[Introduction to Computer Vision] 19. Vision-Language-Action (VLA) Models

133 views Dec 18, 2025
Introduction to Computer Vision (2025-2)
Korea University
Prof. Gyeongsik Moon

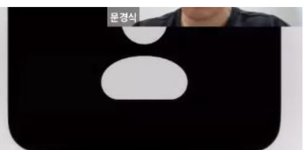
Lecture slides: https://drive.google.com/drive/folder...
References: https://drive.google.com/drive/folder...

INTRODUCTION TO COMPUTER VISION

Lecture 19 – Vision-Language-Action (VLA) Models

Gyeongsik Moon
Visual Computing and AI Lab
Korea University

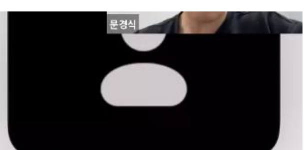
Robot Training with RL



sim_output = []

1. In the physics-based simulator run below loop: for t in range(T)
    1. Forward state (robot's current position/rotation and target position) to the policy network
    2. The policy network outputs an action from the input state
    3. sim_output.append({'state': input_state, 'action': action})
    4. Apply the action using a PD controller
    5. Obtain updated robot's current state
2. Compute reward (high much close to the target) for all elements in sim_output
3. For some elements in sim_output, increase probability of the output action with high reward

Robot Training with RL



sim_output = []

Roll out

1. In the physics-based simulator run below loop: for t in range(T)
    1. Forward state (robot's current position/rotation and target position) to the policy network
    2. The policy network outputs an action from the input state
    3. sim_output.append({'state': input_state, 'action': action})
    4. Apply the action using a PD controller
    5. Obtain updated robot's current state
2. Compute reward (high much close to the target) for all elements in sim_output
3. For some elements in sim_output, increase probability of the output action with high reward

Robot Training with RL

sim_output = []

1. In the physics-based simulator run below loop: for t in range(T)
    1. Forward state (robot's current position/rotation and target position) to the policy network
    2. The policy network outputs an action from the input state
    3. sim_output.append({'state': input_state, 'action': action})
    4. Apply the action using a PD controller
    5. Obtain updated robot's current state
2. Compute reward (high much close to the target) for all elements in sim_output
3. For some elements in sim_output, increase probability of the output action with high reward

Loss function

Robot Training with RL

- There are two problems

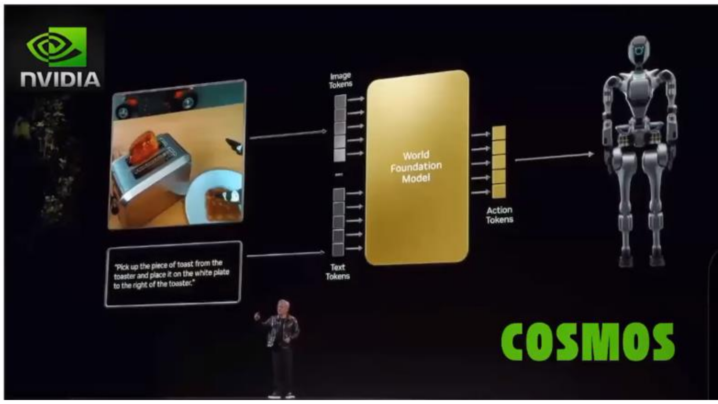
1. Roll out
    - Too slow
    - We need to run for loop in physics simulator
    - That makes the training really slow
2. Loss function
    - No explicit target for the output action
    - RL: "do the estimated action with high reward again!"
    - Supervised learning: "do the 'target' action instead of the estimated action!"

Why Reinforcement Learning?

- If you're in the two situations, you can try RL
    - Infeasible data collection
    - Non-differentiable output function
- Representative examples are agents in video game and robots

Vision-language-action (VLA) models

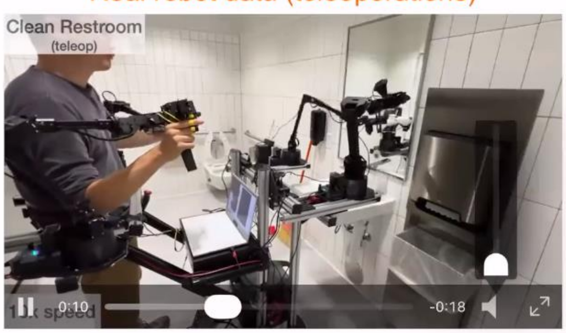
- The good points of RL becomes limitations of RL at the same time
- How to solve this?
    - Vision-language-action (VLA) models



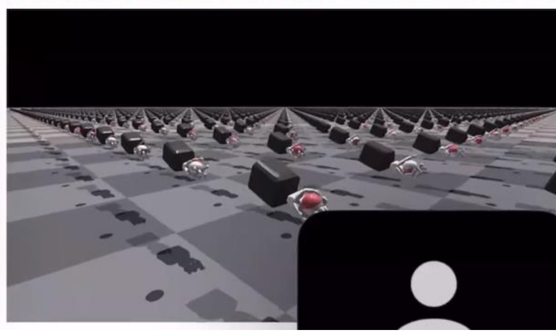
Vision-language-action (VLA) models

- Use supervised learning instead of RL for the robot training
- How to get data?

Real robot data (telooperations)

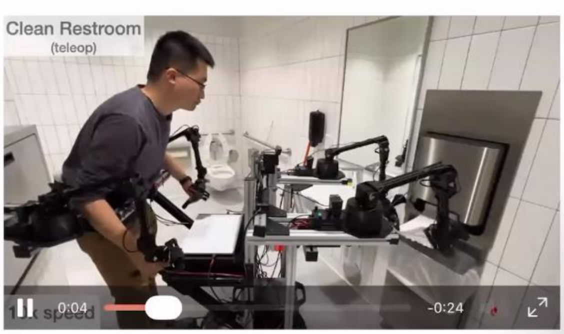


RL in physics simulators (we've learned so far)



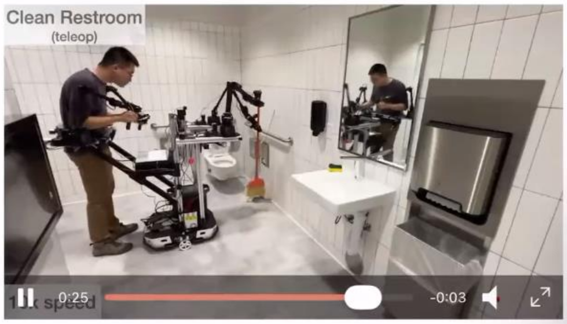
Teleoperations

- Use real robots to collect data
- Humans control robots to get robot trajectories
- Costly and hard to scale up



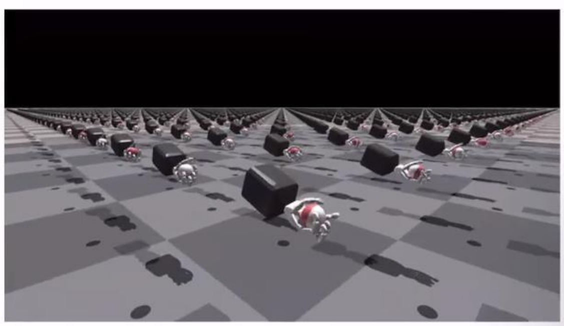
Teleoperations

- Use real robots to collect data
- Humans control robots to get robot trajectories
- Costly and hard to scale up



RL in Physics Simulators

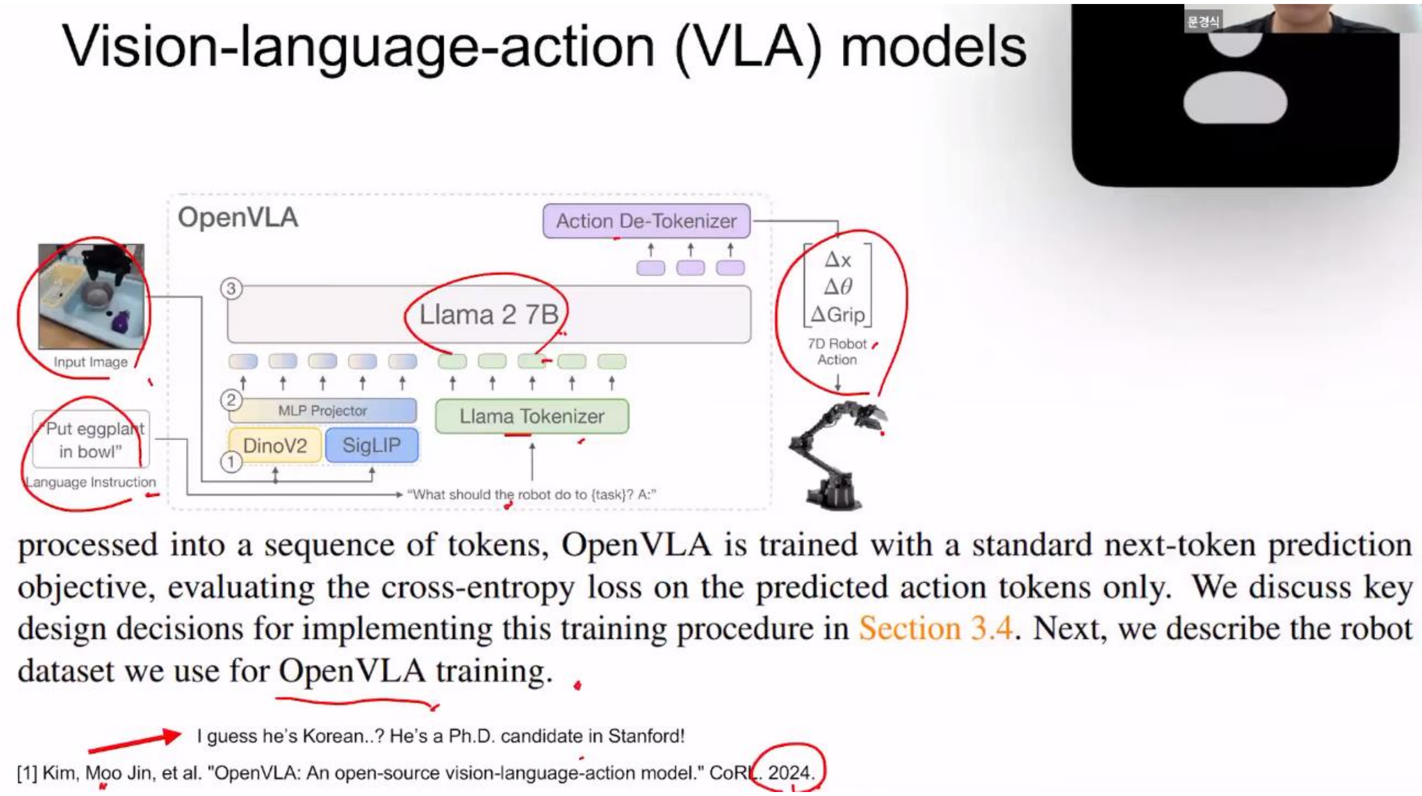
- Use RL for each task instead of building a universal policy network
- For example, train a policy network with RL only for hand-obj grasp
- In this way, we can avoid scale-up issue of RL
- Sim2Real gap: Gap between simulation environments and real worlds



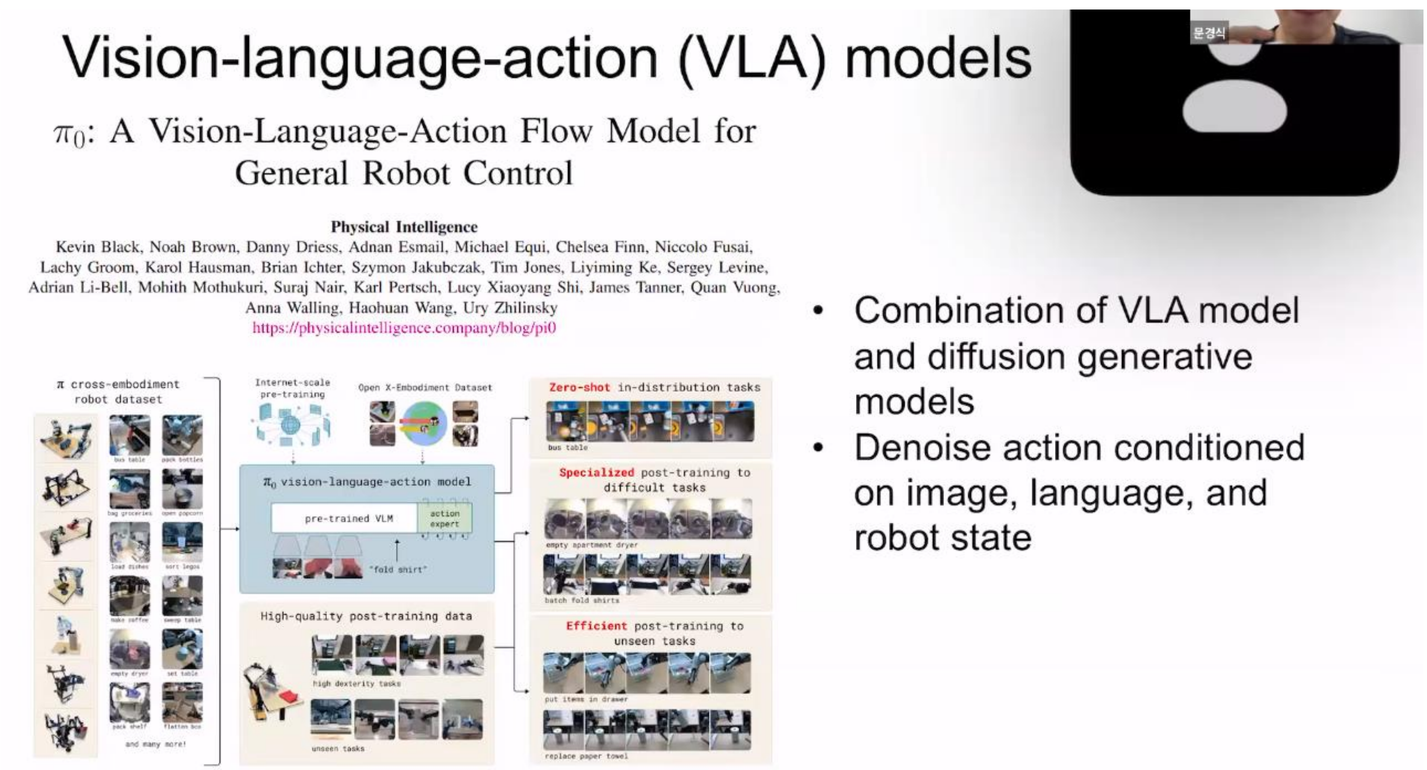
Vision-language-action (VLA) models

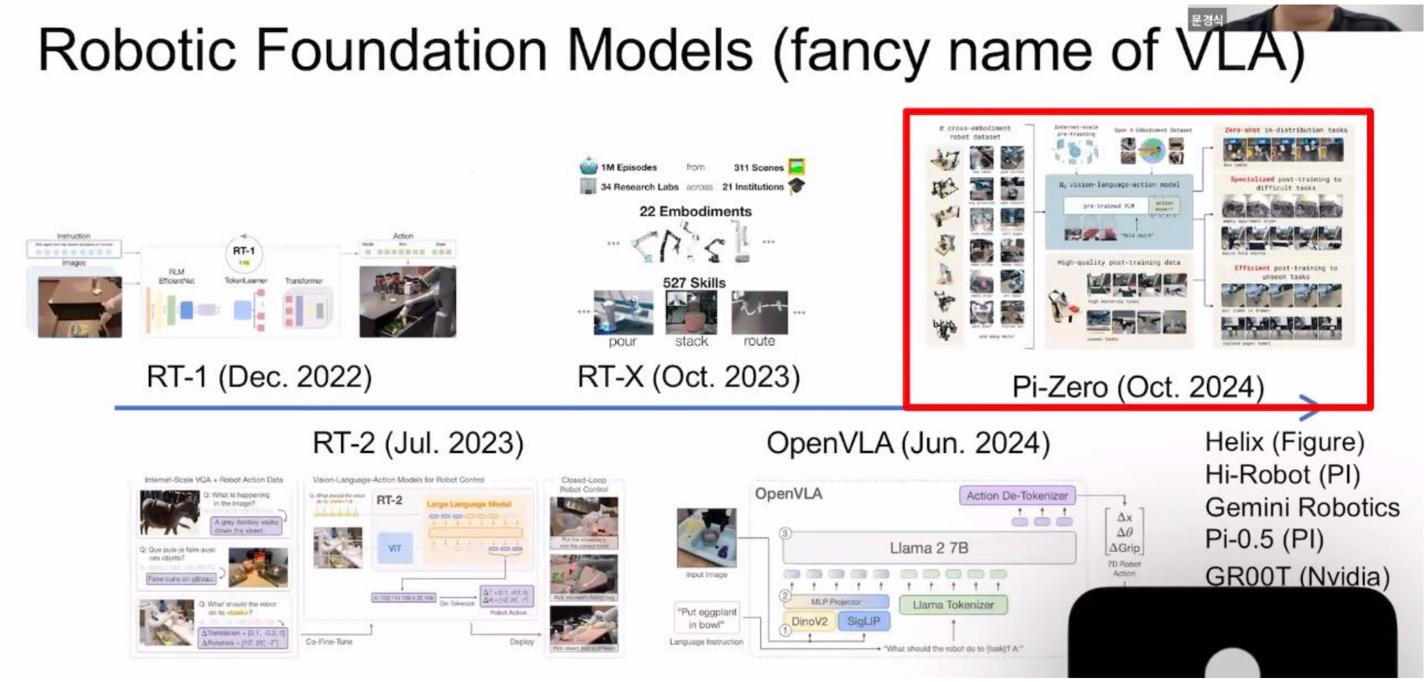
- Use supervised learning instead of RL for the robot training
- *Now, we have data (pairs of (state, action))*
- Loss = distance(net(state) – action_target)
- No roll out
- With explicit target

This is also called **behavior cloning** for **imitation learning**.

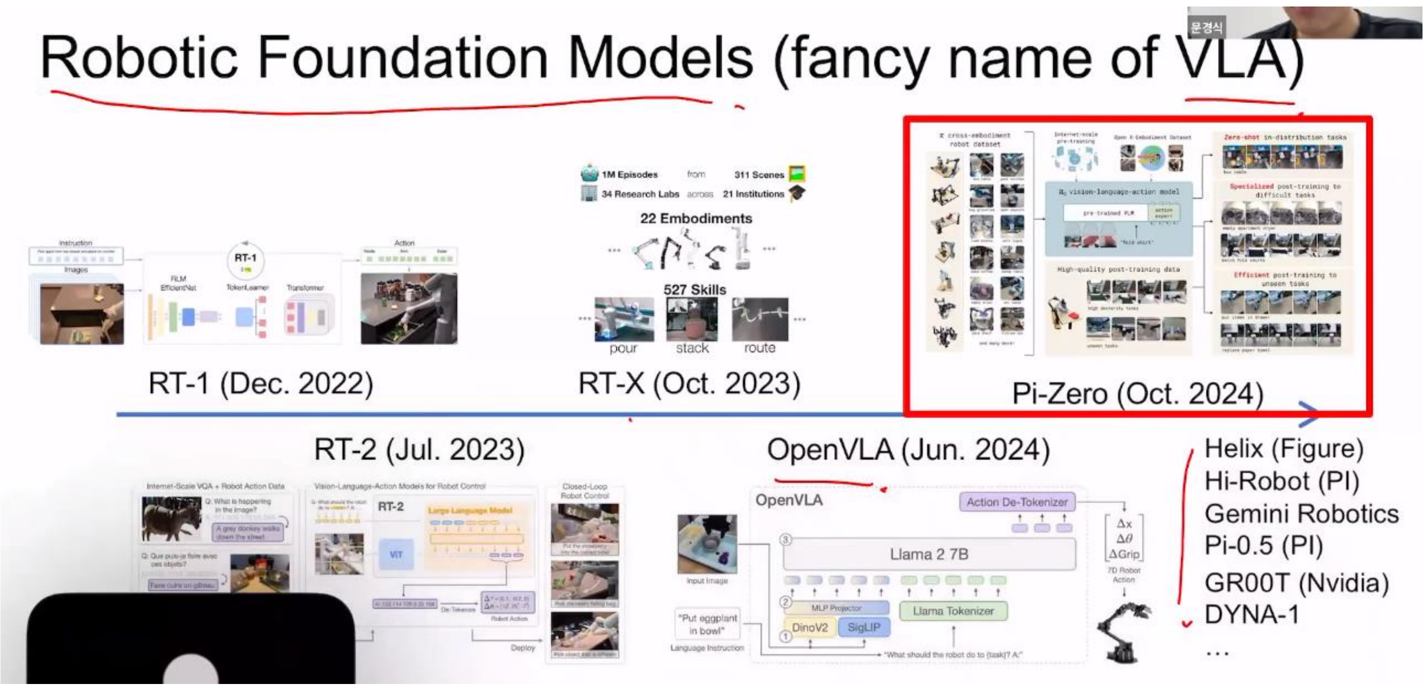


OpenVLA has no diffusion or generative modeling





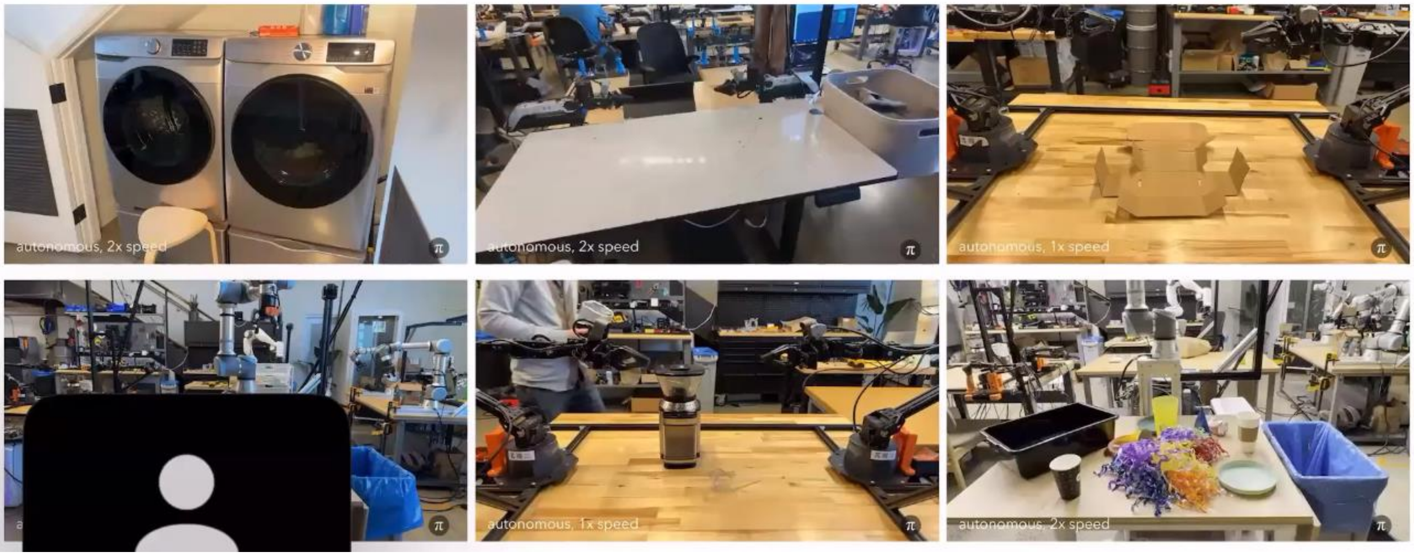
문경식
Robotic Foundation Models (fancy name of VLA)
RT-1
RT-1 (Dec. 2022)
1M Episodes from 311 Scenes
34 Research Labs across 21 Institutions
22 Embodiments
527 Skills
pour
stack
route
RT-X (Oct. 2023)
Pi-Zero (Oct. 2024)
RT-2 (Jul. 2023)
RT-2
Large Language Model
ViT
OpenVLA (Jun. 2024)
OpenVLA
Action De-Tokenizer
Llama 2 7B
MLP Projector
Llama Tokenizer
DinoV2
SigLIP
Input Image
"Put eggplant in bowl"
Language Instruction
"What should the robot do to {task}? A:"
7D Robot Action
Helix (Figure)
Hi-Robot (PI)
Gemini Robotics
Pi-0.5 (PI)
GR00T (Nvidia)



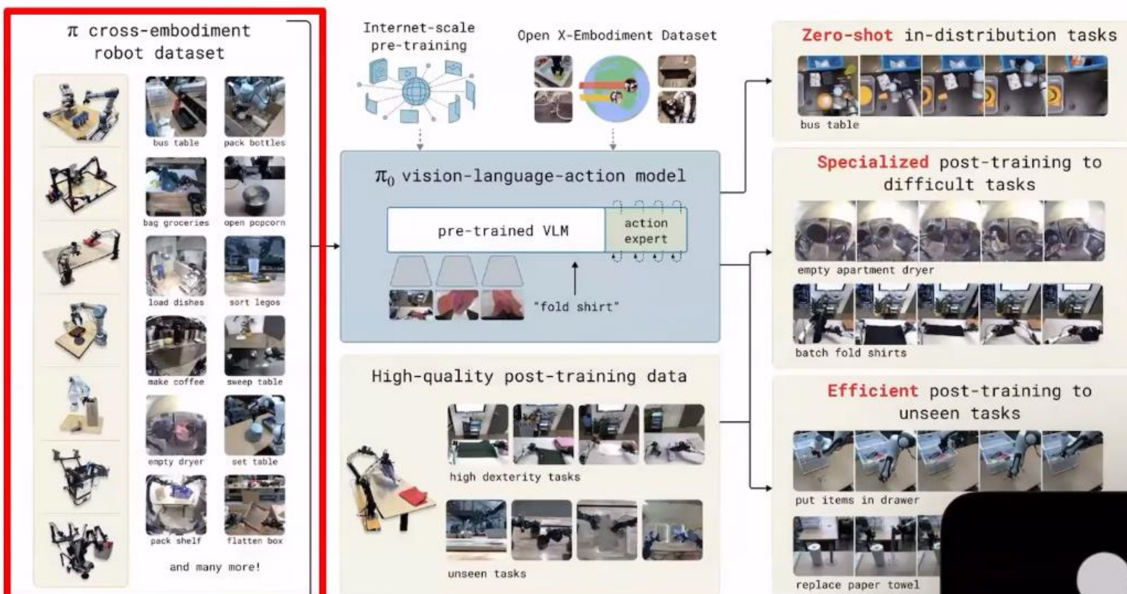
문경식
Robotic Foundation Models (fancy name of VLA)
RT-1
RT-1 (Dec. 2022)
1M Episodes from 311 Scenes
34 Research Labs across 21 Institutions
22 Embodiments
527 Skills
pour
stack
route
RT-X (Oct. 2023)
Pi-Zero (Oct. 2024)
RT-2 (Jul. 2023)
RT-2
Large Language Model
ViT
OpenVLA (Jun. 2024)
OpenVLA
Action De-Tokenizer
Llama 2 7B
MLP Projector
Llama Tokenizer
DinoV2
SigLIP
Input Image
"Put eggplant in bowl"
Language Instruction
"What should the robot do to {task}? A:"
7D Robot Action
Helix (Figure)
Hi-Robot (PI)
Gemini Robotics
Pi-0.5 (PI)
GR00T (Nvidia)
DYNA-1
…

Pi-Zero by Physical Intelligence

https://www.physicalintelligence.company/blog/pi0

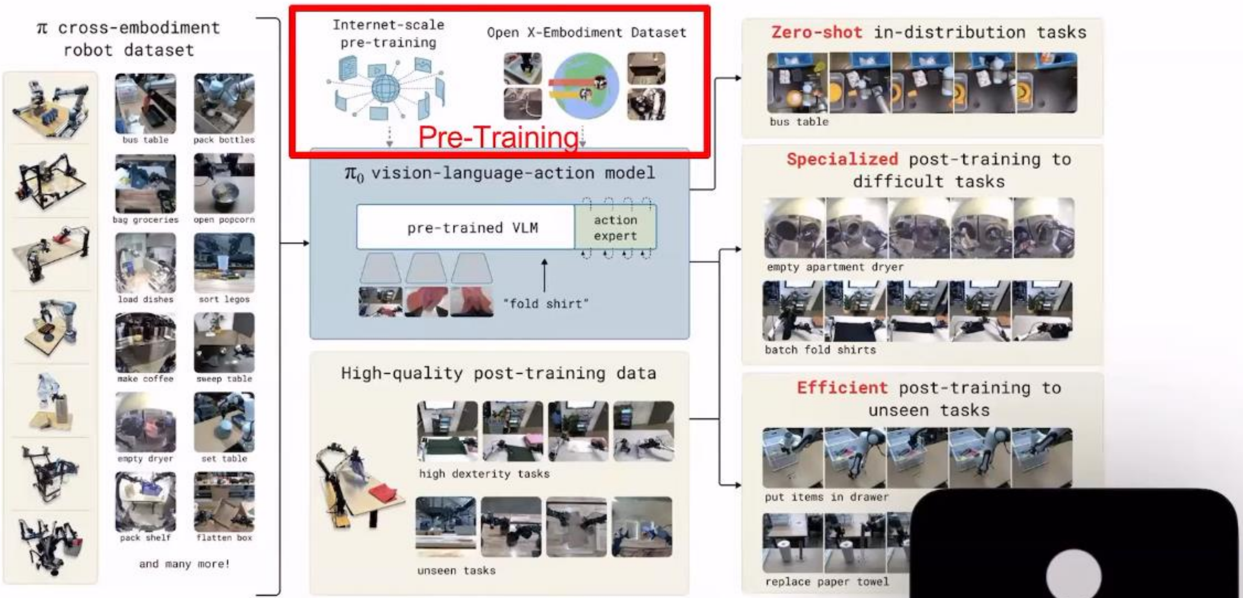


Pi-Zero by Physical Intelligence

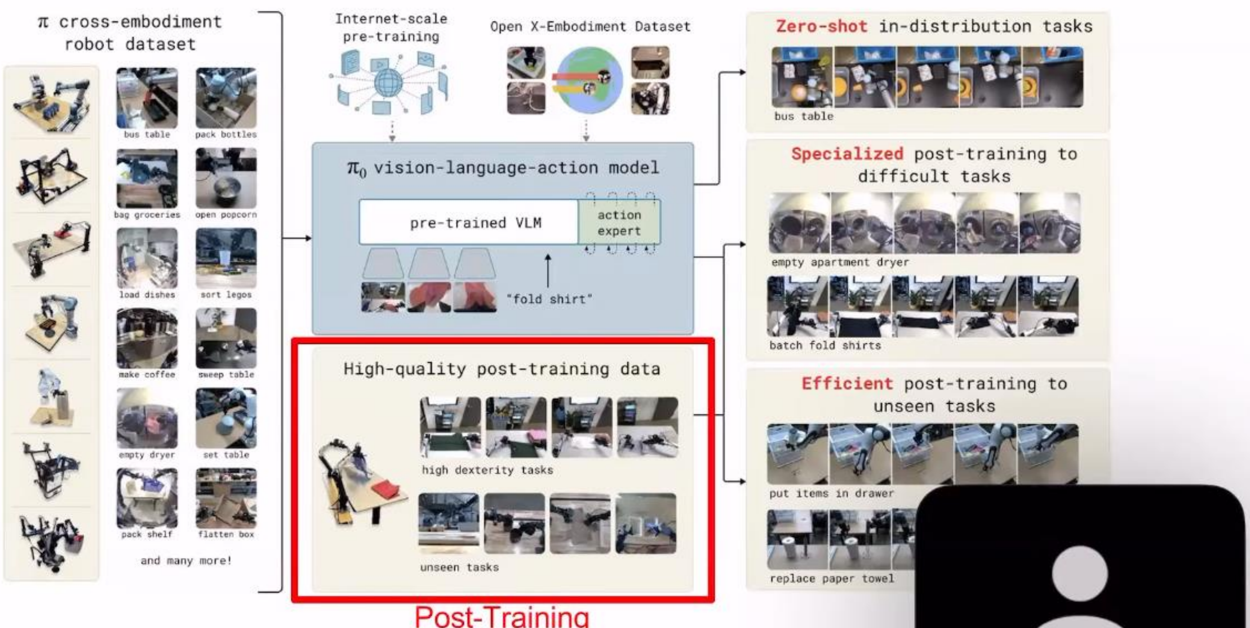


Cross-Embodiment Dataset

Pi-Zero by Physical Intelligence

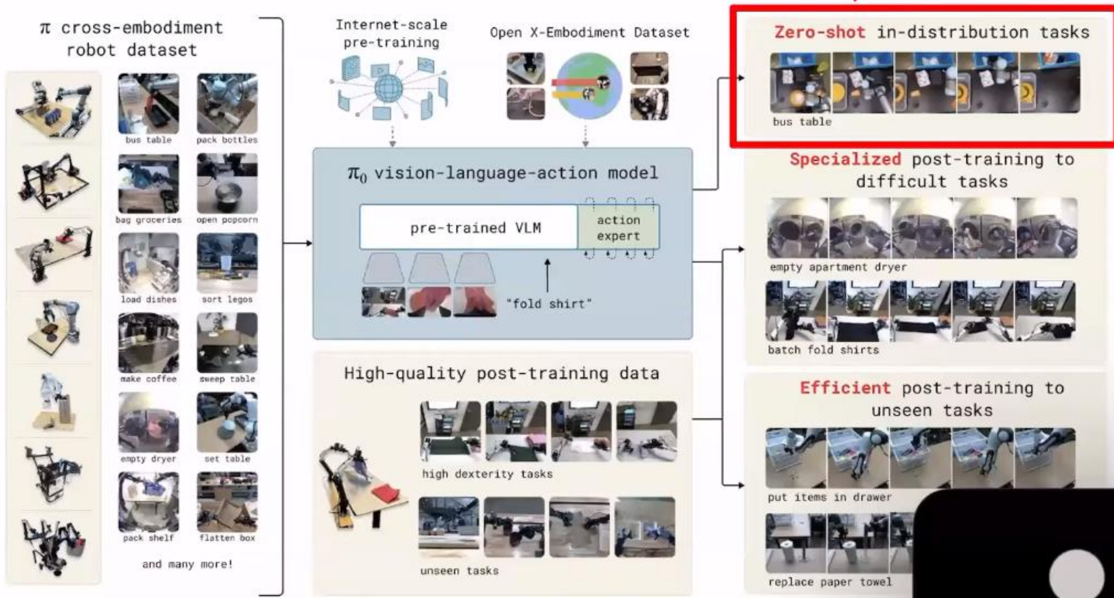


Pi-Zero by Physical Intelligence



Pi-Zero by Physical Intelligence

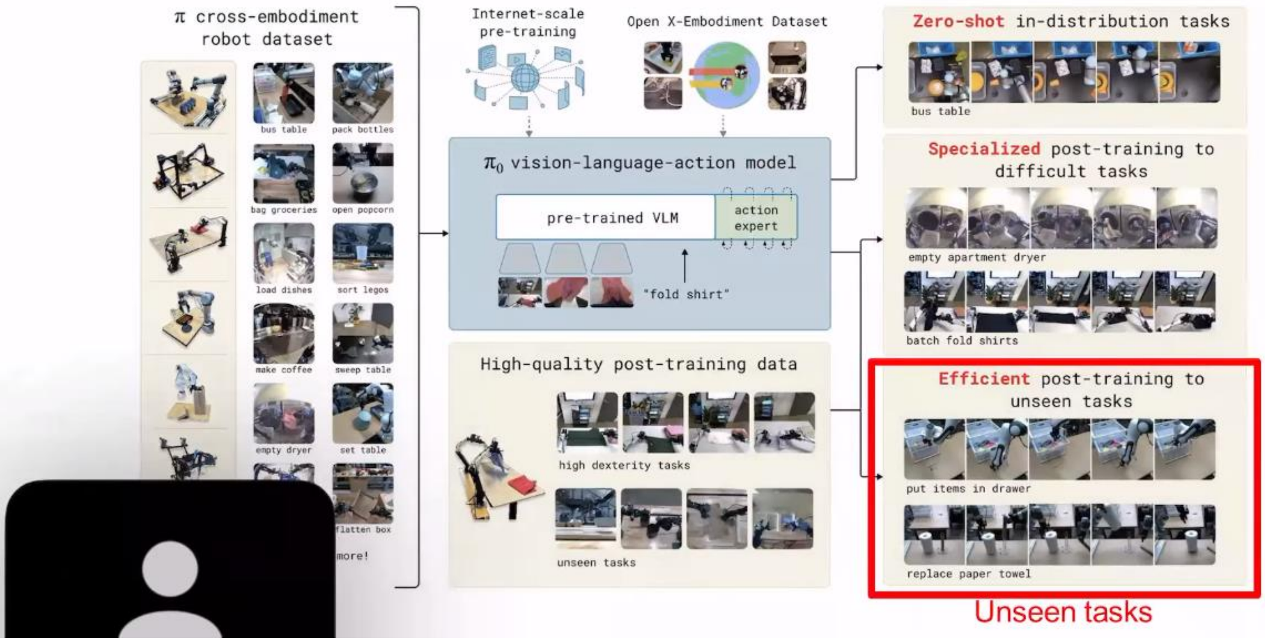
Simple in-distribution tasks



Pi-Zero by Physical Intelligence

Complicated in-distribution tasks

Pi-Zero by Physical Intelligence



Pi-Zero by Physical Intelligence

Physical Intelligence (π)

Open Sourcing π_0

Published February 4, 2025
Email research@physicalintelligence.company
Repo Physical-Intelligence/openpi

README | Apache-2.0 license

openpi

openpi holds open-source models and packages for robotics, published by the Physical Intelligence team.

Currently, this repo contains two types of models:

- the π_0 model, a flow-based diffusion vision-language-action model (VLA)
- the π_0-FAST model, an autoregressive VLA, based on the FAST action tokenizer.

For both models, we provide *base model* checkpoints, pre-trained on 10k+ hours of robot data, and examples for using them out of the box or fine-tuning them to your own datasets.

This is an experiment: π_0 was developed for our own robots, which differ from the widely used platforms such as ALOHA and DROID, and though we are optimistic that researchers and practitioners will be able to run creative new experiments adapting π_0 to their own platforms, we do not expect every such attempt to be successful. All this is to say: π_0 may or may not work for you, but you are welcome to try it and see!

Evaluation of the Robot Learning Models

- Evaluation is primarily conducted in the real world
- Real-world evaluation is costly and noisy
  - "We have large enough budget such that we can still make progress."
- Weak correlation between training loss and real-world success rate.
  - Training objectives vs task-specific metrics, training vs testing horizons

ALOHA 2

Evaluation of the Robot Learning Models

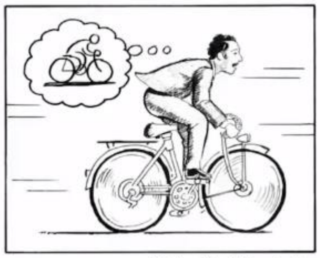
What about evaluation in simulation?

- Sim-to-real gap: rigid / deformable / cloth
- Efficient asset generation
- Digitalization of the real world
- Procedural generation of realistic and diverse scenes
- Correlation between sim and real

ImageNet in Embodied AI?

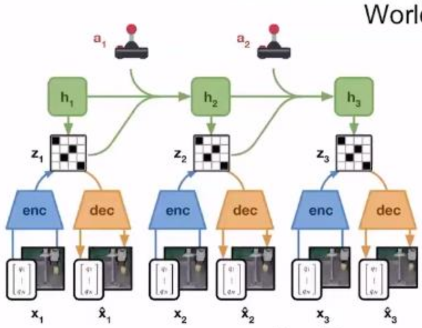
Habitat 3.0

Robotic Foundation Model + World Models

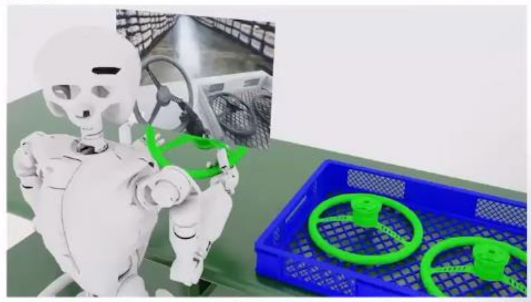


World Models

1X World Models



DayDreamer



Nvidia Cosmos - World Foundation Model

- 3D?
- Structural Prior?
- Learning + Physics?

Foundation Models for Embodied Agents

- Current foundation models are not tailored for embodied agents
  - LLM/VLM can fail in embodied-related tasks
  - Limited understanding of geometric / embodied / physical interactions
  - Reinforcement learning (RL) from human feedback → RL from Embodied Feedback

GPT

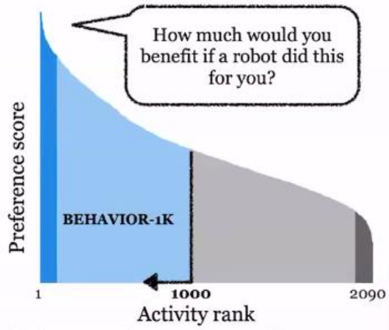
Segment Anything

DINO

Adaptation / Life-Long Learning

- Adapt to new scenarios
- Adapt to human preferences
- Self improve / life-long learning

Adapt to new scenarios



Adapt to human preferences

Impro