

RSI for Embodied Agents

Manling Li
Northwestern University
manling.li@northwestern.edu



NORTHWESTERN
UNIVERSITY



MLL LAB
Machine Learning and Language

How Models Learn (and Fail to Learn) the Physical World



*A robot arm is placing
the banana on top of
the coaster*



What we expect



banana → monkey → jump

Output from Wan2.2

World Models

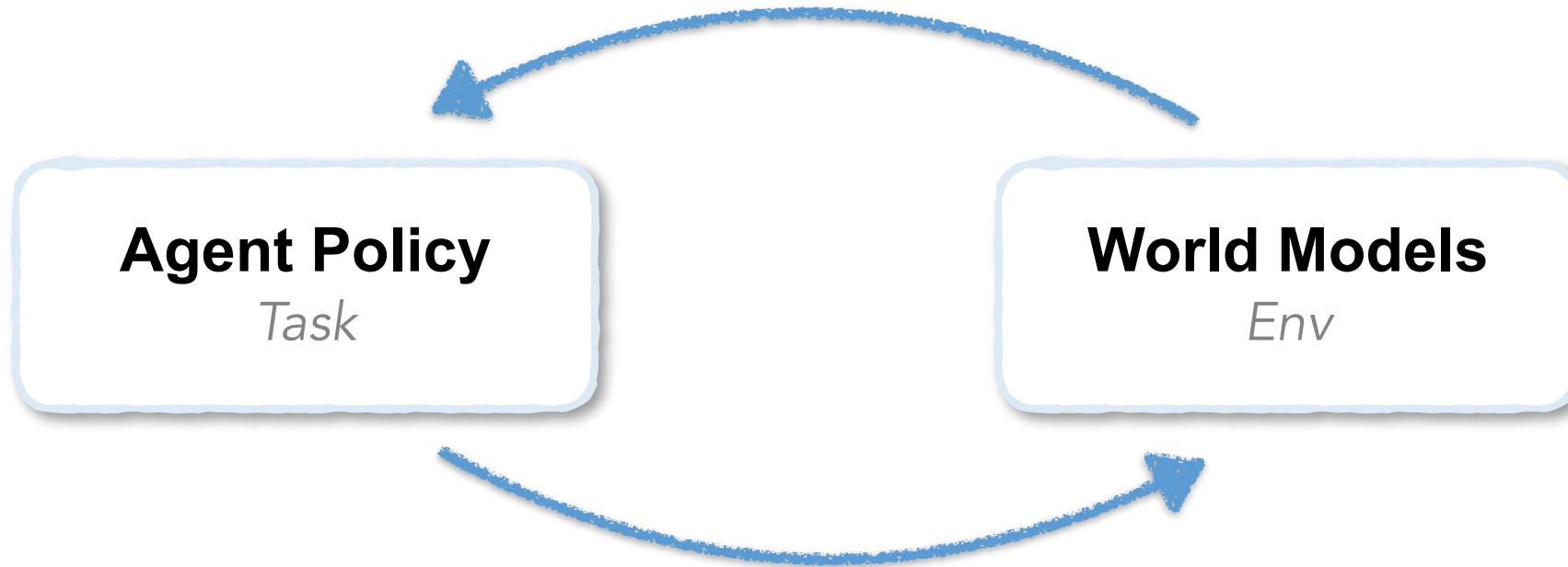
what we want

**?
==**

Surface Behavior

what we truly learn

Co-Evolving: Let's train them!

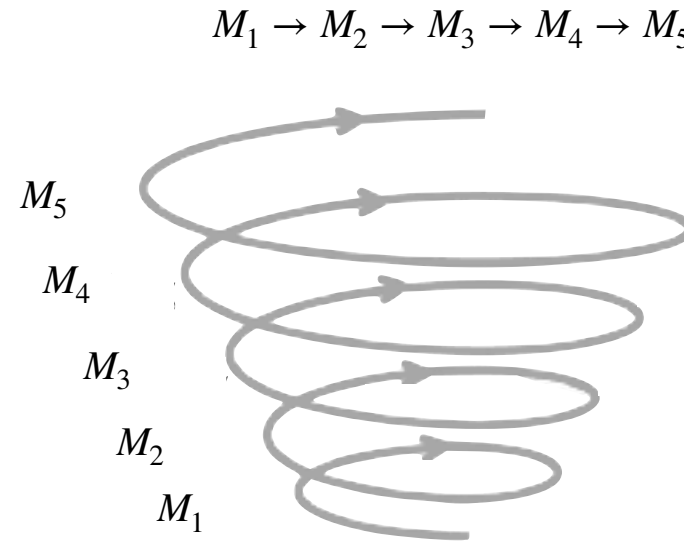


**World models are not about surface behavior,
but the reasoning that matters to the downstream agents**

The definition of Recursive Self-Improvement (RSI)



Irving John Good (1965)

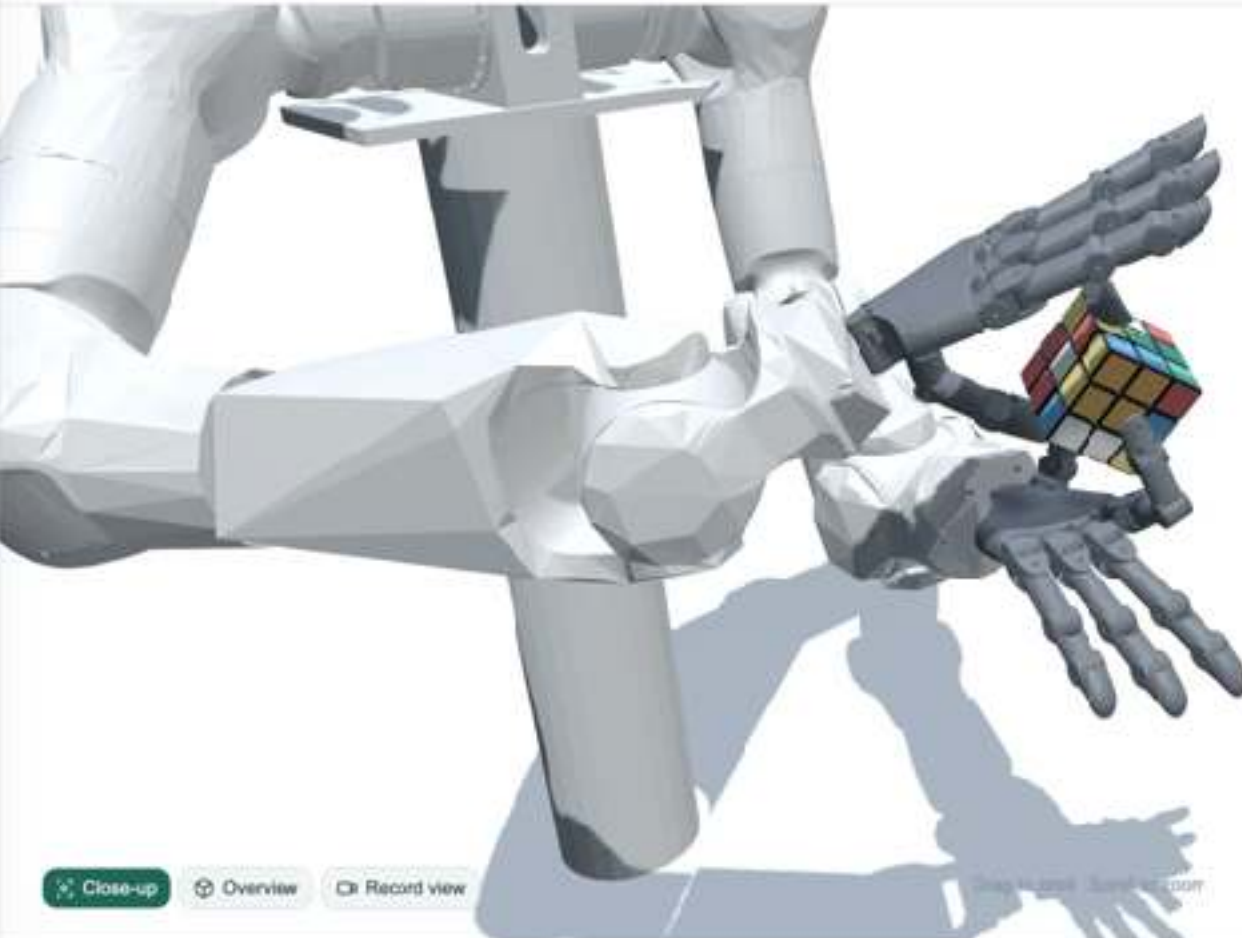


Agentic Systems: GPT-6 for Problem Solving

Yanjie Ze's playground

Dexterous Cube Solving

Physics replay



Cube

Turning

Last aligned state

ACTION

Reorienting

R Move 4 / 10

Details

Close-up Overview Record view

00:28.6 01:58.5

Speed 0.5x 1x 2x

Sim time 110.54 s

Control-driven MultiCo

Move sequence

Agentic Systems: GPT-6 for Problem Solving

Yanjie Ze's
playground

Dexterous Cube Solving

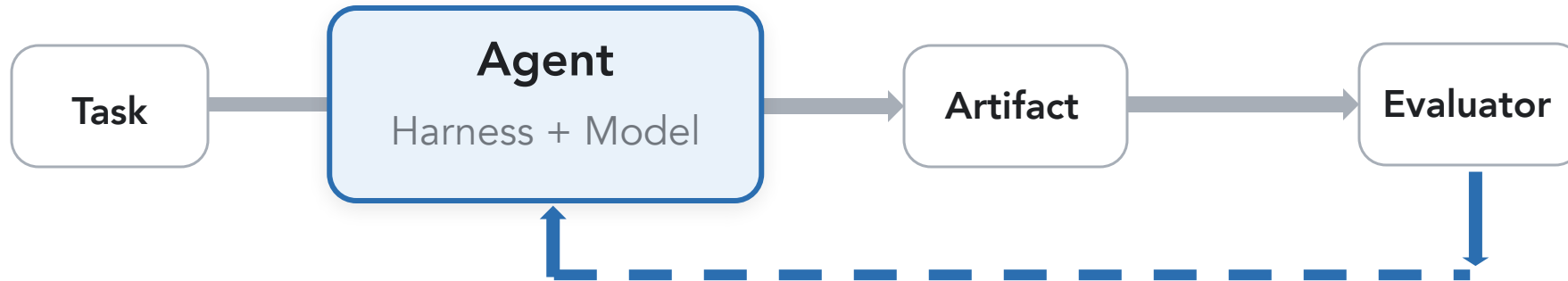
• Physics replay

What can't GPT-6 do well?

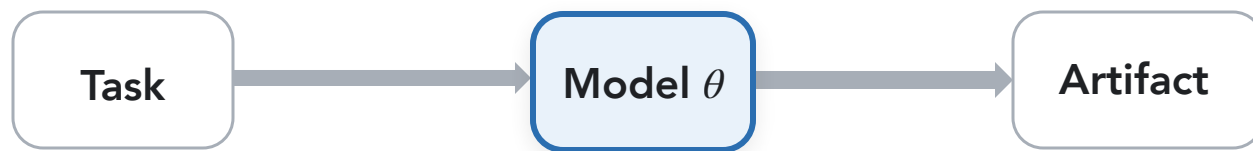
19/20 on a block-to-bowl task,
but only 2/20 on a precise puzzle-insertion-to-groove task

latency: 2.5-3.4 minutes in average

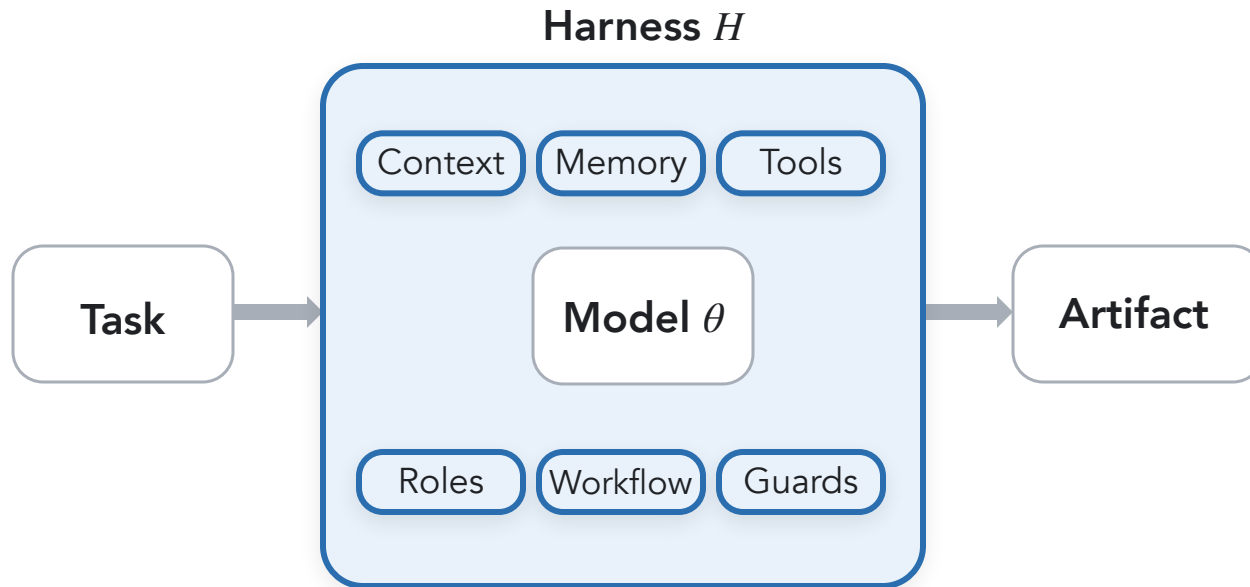
Self-Improving Agents → Generalization / Continual Learning



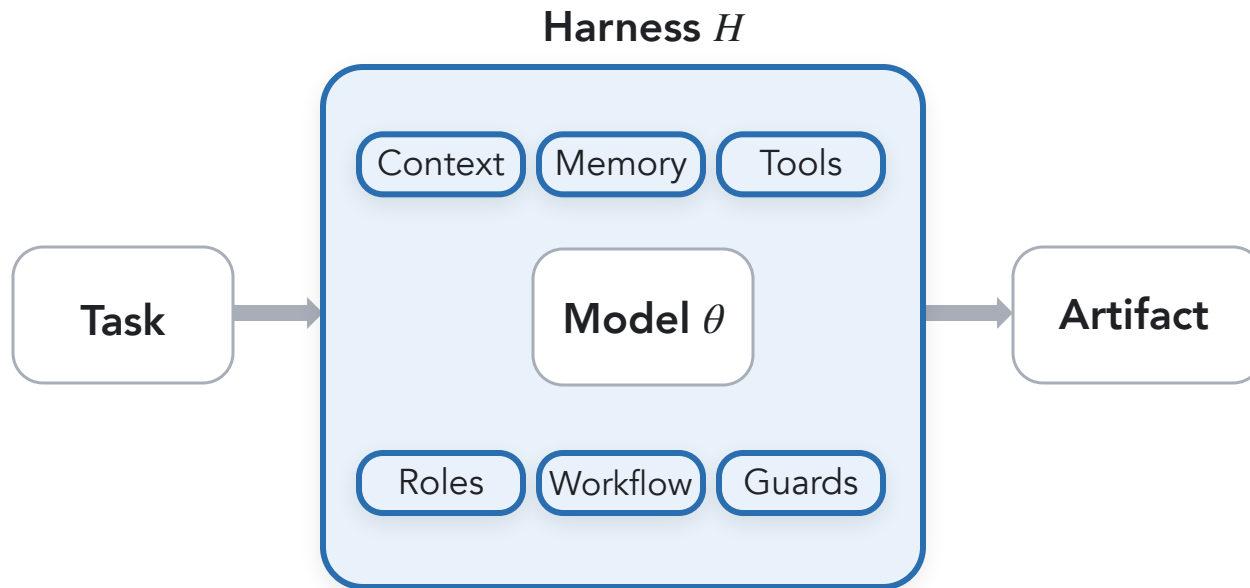
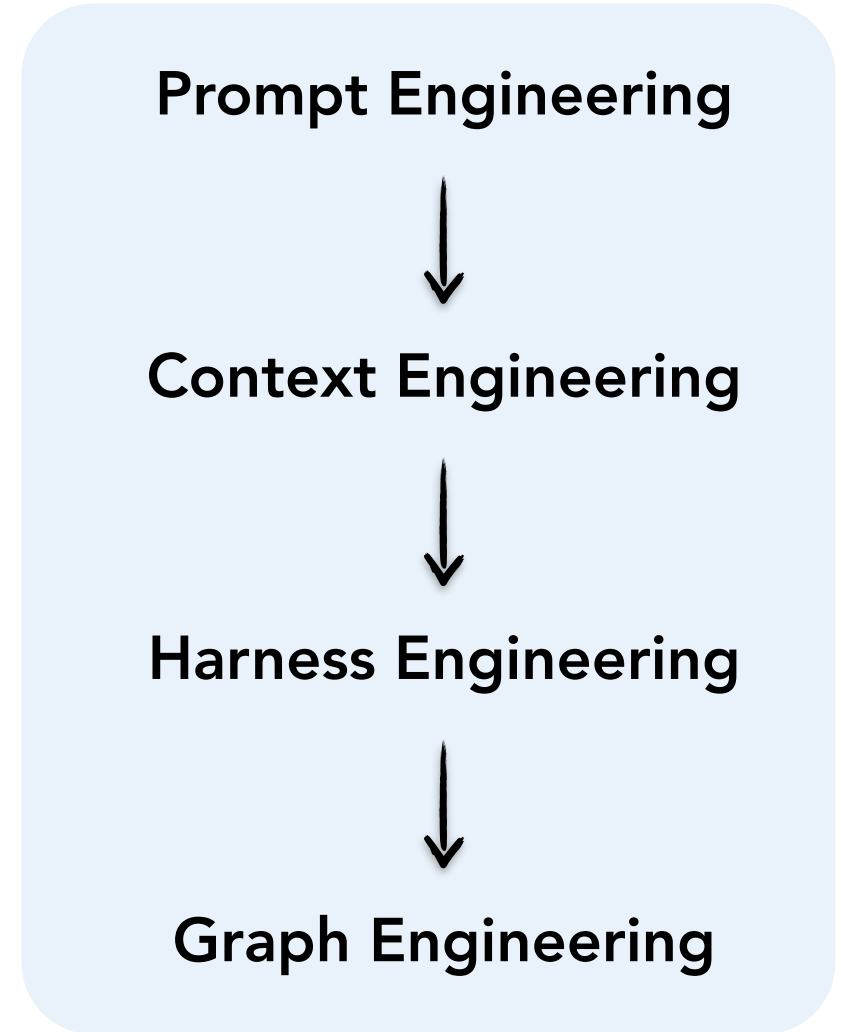
Agent = Model + Harness



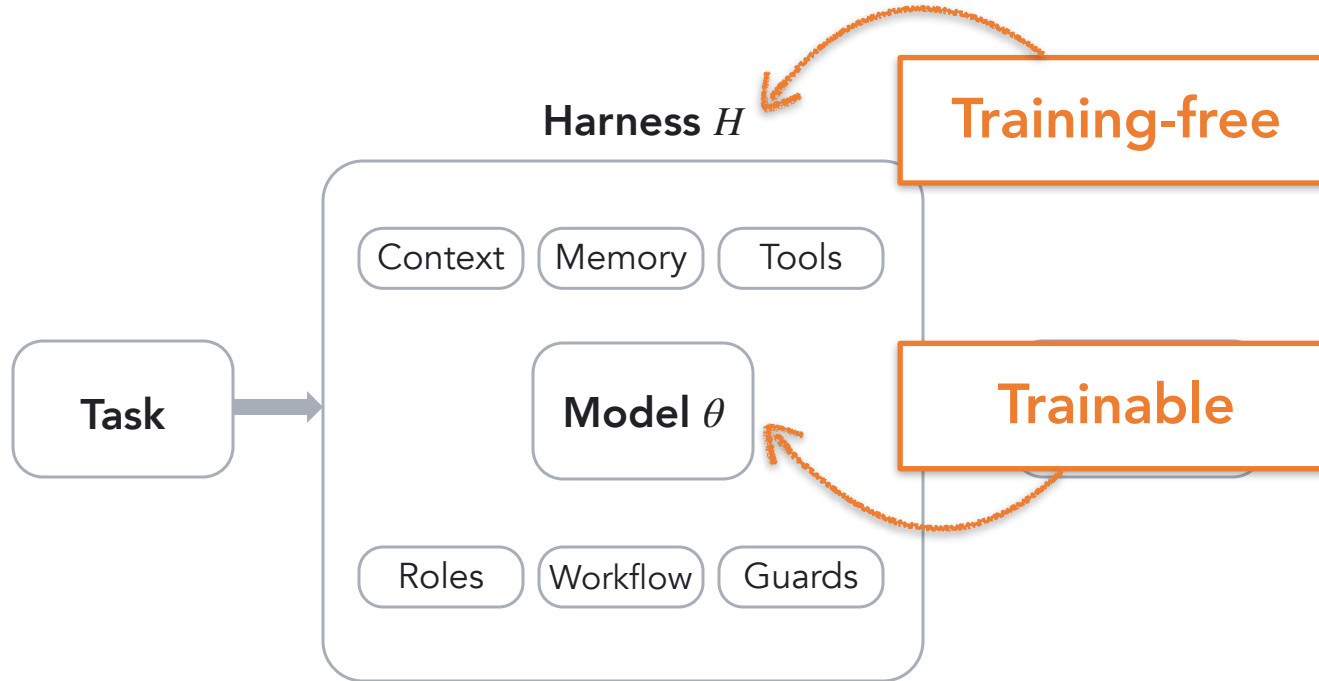
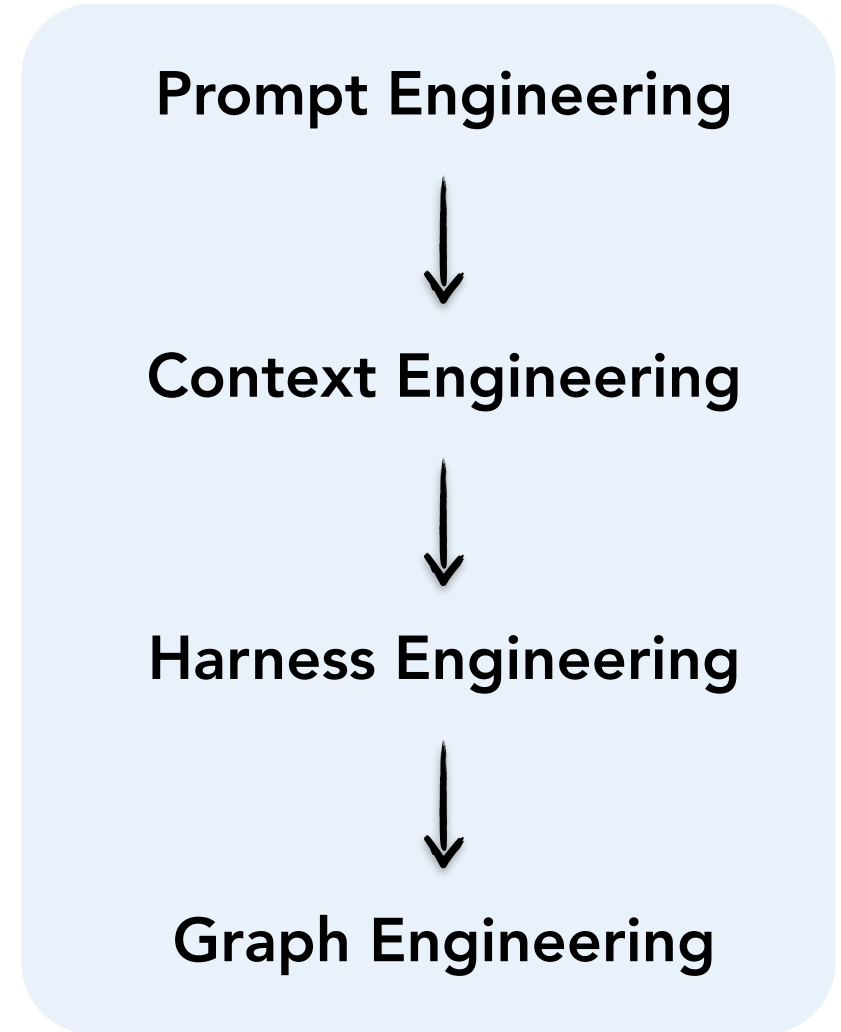
Agent = Model + Harness



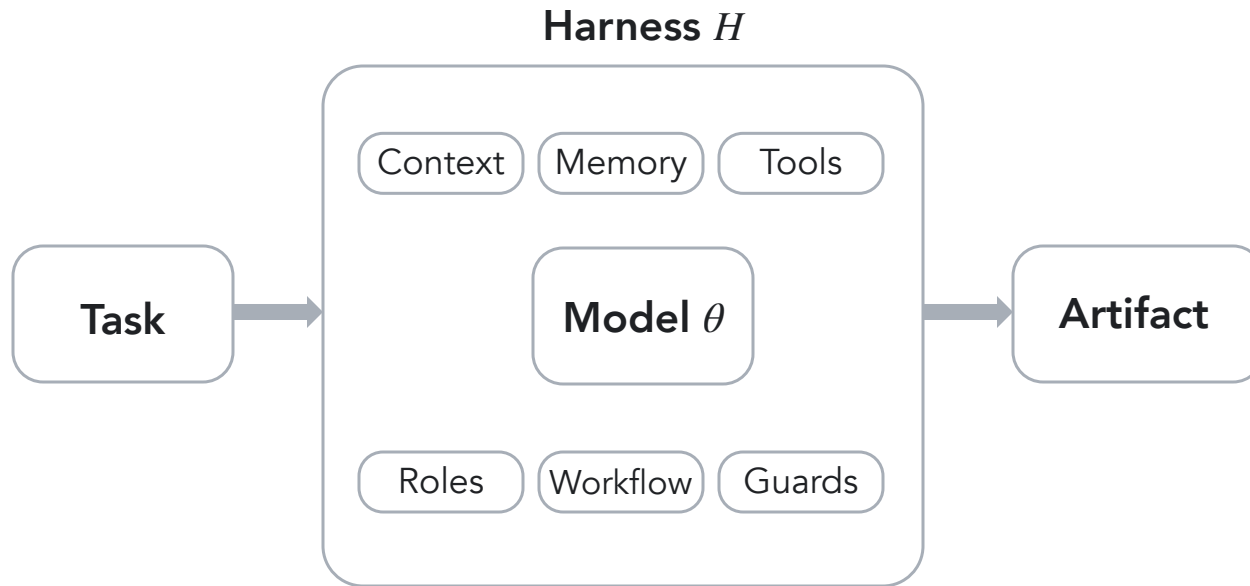
Agent = Model + Harness



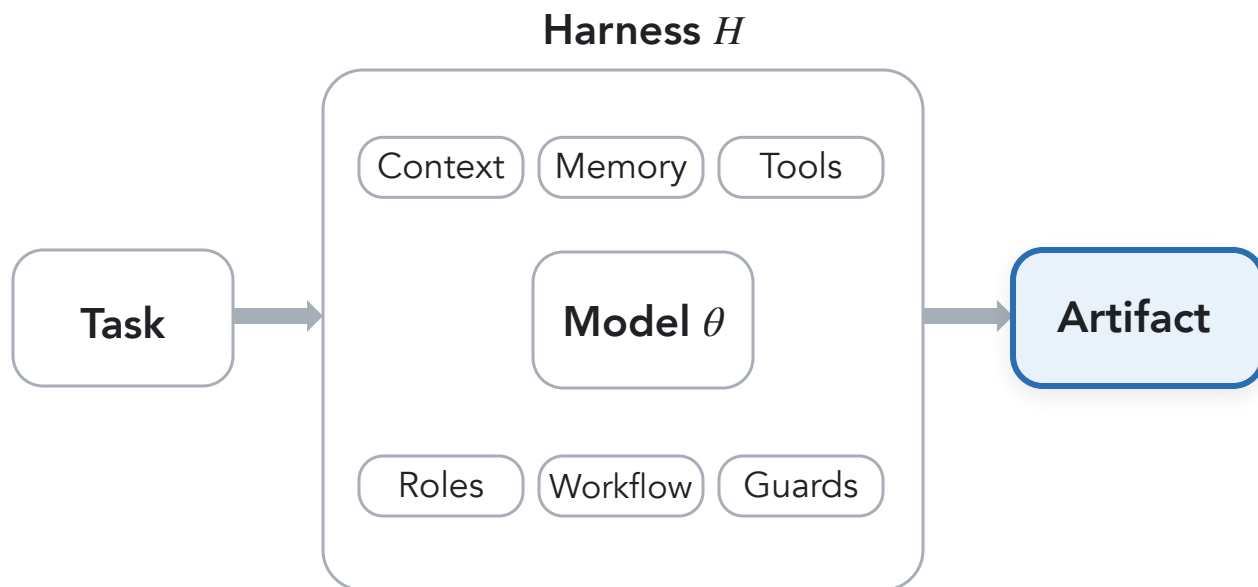
Agent = Model + Harness



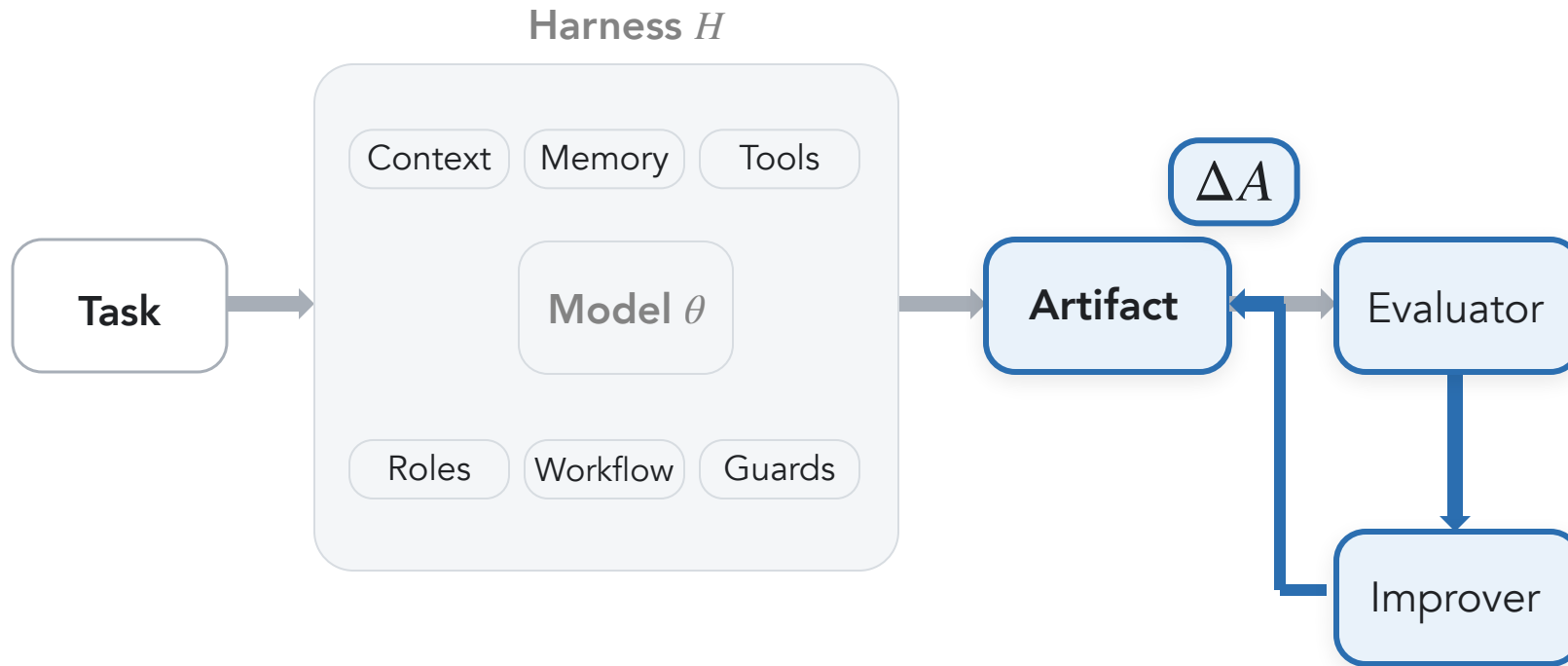
Agent = Model + Harness



Layer 1: Artifact RSI



Layer 1: Artifact RSI



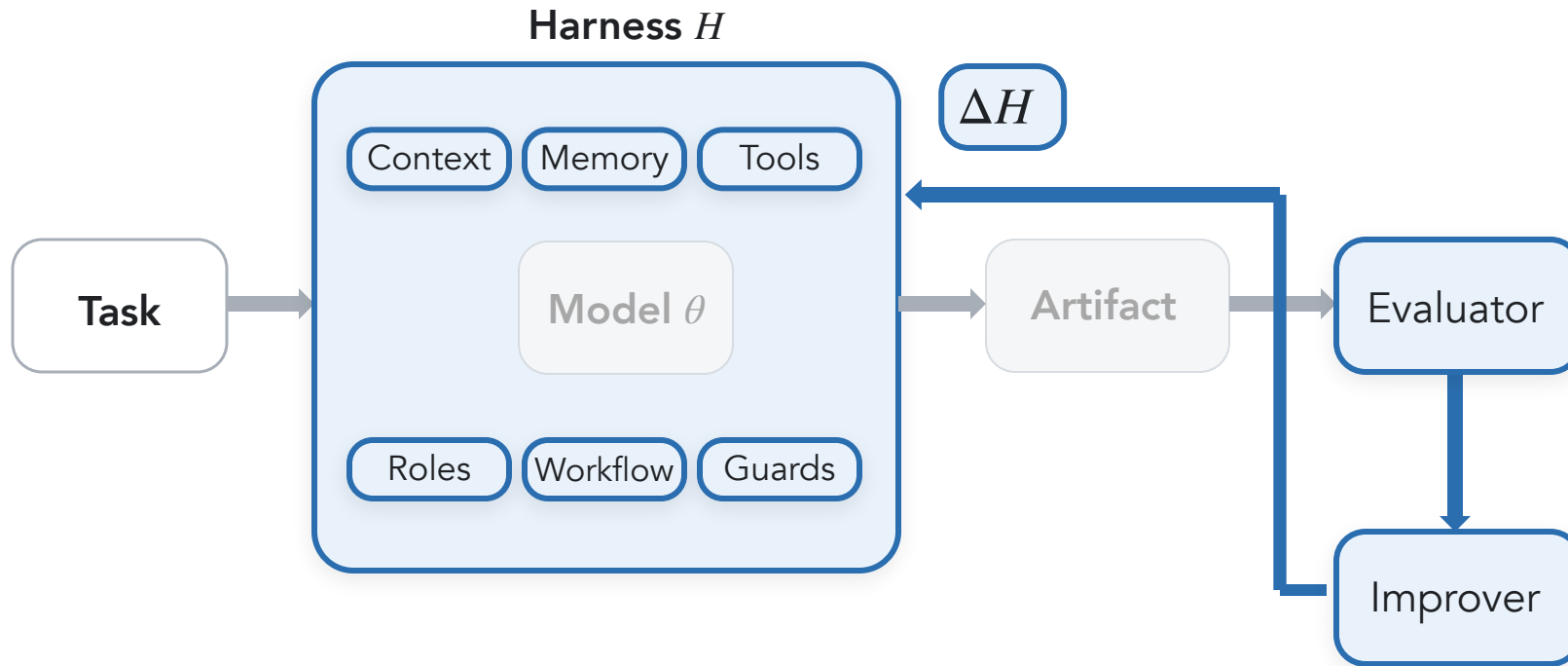
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Layer 2: Harness RSI



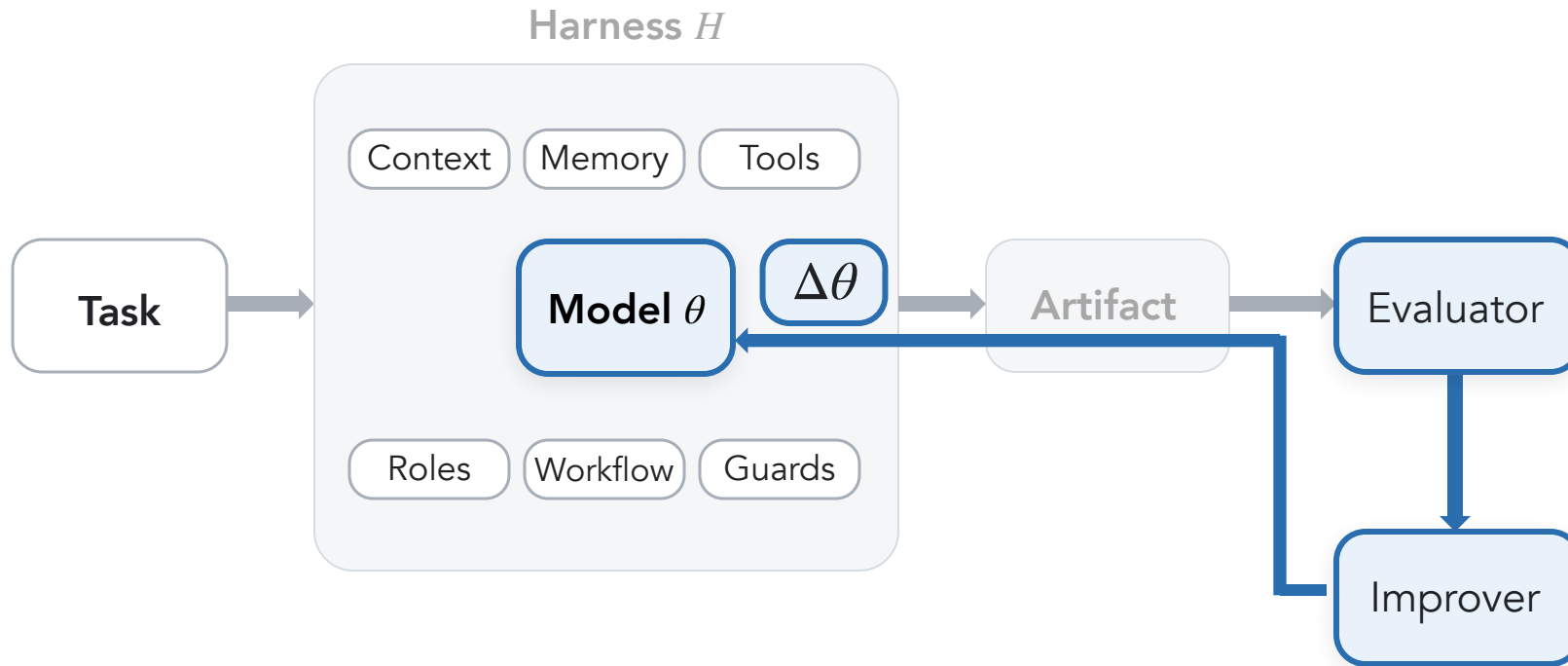
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Layer 3: Model RSI



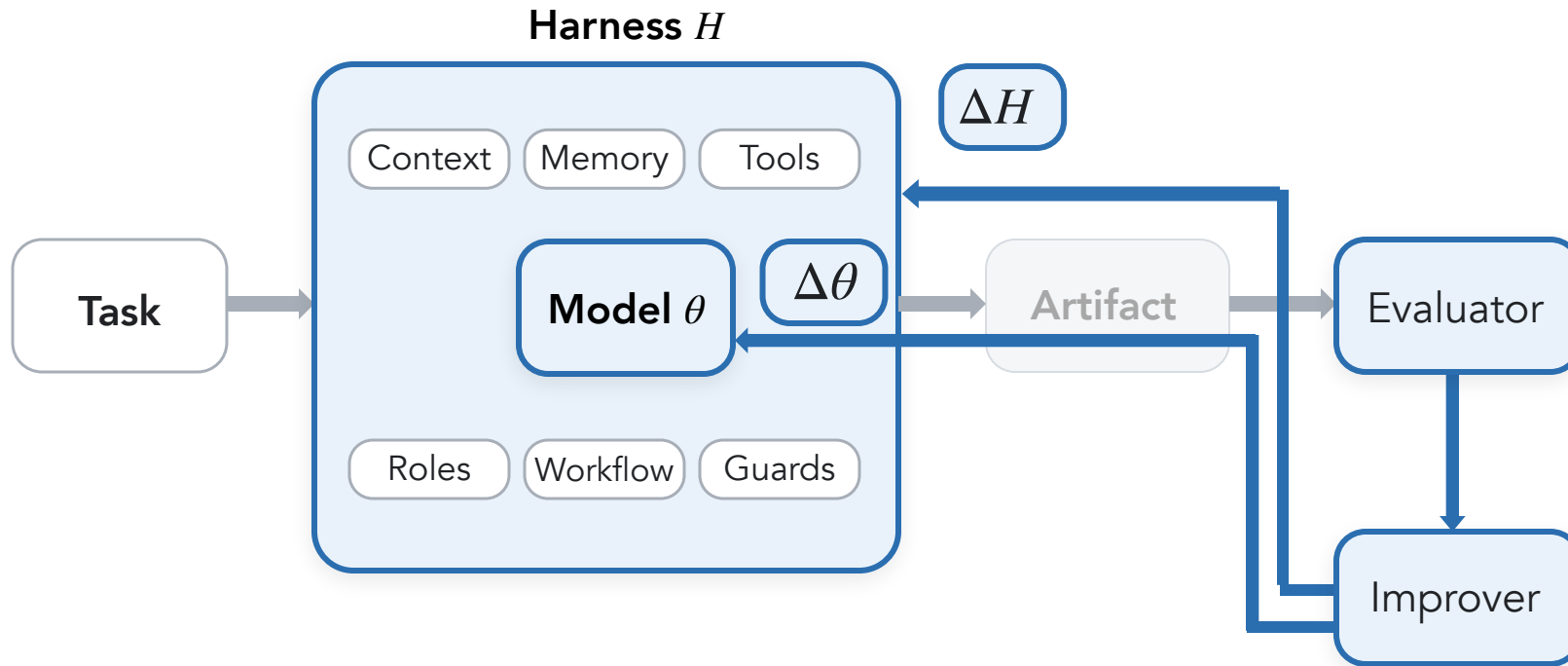
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

The layers can co-evolve



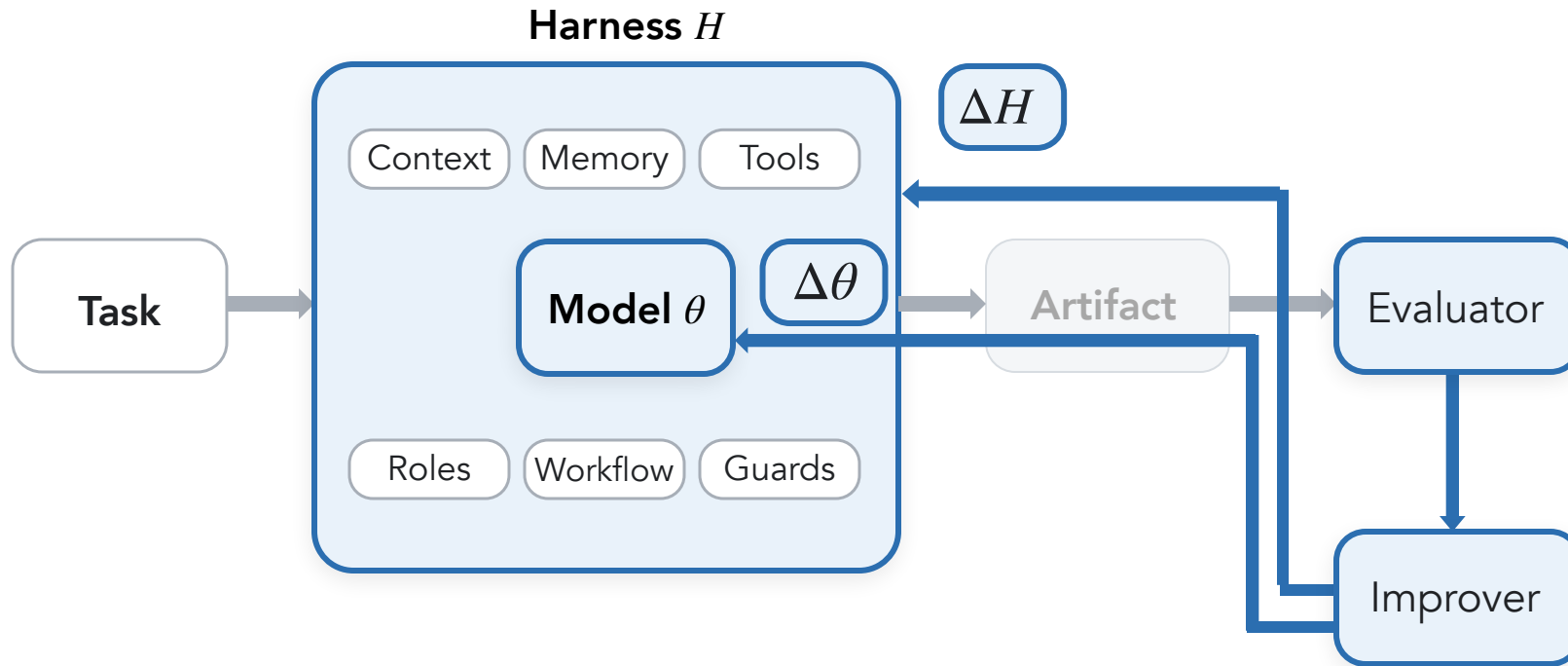
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Talk Outline



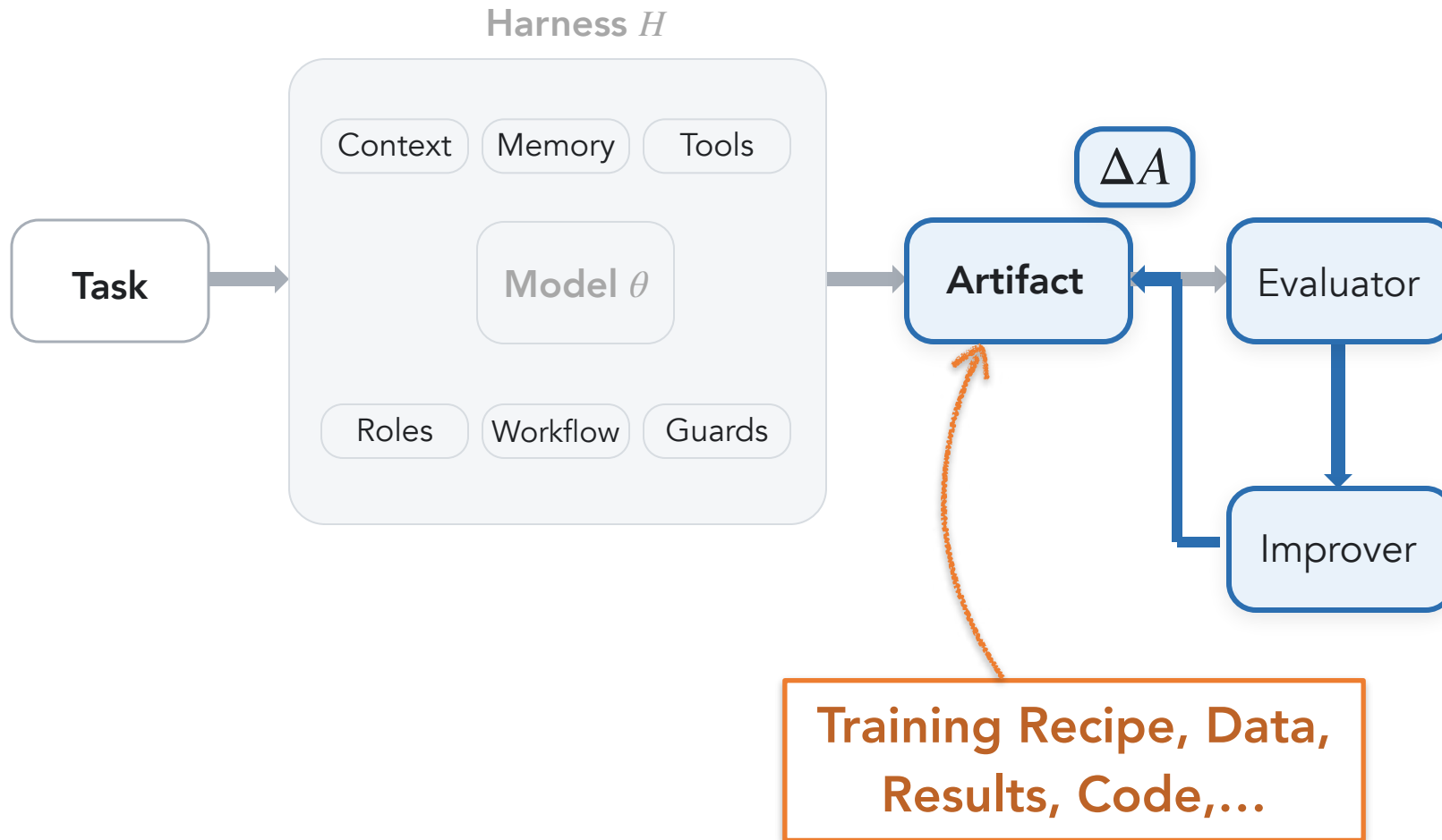
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Layer 1: Artifact RSI



AutoResearch

AlphaEvolve

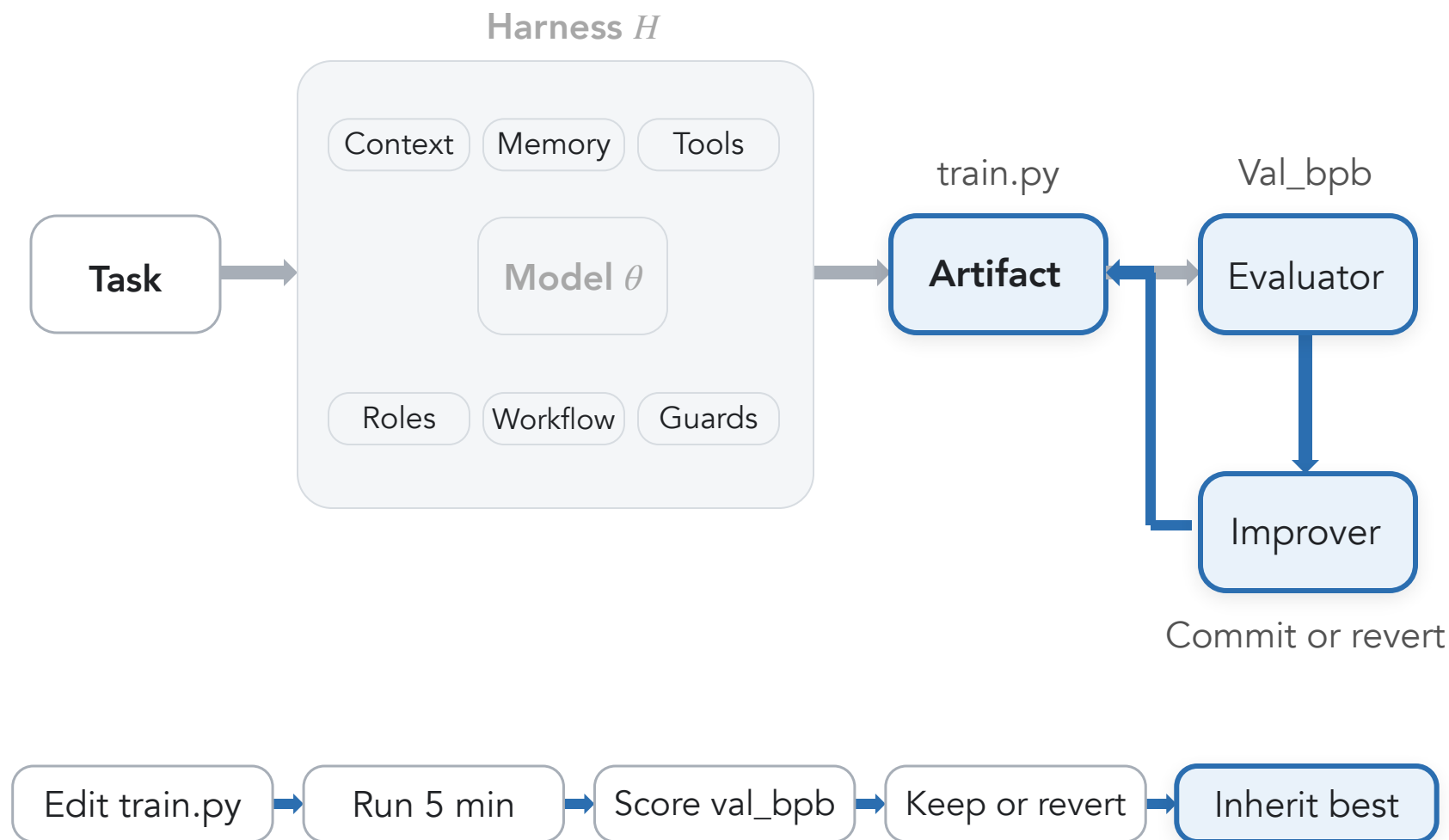
ENPIRE

...

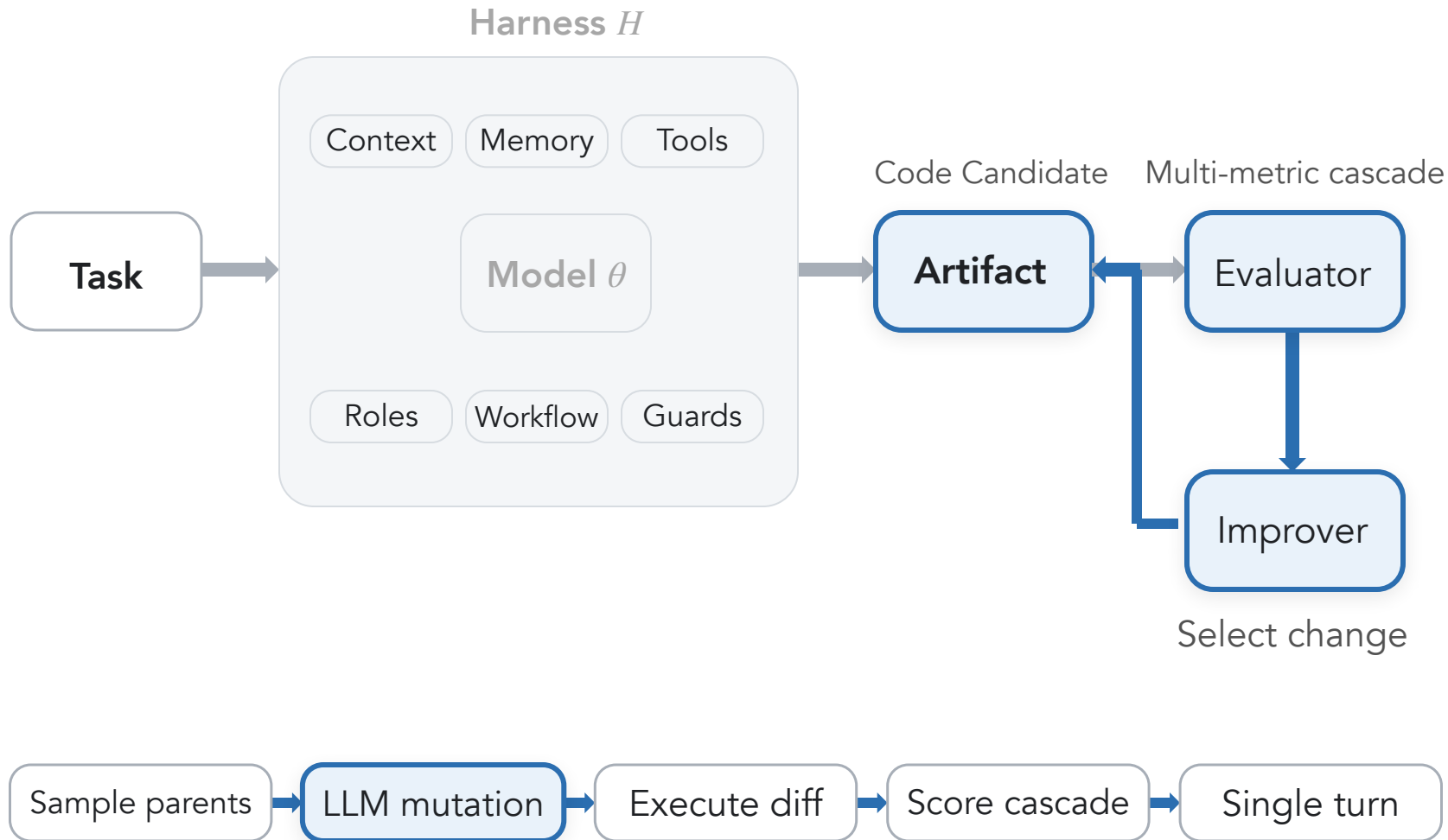
Layer 1: Artifact RSI



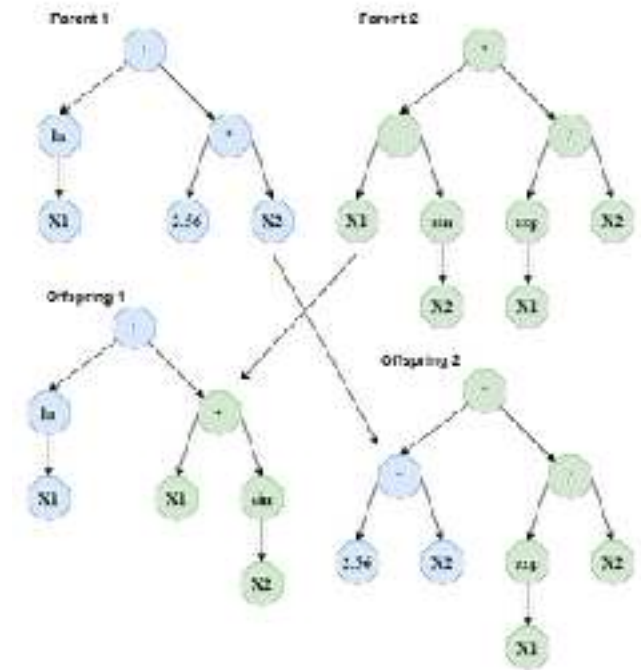
AutoResearch Karpathy Loop (2026.03)



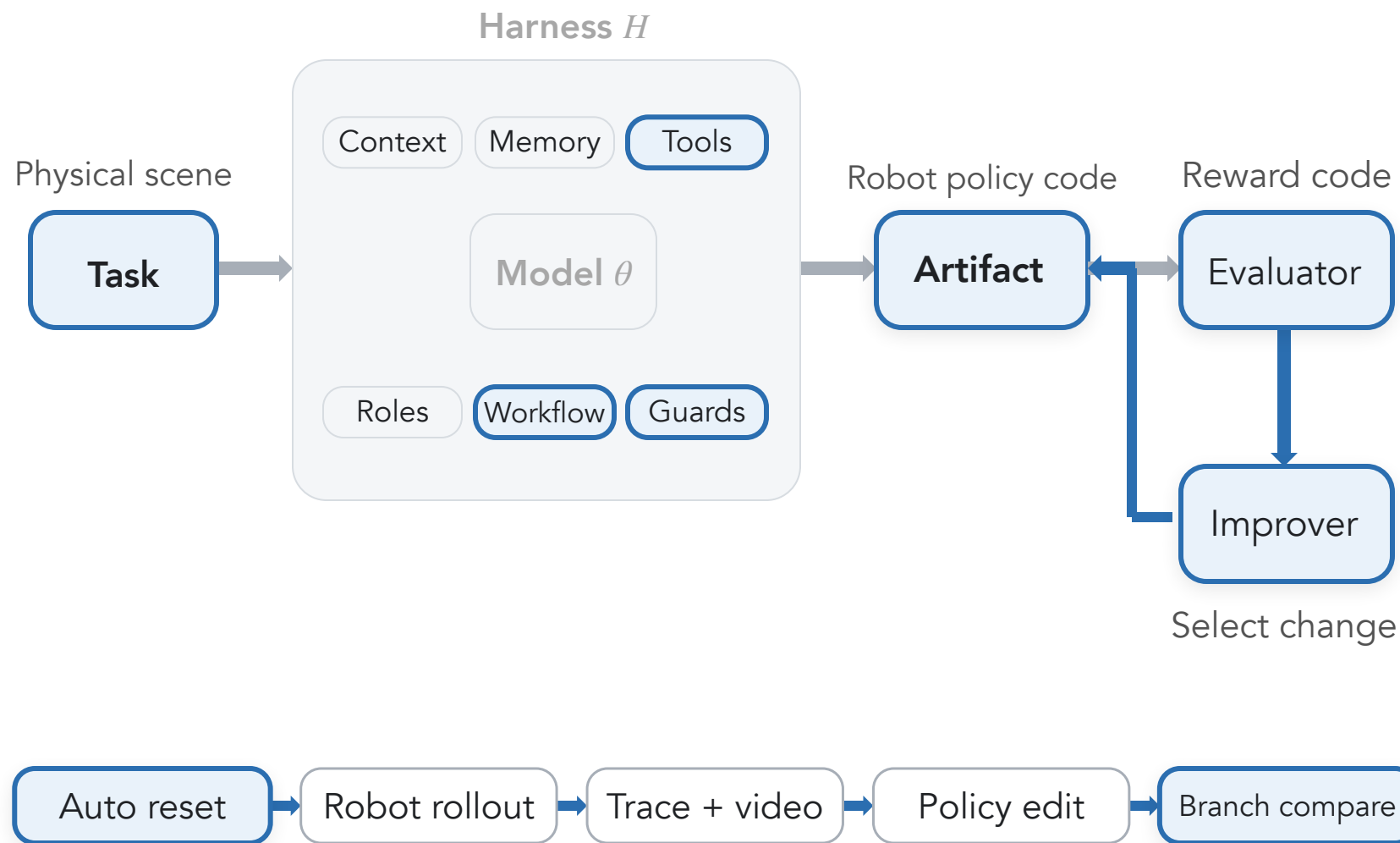
Layer 1: Artifact RSI



AlphaEvolve



Layer 1: Artifact RSI



ENPIRE

**AutoResearch
for Robots**

Resettable World
`reset()`

Reward Function
`get_reward()`

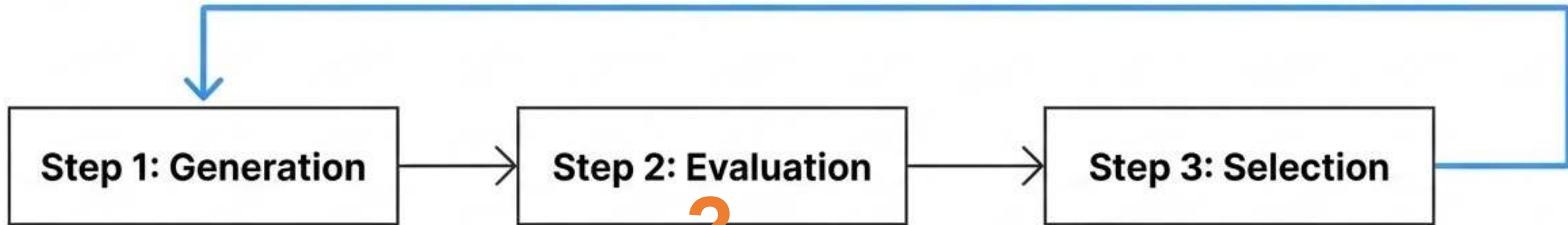
*Up to 99% real-robot success
on pin insertion, etc*

Takeaways

1. The key of AutoResearch is the evaluator

Outer Loop: AutoResearch

Step 4: The optimal code feeds back into the next generation.

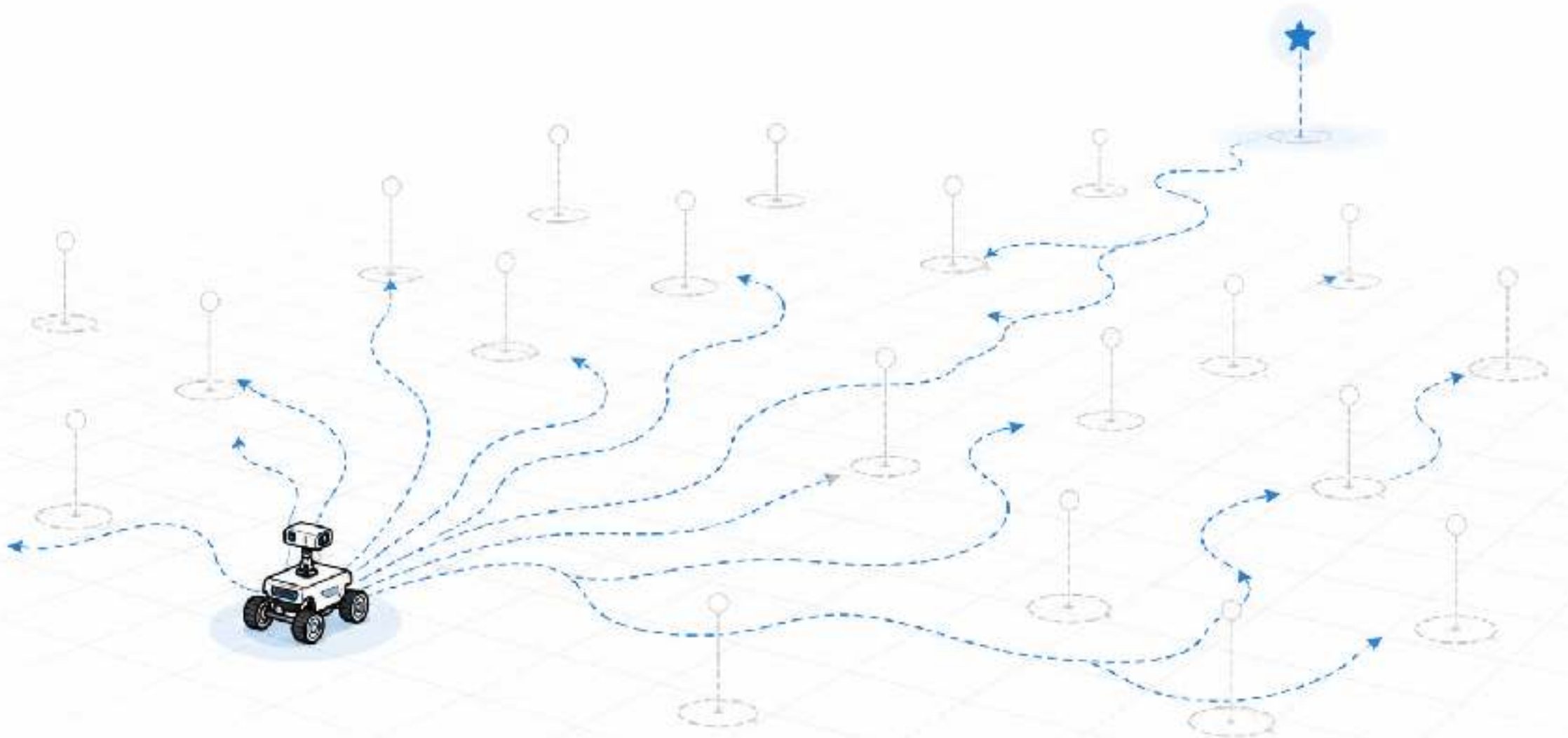


?

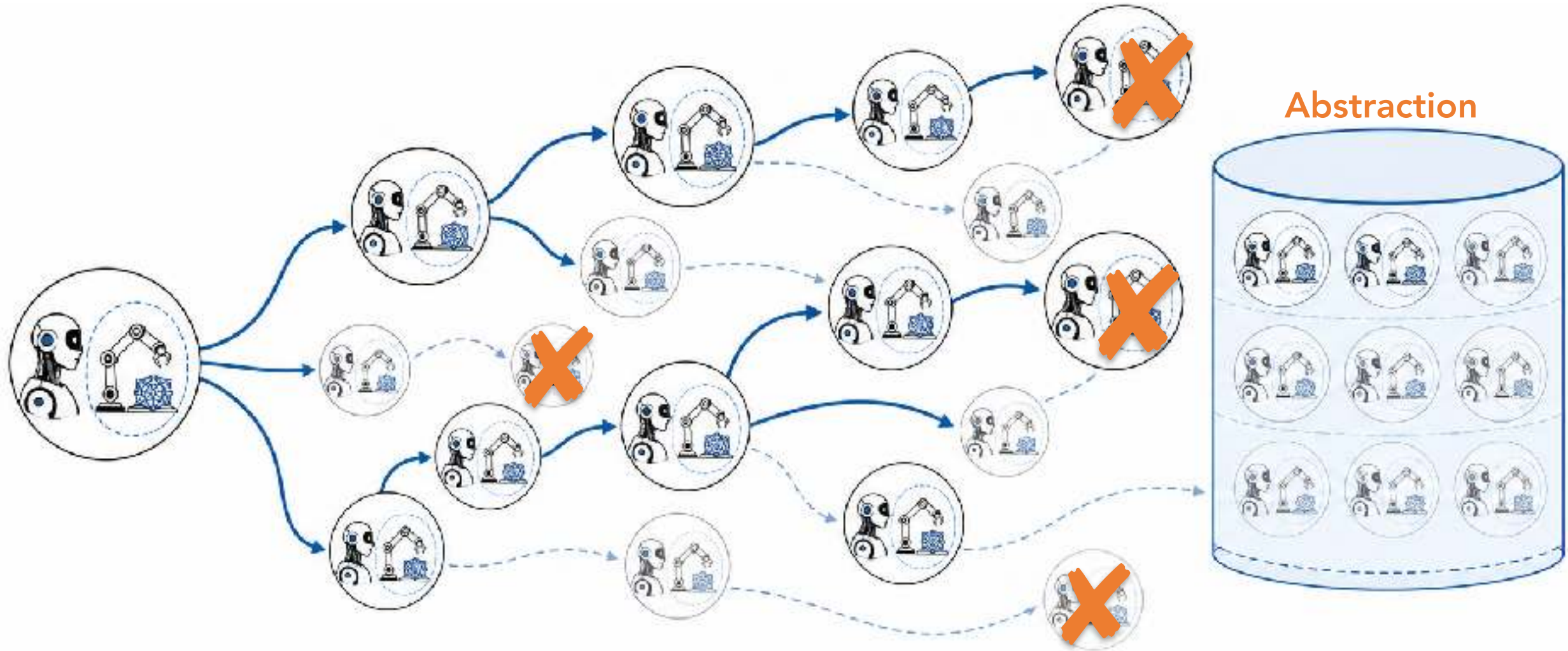
Current bottleneck.

Atlas

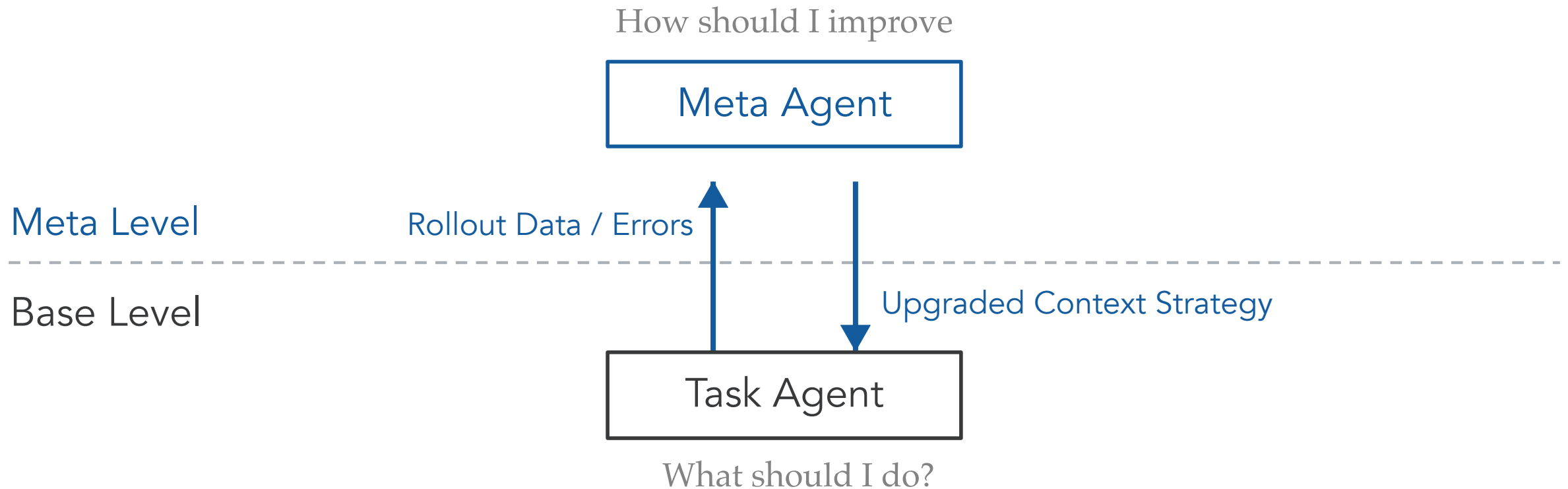
AutoResearch is essentially "searching"



DeepResearch → AutoResearch



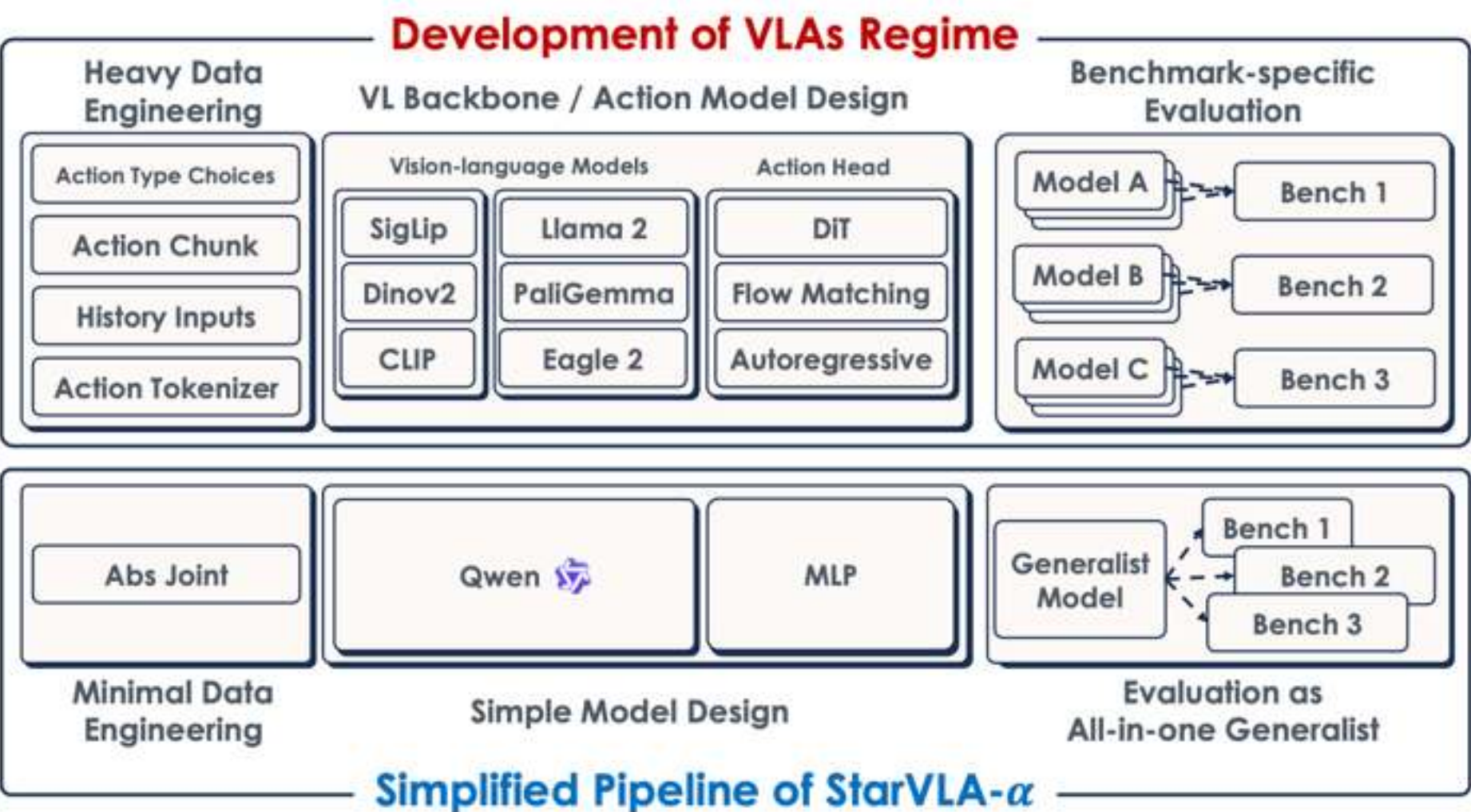
Outer Loop: AutoResearch



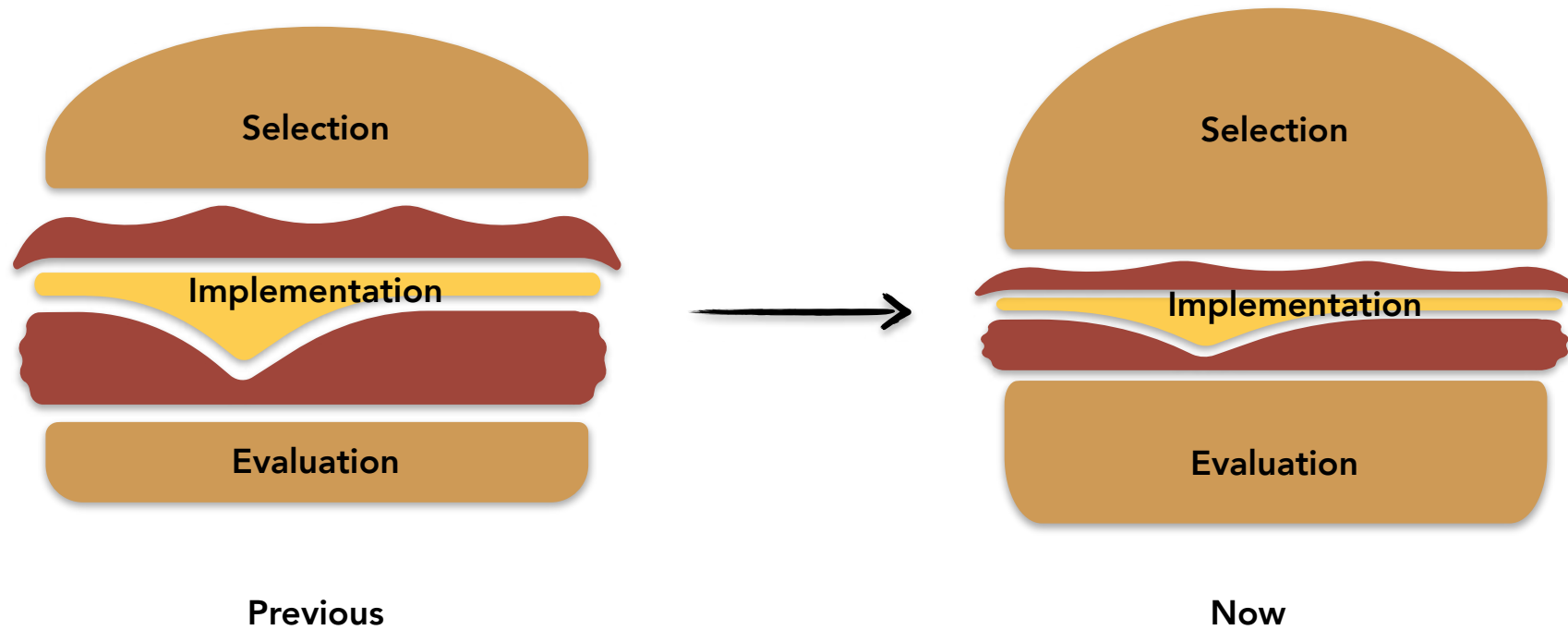
StarVLA: AutoResearch for VLAs

T*: Re-thinking Temporal Search for Long-Form Video Understanding
previous summer intern at SVL
Jinhui Ye^{1,2}, Zihan Wang^{2*}, Haoyen Sun², Keshigeyan Chandrasegaran¹,
Zare Durrani², Cristóbal Eyzaguirre¹, Yonatan Bisk², Juan Carlos Niebles¹, Ehsan Adeli¹,
Li Fei-Fei¹, Jiajun Wu¹, Manling Li^{1,2}

☆ Star 3.6k

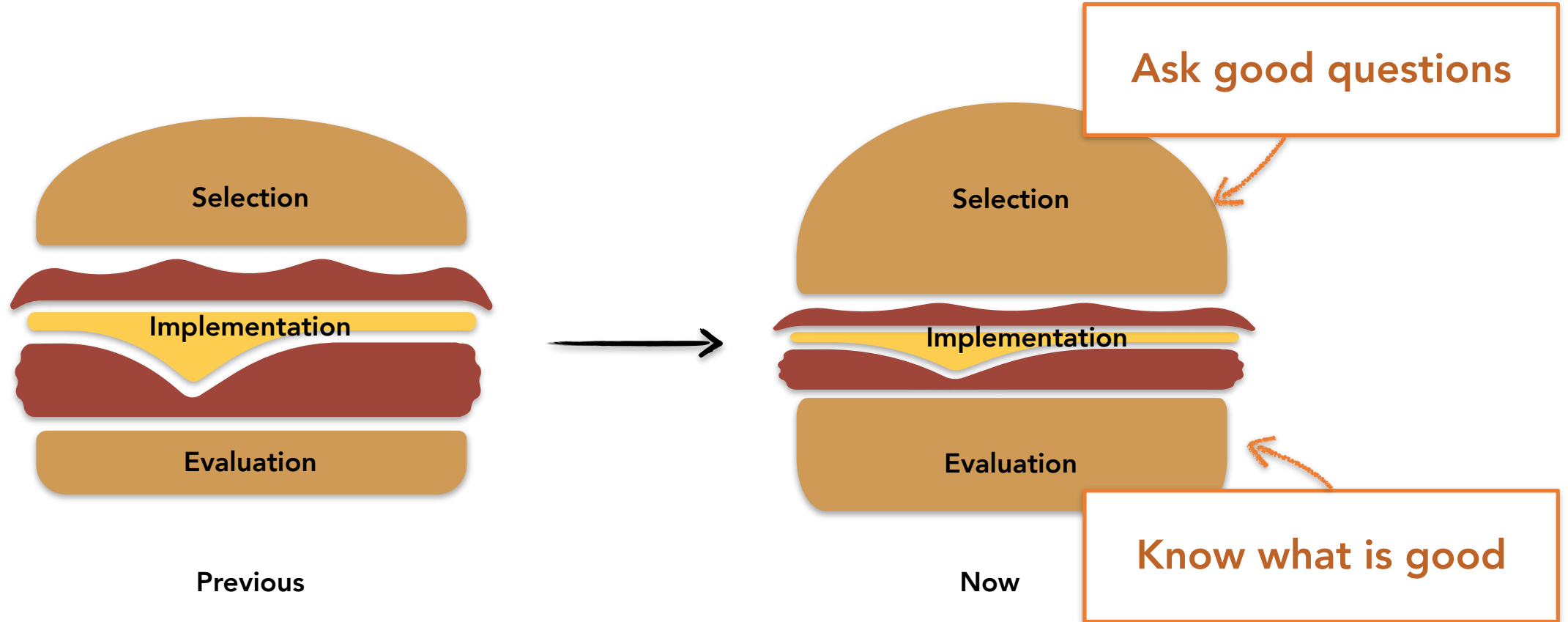


How to work with AutoResearch



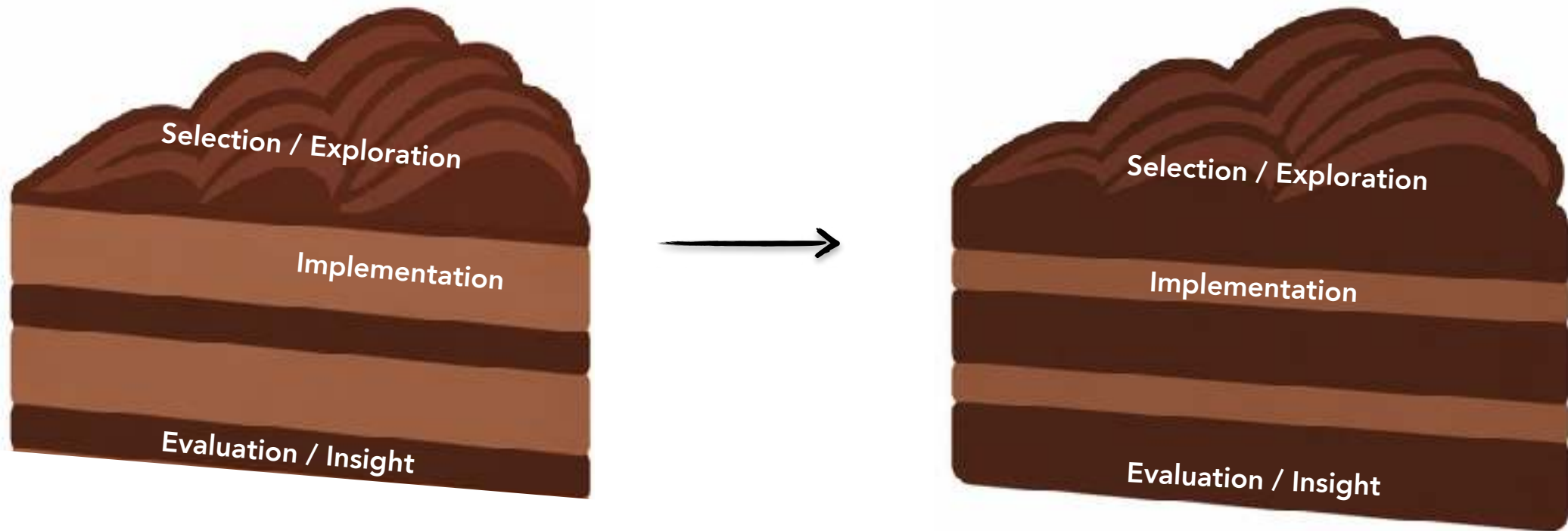
From Prof Chenhao Tan

How to work with AutoResearch

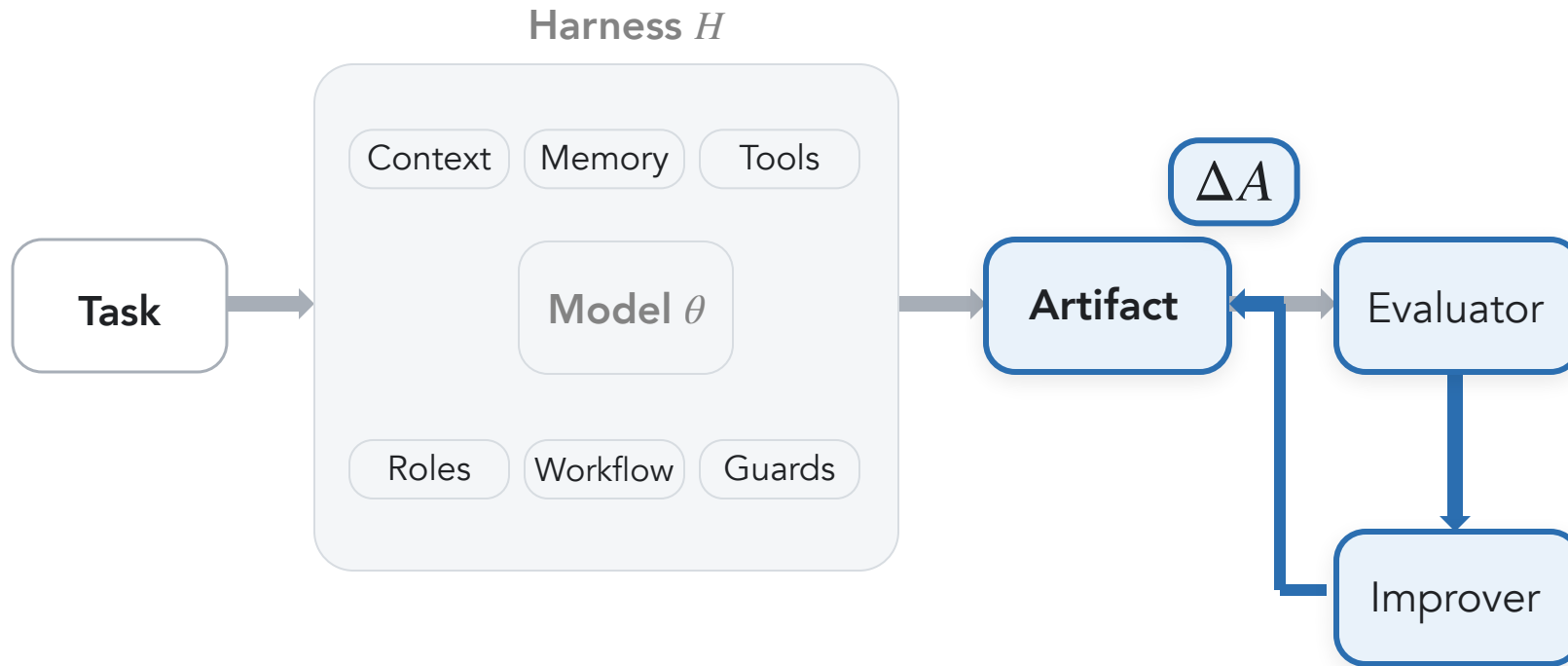


From Prof Chenhao Tan

How to work with AutoResearch



Layer 1: Artifact RSI



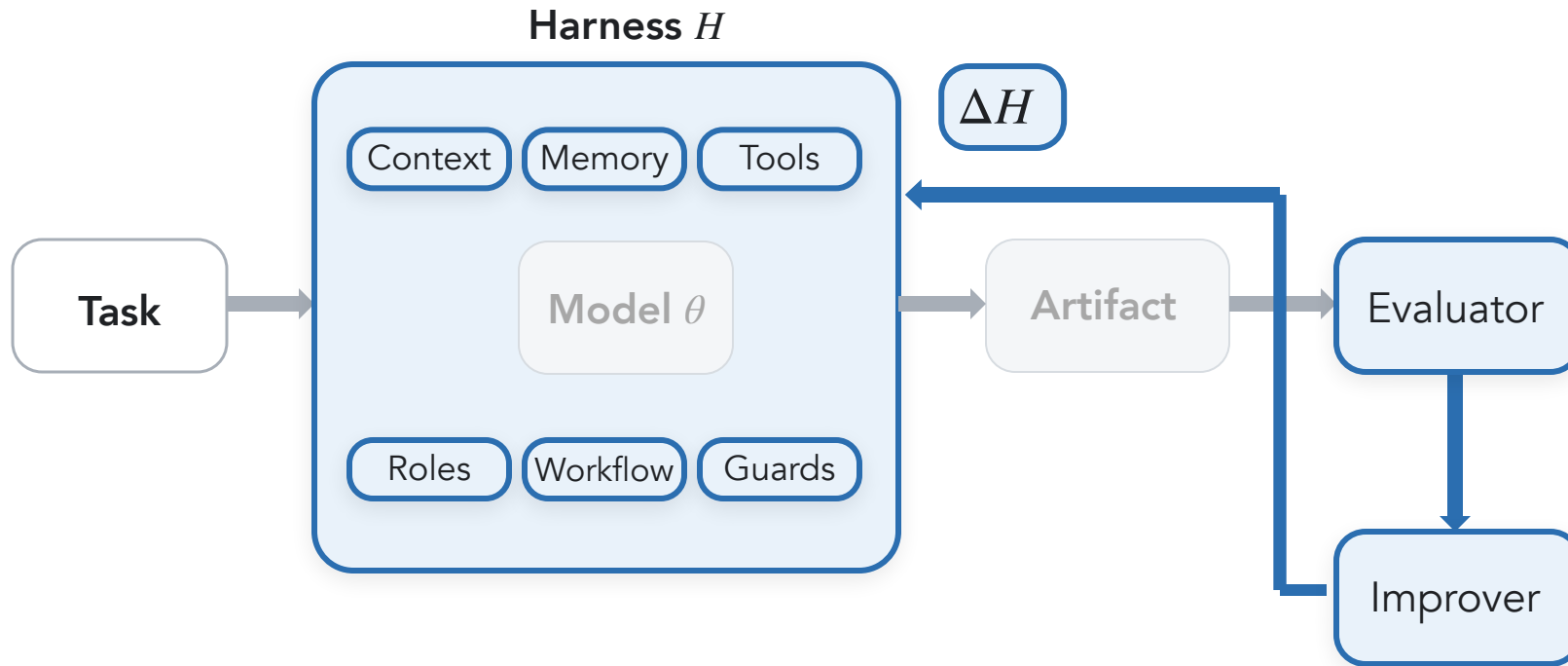
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Layer 2: Harness RSI



Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Artifact RSI → Harness RSI



ENPIRE

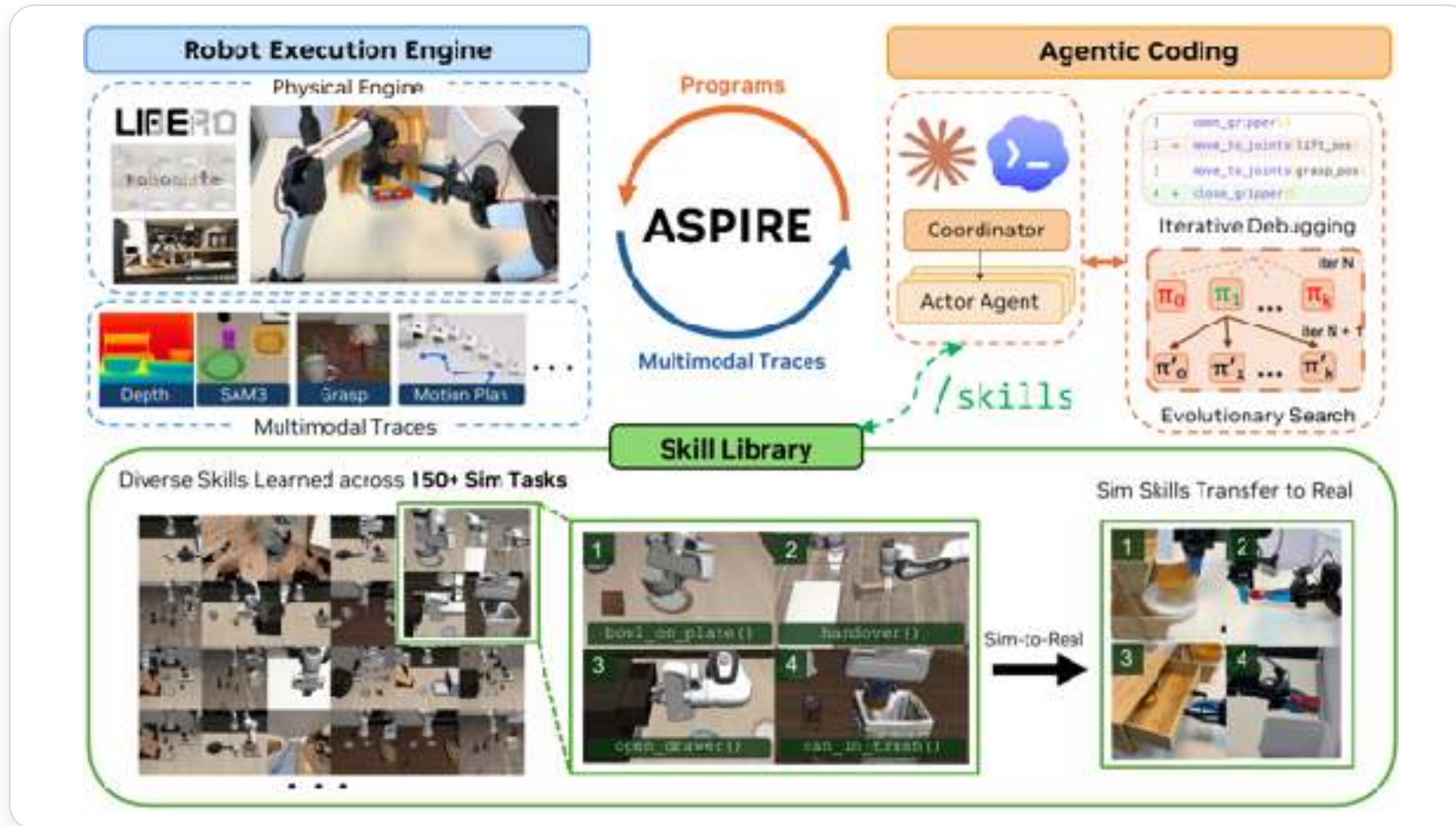
8 real robot stations |
within-task optimization



ASPIRE

150+ simulated tasks |
cross-task skill transfer

ASPIRE Turns Failures Into Skills



TRACE

Per-primitive
multimodal evidence

REPAIR

Localize failures
search and validate

MEMORY

Write to /skills
reuse across tasks

TAKEAWAY

ASPIRE improves reusable debugging knowledge, not model weights.

AutoResearch vs VLAs



ROBOCASA

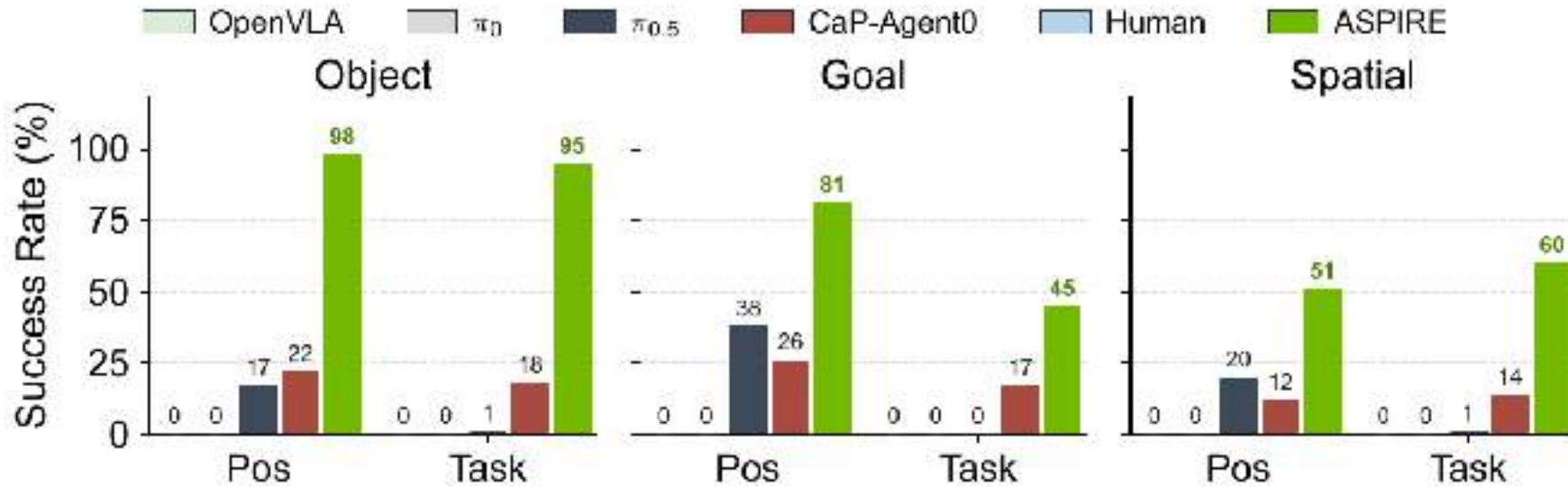
ASPIRE hybrid about 77%
GROOT N1.5 about 53%

LIBERO-PRO

ASPIRE 72%
pi0.5 13%

Takeaway:

Stronger in reasoning-heavy tasks



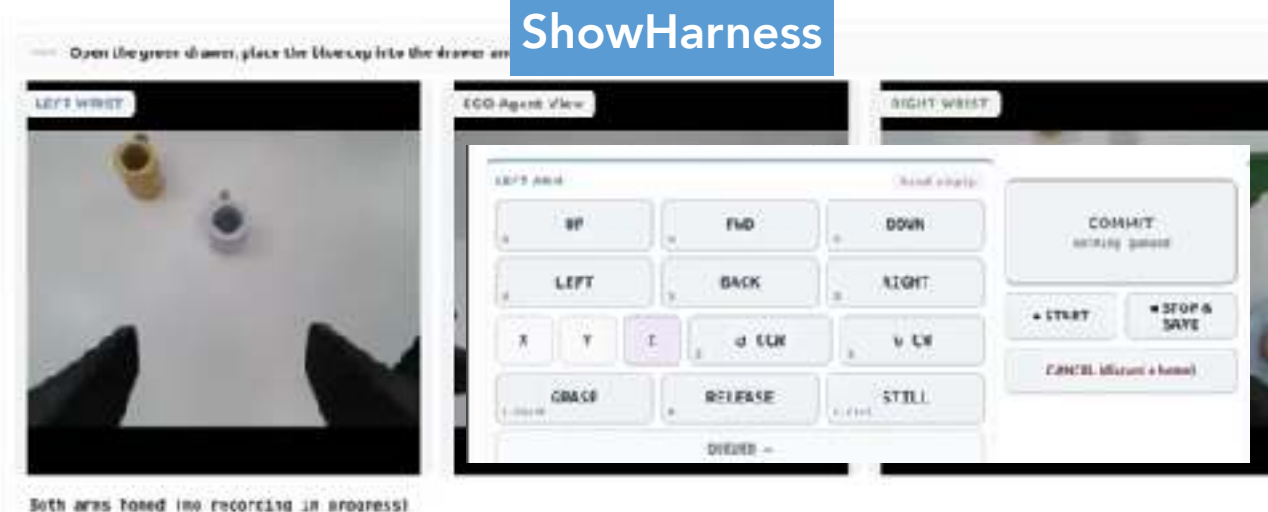
(a) **LIBERO-Pro** [Zhou et al., 2025](#): macro-averaged success over 10 tasks $\times$ 50 held-out environment seeds per suite/perturbation. For each task, ASPIRE learns and collects skills on seeds 51–65 and evaluates one generated program on seeds 1–50.

Harness: the key is Interface of Tools, Skills and Memory

HumanCLAW



ShowHarness



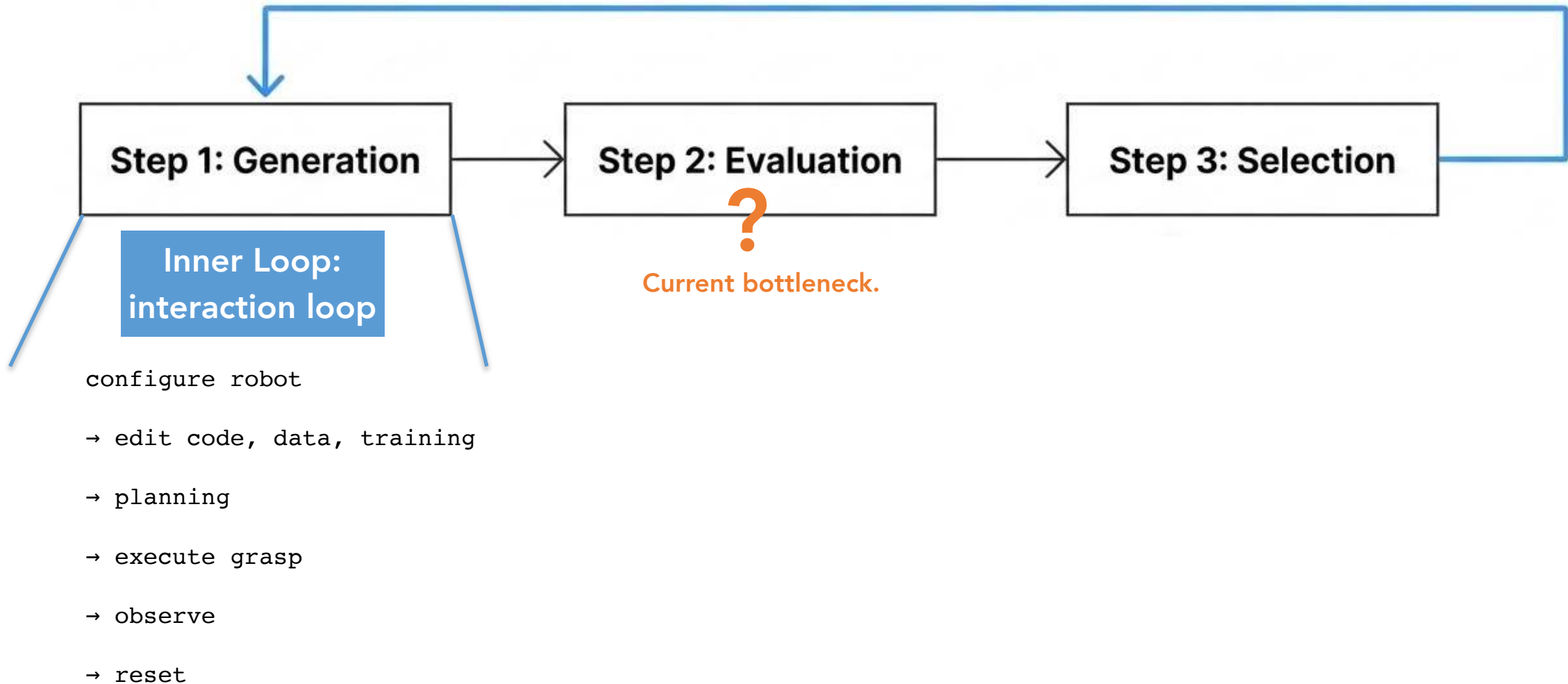
SpatialCLAW



Inner Loop + Outer Loop

Outer Loop: research loop

Step 4: The optimal code feeds back into the next generation.



Takeaways

1. The key of AutoResearch is the evaluator

2. Outer loop = research loop; Inner loop = interaction loop (with harness layer)

Takeaways: What is a better skill?

```
def pick_and_place_mug():  
    move_to(mug)  
    close_gripper()  
    lift()  
    move_to(shelf)  
    open_gripper()
```

Verification (Evaluator):

Mug is no longer on the table
Mug is on the shelf
Mug is stable
Gripper is open
No collision or unsafe motion

2026.03

Skill = Workflow

2026.09

Skill = Verifiable Contract

3. A better skill is not just an action recipe, but a contract for success.

Takeaways

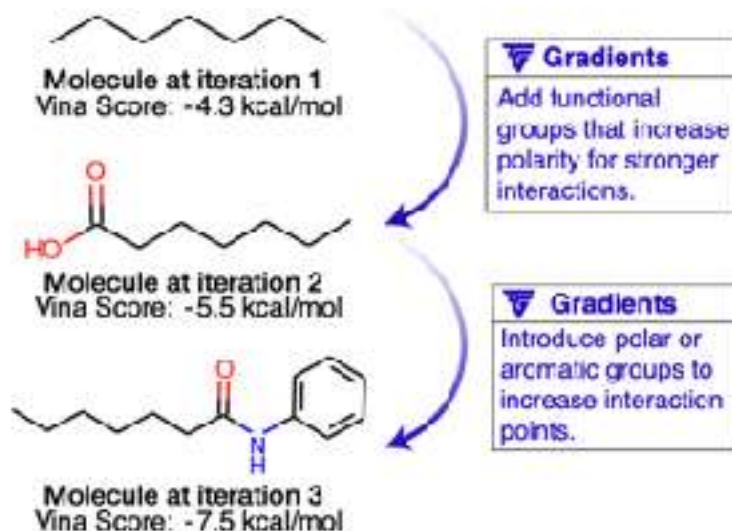
1. The key of AutoResearch is the evaluator

2. Outer loop = research loop; Inner loop = interaction loop (with harness layer)

3. Skills are more abstract when models are stronger

Takeaways: How to "train" harness?

d TextGrad for molecule optimization



e TextGrad for code optimization

```
for i in range(n):  
    if nums[i] < k:  
        balance -= 1  
    elif nums[i] > k:  
        balance += 1  
    if nums[i] == k:  
        result += count.get(balance, 0) +  
            count.get(balance - 1, 0)  
    else:  
        result += count.get(balance, 0)  
        count[balance] = count.get(balance, 0) + 1
```

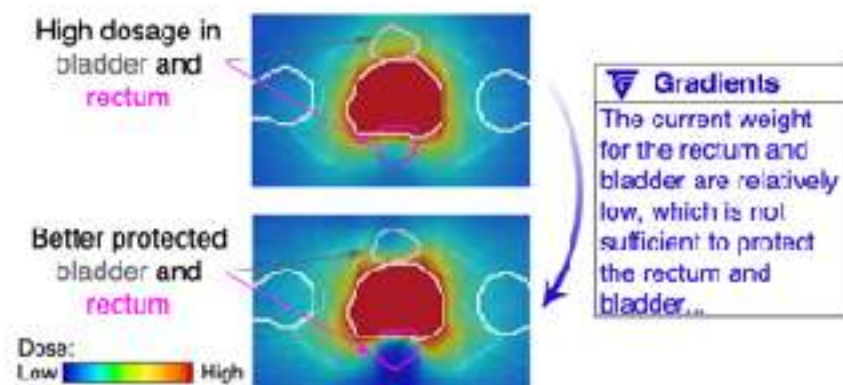
Code at iteration t

```
for i in range(n):  
    if nums[i] < k:  
        balance -= 1  
    elif nums[i] > k:  
        balance += 1  
    else:  
        found_k = True  
    if nums[i] == k:  
        result += count.get(balance, 0) +  
            count.get(balance - 1, 0)  
    else:  
        count[balance] = count.get(balance, 0) + 1
```

Code at iteration t+1

Gradients
**Handling 'nums[i] == k': The current logic does not correctly handle the case when 'nums[i] == k'. The balance should be reset or adjusted differently when 'k' is encountered. ...

f TextGrad for treatment plan optimization



g TextGrad for prompt optimization

You will answer a reasoning question. Think step by step. The last line of your response should be of the following format: 'Answer: \$VALUE' where VALUE is a numerical value.

Prompt at initialization (Accuracy = 77.8%)

You will answer a reasoning question. List each item and its quantity in a clear and consistent format, such as '- item: Quantity'. Sum the values directly from the list and provide a concise summation. Ensure the final answer is clearly indicated in the format: 'Answer: \$VALUE' where VALUE is a numerical value. Verify the relevance of each item to the context of the query and handle potential errors or ambiguities in the input. Double-check the final count to ensure accuracy.

Prompt after optimization (Accuracy = 91.9%)

Takeaways

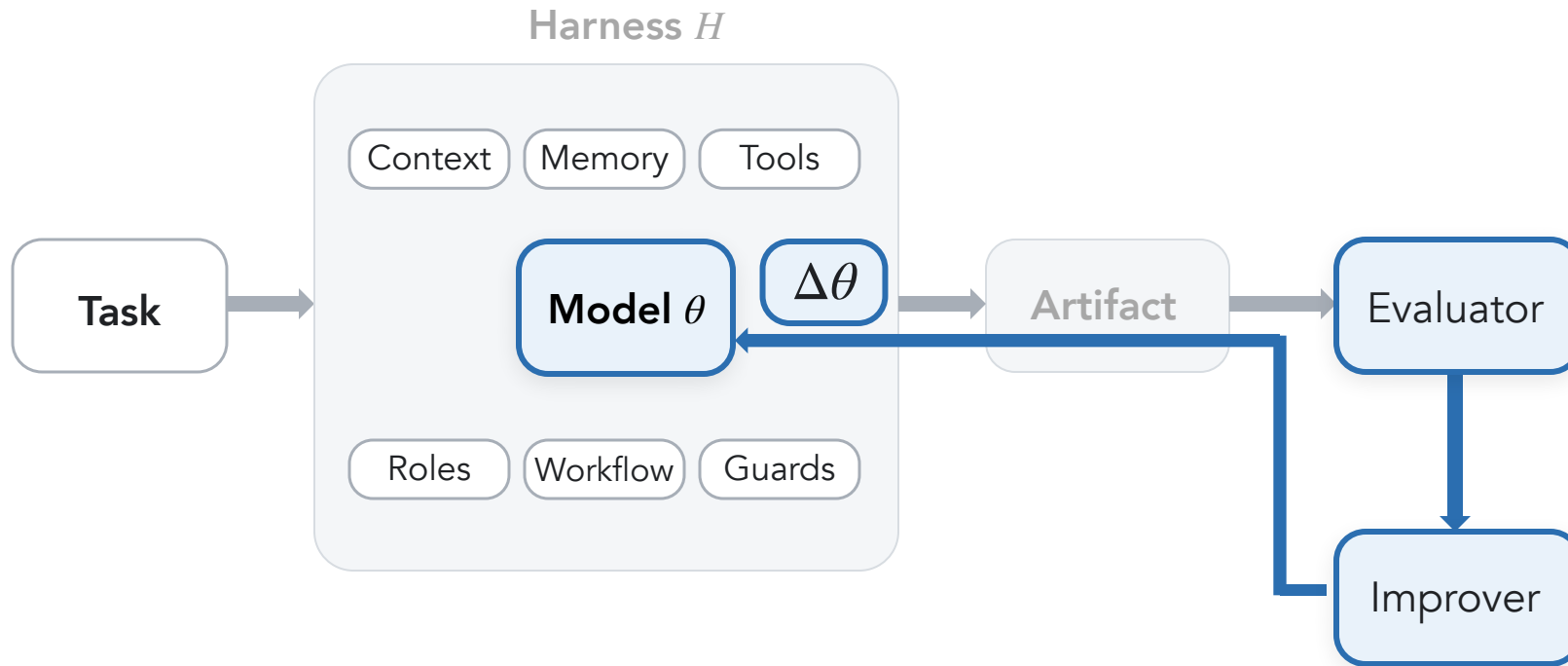
1. The key of AutoResearch is the evaluator

2. Outer loop = research loop; Inner loop = interaction loop (with harness layer)

3. Skills are more abstract when models are stronger

4. Training data: text gradient for training

Layer 3: Model RSI



Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

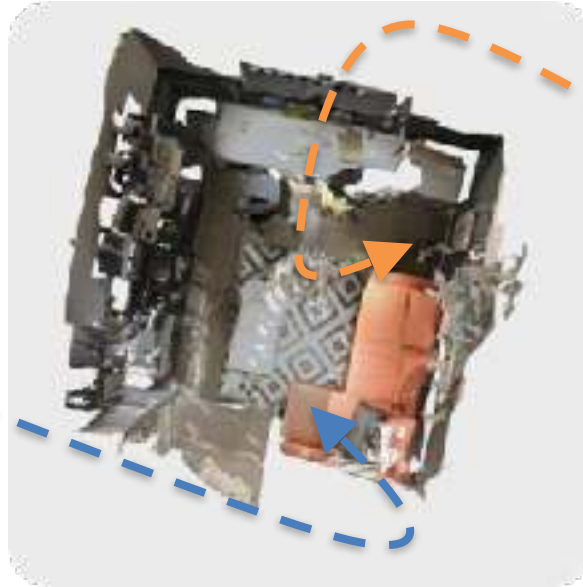
“ RL cannot work ”

Active View Planner

Initial view



Top-down View



Target view



Rotation Actions

turn_left, turn_right, look_up,
look_down, rotate_cw, rotate_ccw

Translation Actions

move_forward, move_backward, move_right,
move_left, move_up, move_down

Answer(tx, ty, tz, rx, ry, rz)

Pure RL Barely Moves

2.5%

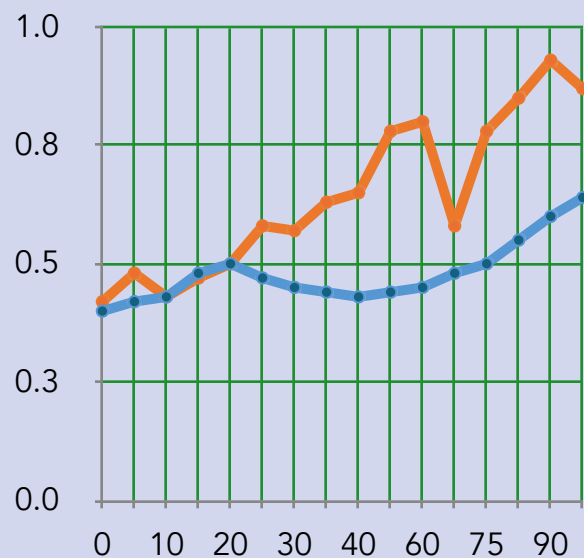
PPO	3.2%
GRPO	5.2%
Bootstrap	6.2%

Sparse rewards stall every RL variant,
we need supervision the agent can build itself.

"familiar" env



Webshop

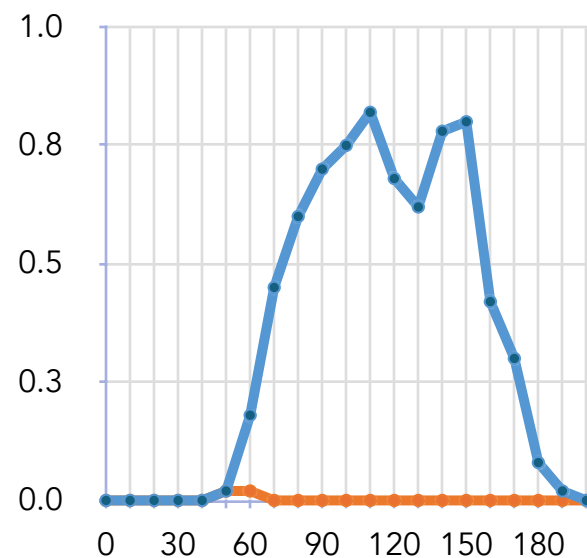


steps

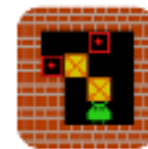
"unfamiliar" env collapse



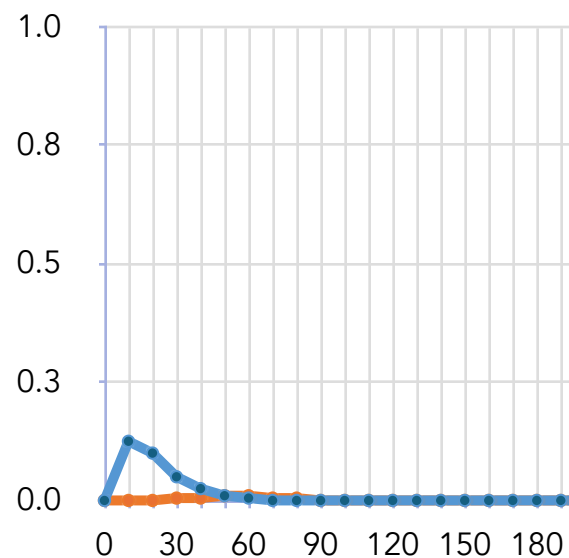
Bandit



steps



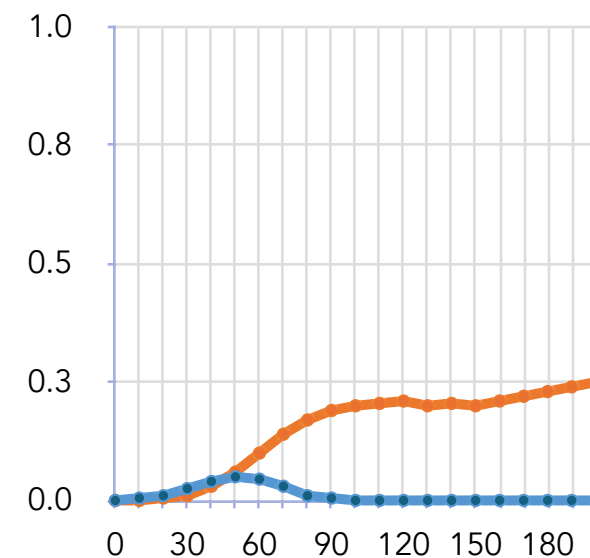
Sokoban



steps



FrozenLake



steps

— PPO — GRPO



Reasoning Collapse into Repetition

0th round

```
#####  
#_O_#  
#_X_#  
#_P_#  
#####
```

Reasoning: *The box is above me, and the target is on the far left. I cannot push it directly. I should first move right, then approach the box from the right.*

Action:
up



```
#####  
#_O_#  
#_XP#  
#_#_#  
#####
```

Reasoning: *Now I can push the box to the left, until the bottom of the target.*

Action:
right



```
#####  
#_O_#  
#_XP_#  
#_#_#  
#####
```

Reasoning: *The box is now at the bottom of the target. I am at the right of the box. To push the box to the target, I need to move to the bottom of the box, and then push it up.*

Action:
right



```
#####  
#_√_#  
#_P_#  
#_#_#  
#####
```

Reasoning: *(box is now at the target)*



Reasoning Collapse into Repetition

#_O_#
#_X_#
#_P_#
#####

Reasoning: *The box is above me, and the target is on the far left. I cannot push it directly. I should first move right, then approach the box from the right.*

Action:
up



200 rounds later...

#_O_#
#_XP_#
#_#_#
#####

bottom of the target. I am at the right of the box. To push the box to the target, I need to move to the bottom of the box, and then push it up.

Action:
right



#_√_#
#_P_#
#_#_#
#####

Reasoning: *(box is now at the target)*



Reasoning Collapse into Repetition

200th round

#_O_#
#_X_#
#_P_#
#####

Reasoning: *The box is above me. I should move up to the box.*

Action:
right, up



#_O_X_#
#_P_#
#_#_#
#####

Reasoning: *I should move to the right of the box.*

Action:
left, left



#_O_X_#
#_P_#
#_#_#
#####

Reasoning: *I should move to the right of the box.*

Action:
down, left, up



#_O_X_#
#_P_#
#_#_#
#####

Reasoning: *I should move to the right of the box.*



Exploitation vs Exploration

Repeat one solution



Exploitation vs Exploration

Alternative solutions

Especially in **OOD** environments



self-improvement through RL

1. What are *failure modes* of RL?

2. What priors should we encourage for exploration?

3. When is the best stage to encourage it?



self-improvement through RL

1. What are failure modes of RL?

→ *Exploration* vs *Exploitation*

2. What priors should we encourage for exploration?

3. When is the best stage to encourage it?



self-improvement through RL

1. *What are failure modes of RL?*

→ *Exploration vs Exploitation*

2. *What priors should we encourage for exploration?*

→ *World Model Reasoning*

3. *When is the best stage to encourage it?*

Priors in Exploration:

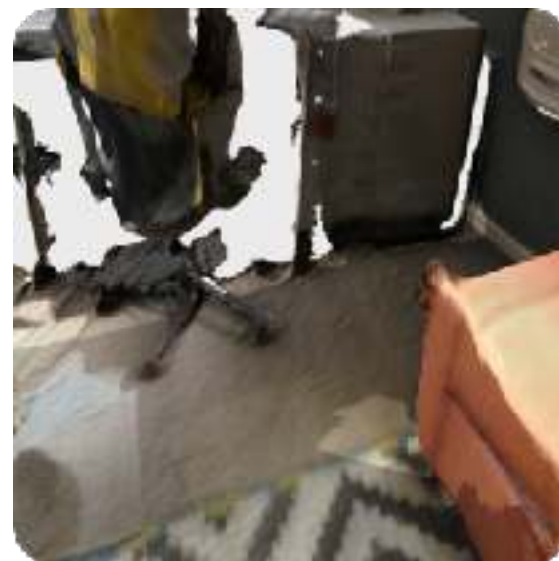
World Modeling

View Transitions as a View World Model



view A

turn left | move forward



view B

self-improvement through RL

1. *What are failure modes of RL?*

→ *Exploration vs Exploitation*

2. *What priors should we encourage for exploration?*

→ *World Model Reasoning*

3. *When is the best stage to encourage it?*

→ *Mid-training/SFT vs RL*

PRIORS

Common
Knowledge

STAGES

Pre-training

World
Modeling

Mid-training

Instruction
Following

Supervised
Finetuning

Reasoning
Planning

Reinforcement
Learning

DATA

Static Internet Data:
Books, Images, Videos...

Play Data:
Env, Tools, External Data

Task Annotations:
<instructions, answers>

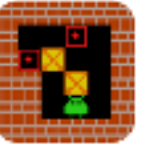
Task Feedback:
Verifiers (Goal Definitions)

Brute-force Scaling

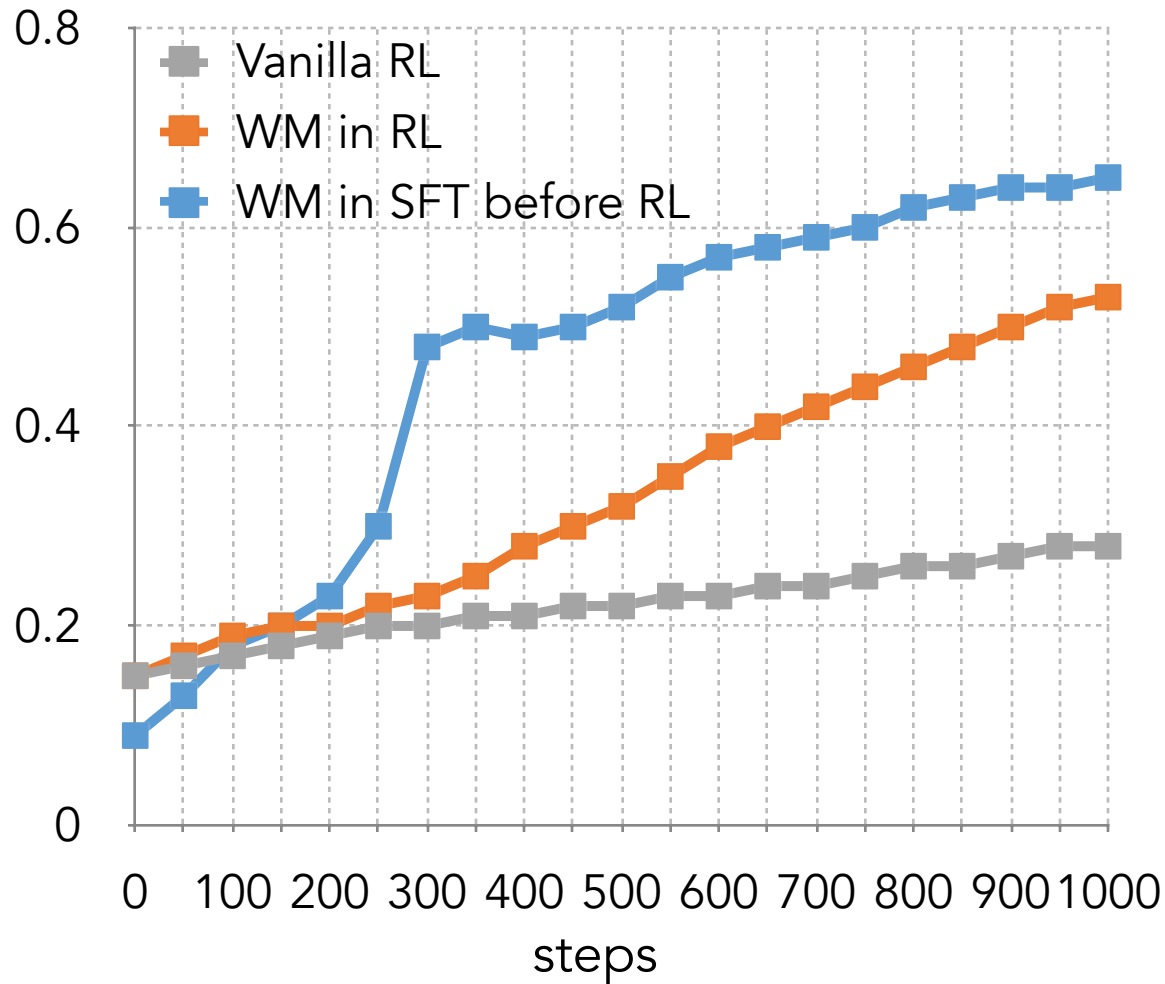


Continual Learning
(We are here now)

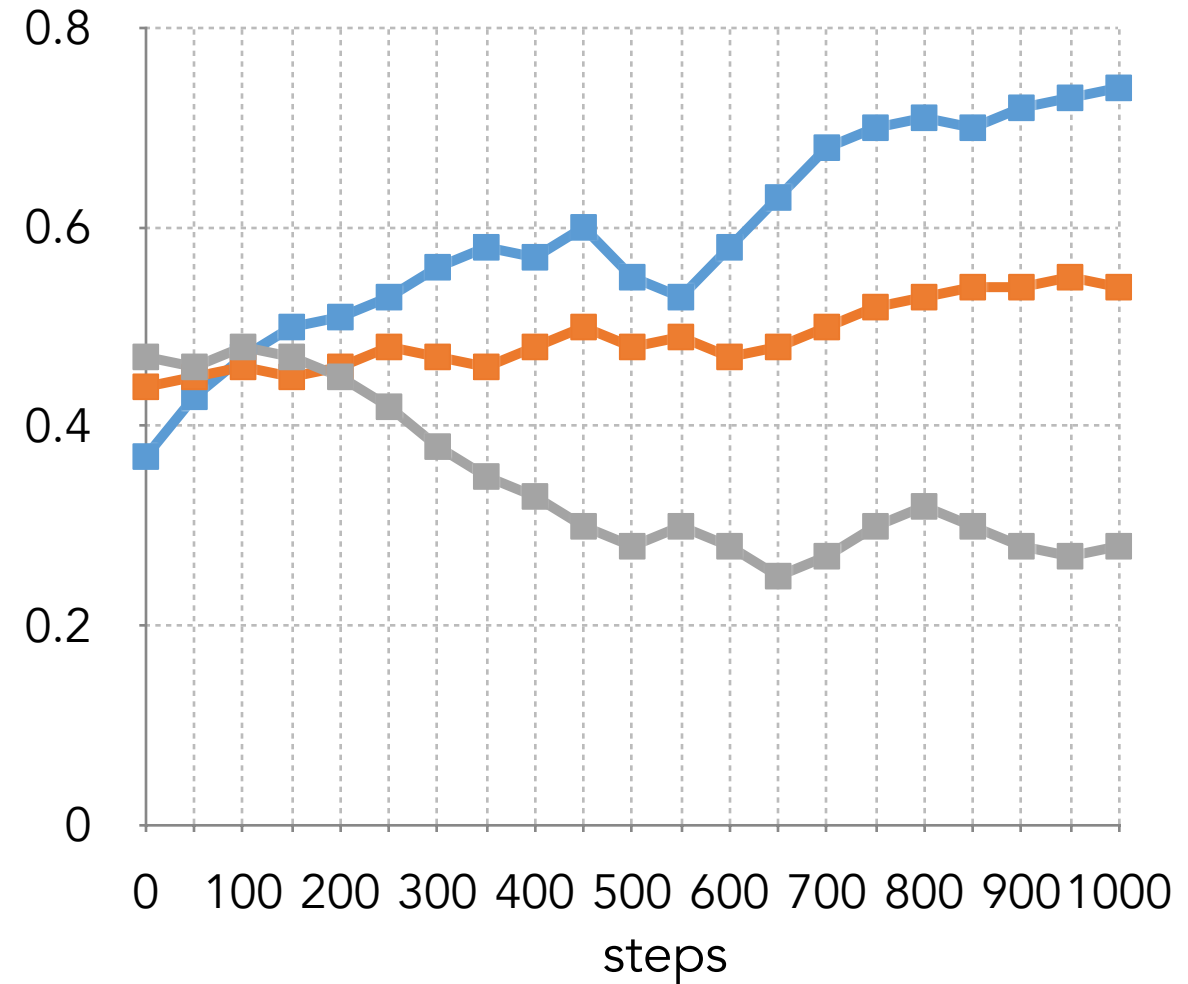
Exploration vs Exploitation



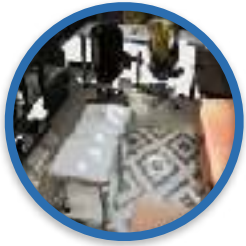
Pass@1



Pass@K

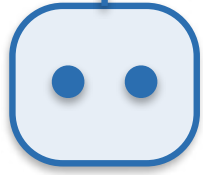
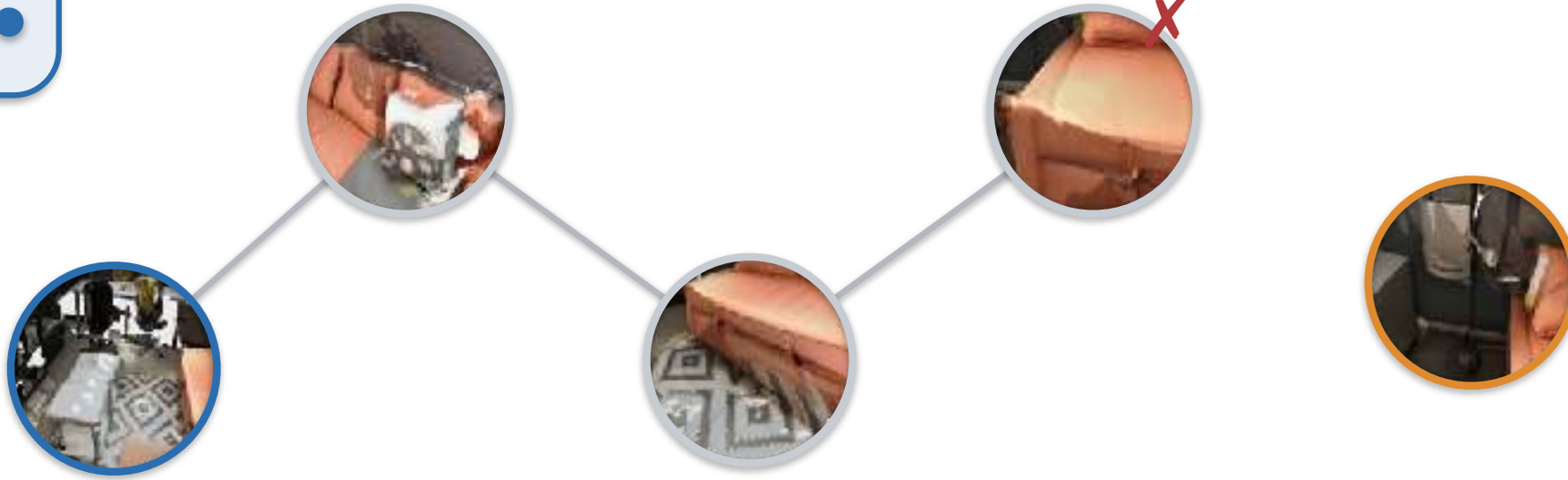


Step 1 • View Graph via On-Policy Exploration



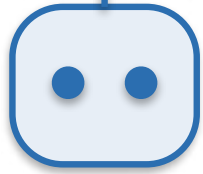
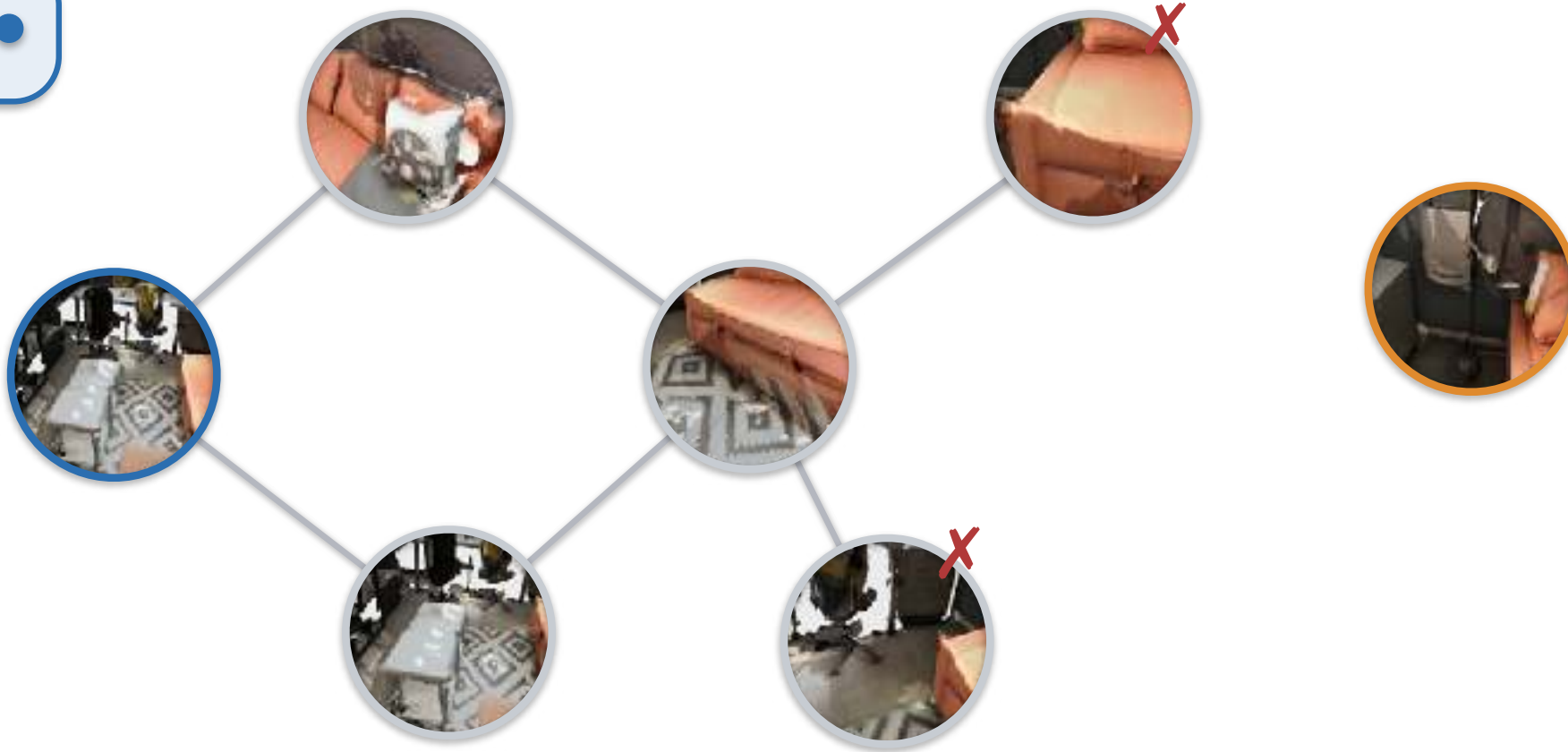
The agent explores **ViewSuite** scenes from its own experience.

Step 1 • View Graph via On-Policy Exploration



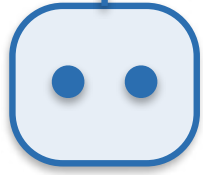
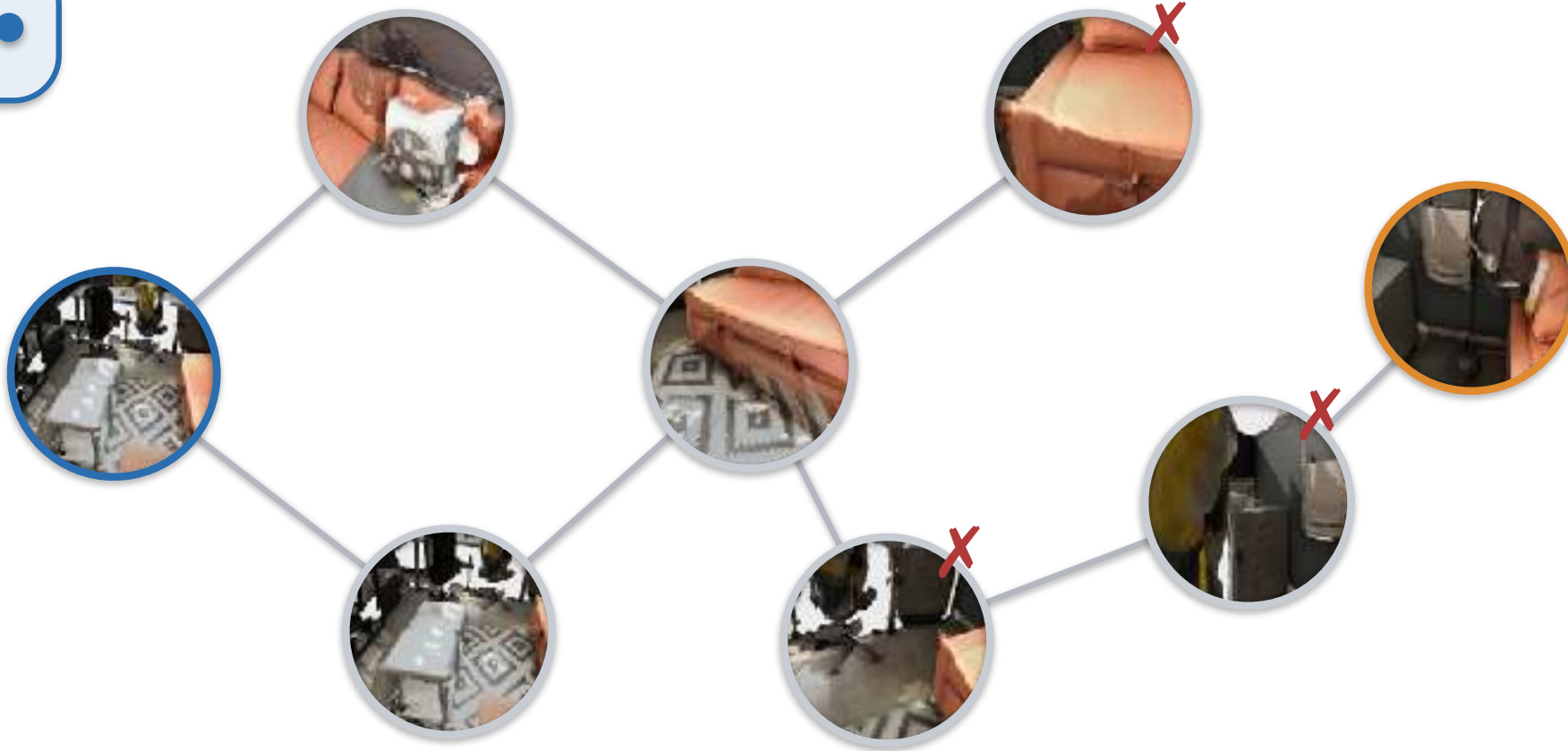
Rollout 1 wanders the scene – and misses.

Step 1 • View Graph via On-Policy Exploration



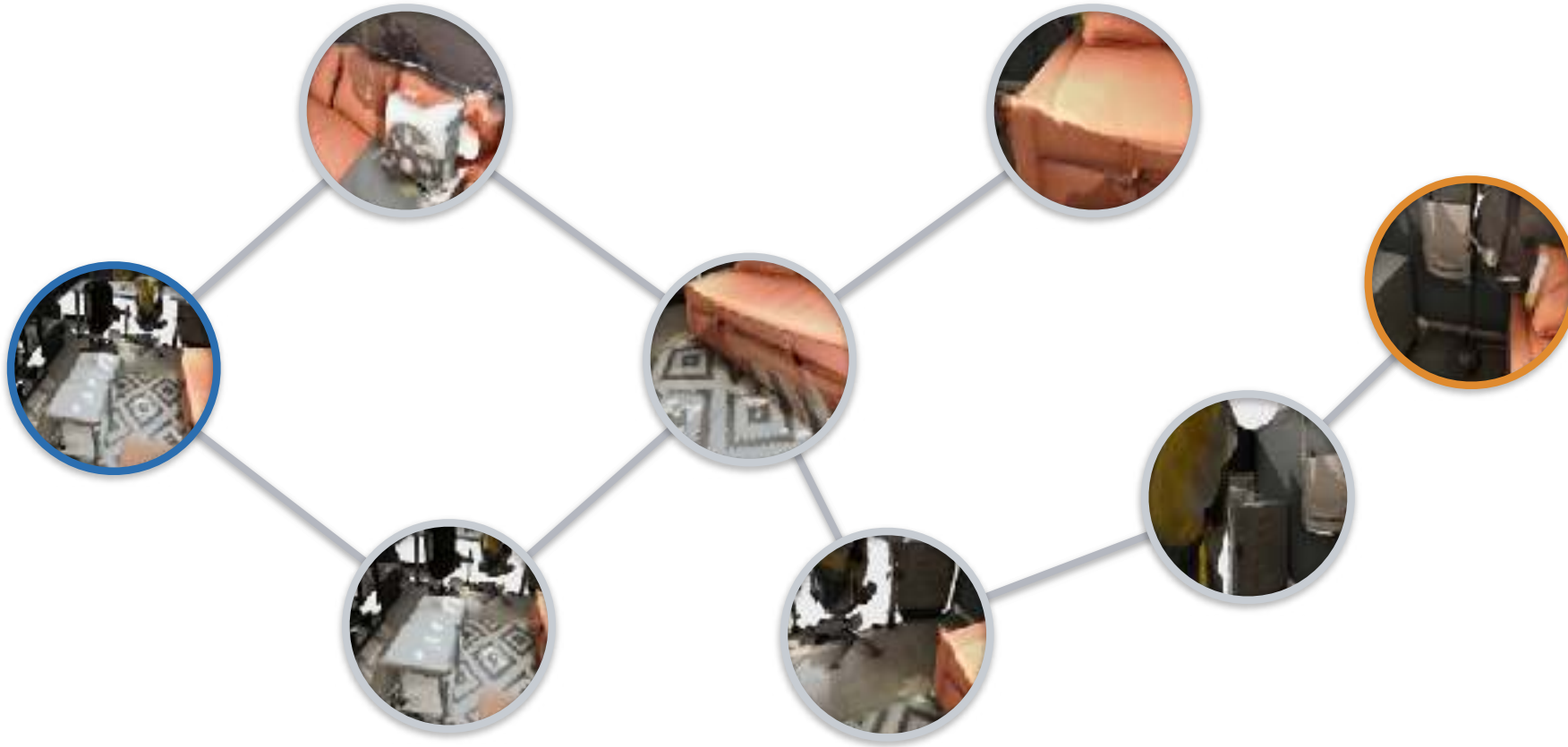
Rollout 2 also fails – but it crosses rollout 1.

Step 1 • View Graph via On-Policy Exploration



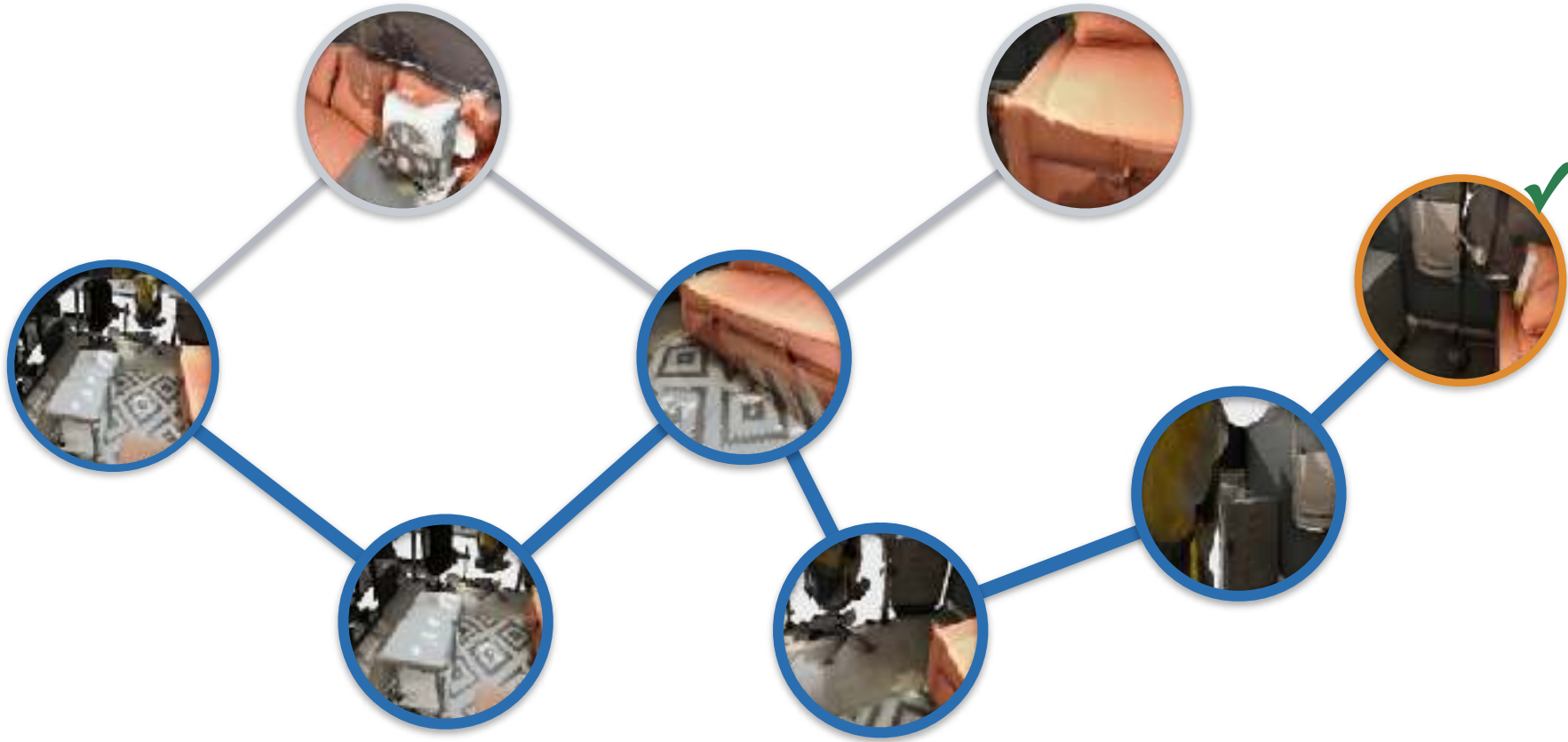
Every rollout misses the target. Yet together they trace the whole scene.

Step 2 • A View Graph Emerges



Shared viewpoints **merge**: nodes are views, edges are actions.

Step 2 • A Path Emerges

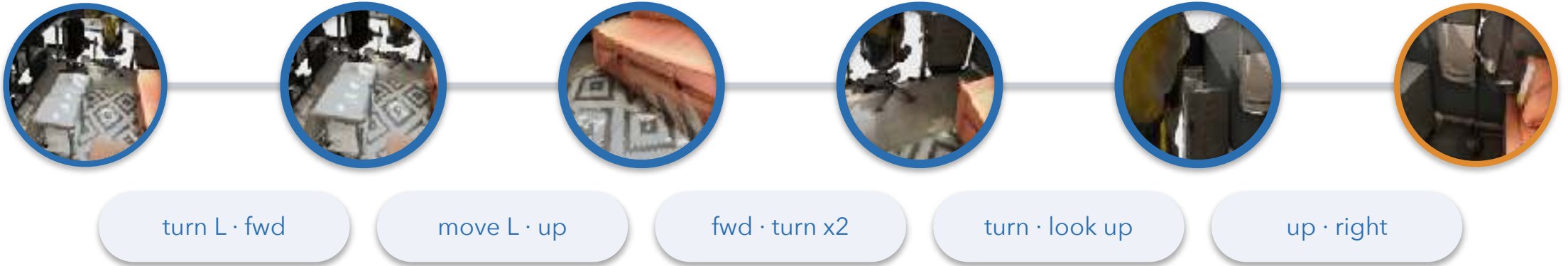


A path the agent **never walked** now connects initial → target.

Step 3 • Reformulate into a Demo

initial view

target view



input: initial + target view output: the action sequence

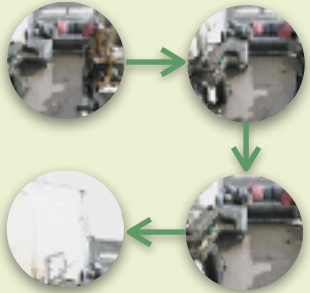
Any sampled path becomes a demo
valid even though no rollout reached the target.

View Planning via Self-Exploration

RL Exploration



PPO with Interactive View
Planning Reward

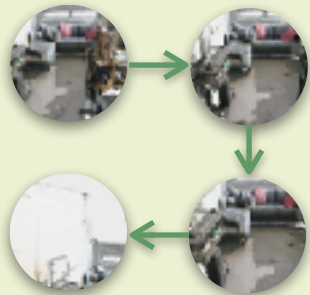


View Planning via Self-Exploration

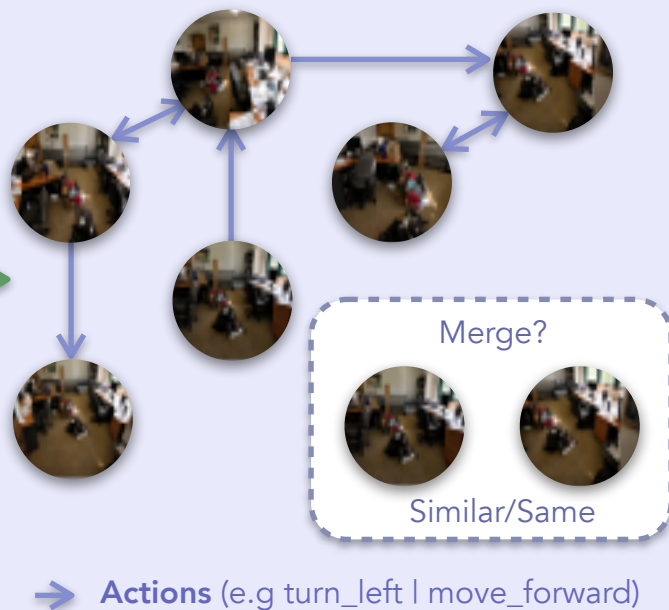
RL Exploration



PPO with Interactive View Planning Reward



On-Policy View Graph Construction

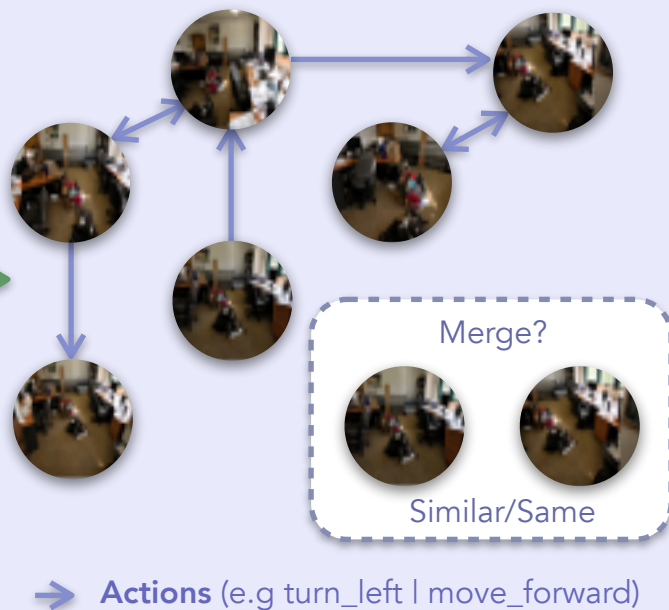


View Planning via Self-Exploration

RL Exploration

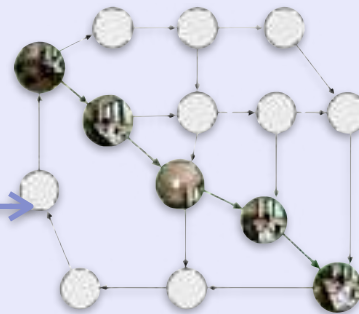


On-Policy View Graph Construction



View Graph Distillation

Path Sampling: length=5



Start Node: Initial View
End Node: Target View
Intermediate Node: Turn-by-turn observations

Reformulated Training Example

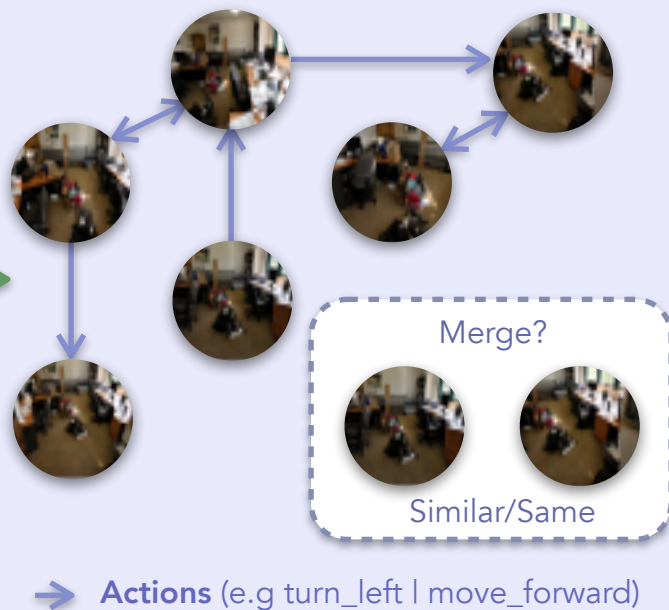


View Planning via Self-Exploration

RL Exploration

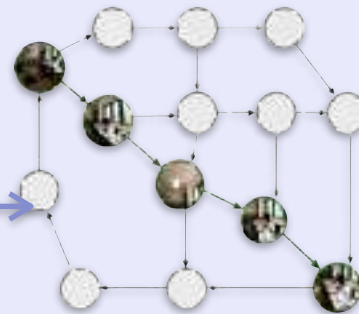


On-Policy View Graph Construction



View Graph Distillation

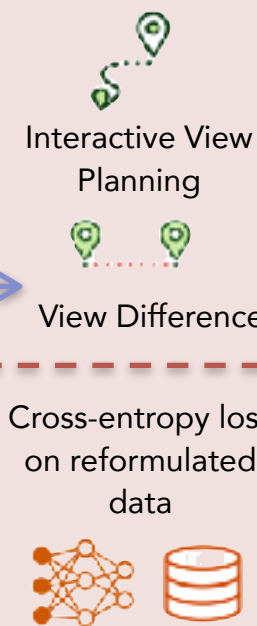
Path Sampling: length=5



Reformulated Training Example



SFT Training

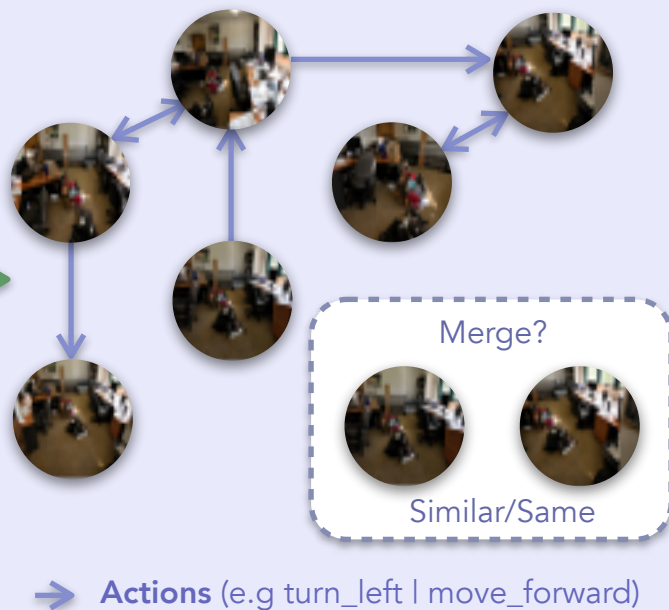


View Planning via Self-Exploration

RL Exploration

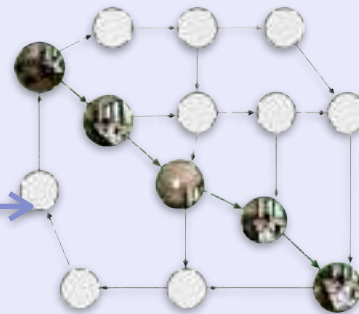


On-Policy View Graph Construction



View Graph Distillation

Path Sampling: length=5



Start Node: Initial View
End Node: Target View
Intermediate Node: Turn-by-turn observations

Reformulated Training Example



SFT Training



Cross-entropy loss on reformulated data



Next Iteration

From 2.5% to 47.8%

Qwen2.5-VL-7B (base)

2.5%

GPT-5.4 Pro

18.5%

Gemini 3.1 Pro

21.4%

Ours

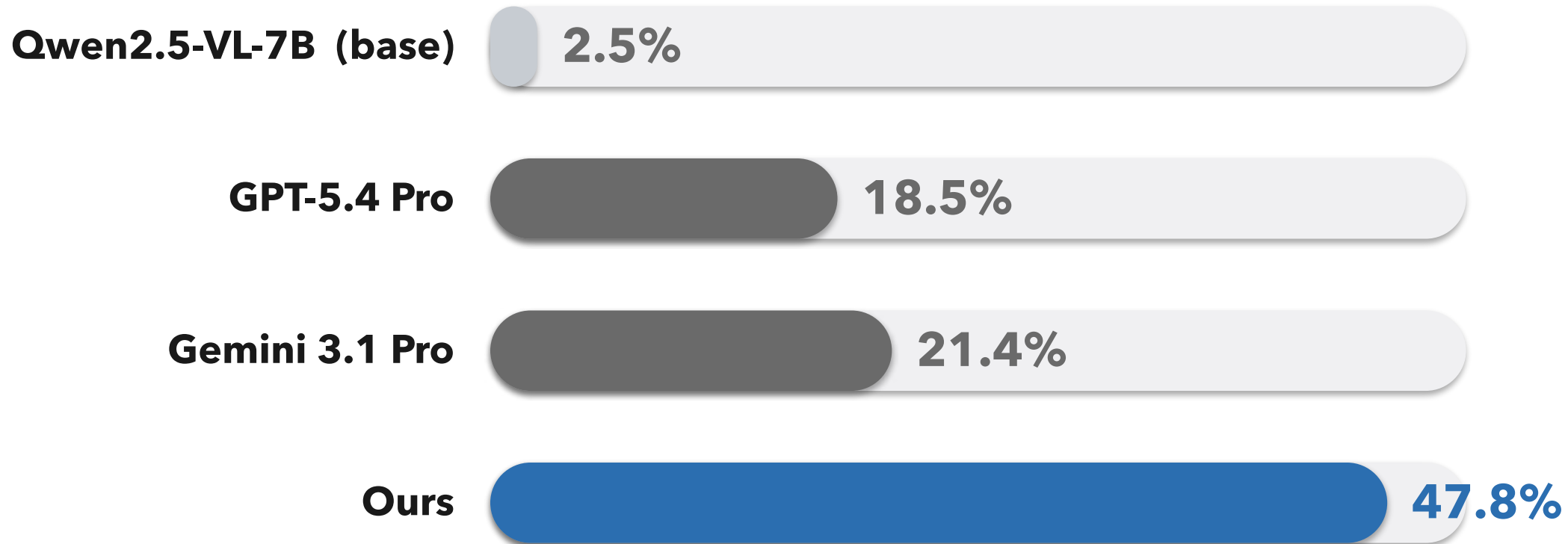
47.8%

[View Planning \(arxiv 2026\)](#)

From **2.5%** to **47.8%**

Qwen2.5-VL-7B (base)

2.5%



Model	Performance (%)
Qwen2.5-VL-7B (base)	2.5%
GPT-5.4 Pro	18.5%
Gemini 3.1 Pro	21.4%
Ours	47.8%

GPT-5.4 Pro

18.5%

Gemini 3.1 Pro

21.4%

Ours

47.8%

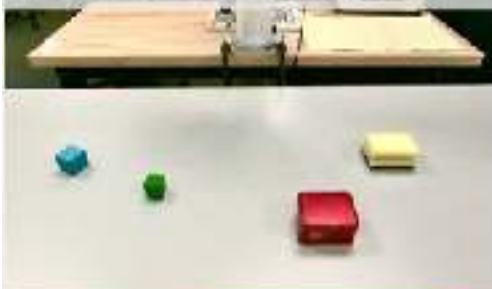
A 7B model trained only on its own exploration **beats every frontier VLM.**

[View Planning \(arxiv 2026\)](#)

Embodied Prior Learning via mid-training



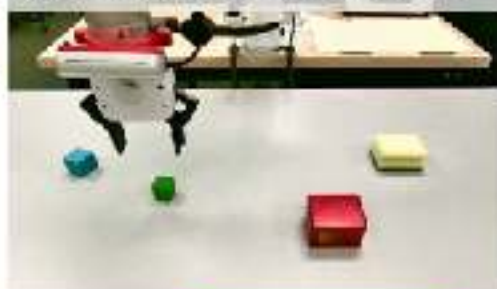
① Move above the green cube



Reasoning: I can see a blue cube at [14, 18, 8], a green cube at [26, 45, 7], a red cube at [41, 90, 11] and a yellow sponge at [12, 115, 10]. I need to place the lime cube on top of the red cube. So, the first step is to Move the gripper above the lime cube at [26, 45, 7].

Action: [26, 45, 26, 0, 60, 90, 1]

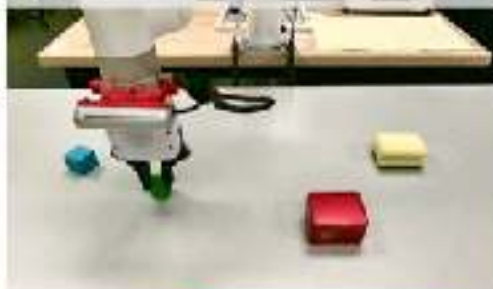
② Lower the gripper and grasp



Reasoning: I can see a blue cube at [14, 18, 8], a green cube at [26, 45, 7], a red cube at [41, 90, 11], and a yellow sponge at [12, 115, 10]. The next step is to lower the gripper to grasp the lime cube. After that, I will close the gripper to pick up the lime cube, lift it, and then...

Action: [26, 45, 7, 0, 60, 90, 0]

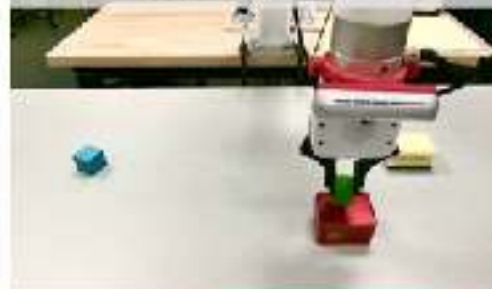
③ Move above the red container



Reasoning: I can see a green cube at [26, 45, 7], and a red cube at [41, 90, 11]. Now, I have successfully picked up the lime cube from its initial position. The next step is to move the lime cube to a position above the red cube.

Action: [41, 90, 25, 0, 60, 90, 0]

④ Release the green cube



Reasoning: I can see a green cube at [14, 18, 8], and a red cube at [41, 90, 11]. I have successfully moved the lime cube above the red cube. The next step is to lower the lime cube onto the red cube and release it by opening the gripper. This will complete the task.

Action: [41, 90, 17, 0, 60, 90, 1]

⑤ Finish the task and stop exec.



Reasoning:
The task has been completed.

Action: None

self-improvement through RL

1. *What are failure modes of RL?*

→ *Exploration vs Exploitation*

2. *What priors should we encourage for exploration?*

→ *World Model Reasoning*

3. *When is the best stage to encourage it?*

→ *Mid-training/SFT vs RL*

Learning reasoning through RL

1. *What are failure modes of RL?*

data problem vs RL problem?

3. *When is the best stage to encourage it?*

→ *Mid-training vs RL*

PRIORS

Common
Knowledge

STAGES

Pre-training

World
Modeling

Mid-training

Instruction
Following

Supervised
Finetuning

Reasoning
Planning

Reinforcement
Learning

DATA

Static Internet Data:
Books, Images, Videos...

Play Data:
Env, Tools, External Data

Task Annotations:
<instructions, answers>

Task Feedback:
Verifiers (Goal Definitions)

Brute-force Scaling



Continual Learning
(We are here now)

What makes a world model reasoning success?

1. transition modeling

Abstraction

2. state estimation

Abstraction

3. on-policy exploration

Prior

4. enough world modeling training

Prior

Widely used by others

Fork 209 Starred 2.5k

Awesome Work Powered or Inspired by RAGEN

- [ROLL](#): Efficient Scaling Library for RL with LLMs  Stars 2k
- [VAGEN](#): Training Visual Agents with multi-turn RL  Stars 431
- [Search-R1](#): Train LLMs to reason and call a search engine with RL  Stars 4.2k
- [ZeroSearch](#): Incentivize LLM search capability without searching  Stars 1.3k
- [Agent-R1](#): Training Powerful LLM Agents with End-to-End RL
- [OpenManus-R1](#): RL tuning for LLM agents  Stars 39k
- [MetaSpatial](#): Reinforcing 3D Spatial Reasoning in VLMs  Stars 200
- [e3](#): Efficient Yet Effective Search Agent Training via RL

REASONING GYM: Reasoning Environments for Reinforcement Learning with Verifiable Rewards

Zafir Stojanovski^{1*} Oliver Stanley^{1,2*} Joe Sharratt^{1*} Richard Jones^{1,2*}

Abdulkareem Adeboye² Jean Kaddour^{2†} Andreas Köpf²

SPIRAL: SELF-PLAY ON ZERO-SUM GAMES INCENTIVIZES REASONING VIA MULTI-AGENT MULTI-TURN REINFORCEMENT LEARNING

Bo Liu¹, Simon Yue^{2*}, Zichen Liu^{1,2*}, Leon Guenther¹
Pengfei Qi^{1,2}, Hamid Hazzeth², Mickael Luth², Chester Tan¹, Weyan Shi², Min Lin², Wei Sun Lee³, Narasimha Joshi^{2*}

THE DEVIL BEHIND MOLTBOOK: ANTHROPIC SAFETY IS ALWAYS VANISHING IN SELF-EVOLVING AI SOCIETIES

Chenru Wang¹, Chaochao Li^{1*}, Songyang Liu¹, Zejian Chen¹, Jinyu Hou¹, Ji Qi¹, Hui Li¹, Lifan Zhang¹, Qiwei Ye^{2*}, Zheng Lin², Xu Chen², Xi Zhang¹, Philip S. Yu¹

WebThinker: Empowering Large Reasoning Models with Deep Research Capability

Xiaoli Li¹, Jinjie Jin^{1*}, Guanting Dong^{1*}, Honglin Qian², Yongkang Wu³
Ji-Rong Wen¹, Yutao Zhu¹, Zhiheng Dong^{1†}

Search-R1

Our pipeline is based on verl (<https://github.com/volcengine/verl>), a highly efficient RL framework. We would also like to thank HryZero (<https://github.com/HryZero/Tiny...>) and RAGEN (<https://github.com/ZihanWang314/R...>) which inspired this work.

12:02 PM · Feb 26, 2025 · 26,884 Views

Tarishq Mathew Abraham, Ph.D. @Sciencelaw · Dec 9, 2024
A new tutorial on RL by Kevin Patrick Murphy, a Research Scientist at Google DeepMind who also wrote several comprehensive, well-regarded textbooks on ML/DL.

This ought to be a good read !!

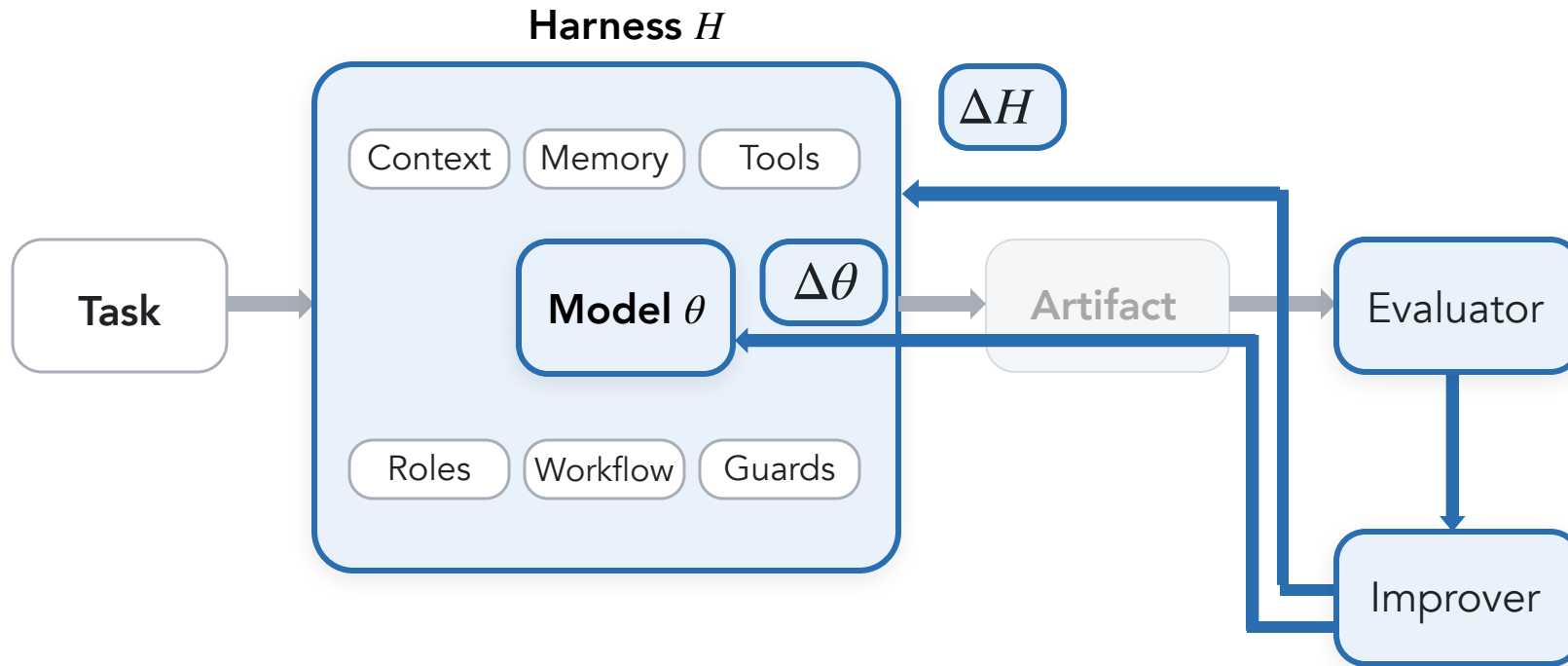
6 x 5	Algorithm	Example	Stars
6.2.5.1	Example: RAGEN		137
6.2.5.2	Decline with invalid actions		158
6.2.5.3	Time-step learning		158
6.2.5.4	Self-play for LLM training		139

VentureBeat @VentureBeat · Apr 23, 2025
Former DeepMind researchers released new method for training reliable AI agents: RAGEN



Wenhu Chen @WenhuChen · Apr 23, 2025
Checked ZihanWang314's RAGEN. Lots of work has been put into RAGEN's code base! Nice work!

The layers can co-evolve



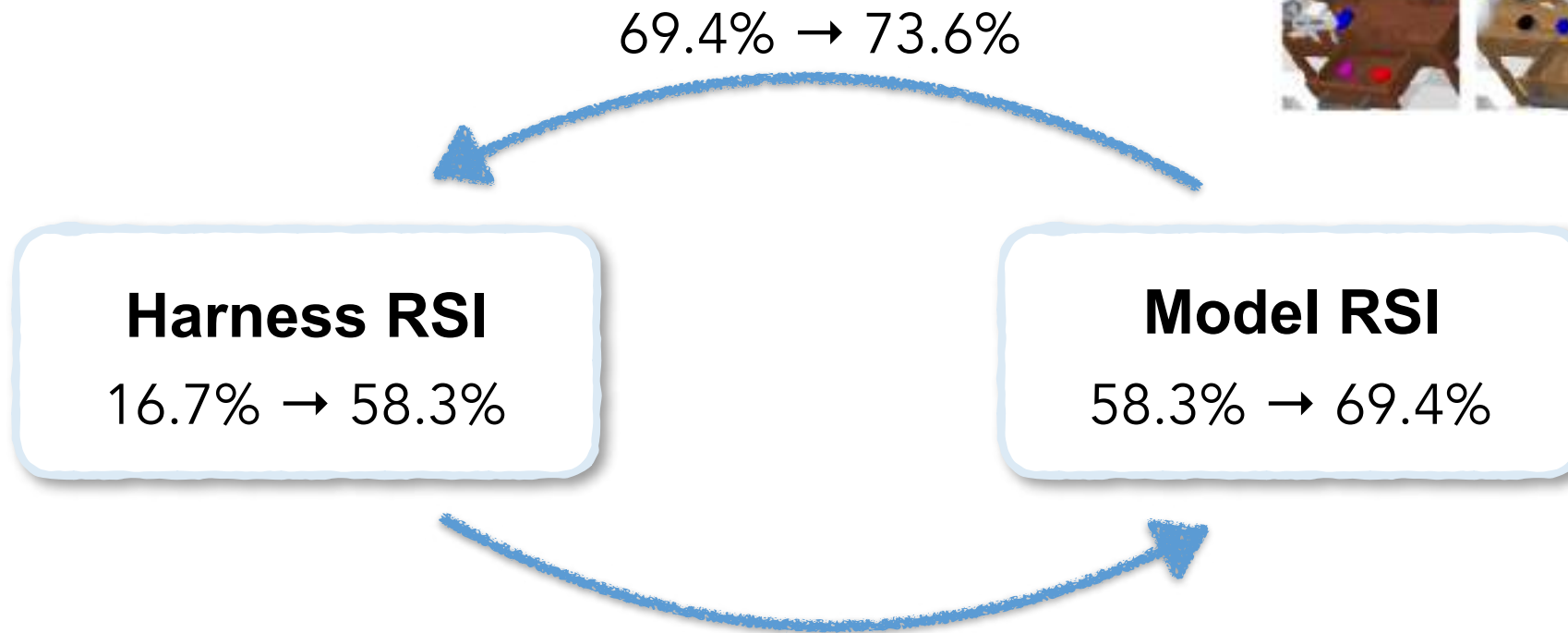
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

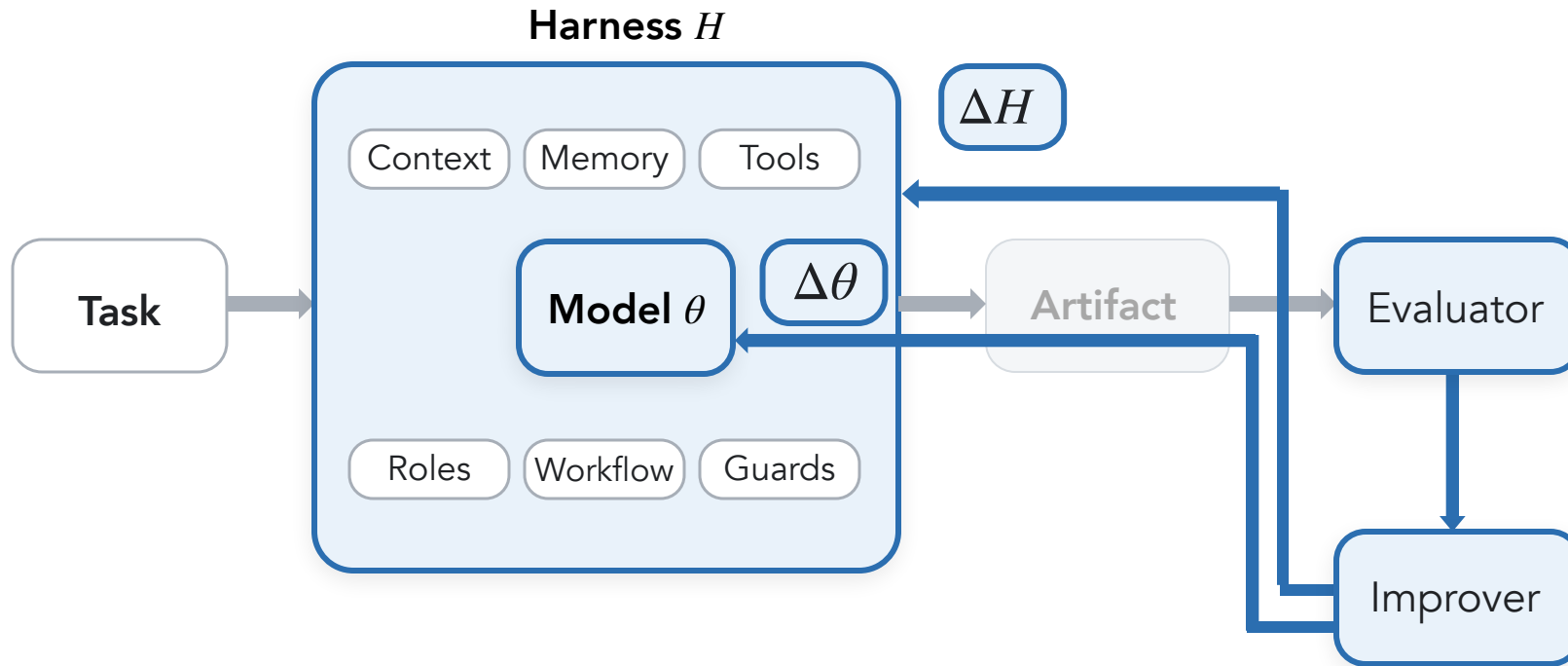
Evaluation of RSI

Co-evolve: Harness RSI + Model RSI



Harness RSI unlocks capability; Model RSI learns to use it.

The layers can co-evolve



Layer 1: Artifact RSI

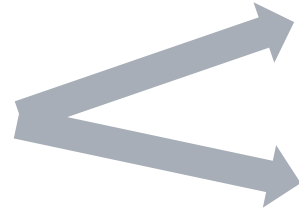
Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Evaluation: What is true RSI?

Same improver
 I



SR^0

SR^T

Weak RSI:

$$SR^T > SR^0$$

The Improver: Remains static.

Early improver
 I^0



$\Delta SR(I^0)$

Late improver
 I^T



$\Delta SR(I^T)$

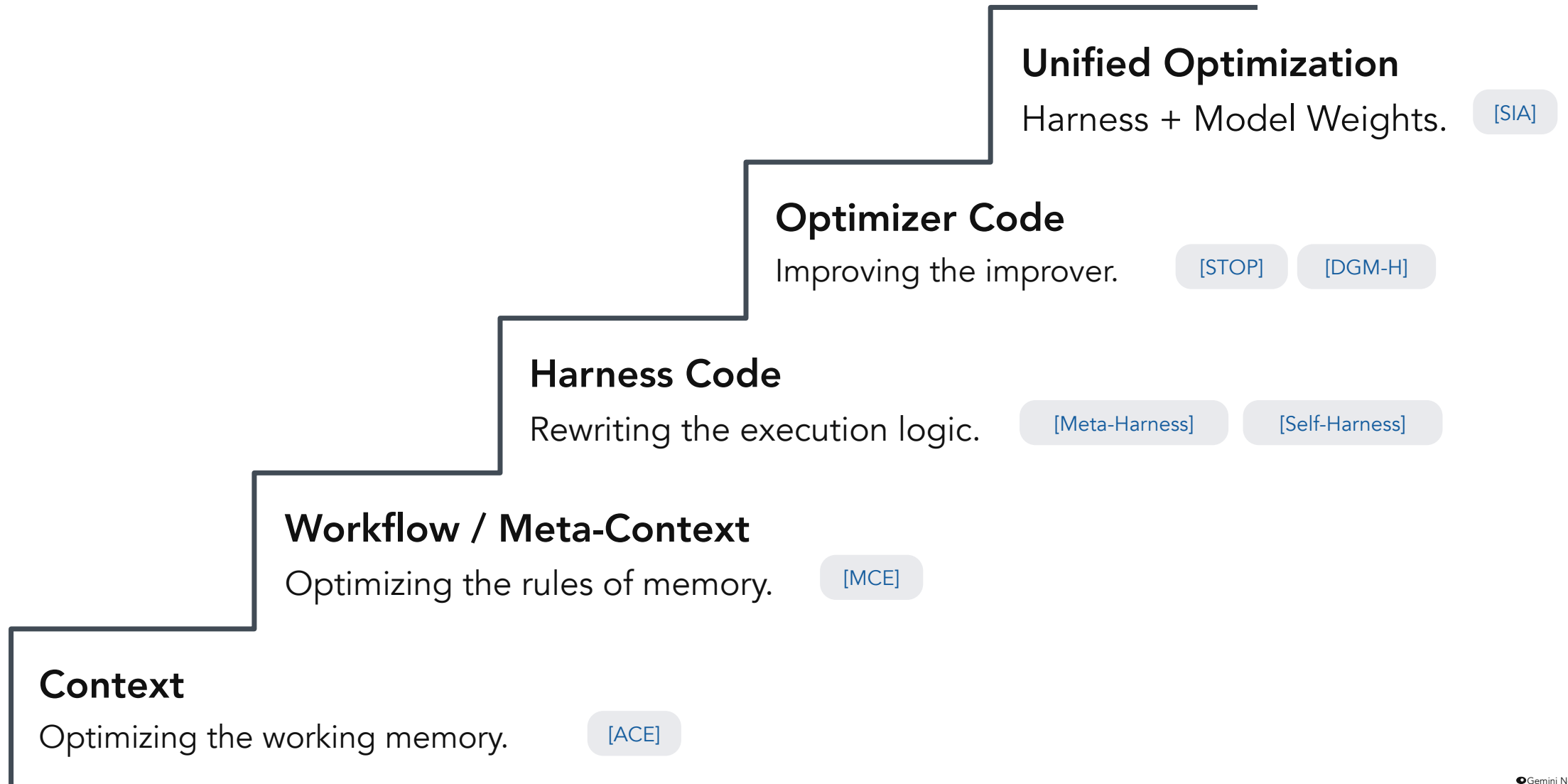
same fresh seed M ,
same budget B

Strong RSI:

$$\Delta SR(I^T) > \Delta SR(I^0)$$

The Improver: Stronger

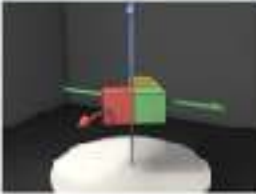
The Ladder of Abstraction



SpatialEvolve: Can we benchmark Spatial RSI?

FIGURAL v3

Table-top puzzle - fairly easy - objects are movable - space has colored shelves above them



MentalRotation

ACTIONS
none - go left - go right - done

THE TASK
rotations of the object

GOAL STATE
The object's orientation matches the GOAL image

Buttons: Mental rotation, Success, Continue



NetFolding

ACTIONS
left - right - done

THE TASK
fold maps along the creases

GOAL STATE
The sheet, unfolding to creases, matches the GOAL image

Buttons: 90 - 180 clockwise, Success, Continue



PerspectiveTadpole

ACTIONS
left - right - done

THE TASK
displace rotations about the center

GOAL STATE
The camera's view matches the TARGET image

Buttons: Perspective Tapping, Success, Continue



SpatialRelation

ACTIONS
none - done

THE TASK
alignments of objects in space

GOAL STATE
Every board contains fields as image

Buttons: Relations recognition, Success, Continue



ActiveGlasses

ACTIONS
left - right - done

THE TASK
displace rotation - progressive control

GOAL STATE
The perfectly hidden object (seen from the system) given

Buttons: View glasses, Success, Continue

ENVIRONMENTAL v3

Room-scale "body transfer" puzzle - somewhat harder - the table serves as a perspective/tapping device (Wearable - Body)



CameraRelocalization

ACTIONS
none, Extrude - rotate left - rotate right - go left - go right - done

THE TASK
status obtained from DECT

GOAL STATE
Reconstruct Perception AND the focus of the GOAL image

Buttons: Perspective Tapping, Success, Continue



NovelPathDiscovery

ACTIONS
none, Extrude - rotate left - rotate right - go left - go right - done

THE TASK
path over unknown terrain

GOAL STATE
The Goal cell reached by a path shorter than the demo

Buttons: Cognitive Map, Success, Continue

COMPOSITE v3

Desktop body transfer puzzle - somewhat harder - the table serves as a perspective/tapping device (Wearable - Body)



PoseMatching

ACTIONS
none, rotate - rotate - rotate - done

THE TASK
pose and scale/location

GOAL STATE
Every board at its SCALE/location, orientation and RGB

Buttons: Rotation, Rotation, Size consistency, Success, Continue, Continue



Assembly

ACTIONS
rotate - place - remove - done

THE TASK
path movements

GOAL STATE
Each piece at its SCALE/location, and/or color on the right side

Buttons: Rotation, Rotation, Ordering, Success, Continue, Continue

Latency should be also evaluated

What can't GPT-6 do well?

19/20 on a block-to-bowl task,
but only 2/20 on a precise puzzle-insertion task

latency: 2.5-3.4 minutes in average

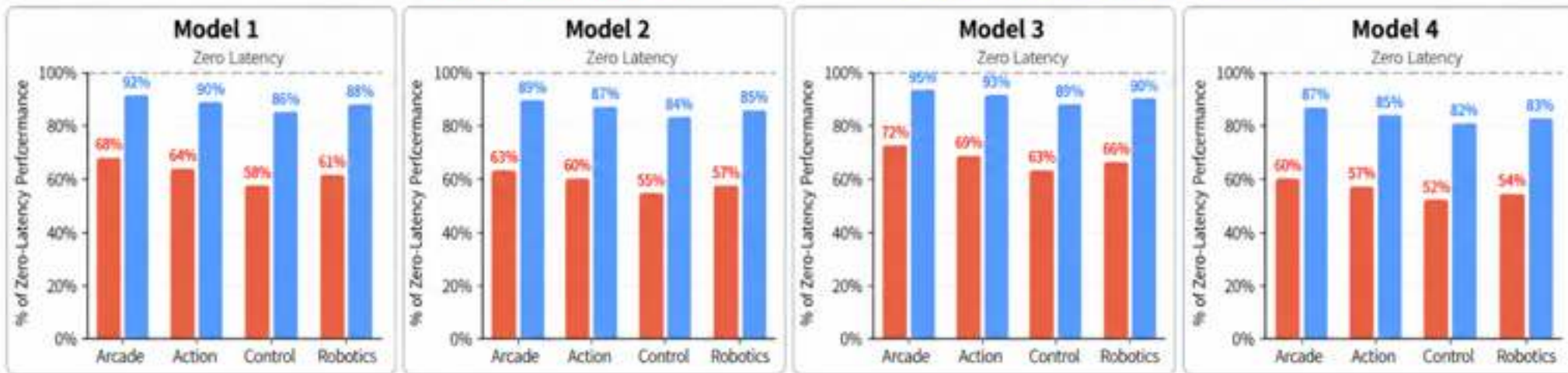
Latency should be also evaluated



Real-time latency creates
a performance gap



Our method recovers most
of the zero-latency performance

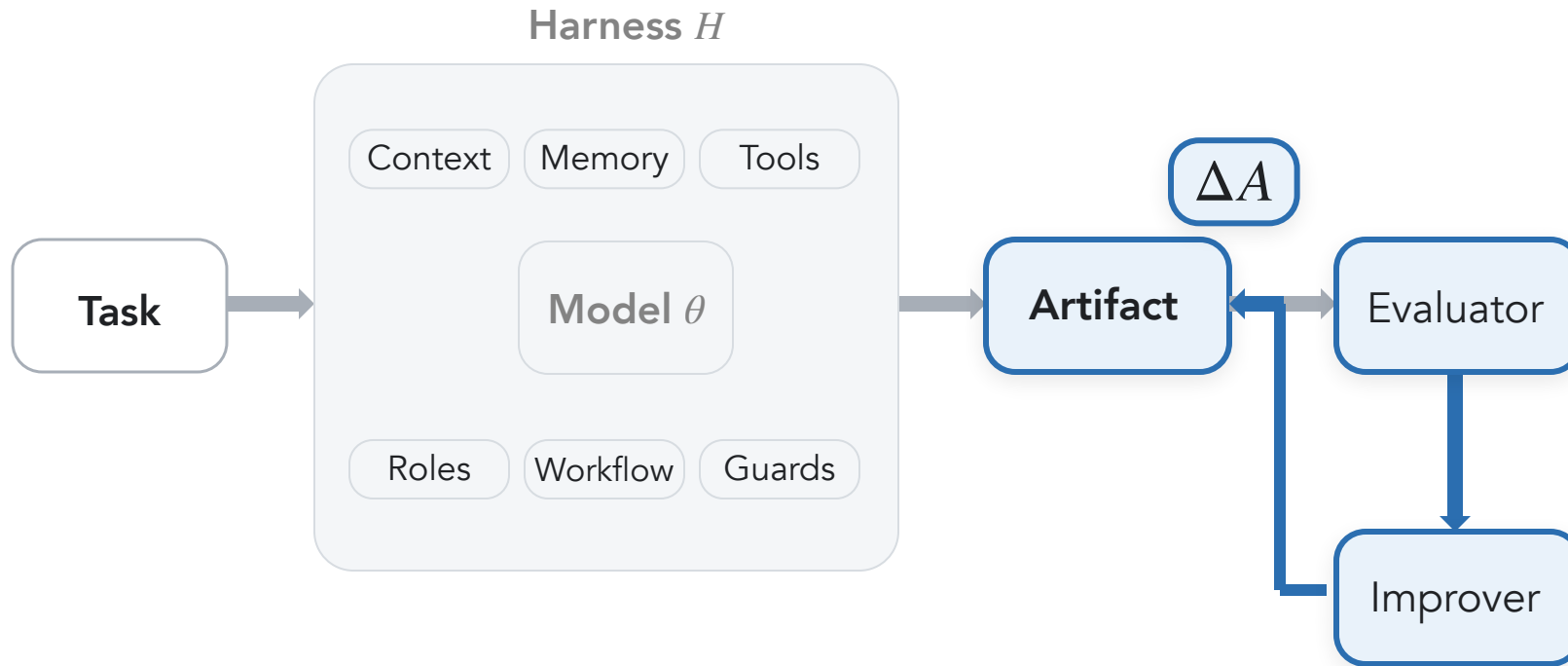


--- Zero Latency

Real-Time Latency

Ours

Summary: Artifact RSI



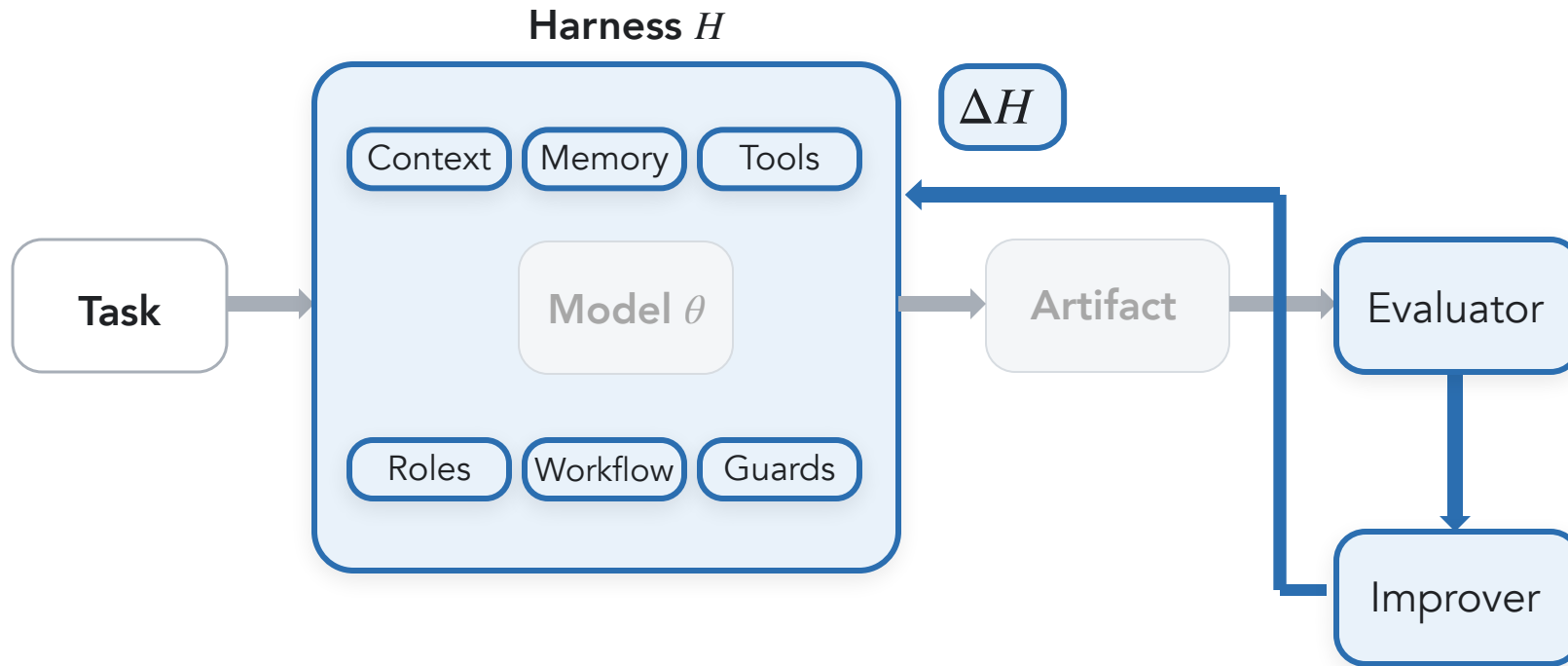
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Summary: Harness RSI



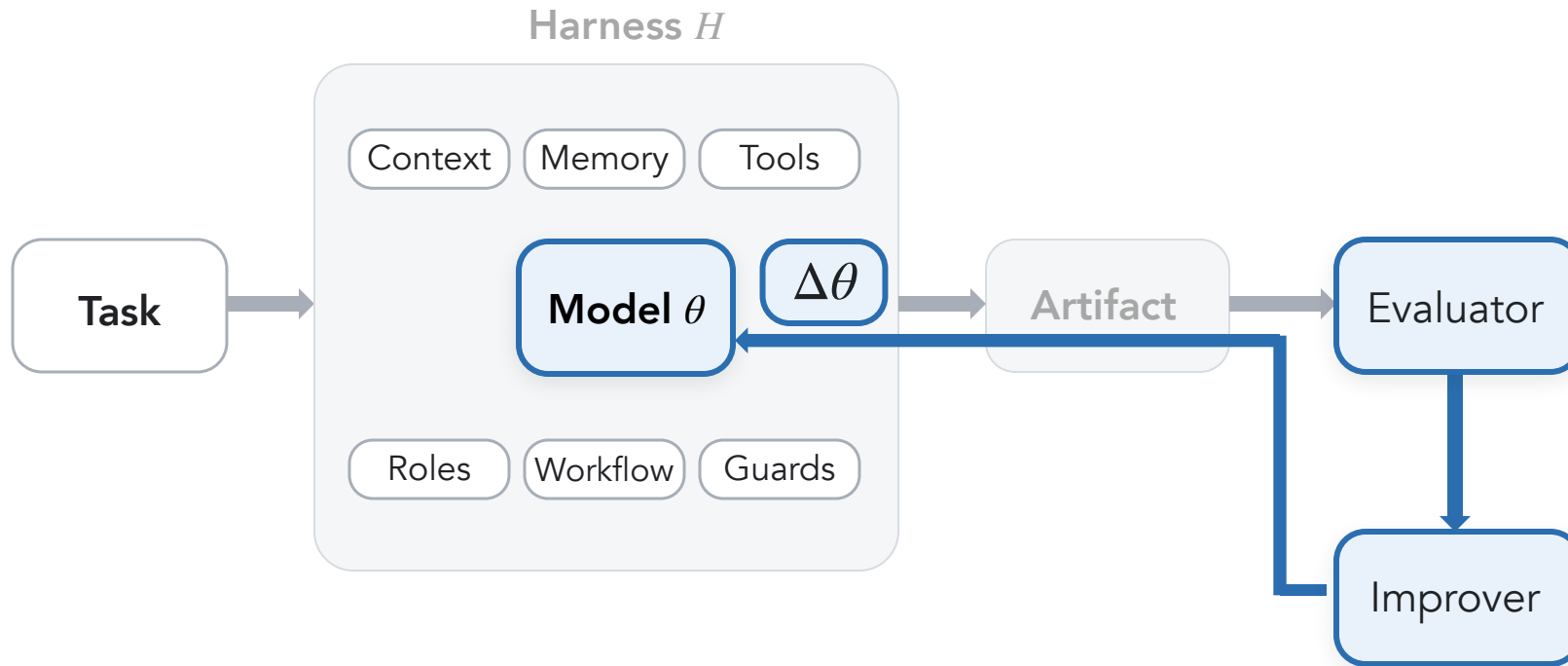
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Summary: Model RSI



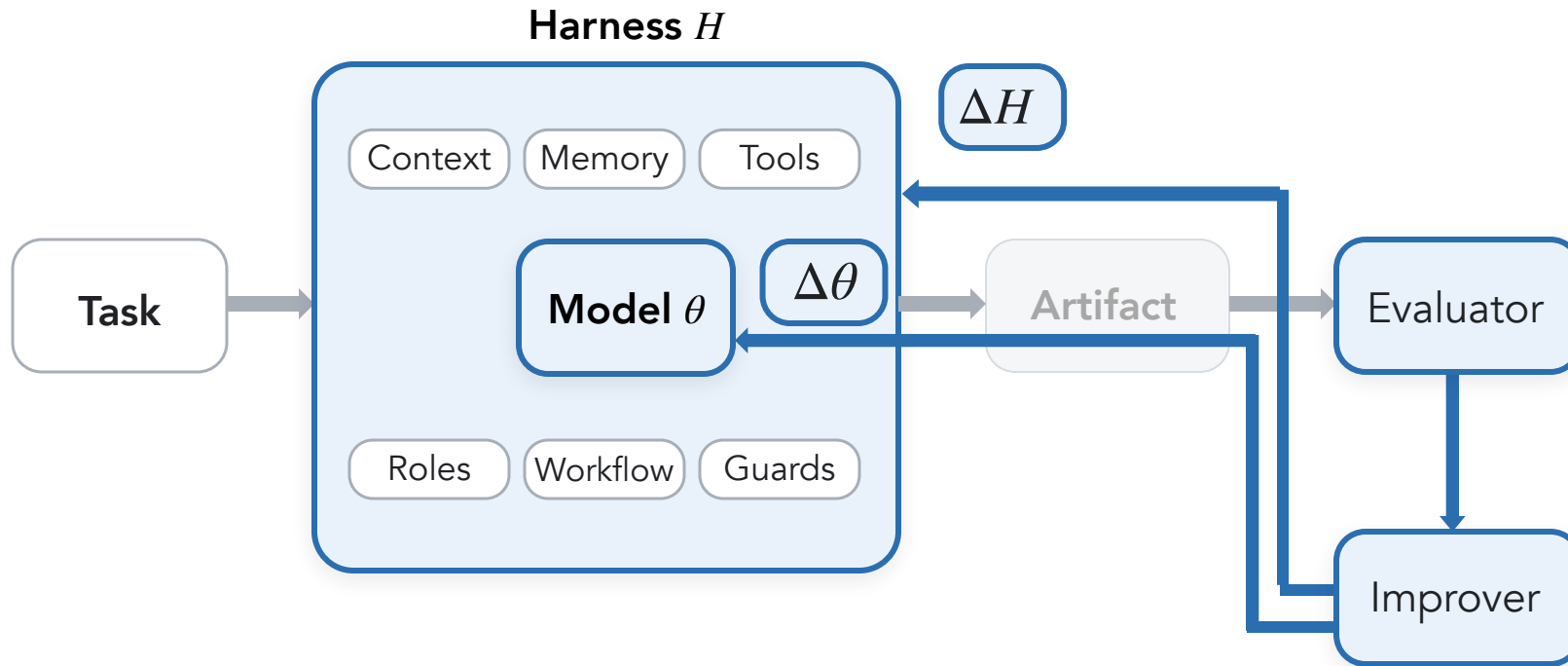
Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Summary: Evaluation



Layer 1: Artifact RSI

Layer 2: Harness RSI

Layer 3: Model RSI

Evaluation of RSI

Takeaways

1. The key of AutoResearch is a good Evaluator

2. Outer loop = research loop; Inner loop = interaction loop (with harness layer)

3. Skills are more abstract when models are stronger

4. Training data: text gradient for training

5. Latency-Aware planning

PRIORS

Common
Knowledge

World
Modeling

Instruction
Following

Reasoning
Planning

Human
Experience

STAGES

Pre-training

Mid-training

Supervised
Finetuning

Reinforcement
Learning

Harness

DATA

Static Internet Data:

Books, Images, Videos...

Play Data:

Env, Tools, External Data

Task Annotations:

<instructions, answers>

Task Feedback:

Verifiers (Goal Definitions)

Prompt Engineering

→ Context Engineering

→ Harness Engineering

→ Graph Engineering

Brute-force Scaling



Continual Learning

(We are here now)



Human + World

(Our future)

A stylized illustration of a robot with a friendly face, wearing large headphones, sitting at a desk. The desk is cluttered with various items: a laptop displaying a complex circuit board or musical score, a keyboard, several potted plants, and containers holding pens and pencils. In the background, there's a window with a view of greenery outside. Floating above the robot are several musical notes, suggesting a connection to music or audio processing. The overall scene is bright and warm, with a soft, golden light filtering through the window.

THANK YOU