Preprints.org

Article

EnerVerse-AC: Envisioning Embodied Environments with Action Condition

Yuxin Jiang , Shengcong Chen , Siyuan Huang , Liliang Chen , Pengfei Zhou , Yue Liao , Xindong He , Chiming Liu , Hongsheng Li , Maoqing Yao , Guanghui Ren [*]

Article

ENERVERSE-AC: Envisioning Embodied Environments with Action Condition

Yuxin Jiang [1,†]**, Shengcong Chen** [1,†]**, Siyuan Huang** [2,†]**, Liliang Chen** [1,‡]**, Pengfei Zhou** [1]**, Yue Liao** [3]**, Xindong He** [1]**, Chiming Liu** [1]**, Hongsheng Li** [3]**, Maoqing Yao** [1,*] **and Guanghui Ren** [1,*]

[1]   AgiBot
[2]   SJTU
[3]   MMLab-CUHK
[†]   Equal contribution.
[‡]   Project leader.

Abstract: Robotic imitation learning has advanced from solving static tasks to addressing dynamic interaction scenarios, but testing and evaluation remain costly and challenging due to the need for real-time interaction with dynamic environments. We propose ENERVERSE-AC (abbr. EVAC), an action-conditional world model that generates future visual observations based on an agent's predicted actions, enabling realistic and controllable robotic inference. Building on prior architectures, EVAC introduces a multi-level action-conditioning mechanism and ray map encoding for dynamic multi-view image generation while expanding training data with diverse failure trajectories to improve generalization. As both a data engine and evaluator, EVAC augments human-collected trajectories into diverse datasets and generates realistic, action-conditioned video observations for policy testing, eliminating the need for physical robots or complex simulations. This approach significantly reduces costs while maintaining high fidelity in robotic manipulation evaluation. Extensive experiments validate the effectiveness of our method. Code, checkpoints, and datasets can be found at https://annaj2178.github.io/EnerverseAC.github.io.

Keywords: world model; video generation; data engine

1. Introduction

The development of robotic imitation learning has significantly advanced robotic manipulation, transitioning the field from solving isolated tasks in static environments to addressing complex and diverse interaction scenarios. Unlike traditional AI domains such as computer vision (CV) or natural language processing (NLP), where model performance can be evaluated using non-interactive and static datasets, robotic manipulation inherently requires real-time interaction between agents and dynamic environments during testing and evaluation. As task diversity grows, assessing policy performance often necessitates direct deployment on physical robots or the creation of large-scale 3D simulation environments—both of which are costly, labor-intensive, and challenging to scale.

Building low-cost, scalable testing and inference environments for robotic manipulation has thus become a critical challenge in robotic imitation learning. Recently, the concept of using video generation models as world simulators has emerged as a promising direction. These models enable agents to observe and interact with dynamic worlds through learned visual dynamics, circumventing the need for explicit physical simulation. While this approach introduces a new avenue for constructing robotic inference pipelines, existing world modeling techniques primarily focus on generating videos from language instructions and predicting actions based on the generated videos. However, these methods fall short of creating true world simulators, which should simulate environment dynamics in response to the agent's actions, enabling realistic and controllable testing.
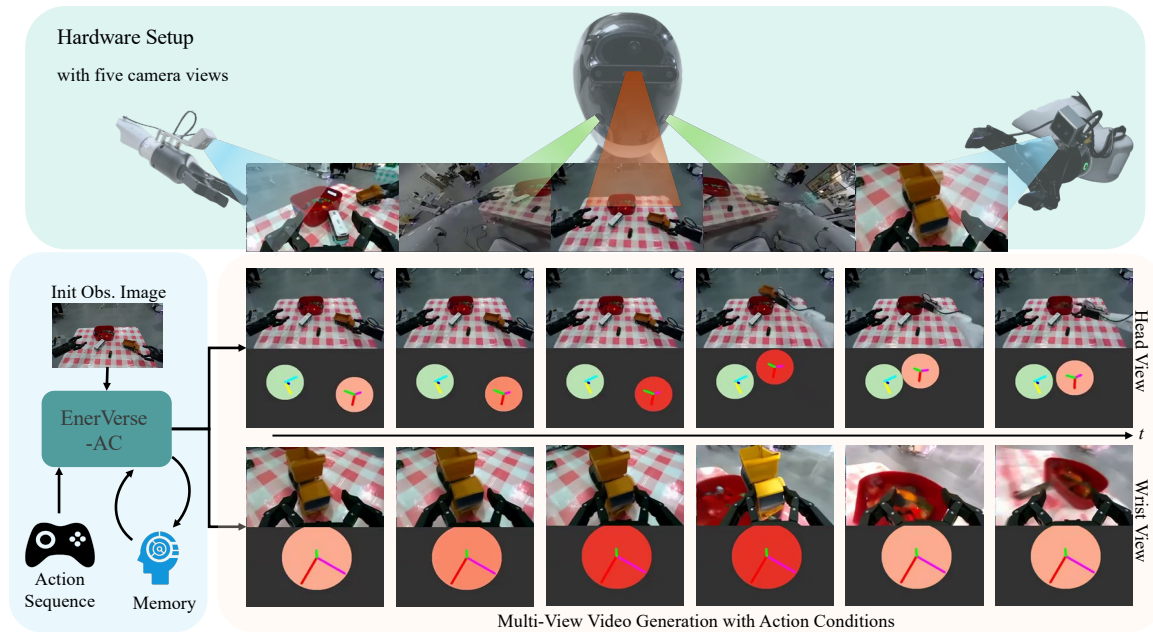
Figure 1. Overview of the EVAC framework. Given initial observation images and an action sequence, EVAC generates multi-view videos conditioned on the provided actions. By incorporating a memory mechanism, EVAC supports the generation of long-term video sequences. The framework handles both static head camera views and dynamic wrist camera views to provide a comprehensive representation of the robotic environment.

To bridge this gap, we propose EVAC, an action-conditional world model that generates future visual observations directly conditioned on the agent's predicted actions. Built upon prior embodied world model architectures like **EnerVerse** [1], ENERVERSE-AC incorporates additional **A**ction-**C**onditioning information to enable more realistic and controllable robotic inference. To achieve this, we designed a multi-level action condition injection mechanism, which uses end-effector projection action maps and delta action encodings. Furthermore, to support the generation of multi-view images, crucial for embodied tasks, we introduce spatial cross-attention modules and ray direction map encoding to process multi-view features. To reflect the movement of camera, we encode the camera's motion using ray map embeddings.

Beyond architectural innovations, the EVAC world model is designed to handle both successful and failure scenarios. In addition to leveraging the Agibot-World dataset [2], we curated a diverse dataset of failure trajectories, significantly expanding the training data's coverage. This enhancement improves the model's ability to generalize across diverse scenarios, ensuring its applicability to real-world robotics tasks.

The proposed EVAC world model serves as both a data engine for policy learning and an evaluator for trained policy models, addressing key challenges in robotic manipulation. As a data engine, EVAC augments limited human-collected trajectories into diverse datasets by segmenting actions (e.g., fetch, grasp, home), applying spatial augmentations, and generating new video sequences, thereby enhancing policy robustness and generalization. As an evaluator, it eliminates the need for complex simulation assets by generating realistic, action-conditioned video observations for iterative policy testing, which can be reviewed by human evaluators or automated systems like Video-MLLMs. This approach significantly reduces reliance on real robot hardware during development, saving costs and time, while maintaining high evaluation fidelity correlated with real-world performance.

2. Related Work

Video Generation Model as World Model. While prior research on generative models has shown promise, [3,4] highlights video generation as an innovative approach to constructing world models. Similarly, [5] aims to develop a universal world model built upon the generative model but focuses

on generating only the next-step frames rather than continuous video sequences. Video generation remains a challenging task with applications across diverse domains. Recent advancements in diffusion models [6] and latent diffusion models [7] have demonstrated progress in generating high-quality images with reduced computational complexity. Furthermore, text-guided and pose-guided video generation methods [8,9] have expanded the applicability of video synthesis technologies.

In robotics, works like [1,10] focus on generating future frames from textual and visual inputs. However, limited attention has been given to video generation conditioned on robotic actions. Gesture-conditioned approaches [11] provide valuable insights but have yet to be tested in robotics, where environments and object interactions are significantly more complex. Advancements in action-conditioned video generation are essential to address these challenges.

Physical Simulators for Robotics. Physical simulators are widely applied in robotics learning tasks. MuJoCo [12] has been used for locomotion and manipulation studies, while PyBullet [13] supports real-time control and sim-to-real experiments. Similarly, Isaac Gym [14] facilitates reinforcement learning in continuous control tasks with large-scale parallel environments. Several studies [15] utilize physical simulators to train policies for solving dexterous manipulation tasks. Despite their utility, physical simulators face notable limitations. The sim-to-real gap often results in overfitting to synthetic environments, reducing real-world performance. Moreover, creating digital assets—including robot embodiments, target objects, and task scenes—remains labor-intensive and requires expert-level effort, further hindering scalability.

Robotics Imitation Learning. Recent advancements in robotics imitation learning focus on developing generalist models capable of efficiently handling diverse tasks across multiple embodiments using extensive multimodal datasets. Models such as RT-1 [16], Gato [17], Octo [18], and OpenVLA [19] integrate pretrained visual and language models with specialized policy heads, enabling remarkable task generalization. Building on this, [20] introduces a dual-brain system, while [21] employs layer-wise information with flow matching techniques for action prediction. Additionally, [2] transitions from direct action prediction to latent action representations, ensuring more effective generalization. However, these approaches rely heavily on large-scale action datasets for training. While some works, such as [22], attempt to reduce data requirements by increasing information density, they still depend significantly on human data collection, underscoring the need for further innovations in data-efficient learning techniques.

3. Method

In EVAC, we adopt a UNet-based video generation model as our baseline, following [1,23]. Beyond this, we propose an action-conditioned framework, as illustrated in Figure 2. Given a RGB video set $O \in \mathbb{R}^{V \times (H+K) \times 3 \times h \times w}$, where V denotes the number of views, H represents the number of observed history frames, K is the number of intended predicted frames, and h, w are the frame height and width, our method is designed to predict future frames based on observed past frames and robotic actions. First, we pass the video set through an encoder ε to obtain the latent representation $z \in \mathbb{R}^{V \times H \times C \times h \times w}$, where C is the latent dimensionality. Using a latent diffusion model, we aim to predict $z_t = p_\theta(z_{t-1}, c, t)$, where c is the condition signal and t is the denoising timestep. In this work, the condition signal originates from the robotic action trajectory $A \in \mathbb{R}^{(H+K) \times d}$, where $d = 7$ represents the end-effector pose with $[x, y, z, roll, pitch, yaw, openness]$ and $d = 14$ in bi-arm case.

To inject the action condition, we use both spatial-aware pose information injection and delta action attention module. Furthermore, we extend traditional 2D video generation to 3D video generation, represented by multi-view frames, to better meet the requirements of robotic manipulation tasks.
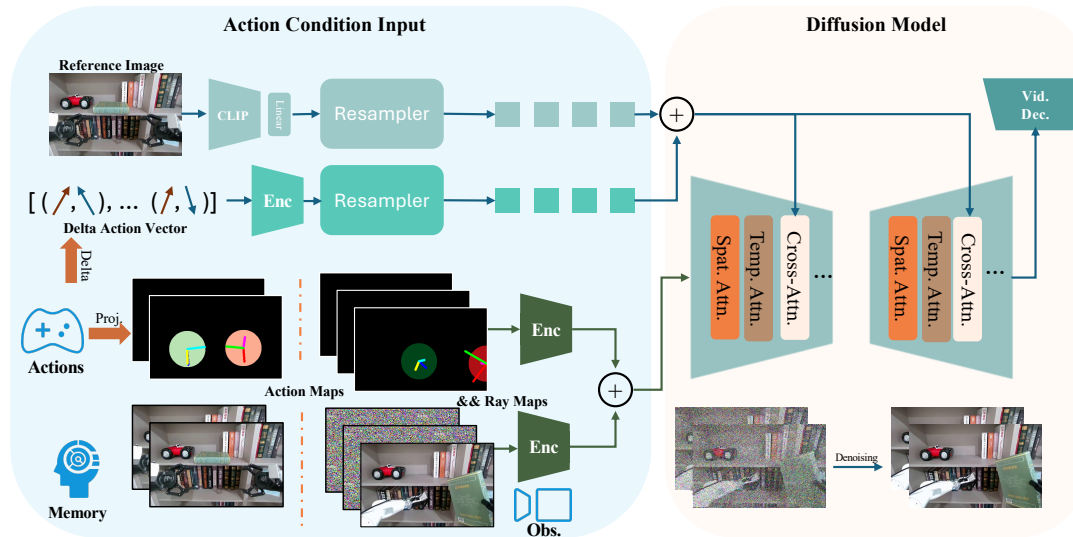
Figure 2. Overview of the EVAC Framework. The framework begins with a reference image, whose feature vector serves as the reference style guidance. The original robotic actions are processed to compute the delta action vector and this temporal information is concatenated with the reference style guidance and injected into the diffusion model via a cross-attention mechanism. Additionally, the action information is projected into action maps, whose feature maps are concatenated with feature maps from both memory and visual observations before being fed into the diffusion network. The diffusion model generates video frames with denoising process, followed by a video decoder to produce the final output. For simplicity, we only demonstrate the single-view case here.

3.1. Mutli-Level Action Condition Injection

Spatial-Aware Pose Injection. [9,24–26] have proposed different ways on controlling video generation by injecting pose information. One common way to align the image with the fine-grained pose trajectory is to use a pixel-alignment method to inject the pose signal. In the field of robotics, end-effector 6D position has been tested as an effective representation of action space. Therefore, to ensure precise visual alignment with the conditioned image, we have developed methodologies to effectively depict the 6D end effector pose of the end effector. Firstly, we convert the end-effector position at timestamp i in world coordinates to the corresponding pixel coordinates using the calibrated camera parameters. Furthermore, to visually represent the roll, pitch, and yaw angles in 2D image space, we employ visual prompting techniques inspired by [27,28]. This approach utilizes unit vectors along each directional axis, providing an intuitive representation of the end-effector's orientation in 3D space.

To illustrate the gripper action at each state, we use a unit circle to encode the action magnitude, where lighter shades correspond to open gripper and darker shades indicate closed gripper. To differentiate between the left and right hand, we employ distinct color schemes for visualizing 6D poses and gripper actions. The 6D pose visualization is rendered on a black background to enhance clarity, as shown in Figure 3. After constructing the action map using the aforementioned visual prompting techniques, we process it with the CLIP [29] vision encoder. The resulting feature maps are concatenated with the feature maps from RGB images along the channel dimension.

Delta Action Attention Module. Furthermore, we designed a Delta Action Attention module which calculates the delta motion between consecutive frames to approximate changes in the end-effector's position and orientation. These delta motions are encoded into a fixed number of latent representations by a linear projector and then via cross-attention [30,31]. The fixed-length latent representation token is then fused with the Reference Image map and injected into the Unet stage through a cross-attention mechanism. By incorporating temporal changes, such as speed and acceleration, the module enhances the model's physical understanding of motion dynamics, enabling it to produce more realistic and diverse video outputs.
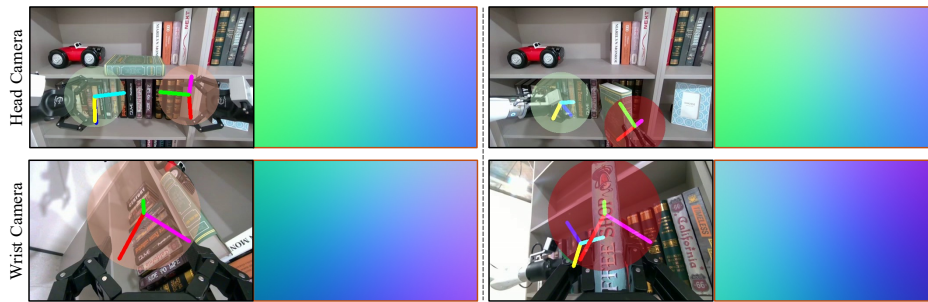
Figure 3. Visualizing EEF Projections and the Ray Maps. The bottom row illustrates wrist camera views, where projections appear nearly identical. Then, ray maps provide additional spatial context to represent movements. The value of the ray maps is visualized with the RGB value.

3.2. Multi-View Condition Injection

In embodied robotics, cross-view information, particularly visual inputs from wrist cameras, is essential for accurate trajectory prediction. To address this, we extend EVAC world model to support multi-view video generation. Following EnerVerse [1], multi-view features are fed where spatial cross-attention modules enable interaction between views. A ray direction map encoding camera parameters is also concatenated into the input features to provide spatial context. Unlike EnerVerse, which only processed static views, EVAC incorporates dynamic wrist camera views that move with the robotic arms. This creates a challenge: when projecting end-effector (EEF) poses onto wrist camera images using Section 3.1 methods, the projection circle remains static, failing to convey the hand's movement, as shown in Figure 3.

Inspired by techniques in [1,32], we encode camera motion using the origins o_r and directions d_r of ray maps $r = (o_r, d_r)$. Specifically, for each camera, we compute the ray maps relative to its poses at all times. Since the wrist cameras move with the arms, the ray maps of the wrist cameras can implicitly encode the motion information of EEF poses. Therefore, the ray maps are concatenated with the trajectory maps to provide enriched trajectory information, improving cross-view consistency.

3.3. Applications

Data Engine for Policy Learning. The EVAC world model can serve as a data engine for robotic policy learning. Specifically, for a new manipulation task (for simplicity, we use the primitive single-object pick-and-place task as an example), a human data collector first captures M trajectories. For each collected trajectory, the beginning (t_b) and ending (t_e) timestamps of the gripper-object contact phase are identified by analyzing changes in gripper openness. These timestamps are then used to segment the trajectory into three distinct phases: fetching, grasping, and homing.

Focusing on the fetching phase as an example, we extract the visual observation O_{t_b} and the corresponding action sequence $[a_{t_b-N}, \ldots, a_{t_b}]$. While the action a_{t_b} is kept fixed, the earlier action a_{t_b-N} is spatially augmented to generate a new action a'_{t_b-N}. After augmentation, interpolation is applied to create new action trajectories for the sequence. Then, O_{t_b} and the reversed action sequence $[a_{t_b}, \ldots, a'_{t_b-N}]$ are fed into the EVAC world model to generate the corresponding video frames. Once the frames are generated, they are re-ordered to create a correctly sequenced dataset. By following this process, the original M trajectories can be augmented into a significantly more diverse set of trajectories, enhancing the robustness and generalization of the policy learning process.

Evaluator for Policy Model. Another application of EVAC is serving as a physical simulator to evaluate trained policy models. Given an initial visual observation O_t and corresponding instructions, the policy model generates action chunks. These action chunks, along with O_t, are then fed into the EVAC world model to generate new observations. This process is iteratively repeated until the actions generated by the policy model fall below a predefined threshold. Subsequently, multiple human evaluators watch the EVAC-generated videos to assess task success.

This evaluation approach offers two key advantages. First, it eliminates the need to create complex simulation assets, as EVAC can better represent certain physical aspects, such as fluid

dynamics, compared to conventional simulators. Second, the video replay can be sped up to save time, or it can potentially be integrated with video-based Video-MLLMs, reducing the need for human evaluation efforts. By leveraging this process, the EVAC world model can largely replace the use of real robot hardware during the initial development stage, significantly reducing deployment efforts. Our experiments reveal a high correlation between evaluation results obtained through EVAC and those observed in real-world scenarios.
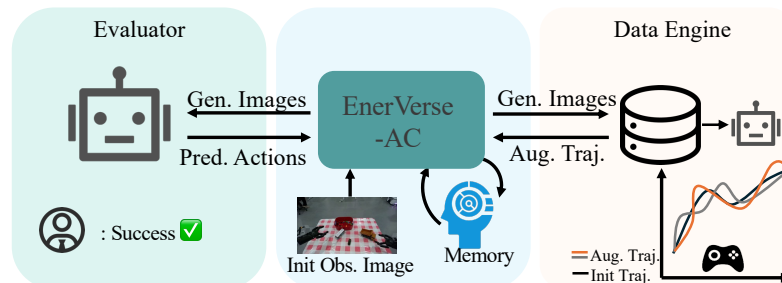

Figure 4. EVAC's ==as== Data Engine and Policy Evaluator.

4. Experiments

4.1. Experiment Details

Dataset The training data for EVAC is primarily sourced from the AgiBot World dataset [2], which contains over 210 tasks and 1 million trajectories. To ensure comprehensive coverage of action trajectories, including both successful and failed cases—critical for enabling EVAC to function as a generalized simulator—we collaborated with the AgiBot-Data team to gain full access to the raw data. From this dataset, we mined a substantial amount of failure cases. Additionally, we developed an automated data collection pipeline to capture real-world failure cases during teleoperation and real-robot inference, further enriching the dataset with diverse scenarios.

Implementation Details Our model is built on UNet-based Video Diffusion Models (VDM) [23]. During training, the CLIP visual encoder and VAE encoder are frozen, while other components, including the UNet, resampler, and linear layers, are fine-tuned. The model is trained with a batch size of 16. For the single-view version, training requires approximately 32 A100 GPUs for 2 days, whereas the multi-view version takes about 32 A100 GPUs for 8 days. We experimentally determined that setting the memory size to 4 and the chunk size to 16 achieves a balance between generation quality and resource cost. The memory consists of 4 historical frames, each derived from the results of the previous chunk generation. For the robotic policy model, we utilize the official single-view version of GO-1 [2].

4.2. Controllable Manipulation Video Generation

As shown in Figure 5, EVAC excels at synthesizing realistic videos of complex robot-object interactions, even in challenging scenarios. A key strength of EVAC lies in its ability to maintain high visual fidelity while accurately following input action trajectories, ensuring reliability for building credible evaluation systems.

The model's chunk-wise autoregressive diffusion architecture and sparse memory mechanism, inspired by [1], enable it to sustain visual stability and scene consistency during continuous chunk-wise inference. Experimental results show that the generated videos remain sharp and reliable for up to 30 consecutive chunks in single-view scenarios and 10 chunks in multi-view settings. However, artifacts and blurring begin to emerge in longer sequences, highlighting a tradeoff between sequence length and visual quality. Figure 6 further illustrates EVAC's ability to preserve scene integrity across multiple chunks during a manipulation task. The snapshots showcase environment consistency over time, demonstrating EVAC's robust performance in maintaining visual coherence during chunk-wise autoregressive inference.
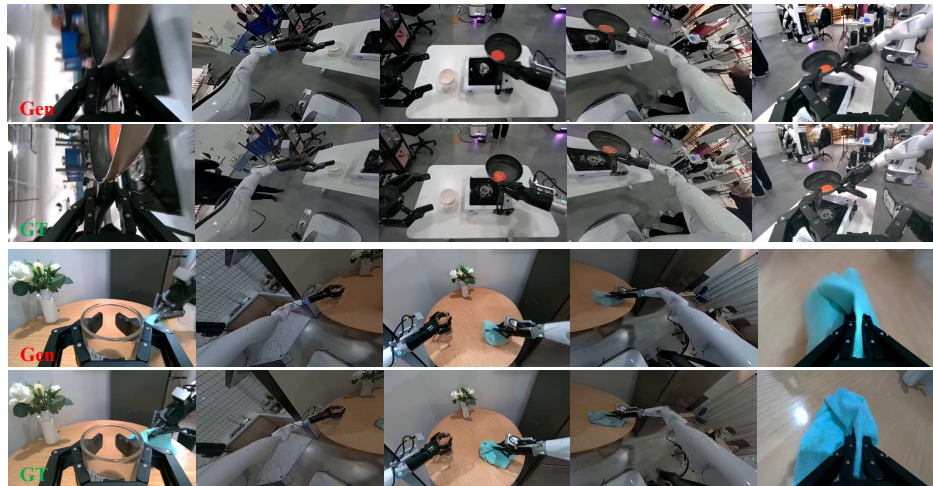
Figure 5. Qualitative results for multi-view video generation.

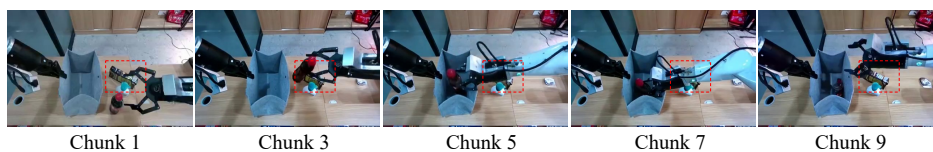| Chunk 1 | Chunk 3 | Chunk 5 | Chunk 7 | Chunk 9 |

Figure 6. Environment consistency during chunk-wise inference. Snapshots from EVAC at various inference stages (Chunks 1, 3, 5, 7, and 9) demonstrate robust performance in maintaining visual fidelity and scene coherence over time.

4.3. EVAC *as Policy Evaluator*

This section evaluates the consistency between the EVAC-based generative simulator and real-world environments. Four manipulation tasks were selected for evaluation; details for these tasks can be found in the Appendix A. For each task, real-world evaluations were conducted first, and the initial frame recordings from these tests were used as the image condition for EVAC evaluations. Success or failure was determined by three independent evaluators who observed either real-world executions or EVAC-generated sequences. As shown in Figure 7 (left), while there were minor differences in absolute success rates between EVAC and real-world evaluations, the relative performance trends across tasks were consistent. These findings demonstrate EVAC's reliability for cross-task policy performance analysis and its ability to closely replicate real-world dynamics. We also provide the qualitative evaluation results in LIBERO [33] simulator in the Appendix.

Another challenge in robot policy learning is the instability during training, e.g. performance fluctuates across training steps. To assess EVAC's ability to reflect these fluctuations, we evaluated the same policy at different training steps using the "Take a Bottle" task as an example. As shown in Figure 7 (right), both EVAC and real-world evaluations captured the same performance trend, with success rates improving as the number of training steps increased. This result confirms that EVAC accurately mirrors real-world performance variations during policy training.
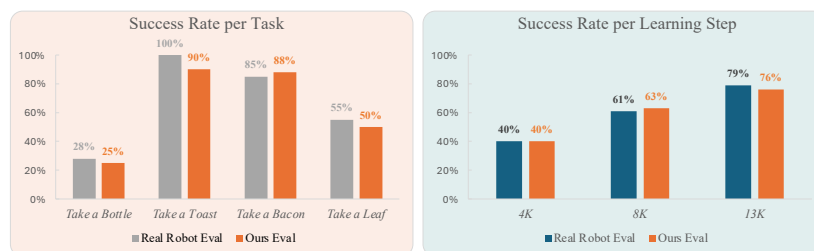

Figure 7. Comparison of Success Rates Across Tasks and Training Steps. (Left) Despite tasks vary a lot, the EVAC simulator consistently aligned its evaluation results with real-world ones. (Right) Success rates of a single policy model evaluated at three training steps. Both EVAC and real-world testing demonstrated a similar performance gradient.

4.4. EVAC *as Data Engine*

In this part, we aim to demonstrate the potential of EVAC to generate novel action trajectories that augment policy training data, leading to improved task performance. The evaluation task involves picking a bottle of water from a paper box and placing it on a table. This task is challenging due to the precise force and manipulation required to extract a tightly packed bottle.

We compare two training setups: (1)Baseline: The policy is trained with only 20 expert demonstration episodes. (2)Augmented Dataset: The policy is trained with the same 20 expert episodes, augmented with 30% additional trajectories generated using the EVAC world model. As shown in Table 1, the success rate (SR) improves significantly from 0.28 to 0.36 when the augmented trajectories are included in the training data. This result highlights the capability of the EVAC world model to enhance policy learning by providing diverse and effective training samples, even when the number of expert demonstrations is limited.

Table 1. Impact of Data Augmentation on Policy Training Success Rate.

Training Data	Success Rate (SR)
Baseline	0.28
Augmented Dataset(Adding 30% synthetic data)	0.36

4.5. Further Analysis

Failure Data Matters. As discussed in Section 4.1, we deliberately collected failure trajectories to expand the action coverage in the training data. To evaluate the effectiveness of this failure data, we trained two models: one with failure trajectories included and the other without. As illustrated in Figure 8, we tested the models using a scenario where the robotic arm was pretending to grasp a bottle of water that was not actually present.

Without failure data, the model tended to overfit to successful examples, leading it to "hallucinate" that the bottle had been successfully grasped despite the absence of physical interaction. In contrast, with the inclusion of failure data, EVAC was able to accurately recognize and distinguish the failed grasp attempt, demonstrating its robustness against overfitting and its ability to handle edge cases effectively.
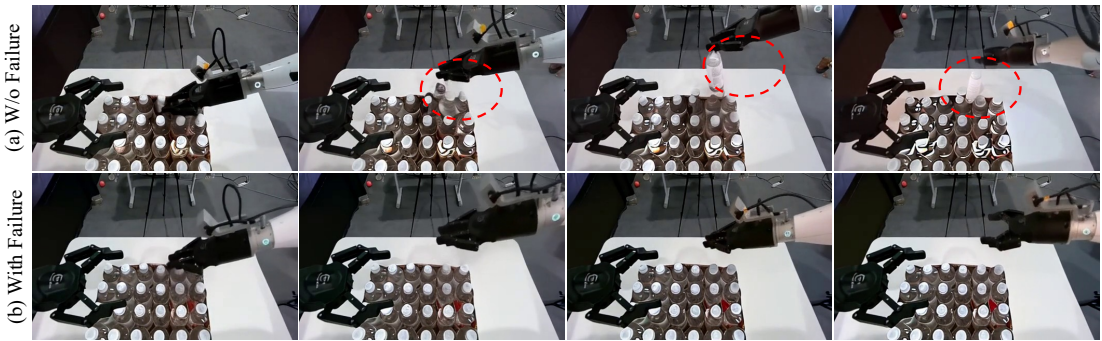

Figure 8. Impact of Failure Data and Data Augmentation on Trajectory Generation. Without failure data: The model overfits to success-only trajectories, incorrectly "hallucinating" that the bottle has been grasped by the robotic arm.

5. Conclusion

In this paper, we introduced EVAC, an embodied world model with action-conditioned capabilities. We proposed multi-level action condition injection strategies and utilized camera ray maps to model dynamic camera's motion. Through extensive experiments, we demonstrated the dual functionality of EVAC: serving as both a data engine for policy learning and an evaluator for trained policy models.

6. Limitations and Discussions

We proposed ENERVERSE-AC, an action-conditional embodied world model generator by implementing a video generation framework. However, several limitations remain in the current work.

First, our approach for interpreting gripper openness using a unit circle representation with varying color intensities may not generalize effectively to more complex end-effectors, such as dexterous hands. Adapting the framework to different robotic hardware configurations will require additional preprocessing steps and refinements.

Second, the wrist camera often captures irrelevant background noise, such as people moving around the robot workspace, which increases the complexity of video generation. As shown in our experiments, this limitation restricts multi-view inference to 10 chunks, compared to the 30 chunks achievable in single-view generation, thereby reducing the overall efficiency of multi-view scenarios.

Additionally, several potential applications of our action-conditioned world model remain unexplored, such as its integration with actor-critic methods for reinforcement learning. Future research could extend our framework by exploring these applications and drawing inspiration from prior work [34–36]. We hope that this work serves as a foundation for advancing embodied world models and inspires further developments in the field.

Appendix A

In this appendix, we first provide additional training details and model parameters in Section A.1. Next, we present an ablation study to evaluate the effectiveness of the proposed delta action attention module in Section A.2. We also include more quantitative results from both real-world and simulator experiments. Finally, we demonstrate the application setup of EVAC as a policy evaluator and a data engine in Section A.4. For further details and to access videos with clearer indications, we strongly recommend visiting the https://annaj2178.github.io/EnerverseAC.github.io/.

Appendix A.1. Training Details

As mentioned in Section 3, we adopt a UNet-based denoising model in EVAC, which takes both latent images and concatenated conditions as inputs. The concatenated conditions include repeated latent features of the condition frame, action maps, ray maps, and a dropout mask indicating whether the condition is dropped. This dropout strategy is designed to improve the model's robustness. The hyperparameters of the model architecture and training setup are provided in Table A1.

Table A1. Model Parameters and Training Configuration. **F** indicates the fisheye camera.

Category	Configuration
Diffusion Parameters	
- Diffusion steps	1000
- Noise schedule	Linear
- β_0	0.00085
- β_T	0.0120
UNet	
- Input channels	19
- Latent image channels	4
- Condition latent image channels	4
- Action map channels	4
- Ray map channels	6
- Dropout mask channel	1
- z-shape	$40 \times 64 \times 4$
- Base channels	320
- Attention resolutions	1, 2, 4
- Channel multipliers	1, 2, 4, 4
- Blocks per resolution	2
- Context dimension	1024

Table A1. *Cont.*

Category	Configuration
Data	
- Video resolution	320×512
- Chunk size	16
- Views	head, head_left(**F**), head_right(**F**), left_hand, right_hand
Training	
- Learning Rate	5×10^{-5}
- Optimizer	Adam(β_0=0.9, β_1=0.999)
- Batch size per GPU (single-view)	8
- Batch size per GPU (multi-view)	1
- Parameterization	v-prediction
- Max steps	100,000
- Gradient clipping	0.5 (norm)

Appendix A.2. More Ablations

The effectiveness of Delta Action Attention Module To ensure the generated videos from EVAC accurately follow action trajectories, we employ a Multi-Level Action Condition Injection strategy. To assess the impact of the Delta Action Attention Module, we conducted ablation studies, with results shown in Figure A1. The task involved intricate pan manipulation dynamics, including rapidly shaking the pan, slowly shaking the pan, upward tossing, and upward shaking.

Figure A1. Results of generated videos under identical conditions with and without the Delta Action Module. The top row shows results with the module (w/ Delta Attention), while the bottom row shows results without it (w/o Delta Attention). The dashed red boxes highlight regions with inconsistent or hallucinated results.

The primary challenge lies in distinguishing between upward tossing and upward shaking, as they exhibit fundamentally different acceleration profiles. Upward tossing involves a sharp, high-acceleration movement, whereas upward shaking follows a smoother, low-acceleration trajectory. Without the Delta Action Module, spatial-aware action recognition models often fail to differentiate between these motions, resulting in incorrect predictions. This leads to temporal inconsistency, such as flickering or the sudden disappearance of objects (e.g., the ham) due to erratic motion transitions, as highlighted by the dashed red boxes in Figure A1.

The Delta Action Module addresses these limitations by introducing acceleration-aware action decomposition. By explicitly modeling the time-derivative of actions, the module captures second-order dynamics (velocity changes), enabling it to differentiate between high-acceleration motions like tossing and low-acceleration motions like shaking. As a result, the Delta Action Module ensures significantly stronger motion consistency and reduces temporal errors compared to configurations without it.

Appendix A.3. Additional Results

Appendix A.3.1. More Real-World Generated Multi-View Results

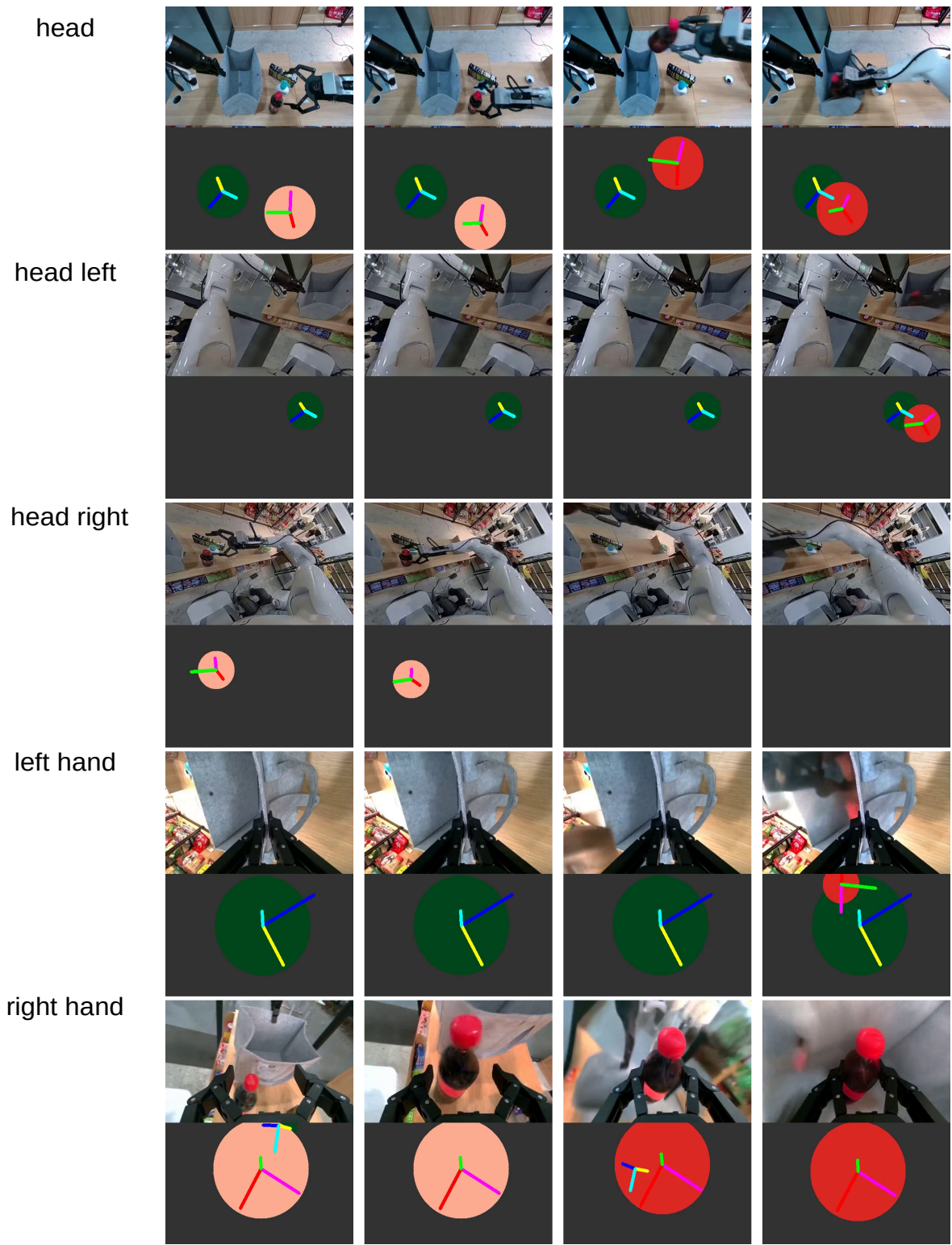In this section, we present an example of multi-view video generation, as shown in Figure A2.

Figure A2. Multi-view generated videos. This task involves placing items from a desk into a bag, specifically packaging a Coke can. Each row displays each synchronized views generated by EVAC, showcasing consistent multi-perspective results at each timestep. We have shown 4 timesteps horizontally, illustrating the dynamic action sequence.

Appendix A.3.2. Libero Results

To demonstrate the consistency and generality of EVAC, we fine-tuned the model using 417 trajectories and visualized generated action trajectories alongside Libero results. Qualitative results are shown in Figure A3.

Figure A3. Comparison of results generated by Libero and EVAC with different action trajectories. Two successful cases of picking an item are shown, along with one failure case of an empty catch.

Appendix A.3.3. Same Initial Conditions but Different Trajectories

To showcase the ability of precise trajectory-following, we present results generated under the same initial conditions but with different trajectories in Figure A4.
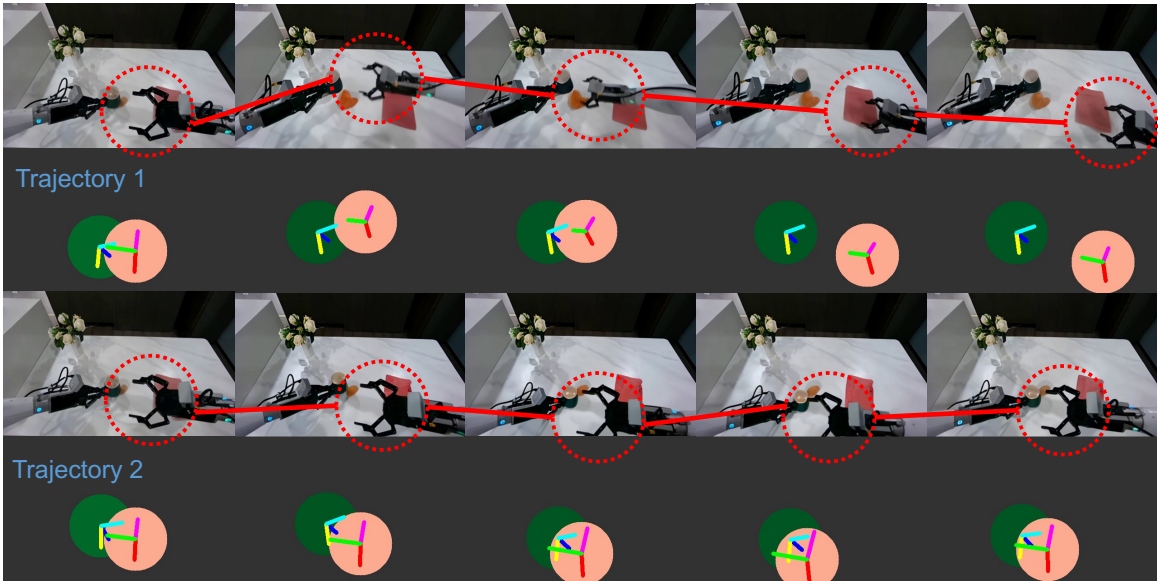
Figure A4. Results with the same initial conditions but different trajectories. The top row shows results for trajectory 1, while the bottom row shows results for trajectory 2.

Appendix A.4. Experiment Setup of EVAC*'s Application*

Appendix A.4.1. EVAC as a Policy Evaluator

We train the official single-view Go-1 [2] without the latent planner module and evaluate its performance in both real-world and simulated (EVAC) environments for comparison. During evaluation, the tester slightly randomizes the initial conditions of each task to obtain more generalized results. An example of the initial conditions for the 4 tasks is provided in Figure A5. Each task is evaluated 40 times, with success defined as the robot successfully retrieving the target item.
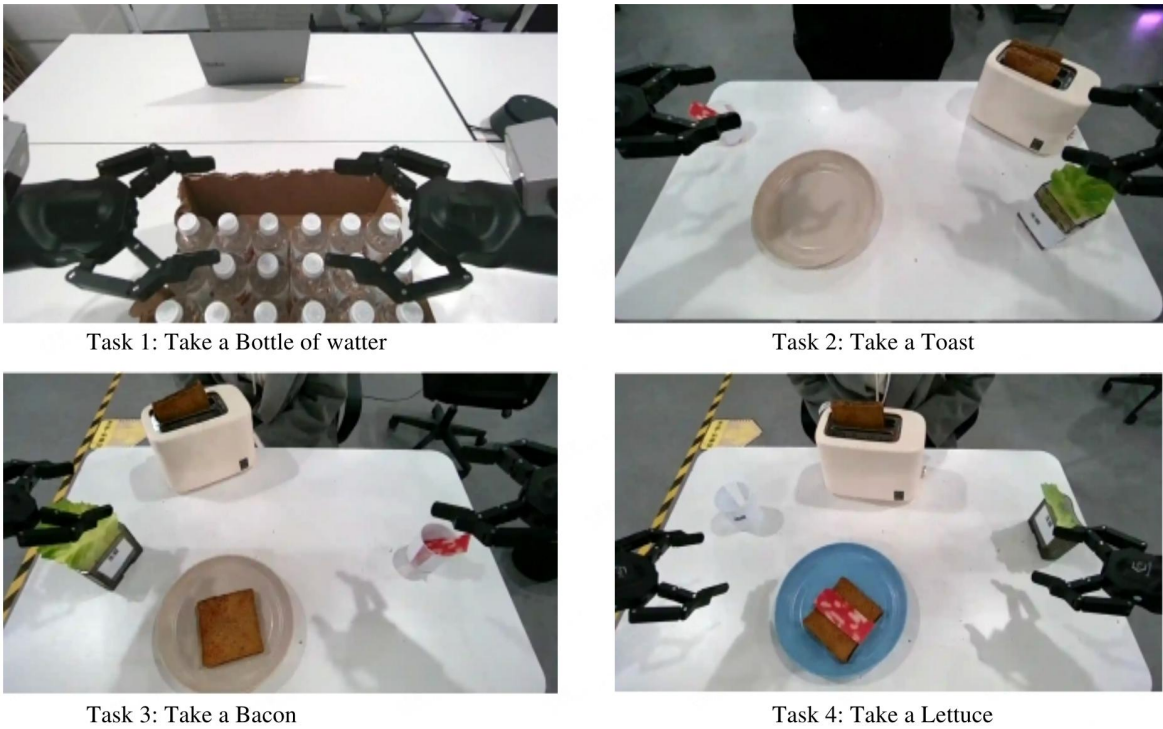

Task 1: Take a Bottle of watter

Task 2: Take a Toast

Task 3: Take a Bacon

Task 4: Take a Lettuce

Figure A5. Example of the initial conditions for the 4 tasks used in evaluation.
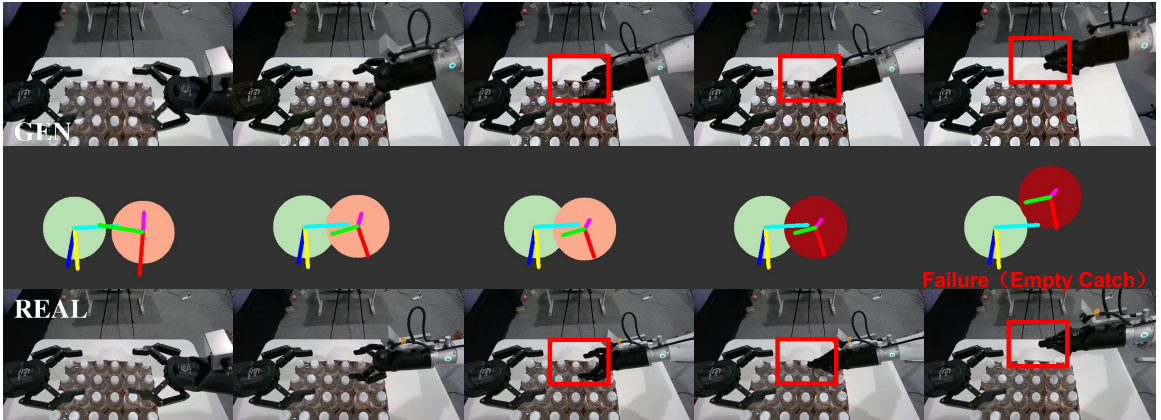
Figure A6. Task 1: Example of retrieving a bottle of water (failure case). The upper row shows the generated video from EVAC, and the lower row shows the rollout in real settings. Both are consistent in their results.
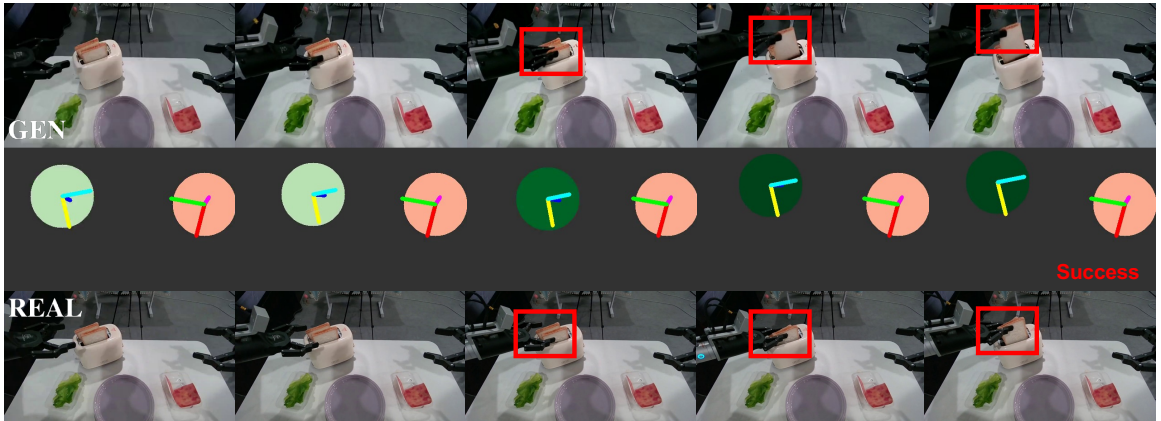

Figure A7. Task 2: Example of retrieving a piece of toast (success case). The upper row shows the generated video from EVAC, and the lower row shows the rollout in real settings. Both are consistent in their results.
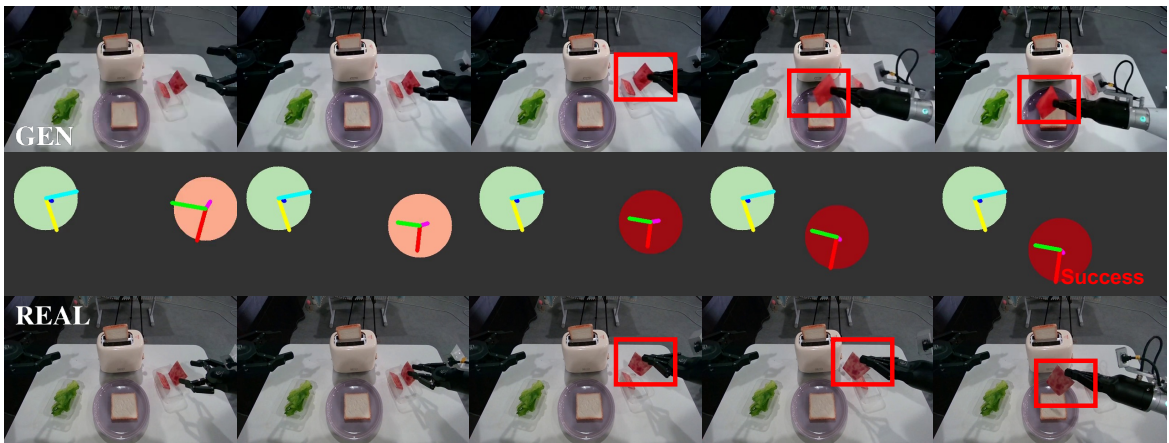

Figure A8. Task 3: Example of retrieving a ham slice (success case). The upper row shows the generated video from EVAC, and the lower row shows the rollout in real settings. Both are consistent in their results.

Figure A9. Task 4: Example of retrieving a lettuce leaf (failure case). The upper row shows the generated video from EVAC, and the lower row shows the rollout in real settings. Both are consistent in their results.
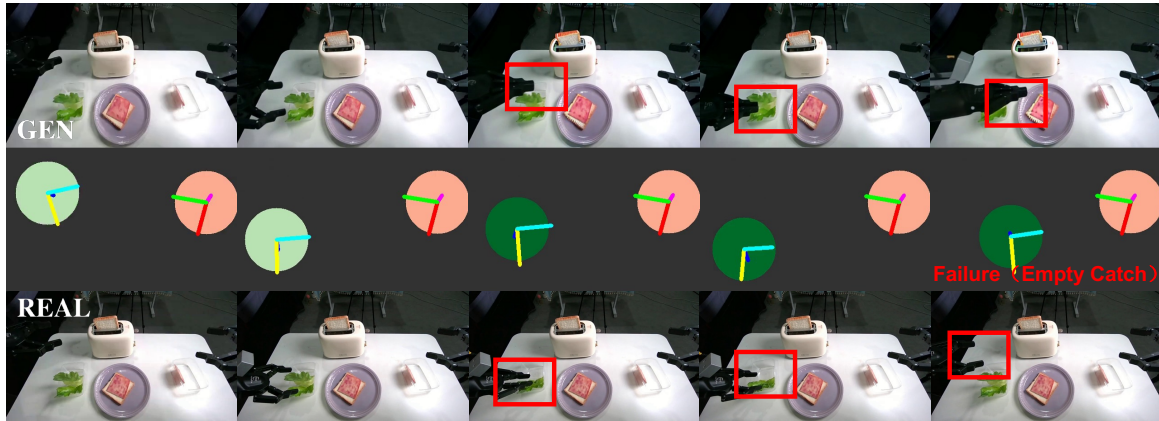
Appendix A.4.2. EVAC as a Data Engine

In this section, we visually present examples of synthetic data used for data augmentation in training, as shown in Figure A10. First, we randomly generate some actions $a'_{t_{b-N}}$ within a fixed range from $a_{t_{b-N}}$. Then, using linear interpolation between $a'_{t_{b-N}}$ and a_{t_b} to generate the action between these two. To generate the images, we first reverse the entire action sequence and use the real image as a condition for EVAC. This allows us to progressively generate images in reverse order. We then reverse the entire image sequence again for use by GO-1. Finally, we can get the image sequence from (Figure A10a) to (Figure A10b) to train a single-view Go-1 model.
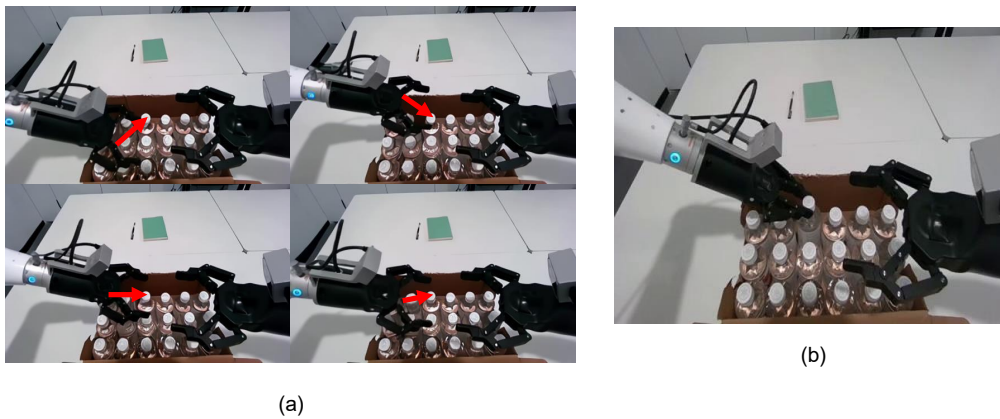

Figure A10. Example of the generated data for augmentation in Table 1. Left: (a) Spatially augmented $a'_{t_{b-N}}$ with four frame examples. The red arrows indicate different directions from the synthetic image toward the target frame. Right: (b) Fixed a_{t_b} frame. Frames between (a) and (b) are generated using linear interpolation.

References

1. S. Huang, L. Chen, P. Zhou, S. Chen, Z. Jiang, Y. Hu, Y. Liao, P. Gao, H. Li, M. Yao, and G. Ren. Enerverse: Envisioning embodied future space for robotics manipulation, 2025. URL https://arxiv.org/abs/2501.01895.
2. Q. Bu, J. Cai, L. Chen, X. Cui, Y. Ding, S. Feng, S. Gao, X. He, X. Huang, S. Jiang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.
3. T. Brooks, B. Peebles, C. Holmes, W. DePue, Y. Guo, L. Jing, D. Schnurr, J. Taylor, T. Luhman, E. Luhman, C. Ng, R. Wang, and A. Ramesh. Video generation models as world simulators. 2024. URL https://openai.com/research/video-generation-models-as-world-simulators.

4. J. Bruce, M. D. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

5. S. Yang, Y. Du, K. Ghasemipour, J. Tompson, L. Kaelbling, D. Schuurmans, and P. Abbeel. Learning interactive real-world simulators, 2024. URL https://arxiv.org/abs/2310.06114.

6. J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020. URL https://arxiv.org/abs/2006.11239.

7. R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. *CoRR*, abs/2112.10752, 2021. URL https://arxiv.org/abs/2112.10752.

8. Y. Ma, Y. He, X. Cun, X. Wang, S. Chen, Y. Shan, X. Li, and Q. Chen. Follow your pose: Pose-guided text-to-video generation using pose-free videos, 2024. URL https://arxiv.org/abs/2304.01186.

9. L. Hu, X. Gao, P. Zhang, K. Sun, B. Zhang, and L. Bo. Animate anyone: Consistent and controllable image-to-video synthesis for character animation, 2024. URL https://arxiv.org/abs/2311.17117.

10. S. Zhou, Y. Du, J. Chen, Y. Li, D.-Y. Yeung, and C. Gan. Robodreamer: Learning compositional world models for robot imagination, 2024. URL https://arxiv.org/abs/2404.12377.

11. B. Wang, N. Sridhar, C. Feng, M. V. der Merwe, A. Fishman, N. Fazeli, and J. J. Park. This&that: Language-gesture controlled video generation for robot planning, 2024. URL https://arxiv.org/abs/2407.05530.

12. 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*, pages 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

13. E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021.

14. 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, 2021. URL https://arxiv.org/abs/2108.10470.

15. OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang. Solving rubik's cube with a robot hand. *CoRR*, abs/1910.07113, 2019. URL http://arxiv.org/abs/1910.07113.

16. A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2023. URL https://arxiv.org/abs/2212.06817.

17. A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

18. O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

19. M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

20. Q. Bu, H. Li, L. Chen, J. Cai, J. Zeng, H. Cui, M. Yao, and Y. Qiao. Towards synergistic, generalized, and efficient dual-system for robotic manipulation, 2025. URL https://arxiv.org/abs/2410.08001.

21. Physical Intelligence. Pi0: [title of the blog post]. https://www.physicalintelligence.company/blog/pi0, 2025. Accessed: 2025-04-23.

22. S. Huang, Y. Liao, S. Feng, S. Jiang, S. Liu, H. Li, M. Yao, and G. Ren. Adversarial data collection: Human-collaborative perturbations for efficient and robust robotic imitation learning. *arXiv preprint arXiv:2503.11646*, 2025.

23. J. Xing, M. Xia, Y. Zhang, H. Chen, W. Yu, H. Liu, X. Wang, T.-T. Wong, and Y. Shan. Dynamicrafter: Animating open-domain images with video diffusion priors. *arXiv preprint arXiv:2310.12190*, 2023.

24. Z. Wang, Z. Yuan, X. Wang, Y. Li, T. Chen, M. Xia, P. Luo, and Y. Shan. Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024.

25. X. Wang, H. Yuan, S. Zhang, D. Chen, J. Wang, Y. Zhang, Y. Shen, D. Zhao, and J. Zhou. Videocomposer: Compositional video synthesis with motion controllability. *Advances in Neural Information Processing Systems*, 36:7594–7611, 2023.

26. T. Wang, L. Li, K. Lin, Y. Zhai, C.-C. Lin, Z. Yang, H. Zhang, Z. Liu, and L. Wang. Disco: Disentangled control for realistic human dance generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9326–9336, 2024.

27. X. Li, L. Xu, J. Liu, M. Zhang, J. Xu, S. Huang, I. Ponomarenko, Y. Shen, S. Zhang, and H. Dong. Crayonrobo: Toward generic robot manipulation via crayon visual prompting. *arXiv preprint arXiv:2505.02166*, 2025.

28. W. Lin, X. Wei, R. An, P. Gao, B. Zou, Y. Luo, S. Huang, S. Zhang, and H. Li. Draw-and-understand: Leveraging visual prompts to enable mllms to comprehend what you want. *arXiv preprint arXiv:2403.20271*, 2024.

29. A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

30. J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

31. A. Jaegle, F. Gimeno, A. Brock, O. Vinyals, A. Zisserman, and J. Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021.

32. R. Gao, A. Hoł yński, P. Henzler, A. Brussee, R. Martin-Brualla, P. Srinivasan, J. T. Barron, and B. Poole. Cat3d: Create anything in 3d with multi-view diffusion models. In *Advances in Neural Information Processing Systems*, volume 37, pages 75468–75494, 2024.

33. B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

34. D. Ha and J. Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems 31*, pages 2451–2463. Curran Associates, Inc., 2018. URL https://papers.nips.cc/paper/7512-recurrent-world-models-facilitate-policy-evolution. https://worldmodels.github.io.

35. D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models, 2022.

36. D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models, 2023.