# World Models for Robotic Manipulation


## RILS LAB

Robot Intelligence Learning & System

임찬우

2026. 08. 14

01

# DreamGen: Unlocking Generalization in Robot Learning through Video World Models

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Lin Yen-Chen, Loic Magne, Ajay Mandlekar, Avnish Narayan, You Liang Tan, Guanzhi Wang, Jing Wang, Qi Wang, Yinzhen Xu, Xiaohui Zeng, Kaiyuan Zheng, Ruijie Zheng, Ming-Yu Liu, Luke Zettlemoyer, Dieter Fox, Jan Kautz, Scott Reed, Yuke Zhu, Linxi Fan

NVIDIA, University of Washington, KAIST, UCLA, UCSD, CalTech, NTU, University of Maryland, UT Austin

CoRL 2025
arXiv May. 2025

https://arxiv.org/abs/2505.12705

Source: Google Scholar (Aug. 2026) / Citations: 162

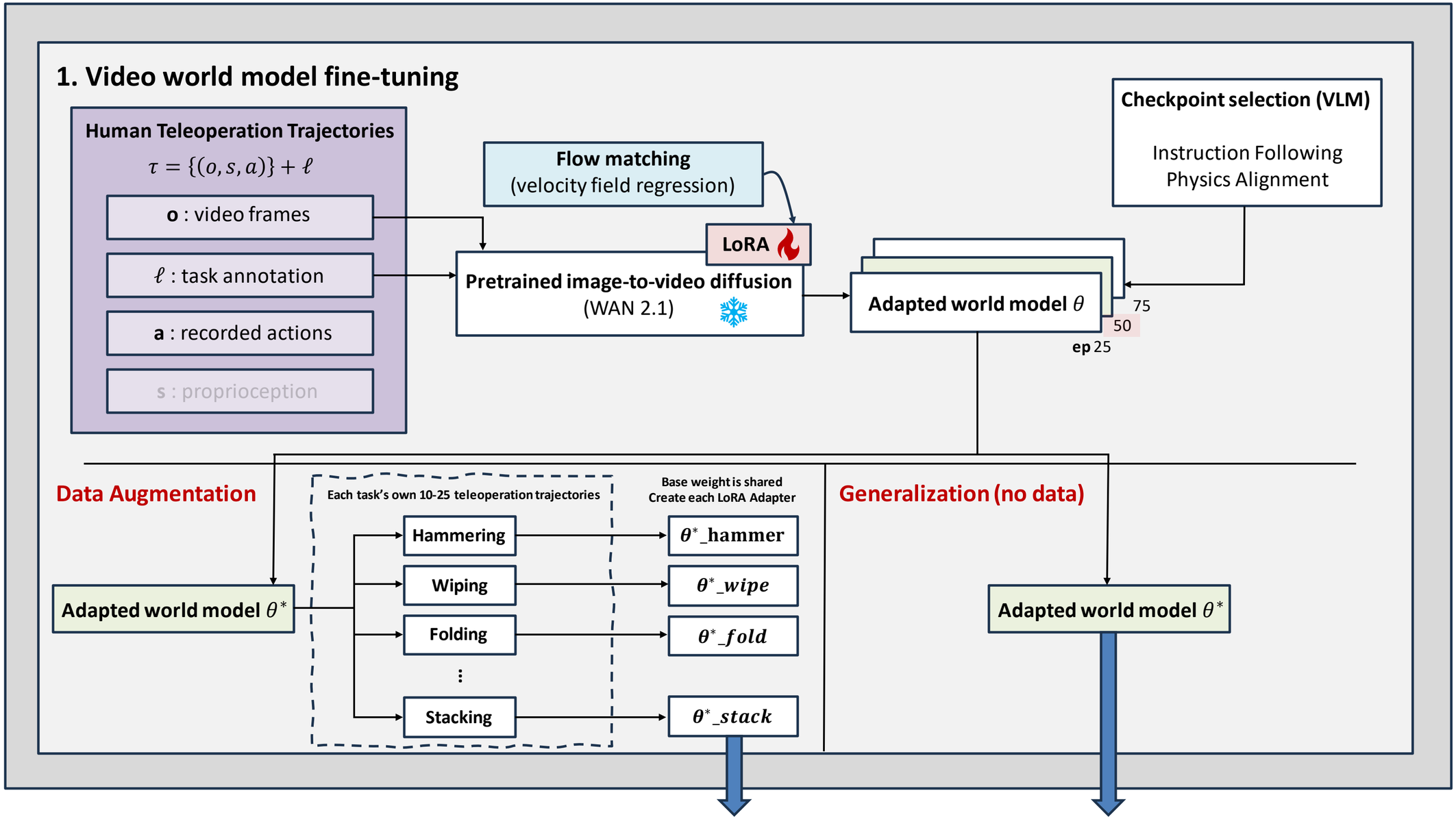
# Problem

- **Teleoperation Cost** — Scales as tasks × environments
  - Every new task or environment requires collecting data from scratch by hand.
  - **Solution**: A video world model generates the data instead.

- **Simulation** — Sim-to-real gap & modeling limits
  - Simulated trajectories suffer from the sim-to-real gap, cannot model liquids or articulated objects, and are confined to human demonstrations.
  - **Solution**: Bypass the simulator and generate photorealistic video directly.

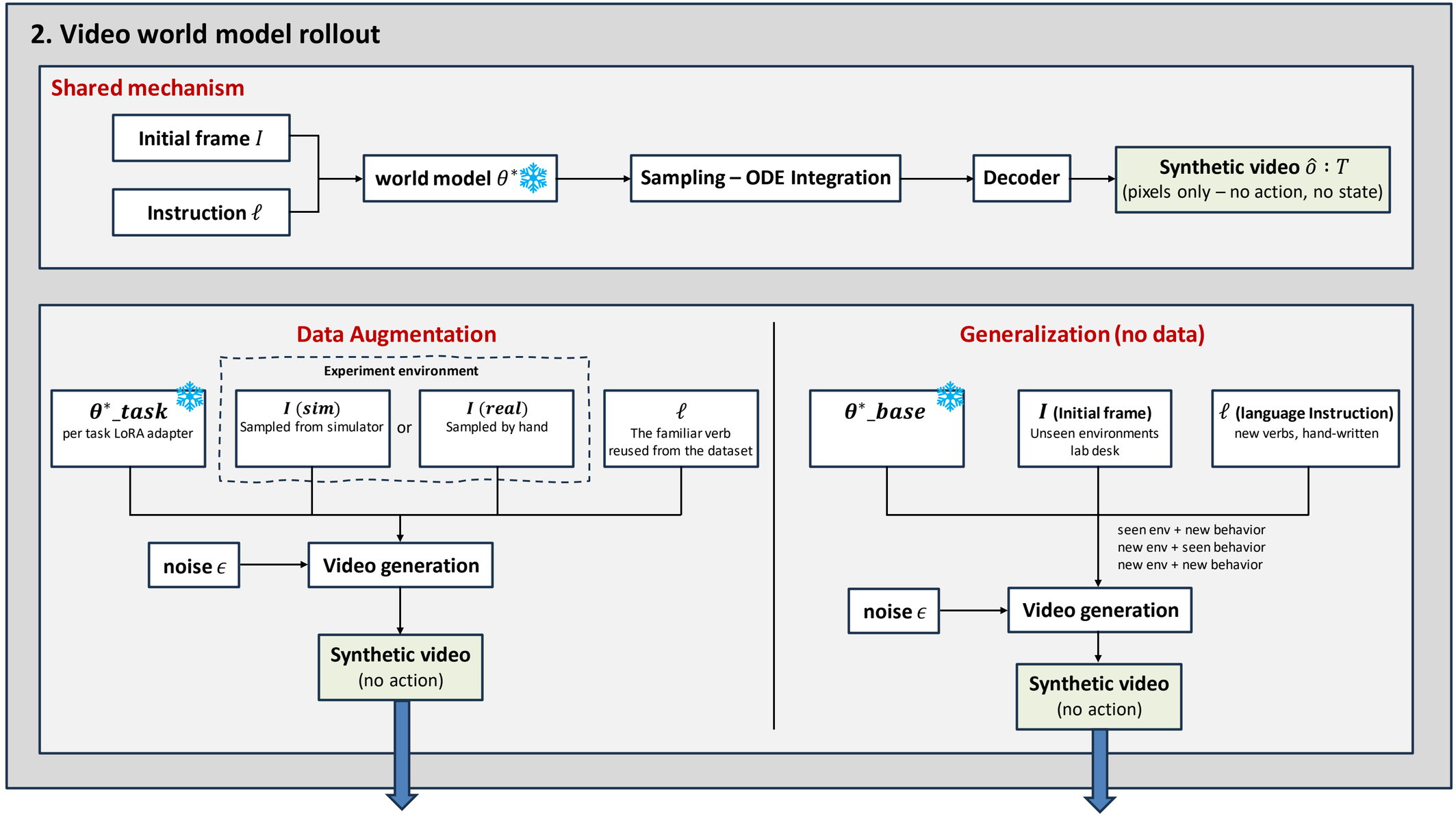
- **Missing Action Labels** — Video alone cannot train a policy
  - The world model emits pixels only. Without actions, behavior cloning has no regression target.
  - **Solution**: An IDM recovers pseudo-actions, completing the neural trajectory.

# Method

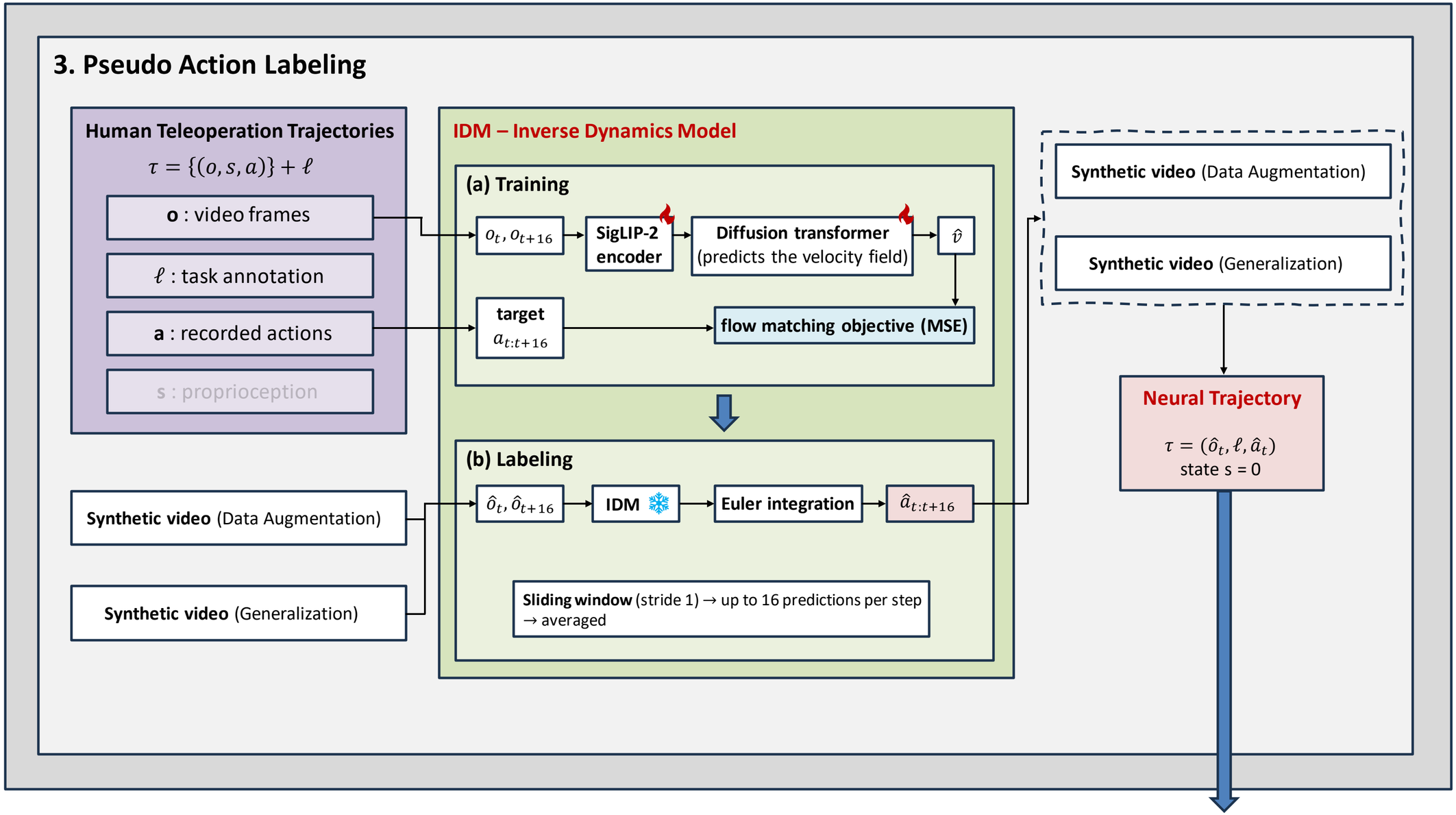

1. Video world model fine-tuning
Human Teleoperation Trajectories
$\tau = \{(o, s, a)\} + \ell$
o : video frames
$\ell$ : task annotation
a : recorded actions
s : proprioception
Flow matching
(velocity field regression)
LoRA
Pretrained image-to-video diffusion
(WAN 2.1)
Checkpoint selection (VLM)
Instruction Following
Physics Alignment
Adapted world model $\theta$
75
50
ep 25
Data Augmentation
Each task's own 10-25 teleoperation trajectories
Base weight is shared
Create each LoRA Adapter
Adapted world model $\theta^*$
Hammering
Wiping
Folding
Stacking
$\theta^*$_hammer
$\theta^*$_wipe
$\theta^*$_fold
$\theta^*$_stack
Generalization (no data)
Adapted world model $\theta^*$

# Method

## 2. Video world model rollout



# Method


3. Pseudo Action Labeling
Human Teleoperation Trajectories
$\tau = \{(o, s, a)\} + \ell$
o : video frames
$\ell$ : task annotation
a : recorded actions
s : proprioception
IDM – Inverse Dynamics Model
(a) Training
$o_t, o_{t+16}$
SigLIP-2 encoder
Diffusion transformer (predicts the velocity field)
$\hat{v}$
target $a_{t:t+16}$
flow matching objective (MSE)
(b) Labeling
$\hat{o}_t, \hat{o}_{t+16}$
IDM
Euler integration
$\hat{a}_{t:t+16}$
Sliding window (stride 1) → up to 16 predictions per step → averaged
Synthetic video (Data Augmentation)
Synthetic video (Generalization)
Synthetic video (Data Augmentation)
Synthetic video (Generalization)
Neural Trajectory
$\tau = (\hat{o}_t, \ell, \hat{a}_t)$
state s = 0

# Method

## 4. Policy Training on Neural Trajectories

**Data Augmentation**

real trajectories 8-25 per task
neural trajectories 40-300 per task
sampled 1:1

**Generalization**

real trajectories none
neural trajectories 50 per task
100% neural

Same policy, different data mixture

**Visuomotor policy**

**Neural Trajectory**

$\tau = (\hat{o}_t, \ell, \hat{a}_t)$
state s = 0

$\hat{o}_t, \ell$ → **Vision tower** → **LLM** → **Diffusion Transformer** → $\hat{a}_{t:t+H}$

**s = 0** → **State encoder** → **Diffusion Transformer**

$\mathcal{L} = \|\hat{v} - v\|^2$
$v = \hat{a} - \epsilon$

$\hat{a}_{t:t+H}$

**Executed on the physical robot**

**Results**

| | |
|---|---|
| **GR1 (4 tasks)** | $37 \rightarrow 46.4\%$ |
| **Franka (3 tasks)** | $23 \rightarrow 37.0\%$ |
| **SO-100 (2 tasks)** | $21 \rightarrow 45.5\%$ |

| | |
|---|---|
| **Seen env · new behavior (14 tasks)** | $11.2 \rightarrow 43.2\%$ |
| **New env (13 tasks)** | $\mathbf{0.0} \rightarrow 28.5\%$ |

02

# Dream to Manipulate: Compositional World Models empowering robot Imitation Learning with Imagination

Leonardo Barcellona, Andrii Zadaianchuk, Davide Allegro, Samuele Papa, Stefano Ghidoni, Efstratios Gavves

University of Padova, University of Amsterdam, Politecnico of Torino, Archimedes/Athena RC

ICLR 2025
arXiv Dec. 2024

https://arxiv.org/abs/2412.14957

Source: Google Scholar (Aug. 2026) / Citations: 55

# Problem

- **Demonstration Cost** — Policies need hundreds of demonstrations
  - Vision-based imitation learning still requires hundreds of demos, and those demos must cover enough variation.
  - **Solution:** Generate new demonstrations inside the world model — works with as few as one demo.

- **Learned World Models** — Break outside the training domain
  - Generative world models (UniSim, Dreamer, RoboDreamer) cannot capture the real physics of the robot's environment.
  - **Solution:** Do not learn the dynamics — plug a physics engine (PyBullet) into $D$ directly.

- **Augmentation Validity** — Editing robot data can silently break the task
  - Changing an object's position invalidates the action sequence; prior methods only recombine existing trajectories.
  - **Solution:** Equivariant transforms move observations and actions together, then the physics engine verifies each result and discards invalid ones.

# Method

## Building the world model

### Input

**RGB-D video X**

$X = (x_1, \ldots, x_n)$

**Segmentation Masks $Y_k$**

DEVA open-vocabulary tracker

**Camera poses**

**Demo trajectory**

### Object-centric Reconstruction → Object Assets

**Masking**

$y_{n,k} \odot x_{n,k}$

k = 5

n = 200

**2DGS**

**Inverse graphics function**

$G_k = h(X, X_{depth}, Y_k)$

**Also reconstructed**

- **Environment** $G_0$
- **Table**
- **Robot A**

**Gaussian Splats $G_k$**

rendering only

**Physics –** both packaged into one **URDF** file

- **TSDF Mesh $M_k$** (collision)
- **$\omega$** (inertial, mass, friction)

k = 5

**Object Asset**

$O_k = (G_k, M_k, \omega)$

**World Model**

$\mathcal{M} = (O, A, D, T)$

# Method

## World Model : Components and Interaction

$\mathcal{M} = (O, A, D, T)$ | $O$: object assets, $A$: agent, $D$: dynamics operator, $T$: task

### Static Assets (built once)

**Object Assets $O$**

appearance
$G_k$ Gaussian Splats

physics
$\omega$ Physics Parameter

physics
$M_k$ Collision Mesh

**Agent $A$**

**Robot**
URDF + Per-link Splats

**Scene extras**

**Environment $G_0$**
background splats

**Table**
RANSAC plane, fixed

### Runtime Cycle (one imagined rollout)

$M_k, \omega, A, Table \rightarrow$ drive the dynamics

**Action $a_t$** → IK → **Dynamics $D$**
$(O_{t+1}, A_{t+1}) = D \cdot (O_t, A_t)$

Step → **World State = $(O_t, A_t)$**
$O_t : \{(\mu_k, \rho_k)\}, k = 1 \ldots K$
$A_t : q\ (joint\ angles)$

explicit state – no latent

Current Waypoint

**Task $T$**
Waypoints + Keypoints
Language Goal (CLIP)

Waypoint reached → advance the index

$\Delta\mu_k, \Delta\rho_k$
(pose delta)

**Gaussian Update $U$**
$p' = p + \Delta\mu$
$r' = \Delta p \cdot r$

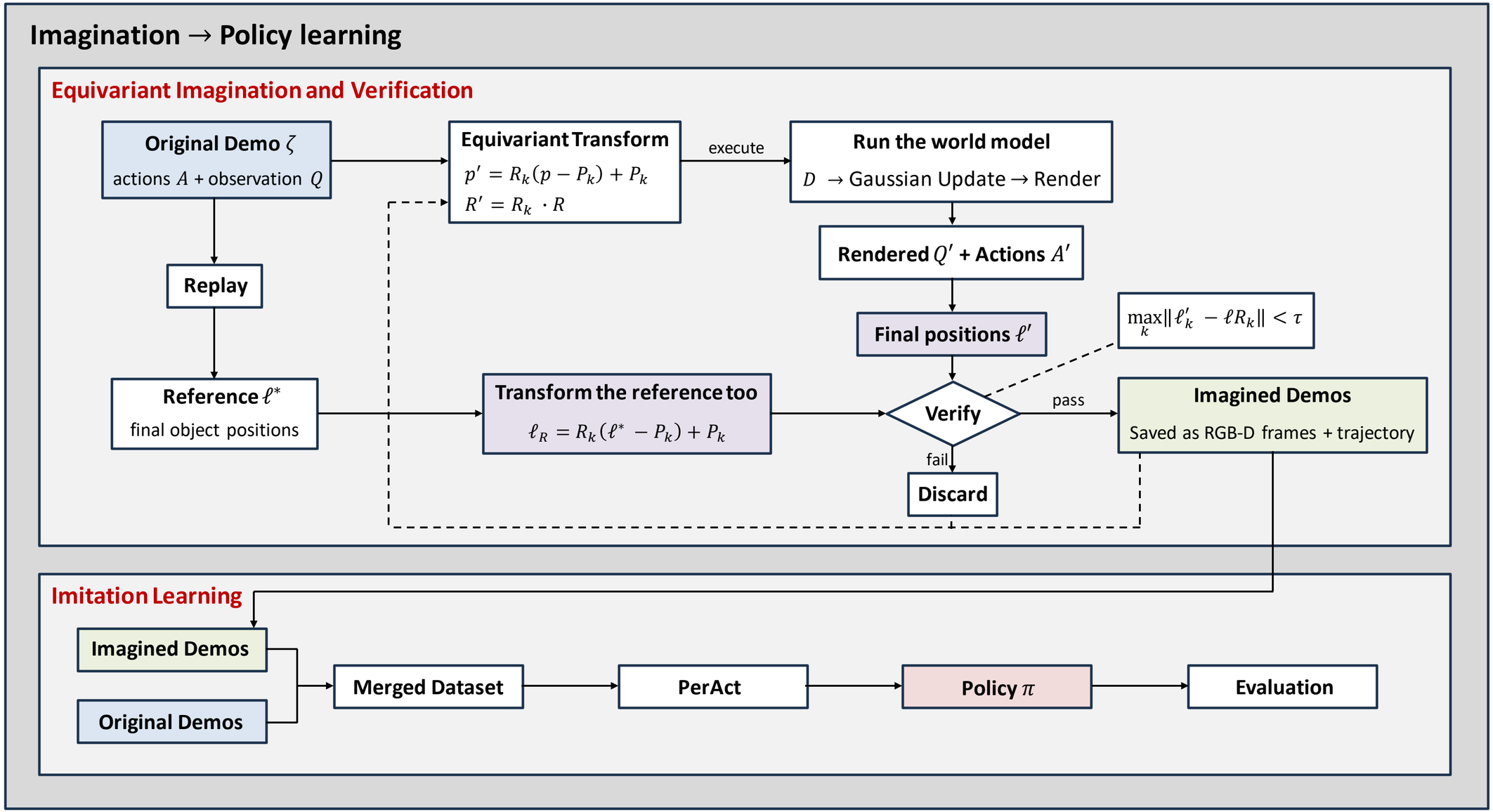
Sync → **Render R**
Rosterizer + Camera

**Imagined Observation**
RGB-D → saved to disk
terminal – feeds PerAct

$G_k \rightarrow$ moved by $U$

$G_0$ and robot splats → rendered by $R$

# Method


Imagination → Policy learning
Equivariant Imagination and Verification
Original Demo $\zeta$
actions $A$ + observation $Q$
Equivariant Transform
$p' = R_k(p - P_k) + P_k$
$R' = R_k \cdot R$
execute
Run the world model
$D$ → Gaussian Update → Render
Rendered $Q'$ + Actions $A'$
Final positions $\ell'$
$\max_k \|\ell'_k - \ell R_k\| < \tau$
Replay
Reference $\ell^*$
final object positions
Transform the reference too
$\ell_R = R_k(\ell^* - P_k) + P_k$
Verify
pass
fail
Discard
Imagined Demos
Saved as RGB-D frames + trajectory
Imitation Learning
Imagined Demos
Original Demos
Merged Dataset
PerAct
Policy $\pi$
Evaluation

03

# PIN-WM: Learning Physics-INformed World Models for Non-Prehensile Manipulation

Wenxuan Li, Hang Zhao, Zhiyuan Yu, Yu Du, Qin Zou, Ruizhen Hu, Kai Xu

National University of Defense Technology, Wuhan University, Shenzhen University,
Guangdong Laboratory of Artificial Intelligence and Digital Economy

RSS 2025
arXiv Apr. 2025

https://arxiv.org/abs/2504.16693

Source: Google Scholar (Aug. 2026) / Citations: 29

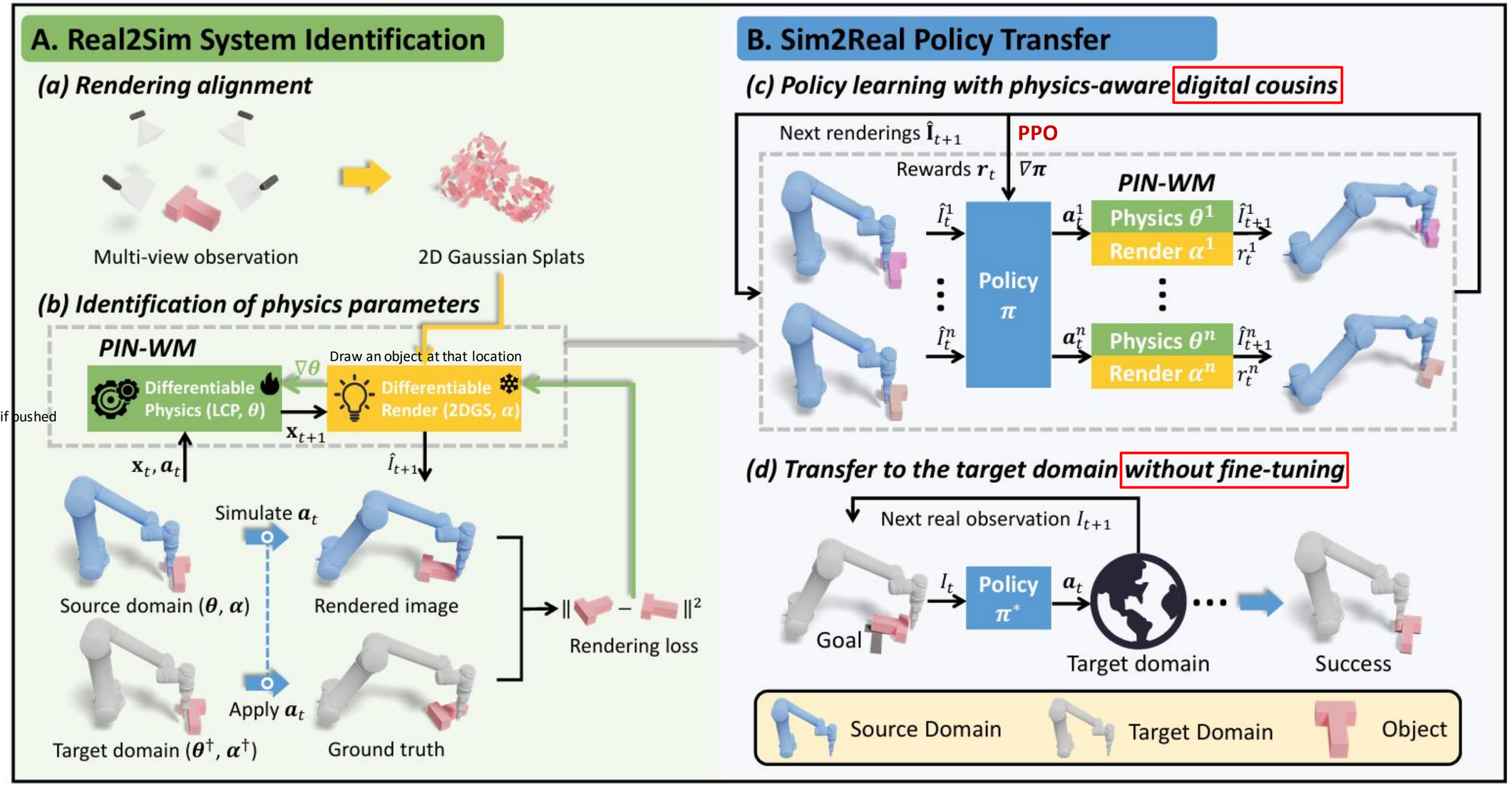
# Problem

- **Data Requirement** — Data-driven world models need many trajectories and break outside their training domain
  - Dreamer V2 gets only 1% push success from 100 trajectories — prediction collapses out of distribution.
  - **Solution**: World model = differentiable physics ∘ differentiable rendering. Only 10 physical scalars are unknown → one task-agnostic push gives 97%.

- **State-Estimation Dependency** — A separate pose estimator leaks its error into the physics
  - Prior identification must estimate object state first, and that error propagates straight into the parameters.
  - **Solution**: 2DGS renders the simulated state into an image, so a pixel loss is optimized directly — no estimator.

- **Residual Sim2Real Gap** — The twin is inexact, and unconstrained randomization dulls the policy
  - Partial, inaccurate observations leave a gap, yet wide domain randomization drops success to 33%.
  - **Solution**: Digital cousins — perturb only physics and lighting ±10% around the identified values → 97% in sim, 75% / 65% on the real robot, no fine-tuning.

# Method


A. Real2Sim System Identification
(a) Rendering alignment
Multi-view observation
2D Gaussian Splats
(b) Identification of physics parameters
PIN-WM
Differentiable Physics (LCP, θ)
Differentiable Render (2DGS, α)
Draw an object at that location
Calculate where an object will go if pushed
Simulate $a_t$
Apply $a_t$
Source domain (θ, α)
Rendered image
Target domain (θ†, α†)
Ground truth
Rendering loss
B. Sim2Real Policy Transfer
(c) Policy learning with physics-aware digital cousins
Next renderings $\hat{I}_{t+1}$
PPO
Rewards $r_t$
PIN-WM
Policy π
Physics θ¹
Render α¹
Physics θⁿ
Render αⁿ
(d) Transfer to the target domain without fine-tuning
Next real observation $I_{t+1}$
Policy π*
Goal
Target domain
Success
Source Domain
Target Domain
Object

04

# GWM: Towards Scalable Gaussian World Models for Robotic Manipulation

Guanxing Lu, Baoxiong Jia, Puhao Li, Yixin Chen, Ziwei Wang, Yansong Tang, Siyuan Huang,

Tsinghua University, State Key Laboratory of General Artificial Intelligence, BIGAI,
School of Electrical and Electronic Engineering, Nanyang Technological University

ICCV 2025
arXiv Aug. 2025

https://arxiv.org/abs/2508.17600

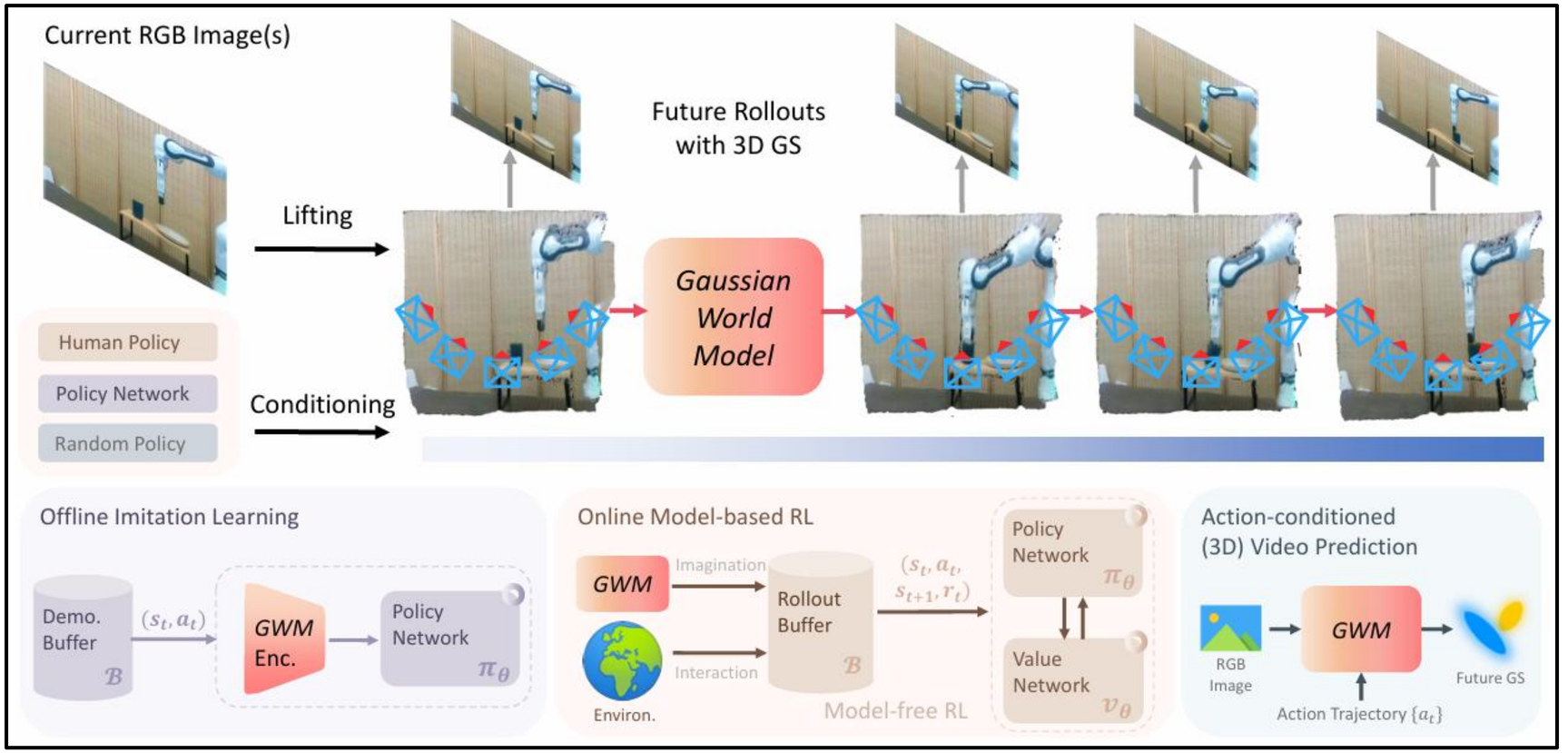
Source: Google Scholar (Aug. 2026) / Citations: 54

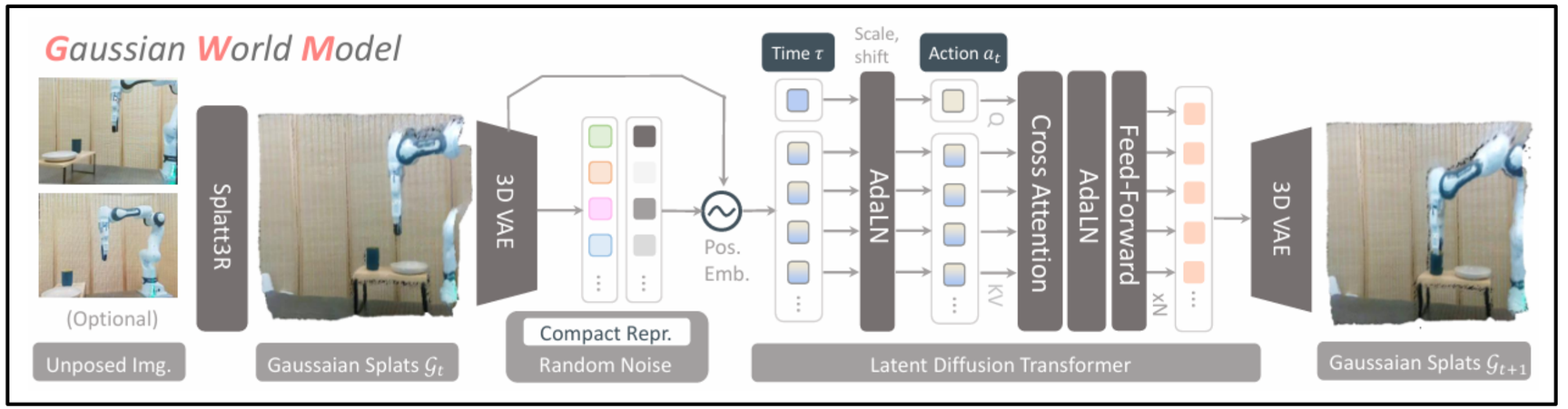
# Problem

- **Spatial Grounding** — Image-based world models don't understand 3D
  - Pixel-space world models (iVideoGPT, UniSim) have no geometric structure, so they collapse under lighting, camera-pose, and texture shifts and cannot localize objects precisely.
  - **Solution**: Represent the world state as explicit 3D Gaussians instead of pixels — the model reasons directly in metric 3D space.

- **Reconstruction Cost** — 3D representations are far too slow for a world model
  - NeRF and vanilla 3D-GS need per-scene optimization (~30k iterations), which cannot support model-based RL that imagines tens of thousands of rollouts.
  - **Solution**: Obtain Gaussians from a feed-forward reconstruction model (Splatt3R) in a single pass, then compress them with a 3D Gaussian VAE into a fixed-length latent.

# Method


Current RGB Image(s)
Lifting
Future Rollouts with 3D GS
Gaussian World Model
Human Policy
Policy Network
Random Policy
Conditioning
Offline Imitation Learning
Demo. Buffer
$(s_t, a_t)$
GWM Enc.
Policy Network
$\pi_\theta$
Online Model-based RL
GWM
Imagination
Interaction
Environ.
Rollout Buffer
$\mathcal{B}$
$(s_t, a_t, s_{t+1}, r_t)$
Model-free RL
Policy Network
$\pi_\theta$
Value Network
$v_\theta$
Action-conditioned (3D) Video Prediction
RGB Image
GWM
Future GS
Action Trajectory $\{a_t\}$


Gaussian World Model
(Optional)
Unposed Img.
Splatt3R
Gaussian Splats $\mathcal{G}_t$
3D VAE
Compact Repr.
Random Noise
Pos. Emb.
Time $\tau$
Scale, shift
Action $a_t$
AdaLN
Q
KV
Cross Attention
AdaLN
Feed-Forward
xN
Latent Diffusion Transformer
3D VAE
Gaussian Splats $\mathcal{G}_{t+1}$

05

# EnerVerse: Envisioning Embodied Future Space for Robotics Manipulation

Siyuan Huang, Liliang Chen, Pengfei Zhou, Shengcong Chen, Yue Liao, Zhengkai Jiang, Yue Hu, Peng Gao, Hongsheng Li, Maoqing Yao, Guanghui Ren

SJTU, AgiBot, Shanghai AI Lab, CUHK MMLab, LV-NUS Lab

NeurIPS 2025
arXiv Jan. 2025

https://arxiv.org/abs/2501.01895

Source: Google Scholar (Aug. 2026) / Citations: 70

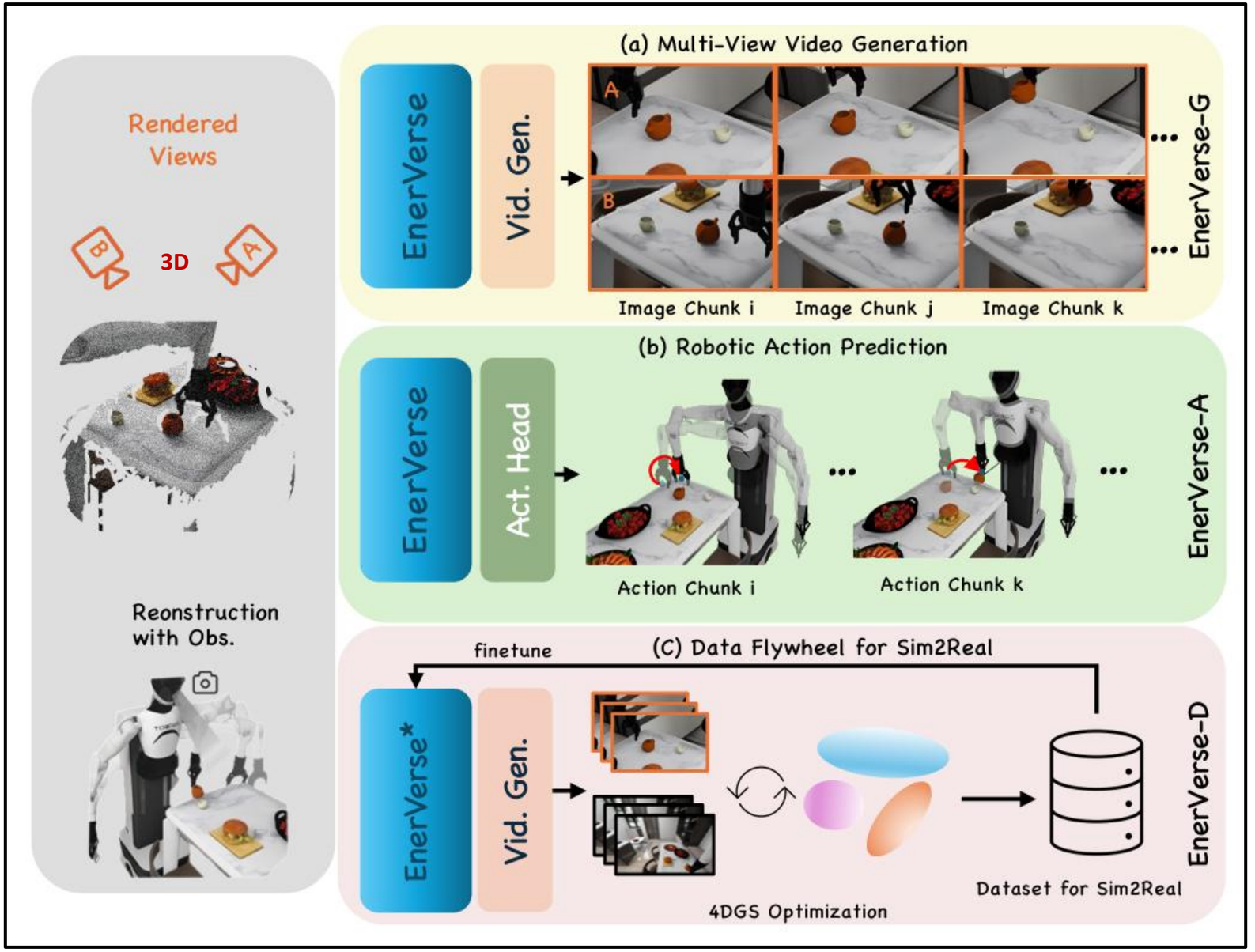
# Problem

- **Long-Horizon Collapse** — Autoregressive video generation drifts and breaks down
  - Feeding consecutive frames as memory is redundant, so errors accumulate and the model collapses on long tasks.
  - **Solution**: Sparse Memory — keep only randomly sampled past frames, drop about 80%. LIBERO-Long 30.8 → 73.0.

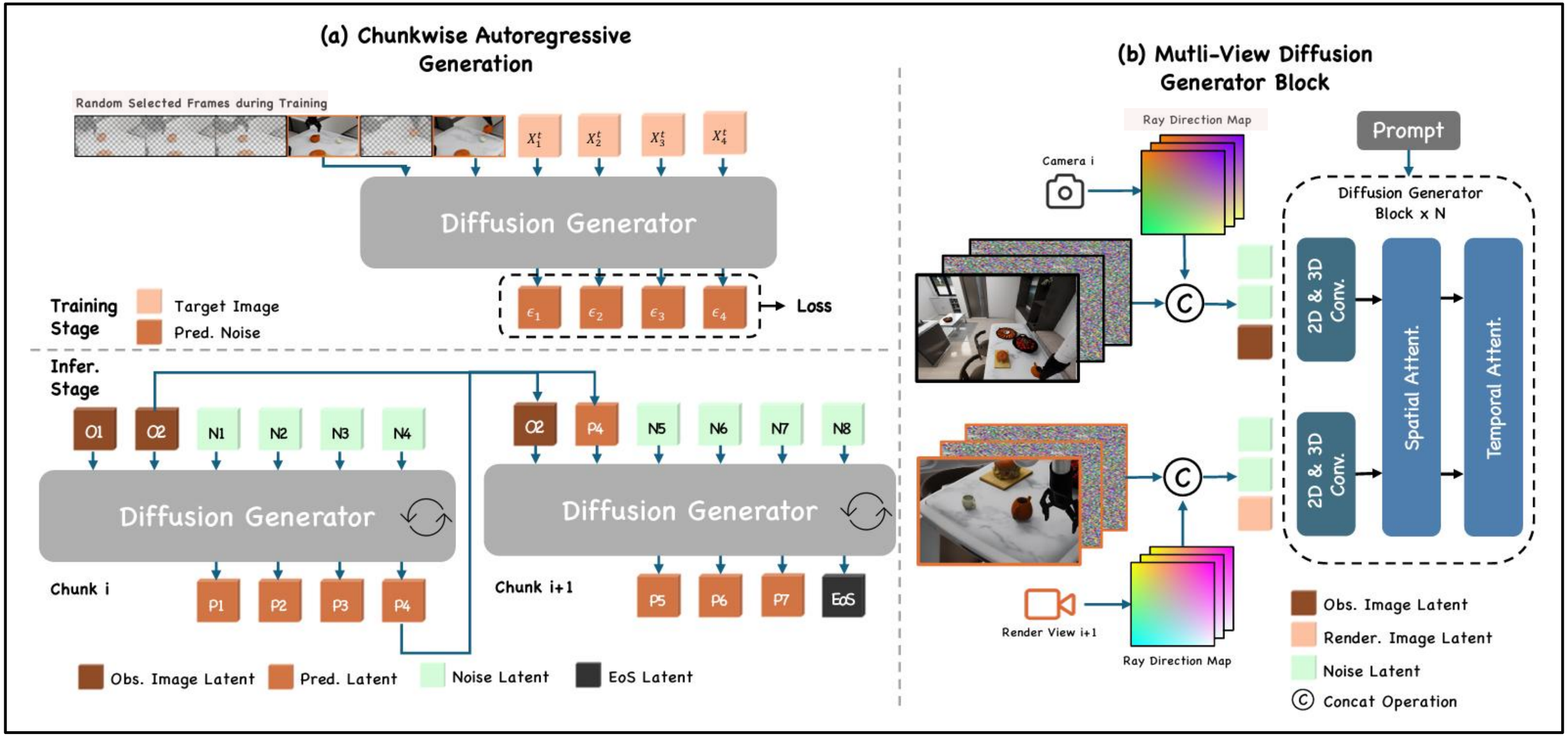
- **3D Grounding from One Camera** — Single-view video cannot resolve depth and occlusion
  - Mounting more cameras raises hardware cost, I/O bandwidth, and system complexity.
  - **Solution**: Pre-train on multi-view video with ray direction maps; at deployment, depth-warp one RGB-D camera into rendered views. 79.0 → 92.1.

- **Multi-View Data Scarcity** — Calibrated multi-camera robot data barely exists
  - Public datasets are single-view only, and simulator data suffers the Sim2Real gap.
  - **Solution**: EnerVerse-D — a data flywheel where the generator imagines missing views and 4D Gaussian Splatting enforces physical consistency. Hallucination −40%.

# Method


Rendered Views
3D
Reonstruction with Obs.
(a) Multi-View Video Generation
EnerVerse
Vid. Gen.
A
B
Image Chunk i
Image Chunk j
Image Chunk k
EnerVerse-G
(b) Robotic Action Prediction
EnerVerse
Act. Head
Action Chunk i
Action Chunk k
EnerVerse-A
finetune
(C) Data Flywheel for Sim2Real
EnerVerse*
Vid. Gen.
4DGS Optimization
Dataset for Sim2Real
EnerVerse-D

# Method


(a) Chunkwise Autoregressive Generation
Random Selected Frames during Training
$X_1^t$ $X_2^t$ $X_3^t$ $X_4^t$
Diffusion Generator
$\epsilon_1$ $\epsilon_2$ $\epsilon_3$ $\epsilon_4$
Loss
Training Stage
Target Image
Pred. Noise
Infer. Stage
O1 O2 N1 N2 N3 N4
Diffusion Generator
Chunk i
P1 P2 P3 P4
O2 P4 N5 N6 N7 N8
Diffusion Generator
Chunk i+1
P5 P6 P7 EoS
Obs. Image Latent
Pred. Latent
Noise Latent
EoS Latent
(b) Mutli-View Diffusion Generator Block
Ray Direction Map
Camera i
Prompt
Diffusion Generator Block x N
2D & 3D Conv.
2D & 3D Conv.
Spatial Attent.
Temporal Attent.
Render View i+1
Ray Direction Map
Obs. Image Latent
Render. Image Latent
Noise Latent
Concat Operation

# Thank you

Question & Disscusion


RILS LAB