Agentic Real2Sim: Physics-based World Modeling with Vision-Language Agents

Guanxiong Chen[1,6,*,#], Qianjun Xia[1,*], Jiawei Peng[2,*], Heng Zhang[1,*], Pengyu Jing[6,*], Bole Ma[3,*], Justin Qian[1], Yixian Cheng[1], Ziyi Jiao[1], Bingyang Zhou[6], Yiduo Qu[7], Luoxin Ye[2], Kaifeng Zhang[4], Kunyi Wang[1], Weijia Zeng[1], Yunuo Chen[5], Pengzhi Yang[6], Ziqiu Zeng[6], Siyuan Luo[6], Huamin Wang[8], Chao Liu[1], Alan Yuille[2], Fan Shi[6], Changxi Zheng[4], Yunzhu Li[4], Chenfanfu Jiang[5], Peter Yichen Chen[1]

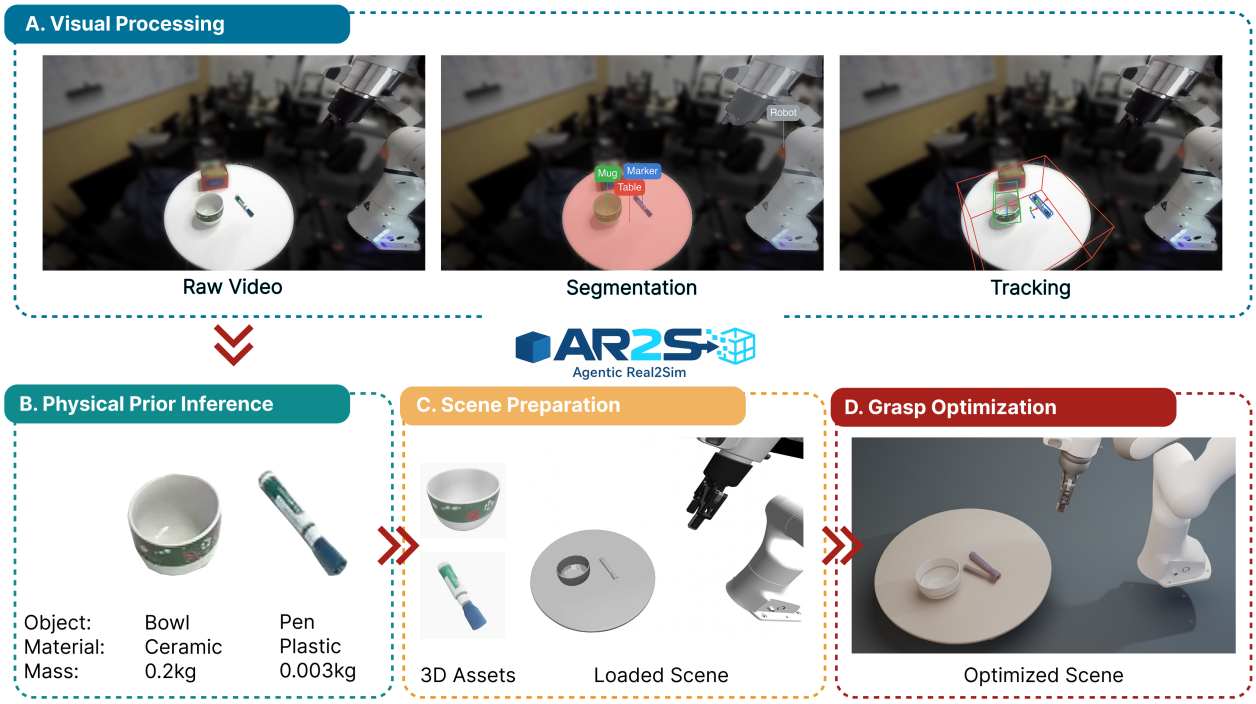

Fig. 1. **Agentic Real2Sim architecture.** A recorded DROID episode is converted into a simulatable digital twin through four linked agents: The visual processing agent from raw video extracts segmentation and depth masks, geometries and object poses. The physical-prior inference agent attaches object identity, material class, mass hints, and other constitutive properties needed by the simulator. The scene preparation agent initializes a scene with the robot's initial state, cameras, and reconstructed geometry, and places objects of interest in the correct locations. Simulator-in-the-loop grasp optimization further optimizes object placements to enable successful grasping. The same episode contract supplies the inputs and feedback channels used by the rigid, deformable, and humanoid adapters.

***Abstract*— Real-to-sim conversion for robotic interaction with objects remains labor-intensive because it requires more than visual reconstruction: a streamlined real2sim process must recover scene geometries and object states, infer physical parameters, and assemble actors, objects, cameras, poses, and trajectories into a runnable physical simulation. Today this process still depends on brittle workflow glue across visual perception tools and simulators: manual tuning of visual foundation models, mesh cleanup, coordinate frame alignments, etc. We introduce *Agentic Real2Sim*, a framework for generalized physical world modeling with vision-language agents that converts a real-world recording of object-robot interaction into a simulatable episodic twin, and connects the resulting twin to downstream policy fine-tuning and evaluation. We evaluate Agentic Real2Sim on rigid-object manipulation, deformable-object interaction, and humanoid motion scenes, spanning domains that are usually handled by separate Real2Sim pipelines. The framework's agentic decisions can be driven by an open-weight VLM backend at a small fraction of the cost of frontier models, while attaining a comparable conversion success rate. The framework further supports custom scene conversion, fine-tuning of pre-trained policy with data generated from converted episodes, and works effectively as a surrogate for real-world policy evaluation. The project site is available at `https://agentic-real2sim.github.io/`.**

[*]Equal technical contribution. [#]Project lead.
[1]University of British Columbia. [2]Johns Hopkins University. [3]Erlangen National High Performance Computing Center (NHR@FAU), Friedrich-Alexander-Universität Erlangen-Nürnberg. [4]Columbia University. [5]University of California, Los Angeles. [6]National University of Singapore. [7]University of Cambridge. [8]Style3D.

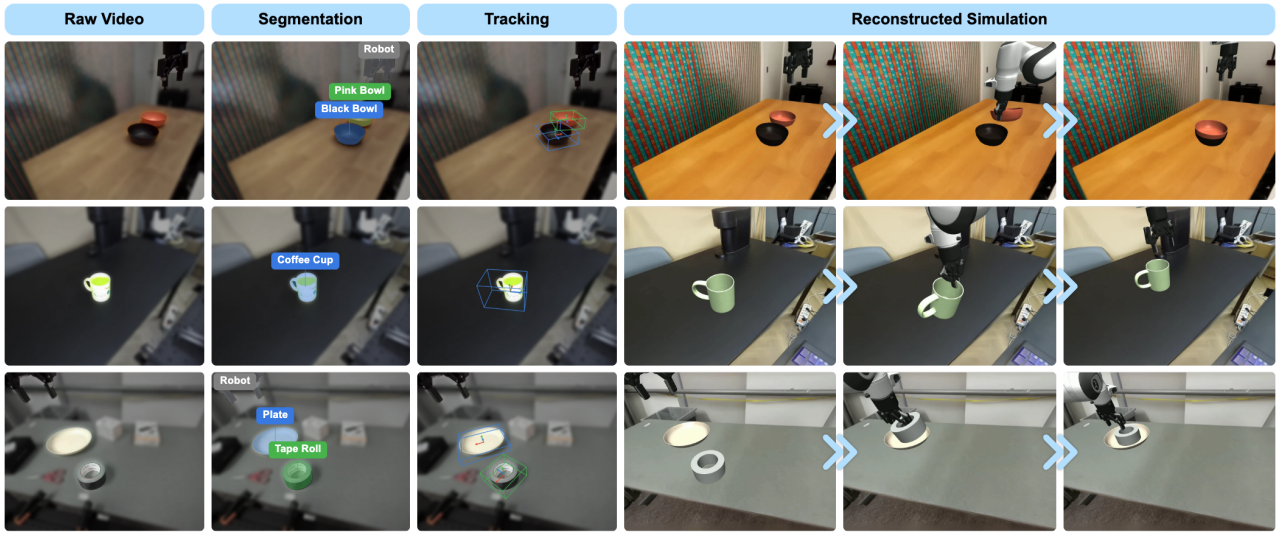

Fig. 2. **DROID episode conversion at scale.** Three representative episodes from **DROID-500**, each pairing the the episode's real-world observation with its synthetic twin. Intermediate pipeline artifacts are exposed for selected scenes: object segmentation and FoundationPose pose-tracking overlays.

I. Introduction

Large real-world robot datasets and increasingly capable simulators have made it tempting to treat real-to-sim conversion as a solved plumbing problem: collect observations, reconstruct assets, drop them into a physics engine, and train or evaluate policies. Yet in practice, physical interaction episodes resist this simplification: a manipulation recording contains a rich set of entities— cameras, manipulated objects with their physical parameters, kinematic and dynamic states, and trajectories which dictate how a robot behaves across space and time. Accumulating these elements into a *simulatable* twin episode still requires a significant amount of manual labor: configuring visual foundation models, cleaning up meshes, aligning coordinate frames, and tuning physical simulators. Further, automating these steps still requires human-made decisions, such as identifying the names of objects of interest and selecting frames for object identification.

Recent work have made strong progress on parts of this problem. Automated Real2Sim frameworks recover simulation-ready assets and physical parameters from robotic interaction [1], while Real2Sim2Real frameworks align visual and dynamic properties for manipulation policy development [2]–[4]. Meanwhile, state-of-the-art VLM-based agentic data-generation frameworks use the generate-critic-improve paradigm to build simulation-ready scenes [5]–[8] or object-level assets [9]. These lines of work point to a missing capability: scalable conversion of recorded physical interaction episodes into physically simulatable digital twins. Given a real-world recording of an interaction, can a framework produce a simulatable artifact that replays the episode, supports closed-loop policy evaluation, and generates more data for policy fine-tuning and evaluation?

We frame this problem as *Agentic Real2Sim*: Given a recorded physical interaction episode, automatically produce a physically realistic digital twin that preserves the actors, objects, trajectories, and physical parameters, then uses the resulting twin for downstream policy evaluation and fine-tuning—see Fig. 1. Our contributions are threefold: (1) We introduce a reproducible agentic real2sim framework to convert robot manipulation episodes [10] into MuJoCo-simulated episodic twins [11]. The system transforms recorded robot-object interactions into simulatable artifacts through visual processing, physical-prior inference, scene preparation, and simulator-in-the-loop refinement. Moreover, we generalize the framework beyond the DROID manipulation dataset, by enabling the same conversion process for deformable manipulation in a PhysTwin-style setting [3], and for humanoid motion with BFM-Zero-style motion context [12]. (2) We demonstrate that the framework supports interchangeable VLM backends, without relying on a single frontier VLM. We demonstrate that with our framework, an open 31B VLM achieves comparable replay-success rate to frontier VLMs on a dataset of 100 episodes, while reducing the cost by up to $31.4\times$. (3) We demonstrate downstream robotic applications with custom scene conversion, data generation through augmentation, and use of generated episodes as a substitute for real-world evaluation.

II. Related Work

Automated real-to-sim alignment. Real2Sim framework such as Scalable RealSim and TwinAligner [1], [2] have shifted from manually authored simulator scenes toward automated geometry and parameter recovery. These work establish that reconstructing photorealistic scenes alone is insufficient for robotic learning: simulators used for manipulation must also recover dynamics, contacts, and task-relevant behavior. Agentic Real2Sim shares this emphasis on physical alignment, but differs in its unit of conversion. Rather than scanning isolated objects or constructing a scene for a new policy-training setup, it converts recorded interaction episodes into simulatable twins that preserve the observed actors, manipulated objects, trajectories, contacts, and ensures visual

alignment.

VLM-driven asset generation. Recent work leverage state-of-the-art visual-language models for generating visually realistic and physically plausible assets for downstream applications. While earlier framework primarily rely on one-off VLM API calls embedded in larger scene-reconstruction and task-generation pipelines [7], [13]–[15], more recent work such as SceneSmith [8], SceneWeaver [16], PhyScensis [6], SimWorld Studio [17], and LychSim [5] have shown the strong potential of agentic VLM pipelines in reconstructing realistic scenes from real-world data, or building articulated assets through agentic code and testing loops [9]. These agentic framework enable VLMs to operate directly over intermediate generation artifacts, using structured, artifact-level feedback to identify omissions or implausibilities, and to iteratively refine subsequent generation decisions. Our work follows this paradigm of using VLM-driven agents for synthetic scene generation and real-to-sim construction, yet places greater emphasis on reconstructing full manipulation behaviors from real-world recordings, as well as generating physical parameters, in addition to placing assets appropriately to recreate visually realistic scenes.

Visual processing and simulation tools. Our pipeline also relies on replaceable visual and simulation components. SAM 3 supplies open-vocabulary segmentation for selecting and tracking relevant objects or regions [18]. SAM 3D is used as a geometry-extraction component, not as the object-discovery mechanism, by recovering 3D object assets from image evidence [19]. FoundationStereo provides stereo depth estimation from rectified camera pairs [20], and FoundationPose provides 6D object pose estimation and tracking for novel objects [21]. Deformable simulation tools such as PhysTwin and EMPM further show how visual evidence can be coupled with simulator rollouts and parameter optimization to recover material or connectivity parameters [3], [4]. These components are important building blocks, but they are not the paper's main contribution. Agentic Real2Sim contributes the episode contract and alignment loop that compose visual extraction, simulator execution, critic feedback, and domain-specific repair into executable episode twins.

Downstream use of synthetic episodes. Simulation-ready episodes enable physical queries beyond visual reconstruction. Prior work on demonstration amplification and simulation data generation studies how converted or synthesized demonstrations can support skill learning and deployment [22], [23]. Our framework can be useful for a multitude of robotics downstream tasks, e.g. transforming real-world teleoperation episodes into digital twins on a custom data collection platform, generating more synthetic data to fine-tune a DROID-pretrained $\pi_{0.5}$ policy [24], or being used as a test bed to evaluate fine-tuned policies.

III. Method

Agentic Real2Sim converts a real physical interaction episode into a simulatable *episodic twin*: a collection of observation streams, actors, manipulated entities, geometries, temporal state, and physical parameters needed to replay or query the original real-world episode in a multi-stage workflow. The stages generalize across domains– rigid-object manipulation, deformable-object interaction, and whole-body humanoid motion; while skill and tool invocations differ between the domains.

A. System Overview

Fig. 1 summarizes the architecture through a rigid-object manipulation example. The *visual processing agent* identifies task-relevant objects, the manipulated object, and their support relations, then selects clean keyframes and segments the actors and objects. It rejects poor masks, reconstructs and scales meshes, tracks 6-DoF object poses, checks scale and tracking quality, and emits a canonical episode folder. The *physical-prior inference agent* estimates each object's identity, material, mass, and contact attributes for simulation, sharing the spirit of NeRF2Physics [25]. The *scene preparation agent* selects the primary camera during preprocessing, calibrates the cameras, aligns the robot base, objects, trajectories, and ground reference, and loads the assembled scene into MuJoCo. Finally, the *simulator-in-the-loop grasp optimization agent* tests small shifts of the manipulated object and retains the placement that produces a successful grasp. The tools and skills exposed to these agents are listed in Table I. The system separates agentic decisions from low-level tools: agents resolve ambiguous evidence, choose episode adapters, and decide how to refine a result, while specialized tools handle extraction, rendering, pose optimization, and grasp optimization. This design is consistent with prior agentic generation systems such as SimWorld Studio and ArtiCraft [9], [17]. We use LangChain as our agentic harness, also used by prior agentic research frameworks [26], for model interfaces, tool calls, and agent orchestration. This infrastructural setup allows us to clearly separate different layers— prompts and skills vs. tools, and easily switch between VLM backends, thereby focusing on episode reconstruction and real-sim alignment.

B. Converting DROID Episodes

Our framework primarily focuses on DROID-style robot-object interactions [10]. The input contains synchronized RGBD streams, calibration data, a robot trajectory, and optionally task description. The visual pipeline first selects the primary external camera from the synchronized streams, then extracts frames from its recording and builds a rectified video. A VLM-driven object discovery node proposes scene entities, after which a keyframe selector chooses a frame for segmentation. A mask critic can reject poor segmentation and route the graph back to keyframe selection with a bounded retry budget. Accepted masks feed mesh recovery, while full-image depth reconstruction [20] further enables mesh scaling and FoundationPose-style object tracking [21]. A tracking critic records pose-quality judgments and can request a new initialization frame. Finally, two VLM queries identify the

TABLE I

TOOLS AND SKILLS EXPOSED TO THE DROID CONVERSION AGENTS.

Agent	Tools and skills
Visual processing agent	`object_discovery`, `object_relevance`, `pickup_object`, and `support_tree`; `segment` *(SAM 3)* with keyframe selection and a mask critic; `mesh_recover` *(SAM 3D)* and `mesh_scale` with `scale_critic`; `pose_tracking` *(FoundationPose)* with `tracking_critic`.
Physical-prior inference agent	`material_classify`.
Scene preparation agent	`camera_select`, `ground_ref`, `calibration`, and base-pose optimization.
Simulator-in-the-loop grasp optimization agent	`grasp_sweep`.

object the robot manipulates and the scene entity that serves as the ground reference.

The resolved visual state is written into an episode folder containing object meshes, scales, pose tracks, robot trajectory, camera metadata, first-frame observations, masks, and task semantics. Next, the scene preparation agent proceeds: it invokes a deterministic calibration substage, optimizes the robot base pose against the robot mask, and estimates the ground plane's orientation and offset beneath the ground reference chosen upstream. The agent then loads the calibrated scene into MuJoCo [11], and makes a decision: either it performs a deterministic sweep over object shift parameters, or it calls a VLM-assisted loop that proposes refinements from replay evidence. The sweep path evaluates a batch of candidate shifts against contact and grasp outcomes; the loop path exposes rendered keyframes and structured replay summaries to the agent before each refinement.

Photorealistic rendering. Agentic Real2Sim combines an interactive foreground with a re-renderable visual context, following the layered rendering strategy of RoboSnap [27]: manipulated objects in the foreground reuse the appearance textures generated by SAM 3D, while the background rendering routine completes regions occluded by the interaction area, reconstructs the static scene as a Gaussian-splatting representation, and registers it to the episode camera and world frames. For a queried camera, the simulator renders foreground color, depth, and alpha channels, and the Gaussian-splatting renderer produces the background view from the same camera pose. Depth-aware compositing combines these layers into photorealistic frames used as policy observations and qualitative real-to-sim comparisons.

C. Converting PhysTwin and BFM-Zero Episodes

We extend the same episode contract beyond rigid DROID manipulation. For deformables, following the PhysTwin/EMPM setting [3], [4], the framework replaces rigid object pose tracking with tracked geometry, point, particle, spring, or material state and using simulator rollouts to check whether recovered parameters reproduce observed deformation. For converting humanoid motion from [12], the framework replaces object-centric scene preparation with motion-context retrieval, embodiment-specific initialization, and closed-loop replay against retargeted reference motion, using motion priors from the humanoid/video domain [28]. For both deformable and humanoid motion conversion tasks, the framework uses the same agentic workflow, while only exposing domain-specific skills and tools, without forcing all episodes into rigid-body representation.

IV. EXPERIMENTS

Prior work have studied critic-guided refinement through critic removal in SceneSmith [8] and reflection ablations in SceneWeaver [16]. We adopt this existing design and focus our evaluation on episodic conversion capabilities, VLM cost, and downstream uses. We therefore do not include a separate actor–critic ablation.

A. Evaluation Metrics

VLM as judge. We evaluate the rigid DROID conversions with replay success, an episode-level metric that measures consistency between the simulated episode and the real episode. Specifically, following the paradigm adopted in [29], we compute the metric following a structured, VLM-based process: an evaluator first prepares a fixed set of real and simulated keyframes with the labels `start`, `middle_1`, `middle_2`, and `end`. Next, the evaluator selects reconstruction candidates deterministically from the latest grasp sweep: a sample is eligible only if the probe marks it as grasped, its replay video exists, and its lift and displacement statistics are finite and within broad sanity bounds. Among the eligible samples, the evaluator sends at most five candidates to the judges, ranked by peak object displacement with sample id as the deterministic tie-breaker. We use three frontier models from different labs as judges: GPT-5.5 (OpenAI), Claude Opus 4.7 (Anthropic), and Gemini 3.5 Flash (Google); judges are deliberately chosen to avoid overlap with the pipeline backend. Each judge independently scores each candidate out of 10, subtracting up to 4 points for identifying the wrong target object, 3 for a wrong final object location, 2 for a wrong action performed, and 1 for a wrong final gripper location. Starting-pose drift is recorded as context rather than directly subtracted; scores of 8 or higher count as pass, 7 as partial, and 6 or lower as fail. Each judge acts as a success finder by nominating its own highest-scored candidate. We define $r_e \in \{0, 1\}$ as the episode replay-success indicator, with $r_e = 1$ if at least one of the three judges assigns a best-candidate score of at least 8 out of 10; otherwise $r_e = 0$. Episodes without a valid judge record are also assigned $r_e = 0$. The reported replay score for the episode is the maximum of the three judge-best scores.

Visual and Pose Metrics. We supplement the VLM-based evaluation with deterministic visual- and pose-based metrics: full-frame PSNR and SSIM for visual fidelity, and ADD-S in mm for pose trajectory accuracy. These metrics were applied for episodes that produced evaluable artifacts.

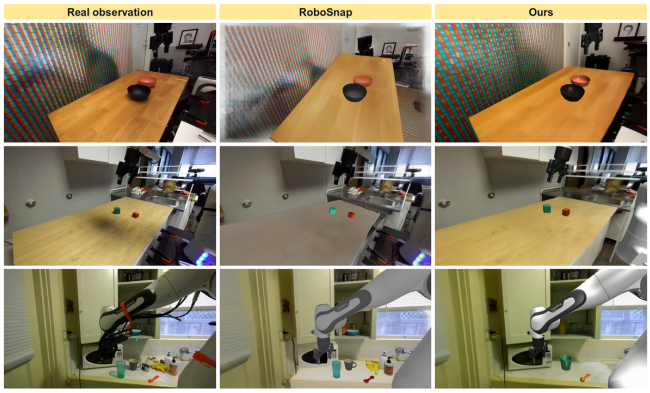

Fig. 3. **Scene reconstruction, ours vs Robosnap.**

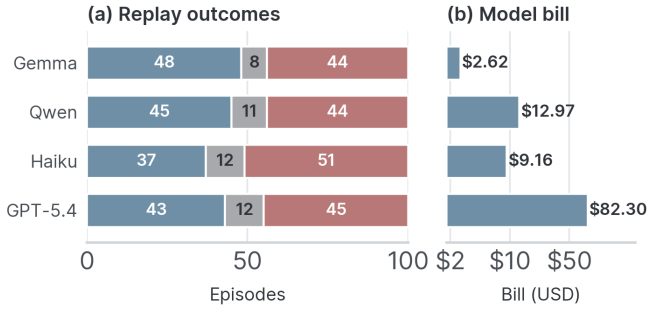

Fig. 4. **Replay outcomes and model bill across VLM backends.** Blue, gray, and red denote replay success ($\geq 8/10$), partial ($7/10$), and failure ($\leq 6/10$ or no valid judge record), respectively. The right panel reports the model bill (USD, log scale) for 100 episodes per backend.

B. DROID Episode Conversion at Scale

We evaluate the rigid object manipulation setting on a randomly selected set of 500 manipulation episodes sampled to span objects, camera viewpoints, occlusion patterns, and manipulation verbs (pick / place / take / remove). All sampled episodes remain in the replay-success denominator, including runs that stop before producing a valid replay record.

Quantitative. Using Gemma 4 31B as the VLM backend, Agentic Real2Sim produces 242 (48.4%) successful, 28 (5.6%) partial, and 230 (46.0%) failed replay outcomes. Among 349 evaluable episodes, we found a mean PSNR of 14.069, SSIM of 0.650, and ADD-S of 107mm.

Qualitative. Fig. 2 pairs real-world observations with their MuJoCo replays for three representative episodes, exposing intermediate pipeline artifacts, e.g. recovered object meshes, FoundationPose-style pose-tracking overlays, and the calibrated robot/ground state. Further, we qualitatively compare our reconstruction of three static scenes from the DROID dataset with the concurrent Robosnap [27]: see Fig. 3. Notice that our results are comparable to those obtained by Robosnap in terms of visual quality; while for the top and the middle episode our reconstruction attain slightly better texture and pose estimates, for the bottom scene Robosnap yields closer-to-ground truth pose and shape estimate of the cup.

Common Failure Modes. Reasons for failure included: objects being nudged, not lifted by the gripper; incomplete segmentation/mesh-recovery (e.g. critical object not segmented, incorrect object chosen for recovery, or a single mesh combining 2+ objects); objects' initial position, scale, or orientation being incorrect, precluding gripping. Factors such as a cluttered scene, occlusion, or low lighting made the conversion less reliable.

C. Multi-VLM Support and Cost Efficiency

Across the four backends assessed, we use the same pipeline configuration, with reasoning enabled for selected agentic steps, and vary only the VLM; all VLM backends are evaluated on the same subset of 100 DROID episodes. The reported model bills include each model's API usage under this configuration. Fig. 4 shows that the four backends have broadly similar observed outcome compositions: Gemma 4 31B, Qwen 3.6 35B, GPT-5.4, and Claude Haiku 4.5 obtain $48/100$, $45/100$, $43/100$, and $37/100$ replay successes, respectively. The clearer separation is cost: relative to Gemma's 2.62 model bill, Claude costs $3.5\times$ as much, Qwen $5.0\times$, and GPT-5.4 $31.4\times$.

We attribute the similar outcomes across backends to how the pipeline scopes the VLM's role. A VLM does not generate geometry or simulate physics itself; it simply orchestrates low-level specialized tools for segmentation, stereo depth, pose tracking and deterministic grasp sweep. It is used only for scope-bounded, schema-constrained decisions such as object discovery, keyframe and mask selection under a capped retry budget, and replay-refinement choices. Because these queries are narrowly scoped, they lie well within the capability of modern reasoning VLMs, and the pipeline transfers across backends without requiring per-model tuning: even the 31B open model attains the highest observed success count while using roughly 3% of GPT-5.4's model bill. Backend choice therefore reduces largely to cost, making open-weight models a practical default for large conversion batches. The replay success stays below 50% for every backend further suggests that the remaining headroom lies mostly in the upstream visual and simulation components rather than in the choice of VLM alone.

One-prompt agent baseline. These backend comparisons leave open whether Gemma can complete episode conversion without our workflow. We therefore tested Gemma 4 31B in Codex on three DROID episodes previously successfully converted with Gemma and Agentic Real2Sim. Each episode received one independent attempt from an empty Python environment, with its images, robot trajectories, calibration and robot assets, plus a common task prompt and output contract. The agent had up to 60 minutes to install dependencies, call tools and revise its code, without AR2S-specific stages, tools or skills. All three sessions ended before the limit, but none delivered a complete, runnable episode twin. Failures included invalid scene XML, replay errors, stationary task objects and unstable robot states. Together with the backend

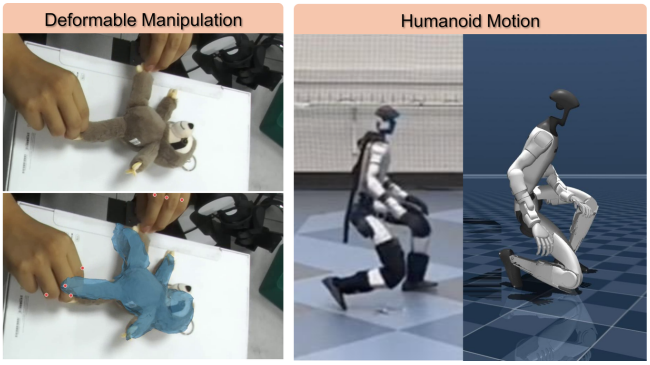

Fig. 5. **Deformable and humanoid episode conversion.** Two non-rigid object manipulation settings that stress the same conversion contract beyond rigid manipulation. *Left:* PhysTwin-style deformable episodes pair real interaction video with the recovered geometry and a deformation overlay. *Right:* Unitree G1 humanoid clips of locomotion and short whole-body motions, retargeted from LAFAN1, where the simulated humanoid is shown against the real frames after the controller's joint-level gains are tuned to match the reference trajectory in closed-loop simulation.

comparison, these observations support the practical value of a reproducible workflow that works across VLMs, including open-weight models.

D. Deformable and Humanoid Episode Conversion

To show our agentic workflow generalizes to deformables, we convert PhysTwin-style episodes with assets such as rope, cloth, plush objects, soft packages, and elastoplastic materials, and compare the real interaction video with the recovered geometry and simulated deformation. For humanoid motion's twin generation: the input data are Unitree G1 clips of locomotion and short whole-body motions, retargeted from LAFAN1 [28]. We compare the simulated humanoid against the reference motion in closed-loop simulation and report qualitative side-by-side motion comparisons. Fig. 5 shows both domains. In Fig. 5 (a), deformable cases pair real interaction video with tracked geometry and a deformation overlay, emphasizing whether the recovered material response follows the visible bending, stretching, and contact sequence. In Fig. 5 (b), we show a simulated humanoid against its reference motion for standing, kneeling, and short-locomotion clips, emphasizing gross pose, balance, and motion-phase agreement.

E. Robotic Applications

Custom scene conversion. Beyond converting the DROID recordings, we validate that our framework extends to data collected on a robot platform that does not appear in DROID. Our capture setup consists of an AIRBOT Play arm [30]— a six-DoF manipulator equipped with the AIRBOT G2 two-finger parallel-jaw gripper, and a single externally mounted ZED 2i stereo camera overlooking the tabletop workspace. Episodes are collected by a human operator who teleoperates the arm through a VR-headset interface (Fig. 6, left). The camera records rectified stereo pairs at 2208×1242 resolution and 15 Hz; arm joint positions and the gripper opening are logged alongside; and the camera-to-robot-base transform is obtained from a one-time station calibration. We package these raw recordings into the DROID episode layout consumed by the pipeline, leaving its ingestion path unchanged; the only embodiment-specific change is a MuJoCo model of the arm and gripper, converted from the vendor-provided URDF and registered with the simulation stage in place of the default DROID Franka model. In the recorded episode, the operator drives the arm to pick a wooden spoon out of a plate resting on the table and set it down on the tabletop beside the plate. Given the packaged episode, the pipeline runs end to end, from object discovery to mesh recovery, followed by scene preparation with simulator-in-the-loop grasp optimization. This yields a MuJoCo twin in which the AIRBOT arm replays the recorded joint trajectory, grasps the spoon, lifts it clear of the plate, and relocates it onto the table, reproducing the outcome of the real episode. Fig. 6 pairs keyframes of the real recording with the synthetic twin rendered from the calibrated real camera viewpoint.

Data augmentation, in-sim policy training and evaluation. An episode twin also serves as a basis to generate training data for control policies. Each converted episode includes a grasp demonstration validated through simulator-in-the-loop grasp optimization. Replaying this demonstration in the twin yields synchronized training trajectories comprising composited exterior and wrist observations, proprioceptive states, and actions in the DROID convention, without additional physical data collection. Because the twin exposes the full simulation state and supports rerendering, each demonstration can be expanded through controlled augmentations that are difficult to apply consistently to a fixed real-world recording—see Fig. 7 (b). These augmentations include *object pose randomization*, in which objects are relocated and the end-effector trajectory is recomputed through inverse kinematics; *grasp resampling*, which uses alternative validated grasp points from the grasp sweep to produce distinct demonstrations of the same task; *background replacement*, which substitutes the 3DGS background from another episode; and *camera viewpoint perturbation*, which rerenders both views from perturbed camera poses. Each variant is resimulated and rerendered rather than edited at the image level, preserving physical and geometric consistency.

The augmented episodes are then used to fine-tune a pretrained $\pi_{0.5}$ policy [24]. In terms of the setup of the evaluation environment, calibrated MuJoCo arm and gripper are positioned in the recovered scene frame, and reconstructed object meshes are initialized at their grasp-optimized poses. The arm and gripper are then simulated under position control at 15 Hz. A wrist-mounted camera, positioned according to the real wrist-camera calibration, completes the DROID camera configuration. Evaluation-related API's are exposed through a Gym-style interface, enabling VLA-based manipulation policies to be evaluated with minimal task-specific integration. We evaluate the resulting environment using the DROID-pretrained VLA model $\pi_{0.5}$ [24], configured to predict absolute joint positions. At each control step, the policy receives the composited exterior and wrist RGB views, the

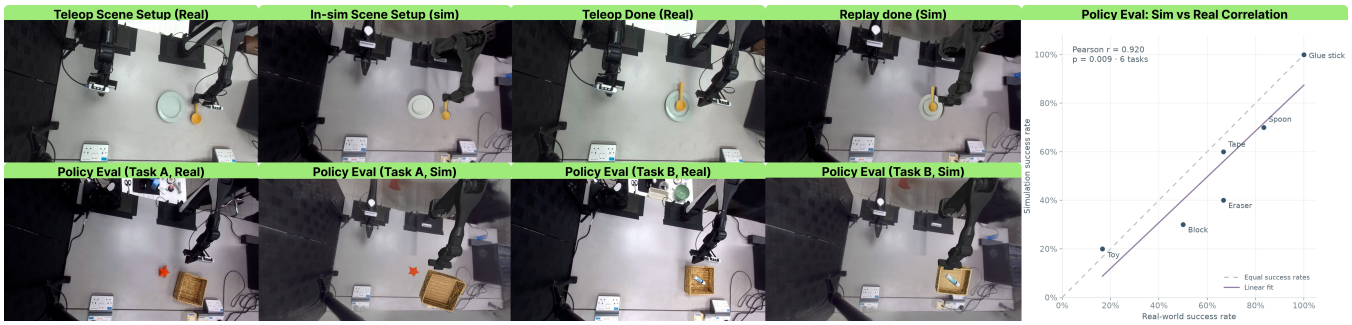

Fig. 6. **Real-world experiments.** *Top:* a teloperation experiment in which a human operates the AIRBOT Play arm through a VR-headset interface to perform the put-spoon-on-plate task, and we reconstruct the episode as a simulated twin. *Bottom* We evaluate a pi-0 policy [24] fine-tuned on real-world data in both real-world task and their simulated counterparts. Two tasks are shown here. Success rate in simulation vs. in real world are shown on the right.

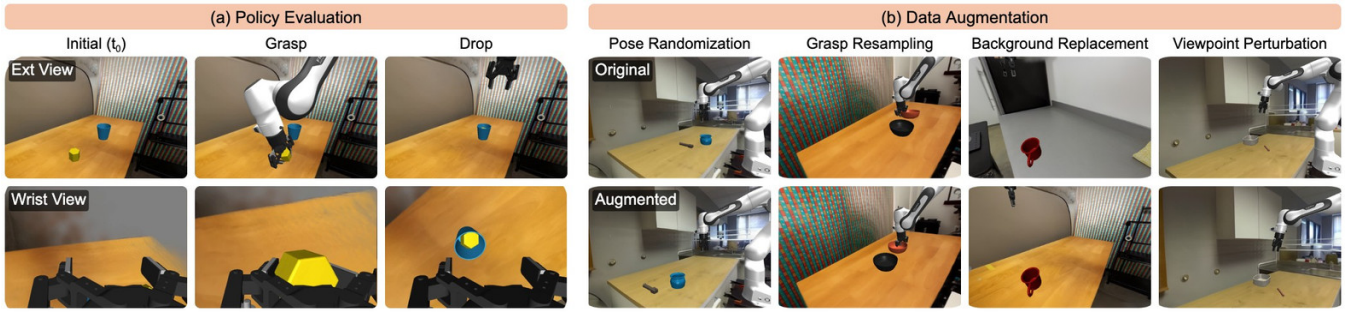

Fig. 7. **Policy evaluation and data augmentation in Agentic Real2Sim episode twins.** (a) Keyframes (initial, grasp, drop) of a closed-loop $\pi_{0.5}$ rollout in a converted twin, shown from the policy's two composited input views (exterior, top; wrist, bottom). (b) Augmentations applied to a converted twin (top: original; bottom: augmented): pose randomization, grasp resampling, background replacement, and viewpoint perturbation. Variants are re-simulated and re-rendered inside the twin rather than image-edited, so every augmented trajectory remains physically and geometrically consistent.

robot joint and gripper states, and the episode's natural-language task description. It predicts a chunk of absolute joint-position commands, which the environment executes open-loop before querying the policy again; thus, the rollout is closed-loop at the action-chunk level. Fig. 7 (a) shows a rollout in a converted twin from the policy's two input views: the policy locates and grasps the target object, then drops it into the cup.

Substituting real-world evaluation. Using these reconstructed episode twins, we investigate whether simulation can serve as a proxy for real-world policy evaluation. This setting is deliberately separated from the augmentation experiment above: the policy evaluated here does not use synthetic trajectories generated by our framework during training; instead, we fine-tune a π_0 policy [31] exclusively on real-world demonstrations, and execute the same policy in both a hold-out set of six pick-and-place tasks and their reconstructed simulated counterparts. This allows us to directly test assess policy performance measured in simulation is predictive of its performance on the real robot. Since the policy has never been trained on trajectories generated by the reconstructed twins, agreement between the two domains cannot be attributed to exposure to the simulated evaluation environments during training.

The bottom row of Fig. 6 shows representative closed-loop policy evaluations for two tasks. Note that the simulated rollout is not a replay of the real policy trajectory: the policy receives observations from the simulated environment and produces actions online, so deviations in visual appearance, object geometry, contact, or robot dynamics can directly affect the resulting task outcome. These deviations may also accumulate over the course of a rollout, making task success a stricter test of the reconstructed environment than matching individual rendered frames or replaying a fixed action sequence.

We evaluate this correspondence across six pick-and-place tasks, and compare task success rates measured in simulation against those measured in the real world. As shown on the right of Fig. 6, simulation and real-world performance exhibit a strong Pearson correlation of $r = 0.920$ ($p = 0.009$). Although the absolute success rates do not exactly match for every task, simulation largely preserves the relative performance across tasks, with easier and harder tasks showing consistent trends in both domains.

These results indicate that the reconstructed episode twins preserve task-relevant visual and physical factors sufficiently well for closed-loop policy evaluation. In practice, such correspondence enables simulation to act as an intermediate evaluation stage for screening policy variants, identifying promising candidates for deployment, and estimating expected real-world performance before committing to repeated physical trials.

V. Conclusions, Limitations and Future Work

We introduced *Agentic Real2Sim*, a framework for converting real-world physical interaction recordings into simulatable digital twins. The framework preserves the actors,

manipulated entities, geometry, physical parameters, camera setup and objects along with end effector trajectories needed for simulation. For the task of converting DROID episodes, Agentic Real2Sim converts robot-object manipulation recordings into MuJoCo episode twins through visual processing, physical-prior inference, scene preparation, and simulator-in-the-loop refinement. The same episode contract also provides a common interface for PhysTwin-style deformable manipulation and BFM-Zero-style humanoid motion, allowing the pipeline to span settings that are usually handled by separate Real2Sim systems. Also, our framework decouples agentic decisions from any single VLM provider: on DROID-500, an open 31B backend attains comparable observed replay-success outcomes to proprietary alternatives while using only about 3% of GPT-5.4's model cost. Finally, the framework supports custom scene conversion on manipulation data outside the DROID dataset, while the aligned episode twins support downstream policy evaluation and provide generated interaction data for fine-tuning a DROID-pretrained $\pi_{0.5}$ policy [24].

Our work still has many limitations. First, it focus on rigid-object DROID episodes, and there is still plenty of room for improvement in terms of successful conversion rate. Second, the framework does not support object-level system identification or directly validate physical parameters, since each converted demonstration provides limited interaction evidence per object. Future work will extend automated conversion to additional deformable episodes and systematically evaluate agentic components and VLM backends for reliability and replay stability.

Acknowledgment

This work was supported in part by the NVIDIA Academic Grant Program. We thank NVIDIA for supporting academic research and for providing resources that helped facilitate this work. We thank Yiwen Chen for his help with data collection; We also thank Beichen Li of OpenAI for support with computational resources.

References

[1] N. Pfaff *et al.*, "Scalable real2sim: Physics-aware asset generation via robotic pick-and-place setups," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct 2025, p. 6296–6303.

[2] H. Fan *et al.*, "Twinaligner: Visual-dynamic alignment empowers physics-aware real2sim2real for robotic manipulation," *arXiv preprint arXiv:2512.19390*, 2025.

[3] H. Jiang *et al.*, "Phystwin: Physics-informed reconstruction and simulation of deformable objects from videos," in *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct 2025, p. 7219–7230.

[4] Y. Chen *et al.*, "Empm: Embodied mpm for modeling and simulation of deformable objects," *IEEE Robotics and Automation Letters*, vol. 11, no. 4, p. 4179–4186, Apr 2026.

[5] W. Ma *et al.*, "Lychsim: A controllable and interactive simulation framework for vision research," *arXiv preprint arXiv:2605.12449*, 2026.

[6] Y. Wang *et al.*, "Physcensis: Physics-augmented llm agents for complex physical scene arrangement," *arXiv preprint arXiv:2602.14968*, 2026.

[7] A. Zook *et al.*, "Grs: Generating robotic simulation tasks from real-world images," in *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, June 2025, p. 594–603.

[8] N. Pfaff *et al.*, "Scenesmith: Agentic generation of simulation-ready indoor scenes," *arXiv preprint arXiv:2602.09153*, 2026.

[9] M. Zhou *et al.*, "Articraft: An agentic system for scalable articulated 3d asset generation," *arXiv preprint arXiv:2605.15187*, 2026.

[10] A. Khazatsky *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," in *Robotics: Science and Systems XX*, ser. RSS2024. Robotics: Science and Systems Foundation, July 2024.

[11] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Oct 2012, p. 5026–5033.

[12] Y. Li *et al.*, "Bfm-zero: A promptable behavioral foundation model for humanoid control using unsupervised reinforcement learning," 2025.

[13] Y. Yang *et al.*, "Holodeck: Language guided generation of 3d embodied ai environments," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2024, p. 16277–16287.

[14] P. Katara, Z. Xian, and K. Fragkiadaki, "Gen2sim: Scaling up robot learning in simulation with generative models," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2024, p. 6672–6679.

[15] N. Ranawaka *et al.*, "Simfoundry: Modular and automated scene generation for policy learning and evaluation," *arXiv preprint arXiv:2606.28276*, 2026.

[16] Y. Yang *et al.*, "Sceneweaver: All-in-one 3d scene synthesis with an extensible and self-reflective agent," *Advances in neural information processing systems*, vol. 38, pp. 140 319–140 351, 2026.

[17] H. Kang *et al.*, "Simworld studio: Automatic environment generation with evolving coding agent for embodied agent learning," *arXiv preprint arXiv:2605.09423*, 2026.

[18] N. Carion *et al.*, "Sam 3: Segment anything with concepts," *arXiv preprint arXiv:2511.16719*, 2025.

[19] X. Chen *et al.*, "Sam 3d: 3dfy anything in images," *arXiv preprint arXiv:2511.16624*, 2025.

[20] B. Wen *et al.*, "Foundationstereo: Zero-shot stereo matching," *CVPR*, 2025.

[21] B. Wen *et al.*, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2024, p. 17868–17879.

[22] C. Garrett *et al.*, "Skillmimicgen: Automated demonstration generation for efficient skill learning and deployment," in *8th Annual Conference on Robot Learning*, 2024.

[23] W. Wan *et al.*, "Lodestar: Long-horizon dexterity via synthetic data augmentation from human demonstrations," in *9th Annual Conference on Robot Learning*, 2025.

[24] Physical Intelligence *et al.*, "$\pi_{0.5}$: a vision-language-action model with open-world generalization," *arXiv preprint arXiv:2504.16054*, 2025.

[25] A. J. Zhai *et al.*, "Physical property understanding from language-embedded feature fields," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2024, pp. 28 296–28 305.

[26] A. M. Bran *et al.*, "Chemcrow: Augmenting large-language models with chemistry tools," *arXiv preprint arXiv:2304.05376*, 2023.

[27] S. Zhang *et al.*, "Robosnap: One-shot real-to-sim scene generation for generalizable robot learning and evaluation," *arXiv preprint arXiv:2607.06699*, 2026.

[28] F. G. Harvey *et al.*, "Robust motion in-betweening," *ACM Transactions on Graphics*, vol. 39, no. 4, Aug. 2020.

[29] Z. Cao *et al.*, "Physx-omni: Unified simulation-ready physical 3d generation for rigid, deformable, and articulated objects," *arXiv preprint arXiv:2605.21572*, 2026.

[30] DISCOVER Robotics, "AIRBOT: Embodied intelligence robot platform," https://airbots.online/, 2026, accessed: 2026-09-13.

[31] K. Black *et al.*, "π_0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.