

## A Additional Comparisons to RT-1-X

We further evaluate V-GPS on top of RT-1-X [6], a 35M parameter transformer policy pre-trained on the OXE dataset, in SIMPLER tasks. The complete results are presented in Table 3. V-GPS enhances RT-1-X’s performance by an average of **+21.6%**, improving all five policies across various embodiments. This is achieved using one single pre-trained value function for all policies and tasks.

	Task	Octo-s	Octo-s +Ours	Octo-b	Octo-b +Ours	Octo-s-1.5	Octo-s-1.5 +Ours	OpenVLA	OpenVLA +Ours	RT-1-X	RT-1-X +Ours
WidowX	Spoon on towel	0.52	0.50	0.25	0.16	0.01	0.07	0.00	0.02	0.01	0.03
	Carrot on plate	0.15	0.18	0.18	0.20	0.00	0.00	0.00	0.06	0.06	0.07
	Stack blocks	0.07	0.09	0.00	0.00	0.00	0.02	0.06	0.00	0.00	0.00
	Eggplant basket	0.49	0.59	0.28	0.37	0.01	0.07	0.16	0.54	0.01	0.01
	Average	0.30	0.34 <b>+10.6%</b>	0.17	0.18 <b>+2.82%</b>	0.01	0.04 <b>+700%</b>	0.05	0.15 <b>+182%</b>	0.02	0.03 <b>+35.7%</b>
Google Robot	Pick Can	0.31	0.30	0.00	0.30	0.05	0.47	0.72	0.78	0.19	0.32
	Put Near	0.12	0.17	0.25	0.06	0.10	0.21	0.68	0.44	0.44	0.43
	Average	0.21	0.23 <b>+9.30%</b>	0.12	0.18 <b>+44.0%</b>	0.08	0.34 <b>+353%</b>	0.70	0.61 <b>-12.9%</b>	0.31	0.37 <b>+19.5%</b>
Total	Average	0.28	<b>0.31</b> <b>+10.2%</b>	0.16	<b>0.18</b> <b>+13.5%</b>	0.03	<b>0.14</b> <b>+394%</b>	0.27	<b>0.31</b> <b>+13.6%</b>	0.12	<b>0.15</b> <b>+21.6%</b>

**Table 3: (SIMPLER [11] performance with RT-1-X)** We further evaluate V-GPS on top of RT-1-X. V-GPS enhances the success rates of RT-1-X by an average of **+21.6%**, improving all five generalist policies across multiple embodiments.

## B V-GPS Implementation Details

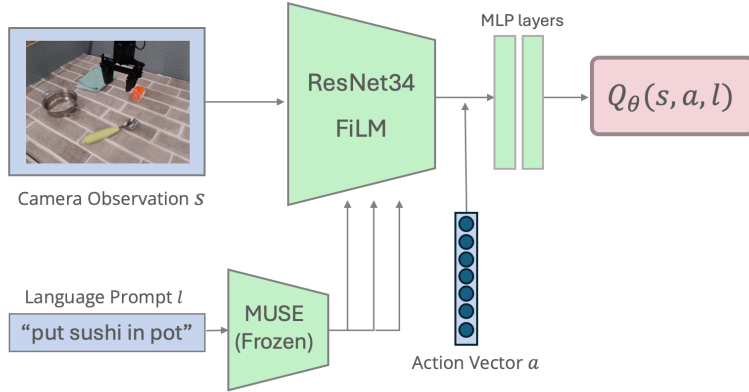
In this section, we provide the implementation details of V-GPS for value function pre-training, and test-time action re-ranking. The hyperparameters are listed in Table 4.

### B.1 Value Function Pre-Training

Our language-conditioned Q function  $Q_\theta(s, a, l)$  uses a ResNet-34 image encoder with FiLM language conditioning as shown in Figure 5. The image observation is first passed through the ResNet-34 encoder, while the language instruction, processed by a frozen MUSE encoder, is applied to every block in ResNet using FiLM conditioning. The 7-dimensional actions are concatenated with the final output from the ResNet, then passed through two 256-unit hidden layers, and finally, a scalar Q value is predicted. We followed the official IQL implementations and hyperparameters, using an expectile  $\tau = 0.7$ , discount factor  $\gamma = 0.98$ , clipped double Q-learning [59], and shifted reward values of 0 and  $-1$ . We assigned the final 3 steps of each trajectory as positive rewards 0, and the rest as negative rewards  $-1$ . We use the Adam optimizer with a learning rate of  $3e-4$ . During training, we augment the image observations with random cropping and color jitter. The value function used in our real-world evaluation is trained on the Bridge V2 dataset for 400,000 steps with a batch size of 256. The cross-embodied value function used in our simulated evaluation is trained on a mix of the Bridge V2 and Fractal datasets for 200,000 steps with a batch size of 512.

### B.2 Test-Time Action Re-Ranking

During test-time, we sample  $K = 10$  action proposals from the base policy  $\pi$  at each time step, and then re-rank the proposed actions using the Q function with Equation 3. In the real-world evaluations, we found selecting the action greedily by setting  $\beta \rightarrow 0$  leads to satisfactory results. In simulation, we sweep over  $\beta = \{0, 0.1, 1.0\}$  and report the best result.



**Figure 5: (Model Architecture.)** Our value function uses a ResNet-34 image encoder with FiLM language conditioning.

IQL expectile $\tau$	0.7
discount factor	0.98
learning rate	3e-4
positive reward steps	3
number of actions to sample $K$	10
softmax temperature $\beta$	0, 0.1, 1.0

**Table 4: (V-GPS hyperparameters)**

## C Baseline Implementation Details

For Octo-`{small, base, small-1.5}`, we used the publicly released checkpoints from <https://huggingface.co/rail-berkeley>. For RT-1-X, we used the publicly released JAX checkpoint from [https://github.com/google-deepmind/open\\_x\\_embodiment](https://github.com/google-deepmind/open_x_embodiment). For OpenVLA, we used their public checkpoint from <https://huggingface.co/openvla/openvla-v01-7b>.

Our real-world evaluation is implemented on top of the evaluation codes provided from <https://github.com/octo-models/octo>, and the simulated evaluation is based on <https://github.com/simpler-env/SimplerEnv>.

## D Experimental Setup

**(Real world)** We conducted our real-world evaluations on 8 tasks across 4 different scenes as shown in Figure 3. We provide the language instructions we used for each task in Table 5. We conduct 30 trials per task and report the average success rates in Table 1. We randomize the configurations and orientations of each object for each trial.

**(SIMPLER)** We conducted the simulated evaluations on 6 tasks in the SIMPLER environment, including 4 tasks on the WidowX robot platform and 2 on the Google Robot platform as shown in Figure 3. We used the default language instructions for each task as shown in Table 6. For RT-1-X and Octo-`{small, base, small-1.5}`, we conducted 100 trials for each of three different random seeds. For OpenVLA, we performed 50 trials per task due to its slower inference speed. The average success rates are reported in Table 2.

	Language Instructions
Scene A	put the sushi in the pot put the sweet potato on the cloth
Scene B	put the red object in the pot put the red object on the cloth
Scene C	put the mushroom in the pot put the mushroom on the cloth
Scene D	put the sushi in the pot put the spoon on the cloth

**Table 5: (Real-world scenes and tasks)** We evaluate V-GPS in 8 tasks across 4 different real-world scenes.

	Language Instructions
WidowX	put the spoon on the towel put carrot on plate stack the green block on the yellow block put eggplant into yellow basket
Google Robot	pick coke can move {object1} near {object2}

**Table 6: (SIMPLER scenes and tasks)** We evaluate V-GPS in 6 tasks across 2 different embodiments in SIMPLER environment.

## 547 E Reproducibility Statement

548 We will release the training codes, pre-trained checkpoints, and evaluation codes in the final version.