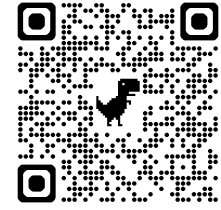




ICML

International Conference
On Machine Learning



Virtual Poster



NeurVLA: Unleashing Failure-Handling Capability of Vision-Language-Action Models via Neural-Symbolic Reasoning

Xuqi Liu^{1*} Minghe Gao^{1*} Juncheng Li¹ ✉ Siliang Tang¹
¹ Zhejiang University

* Equal contribution ✉ junchengli@zju.edu.cn

Motivation and Introduction

Two closely interconnected issues in existing VLA models:

- Coarse-grained failure correction
- Unreliable failure prevention

Key observation underlying these limitations:

- Fine grained failure correction requires *precise analysis* of object states from observations
- Reliable failure prevention demands *explicit modeling and verification* of action induced state transitions

Contribution:

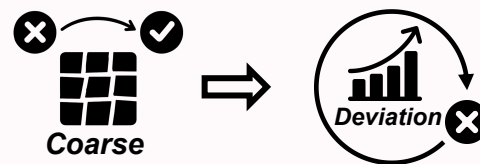
- We propose **NeurVLA**, a reasoning framework that jointly addresses failure correction and prevention for VLA models by integrating *Observation-Grounded Failure Correction* with *Trajectory-Modeling Failure Prevention*.
- **NeurVLA** further proposes *Reasoning-Guided Action Learning* to internalize the failure-handling capability derived from the above reasoning processes in VLA models.
- Extensive experiments demonstrate that **NeurVLA** consistently achieves strong performance and robust generalization across diverse embodied tasks.

Why do existing methods underperform?

❌ *Discretized Adjustment*

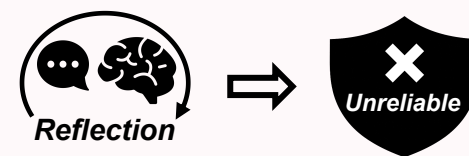
❌ *Approximate Perception*

(1) *Coarse-grained failure correction*



Coarse-grained failure correction leads to accumulated deviations and recurring failures.

(2) *Unreliable failure prevention*



Semantic self-reflection yields unreliable failure prevention.

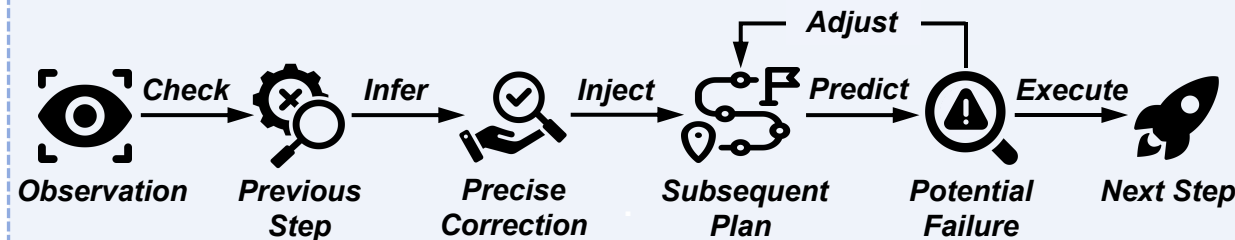
Why does NeurVLA perform better?

✅ *Precise Correction*

✅ *Explicit Modeling*

(1) *Observation-Grounded Failure Correction*

(2) *Trajectory-Modeling Failure Prevention*



(3) *Reasoning-Guided Action Learning*

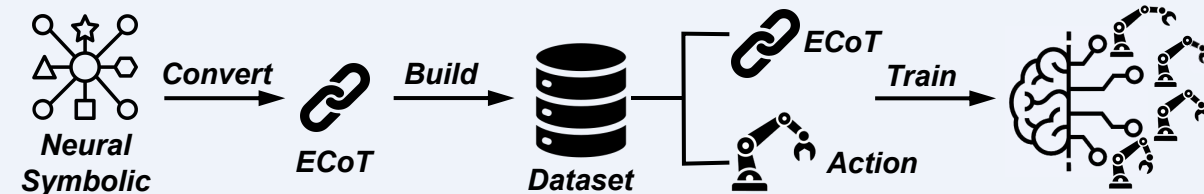


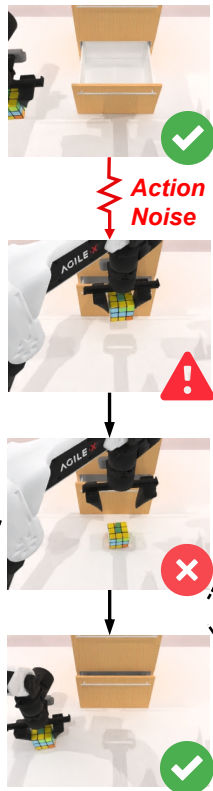
Figure 1. **NeurVLA** is a framework that overcomes core limitations of existing VLA methods, namely coarse-grained failure correction and unreliable failure prevention, and further internalizes failure-handling capability into VLA model.

Method

NeurVLA divides VLA task completion into several *iterations*, where each *iteration* consists of *generating a reasoning process* and *executing a semantically meaningful action*. In each *iteration*, given the current observation o and the task instruction $inst$, **NeurVLA**:

- (i) assesses whether the previous action has been successfully completed based on the observation, and injects fine-grained corrective steps when necessary;
- (ii) analyzes the updated plan by explicitly modeling motion trajectories and potential collisions, iteratively refining the plan until safety is verified;

NeurVLA Capability Construction Pipeline



Instruction: Place the cube into the cabinet's lower drawer.



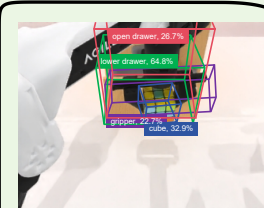
Observation:



(1) Observation-Grounded Failure Correction

`</> Check_Previous_Step.py`

```
cube_box, drawer_box = get_box(["cube", "lower drawer"])
is_inside = contain(drawer_box, cube_box)
is_open = verify_property("lower drawer", "open")
cabinet_box, gripper_box = get_box(["cabinet", "gripper"])
drawer_open_width = spatial_relation(
    drawer_box, cabinet_box)["dz"]
min_open_width = cube_width + ε
if is_holding(gripper_box, cube_box):
    .....
```



cube_inside_drawer: False

```
is_drawer_open: True
drawer_open_width: 0.04
min_open_width: 0.16
gripper_hold_cube: False
```

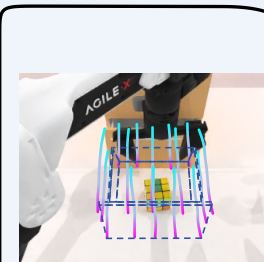
Subsequent Plan:

- Correction:** Pull out lower drawer at least 16 cm
- Correction:** Grasp cube and lift it by at least 10 cm
- Move cube above the lower drawer opening
- Put down cube

(2) Trajectory-Modeling Failure Prevention

`</> Predict_Plan_Failure.py`

```
drawer_patch, cube_patch = get_patch(
    ["lower drawer", "cube"])
drawer_points, cube_points = sample(
    [drawer_patch, cube_patch], 8)
for step in subsequent_plan:
    drawer_trace = predict_trace(step, drawer_points)
    cube_trace = predict_trace(step, cube_points)
    collision_info = check_collision(
        [drawer_trace, cube_trace])
    .....
```



collision_info: "Collision predicted. The cube's box lies within the lower drawer's opening range, causing a collision."

Adjusted Plan:

- Grasp cube and move it left by at least 20 cm
- Correction:** Pull out lower drawer at least 16 cm
- Correction:** Grasp cube and lift it by at least 10 cm
- Move cube above the lower drawer opening
- Put down cube

Repeat until no collision is detected

Figure 2. Overview of **NeurVLA**. **NeurVLA** jointly constructs and internalizes robust failure-handling capabilities for VLA models. It (1) programmatically checks completion of the previously executed actions from observations to enable fine-grained failure correction, (2) predicts potential failures in the subsequent plan via executable trajectory simulation for reliable failure prevention, and, after using policy code to obtain the corresponding actions, it (3) internalizes failure-handling capability by first converting the above reasoning processes into introspective ECoTs, and then performing SFT and contrastive learning on ECoT-actions.

Method

NeurVLA divides VLA task completion into several *iterations*, where each *iteration* consists of *generating a reasoning process* and *executing a semantically meaningful action*. In each *iteration*, given the current observation o and the task instruction $inst$, **NeurVLA**: (iii) After translating the plan into policy code blocks to interact with the environment, it converts the above reasoning processes into ECoTs, and performs *Reasoning-Guided Action Learning*.

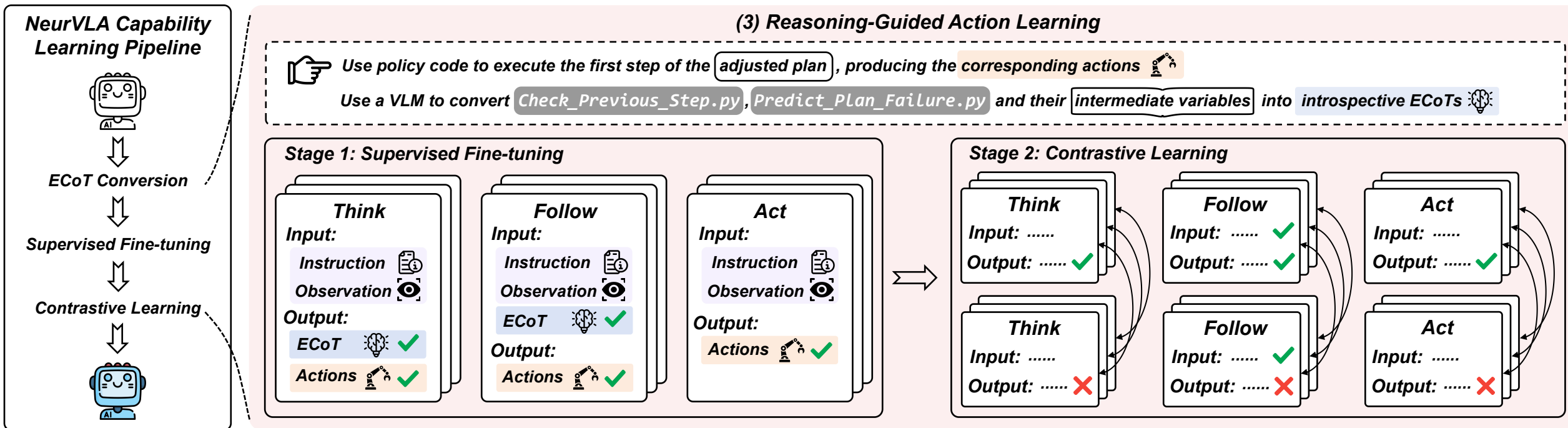


Figure 2. Overview of **NeurVLA**. **NeurVLA** jointly constructs and internalizes robust failure-handling capabilities for VLA models. It (1) programmatically checks completion of the previously executed actions from observations to enable fine-grained failure correction, (2) predicts potential failures in the subsequent plan via executable trajectory simulation for reliable failure prevention, and, after using policy code to obtain the corresponding actions, it (3) internalizes failure-handling capability by first converting the above reasoning processes into introspective ECoTs, and then performing SFT and contrastive learning on ECoT-actions.

Simulation Experiments

Table 1. Comparison of VLA models on LIBERO.

Method	Spatial	Object	Goal	Long	Avg
<i>Regression or classification-based VLA</i>					
OpenVLA	84.7	88.4	79.2	53.7	76.5
SpatialVLA	88.2	89.9	78.6	55.5	78.1
UniVLA	96.5	96.8	95.6	92.0	95.2
NORA	92.2	95.4	89.4	74.6	87.9
- <i>CollabVLA</i>	93.4	96.2	92.2	81.3	90.8
- <i>FailSafe</i>	94.2	96.5	93.5	85.8	92.5
- NeurVLA	97.1	98.3	96.7	92.2	96.1
<i>Flow-matching or diffusion-based VLA</i>					
π_0	96.8	98.8	95.8	85.2	94.2
ThinkAct	88.3	91.4	87.1	70.9	84.4
FPC-VLA	87.0	92.0	86.2	82.2	86.9
InstructVLA	92.4	95.6	92.0	76.6	89.2
- <i>CollabVLA</i>	93.6	96.4	93.2	80.7	91.0
- <i>FailSafe</i>	93.9	96.5	93.6	81.1	91.3
- NeurVLA	97.5	99.2	97.3	92.4	96.6

Table 2. Performance comparison on SimplerEnv.

Method	Google Robot								WidowX Robot					Avg
	Pick		Move		Open/Close		Avg	Put	Put	Stack	Put	Avg		
	Coke	Can	Near		Drawer			Spoon	Carrot	Blocks	Eggplant		Success	
VM	VA	VM	VA	VM	VA	VM	VA							
<i>Regression or classification-based VLA</i>														
OpenVLA	16.3	54.5	46.2	47.7	35.6	17.7	32.7	40.0	0.0	0.0	0.0	4.1	1.0	24.6
CoT-VLA	60.3	71.8	61.5	57.2	48.0	35.2	56.6	54.7	17.9	18.2	15.1	42.6	23.5	44.9
SpatialVLA	81.0	89.5	69.6	71.7	59.3	36.2	70.0	65.8	20.8	20.8	25.0	70.8	34.4	56.7
NORA	81.6	56.2	69.0	77.9	30.1	24.5	60.2	52.9	71.3	40.9	35.7	83.8	57.9	57.0
- <i>CollabVLA</i>	85.8	69.1	77.8	86.1	42.8	32.4	68.8	62.5	76.5	49.2	46.1	85.9	64.4	65.2
- NeurVLA	90.6	89.8	82.1	84.5	63.4	41.7	78.7	72.0	82.7	56.3	52.8	89.9	70.4	73.7
<i>Flow-matching or diffusion-based VLA</i>														
π_0	72.7	75.2	65.3	63.7	38.3	25.6	58.8	54.8	29.1	0.0	16.7	62.5	27.1	46.9
ThinkAct	92.0	84.0	72.4	63.8	50.0	47.6	71.5	65.1	58.3	37.5	8.7	70.8	43.8	60.1
FPC-VLA	95.3	91.3	93.8	86.7	76.4	30.7	88.5	69.6	79.2	58.3	45.8	75.0	64.6	74.2
InstructVLA	79.6	92.3	68.3	71.9	52.3	61.7	66.7	75.3	47.1	40.4	20.6	71.5	44.9	62.3
- <i>CollabVLA</i>	86.1	93.6	80.4	82.7	63.2	65.3	76.6	80.5	62.2	49.7	31.9	76.3	55.0	70.7
- NeurVLA	96.5	98.2	94.8	95.1	74.9	76.6	88.7	90.0	83.8	63.5	52.8	91.2	72.8	83.8

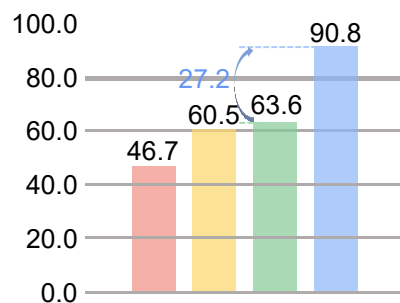
Table 3. Evaluation on the CALVIN ABC→D benchmark.

Method	Tasks completed in a row (%) \uparrow					Avg Len.
	1	2	3	4	5	
<i>Regression or classification-based VLA</i>						
OpenVLA	91.3	77.8	62.0	52.1	43.5	3.27
UniVLA	95.5	85.8	75.4	66.9	56.5	3.80
NORA	93.9	82.7	70.2	61.4	51.4	3.60
- <i>FailSafe</i>	95.6	87.1	78.3	72.5	64.4	3.98
- NeurVLA	96.4	89.0	82.1	74.8	66.3	4.09
<i>Flow-matching or diffusion-based VLA</i>						
π_0	93.8	85.0	76.7	68.1	59.9	3.84
UniCoD	97.3	89.5	82.3	75.2	67.0	4.11
InstructVLA	93.1	83.8	72.6	63.6	55.4	3.69
- <i>FailSafe</i>	95.4	87.9	77.8	70.9	63.8	3.96
- NeurVLA	97.9	90.3	83.4	75.8	69.6	4.17

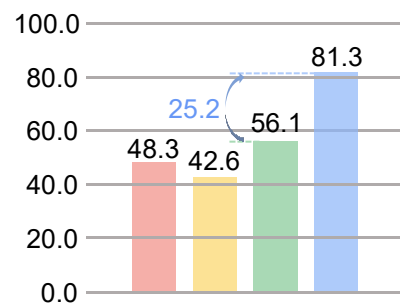


NeurVLA consistently improves task success rates across multiple robotic manipulation benchmarks, demonstrating strong generalization across diverse tasks, environments, and VLA architectures.

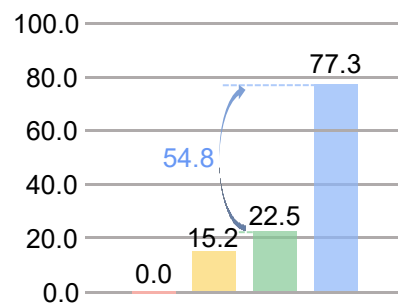
Real-world Experiments



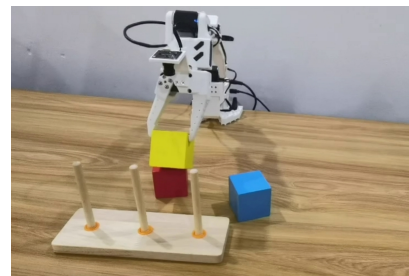
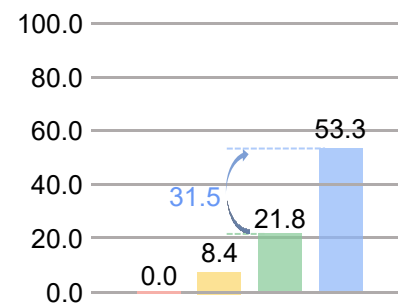
Put All Blocks into Lower Drawer



Fold Towel Twice



Pour Ball into Brown Cup



3-Disc Colored Tower of Hanoi

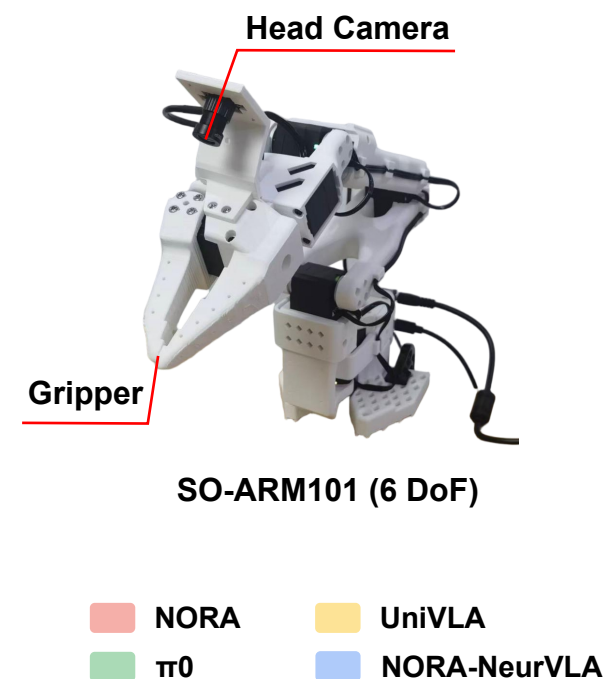
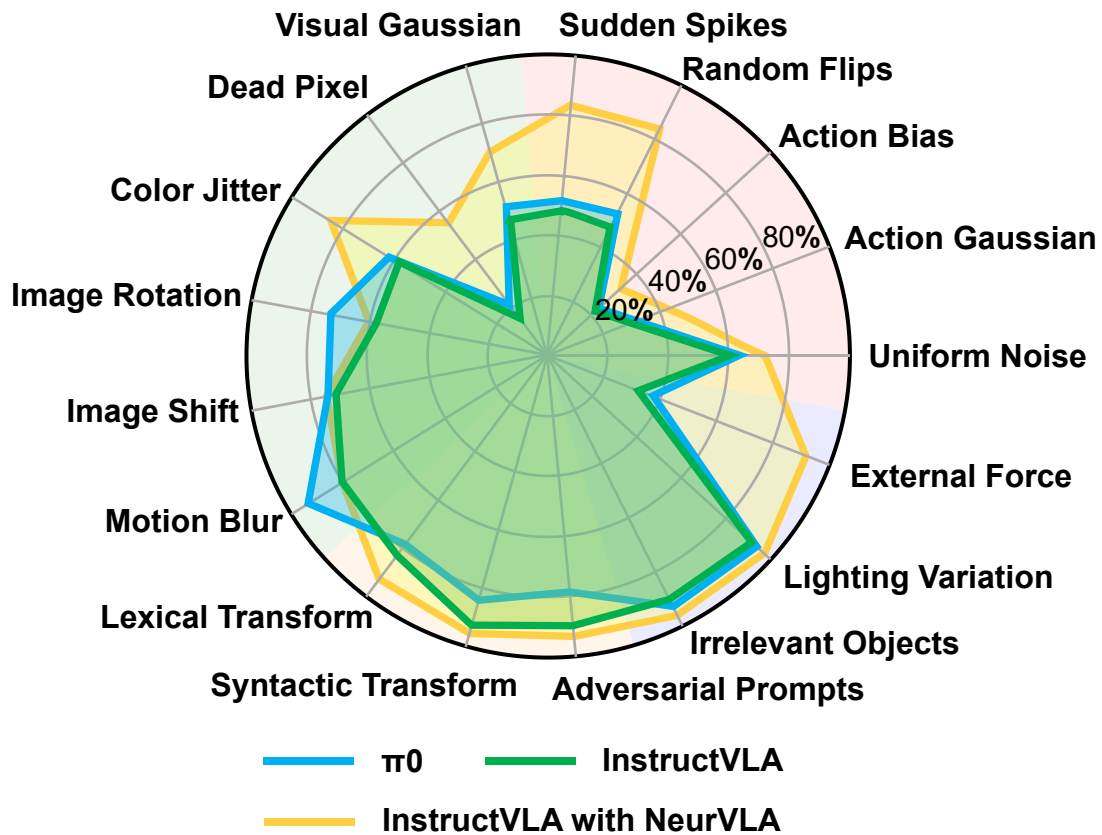


Figure 3. Comparison of NORA, UniVLA, π_0 , and NORA with **NeurVLA** on four real-world tasks.

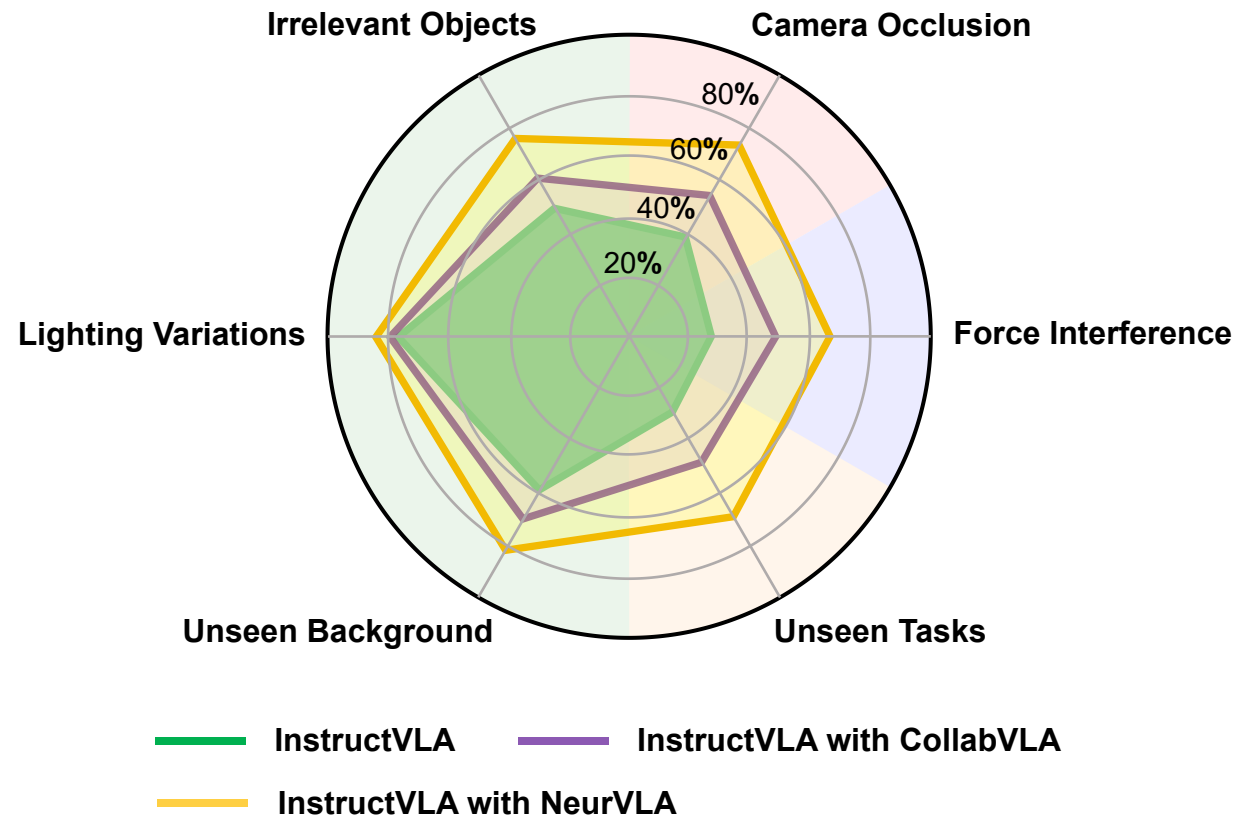


NeurVLA can effectively handle complex daily manipulation tasks that inherently require long-horizon reasoning, fine-grained control, and failure-handling capabilities.

Robustness Experiments



(a) Performance under various simulated perturbations.



(b) Performance under various real-world perturbations.

Figure 4. Robustness of **NeurVLA** across different perturbations in both simulated and real-world environments.



NeurVLA maintains high success rates under action disturbances, environmental changes, visual noise, and external interference, indicating more reliable deployment stability.

Thanks!