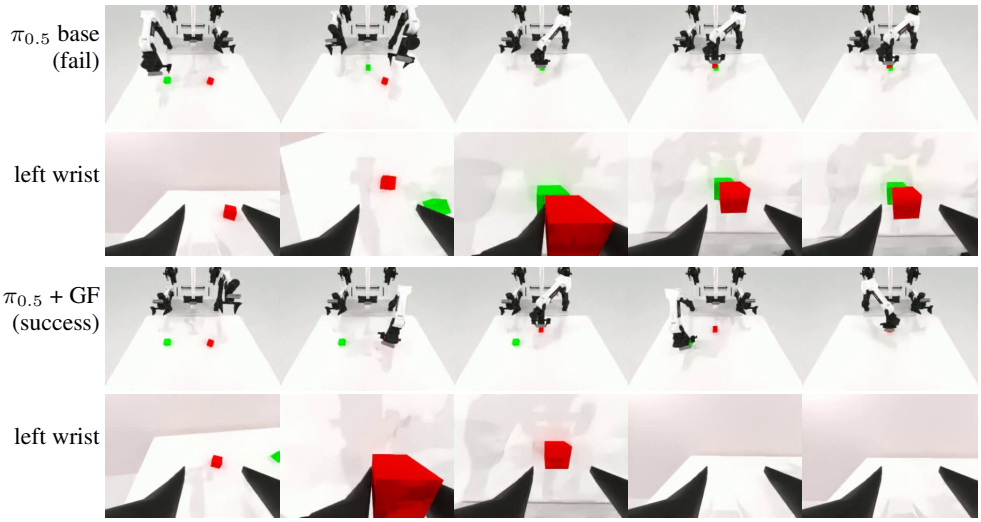
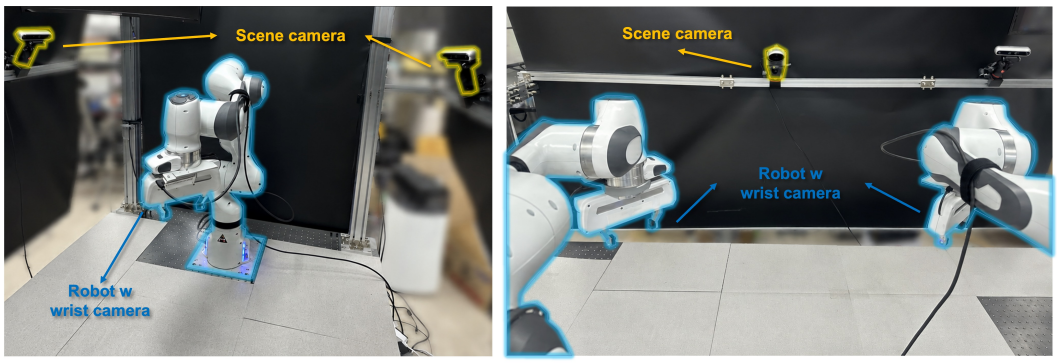


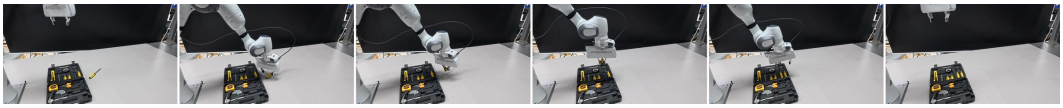
Figure 12: **Example rollouts on RoboTwin 2.0** of $\pi_{0.5}$ with and without Grounding Force on stack_blocks_two, in the third-person camera (top) and the left wrist camera (bottom), at five moments spaced evenly over the rollout.



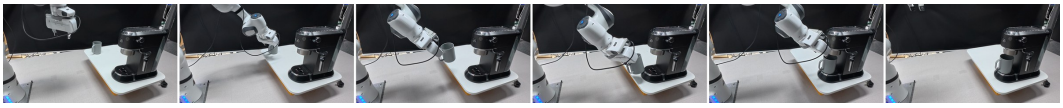
(a) Single arm (b) Bimanual

Figure 13: **Real-world setups.** (a) The single-arm setup, with the wrist camera on the gripper and two scene cameras on the rail. (b) The bimanual setup, with one wrist camera on each gripper and one scene camera at the center of the rail.

Screwdriver "Pick up the screwdriver and place it in the box."



Coffee setup "Set up the cup in the coffee machine."



Gear insertion "Pick up the red gear and place it on the other pin of the board."



Beer bottle "Pick up the opener and open the beer bottle."



Bag zipper "Close the zipper of the black bag."

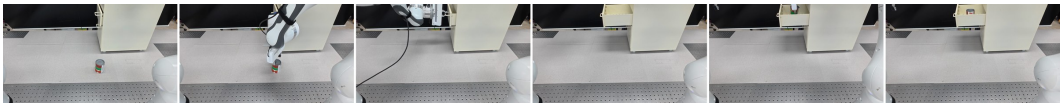


Water bottle "Twist open the lid of the water tumbler."

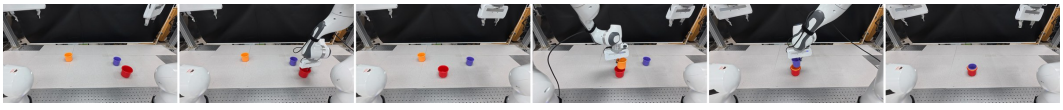


Figure 14: **Single-arm rollouts** of $\pi_{0.5}$ with Grounding Force, six frames per task, with the instruction of each task.

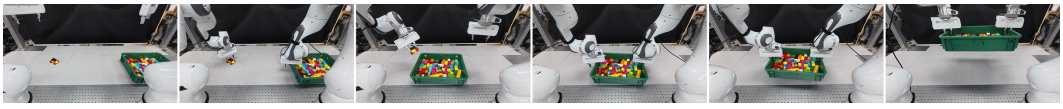
Drawer and can "Open the drawer and put the can into it."



Cup stacking "Stack the cups in the center, red first, then orange, then purple."



Lego box "Pull the Lego box, place the Lego piece in it with the other arm, and lift the box with both arms."



Cube stacking "Stack the cubes in the center, red at the bottom, then green, then white."

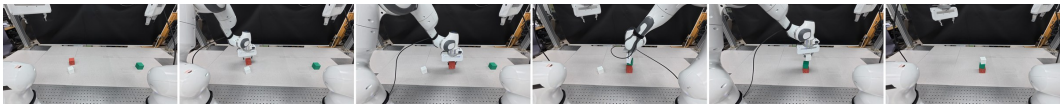


Figure 15: **Bimanual rollouts** of $\pi_{0.5}$ with Grounding Force, six frames per task, with the instruction of each task.

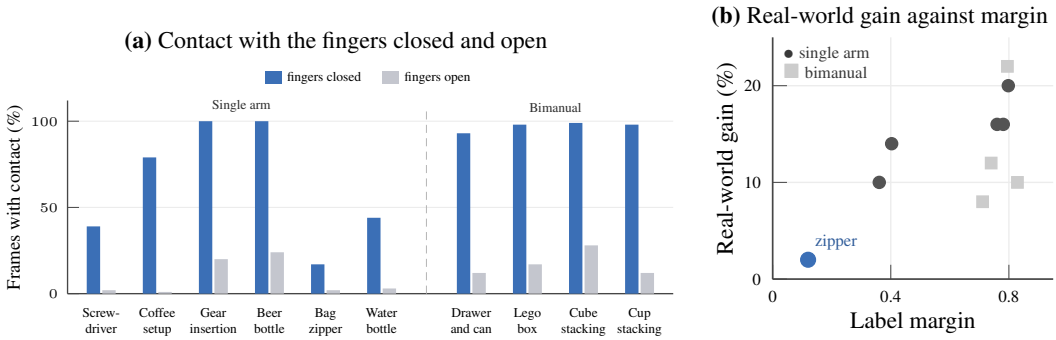


Figure 16: **Checks of the real-robot pseudo-labels against the gripper**, as medians over the 50 episodes of each task (and both arms on the bimanual tasks). (a) Fraction of frames whose contact map is active, with the fingers closed and open. (b) Real-world gain of Fig. 5b against the label margin, the difference between the two fractions.