
When Would Vision-Proprioception Policy Fail in robot Manipulation? *Supplementary Material*

Anonymous Author(s)

Affiliation

Address

email

1 A Implementation Details

2 In this section, we provide the implementation details of our work, including the experimental settings
3 and algorithm specifics.

4 A.1 Experiment Settings

5 Simulation experiments are conducted on two representative environments: MetaWorld Yu et al.
6 (2020) and RoboSuite Zhu et al. (2020). Tasks in MetaWorld are relatively simple, featuring a 4-
7 dimensional action space that includes the gripper’s position and its opening degree, while RoboSuite
8 tasks involve complex scenarios, longer task sequence horizons and richer physical interactions,
9 with the action space further including the orientation of the gripper. During policy learning, task in
10 MetaWorld uses 100 hard-coded expert demonstrations from its official open-source implementation,
11 while task in RoboSuite Zhu et al. (2020) leverages 500 synthesized trajectories that generated from
12 human demonstrations Mandlekar et al. (2023).

13 We use an observation history window H of 5 with the learning rate $3e^{-4}$ to train the vision-
14 proprioception policy network. The parameters of the policy are optimized by Adam. The vision-
15 proprioception policy consists of two feature extraction chunks and a policy head. For the visual input,
16 spatial features are first extracted using a ResNet-18 He et al. (2016) backbone, then projected into a
17 512-dimensional hidden representation. A 4-layer temporal transformer with a hidden dimension of
18 256 is applied to capture temporal features. For proprioceptive input, spatial features are extracted
19 using a 3-layer MLP. These features are fused via concatenation and then passed through a 3-layer
20 MLP to output a sequence of actions of length $L = 9$. We train with batch size 128 on a single
21 NVIDIA RTX 3090 GPU for 100 epochs (around 2 hours for 100 MetaWorld demonstrations and 8
22 hours for 500 RoboSuite demonstrations).

23 For real-world experiments, we use a 6-DoF xArm 6 robotic arm equipped with a Robotiq gripper
24 for all one-arm tasks. The visual information is provided by RGB images captured by an Intel
25 RealSense D435i camera mounted on the wrist. Moreover, we utilize the open-source Cobot Magic
26 platform to support tasks requiring dual-arm collaboration. It features four robot arms and three
27 Intel RealSense D435 RGB-D cameras. Due to extended reach of the dual-arm setup, the two
28 wrist-mounted cameras can’t fully capture the task scene. Therefore, we also incorporate the front
29 camera on the platform to obtain a more comprehensive RGB observation of the task. Policies of
30 real-world experiments are trained with 50 demonstrations collected by teleoperation. For one-arm
31 tasks, the vision-proprioception policy is the same as in simulation. For dual-arm tasks, we modified
32 the ACT Zhao et al. (2023) architecture. we train the left-arm policy using observations from the
33 left-wrist camera, the front camera, and the left-arm proprioceptive information, and similar settings
34 are applied to the right-arm policy.

In all tasks, the initial position of target object varies randomly in each validation, while the initial position of gripper remains fixed. In out-of-distribution (OOD) scenarios, the initial distribution of object positions differs from that in the training dataset of expert demonstrations.

A.2 Details of GAP

In this study, we illustrate that vision-proprioception policy would fail during motion-transition phases due to its suppressed vision modality. To alleviate this, we propose the Gradient Adjustment with Phase-guidance (GAP) algorithm, enabling dynamic collaboration between vision and proprioception within vision-proprioception policy.

The complete list of hyperparameters used in Equations 4 and 5 is provided in Table 1. Since the motion inconsistency of the gripper’s opening state is measured using binary (0/1) values, we set β to a lower value to balance the numerical scale, while keeping $\alpha = 1$ to account for the overall gripper motion. To enable fine-grained gradient adjustment, we set the parameter λ controlling the degree of adjustment to a moderate level, avoiding excessive modulation that could lead to unstable gradient updates and potential policy collapse. To further ensure training stability, we apply gradient adjustment only during the early stage of policy learning, such as the first 50 epochs. For bimanual tasks, we predict the motion transition indicators based solely on the proprioceptive signals of the left arm and perform gradient adjustment accordingly. The similar strategy is applied to the right arm.

Table 1: List of hyperparameters used in GAP.

| Hyperparameters | Value |
|-----------------|-----------|
| α | 1 |
| β | $2e^{-3}$ |
| λ | 0.3 |

B Additional Experiments

B.1 Experiments across Multiple Seeds

In addition to the results provided in Section Experiments, we report the average success rate and standard deviation of our GAP algorithm across multiple seeds. Due to computational resource limitations, we calculate them only in part of experiments. As reported in Table 2, GAP achieves higher average success rates and consistently outperforms other methods, with relatively low standard deviations. It indicates that GAP can enhance vision-proprioception policies across a diverse range of environments, thereby demonstrating its robustness and stability.

Table 2: Comparisons with other methods in both simulated and real-world environments. Average success rate and standard deviation are calculated on 5 seeds.

| Setup | Meta-World | | RoboSuite | | Real One-Arm | Real Dual-Arm |
|---------------------------|------------|-----------|-----------|-----------|------------------------|-------------------|
| Task Method | assembly | push-wall | stack | threading | use rag to sweep table | lift lid and pour |
| Vision-only | 82.6±3.0 | 63.2±1.9 | 65.8±2.4 | 43.6±1.7 | 45±3.5 | 46±4.2 |
| Concatenation | 74.6±2.1 | 54.4±2.9 | 54.6±2.3 | 33.2±1.9 | 24±4.1 | 24±4.2 |
| MS-Bot Feng et al. (2024) | 91.0±2.1 | 66.2±1.6 | 69.4±1.9 | 51.2±1.3 | 53±2.7 | 49±4.2 |
| Aux Fu et al. (2024) | 90.4±1.8 | 51.0±3.2 | 54.0±2.2 | 45.4±2.1 | 51±4.2 | 40±3.5 |
| Mask Liu et al. (2024) | 89.2±1.6 | 79.2±3.7 | 61.8±1.9 | 47.2±1.6 | 34±4.2 | 36±4.2 |
| GAP (Ours) | 94.2±0.8 | 73.6±1.3 | 77.6±0.9 | 53.0±1.2 | 65±3.5 | 76±2.2 |

60 B.2 Experiments with Various Policy Heads

61 The previous experiments have demonstrated the effectiveness of our GAP on policies with MLP-
62 based policy heads. Additionally, we conducted experiments using the popular UNet-based diffusion
63 policy head Chi et al. (2023) and reported the results in Table 3. The results show that policies using
64 a diffusion-based policy head are generally better than those with MLP-based heads, while applying the
65 GAP method still outperforms vision-only policies and other methods.

Table 3: Comparisons with other methods in simulated environments. Vision-Proprioception policies are equipped with UNet-based diffusion policy head.

| Suite | Meta-World | | | RoboSuite | |
|---------------------------|-------------|-----------|-------------|-----------|-----------|
| Task | disassemble | push-wall | bin-picking | stack | threading |
| Method | | | | | |
| Vision-only | 92% | 70% | 66% | 73% | 51% |
| Concatenation | 82% | 62% | 51% | 62% | 40% |
| MS-Bot Feng et al. (2024) | 93% | 75% | 72% | 77% | 58% |
| Aux Fu et al. (2024) | 93% | 71% | 75% | 72% | 55% |
| Mask Liu et al. (2024) | 87% | 83% | 67% | 66% | 53% |
| GAP (Ours) | 97% | 88% | 81% | 87% | 63% |

66 C Task Description

67 In this subsection, we provide detailed descriptions of tasks we evaluated and visualize additionally
68 simulation tasks in Figure 1. The evaluations comprehensively cover a wide range of manipulation
69 tasks, including simple pick-and-place tasks, rotation-sensitive tasks, as well as long-horizon and
70 contact-rich tasks and includes one-arm and dual-arm robot setups.

- 71 • pick-place: The robot arm first needs to locate the position of the cylinder, then move toward
72 it. After grasping the object, it locates the target circle and moves the object to the target
73 position.
- 74 • assembly: The robot arm first locates the position of the part, then moves toward it. After
75 grasping the part, it locates the position of the base, moves the part above it, and finally
76 assembles them.
- 77 • disassemble: The robot arm first locates the position of the part, then moves toward it. After
78 grasping the part, it disassembles the part from the base.
- 79 • push-wall: The robot arm first locates the position of the cylinder, then moves toward it.
80 After grasping it, the arm pushes it to a target point behind the wall. The target point is not
81 visible through visual observation, but its location remains fixed across all evaluations.
- 82 • bin-picking: The robot arm first locates the position of the green cube, then moves toward
83 it. After grasping it, the arm removes it from the red container and finally places it into the
84 blue container.
- 85 • hammer: The robot arm first locates the position of the drawer and then moves toward it.
86 After opening the drawer, it locates the position of the hammer, grasps it, places it inside the
87 drawer, and finally closes the drawer.
- 88 • stack: The robot arm first locates the position of the red cube and then moves toward it.
89 After grasping it, it locates the position of the green cube and finally stacks the red cube on
90 top of the green cube.
- 91 • threading: The robot arm first locates the position of the pin and then moves toward it. After
92 grasping it, it locates the position of the pinhole, then moves and rotates to align the pin with
93 the hole, and finally inserts the pin into the hole.

- 94 • press button: The robot arm first locates the position of the button, then moves toward it,
95 closes the gripper, and presses the button.
- 96 • put cube in drawer: The robot arm first locates the position of the cube, then moves toward
97 it. After grasping the cube, it locates the drawer and finally places the cube in the drawer.
- 98 • use rag to sweep table: The robot arm first locates the position of the rag, then moves toward
99 the rag. It rotates the arm to grasp the rag, takes it down from the shelf, and finally rotates
100 the rag to sweep the table.
- 101 • handover: The robot first locates the position of the rectangular block, then its right arm
102 moves toward it. After grasping the block, the right arm moves it to the center, where the
103 left arm takes over the block in a handover. Finally, the left arm places it on the left side of
104 the table.
- 105 • put thermos into bag: The robot first locates the position of the bag, then its right arm moves
106 toward it, lifts it, and rotates the wrist to open the bag. Next, robot locates the thermos, uses
107 the left arm to grasp it, and puts it into the bag.
- 108 • lift lid and pour: The robot first locates the position of the cup, then its left arm moves
109 toward it to grasp and lift the lid. Next, robot locates the graduated cylinder, uses the right
110 arm to grasp it, moves it above the cup, and finally rotates the cylinder to pour.

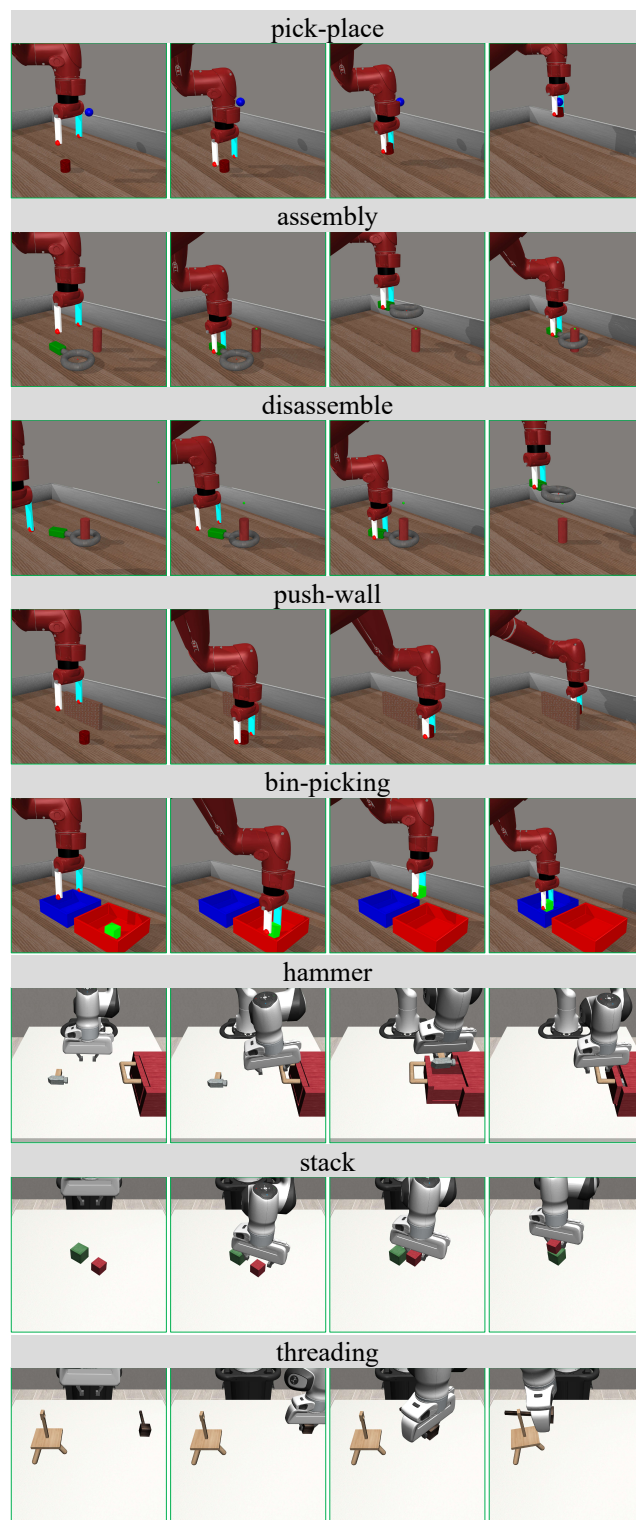


Figure 1: Visualization of tasks we evaluated in simulation environments.

References

- T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of Conference on Robot Learning (CoRL)*, pages 1094–1100, 2020.
- Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, Y. Zhu, and K. Lin. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.
- A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. In *Conference on Robot Learning (CoRL)*, pages 1820–1864. PMLR, 2023.
- K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- T. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.
- R. Feng, D. Hu, W. Ma, and X. Li. Play to the score: Stage-guided dynamic multi-sensory fusion for robotic manipulation. In *Proceedings of Conference on Robot Learning (CoRL)*, 2024.
- Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn. Humanplus: Humanoid shadowing and imitation from humans. In *Proceedings of Conference on Robot Learning (CoRL)*, 2024.
- S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.
- C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.