ETH zürich



Offline RL, Vision–Language–Action Models,

Reward Models and World Models

Denis Tarasov · Real World Robotics
9 November 2026

SoftRobotics Laboratory

Offline RL: the setting



Train from fixed data, with no new robot interaction

$$\mathcal{D} = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^{N}$$

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right]$$

Logged transitions

s, a, r, s′: state, action, reward, next state

Expected return

γ: discount factor
τ: policy trajectory

Offline RL learns from suboptimal actions and failures.

Behavior cloning copies them, which can reduce performance.

Offline RL / Levine et al. 2020

Offline RL: extrapolation error



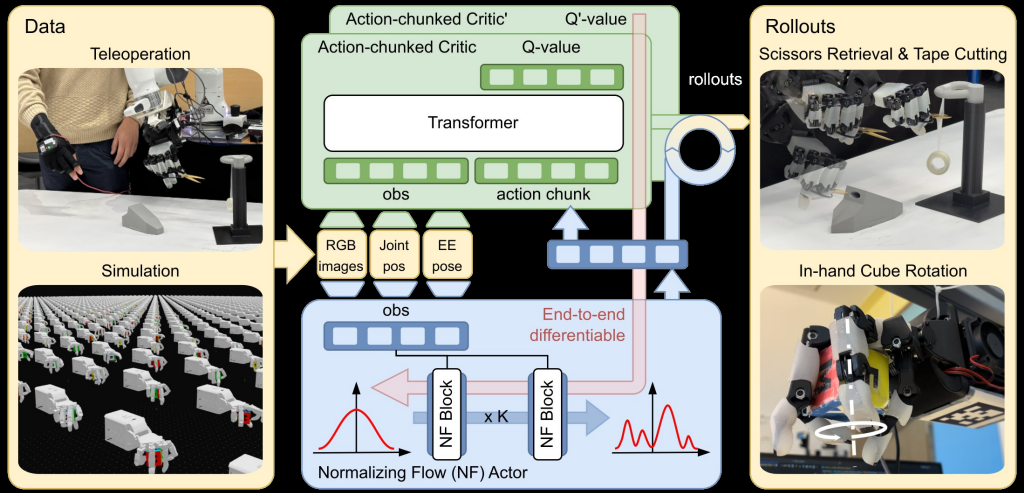
Q-values for unseen actions can be wrong.
Maximization can exploit those errors.

TD3+BC	Keep the policy close to dataset actions
CQL	Penalize high values for unsupported actions
IQL	Learn implicit regularization without OOD sampling

TD3+BC / CQL / IQL

SERNF: sample-efficient robot fine-tuning

Improve a pretrained policy with few real-world rollouts



Reuse each rollout

Off-policy updates from successes and failures

Normalizing-flow actor

Exact likelihood for conservative updates

Chunk-level critic

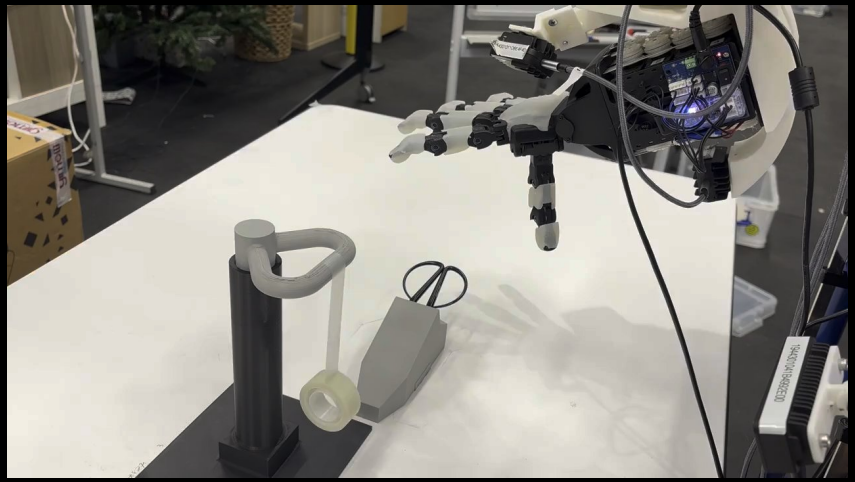
Credit over action sequences

Yang, Tarasov et al. 2026

SERNF: gains with limited robot data

16% to 84% cutting success

60 additional robot rollouts

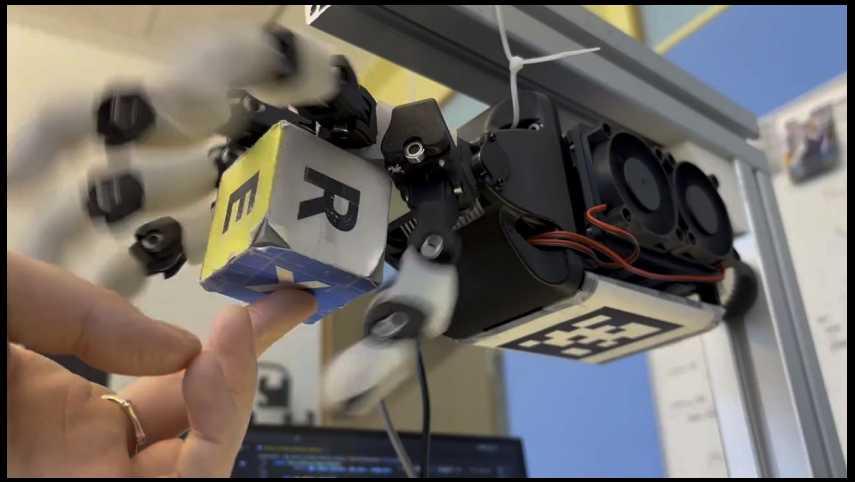


Scissors retrieval and tape cutting

121 initial trajectories (71 successful)

6.25 rotations per minute

105 min of real-world data



Palm-down cube rotation

Initialized from a simulation teacher

Three learned components



Policy	$\pi_\theta(A_t \mid o_{\leq t}, \ell)$	**Choose motor commands** Move the marker above the cup
Reward model	$r_\phi(o_{\leq t}, \ell)$	**Score the observed result** Did the marker end up in the cup?
World model	$p_\psi(o_{t+1:t+H} \mid o_{\leq t}, A_t)$	**Predict the action's outcome** Where would the marker land?

$o_{\leq t}$: observation history $\quad \ell$: instruction $\quad A_t$: action chunk $\quad H$: horizon

π0 / RoboReward / V-JEPA 2



Part 1

Vision–Language–Action Models

The foundation-model motivation



Language models

Larger models + more diverse data

One pretrained model transfers to many language tasks.

Robot foundation models

Scale models and robot experience

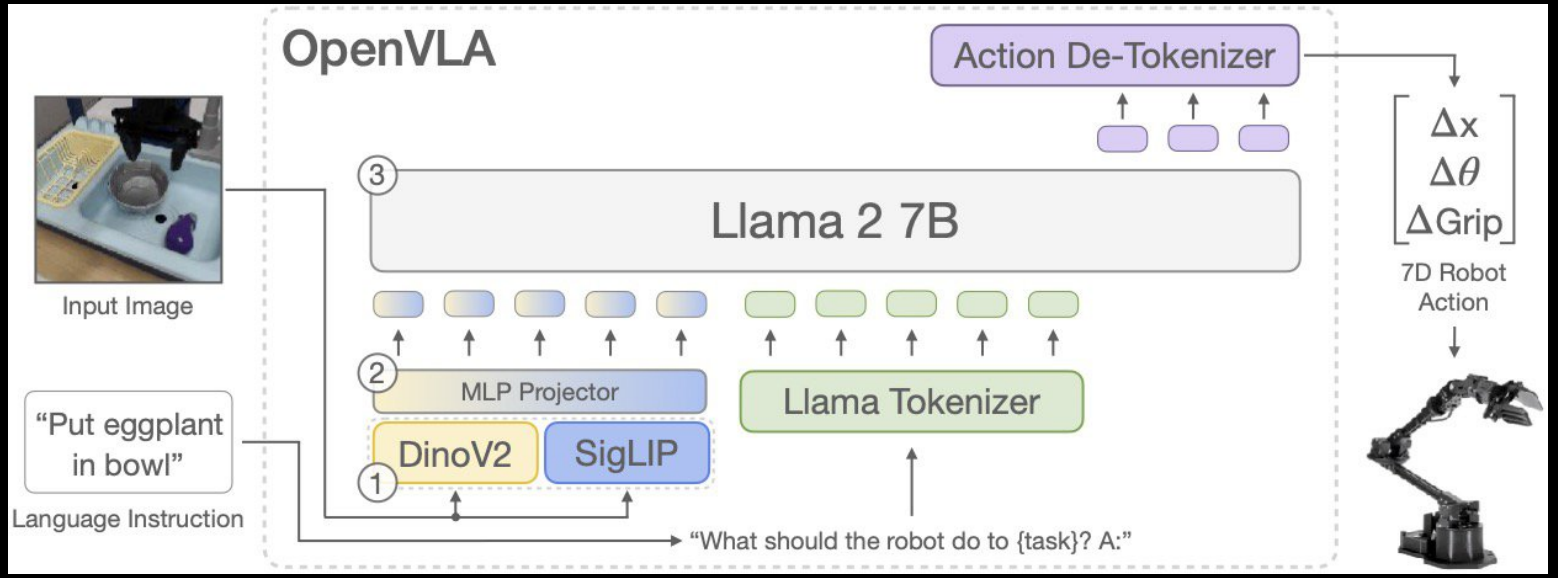
Aim for one policy that transfers across tasks, objects and robots.

The goal: reuse broad experience instead of collecting a new dataset for every task.

Robot data, embodiment differences and physical feedback remain limiting factors.

Scaling Laws for Neural Language Models / RT-2 / π0

A VLA predicts robot actions



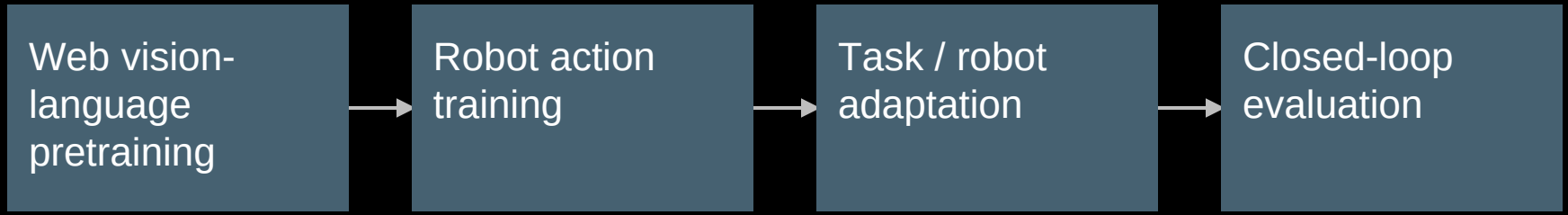
Images + instruction + model-specific robot state

Action output still needs units, frames and a controller

OpenVLA

Data determines the transfer problem





Web data teaches object and language understanding.

Robot demonstrations teach which motor commands move those objects.

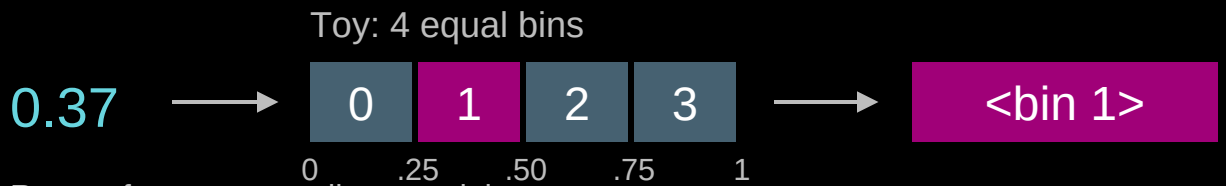
Deployment still requires matching joint units, camera views and control rate.

π0 / π0.5 / OpenVLA

Three ways to represent actions

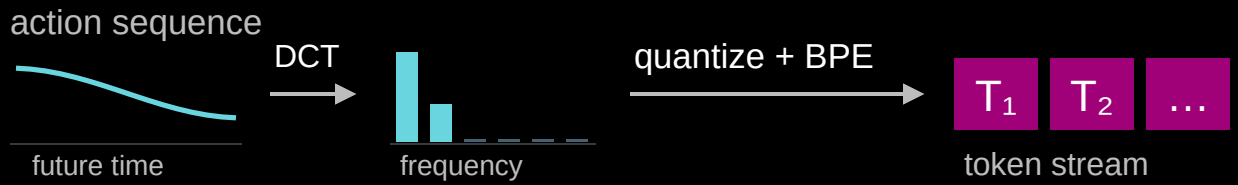
Scalar action tokens

One value at a time



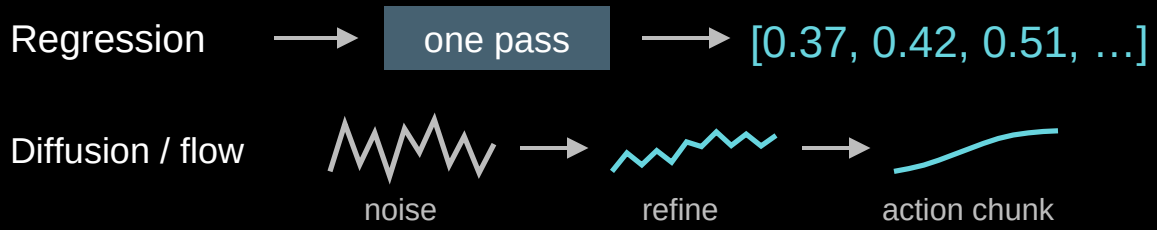
Compressed tokens (FAST)

Whole action chunk



Continuous action head

Real-valued outputs



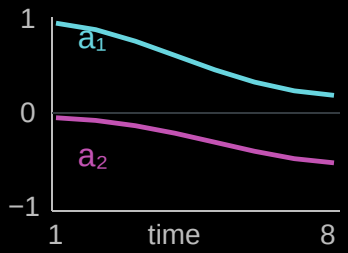
Token decoding and iterative sampling are different loops.

OpenVLA / FAST / OpenVLA-OFT / Diffusion Policy

FAST: compressing an action chunk

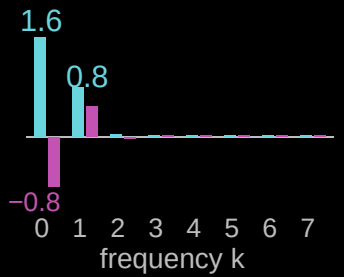
Toy example: 8 timesteps × 2 coordinates. BPE merges shown are illustrative.

1 Action chunk



Per-coordinate normalization

2 DCT in time



Smooth motion: mostly low k

3 Quantize

$$Q = \mathrm{round}(10C)$$

k	a_1	a_2
0	16	−8
1	8	5
2	0	0
3–7	0	0

$$0.04 \times 10 = 0.4 \mapsto 0$$

4 Flatten + BPE

All coordinates at k = 0 first

[16, −8, 8, 5, 0, 0, …, 0]

12 zeros

A B Z Z Z

A = [16, −8] B = [8, 5]

Z = [0, 0, 0, 0]

Inference: the VLA predicts tokens, then the decoder recovers the chunk

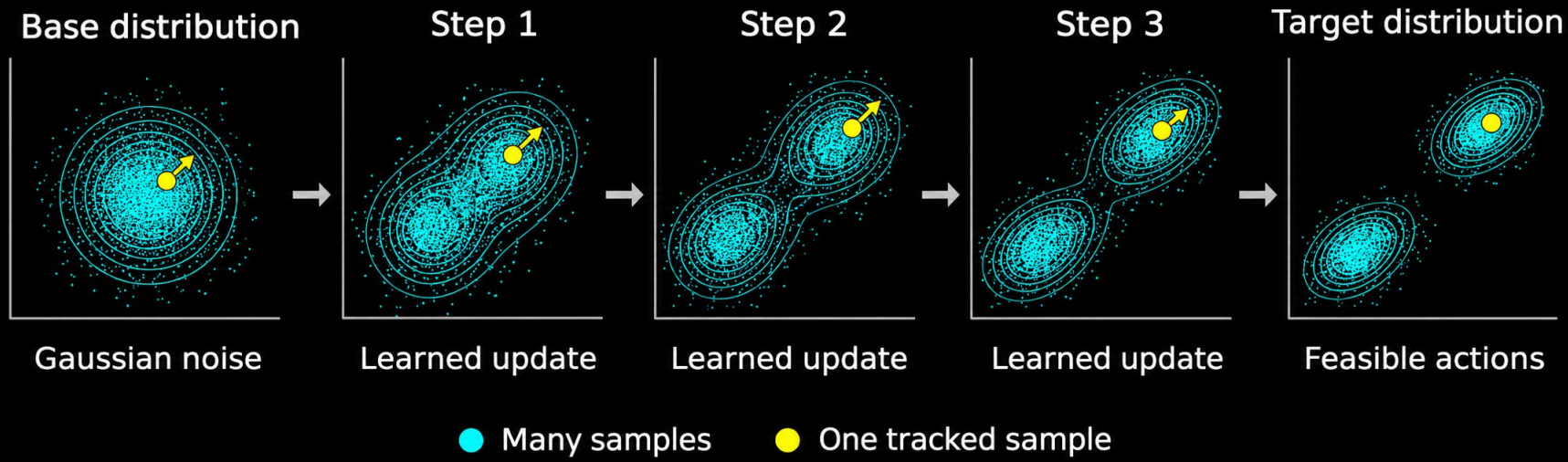
Image + instruction → VLA next-token model → A B Z Z Z → BPE decode ÷10, inverse DCT →  actions

Rounding loses precision. BPE is lossless and shortens the sequence.

FAST

Flow matching transports noise into actions

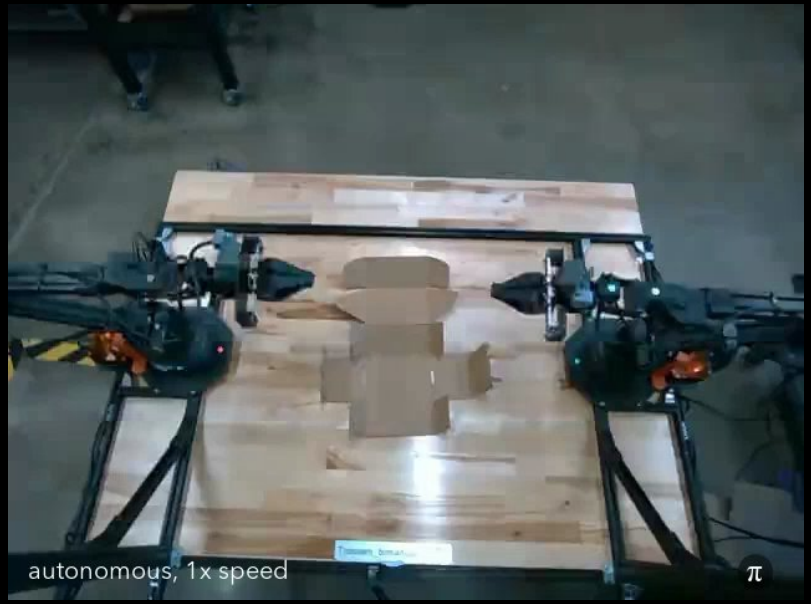
A learned velocity field updates the sample at each sampling step.



$$x_0 \sim \mathcal{N}(0, I), \qquad x_{\tau+\Delta\tau} = x_\tau + \Delta\tau\, v_\theta(x_\tau, \tau, c)$$

c = observation + instruction
Yellow arrows: learned velocity

π_0: a VLM with a continuous action expert



Pretrained 3B vision-language backbone

Flow-matching action expert produces continuous chunks

Broad robot data + task-specific post-training

October 2024. The key advance was the combination of architecture, data and dexterity.

π0 paper / π0 project and videos / RT-2

$\pi_{0.5}$: generalization to unseen homes

π_0 already supplied the VLM backbone and flow-matching action expert

Training additions

Broader supervision

Robot demonstrations + web data
+ next-subtask labels

Two-stage training

FAST-token pretraining,
then add the continuous action expert

One model, two levels

Image + “clean the bedroom”

↓

Predict: “pick up the pillow”

↓

Flow expert generates an action chunk

Evidence: household tasks in homes absent from training

$\pi_{0.5}$ paper / official project page (April 2025)

$\pi_{0.5}$ demo: “clean the bedroom”



Held-out home

Scene excluded from training data

Autonomous rollout

Official video at 5× speed

Watch the sequence of object interactions

$\pi_{0.5}$ paper / official project page (April 2025)

$\pi_{0.6}$ and $\pi^*_{0.6}$: learning from experience



$\pi_{0.6}$: supervised VLA
4B VLM + 860M action expert

$\pi^*_{0.6}$: RECAP adds RL

Autonomous trials, reward feedback and human corrections

Source clip: a human intervenes to correct the robot during data collection.

π*0.6 and RECAP / RECAP paper

Why predict beyond the next 10 actions?

Predict 50 actions → Execute the first 10 → Observe and replan



10 actions used now

40 actions describe a possible continuation

Coherent motion

A longer sequence can capture one consistent movement strategy.

Fresh feedback

Executing a short prefix lets the robot adapt to its next observation.

Overlap for RTC

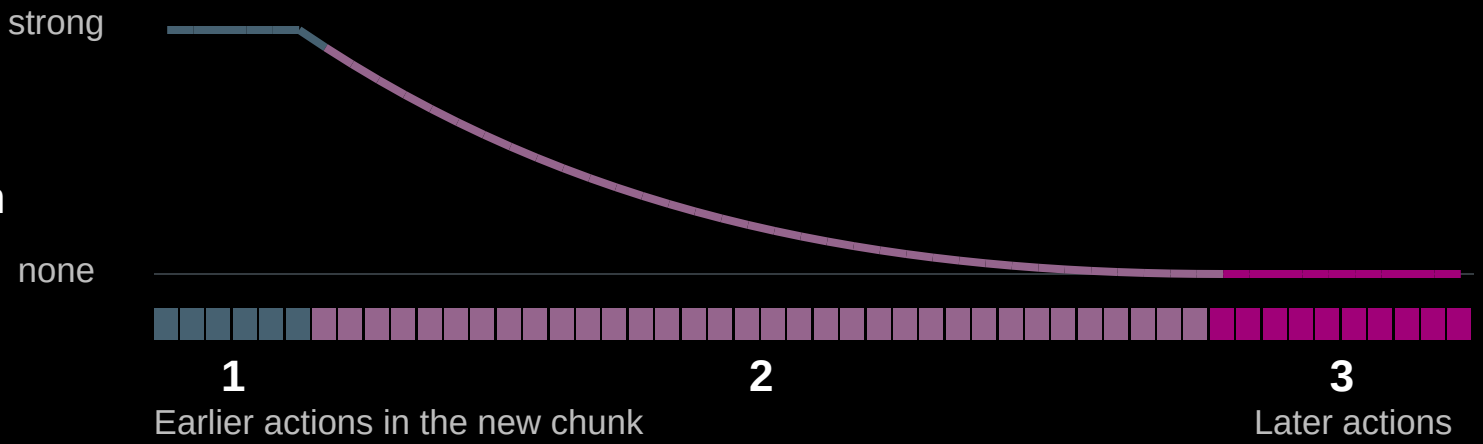
Unused future actions guide the next chunk and its handover.

Predicting only 10 is valid. The 50/10 choice is illustrative and must be tuned.

Diffusion Policy, §§2.3, 4.3 and Fig. 5 / RTC, §4.2

RTC: moving while the model predicts

The robot follows the old plan while the next chunk is generated



1 Committed actions

Already executing during inference

2 Soft masking

Gradually weaken the pull toward old future actions

3 New continuation

No old-plan constraint beyond its endpoint

Black et al., Real-Time Execution of Action Chunking Flow Policies (2025), §4.2 / Fig. 3

RTC in a contact-rich task



Watch the chunk transitions and pauses

Mechanism: committed prefix + adaptable suffix

Original comparison, 1× speed

RTC

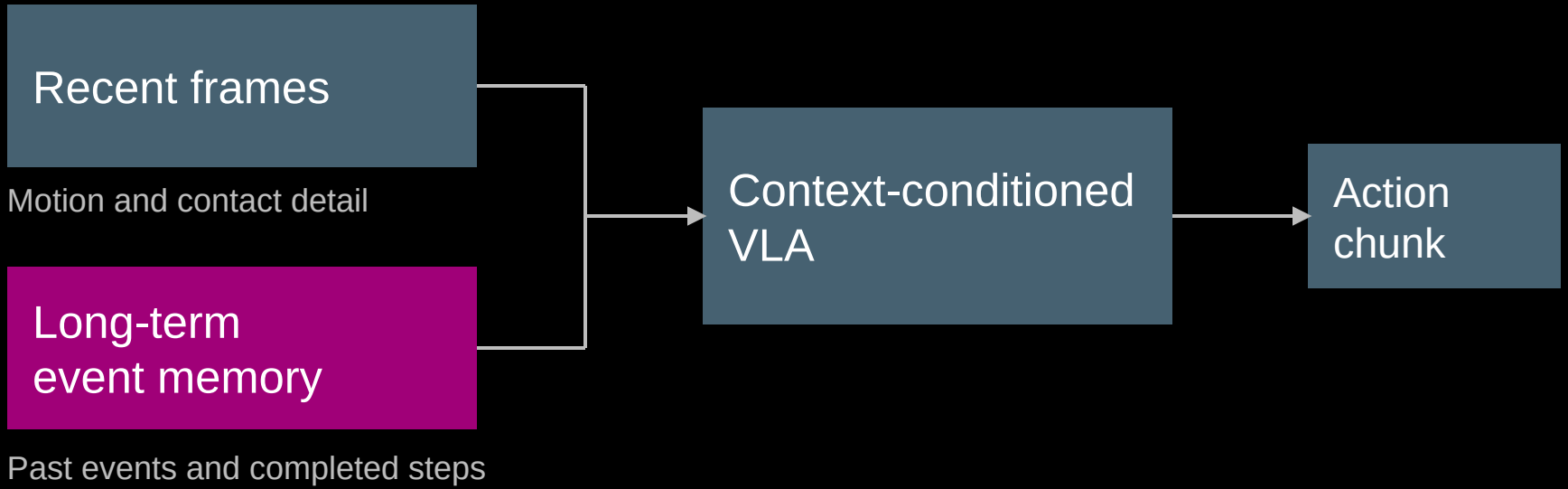
Selected VLA directions in 2026



Model / report	Direction	Evidence boundary
π0.7 · Apr 2026	Steerable, multimodal conditioning	Technical report and author evaluations
GR00T N1.7 · 2026	Reasoning VLA + humanoid workflow	Released model / technical documentation
Gemini Robotics 2 · Jul 2026	Whole-body control and dexterity	Company report; access differs by model
GEN-1.5 · Aug 2026	In-context physical demonstrations	Company report; short-horizon tasks

π0.7 / Isaac GR00T N1.7 / Gemini Robotics 2 / GEN-1.5

Memory addresses partial observability



Both histories condition the next action chunk.

MEM / GEN-1.5

$\pi_{0.7}$: composing learned skills for new tasks

Detailed conditioning makes diverse, mixed-quality data useful

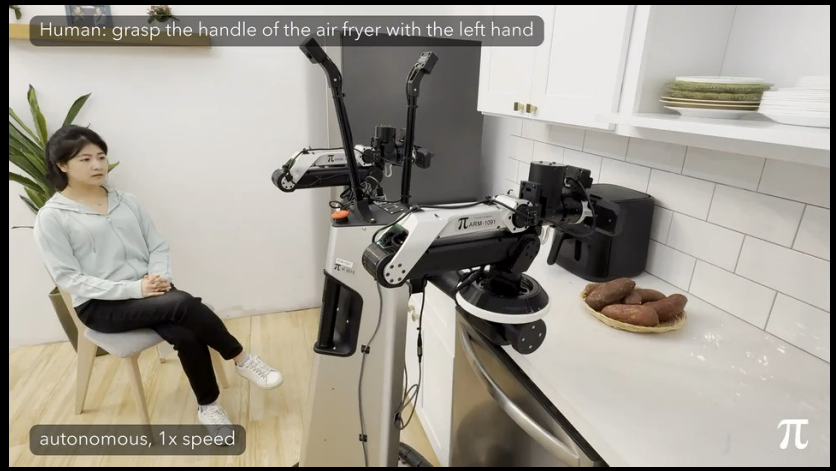
Learn from varied data

Language, quality labels and visual subgoals distinguish how a task was performed.

Transfer across robots

80% shirt folding on UR5e using generated visual subgoals

No folding demonstrations on that robot



Human coaching (source speed varies)

Coaching logs train a high-level policy for later autonomous execution

$\pi_{0.7}$ technical report (2026), Figs. 12, 16, 18 / official coaching demo



Part 2

Reward Models

Reward, progress and value

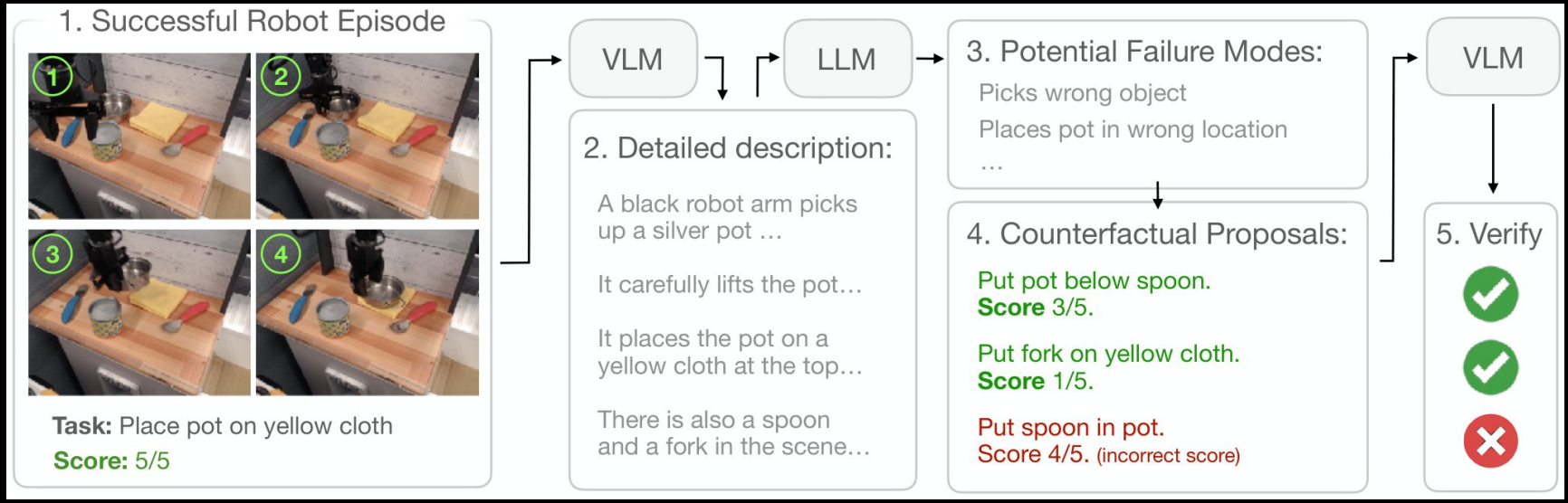


Quantity	Prediction target	Meaning
Reward r	Chosen score for a transition or episode	Defines what the policy optimizes
Progress Φ	Task completion so far	Can be used directly as reward
Value Vπ	Expected future cumulative reward	Depends on the policy and reward

Progress is one possible reward target.
Inputs depend on the method and the information the task requires.

RoboReward (§3) / PROGRESSOR / π*0.6–RECAP (§IV-A)

RoboReward learns coarse task progress



Five-level progress supervision

Counterfactual instructions and clipped trajectories add negatives

RoboReward

Training a learned reward



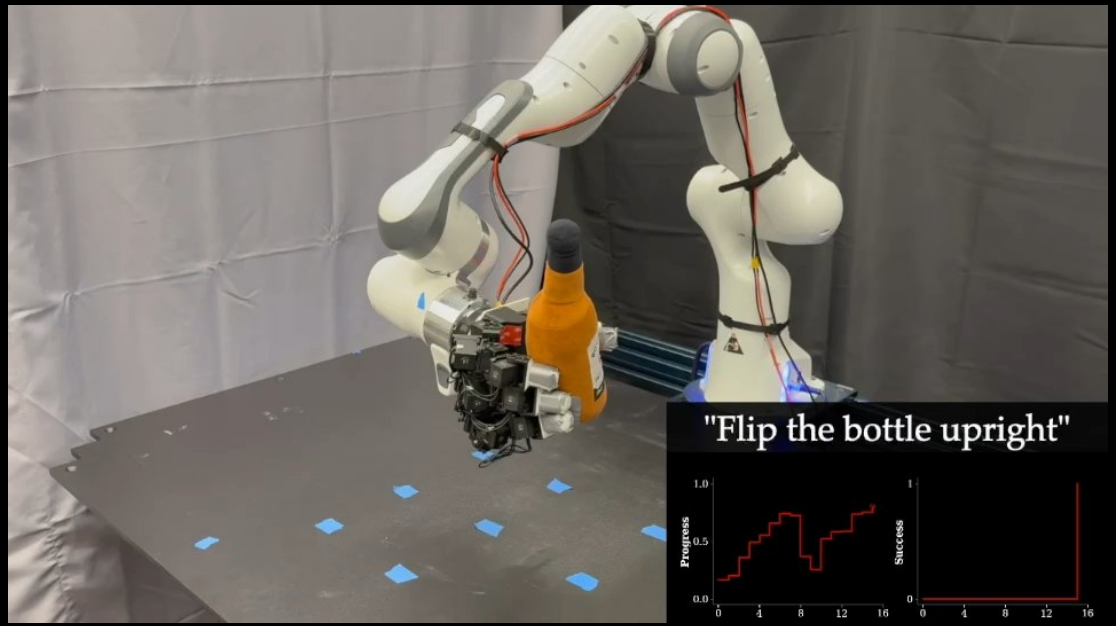
$$P(\tau_A \succ \tau_B \mid \ell) = \sigma(S_\phi(\tau_A, \ell) - S_\phi(\tau_B, \ell))$$

Supervision: success, progress or pairwise preference

Coverage: expert, failed, interrupted and wrong-task episodes

Validation: held-out labels and downstream policy learning

Robometer combines progress and preference



Within one trajectory

Estimate progress and success as the task unfolds.

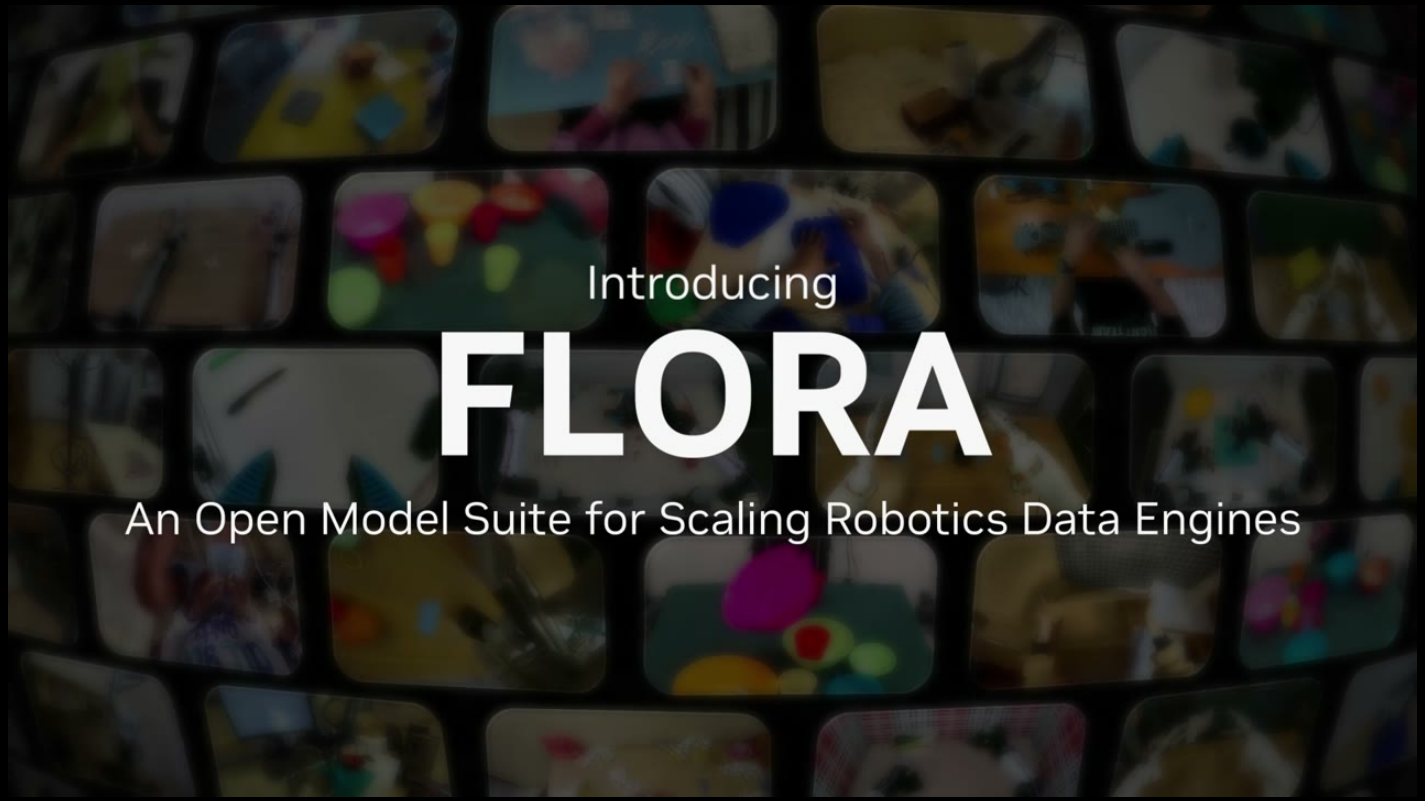
Between trajectories

Learn which attempt makes better progress.

Learned progress scores track how the observed task unfolds.

Robometer

FLORA: automated labels for robot videos



NVIDIA FLORA / Official overview video (67 seconds, original speed) / October 2026

Reward errors become optimization targets

Visible near miss
Predicted score: 0.9

Occluded success
Predicted score: 0.8

Illustrative reward-model mistake

A policy optimizer can exploit the mistaken ordering

Test false positives, failures and policy-induced distribution shift

The important error is the one the policy can repeatedly exploit

RoboReward / Robometer

RECAP: value and advantage estimation



1 Fit a separate V model to completed rollouts

Use demonstrations, autonomous attempts and corrections.

$$G_t = \sum_{k=t}^{T} r_k, \qquad V_\phi(o_t, \ell) \approx \mathbb{E}[G_t \mid o_t, \ell]$$

V predicts remaining return for the behavior mixture in the dataset.

2 Score recorded actions against that baseline

$$\widehat{A}_t = \underbrace{\sum_{k=0}^{N-1} r_{t+k}}_{\text{observed rewards}} + \underbrace{V_\phi(o_{t+N}, \ell)}_{\text{expected future return}} - \underbrace{V_\phi(o_t, \ell)}_{\text{starting baseline}}$$

π*0.6 / RECAP, §IV-A, §V-C and Appendix F

RECAP: improving the policy with advantage labels



3 Train the VLA on both advantage labels

$$I_t = \mathbf{1}[\widehat{A}_t > \epsilon_\ell]$$

Above threshold: “Advantage: positive”
Otherwise: “Advantage: negative”

Predict each recorded action chunk using its label as an input.

$$\min_\theta \ \mathbb{E}_{\mathcal{D}}[\mathcal{L}_{\text{FAST}}(\theta; o_t, \ell, I_t, a) + \lambda \mathcal{L}_{\text{FM}}(\theta; o_t, \ell, I_t, a)]$$

Action loss: FAST token cross-entropy + continuous-action flow matching

4 At deployment, request positive advantage

$$a_{t:t+H} \sim \pi_\theta(\cdot \mid o_t, \ell, I_t = 1)$$

The VLA directly generates actions.
V is used to make training labels.

Collect more rollouts, refit V, relabel the data, and train the policy again.

π*0.6 / RECAP, §IV-B, Eq. 3–4, §V-B and Appendix F



Part 3

World Models

Simulation, planning and imagination training

Video simulation

Predict observations
for supplied actions.

Example: DreamDojo

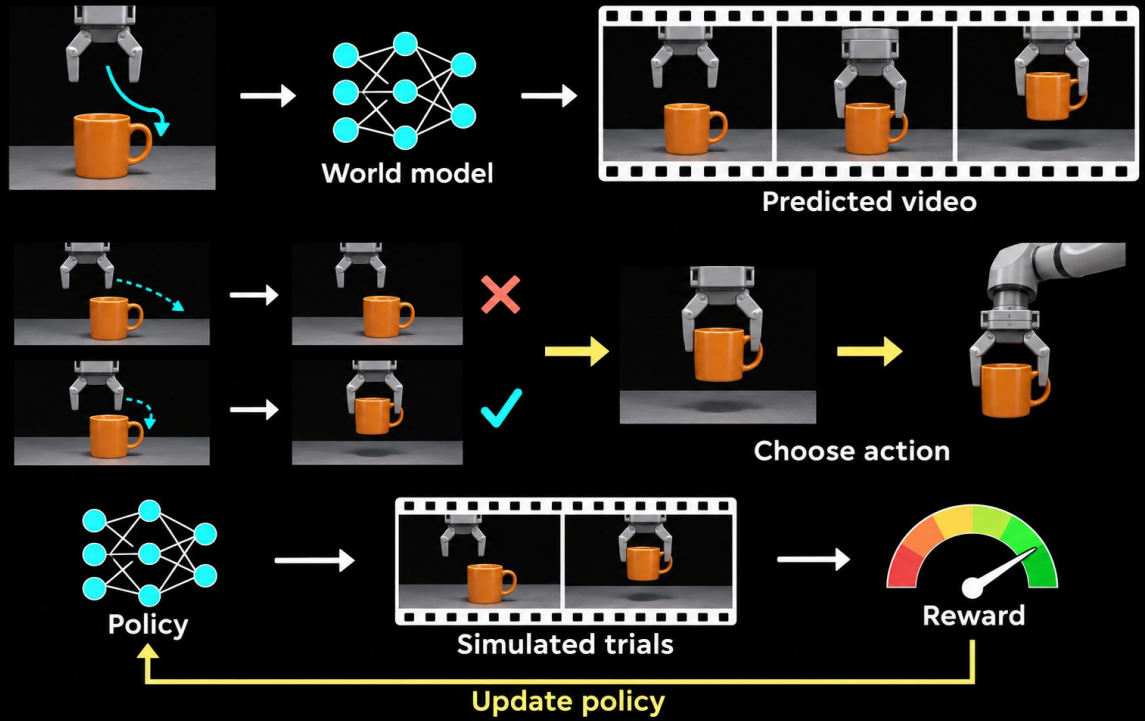
Planning

Compare predicted outcomes.
Execute the best candidate.

Policy weights stay fixed

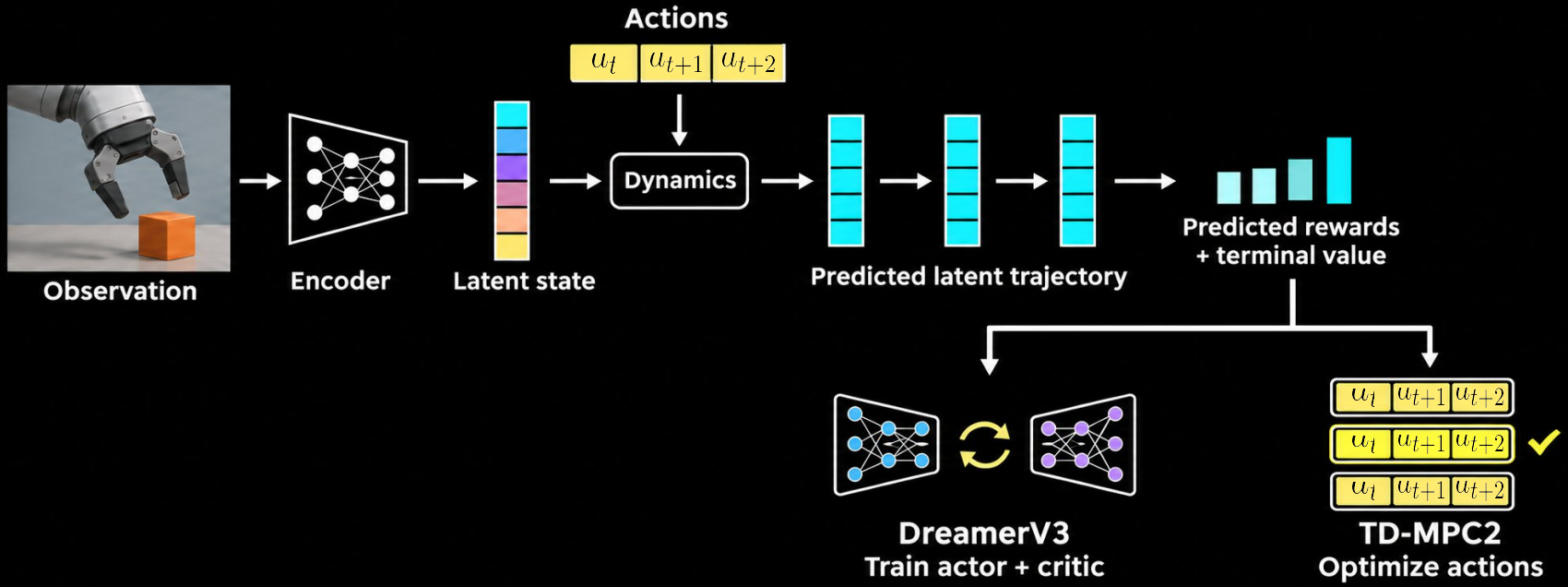
Imagination training

Use simulated returns
to update the policy.



A video simulator can provide experience for both planning and policy training.

Latent dynamics can support control



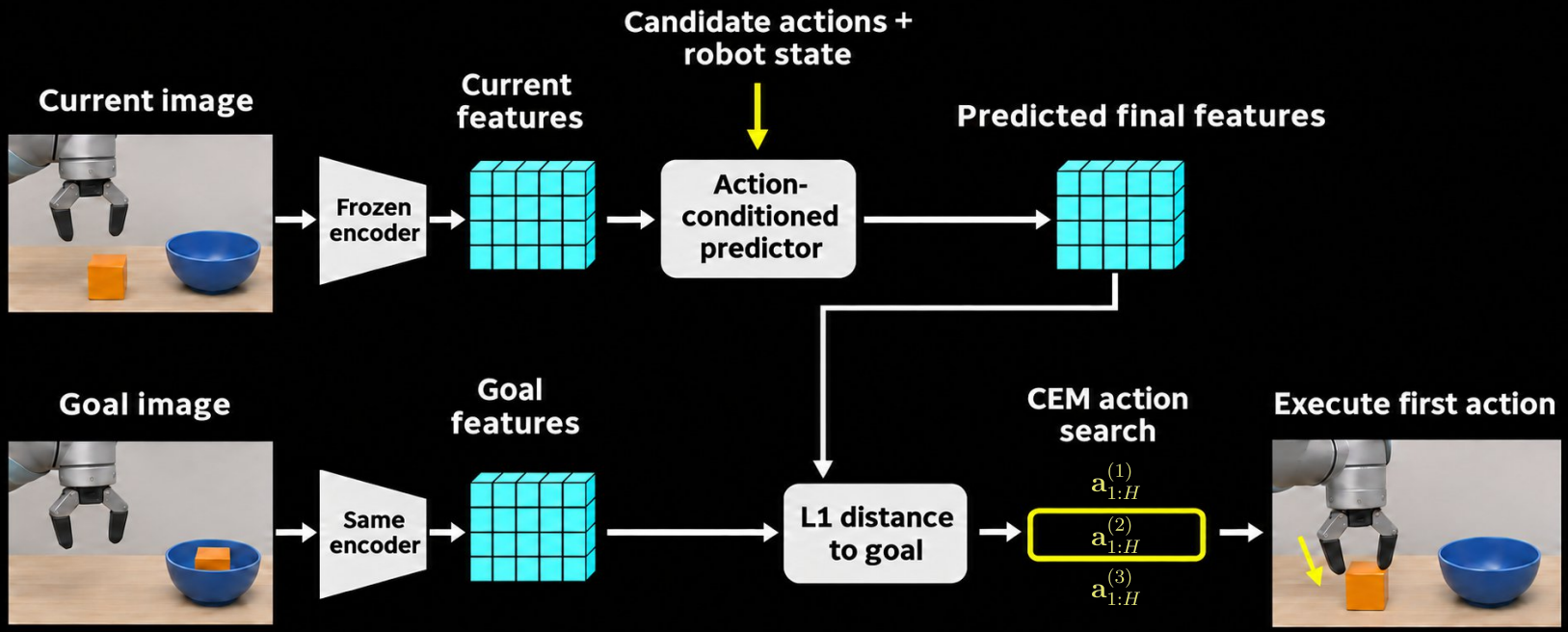
DreamerV3 deploys its trained actor without lookahead search.

TD-MPC2 executes the first planned action, then observes and replans.

DreamerV3, Learning algorithm and Fig. 3 / TD-MPC2, §3.1–3.2

V-JEPA 2-AC: planning with visual features

Video pretraining, then a frozen encoder and a predictor trained on DROID.



Execute one action, observe again, and replan. Model weights stay fixed.

Planning with a world model



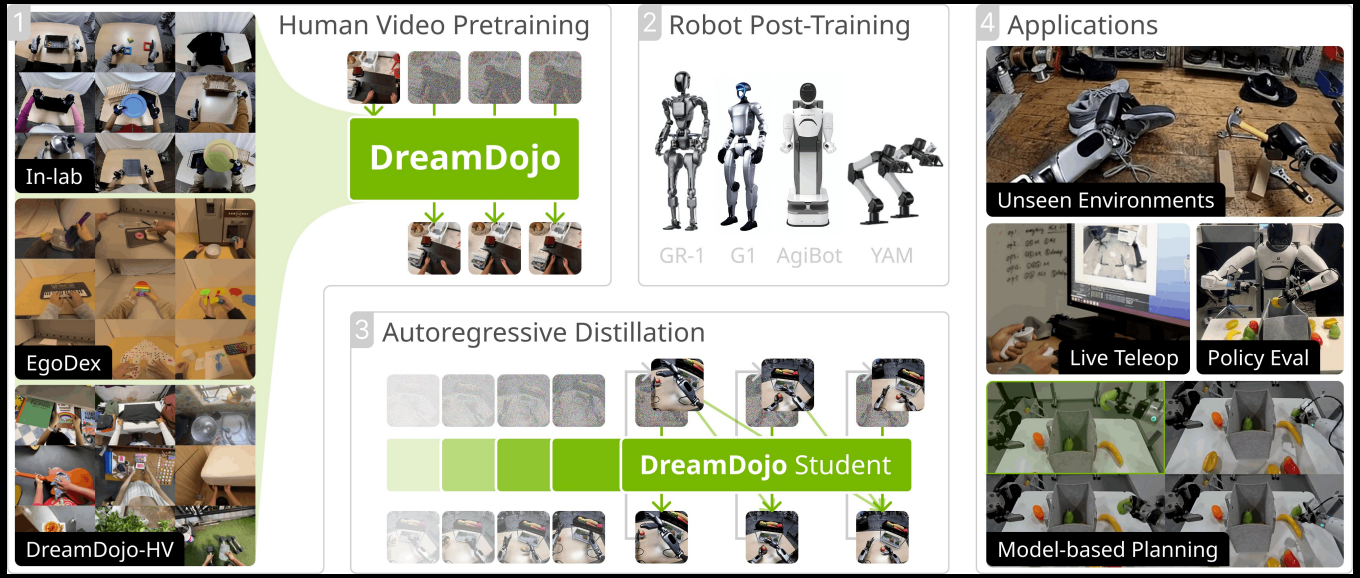
Task: put the marker in the cup

1 Propose	Sample several possible action chunks.
2 Predict	Roll each chunk forward with the learned world model.
3 Compare	Score the predicted outcome against the goal.
4 Act and repeat	Execute a short prefix, observe the real result, then replan.

Example: reject a predicted rim collision and choose an approach from above.

V-JEPA 2 / TD-MPC2

DreamDojo learns an action-conditioned simulator

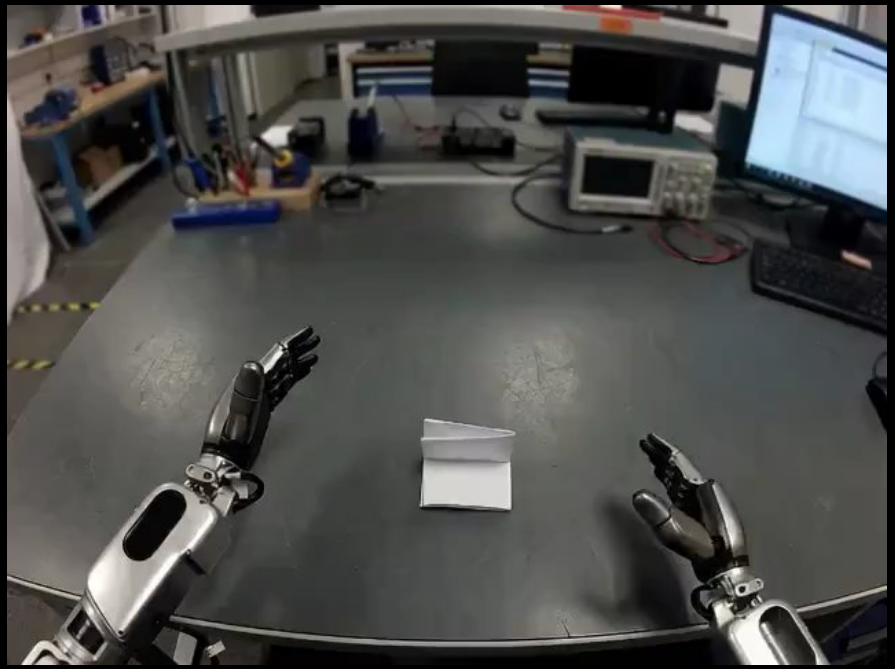


Human video + latent actions for pretraining

Robot-action post-training and few-step distillation

DreamDojo

Generated futures need counterfactual tests



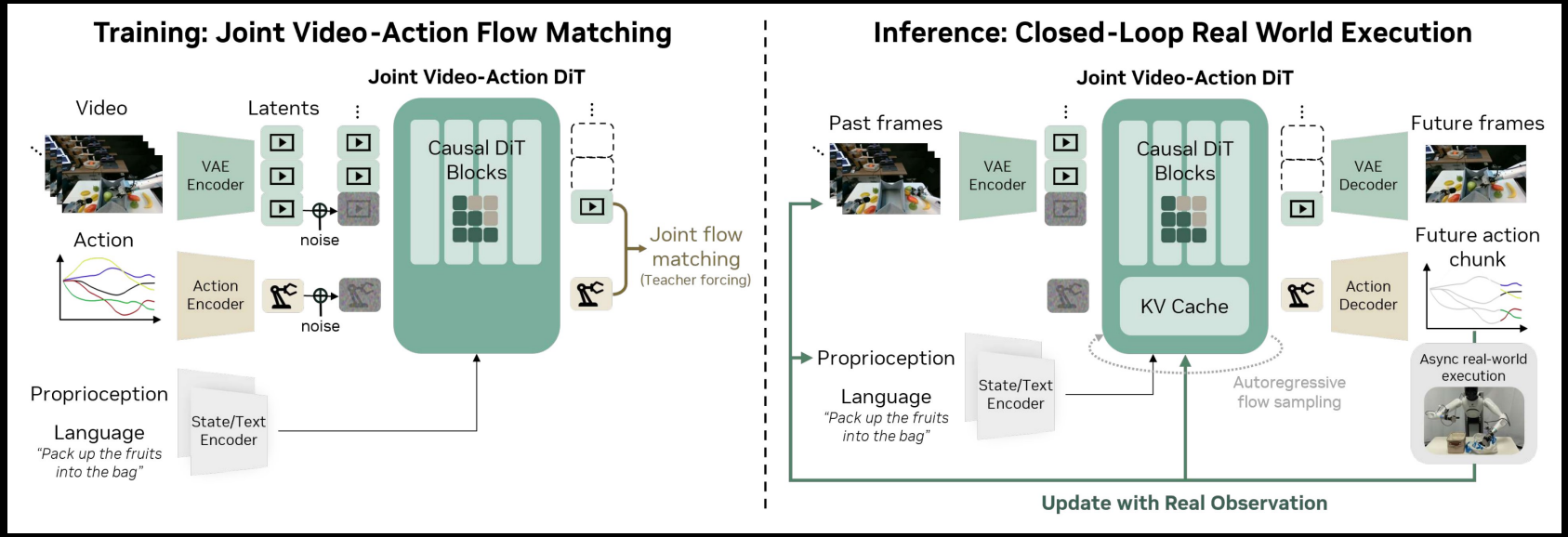
Generated future conditioned on robot actions

GENERATED video,
not a real-robot rollout

Change the action and
check the resulting
interaction

DreamDojo / Cosmos 3

DreamZero jointly predicts video and actions



Joint prediction couples future observations and actions

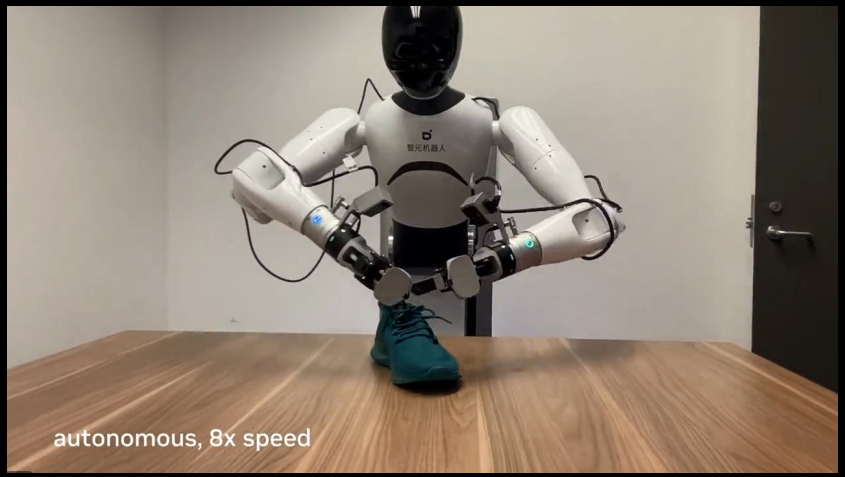
Reported 7 Hz uses two GB200 GPUs (14B model)

DreamZero

DreamZero: real-robot rollouts

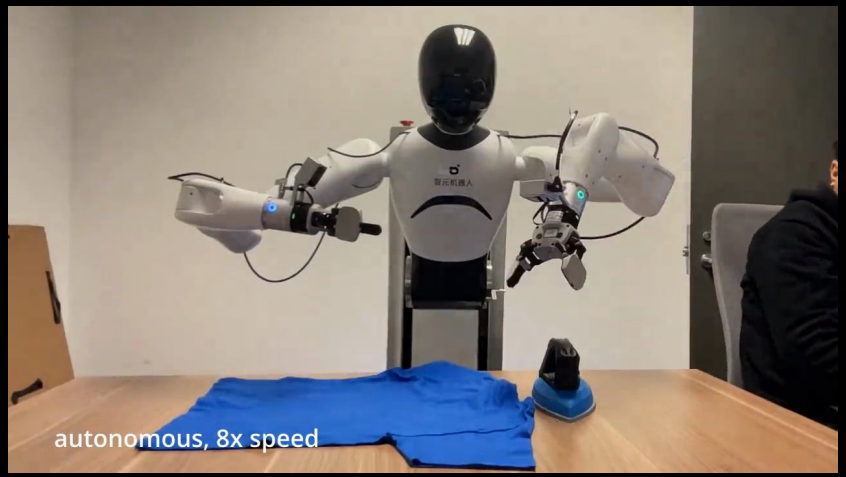
Tasks absent from the robot pretraining set

Untie the shoelace



Autonomous execution · source video at 8× speed

Iron the shirt



Autonomous execution · source video at 8× speed

DreamZero official project page / AgiBot unseen-task evaluation

Cosmos 3: reasoning and generation



Mode	Supplied information	Prediction
Forward dynamics	Visual context + actions	Future video
Inverse dynamics	Observed video transitions	Actions explaining the motion
Policy (joint video-action)	Visual context + task instruction	Actions + future video
Understanding	Image / video + text prompt	Text response

Autoregressive reasoner + flow-matching generator
Separate parameter sets connected through attention

Cosmos 3 technical report, §2.2–2.3, Fig. 4–5 and §4.2.5 (June 2026)