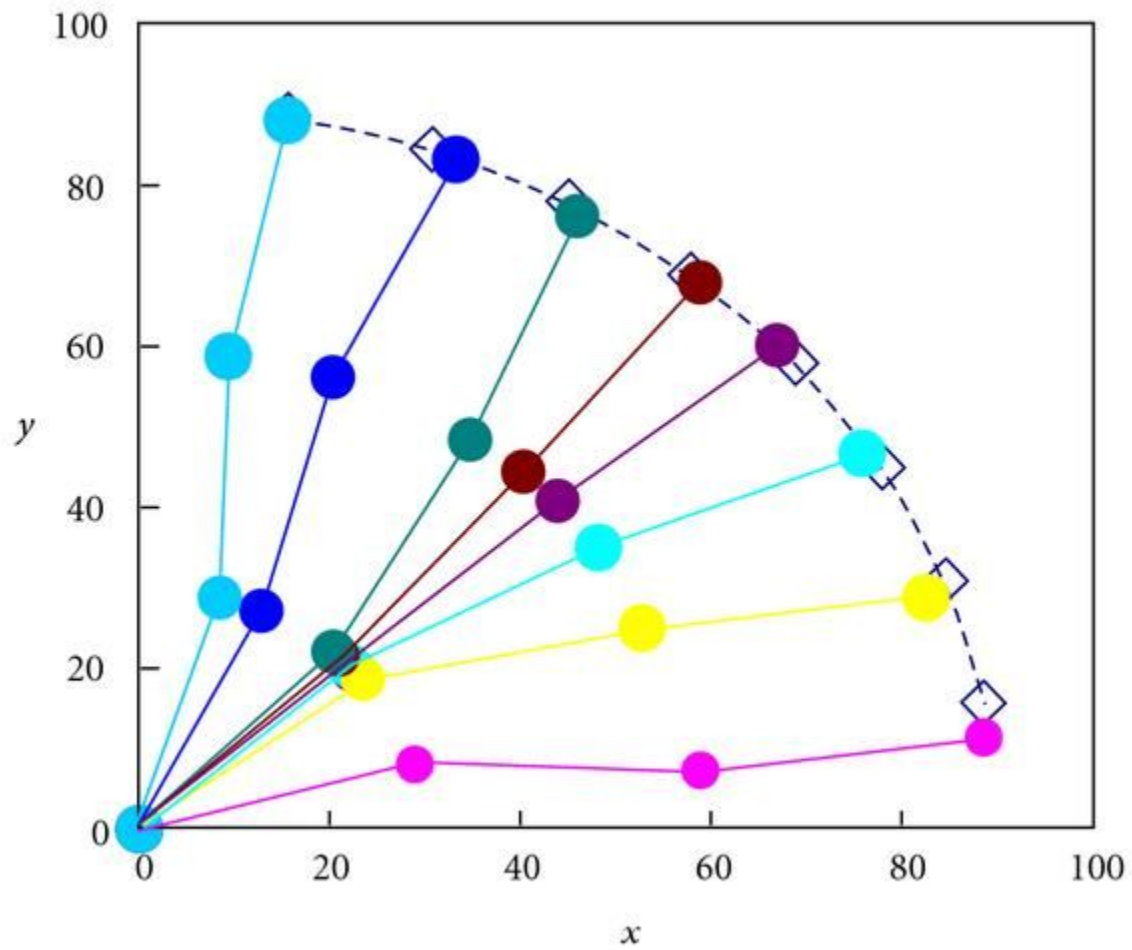


Do What You Say: Steering Vision-Language-Action Models via Runtime Reasoning-Action Alignment Verification

Motivation





“As a result, [a typical VLA’s] capability to handle complex instructions, long-horizon goals, or out-of-distribution scenarios remains limited.”

Huang, Chi-Pin, et al. "Thinkact: Vision-language-action reasoning via reinforced visual latent planning." arXiv preprint arXiv:2507.16815 (2025).

Even with CoT...

The screenshot displays the Cursor IDE interface with a dark theme. The main window is divided into three panes:

- Left Pane (Agents):** Contains a search bar and a list of agents. The selected agent is "Time measurement in datagen.py".
- Center Pane (Chat/Reasoning):** Displays a conversation with an AI agent. The message history includes:
 - Agent: "datagen.py (147-150) didn't I make this a generator statement? How come on each pose it doesn't save an capture to disk?"
 - Thought for 8s
 - Reasoning: "Making it a generator expression doesn't change when things hit disk — it just changes when the Capture objects are created (lazily vs. eagerly). convert_to_dataset still receives the whole sequence and processes it in two passes:"
 - Code Snippet:

```
index = build_index(captures)
index_path = index_dir / f"index{scene_tag}.json"
with open(index_path, "w") as f:
    json.dump(index, f, indent=2)

write_capture(captures, render_dir)
```
 - Conclusion: "build_index iterates captures first, then write_capture iterates it again. With a generator that's actually a bug — the first pass consumes it, so write_capture sees nothing. To write each capture as it's produced, you'd need to restructure things so the writing happens inside the loop rather than after it. The current convert_to_dataset is a batch operation by design."
- Right Pane (Code Editor):** Shows the Python code for `poses_to_scene` and `make_capture` functions. The `make_capture` function uses `np.linalg.inv` to calculate the camera calibration matrix.

At the bottom, a terminal window shows system logs and network activity, including messages like "Socket close event received" and "forwarding established".

Even with CoT...

Time measurement in datagen.py

datagen.py (147-150) didn't I make this a generator statement? How come on each pose it doesn't save a capture to disk?

Thought for 8s

Making it a generator expression doesn't change when things hit disk — it just changes when the Capture objects are created (lazily vs. eagerly). convert_to_dataset still receives the whole sequence and processes it in two passes:

```
def convert_capture.py Lines 84-89
    index = build_index(captures)
    index_path = index_dir / f"index{scene_tag}.json"
    with open(index_path, "w") as f:
        json.dump(index, f, indent=2)
    write_capture(captures, render_dir)
```

build_index iterates captures first, then write_capture iterates it again. With a generator that's actually a bug — the first pass consumes it, so write_capture sees nothing.

To write each capture as it's produced, you'd need to restructure things so the writing happens inside the loop rather than after it. The current convert_to_dataset is a batch operation by design.

```
def poses_to_scene(base_frame_poses):
    first = tw.Node(geometry=axis,
    last = tw.Node(geometry=axis,
    tw.add_node(scene, first, parent
    tw.add_node(scene, last, parent

    tw.ground_plane(scene, z=target
    add_label_billboards(scene, [ba

    return scene

def make_capture(robot2base, target
    robot2target = np.linalg.inv(ta
    return Capture(
        offset=robot2target[:3, -1]
        robot2base=robot2base,
        euler_target_to_robot=R.fro
        left_image=zed_pack.left_im
        left_depth=zed_pack.left_de
        right_image=zed_pack.right_
        left_calib=zed_pack.left_K,
```

2026-04-06 09:05:56.077 [info] [command] [...]
Socket close event received
2026-04-06 09:05:56.098 [info] [for
1:34323] [827a3bb2-3ae0-4e39-aa17-3d:
2026-04-06 09:05:56.104 [info] [for
1:62646 -> 127.0.0.1:34323] [827a3bb:
forwarding established
2026-04-06 09:05:56.122 [info] [for
1:34323] [a0e1545a-f819-41ae-ba69-fa:
2026-04-06 09:05:56.127 [info] [for
1:62646 -> 127.0.0.1:34323] [a0e1545:
forwarding established
2026-04-06 09:05:56.232 [info] Saved



What the VLA says



What the VLA actually does

Problem statement: can we make them think and **also** do what they say

Steering for **E**mbodied Reasoning-**A**ction **A**lignment



SEAL

VLA-CoT

**Verify actions
align with
reasoning**

Outline

The motivation and problem statement - **done**

Related work

The method

Results

Limitations and future work

Related work

SEAL

The diagram features a central orange rounded rectangle labeled 'SEAL'. To its left is a light red oval labeled 'CoT'. To its right is a light purple oval labeled 'Skill learning'. Below the 'SEAL' rectangle is a large yellow oval labeled 'TAMP'. The text 'Related work' is positioned to the left of the 'CoT' oval.

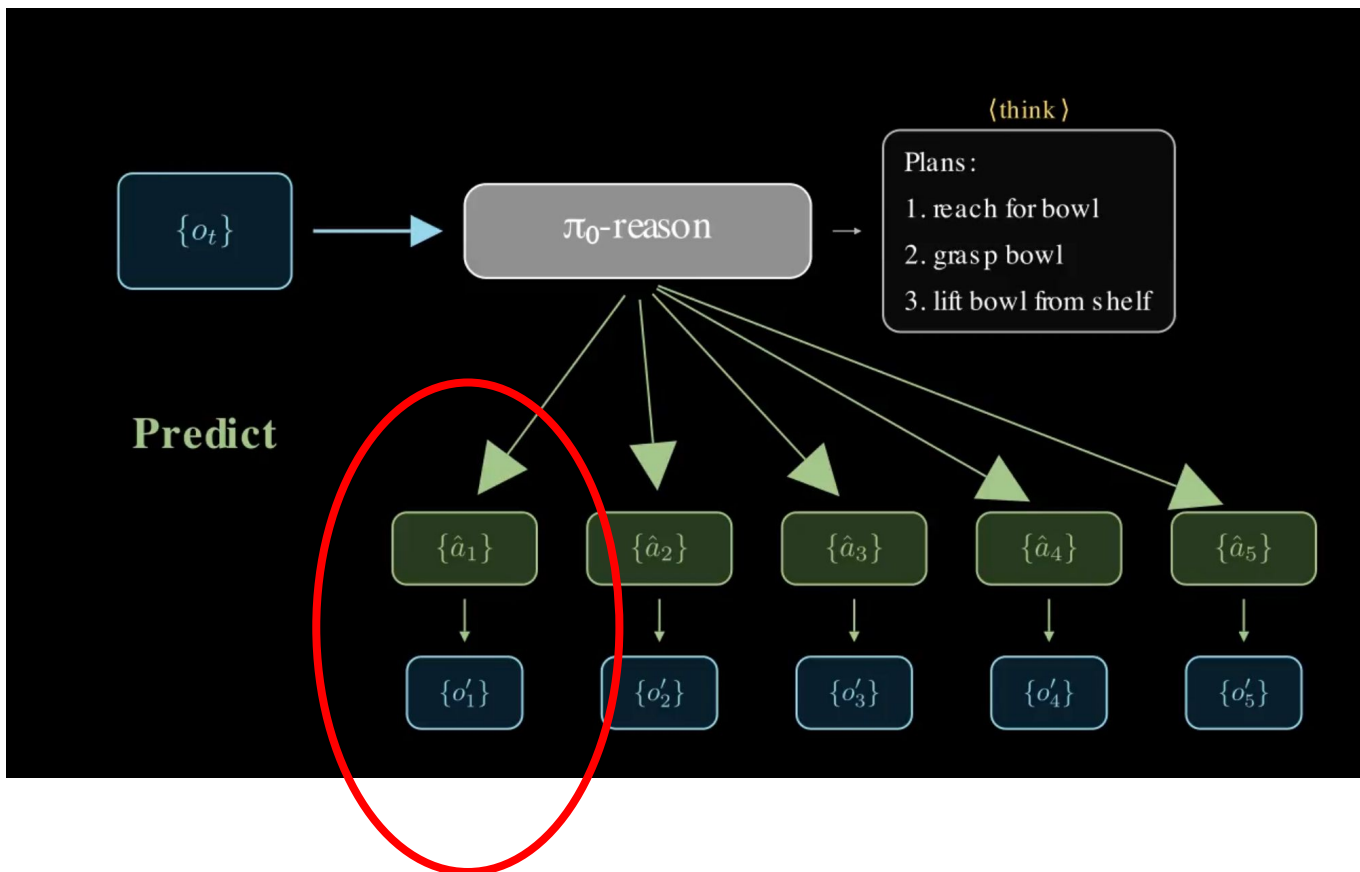
CoT

Skill learning

TAMP

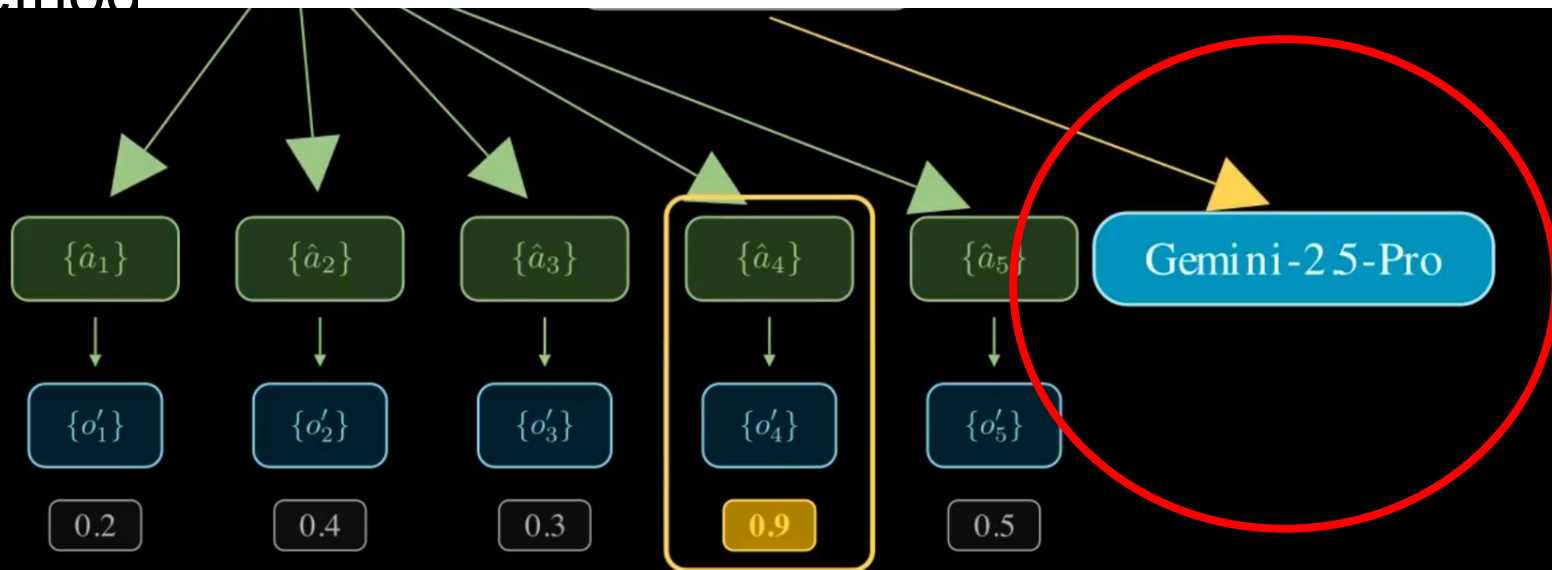
The method

The method



The method

Verify



The method

Gemini-2.5-Pro

reach for bowl

grasp bowl

lift bowl from shelf

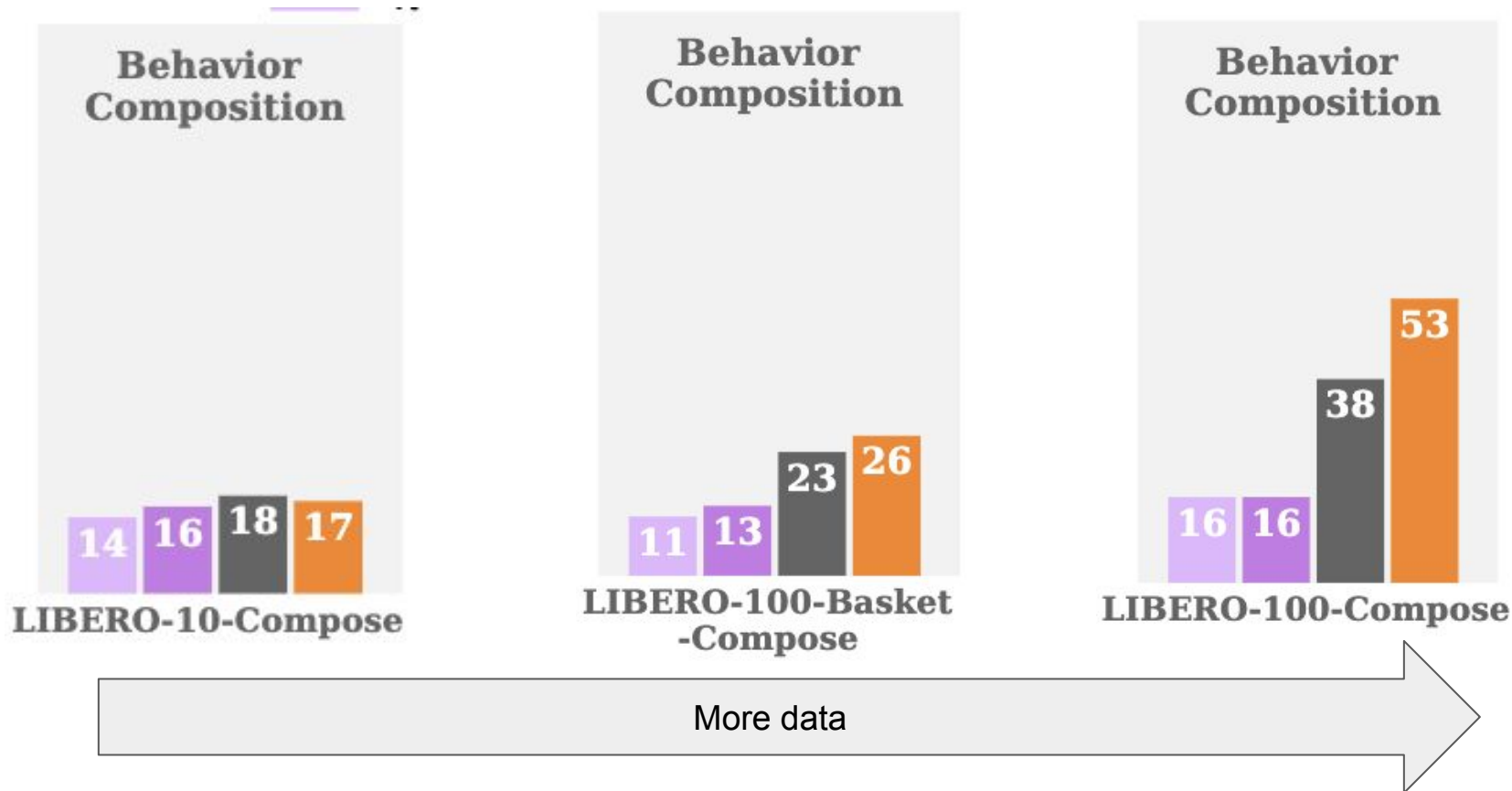
$\{ (o_1, a_1) , (o_2, a_2) , (o_3, a_3) , (o_4, a_4) , (o_5, a_5) , (o_6, a_6) \}$

Results

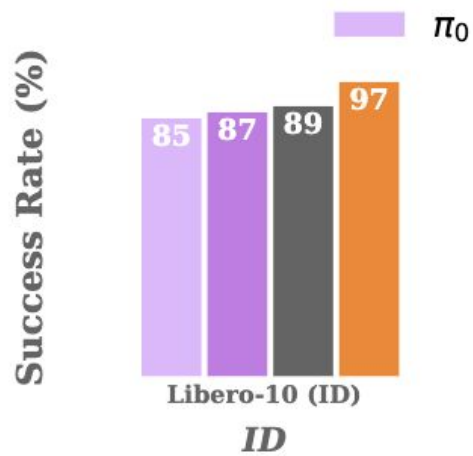
Results



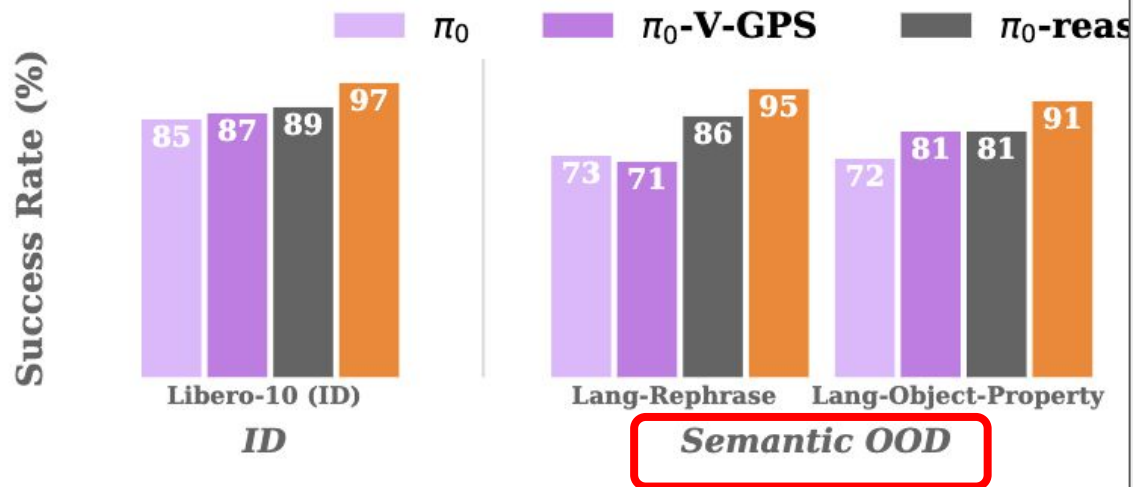
Results



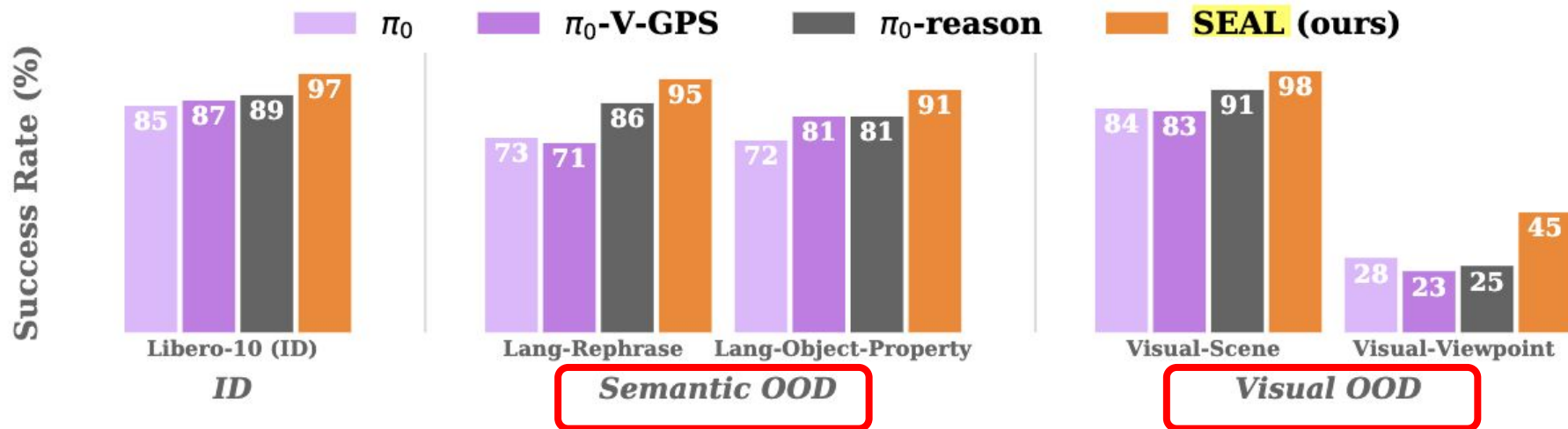
Results



Results



Results



Limitations & future work

-

