arXiv:2610.18835v2 [cs.RO] 14 Oct 2026

RoboSimStudio: An Interactive Toolkit for Rapid Real2Sim of Robot Manipulation Tasks

Kaiyuan Chen[2*], **Shuangyu Xie**[2*], **Uksang Yoo**[2],
Xinyue Zhu[2], Cael Magner[2], Alvin Zhu[2], Daehwa Kim[2],
Eric Heiden[1], Erwin Coumans[1], Miles Macklin[1], Spencer Huang[1], Ken Goldberg[2]
*Equal contribution
[1]NVIDIA [2]University of California, Berkeley

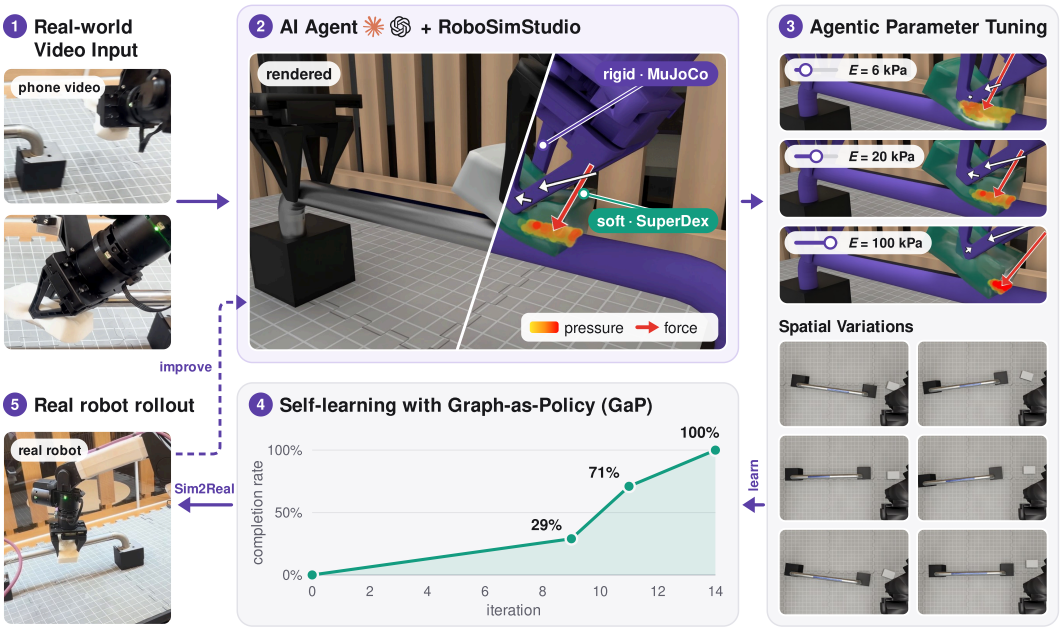


Figure 1: **RoboSimStudio** takes an input a smartphone video of a robot or human performing a task and outputs a multi-physics simulation that humans and coding agents can inspect and tune. For example, given the video of a sponge-wiping task, RoboSimStudio uses Claude [1] or Codex [2] agents build the simulation. RoboSimStudio couples the physics through Newton [3] and the sponge with SuperDex [4]. A human or a coding agent can interact with the simulation and tunes its solver, contact model, and parameters in a loop, until its behavior matches the video. For robustness, the simulation is then varied in its physical parameters (e.g. Young's modulus E) and its spatial layout. With the varied spatial layout and physical parameters, Graph-as-Policy [5] uses simulation feedback to improve success rate and cycle time before the policy runs on the real robot.

Abstract: It has been difficult to accurately model the physics of the interaction between a robot and the objects it manipulates, including contact mechanics, coupled contact dynamics, deformations. stiffness and friction. We present RoboSimStudio (RSS), an interactive toolkit that makes choices of solver, contact model, and physical parameters explicit and editable, allowing both humans and coding agents to quickly create and iteratively refine simulations based on observed physical behavior. We implement RSS with the open Newton physics simulation system and describe experiments with human-guided real2sim using (1) visual evaluation, (2) force-feedback and (3) agentic Real2Sim2Real. Code, video and additional information can be found on the project website: https://robosimstudio.github.io

1 Introduction

Simulating robot manipulation requires modeling the interaction between a robot and the objects it manipulates, including robot kinematics, coupled contact dynamics, and deformation (Figure 7b). Emerging simulation frameworks, such as Newton [3], Genesis [6], and SuperDex [4], increasingly cover these different forms of physics by supporting rigid, articulated, and deformable-body simulation within a common framework. However, accurately modeling robot manipulation behavior to match real-world physics, or Real2Sim, still requires choosing appropriate physics models, contact models, and physical parameters, and iteratively refining these choices against observed behavior. In this paper, we extend existing simulation frameworks to streamline Real2Sim.

We recognize that both humans and LLM coding agents share a fundamental interaction pattern for Real2Sim: they modify a scene, execute physical interactions, observe discrepancies, and refine the model based on feedback. Specifically, humans perform this loop through direct manipulation and immediate visual or force feedback, while coding agents [1, 2] use iterative reasoning and acting [7] to inspect simulator states and revise scene configurations and robot behaviors. Motivated by this insight, we make multi-physics simulations directly interactive and editable by both humans and coding agents to enable rapid iterative Real2Sim refinement.

We present RoboSimStudio (RSS), an interactive toolkit for physics-based simulation of robot manipulation tasks. It uses Viser [8] and Newton [3] for multi-physics simulation, supports co-simulation with external physics engines, and makes choices of solver, contact model, and physical parameters explicit and editable. An interactive display and sliders allow a human or AI coding agent [1, 2] to rapidly adjust parameters to align with a real robot and environment. For coding agents, RoboSimStudio exposes simulator state together with spatial-temporal text and visual feedback, allowing agents to observe the consequences of their changes and iteratively refine the simulation.

We conduct evaluation on RoboSimStudio through human-guided and agentic Real2Sim experiments. Using the interactive studio, we construct 18 contact-rich manipulation simulations involving rigid objects, articulated mechanisms, cables, cloth, and soft-body deformation. We further demonstrate force-feedback interaction, where a human operator can control a simulated robot and perceive contact forces through a low-cost leader arm. Finally, we conduct a pilot study of agentic Real2Sim2Real using a bimanual sponge-wiping task (Figure 1). Given a smartphone video of the real task, off-the-shelf coding agents construct multi-physics simulations. A Graph-as-Policy (GaP) [5] is then improved using simulation feedback and transferred to the real robot. We also demonstrate how a real-world failure can be used to guide an agent to revise the simulation's physical constraints.

In summary, this work makes three contributions: (1) RoboSimStudio, a simulation toolkit with configurable contact models and physical parameters for contact-rich robot manipulation; (2) Interactive interfaces on RoboSimStudio for humans and coding agents that streamline Real2Sim process; (3) 18 manipulation tasks with visual and force-feedback interaction, and a pilot study on real2sim2real with RoboSimStudio that coding agents construct a multi-physics simulation from a smartphone video.

2 Related Work

Physics Simulation for Robot Learning. Physics simulation has enabled large-scale robot learning by providing environments for policy training and evaluation. MuJoCo [9] supports articulated robot dynamics and contact-rich manipulation, while GPU-accelerated frameworks such as Isaac Gym and Isaac Lab [10, 11, 12], ManiSkill3 [13], MuJoCo Playground [14], and Genesis [6] support parallel simulation for robot learning. Frameworks such as robosuite [15] further provide standardized manipulation environments. More recently, Newton [3], built on Warp [16], integrates multiple physics solvers, including MuJoCo-Warp [17], Vertex Block Descent (VBD) [18], and XPBD [19]. Other physics engines, such as SuperDex [4], provide specialized models for contact-rich soft-body

Table 1: **Comparison of robotics simulation frameworks.** Comparison of physics capabilities, interactive interfaces, and Real2Sim workflows. ✓ indicates built-in support, ∼ indicates partial support, and "Custom" indicates that additional implementation is required.

Feature	MuJoCo	IsaacLab	ManiSkill3	Genesis	Newton	RoboSimStudio
Physics Simulation						
Rigid/articulated bodies	✓	✓	✓	✓	✓	✓
Deformable objects	✓	✓	∼	✓	✓	✓
Coupled multi-physics solvers	–	∼	–	✓	∼	✓
Interactive Interfaces						
Interactive scene inspection	✓	✓	✓	✓	✓	✓
Physics parameter tuning	∼	✓	∼	∼	∼	✓
Structured feedback for coding agents	Custom	Custom	Custom	Custom	Custom	✓
Shared human-agent refinement	Custom	Custom	Custom	Custom	Custom	✓
Real2Sim Workflows						
Video-driven agentic Real2Sim	–	–	–	–	–	✓
Force-feedback Real2Sim	–	∼	–	–	–	✓
Failure-driven simulation refinement	–	–	–	–	–	✓

simulation. While these frameworks provide increasingly diverse physics capabilities, constructing manipulation environments still requires selecting appropriate solvers, configuring contact models, and tuning physical parameters. RoboSimStudio builds on Newton and supports co-simulation with external physics engines, allowing different physics models to be composed within the same scene and interactively refined by humans and coding agents.

Agentic Robotics. Large language models (LLMs) are now being used in robotics for task planning, robot control, and policy generation. Early approaches use language models to ground instructions in robot affordances [20], generate plans and executable robot programs [21, 22, 23, 24], and improve robot behavior through feedback [25]. More recent works use LLMs to generate reward functions [26, 27], visual question answering [28, 29], domain randomization strategies [30], and simulation tasks for robot learning [31, 32, 33]. With iterative reasoning and acting frameworks such as ReAct [7], general-purpose coding agents [1, 2] can modify code, execute programs, and refine their solutions based on feedback. Such agents have been applied to reconstructing articulated objects [34], robot data collection [35], and iterative policy improvement, such as CaP-X [36] and GaP [5]. MAESTRO [37] orchestrates robotics modules through VLM-written programs, ASPIRE [38] grows a reusable code skill library across many simulated tasks, and ENPIRE [39] lets coding agents improve policies through real-world trial and error. DexAgent [40] turns a single egocentric human video into dexterous robot trajectories, and RPG [41] practices tasks in simulation and revises skills from diagnosed failures before real-world deployment. A coding agent has also solved the Push-T benchmark without demonstrations by running its own experiments in simulation [42]. The effectiveness of these agents also depends on their ability to interact with external environments and access execution feedback [43]. RoboSimStudio builds on this insight by providing coding agents with an interactive physics simulation environment, where scene configurations, physics models, and simulation feedback are directly accessible for agentic Real2Sim construction and refinement.

Real2Sim. Real2Sim approaches reconstruct simulation environments from real-world observations to support robot policy training and evaluation. Prior work has explored constructing digital twins and digital cousins from images, videos, and other observations [44, 45, 46, 47, 48, 49]. Other approaches use generative models and LLMs to automatically construct simulation tasks, environments, and reward functions [31, 50, 51, 27], or generate robot policies directly [21, 5]. Blox-Net [52] uses vision-language models to design block assemblies, refines the designs through perturbation analysis in simulation, and builds them with a real robot. These approaches reduce the effort required for scene construction and policy development, but often focus on specific re-

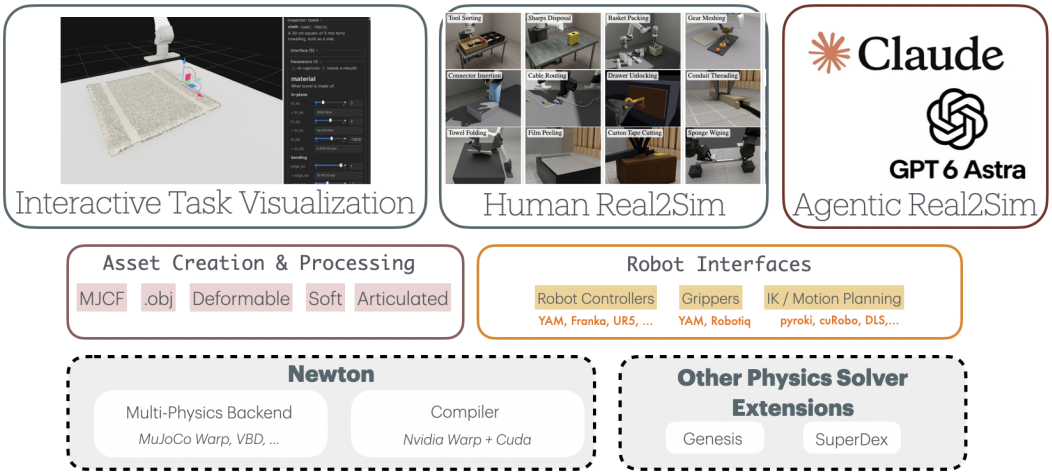


Figure 2: **RoboSimStudio Architecture.** RoboSimStudio integrates asset creation, robot interfaces, and multi-physics simulation through Newton and external physics solvers. It supports interactive task visualization, human-guided Real2Sim, and agentic Real2Sim through a shared simulation framework.

construction or generation pipelines. Recent frontier models, such as Claude Fable 5.1 [1], GPT-6 Astra [2], and Gemini 4 Argon [53], can assist in reconstructing visually plausible 3D scenes from videos, including producing Blender [54] renderings. However, visual realism does not guarantee physical accuracy, particularly for contact-rich interactions and deformable objects such as cables, cloth, and sponges. RoboSimStudio therefore exposes editable physics models, contact parameters, and simulator feedback, enabling humans and coding agents to iteratively refine simulated physical behavior to match real-world observations rather than appearance alone.

3 RoboSimStudio

Figure 2 shows the architecture of RoboSimStudio. Physics is mainly provided by Newton [3] and its solvers, with external physics engines that can be co-simulated by exchanging contact and forces. RoboSimStudio composes a scene with robots, objects, and physics into one simulation while keeping modeling choices, including the solver, contact model, and physical parameters, explicit and editable. The composed scene can be accessed through code, an interactive studio for a human, or an iterative interface for a coding agent to support task creation and agentic real2sim.

Robot Interfaces. RoboSimStudio provides a common robot interface for Franka Panda, UR5e, and single- and bimanual YAM, and can be extended to additional robots. A robot is specified declaratively by its model, joints, mounting, and hand geometry, and exposes common interfaces for forward and inverse kinematics, motion generation, and control. Inverse kinematics supports interchangeable solvers, such as pyroki [55], while motions include joint-space and Cartesian moves, guarded contact motions, and collision-aware planning, such as cuRobo [56]. Policies can command the robot in joint or end-effector action spaces, while privileged simulator state about manipulated objects is kept separate from observations available to the robot.

Multi-Physics Simulation and Coupling. Manipulation scenes often combine articulated robots, rigid objects, and deformable materials, for which no single physical model is sufficient. In RoboSimStudio, objects specify what they are, while the scene specifies which solver models each type of physics. By default, articulated robots use MuJoCo-Warp [17], cables and cloth use Vertex Block Descent [18], and soft bodies can be simulated by external engines [4, 6] co-simulated with Newton. RoboSimStudio couples these solvers by exchanging contact forces between Newton solvers or poses and contact wrenches with external engines. The solver, coupling method, contact model, and material parameters remain explicit entries in the scene and can therefore be changed

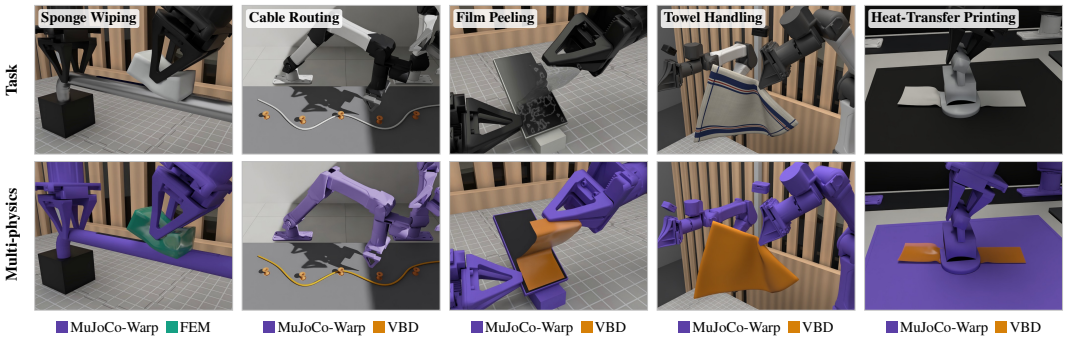


Figure 3: **Multi-physics coupling in RoboSimStudio.** Five manipulation tasks using different physics solvers and coupling methods. Top: rendered simulation scenes. Bottom: visualization of the physics solvers used for each object, including MuJoCo-Warp (violet), finite-element simulation (FEM, teal), and Vertex Block Descent (VBD, amber). All tasks use the same robot interface and scene description framework in RoboSimStudio.

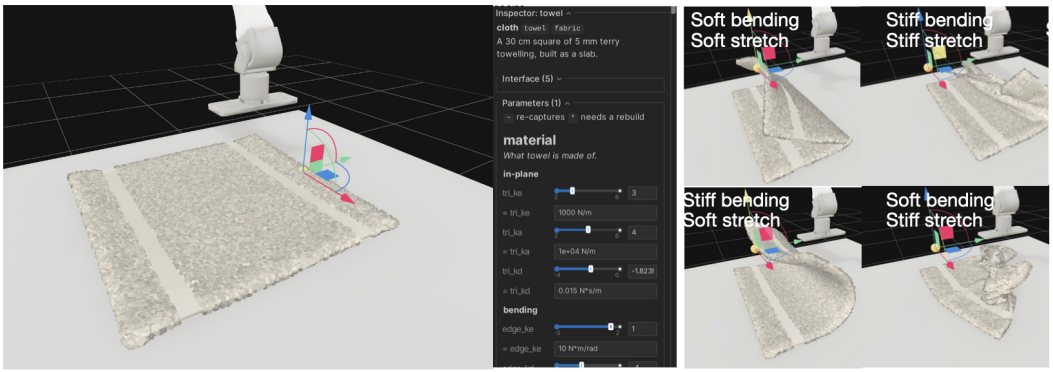


Figure 4: **RoboSimStudio Interface.** Left: a towel and its inspector with material controls, for example, tri_ke controls the stretch stiffness and edge_ke controls the bending stiffness. Right: one towel, four bending and stretch stiffness: the same dragged fold at the same instant.

without rewriting the task. Figure 3 shows five manipulation tasks that use different physics solvers and coupling methods.

Interactive Interface for Humans. Humans interact with RoboSimStudio through a browser-based studio built on Viser [8]. Objects can be selected, dragged, and inspected, and robot arms can be controlled directly, allowing a user to probe physical behavior before writing a policy. Each object's inspector exposes its physical parameters with their units and ranges. Parameters such as stiffness and damping can be changed while the simulation is running, while parameters that require rebuilding the scene are marked explicitly. Figure 4 shows the same towel fold under four choices of bending and stretch stiffness, illustrating how simulation tuning becomes direct interaction with the resulting physical behavior.

Iterative Interface for Agents. Given as input video of a real task being performed, a coding agent can tune a simulation through the same basic loop: modify the scene, execute a behavior, and observe the result. RoboSimStudio organizes a rollout into robot motion primitives and converts simulator state into spatial-temporal feedback that describes target error, object motion, contact events, grasping or slipping, and progress toward the task goal. Because this feedback is computed from simulator state, it exposes physical events such as contact that may be difficult to infer from images alone. Together with visual observations, this allows an agent to attribute an outcome to the action that caused it, modify either the policy or the simulation parameters, and run the task iteratively until the simulation behavior matches the input video.

Figure 5: **18 Challenging contact-rich deformable manipulation task simulations created by RoboSimStudio**

Human-in-the-Loop Agentic Real2Sim. RoboSimStudio allows humans and coding agents to interact with the same simulation during Real2Sim. While the coding agent constructs and modifies the scene, a human can observe the simulation through the interactive studio built on Viser, directly adjust physical parameters, and provide feedback to guide the agent. Changes made by the agent are reflected in the studio, allowing the human to evaluate simulated behavior against real-world observations. The human can also provide instructions and feedback asynchronously without interrupting the agent's execution. This combines human guidance on task behavior and physical realism with the agent's ability to construct scenes, debug simulation errors, and perform repeated evaluations. For example, when tuning a cable simulation, a human can use sliders in the Viser interface to adjust stiffness, damping, and contact parameters while observing the resulting cable behavior in real time. This provides immediate visual feedback for parameter exploration, while the coding agent can use the same parameters and simulation feedback for automated evaluation and refinement. The shared interface allows humans to provide feedback at a higher frequency and guide the agent toward appropriate physics models and parameters without interrupting its workflow.

4 Case Studies

4.1 Human Real2Sim with Visual Feedback

We first study human-guided Real2Sim using visual feedback. Starting from observations of a real manipulation task, a user constructs the corresponding scene in RoboSimStudio by selecting robot and object assets, arranging their geometry, and choosing the relevant physics models. The interactive studio then supports a tight visual refinement loop: the user executes the task, compares the simulated motion and deformation with the real behavior, and directly adjusts contact and material parameters until the two qualitatively agree. Using this workflow, we created the 18 manipulation simulations shown in Figure 5, spanning rigid-body contact, articulated motion, cables, cloth, and soft-body deformation. For example, three of these tasks highlight different challenges in simulating contact-rich manipulation: We highlight five examples that require different physics models and contact handling:

- *Connector insertion.* Simulating precise assembly requires accurate geometric contact between objects with very small clearances. The task involves inserting a USB-C plug into its receptacle with sub-millimeter clearance. RoboSimStudio supports mesh-based contact and adjustable collision resolution to capture the geometry required for insertion.
- *Leaf pressing.* Simulating thin deformable objects requires capturing their bending behavior and recovery under contact. The task involves pressing a curved leaf and releasing it to recover its shape. RoboSimStudio supports shell-based simulation with adjustable solver and contact parameters to model the leaf's deformation.

- *Strip polishing.* Simulating surface processing requires modeling physical changes caused by contact forces and relative motion. The task involves polishing a plastic strip against a rotating tool to remove surface marks. RoboSimStudio exposes contact forces from the physics solver, allowing material removal to be modeled using Preston's equation [57].

4.2 Human Real2Sim with Force Feedback

While visual feedback allows users to inspect simulated behavior, it can be difficult to assess contact interactions through observation alone. For example, to perform tool insertion, it would be better for the user to feel the contact and friction without visually checking the contact between geometries. To complement visual inspection, RoboSimStudio supports force-feedback teleoperation, allowing operators to physically perceive simulated contact and evaluate the underlying contact models and physical parameters. RoboSimStudio uses a low-cost Gello leader arm [58] to drive a simulated robot in RoboSimStudio, while contact in simulation is rendered back to the operator through the leader's motors. The leader's joint angles command the simulated arm, and differences between the commanded and simulated configurations provide a feedback signal when contact prevents the simulated robot from following the operator. To avoid rendering ordinary simulation lag as contact, RoboSimStudio uses a speed-dependent deadband and releases feedback whenever the simulation pauses or falls behind. The gripper similarly uses its measured, rather than commanded, opening so that grasping an object can be distinguished from closing in free space. The human operator therefore feels the forces produced by the chosen solver and physical parameters, making teleoperation useful both for collecting demonstrations and for directly evaluating the simulated contact model. In the tool-hanging task shown in Figure 6, the operator must pick up a thin tool from the table and insert it into a hanger. Because the critical interaction occurs at contact between the tool and the hanger, force feedback from RoboSimStudio helps the operator perceive and adjust the motion accordingly.

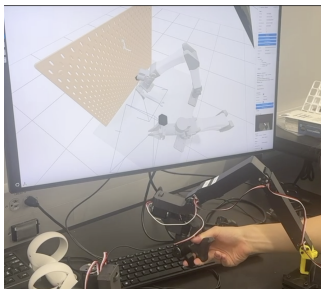
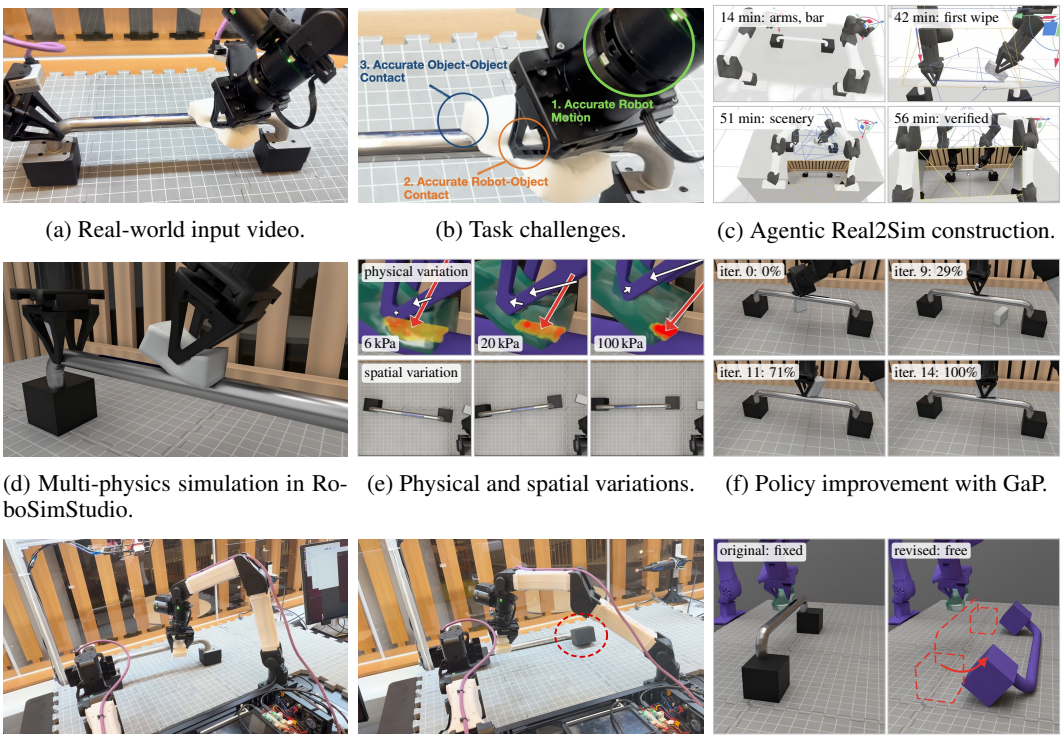


Figure 6: Tool-hanging Teleoperation with RoboSimStudio

4.3 Agentic Real2Sim2Real

We study agentic Real2Sim2Real using a bimanual sponge-wiping task, where one robot arm holds a bar while the other uses a foam sponge to wipe off a marker stripe (Fig. 7a). Given a smartphone video without CAD models or calibrated cameras, two off-the-shelf coding agents, Claude Fable 5.1 in Claude Code [1] and GPT-6 Astra in Codex [2], independently constructed multi-physics simulations in RoboSimStudio in approximately one hour. The agents estimated scene geometry from the video, configured robot and object assets, and iteratively refined the simulation using visual observations and spatial-temporal feedback (Fig. 7c). For example, simulator feedback helped identify errors in robot kinematics and contact conditions, allowing the agent to adjust the wiping motion and establish sponge contact with the bar. One agent constructed and verified the simulated task within 56 minutes.

The resulting simulation couples MuJoCo-Warp within Newton for articulated and rigid bodies with SuperDex for sponge deformation, allowing contact forces and pressure distributions to be evaluated during wiping (Fig. 7d). We then use RoboSimStudio to vary the sponge's stiffness and generate 30 spatial layouts with different bar and sponge configurations (Fig. 7e). Using spatial-temporal simulation feedback, Graph-as-Policy (GaP) [5] improved the simulated wiped area from 0% to 100% over 14 iterations (Fig. 7f). The improved policy was subsequently deployed on the real robot to complete the wiping task (Fig. 7g). However, one real-world rollout revealed a failure in which the wiping motion displaced a bar support that was fixed in the simulation, causing the sponge to lose contact (Fig. 7h). This failure highlights the importance of refining simulation models based on real-world feedback.



(a) Real-world input video. (b) Task challenges. (c) Agentic Real2Sim construction.

(d) Multi-physics simulation in RoboSimStudio. (e) Physical and spatial variations. (f) Policy improvement with GaP.

(g) Sim2Real policy deployment. (h) Real-world failure from unmodeled physics. (i) Agentic simulation refinement.

Figure 7: **Agentic Real2Sim2Real for sponge wiping.** (a) A real-world bimanual manipulation task, where one arm holds a bar while the other wipes a marker stripe using a sponge. (b) Simulation challenges involving robot motion, object-object contact, and robot-object contact. (c) A coding agent constructs and iteratively refines the simulation through the interactive studio, completing a verified simulated wipe in 56 minutes. (d) A multi-physics simulation in RoboSimStudio. (e) Physical variation of sponge stiffness and spatial variation of the bar and sponge across 30 generated layouts (bottom). (f) GaP [5] improves the simulated wiped area from 0% to 100% over 14 iterations using spatial-temporal simulation feedback from rss. (g) The improved policy is deployed on the real robot to perform sponge wiping. (h) A real-world failure caused by displacement of a bar support that was fixed in the original simulation. (i) Given the observed failure, the coding agent revises the physical simulation to allow the bar and its supports to move under contact. Under an off-center wiping motion, the revised simulation captures bar displacement and tipping.

Real-world Feedback and Simulation Refinement. We further study how RoboSimStudio supports iterative simulation refinement using feedback from real-world deployment. Given the failure video and a description of the observed failure, the coding agent identifies that the original simulation incorrectly fixes the bar supports to the environment. The agent modifies the scene by replacing the fixed constraint with a free rigid body, allowing the bar and its supports to move under contact forces (Fig. 7i). The agent then evaluates the revised simulation under different wiping trajectories and contact conditions. While the original fixed-bar simulation cannot reproduce the observed displacement, the revised simulation captures bar motion and tipping when the sponge applies sufficient force or moves away from the bar's center. This suggests how explicit physical constraints and editable scene parameters in RoboSimStudio allow coding agents to identify missing physical interactions and refine the simulation based on real-world observations. The revised simulation can then be used for further policy evaluation and improvement to reduce similar failures.

5 Conclusion

We present RoboSimStudio, an interactive toolkit for simulating robot manipulation that makes the choice of solver, contact model, and physical parameters explicit and editable. RoboSimStudio supports multi-physics simulation by coupling solvers within Newton and external physics engines, and provides interactive interfaces for both humans and coding agents to construct and refine simulations. We demonstrated its capabilities through 18 contact-rich manipulation tasks involving rigid bodies, articulated objects, cables, cloth, and soft-body deformation. In a pilot study of agentic Real2Sim2Real, coding agents constructed a multi-physics simulation of a bimanual sponge-wiping task from a smartphone video in approximately one hour. A Graph-as-Policy improved through simulation feedback was successfully deployed on the real robot, and real-world failure feedback was used to guide further simulation refinement.

Limitations and Future Work. This paper reports a developing system with a pilot study suggesting the feasibility of interactive and agentic Real2Sim2Real. The agentic simulation construction relies on existing material models, physics couplings, and robot assets provided by RoboSimStudio, and its capabilities remain limited by the supported physics solvers. Co-simulation with external physics engines also introduces computational overhead that may limit simulation throughput. Furthermore, we evaluated sim-to-real transfer on a single manipulation task, and assessed the agreement between simulated and real-world behavior primarily through qualitative observations. Future work will include more systematic quantitative evaluations across diverse manipulation tasks, including simulation accuracy, contact forces, object trajectories, agentic construction time, and real-world policy success rates.

References

[1] Anthropic. Claude Code, 2025. URL https://claude.com/claude-code. Agentic coding tool; introduced February 2025. Accessed 2026-10-04.

[2] OpenAI. Codex, 2025. URL https://openai.com/index/introducing-codex/. Coding agent; introduced May 2025. Accessed 2026-10-04.

[3] The Newton Contributors. Newton: GPU-accelerated physics simulation for robotics and simulation research, Apr. 2025. URL https://github.com/newton-physics/newton. Open-source software (Linux Foundation project initiated by Disney Research, Google DeepMind, and NVIDIA).

[4] The Mochi Team. An implicit physics engine for contact-rich simulation, 2026. URL https://github.com/facebookresearch/project_superdex. Project SuperDex.

[5] K. Chen, S. Xie, L. Fu, J. Yu, W. Pacini, et al. GaP: A Graph-as-Policy multi-agent self-learning harness for variational automation tasks. In *Proceedings of The 10th Conference on Robot Learning (CoRL)*, Austin, TX, USA, 2026. URL https://arxiv.org/abs/2607.05369.

[6] Genesis Authors. Genesis: A generative and universal physics engine for robotics and beyond, Dec. 2024. URL https://github.com/Genesis-Embodied-AI/genesis-world.

[7] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

[8] B. Yi, C. M. Kim, J. Kerr, G. Wu, R. Feng, A. Zhang, J. Kulhanek, H. Choi, Y. Ma, M. Tancik, and A. Kanazawa. Viser: Imperative, web-based 3D visualization for Python. In *Proceedings of Robotics: Science and Systems (RSS)*, Sydney, Australia, July 2026. doi:10.15607/RSS.2026.XXII.175.

[9] 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 (IROS)*, pages 5026–5033. IEEE, Oct. 2012. doi:10.1109/IROS.2012.6386109.

[10] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac Gym: High performance GPU based physics simulation for robot learning. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/28dd2c7955ce926456240b2ff0100bde-Abstract-round2.html.

[11] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, June 2023. doi:10.1109/LRA.2023.3270034.

[12] M. Mittal, P. Roth, J. Tigue, A. Richard, O. Zhang, et al. Isaac Lab: A GPU-accelerated simulation framework for multi-modal robot learning. arXiv preprint arXiv:2511.04831, 2025. URL https://arxiv.org/abs/2511.04831.

[13] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, et al. Demonstrating GPU parallelized robot simulation and rendering for generalizable embodied AI with ManiSkill3. In *Proceedings of Robotics: Science and Systems (RSS)*, Los Angeles, CA, USA, June 2025. doi:10.15607/RSS.2025.XXI.021.

[14] K. Zakka, B. Tabanpour, Q. Liao, M. Haiderbhai, S. Holt, J. Y. Luo, A. Allshire, E. Frey, K. Sreenath, L. A. Kahrs, C. Sferrazza, Y. Tassa, and P. Abbeel. Demonstrating MuJoCo Playground. In *Proceedings of Robotics: Science and Systems (RSS)*, Los Angeles, CA, USA, June 2025. doi:10.15607/RSS.2025.XXI.020.

[15] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, K. Lin, A. Maddukuri, S. Nasiriany, and Y. Zhu. robosuite: A modular simulation framework and benchmark for robot learning. arXiv preprint arXiv:2009.12293, 2020. URL https://arxiv.org/abs/2009.12293.

[16] M. Macklin. Warp: A high-performance Python framework for GPU simulation and graphics, Mar. 2022. URL https://github.com/NVIDIA/warp. NVIDIA GPU Technology Conference (GTC).

[17] Google DeepMind and NVIDIA. MuJoCo Warp (MJWarp): GPU-optimized version of the MuJoCo physics simulator, designed for NVIDIA hardware, 2025. URL https://github.com/google-deepmind/mujoco_warp.

[18] A. H. Chen, Z. Liu, Y. Yang, and C. Yuksel. Vertex Block Descent. *ACM Transactions on Graphics*, 43(4):1–16, July 2024. doi:10.1145/3658179.

[19] M. Macklin, M. Müller, and N. Chentanez. XPBD: Position-based simulation of compliant constrained dynamics. In *Proceedings of the 9th International Conference on Motion in Games (MIG)*, MIG '16, pages 49–54. ACM, Oct. 2016. doi:10.1145/2994258.2994272.

[20] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, et al. Do as I can, not as I say: Grounding language in robotic affordances. In K. Liu, D. Kulic, and J. Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning (CoRL)*, volume 205 of *Proceedings of Machine Learning Research*, pages 287–318. PMLR, 2023. URL https://proceedings.mlr.press/v205/ichter23a.html.

[21] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as Policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, May 2023. doi:10.1109/ICRA48891.2023.10160591.

[22] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg. ProgPrompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, May 2023. doi:10.1109/ICRA48891.2023.10161317.

[23] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. VoxPoser: Composable 3D value maps for robotic manipulation with language models. In J. Tan, M. Toussaint, and K. Darvish, editors, *Proceedings of The 7th Conference on Robot Learning (CoRL)*, volume 229 of *Proceedings of Machine Learning Research*, pages 540–562. PMLR, 2023. URL https://proceedings.mlr.press/v229/huang23b.html.

[24] S. H. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor. ChatGPT for robotics: Design principles and model abilities. *IEEE Access*, 12:55682–55696, 2024. doi:10.1109/ACCESS.2024.3387941.

[25] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, et al. Inner Monologue: Embodied reasoning through planning with language models. In K. Liu, D. Kulic, and J. Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning (CoRL)*, volume 205 of *Proceedings of Machine Learning Research*, pages 1769–1782. PMLR, 2023. URL https://proceedings.mlr.press/v205/huang23c.html.

[26] W. Yu, N. Gileadi, C. Fu, S. Kirmani, K.-H. Lee, et al. Language to rewards for robotic skill synthesis. In J. Tan, M. Toussaint, and K. Darvish, editors, *Proceedings of The 7th Conference on Robot Learning (CoRL)*, volume 229 of *Proceedings of Machine Learning Research*, pages 374–404. PMLR, 2023. URL https://proceedings.mlr.press/v229/yu23a.html.

[27] Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar. Eureka: Human-level reward design via coding large language models. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=IEduRUO55F.

[28] K. E. Chen, S. Xie, Z. Ma, P. Sanketi, and K. Goldberg. Robo2VLM: Improving visual question answering using large-scale robot manipulation data. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pages 24644–24679. Curran Associates, Inc., 2025. doi:10.52202/085713-0733. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/1f467c3e37abf9f86c78f44c6a27ee7c-Abstract-Datasets_and_Benchmarks_Track.html.

[29] S. Xie, K. Chen, Z. Chen, S. Adebola, Y. Huang, Z. Ma, T. Qiu, W. Yuan, D. Shah, P. R. Sanketi, and K. Goldberg. RoboVista: Evaluating vision language models for diverse robot applications. In *Proceedings of Robotics: Science and Systems (RSS)*, Sydney, Australia, July 2026. doi:10.15607/RSS.2026.XXII.095.

[30] Y. J. Ma, W. Liang, H.-J. Wang, Y. Zhu, L. Fan, O. Bastani, and D. Jayaraman. DrEureka: Language model guided sim-to-real transfer. In *Proceedings of Robotics: Science and Systems (RSS)*, Delft, Netherlands, July 2024. doi:10.15607/RSS.2024.XX.094.

[31] L. Wang, Y. Ling, Z. Yuan, M. Shridhar, C. Bao, Y. Qin, B. Wang, H. Xu, and X. Wang. GenSim: Generating robotic simulation tasks via large language models. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=OI3RoHoWAN.

[32] P. Hua, M. Liu, A. Macaluso, Y. Lin, W. Zhang, H. Xu, and L. Wang. GenSim2: Scaling robot data generation with multi-modal and reasoning LLMs. In P. Agrawal, O. Kroemer, and W. Burgard, editors, *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pages 5030–5066. PMLR, 2025. URL https://proceedings.mlr.press/v270/hua25a.html.

[33] W. Ye, F. Liu, Z. Ding, Y. Gao, O. Rybkin, and P. Abbeel. Video2Policy: Scaling up manipulation tasks in simulation through internet videos. arXiv preprint arXiv:2502.09886, 2025. URL https://arxiv.org/abs/2502.09886.

[34] Z. Mandi, Y. Weng, D. Bauer, and S. Song. Real2Code: Reconstruct articulated objects via code generation. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=CAssIgPN4I.

[35] M. Ahn, D. Dwibedi, C. Finn, M. Gonzalez Arenas, K. Gopalakrishnan, et al. AutoRT: Embodied foundation models for large scale orchestration of robotic agents. arXiv preprint arXiv:2401.12963, 2024. URL https://arxiv.org/abs/2401.12963.

[36] L. Fu, J. Yu, K. El-Refai, E. Kou, H. Xue, H. Huang, W. Xiao, L. Fei-Fei, G. Shi, J. Wu, S. S. Sastry, Y. Zhu, K. Goldberg, and L. Fan. CaP-X: A framework for benchmarking and improving coding agents for robot manipulation. In T. Zhang, M. Dudik, M. Jaggi, A. Agarwal, S. Li, D. Schuurmans, J. Zhu, F. Berkenkamp, H. Dong, and A. Bietti, editors, *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, volume 306 of *Proceedings of Machine Learning Research*, pages 32155–32212. PMLR, 2026. URL https://proceedings.mlr.press/v306/fu26u.html.

[37] J. Shi, R. Yang, K. Chao, S. B. Wan, Y. Shao, J. Lei, J. Qian, L. Le, P. Chaudhari, K. Daniilidis, C. Wen, and D. Jayaraman. Maestro: Orchestrating robotics modules with vision-language models for zero-shot generalist robots. arXiv preprint arXiv:2511.00917, 2025.

[38] R. Lu, Y. Wu, E. Kou, L. Fu, W. Xiao, A. Mandlekar, Y. Xu, G. Shi, K. Goldberg, A. Chen, M. Chowdhury, Y. Zhu, L. Fan, and G. Wang. ASPIRE: Agentic /skills discovery for robotics. arXiv preprint arXiv:2607.00272, 2026.

[39] W. Xiao, J. Xie, T. Zhang, H. Lin, L. Fu, H. Xue, J. Lu, Y. Yang, C. Dai, Z. Wang, J. Wu, G. Wang, S. S. Sastry, K. Goldberg, L. Fan, Y. Zhu, and G. Shi. ENPIRE: Agentic robot policy self-improvement in the real world. In *Proceedings of The 10th Conference on Robot Learning (CoRL)*, 2026.

[40] Y. Wang, Y. Li, L. Fei-Fei, J. Wu, and H. Huang. DexAgent: An agentic Human2Sim2Robot framework for dexterous manipulation with self-evolving tool library. arXiv preprint arXiv:2609.35318, 2026.

[41] Y.-J. Wang, H. Jiang, S. Deng, H. Xue, W. Ye, R. Duan, N. Haghtalab, S. S. Sastry, P. Abbeel, and H. Qi. Reconstruct, practice, go real: Guided self-improvement for embodied agents. arXiv preprint arXiv:2610.02204, 2026.

[42] S. Xie, K. Chen, and K. Goldberg. Revisiting the "Push-T" robot manipulation task with agentic robotics. arXiv preprint arXiv:2608.18227, 2026. URL https://arxiv.org/abs/2608.18227.

[43] J. Yang, C. E. Jimenez, A. Wettig, K. Lieret, S. Yao, K. Narasimhan, and O. Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pages 50528–50652. Curran Associates, Inc., 2024. doi:10.52202/079017-1601. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5a7c947568c1b1328ccc5230172e1e7c-Abstract-Conference.html.

[44] M. Torne Villasevil, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. In *Proceedings of Robotics: Science and Systems (RSS)*, Delft, Netherlands, July 2024. doi:10.15607/RSS.2024.XX.015.

[45] Q. Chen, A. Walsman, M. Memmel, K. Mo, A. Fang, D. Fox, and A. Gupta. URDFormer: A pipeline for constructing articulated simulation environments from real-world images. In *Proceedings of Robotics: Science and Systems (RSS)*, Delft, Netherlands, July 2024. doi:10.15607/RSS.2024.XX.124.

[46] T. Dai, J. Wong, Y. Jiang, C. Wang, C. Gokmen, R. Zhang, J. Wu, and L. Fei-Fei. Automated creation of digital cousins for robust policy learning. In P. Agrawal, O. Kroemer, and W. Burgard, editors, *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pages 4912–4943. PMLR, 2025. URL https://proceedings.mlr.press/v270/dai25a.html.

[47] L. Le, J. Xie, W. Liang, H.-J. Wang, Y. Yang, Y. J. Ma, K. Vedder, A. Krishna, D. Jayaraman, and E. Eaton. Articulate-Anything: Automatic modeling of articulated objects via a vision-language foundation model. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=s3FTX4Ay55.

[48] X. Li, K. Hsu, J. Gu, O. Mees, K. Pertsch, et al. Evaluating real-world robot manipulation policies in simulation. In P. Agrawal, O. Kroemer, and W. Burgard, editors, *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pages 3705–3728. PMLR, 2025. URL https://proceedings.mlr.press/v270/li25c.html.

[49] M. N. Qureshi, S. Garg, F. Yandun, D. Held, G. Kantor, and A. Silwal. SplatSim: Zero-shot Sim2Real transfer of RGB manipulation policies using Gaussian splatting. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6502–6509. IEEE, May 2025. doi:10.1109/ICRA55743.2025.11128339.

[50] Y. Wang, Z. Xian, F. Chen, T.-H. Wang, Y. Wang, K. Fragkiadaki, Z. Erickson, D. Held, and C. Gan. RoboGen: Towards unleashing infinite data for automated robot learning via generative simulation. In R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 51936–51983. PMLR, 2024. URL https://proceedings.mlr.press/v235/wang24cc.html.

[51] Y. Yang, F.-Y. Sun, L. Weihs, E. Vanderbilt, A. Herrasti, W. Han, J. Wu, N. Haber, R. Krishna, L. Liu, C. Callison-Burch, M. Yatskar, A. Kembhavi, and C. Clark. Holodeck: Language guided generation of 3D embodied AI environments. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16277–16287. IEEE, June 2024. doi:10.1109/CVPR52733.2024.01536.

[52] A. Goldberg, K. Kondap, T. Qiu, Z. Ma, L. Fu, J. Kerr, H. Huang, K. Chen, K. Fang, and K. Goldberg. Blox-Net: Generative design-for-robot-assembly using VLM supervision, physics simulation, and a robot with reset. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15493–15500. IEEE, May 2025. doi:10.1109/ICRA55743.2025.11127489.

[53] Google. Gemini 4 Argon, 2026. Released to developers October 13, 2026.

[54] Blender Online Community. Blender: A 3D modelling and rendering package, 2026. URL https://www.blender.org. Blender Foundation, Amsterdam.

[55] C. M. Kim, B. Yi, H. Choi, Y. Ma, K. Goldberg, and A. Kanazawa. PyRoki: A modular toolkit for robot kinematic optimization. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1312–1319. IEEE, Oct. 2025. doi:10.1109/IROS60139.2025.11246651.

[56] B. Sundaralingam, S. K. S. Hari, A. Fishman, C. Garrett, K. V. Wyk, V. Blukis, A. Millane, H. Oleynikova, A. Handa, F. Ramos, N. Ratliff, and D. Fox. cuRobo: Parallelized collision-free minimum-jerk robot motion generation. arXiv preprint arXiv:2310.17274, 2023. URL https://arxiv.org/abs/2310.17274. Technical report.

[57] F. W. Preston. The theory and design of plate glass polishing machines. *Journal of the Society of Glass Technology*, 11:214–256, 1927.

[58] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel. GELLO: A general, low-cost, and intuitive teleoperation framework for robot manipulators. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12156–12163. IEEE, Oct. 2024. doi:10.1109/IROS58592.2024.10801581.