Stanford Talk / June 9th 2026

Beyond Diffusion Policies: Drifting Field Policy for One-Step Generative Robot Control

Juil Koo

PhD student @ KAIST

Research Intern @ Meta

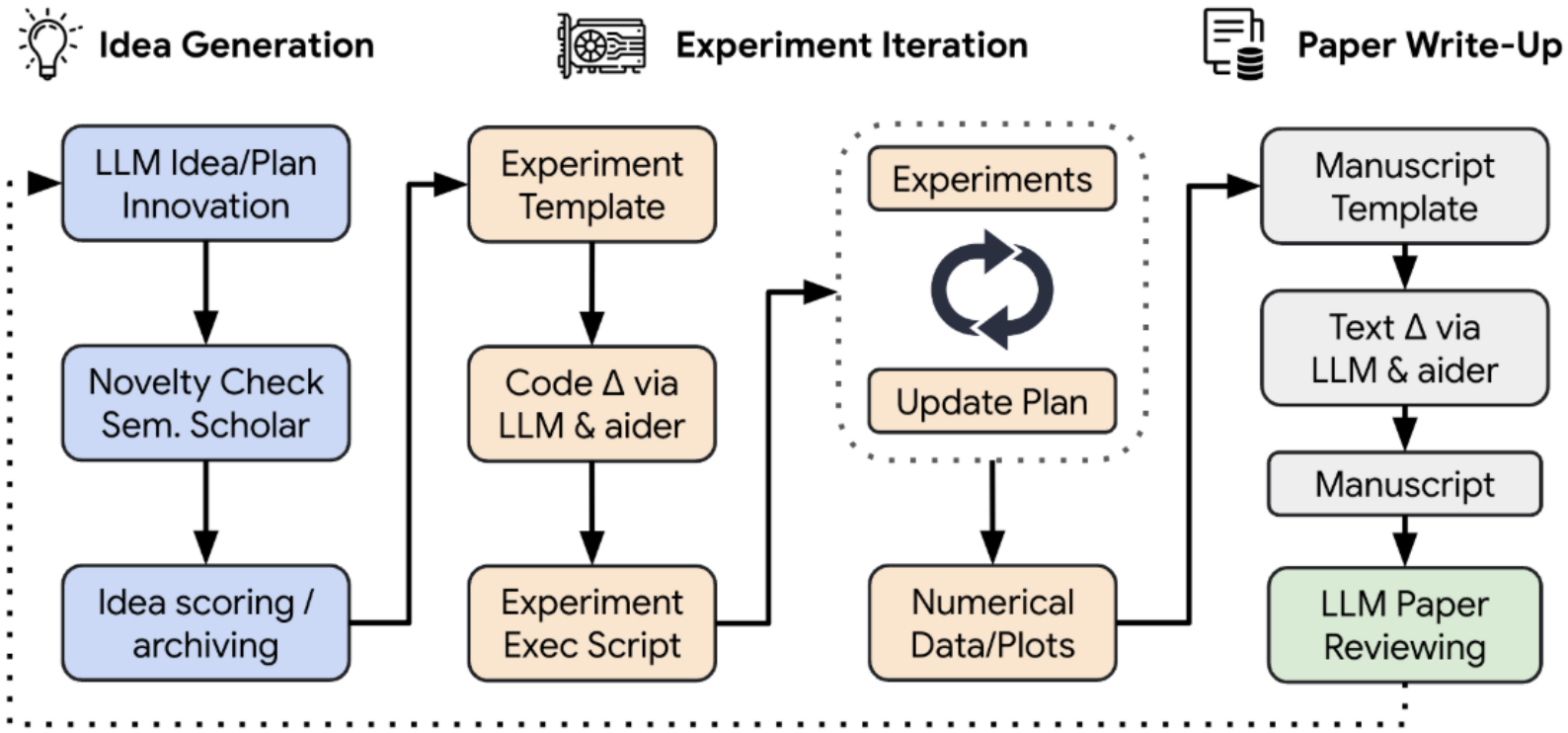
KAIST

AI Systems in 2026 / AI that Lives on Your Computer

sakana.ai

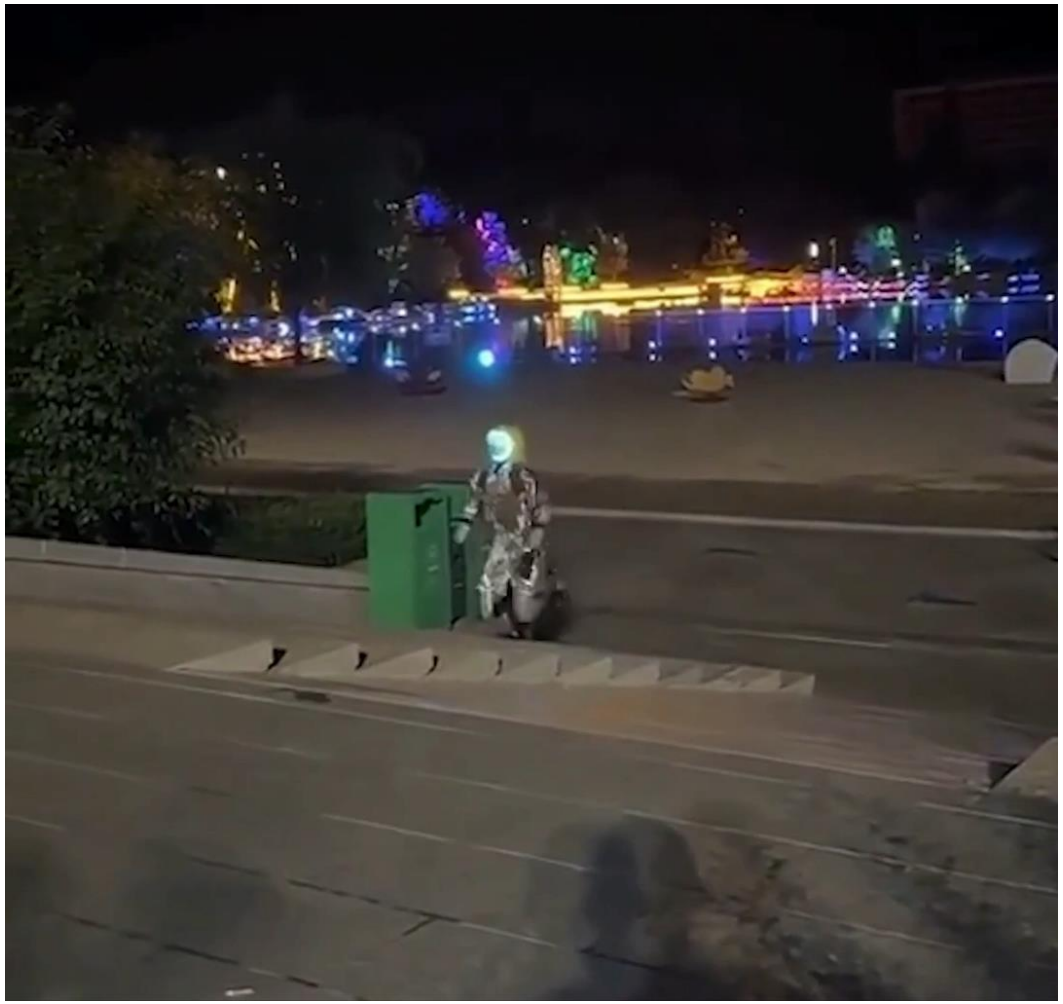
The AI Scientist: Towards Fully Automated AI Research, Now Published in *Nature*

March 26, 2026



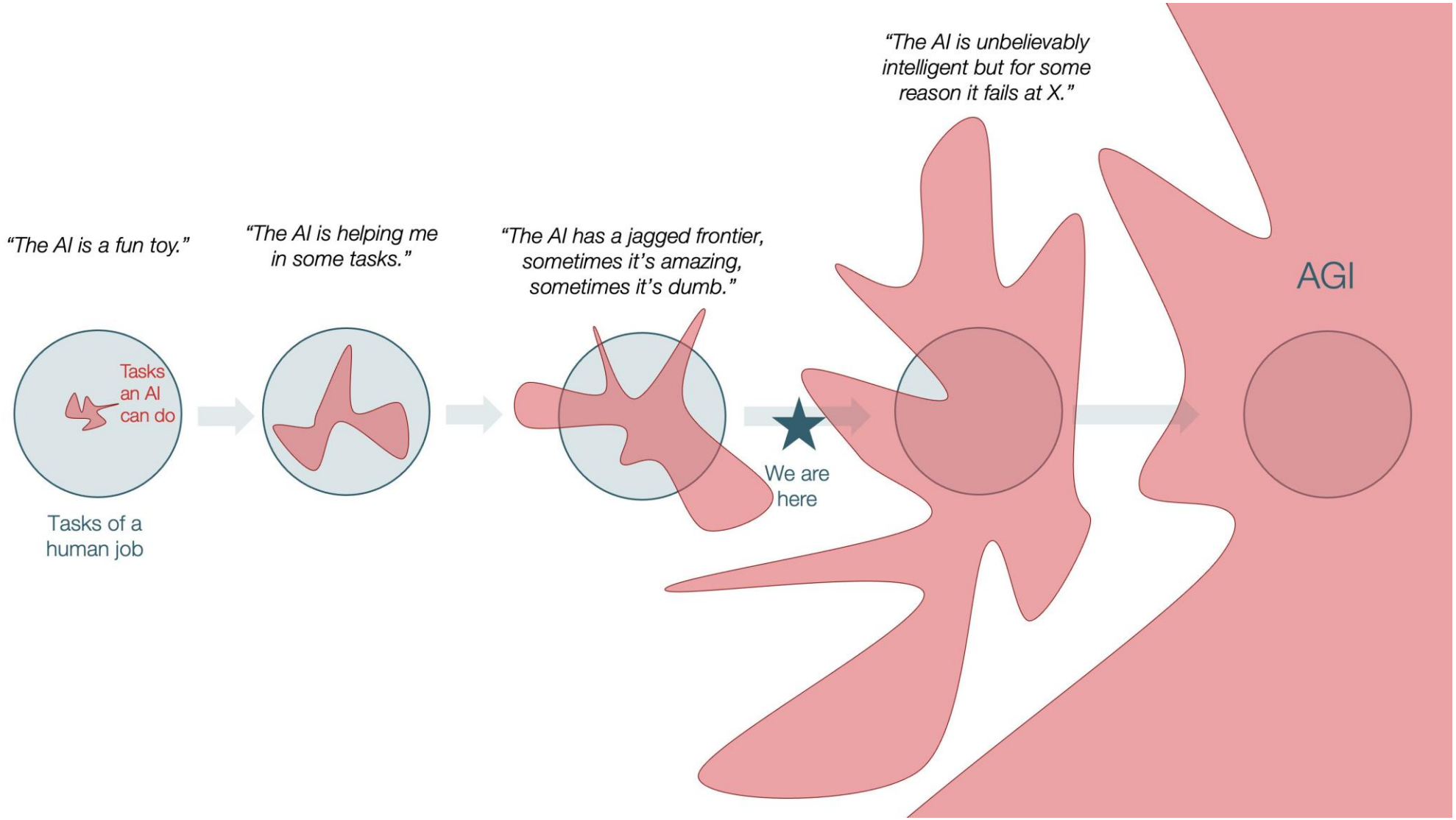
Embodied AI Systems in 2026

“Can you please do the shopping for me?”



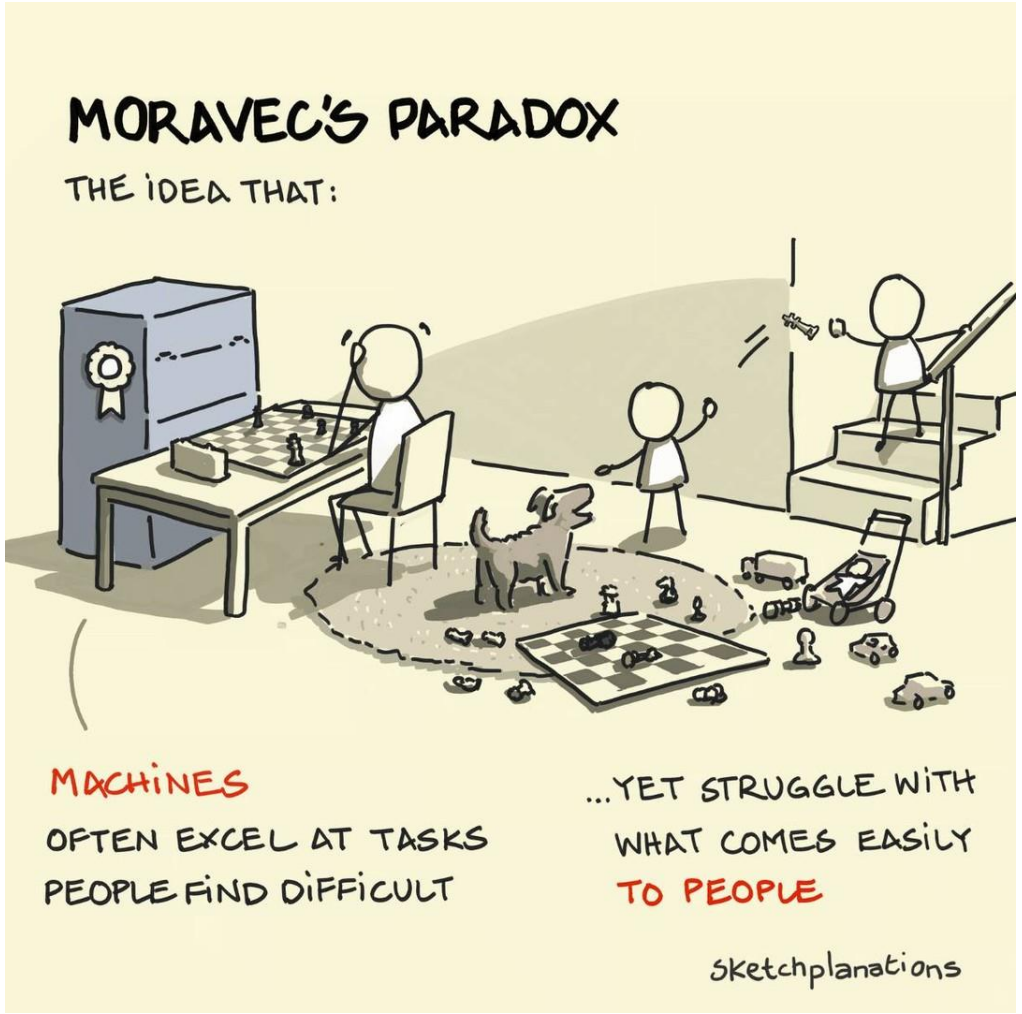
https://www.youtube.com/watch?v=OqsqzPCLvE4

Jagged Intelligence



Moravec's Paradox

"It is … easy to make computers exhibit adult level performance on intelligence tests …, and ***difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility****." –Hans Moravec*



From Reasoners to Physical Agents

OpenAI Imagines Our AI Future

Stages of Artificial Intelligence

Level 1	Chatbots, AI with conversational language
Level 2	Reasoners, human-level problem solving
Level 3	Agents, systems that can take actions
Level 4	Innovators, AI that can aid in invention
Level 5	Organizations, AI that can do the work of an organization

Current status of AI

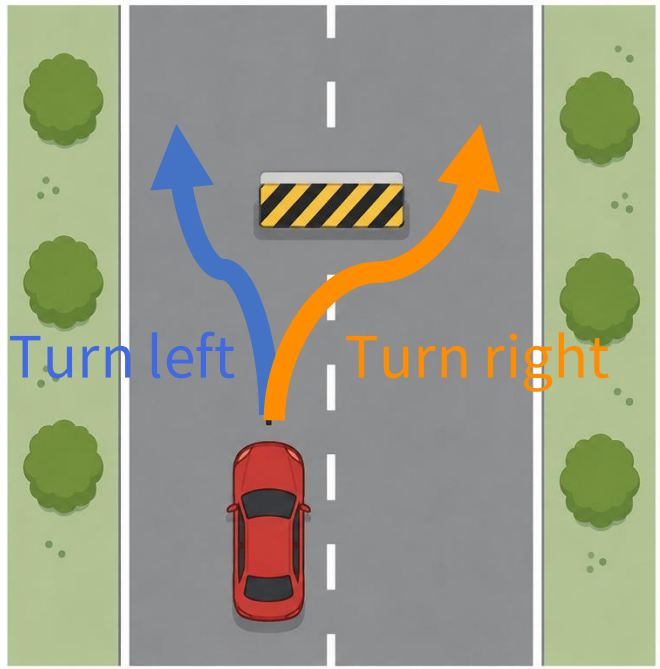
Three Requirements for Action Experts

- Multimodal action distributions
- Real-time inference
- Learning from experience

Action Multimodality

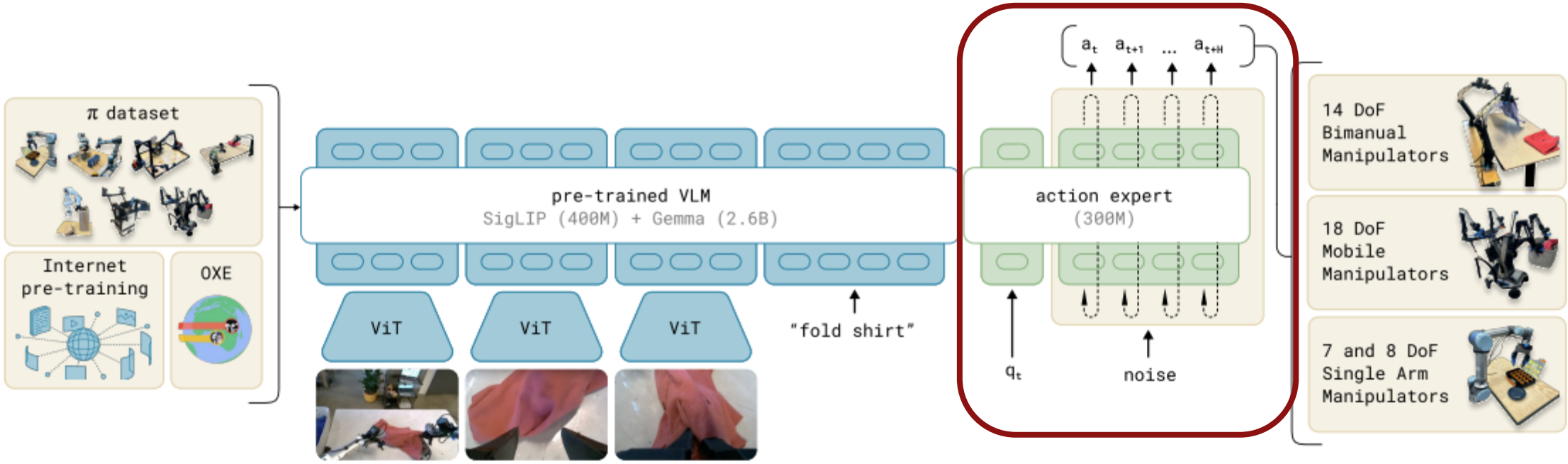
Multiple valid actions for the same observation.

“Grasp the mug on the table.”



Generative Policies in Embodied AI

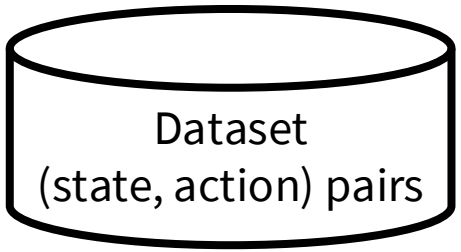
Generative policies are becoming the action head of embodied foundation models.



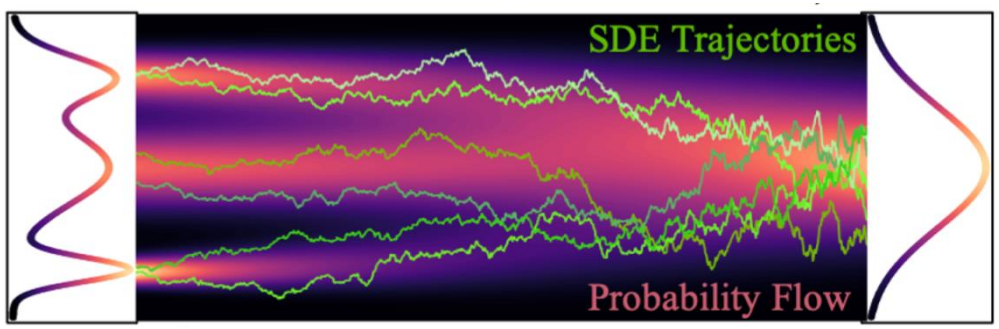
[Physical Intelligence Team, π_0, 2024]

Behavior Cloning by Diffusion Policies

Imitating demonstrations



Training
diffusion models



Capturing multimodality

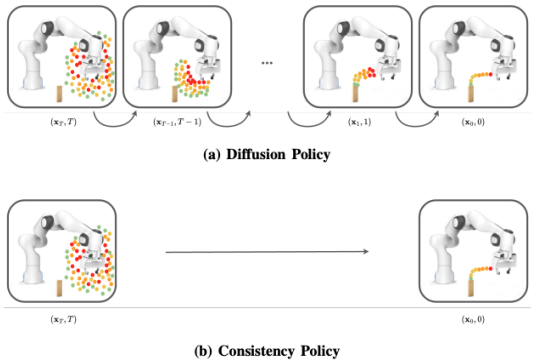
Gaussian Policy

Diffusion Policy

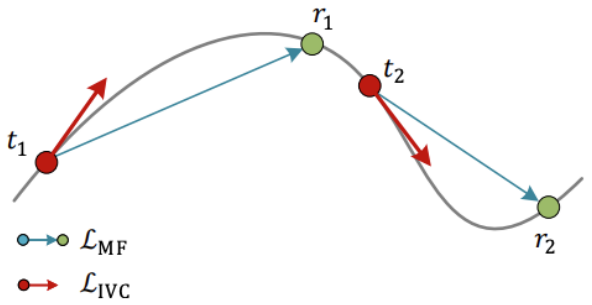
[Chi et al., DiffusionPolicy, RSS 2023]

What's Next in Generative Policies?

1. Real-Time Control

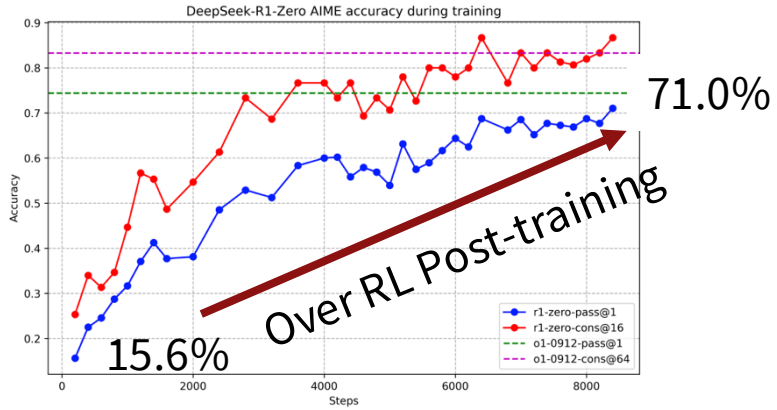


[Prasad et al., Consistency Policy, *RSS 2024*]

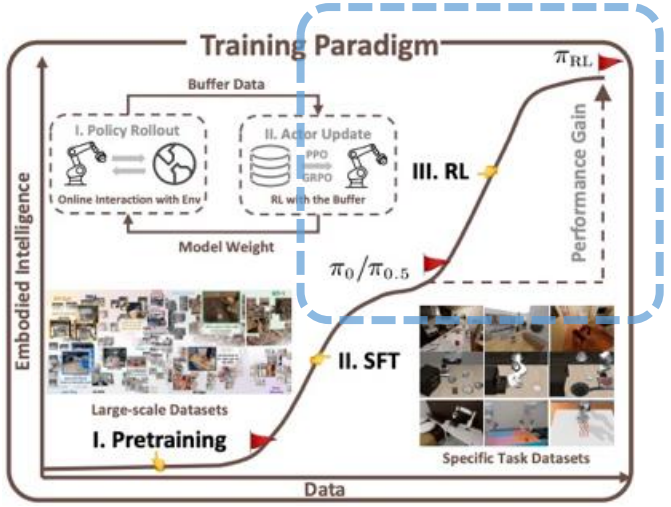


[Zhan*, Tao* et al., MeanFlow Policy, ICLR 2026]

2. Learning from experience (RL Post-Training)



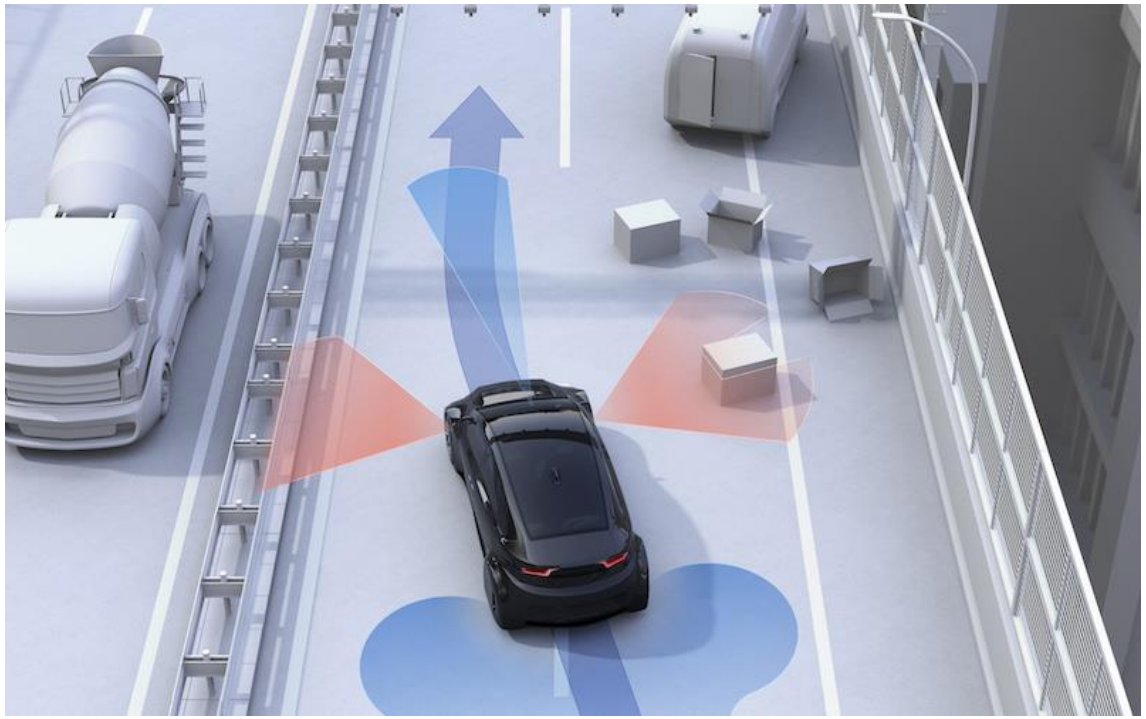
[DeepSeek-AI Team, DeepSeek-R1, *arXiv 2025*]



[Chen* Liu*, Zhang* et al., π_{RL}, *arXiv 2026*]

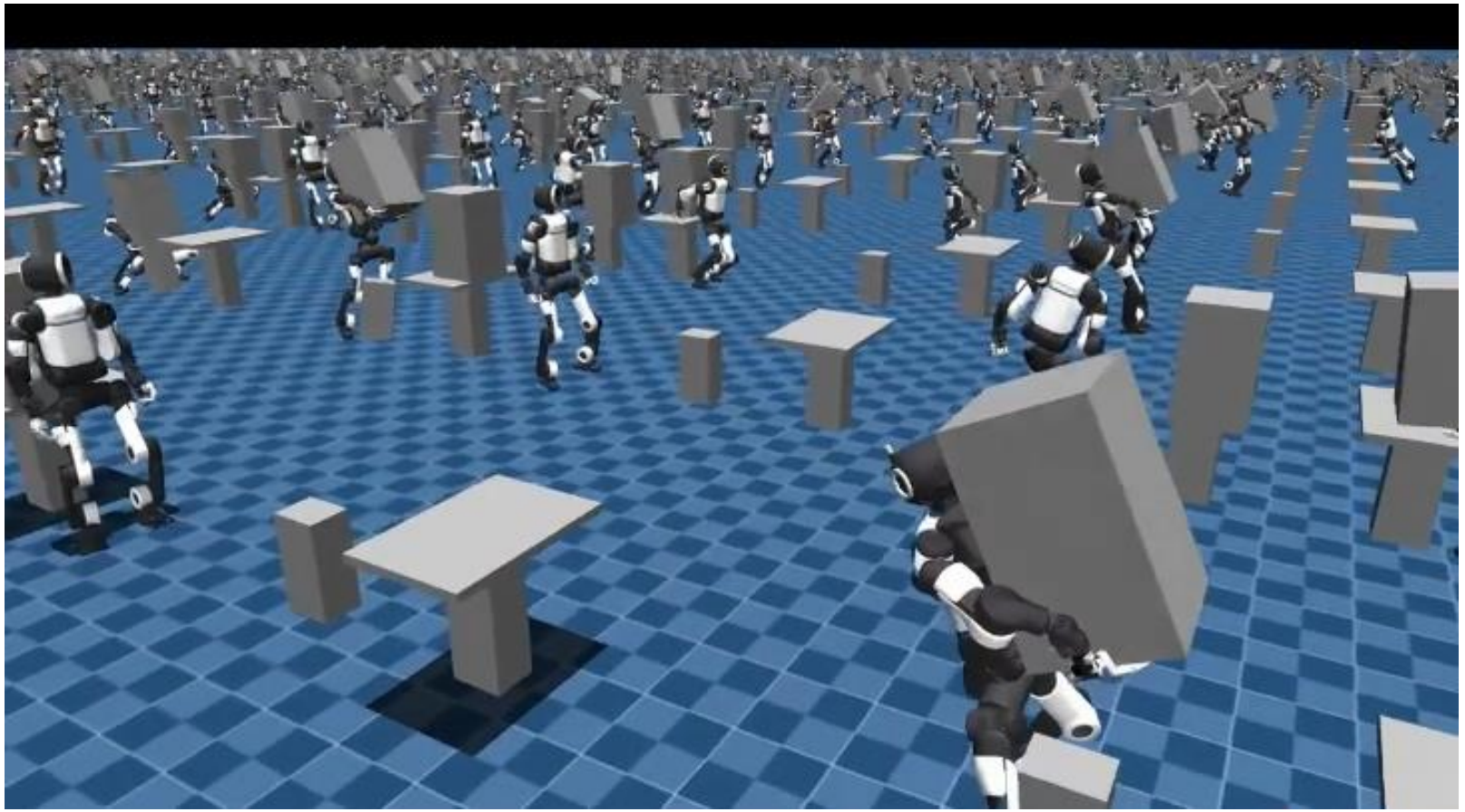
Real-Time Control Demands

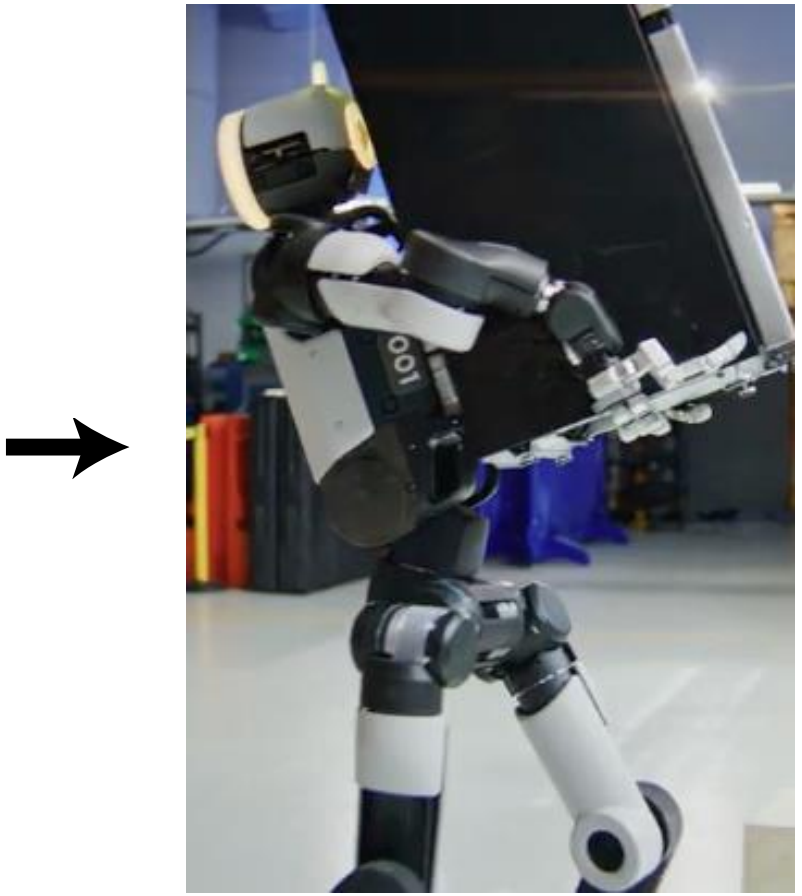
[Physical Intelligence Team, π, 2025]



[https://innotechtoday.com/autonomous-cars/]

Policy Improvement through Experience



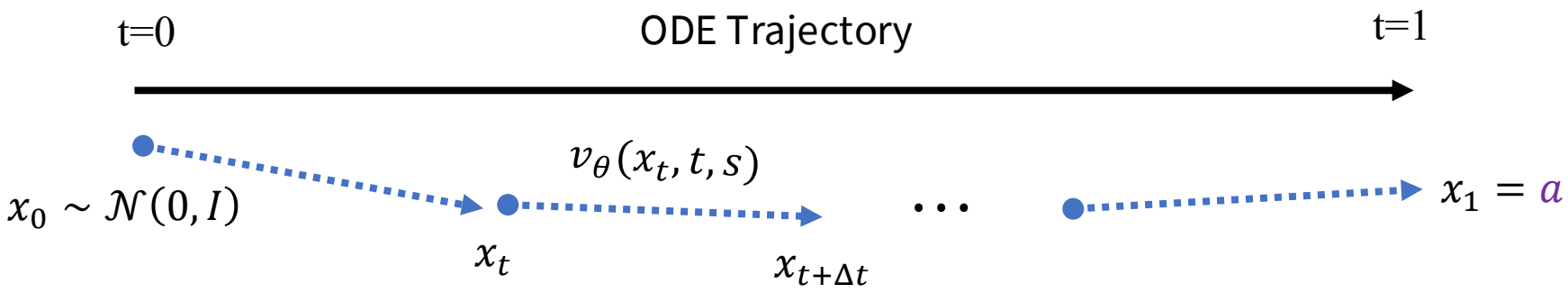


[https://bostondynamics.com/blog/training-a-humanoid-robot-for-hard-work]

Action Generation of Diffusion Policies

Diffusion and flow policies generate actions through ODE trajectories:

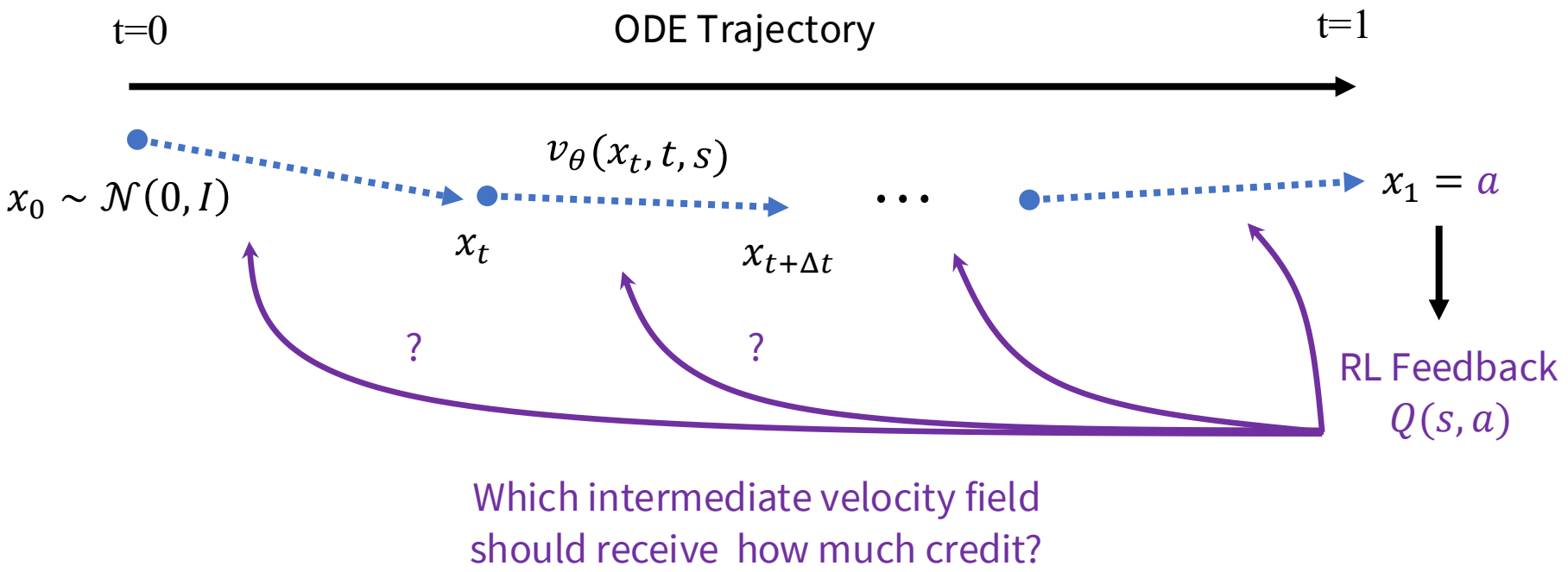
$$a = x_1 = x_0 + \int_0^1 v_\theta(x_t, t, s)\, dt.$$



s: the state
a: action

Output-to-Trajectory Credit Assignment

RL feedback is defined on the final executed action, while diffusion policies need to update all intermediate $v_\theta(x_t, t, s)$.

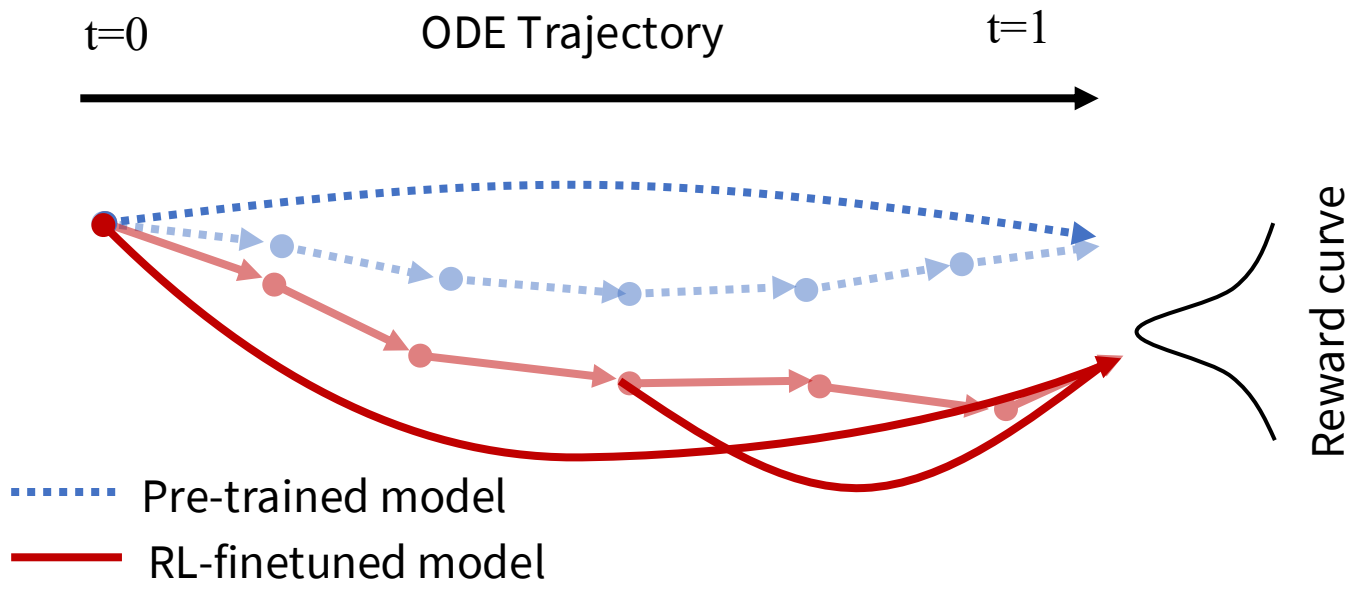


s: the state
a: action

One-Step Diffusion Variants

Important: Few-step variants (MeanFlow, Consistency Models) still have ODE dependence, i.e., all intermediate shortcuts must align with the same endpoint.

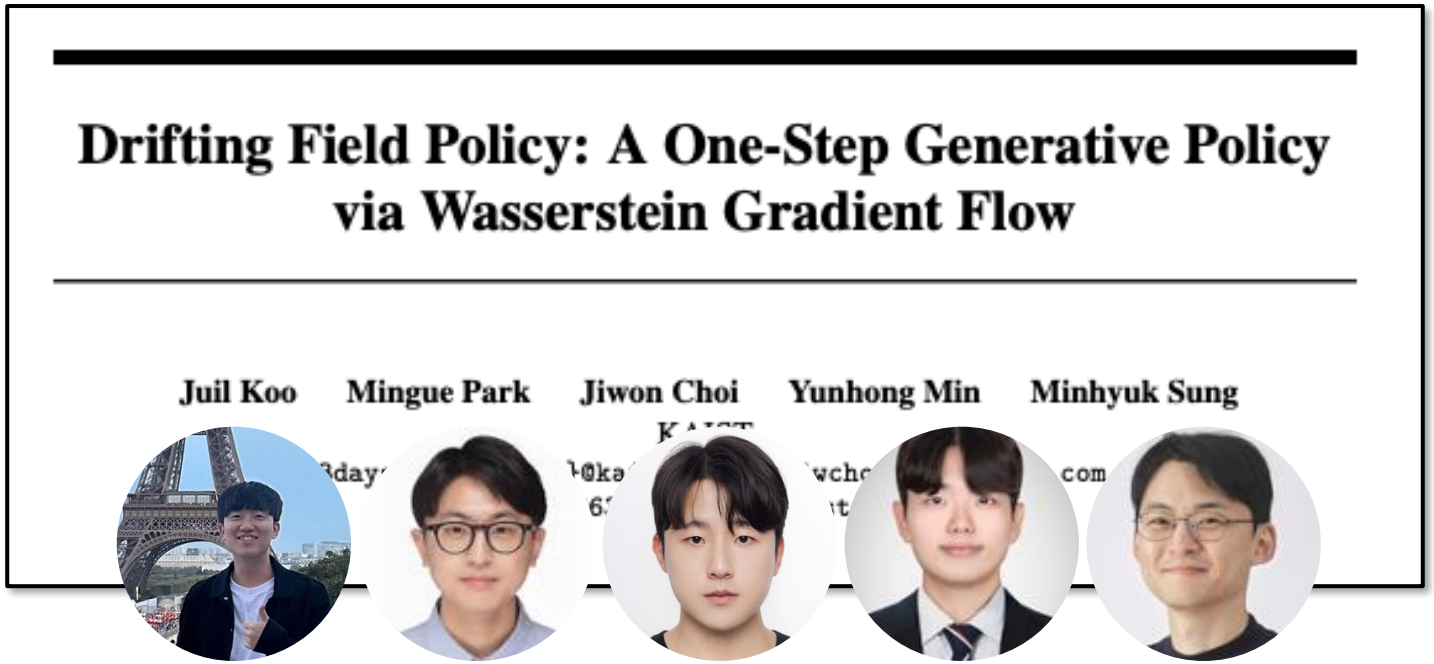
Sampling is one-step, yet training remains trajectory-level.



Drifting Field Policy: One-Step Pushforward Policy

Drifting field policy (DFP) directly parameterizes the action distribution with a single-pass pushforward map f_θ:

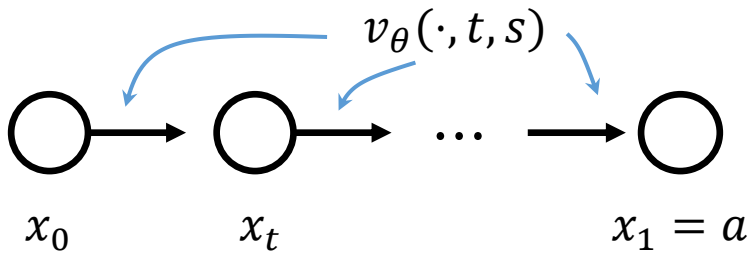
$$a = f_\theta(\epsilon, s), \qquad \epsilon \sim p_\epsilon,$$
$$\pi_\theta(\cdot\,|s) = [f_\theta(\cdot, s)]_{\#} p_\epsilon.$$



s: the state
a: action

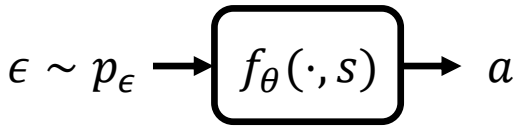
Comparison to Diffusion Policy

Diffusion Policy



ODE-based generation

Drifting Field Policy



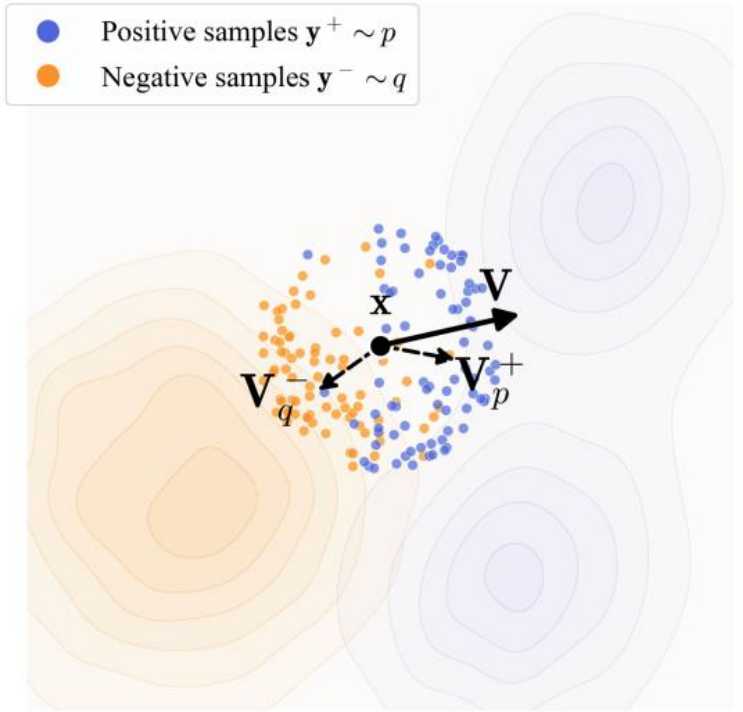
Pushforward map generation

- One-step action generation.
- No ODE trajectory → No output-to-trajectory credit assignment.

Drifting Fields for Distribution Matching

- p: data distribution
- $q \coloneqq [f_\theta]_\# p_\epsilon$: pushforward distribution induced by a single-pass function f_θ

The *drifting field* $\mathbf{V}_{p,q} = \mathbf{V}_p^+ - \mathbf{V}_q^-$ moves generated particles x toward positives p and aw ay from negatives q.



Drifting Field Construction

$$\mathbf{V}_{p,q}(x) = \mathbf{V}_p^+(x) - \mathbf{V}_q^-(x)$$
$$= \frac{\mathbb{E}_{y^+\sim p}[k(x,y^+)(y^+ - x)]}{\mathbb{E}_{y^+\sim p}[k(x,y^+)]} - \frac{\mathbb{E}_{y^-\sim q}[k(x,y^-)(y^- - x)]}{\mathbb{E}_{y^-\sim q}[k(x,y^-)]}$$

$k(x,y) = \exp(-\frac{\|x-y\|^2}{2h^2})$: Gaussian kernel

RL Finetuning Objective

RL finetuning aims to match the reward-tilted target distribution as follows:

$$\pi^{+}(\cdot\,|s) = \arg\max_{\pi} \mathbb{E}_{a\sim\pi(\cdot|s)}\left[Q_{\phi}(s,a)\right] - \alpha D_{\mathrm{KL}}(\pi(\cdot\,|s)\,\|\,\pi_{\mathrm{old}}(\cdot\,|s))$$

$$= \frac{\pi_{\mathrm{old}}(a|s)\exp\left(Q_{\phi}(s,a)/\alpha\right)}{Z(s)}$$

$Q_{\phi}(s,a)$: a critic network that estimates the value of a state-action pair (s,a).
π_{old}: old policy (soft-updated).

Policy Improvement as Distribution Matching

Same training objective, different target distribution.

Distribution matching: $p = p_{\text{data}}$

Policy improvement: $p = \pi^+$ $\quad \boxed{\pi^+(a|s) \propto \pi_{\text{old}}(a|s)\exp(Q_\phi(s,a)/\alpha).}$

$$\mathcal{L}_{\text{drift}}(\theta; p, q) = \mathbb{E}_{\epsilon \sim p_\epsilon}\left[\left\| x - \text{sg}\left(x + \mathbf{V}_{p,q}(x)\right)\right\|^2\right], \text{where } x = f_\theta(\epsilon), \quad \textbf{(Distribution Matching)}$$

$$\mathcal{L}_{\text{PI}}(\theta; \pi^+, \pi_\theta) = \mathbb{E}_{s,\epsilon}\left[\left\| \hat{a} - \text{sg}\left(\underbrace{\hat{a} + \mathbf{V}_{\pi^+,\pi_\theta}(\hat{a})}_{\hat{a}_{\text{target}}}\right)\right\|^2\right], \text{where } \hat{a} = f_\theta(\epsilon, s). \quad \textbf{(Policy Improvement)}$$

[Koo et al., Drifting Field Policy, *arXiv 2026*]

Policy Improvement as Distribution Matching

Same training objective, different target distribution.

Distribution matching: $p = p_{\text{data}}$

Policy improvement: $p = \pi^+$

$$\pi^+(a|s) \propto \pi_{\text{old}}(a|s)\exp(Q_\phi(s,a)/\alpha).$$

$$\mathcal{L}_{\text{drift}}(\theta; p, q) = \mathbb{E}_{\epsilon\sim p_\epsilon}\left[\left\| x - \text{sg}\left(x + \mathbf{V}_{p,q}(x)\right)\right\|^2\right], \text{where } x = f_\theta(\epsilon), \quad \textbf{(Distribution Matching)}$$

$$\mathcal{L}_{\text{PI}}(\theta; \pi^+, \pi_\theta) = \mathbb{E}_{s,\epsilon}\left[\left\| \hat{a} - \text{sg}\left(\hat{a} + \mathbf{V}_{\pi^+,\pi_\theta}(\hat{a})\right)\right\|^2\right], \text{where } \hat{a} = f_\theta(\epsilon, s). \quad \textbf{(Policy Improvement)}$$

DFP directly updates the generated samples toward high-value regions, without output-to-trajectory credit assignment.

[Koo et al., Drifting Field Policy, *arXiv 2026*]

Interpretation of DFP Update

$$\mathbf{V}_{\pi^+,\pi_\theta}(a|s) = h^2\left[\nabla_a \log \pi^+_{\text{kde}}(a|s) - \nabla_a \log \pi_{\theta,\text{kde}}(a|s)\right] \quad (1)$$

$$\simeq \frac{1}{\alpha}\underbrace{\nabla_a Q_\phi(s,a)} + \underbrace{\left(\nabla_a \log \pi_{\text{old}}(a|s) - \nabla_a \log \pi_\theta(a|s)\right)} \quad (2)$$

$\nabla_a Q_\phi$ ascent

Trust region around π_{old} via score matching

It updates the particle toward high-value regions, while preventing it from deviating too f ar from π_{old}.

(1): By the drifting field / score matching connection.
(2): By $\nabla \log \pi^+ = \frac{1}{\alpha}\nabla Q + \nabla \log \pi_{\text{old}}$

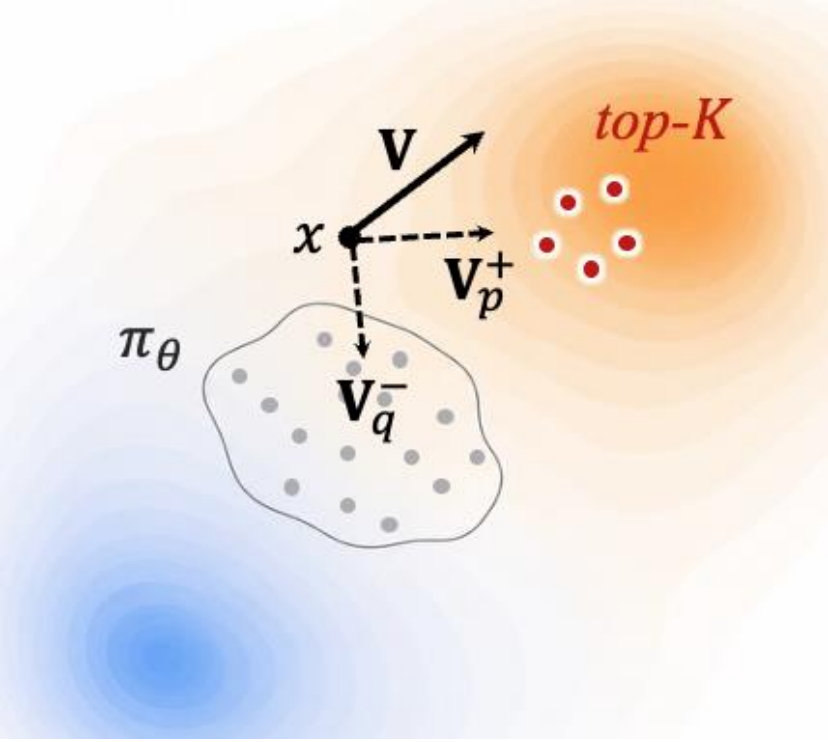
[Koo et al., Drifting Field Policy, *arXiv 2026*]

How to Approximate π^+

$$\pi^+(\cdot\,|s) = \frac{\pi_{\text{old}}(a|s)\exp\big(Q_\phi(s,a)/\alpha\big)}{Z(s)}$$

We approximate the ideal target policy π^+ with an empirical *top-K* samples from π_{old}:

$$P_K(s) \coloneqq \text{TopK}_j\, Q_\phi\big(s, a^{(j)}\big), \qquad a^{(1)}, \ldots, a^{(N)} \sim \pi_{\text{old}}(\cdot\,|s).$$



Practical Approximation

1. Sample N candidates from $\pi_{\text{old}}(a|s)$.
2. Score candidates with Q_ϕ.
3. Select the top-K high-value actions.
4. Update π_θ toward the top-K region.

$$\mathcal{L}_{\text{top-K}}(\theta; P_K, \pi_\theta) = \mathbb{E}_{s,\epsilon}\left[\left\|\hat{a} - \text{sg}\left(\hat{a} + \mathbf{V}_{P_K,\pi_\theta}(\hat{a})\right)\right\|^2\right]$$

[Koo et al., Drifting Field Policy, *arXiv 2026*]

DFP Algorithm

Algorithm 1 Drifting Field Policy (DFP), RL fine-tuning

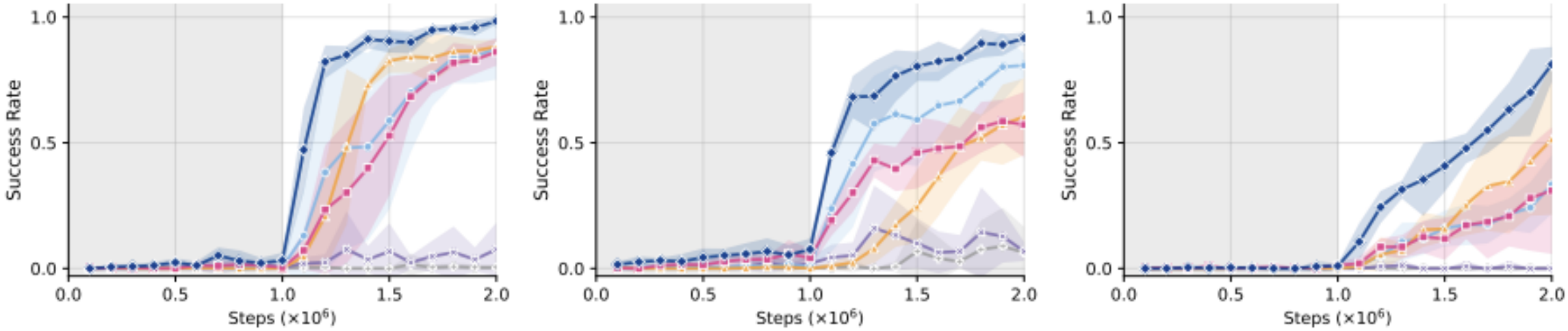
Input: BC-pretrained policy $\pi_\theta(\cdot|s) = [f_\theta(\cdot, s)]_{\#}p_\epsilon$ and critic Q_ϕ
Initialize old policy $\pi_{\text{old}} \leftarrow \pi_\theta$
for RL finetuning step **do**
  Update θ by minimizing $\mathcal{L}_{\text{BC}}(\theta) + \lambda\,\mathcal{L}_{\text{top-}K}(\theta)$
  Update ϕ via the Bellman backup
  Update old policy: $\theta_{\text{old}} \leftarrow \tau\,\theta + (1-\tau)\,\theta_{\text{old}}$
end for

$\mathcal{L}_{BC}$: Behavior cloning regularization with stored data.

[Koo et al., Drifting Field Policy, *arXiv 2026*]

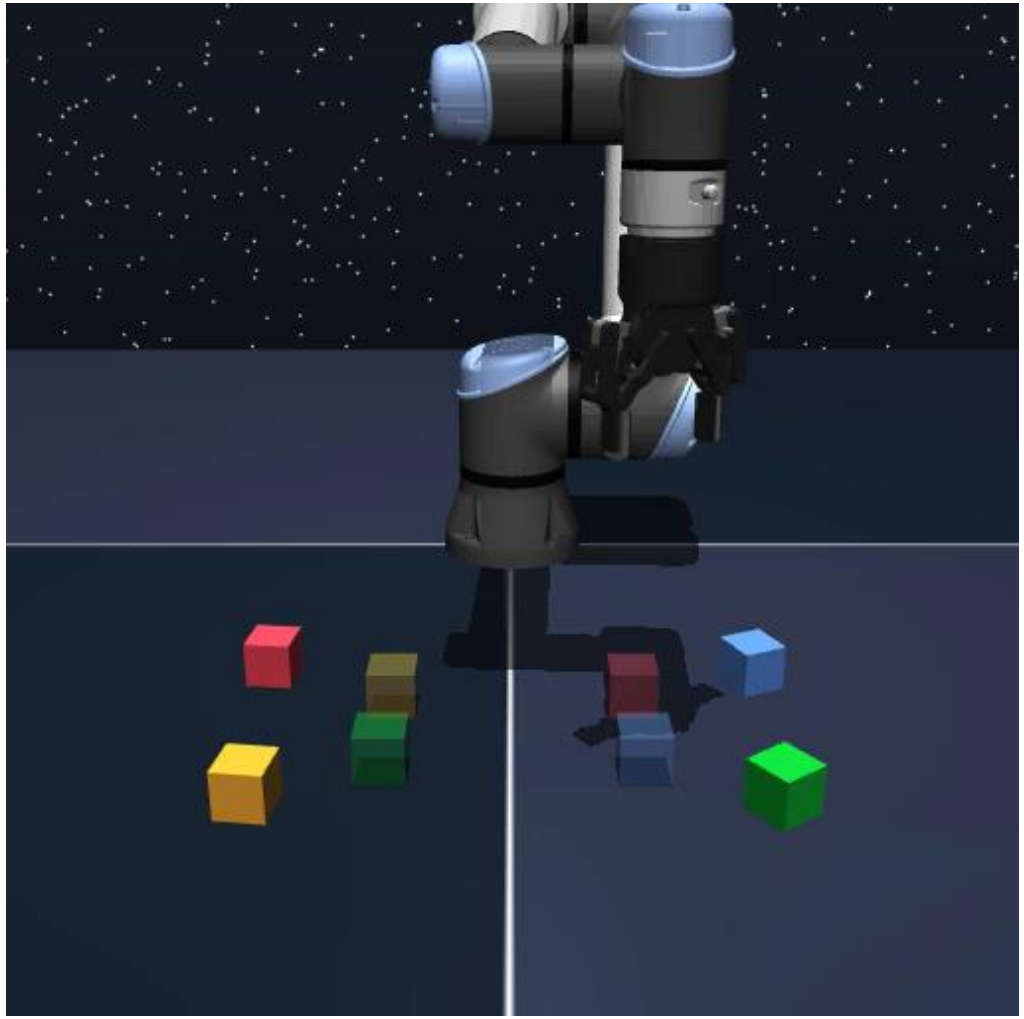
Experiment Results

Comparison to Diffusion Policies

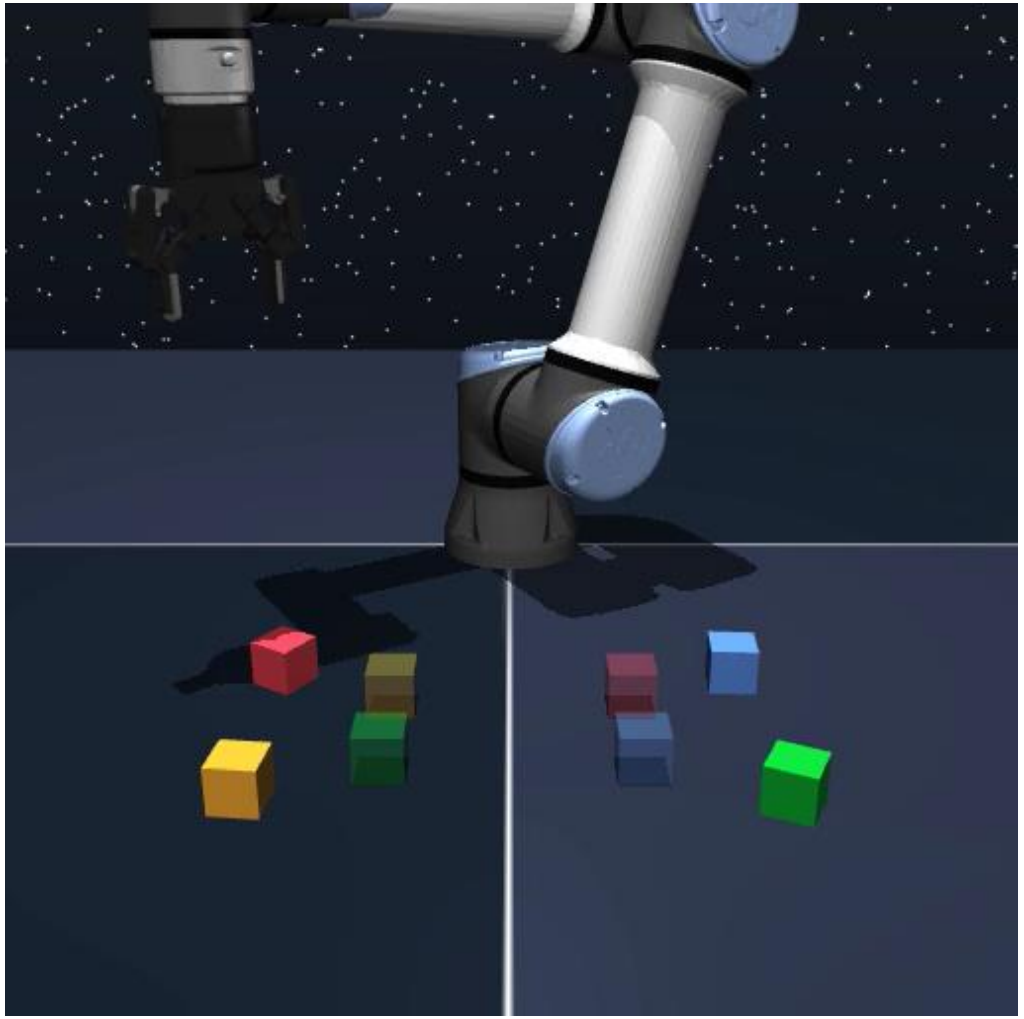


[Koo et al., Drifting Field Policy, *arXiv 2026*]

Qualitative Results



MVP (MeanFlow-Based)



DFP (Ours)

[Koo et al., Drifting Field Policy, *arXiv 2026*]

Top-*K*-*of*-*N* Ablation Study

With $N = 16$, DFP is robust across K, with the best performance at $K = 4$.

All DFP variants outperform MVP (80.4 avg.).

	Robo.	**Cube-2.**	**Cube-3.**	**Cube-4.**	**Avg.**
$K = 1$	<u>95.0</u>	<u>99.7</u>	54.5	94.2	85.8
$K = 2$	**94.6**	**100.0**	<u>76.2</u>	<u>96.8</u>	<u>91.9</u>
$K = 4$	93.9	<u>99.7</u>	**90.4**	**98.4**	**95.6**
$K = 8$	88.6	99.8	85.5	96.2	92.5

[Koo et al., Drifting Field Policy, *arXiv 2026*]

Conclusion

- Action experts require:
    1. Multimodal action generation
    2. Real-time inference
    3. Policy improvement through RL
- Diffusion policies have made progress, but a mismatch remains in RL training: action-level feedback and their ODE-based parameterization.
- DFP removes ODE dependency with a single-pass pushforward policy:
    1. One-step generation.
    2. Direct updates of generated actions toward high-value regions.
- Can drifting models become a general alternative for RL finetuning generative models?
    1. How can we scale drifting models to high-dimensional outputs?
    2. How can we leverage pre-trained diffusion models?

Thank You