GICAT: GOAL-CONDITIONED IN-CONTEXT ACTION TRANSLATION FOR GENERALIZABLE ROBOT MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

General-purpose robots need to interpret diverse task goals, anticipate feasible changes in the environment, and realize those changes through embodiment-specific control. Video generation models provide a promising action-free planning interface, but their planning predictions cannot be directly executable by robots. Therefore, we present GICAT, a decoupled closed-loop framework that separates video planning from action generation. Given a current image and task instruction, a video generation model in GICAT predicts a short-horizon visual future. Unlike frame-by-frame action translation, an in-context action translation part of GICAT conditions a flow-matching model on the current image, goal image, robot state, and retrieved in-context demonstrations to generate executable action chunks. By training the action translation model as an in-context learner, we can apply it to unseen tasks during the inference stage without any weight updates. During execution, GICAT follows the predicted action chunk in a receding-horizon loop and replans from real environment observations. GICAT outperforms decoupled video-to-action baselines on Meta-World and ManiSkill and shows stronger cross-task generalization to unseen tasks during in-context action translation training. These results highlight the potential of decoupled video planning and in-context action learning as an effective route toward generalizable video-model-based robot control.

1 INTRODUCTION

Learning-based robotics has moved from task-specific policies trained on narrow demonstrations toward foundation model policies that combine perception, language, and action. Vision-language-action (VLA) models instantiate this direction by adapting large vision-language backbones to robot trajectories, enabling a single policy to map images and language instructions to motor commands while inheriting semantic knowledge from web-scale pretraining (Brohan et al., 2023a; Kim et al., 2025). However, this interface entangles high-level reasoning, physical prediction, and embodiment-specific action decoding inside one end-to-end model. Scaling such policies therefore remains tied to action-labeled robot data, which is costly to collect, heterogeneous across platforms, and difficult to transfer across embodiments.

World models offer a more modular route by separating prediction from control. Latent-dynamics methods use imagined rollouts for planning or policy optimization (Hafner et al., 2019; 2025; Hansen et al., 2024), but are typically optimized for action-conditioned control within a fixed environment interface. Other approaches jointly model future observations and robot actions for manipulation (Wu et al., 2024; Zhu et al., 2025), improving perception-control consistency while still requiring action-labeled data and a specific action representation. A more decoupled paradigm uses video generation as a planning interface, then translates predicted visual futures into actions with a downstream controller or inverse dynamics model (IDM) (Du et al., 2023; 2024; Zhou et al., 2024). This video-model with IDM design separates what should happen in the scene from how a specific robot executes it (Du et al., 2023; Li et al., 2026), but shifts the main bottleneck to the IDM. The IDM must realize image-space changes as kinematically valid actions across scenes and tasks.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

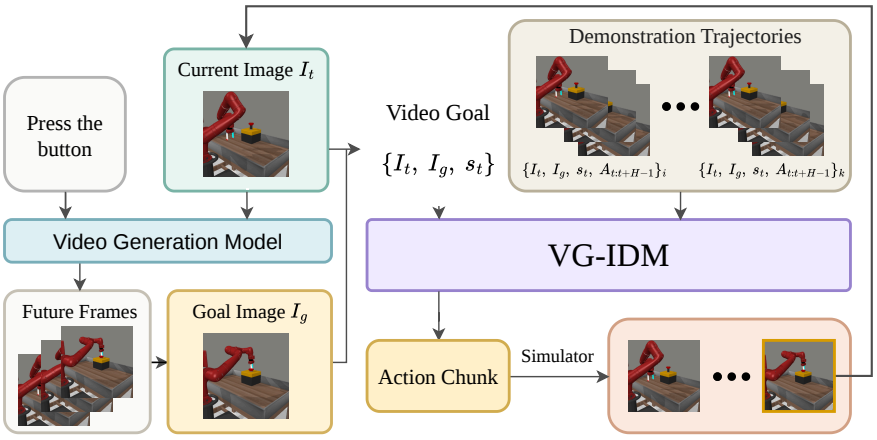


Figure 1: Overview of GICAT. Video-generated goals are translated into executable action chunks by VG-IDM with in-context demonstrations in a closed-loop pipeline.

In this work, we focus on the problem of translating video goal conditions into executable robot action chunks and we define the video goal as the tuple consisting of the current image, the goal image, and the current robot state. To address the problem, we propose the Video Goal Inverse Dynamics Model (VG-IDM), a retrieval-conditioned action model that maps this video goal condition and retrieved expert demonstrations to a multi-step action chunk. Given the current video-goal condition, VG-IDM retrieves related snippets from expert trajectory datasets as in-context demonstrations. Trained across diverse video goals and demonstration contexts, the model learns to infer task-specific action translations from these demonstrations rather than relying on a fixed visual-to-action mapping. At inference time, it can therefore leverage retrieved demonstrations to adapt its action translation to unseen task video goals without any parameter updates. VG-IDM uses flow matching to learn a distribution over action chunks with a transformer policy (Lipman et al., 2023), enabling multimodal action generation and receding-horizon execution. We pair VG-IDM with a video generation model that predicts goal images from the current observation and task instruction. We call the resulting two-stage framework Goal-conditioned In-Context Action Translation (GICAT), whose overall pipeline is shown in Figure 1.

We evaluate GICAT on two benchmarks, Meta-World (McLean et al., 2025) and ManiSkill (Tao et al., 2025), and compare it with several baseline methods. The results show that GICAT achieves better stability and generalization on both seen and unseen tasks.

Our contributions are threefold:

- First, we propose VG-IDM, a video-goal-conditioned inverse dynamics model for translating video goals into executable action chunks.
- Second, we combine VG-IDM with a video generation model to build GICAT, a two-stage framework for video-based robot manipulation.
- Third, we evaluate GICAT on Meta-World and ManiSkill, demonstrating strong robustness and generalization across diverse manipulation tasks.

2 Related Work

Vision-Language-Action Models Vision-language-action (VLA) models learn generalist robot policies that map visual observations and language instructions directly to actions. Large robot transformers and recent VLA systems, including RT-1, RT-2, RT-X, OpenVLA, Octo, and π_0, show that large-scale robot data and vision-language pretraining can improve semantic generalization and cross-task control (Brohan et al., 2023b;a; O'Neill et al., 2024; Kim et al., 2025; Ghosh et al., 2024; Black et al., 2025). However, these methods tightly couple task understanding and embodiment-specific control within a single policy. In contrast, GICAT decouples visual planning from action generation, resulting in a more interpretable and generalizable framework.

Video Generation Models for Robot Planning Video generation models provide an action-free representation for predicting how a scene should evolve. General video generators have improved the quality and temporal coherence of generated videos (Ho et al., 2022; Blattmann et al., 2023; Kondratyuk et al., 2024), and robotics methods have used generated images or videos as plans for control. UniPi formulates decision making as text-conditioned video generation followed by action extraction (Du et al., 2023); VLP searches over video-language plans (Du et al., 2024); SuSIE uses image editing to propose visual subgoals (Black et al., 2024); and RoboDreamer studies compositional video prediction for unseen object-action combinations (Zhou et al., 2024). These works demonstrate the value of video as a planning medium, but generated video goals still need to be grounded into executable robot actions.

World Action Models Another line of work jointly models future observations and actions. GR-1 and GR-2 predict both videos and robot actions after large-scale video pretraining and robot-data adaptation (Wu et al., 2024; Cheang et al., 2024). Unified World Models couple video and action diffusion so that one model can serve as a policy, forward model, inverse model, or video generator (Zhu et al., 2025). DreamZero and τ_0-WM further build large video-action world models for action generation and candidate rollout evaluation (Ye et al., 2026; Zhou et al., 2026). These approaches improve consistency between predicted scene evolution and action, but they require training or adapting large coupled video-action models. GICAT instead keeps video planning and action translation decoupled, reducing training cost while preserving a more interpretable architecture.

Video Generation Models with IDMs The closest works to ours separate visual planning from action grounding. UniPi generates future frames and extracts actions with an inverse dynamics model (Du et al., 2023). AVDC reduces action supervision by using synthetic videos and dense frame correspondences to infer actions (Ko et al., 2024). Dreamitate and Gen2Act use generated videos as motion references and translate them into robot execution through tracking or video-conditioned policies (Liang et al., 2025; Bharadhwaj et al., 2025). VERA keeps the video planner fixed and trains an embodiment-specific Jacobian inverse dynamics model for closed-loop control (Li et al., 2026), while recent alignment work shows that inverse dynamics can expose the gap between visually plausible videos and executable actions (Wang et al., 2026). GICAT follows this decoupled paradigm, but uses a retrieval-conditioned VG-IDM that maps video-goal conditions and in-context demonstrations to multi-step action chunks.

3 METHOD

GICAT follows a two-stage design for generalizable robot manipulation from video plans. First, a video generation model fine-tuned on robot manipulation datasets is used as a video planner to predict how the current observation could evolve toward the task goal and provide a video plan. Second, VG-IDM translates this video plan into a sequence of low-level robot actions through a retrieval-conditioned inverse dynamics model. The details of the data preparation, model components, and training setup are provided in Appendix A.

3.1 PROBLEM FORMULATION AND METHOD OVERVIEW

At time t, we sample a current robot observation that is an RGB image I_t, proprioceptive state of the robot s_t, and a task prompt p. A video planner first predicts a plausible visual future,

$$\hat{Q}_t = \mathcal{V}_\psi(I_t, p) = [\hat{I}_{t+1}, \ldots, \hat{I}_{t+M}], \tag{1}$$

where $\mathcal{V}_\psi$ is a text-and-image-conditioned video generation model. We then use the final frame of the predicted video as the visual goal I_g. We define the video goal V_g as

$$V_g = \{I_t, I_g, s_t\}. \tag{2}$$

To support in-context action translation, we retrieve a small set of expert demonstration segments relevant to the current video goal, forming the demonstration context C_t. Then, the IDM converts the video goal V_g and retrieved demonstration context into an executable action chunk,

$$A_{t:t+H-1} \sim \pi_\theta(V_g | C_t), \tag{3}$$

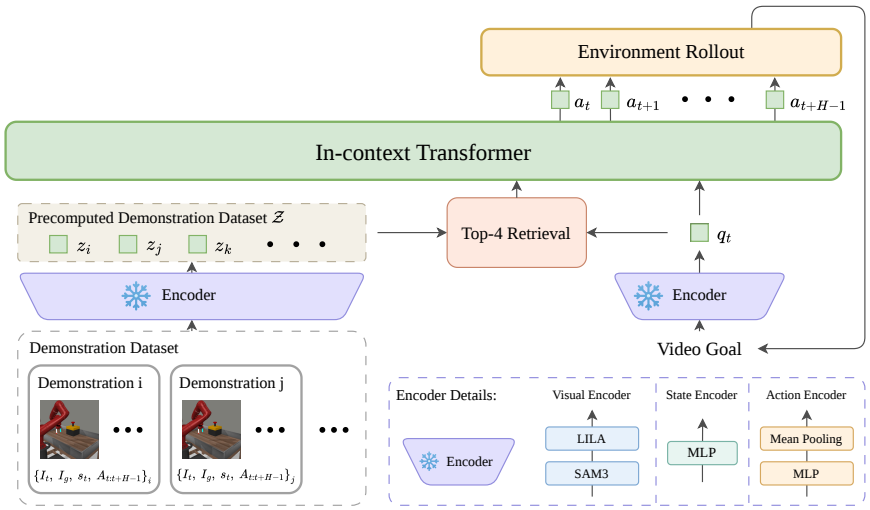


Figure 2: Overview of VG-IDM. The current video goal retrieves relevant demonstration tokens, which are used by the in-context transformer to generate executable action chunks.

where $A_{t:t+H-1} = [a_t, \ldots, a_{t+H-1}]$ is an H-step action sequence and C_t denotes a small set of retrieved in-context demonstrations.

This decomposition separates video planning from embodied execution: the video generation model proposes task-level visual progress, while the IDM grounds the predicted visual transition into robot-specific actions. The key challenge lies in bridging the substantial gap between the high-dimensional and appearance-sensitive visual observation space and the low-dimensional action space that requires precise control. Moreover, a generalizable IDM must go beyond memorizing task-specific visual-to-action correspondences and instead infer the interaction and embodiment dynamics implicitly encoded in expert demonstrations, such that these dynamics can be transferred to previously unseen tasks. To address these challenges, we formulate the IDM as a retrieval-conditioned flow-matching action generator. By conditioning action prediction on both the video goal and dynamically retrieved expert snippets, the model can use relevant execution patterns as in-context dynamics priors, supporting more robust visual-to-action grounding and generalization beyond the training tasks.

3.2 Video Goal Inverse Dynamics Learning

VG-IDM addresses the problem of translating a predicted visual transition into low-level robot actions. A direct solution would be to regress an action sequence from the V_g. However, this mapping is highly multimodal: the same visual change can be achieved by different contact sequences, gripper timings, and intermediate motions. We therefore train VG-IDM as a retrieval-conditioned flow-matching inverse dynamics model. During training, demonstrations are converted into current-goal-action tuples. The model learns a conditional action distribution with a DiT-style action transformer, while retrieved expert snippets provide in-context execution references for the current visual transition.

Data preparation. We train VG-IDM from robot demonstration trajectories. Each trajectory contains RGB observations, proprioceptive states, and low-level actions. For each trajectory, we sample a current time step t, a future offset h, and an action horizon H to construct a training tuple $(I_t, I_{t+h}, s_t, A_{t:t+H-1})$, and $A_{t:t+H-1} = [a_t, \ldots, a_{t+H-1}]$ is the expert action chunk. During the IDM training, I_{t+h} serves as the visual goal I_g, providing supervision for mapping visual progress to the action sequence that realizes it.

Robot-centric visual encoding. For visual grounding, we adopt a robot-centric representation for the IDM. Since the IDM predicts actions from the current and future robot observations, its

visual representation should emphasize robot-arm information while suppressing irrelevant appearance variations such as background textures and lighting. Otherwise, the model may exploit scene-specific visual correlations instead of learning transferable embodied dynamics.

Following this first-principles reasoning, given a raw RGB frame, we apply SAM3 to obtain a robot-arm-focused masked image and encode it using a frozen LILA visual backbone (Carion et al., 2026; Araslanov et al., 2026). The resulting feature is denoted by $\phi(I)$.

Retrieval-conditioned action grounding. The retrieval module provides in-context execution references for each query transition, as illustrated in Fig. 2. We first encode the current query using the current image, goal image, and robot state:

$$q_t = f_q([\phi(I_t), \phi(I_g), f_s(s_t)]), \tag{4}$$

where $f_s(\cdot)$ denotes the state encoder and $[\cdot]$ denotes feature concatenation.

Prior to training and inference, we preprocess all demonstration segments into retrieval keys. For each demonstration segment $(I_i, I_{i+h}, s_i, A_{i:i+H-1})$, we compute an action-free retrieval key $q_i = f_q([\phi(I_i), \phi(I_{i+h}), f_s(s_i)])$. At each query step, we compare q_t with these keys by cosine similarity and retrieve the four most similar segments. Each retrieved segment is then encoded into a context token $z_i^c = f_c([\phi(I_i), \phi(I_{i+h}), f_s(s_i), \bar{f}_a(A_{i:i+H-1})])$, where $\bar{f}_a(\cdot)$ denotes the temporally pooled action representation. The resulting context is

$$C_t = \left\{ z_i^c \mid i \in \mathrm{Top4}_j[\cos(q_t, q_j)] \right\}. \tag{5}$$

DiT with flow matching. VG-IDM in GICAT models the action chunk as a conditional flow-matching generation problem. Let A_1 denote the ground-truth expert action chunk. We sample Gaussian noise $A_0 \sim \mathcal{N}(0, I)$ and a continuous time variable $\tau \sim \mathcal{U}(0, 1)$. The interpolated action chunk is

$$A_\tau = (1 - \tau)A_0 + \tau A_1, \tag{6}$$

with the corresponding target velocity

$$u_\tau = A_1 - A_0. \tag{7}$$

The model predicts this velocity field conditioned on V_g and retrieved context:

$$\hat{u}_\theta = v_\theta(A_\tau, \tau, V_g | C_t). \tag{8}$$

The conditioning tokens consist of the current-image token z_t, the goal-image token z_g, the state token z_s, and a set of context tokens $\{z_i^c\}_{i=1}^N$. The noisy action chunk is encoded into action tokens and augmented with the time embedding of τ. The full transformer input is $[z_t, z_g, z_s, z_1^c, \ldots, z_N^c, z_{a_1}^\tau, \ldots, z_{a_H}^\tau]$. We use type embeddings to distinguish current-image, goal-image, state, context, and action tokens, and rotary position embeddings to encode token order. A causal action mask allows each action token to attend to all conditioning tokens and previous action tokens, while preventing access to future action tokens.

The training objective combines the flow-matching loss with a temporal continuity regularizer:

$$\mathcal{L} = \|v_\theta(A_\tau, \tau, V_g | C_t) - u_\tau\|_2^2 + \lambda_{\mathrm{cont}} \frac{1}{H-1} \sum_{j=1}^{H-1} \|\hat{u}_{j+1} - \hat{u}_j\|_2^2. \tag{9}$$

Additional training details are provided in Appendix A.

3.3 Video Planning and Closed-Loop Execution

We adapt a pretrained text-and-image-to-video model, Wan2.2-TI2V-5B, as the visual planner using LoRA fine tuning (Team Wan et al., 2025; Hu et al., 2022). Given the current image I_t and task prompt p, the planner generates a short-horizon future video, whose final frame is used as the visual goal I_g for VG-IDM. Details of the video model adaptation and training setup are provided in Appendix B.

During closed-loop execution, GICAT operates in a receding-horizon manner, as shown in Fig. 2. At each planning round, the system observes the current image I_t and robot state s_t, generates a new

visual plan conditioned on I_t and p, and uses the final generated frame as the visual goal. VG-IDM then retrieves relevant demonstration context and predicts an H-step action chunk $A_{t:t+H-1}$. After execution, the system observes the updated environment and starts the next planning round. This closed-loop design prevents long-horizon execution from relying entirely on an open-loop generated video, since each planning round is grounded in the latest environment observation rather than previously generated frames.

4 EXPERIMENTS

We evaluate GICAT and other baseline methods on Meta-World and ManiSkill to study how effectively different inverse dynamics models translate video planner guidance into executable robot action chunks. The evaluation is related to video generation quality, since the generated visual plan provides the goal signal for control. However, our focus is not to assess the video model as a standalone generative model. Instead, we fix the video planner across methods and measure how well each action model converts the same generated visual plan into low-level closed-loop control. Qualitative rollout results of GICAT on the two benchmarks are shown in Fig. 3. The experimental settings and results are presented below.

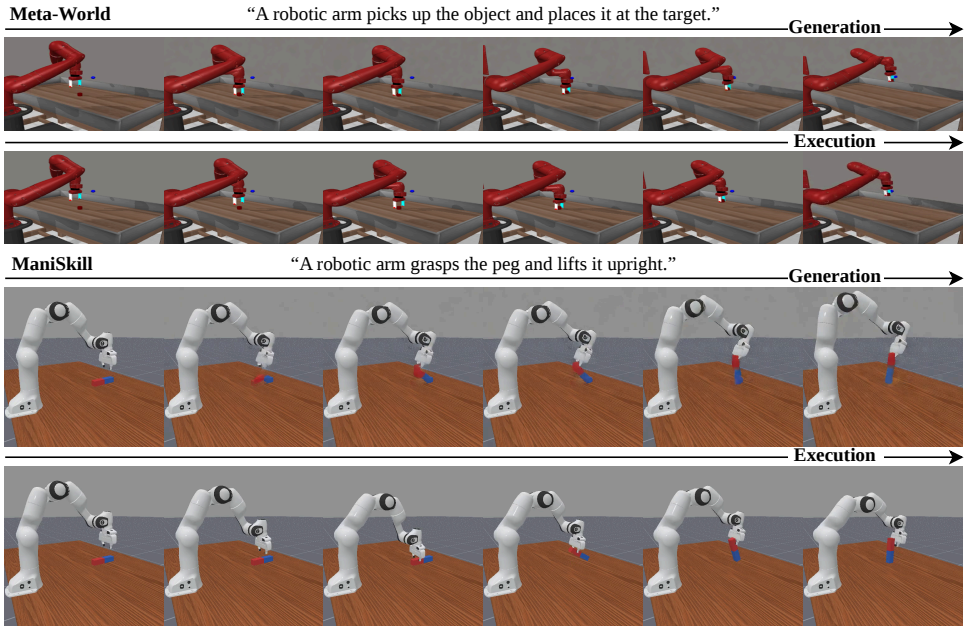


Figure 3: GICAT performance on Meta-World and ManiSkill. For each benchmark, we show the video plan generated by the video generation model and the corresponding execution produced by VG-IDM conditioned on the video goal.

4.1 EXPERIMENTAL SETUP

Benchmark. We evaluate our method on two complementary robotic manipulation benchmarks: Meta-World and ManiSkill. Meta-World provides a standardized multi-task benchmark with 50 tabletop manipulation tasks performed by a Sawyer robot arm. The action space is 4-dimensional, consisting of 3D end-effector displacement control and one-dimensional gripper control, which makes it well suited for assessing multi-task generalization in a low-dimensional and structurally consistent setting. In ManiSkill, we use a Franka Emika Panda robot arm to perform tabletop manipulation tasks, where the policy outputs 7-dimensional actions, consisting of 3D end-effector position control, 3D end-effector orientation control, and one-dimensional gripper control. This setting enables us to evaluate the robustness of our method in a higher-dimensional control space with

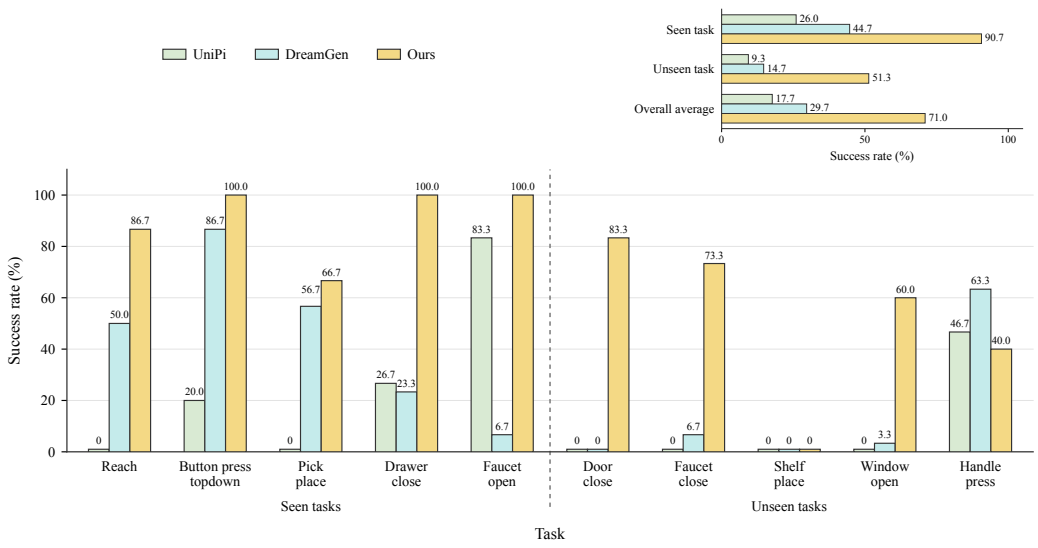


Figure 4: Closed-loop success rates on ten Meta-World tasks. We compare UniPi, DreamGen, and GICAT across five seen tasks and five unseen tasks. GICAT achieves the highest success rate on most tasks and shows substantially stronger generalization on unseen tasks.

more complex contact dynamics. Together, these two benchmarks allow us to examine whether our method remains effective across different control dimensions, robot embodiments, and task implementations.

Video planner. All video models used in our experiments are based on the Wan2.2-TI2V-5B model fine-tuned with LoRA for robotic manipulation videos. We fine-tune two separate video models as planners for the Meta-World and ManiSkill benchmarks, respectively.

During evaluation, the planner takes the current image and the task prompt as input, and generates a 9-frame future video. In Meta-World, videos are rendered from the corner camera at a resolution of 256×256. For ManiSkill, we use a higher rendering resolution of 512×512 during evaluation, since we found that the tasks in the ManiSkill simulator are more complex and require finer visual details.

Since different inverse dynamics models share the same planner, the comparison mainly focuses on how each inverse dynamics model converts the predicted visual future into executable actions. We also note that although our evaluation is designed to assess the ability of different inverse dynamics models to execute visual goals, the final success rate is still closely tied to the quality of the future visual frames generated by the video planner.

Baselines. We compare our method with two video-to-action baselines. The first baseline is DreamGen (Jang et al., 2025), which relies only on visual inputs and does not use robot state and retrieval context. It uses a SigLIP2-Large visual encoder (Tschannen et al., 2025), visual self-attention, and a DiT-based action model trained with flow matching.

The second baseline is a UniPi-style inverse dynamics model (Du et al., 2023), which takes the predicted next frame as input and directly regresses executable actions using a small CNN trained with mean-squared error. This baseline represents a simpler frame-conditioned action regression approach.

We choose these two baselines because they cover two representative video-to-action paradigms: a stronger generative inverse dynamics baseline based on flow matching, and a simpler next-frame-conditioned regression baseline. Comparing against both allows us to evaluate whether our method improves over visual action prediction approaches with different model capacities, action horizons, and conditioning information.

4.2 META-WORLD

In Meta-World, we select ten tasks for evaluation, consisting of five seen tasks and five unseen tasks. The seen tasks are reach-v3, button-press-topdown-v3, pick-place-v3, drawer-close-v3, and faucet-open-v3. The unseen tasks are door-close-v3, faucet-close-v3, shelf-place-v3, window-open-v3, and handle-press-v3. We choose these tasks because they cover diverse manipulation capabilities. Among these tasks, the five seen tasks correspond to task categories that are included in the VG-IDM training data, whereas the remaining five unseen tasks belong to task categories that are not present in the training set. This split evaluates closed-loop performance and IDM generalization under a shared video planner. During evaluation, GICAT uses the Context80 retrieval pool, containing eight demonstration trajectories from each of the ten evaluation tasks.

Evaluation on Meta-World Tasks Fig. 4 reports the closed-loop performance of GICAT and two baseline policies on the same ten-task Meta-World suite. GICAT achieved an overall success rate of 71.0%, substantially outperforming DreamGen (29.7%) and UniPi (17.7%). On the seen tasks used for IDM training, GICAT reached 90.7% success, indicating stronger execution accuracy and stability than the baselines. More importantly, on tasks held out from IDM training, GICAT achieved 51.3% success, compared with 14.7% for DreamGen and 9.3% for UniPi. These results show that GICAT generalizes more effectively to unseen manipulation tasks. We attribute this generalization ability to its decoupled architecture and to the task-prior information provided by in-context demonstrations, which we further examine through detailed ablation studies shown in Fig. 5.

4.3 MANISKILL

In ManiSkill, we evaluate six tasks, including four task types seen during VG-IDM training and two unseen task types. The seen tasks are PickCube, LiftPegUpright, PushCube, and StackCube, while the unseen tasks are PokeCube and PullCube. We select these six tasks from the diverse manipulation tasks supported by ManiSkill because they form a representative suite for evaluating different interaction skills. This split evaluates closed-loop performance and IDM generalization under a shared video planner. During evaluation, GICAT uses the Context48 retrieval pool, containing eight demonstration trajectories from each of the six evaluation tasks.

Table 1: Closed-loop rollout success rates on ManiSkill. Each task is evaluated over 30 rollouts. The first four tasks in Table 1 correspond to task types seen during IDM training, while the remaining two task types are absent from the training data.

Task	UniPi	DreamGen	Ours
PickCube-v1	0.0%	0.0%	0.0%
LiftPegUpright-v1	0.0%	0.0%	**66.7%**
PushCube-v1	13.3%	13.3%	**23.3%**
StackCube-v1	0.0%	0.0%	0.0%
PullCube-v1	0.0%	0.0%	**3.3%**
PokeCube-v1	10.0%	10.0%	**13.3%**
Seen average	3.3%	3.3%	**22.5%**
Unseen average	5.0%	5.0%	**8.3%**
Overall	3.9%	3.9%	**17.8%**

Evaluation on ManiSkill Tasks Table 1 reports the closed-loop performance of GICAT and two baseline policies on the six-task ManiSkill suite. GICAT achieved the highest overall success rate of 17.8%, outperforming DreamGen (3.9%) and UniPi (3.9%) under the same video-planner setting. On the seen tasks used for IDM training, GICAT reached 22.5% success, compared with 3.3% for DreamGen and 3.3% for UniPi. The gains were mainly observed on LiftPegUpright-v1 and PushCube-v1, where GICAT succeeded in 20/30 and 7/30 rollouts, respectively. On the held-out IDM tasks, GICAT also achieved the best average success rate of 8.3%, compared with 5.0% for DreamGen and 5.0% for UniPi. Although the absolute success rates on ManiSkill remain lower than those on Meta-World, these results show the same overall trend: the retrieval-conditioned IDM

translates the shared visual plan into executable actions more effectively than the visual-only inverse dynamics baselines in a higher-dimensional control setting.

4.4 Ablation

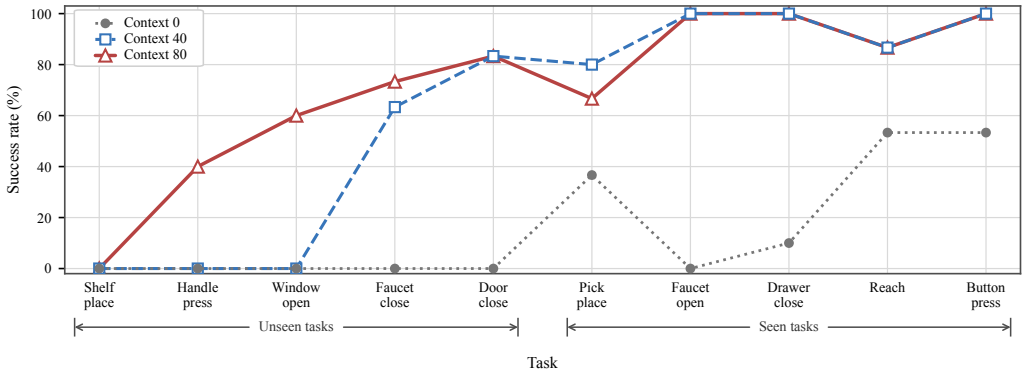


Figure 5: Ablation results under different context configurations on Meta-World. We compare GICAT with no context, Context40, and Context80 across five seen tasks and five unseen tasks. Each task is evaluated over 30 closed-loop rollouts.

To examine whether GICAT effectively learns task- and action-relevant information from the demonstrations provided as context, we evaluate its closed-loop rollout performance on multiple Meta-World tasks under different context sizes. We consider three settings: Context0, which uses no demonstration trajectories; Context40, which contains 40 trajectories from seen tasks only; and Context80, which contains 80 trajectories covering all task categories.

As shown in Fig. 5, Context0 performs substantially worse than the other two settings, while Context80 shows notably stronger generalization on unseen tasks. On seen tasks, Context80 remains competitive overall, despite a slight drop on Pick Place compared with Context40, possibly because the demonstrations in Context40 are more closely matched to the evaluated seen tasks. These results indicate that GICAT can effectively exploit information from demonstrations, improving both the stability and generalization of video-plan-to-action generation.

5 Discussion and Conclusion

Conclusion In this work, we propose GICAT, a two-stage framework that pairs video planning with VG-IDM for closed-loop robot manipulation. VG-IDM uses flow matching and retrieved in-context demonstrations to translate video-goal conditions into action chunks without inference-time parameter updates. Experiments on Meta-World and ManiSkill show that GICAT achieves higher success rates than video-to-action baselines on both seen and unseen tasks under shared video planners. These results support retrieval-conditioned action translation as a promising approach for video-based robot control. Future work will focus on reducing video-generation latency and evaluating GICAT on physical robots.

Limitations Due to hardware and time constraints, we have not yet deployed GICAT on a physical robot, so the current results do not directly evaluate robustness to real-world sensing noise, actuation delay, calibration errors, or sim-to-real dynamics gaps. In addition, GICAT adopts a decoupled two-stage design, making closed-loop execution dependent on the quality and speed of the video model. In particular, video generation introduces non-negligible inference latency, which remains a bottleneck for real-time closed-loop control.

AI USE STATEMENT

We used OpenAI ChatGPT and Codex to assist with grammar checking, formatting, literature retrieval and discovery, and drafting portions of the manuscript. We also used these tools to assist with writing and editing portions of the research code, as well as to support experimental design and provide feedback on experiments. All AI-assisted content and code were reviewed and verified by the authors. The authors take full responsibility for the final content of this work, including its text, references, methods, results, and conclusions.

REFERENCES

Nikita Araslanov, Martin Sundermeyer, Hidenobu Matsuki, David Joseph Tan, and Federico Tombari. Featurising pixels from dynamic 3D scenes with linear in-context learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2Act: Human video generation in novel scenarios enables generalizable robot manipulation. In *Proceedings of The 9th Conference on Robot Learning*, 2025.

Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pre-trained image-editing diffusion models. In *International Conference on Learning Representations*, 2024.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A Vision-Language-Action Flow Model for General Robot Control. In *Proceedings of Robotics: Science and Systems*, 2025.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets, 2023. arXiv:2311.15127.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of The 7th Conference on Robot Learning*, 2023a.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, et al. RT-1: Robotics Transformer for Real-World Control at Scale. In *Proceedings of Robotics: Science and Systems*, 2023b.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. SAM 3: Segment anything with concepts. In *International Conference on Learning Representations*, 2026.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. GR-2: A generative video-language-action model with web-scale knowledge for robot manipulation, 2024. arXiv:2410.06158.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. In *Advances in Neural Information Processing Systems*, 2023.

Yilun Du, Sherry Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Brian Ichter, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, et al. Video language planning. In *International Conference on Learning Representations*, 2024.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, et al. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, 2024.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 2025.

Nick Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, et al. Imagen video: High definition video generation with diffusion models, 2022. arXiv:2210.02303.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, et al. DreamGen: Unlocking generalization in robot learning through video world models. In *Proceedings of The 9th Conference on Robot Learning*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. OpenVLA: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning*, 2025.

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to act from actionless videos through dense correspondences. In *International Conference on Learning Representations*, 2024.

Dan Kondratyuk, Lijun Yu, Xiuye Gu, Jose Lezama, Jonathan Huang, Grant Schindler, Rachel Hornung, Vighnesh Birodkar, Jimmy Yan, Ming-Chang Chiu, et al. VideoPoet: A large language model for zero-shot video generation. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Sizhe Lester Li, Evan Kim, Xingjian Bai, Tong Zhao, Tao Pang, Max Simchowitz, and Vincent Sitzmann. Turning video models into generalist robot policies, 2026. arXiv:2605.27817.

Junbang Liang, Ruoshi Liu, Ege Ozguroglu, Sruthi Sudhakar, Achal Dave, Pavel Tokmakov, Shuran Song, and Carl Vondrick. Dreamitate: Real-world visuomotor policy learning via video generation. In *Proceedings of The 8th Conference on Robot Learning*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Reginald McLean, Evangelos Chatzaroulas, Luc McCutcheon, Frank Röder, Tianhe Yu, Zhanpeng He, K.R. Zentner, Ryan Julian, J Terry, Isaac Woungang, et al. Meta-World+: An improved, standardized, RL benchmark. In *Advances in Neural Information Processing Systems*, 2025.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and RT-X models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, et al. Demonstrating GPU Parallelized Robot Simulation and Rendering for Generalizable Embodied AI with ManiSkill3. In *Proceedings of Robotics: Science and Systems*, 2025.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models, 2025. arXiv:2503.20314.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. SigLIP 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Ruixiang Wang, Qingming Liu, Yueci Deng, Guiliang Liu, Zhen Liu, and Kui Jia. EVA: Aligning video world models with executable robot actions via inverse dynamics rewards, 2026. arXiv:2603.17808.

Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *International Conference on Learning Representations*, 2024.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies, 2026. arXiv:2602.15922.

Pengfei Zhou, Shengcong Chen, Di Chen, Jiaxu Wang, Rongjun Jin, Bingwen Zhu, Yike Pan, Songen Gu, Kuanning Wang, Shufeng Nan, et al. τ_0-WM: A unified video-action world model for robotic manipulation, 2026. arXiv:2606.01027.

Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. RoboDreamer: Learning compositional world models for robot imagination. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified World Models: Coupling Video and Action Diffusion for Pretraining on Large Robotic Datasets. In *Proceedings of Robotics: Science and Systems*, 2025.

APPENDIX

We evaluate GICAT on two simulated robot manipulation benchmarks, Meta-World and ManiSkill. This appendix provides the implementation and experimental details.

A EXPERIMENTAL TRAINING DETAILS

A.1 META-WORLD

A.1.1 TASKS AND TRAINING DATASETS

We conduct simulation experiments in Meta-World, a Sawyer-manipulation benchmark with 50 robotic control tasks. The benchmark covers reaching, pushing, object transport, drawer and door manipulation, button pressing, faucet manipulation, window manipulation, and assembly-style insertion. This task diversity allows us to test whether a single video-goal-conditioned inverse dynamics model (IDM) can translate visual plans into executable action chunks across different objects, goals, and contact patterns.

Observation Space. Each collected episode stores a fixed third-person RGB stream and a compact robot state. We render the RGB observation from the Meta-World corner camera at 256×256 resolution. In our setting, VG-IDM uses the current RGB frame and a future RGB goal frame as the visual condition for action generation, together with the current robot state. We also derive robot-arm-only observations from the same frames using SAM3 segmentation. These arm-only observations reduce object and background clutter, so retrieval and action translation focus primarily on the robot geometry and motion implied by the visual transition.

Action Space. Meta-World uses a shared four-dimensional continuous action space across tasks. The first three dimensions command end-effector displacement in Cartesian space. The fourth dimension controls the gripper. VG-IDM directly predicts multi-step action chunks. For Meta-World, each IDM target is an eight-step action chunk in this four-dimensional action space.

Data Collection. We collect demonstrations by rolling out the scripted expert policies provided by Meta-World from randomized initial states. For each successful rollout, we render the fixed corner-camera view and record the 39-dimensional robot state and the four-dimensional action at every timestep. After collecting the raw trajectories, we apply SAM3-based robot-arm segmentation and precompute frozen LILA visual features for the masked frames. During IDM training, a sample is formed by choosing a timestep t, using (I_t, s_t) as the current observation and state, using I_{t+8} as the visual goal, and supervising the model with the action chunk $\{a_t, \ldots, a_{t+7}\}$.

Table 2: Source demonstrations used for VG-IDM training on Meta-World.

Task	Episodes	Frames
reach-v3	950	39,908
button-press-topdown-v3	100	6,519
pick-place-v3	100	5,251
peg-insert-side-v3	100	9,585
door-open-v3	100	8,069
drawer-close-v3	100	7,857
faucet-open-v3	100	5,912
push-back-v3	100	8,500
window-close-v3	100	8,044
sweep-into-v3	100	5,441
Total	1850	105,086

For the main Meta-World IDM, GICAT is trained on the ten source tasks summarized in Table 2. These tasks provide 1,850 successful training episodes and 105,086 training frames. We additionally

use 230 held-out successful episodes for offline validation and 100 successful episodes as the default retrieval context pool during IDM training. At closed-loop evaluation time, the trained IDM is reused without task-specific parameter updates on the ten Meta-World tasks reported in the main text. Five evaluation tasks overlap the IDM training set, including reach-v3, button-press-topdown-v3, pick-place-v3, drawer-close-v3, and faucet-open-v3. The remaining five evaluation tasks, including door-close-v3, faucet-close-v3, shelf-place-v3, window-open-v3, and handle-press-v3, are held out from IDM training.

A.1.2 VG-IDM TRAINING

Model Architecture. GICAT learns a retrieval-conditioned inverse dynamics operator that maps a visual transition to a short action sequence. For each training sample, the model receives the current observation I_t, a future goal observation I_{t+h}, the current robot state $s_t \in \mathbb{R}^{d_s}$, a retrieved context set $C_t = \{(I_i, I_{i+h}, s_i, A_{i:i+H-1})\}_{i=1}^{N}$, and a noisy action chunk $A_\tau \in \mathbb{R}^{H \times d_a}$. It predicts the flow matching velocity field

$$\hat{u}_\theta = v_\theta(A_\tau, \tau, I_t, I_{t+h}, s_t, C_t), \tag{10}$$

which is integrated at inference time to produce the executable action chunk $A_{t:t+H-1}$. The implementation consists of the following components:

- **Visual feature projection:** stored 768-dimensional LILA features are used as visual inputs and projected to the Transformer hidden dimension. The Meta-World features are computed from SAM3 robot-arm masks.
- **State encoder:** a three-layer MLP maps the proprioceptive state vector to the shared hidden dimension.
- **Action and time encoders:** each action vector is projected to an action token, and a sinusoidal flow matching time embedding is added to every action token in the chunk.
- **Context encoder:** each retrieved snippet is compressed into a single context token by fusing its current-image feature, goal-image feature, state feature, and temporally pooled action-chunk feature.
- **Action Transformer:** a pre-norm Transformer processes current, goal, state, context, and action tokens jointly. Rotary position embeddings encode token order, and learned type embeddings distinguish the five token types.
- **Action head:** an MLP decodes the hidden state of each action token into a velocity vector in the benchmark-specific action space.

Input Encoding and Retrieval. Let $\phi(I)$ denote the precomputed SAM3 and LILA feature for an observation. Because the reported Meta-World IDM uses precomputed visual features, training does not run the visual backbone online. Instead, the stored feature is mean-pooled when needed and projected to the hidden dimension d. The current image, goal image, and robot state are encoded as

$$z_t = W_{\text{vis}}\phi(I_t), \qquad z_g = W_{\text{vis}}\phi(I_{t+h}), \qquad z_s = \psi_s(s_t), \tag{11}$$

where W_{vis} is the learned visual projection from the stored LILA feature space to the Transformer hidden dimension, and ψ_s is the learned state encoder. These three tokens are fused into a retrieval query,

$$q_t = \eta_q([z_t; z_g; z_s]). \tag{12}$$

At the start of every epoch, the retrieval memory is rebuilt by encoding all snippets in the context dataset with the current query encoder. For each training query, GICAT retrieves the top-N nearest snippets by cosine similarity. We use query-based top-N retrieval in the reported runs and mask out the exact same trajectory-time pair when it appears in the retrieval memory.

For a retrieved snippet $(I_i, I_{i+h}, s_i, A_{i:i+H-1})$, the context token is

$$z_i^c = \eta_c\left([W_{\text{vis}}\phi(I_i); W_{\text{vis}}\phi(I_{i+h}); \psi_s(s_i); \text{pool}(\psi_a(A_{i:i+H-1}))]\right), \tag{13}$$

where ψ_a is the learned action encoder and the pooling operation averages the encoded action tokens over the chunk dimension. Each retrieved example is therefore represented by one token that carries a paired visual transition, state, and expert action pattern.

Algorithm 1 VG-IDM forward pass.

Require: current image I_t, goal image I_{t+h}, robot state s_t, retrieval memory $\mathcal{M}$, noisy action chunk A_τ, flow matching time τ
Ensure: predicted velocity chunk $\hat{u}_\theta$

$z_t \leftarrow W_{\text{vis}}\phi(I_t)$, $z_g \leftarrow W_{\text{vis}}\phi(I_{t+h})$, $z_s \leftarrow \psi_s(s_t)$
$q_t \leftarrow \eta_q([z_t; z_g; z_s])$
$C_t \leftarrow \text{TopN}_{(I_i, I_{i+h}, s_i, A_i)\in\mathcal{M}} \cos(q_t, q_i)$
for each retrieved snippet $(I_i, I_{i+h}, s_i, A_{i:i+H-1}) \in C_t$ **do**
    $z_i^c \leftarrow \eta_c([W_{\text{vis}}\phi(I_i); W_{\text{vis}}\phi(I_{i+h}); \psi_s(s_i); \text{pool}(\psi_a(A_{i:i+H-1}))])$
end for
for $j = 1, \ldots, H$ **do**
    $z^\tau_{a,j} \leftarrow \psi_a(A_{\tau,j}) + \psi_\tau(\tau)$
end for
$X \leftarrow [z_t, z_g, z_s, z_1^c, \ldots, z_N^c, z^\tau_{a,1}, \ldots, z^\tau_{a,H}]$
add learned type embeddings to X and assign rotary positions
build the action-causal mask $\mathcal{B}$ with $3 + N$ memory tokens and H action tokens
$Y \leftarrow \text{Transformer}(X; \mathcal{B})$
$\hat{u}_\theta \leftarrow \text{ActionHead}(Y_{3+N+1:3+N+H})$
return $\hat{u}_\theta$

Action-Chunk Transformer. Given a flow matching time τ, the noisy action token at chunk position j is

$$z^\tau_{a,j} = \psi_a(A_{\tau,j}) + \psi_\tau(\tau), \qquad j = 1, \ldots, H, \tag{14}$$

where ψ_τ is a sinusoidal time embedding followed by an MLP. The full Transformer input is the concatenated token sequence

$$X = [z_t, z_g, z_s, z_1^c, \ldots, z_N^c, z^\tau_{a,1}, \ldots, z^\tau_{a,H}]. \tag{15}$$

The first $3 + N$ tokens are treated as memory tokens. Memory tokens attend only to other memory tokens. Action token j attends to all memory tokens and to action tokens $1, \ldots, j$, but not to future action tokens. This mask lets the model condition every action prediction on the full visual-state-context query while preserving an autoregressive ordering within the predicted action chunk.

The complete VG-IDM forward pass, including query encoding, context retrieval, context-token construction, and action decoding, is summarized in Algorithm 1.

Training Objective. For each ground-truth expert action chunk $A_1 = A_{t:t+H-1}$, we sample Gaussian noise $A_0 \sim \mathcal{N}(0, I)$ and a scalar time $\tau \sim \mathcal{U}(0, 1)$. The straight-line interpolation and target velocity are

$$A_\tau = (1 - \tau)A_0 + \tau A_1, \qquad u_\tau = A_1 - A_0. \tag{16}$$

The primary loss is the mean-squared error between the predicted and target velocity fields:

$$\mathcal{L}_{\text{FM}} = \|v_\theta(A_\tau, \tau, I_t, I_{t+h}, s_t, C_t) - u_\tau\|_2^2. \tag{17}$$

We additionally penalize high-frequency variation across adjacent predicted velocity tokens:

$$\mathcal{L}_{\text{cont}} = \frac{1}{H-1} \sum_{j=1}^{H-1} \|\hat{u}_{\theta,j+1} - \hat{u}_{\theta,j}\|_2^2. \tag{18}$$

The final training objective is

$$\mathcal{L} = \mathcal{L}_{\text{FM}} + \lambda_{\text{cont}}\mathcal{L}_{\text{cont}}, \tag{19}$$

with $\lambda_{\text{cont}} = 0.01$ in all reported runs.

Training Protocol and Hyperparameters. Training samples are constructed by choosing a valid timestep t from a successful Meta-World demonstration, using I_{t+8} as the goal frame, and supervising the model with the action chunk $\{a_t, \ldots, a_{t+7}\}$. We train with diverse-task batches, AdamW, bfloat16 autocast on CUDA, and offline validation on held-out successful demonstrations. The main architecture and training hyperparameters are summarized in Table 3.

For closed-loop Meta-World evaluation, the trained IDM weights are fixed. Changing from the seen-task context pool to the all-task context pool changes only the retrieval memory supplied at inference time; it does not update the IDM parameters.

Table 3: VG-IDM architecture and training hyperparameters.

Parameter	Value
Training demonstrations	1,850 episodes, 105,086 action steps
Validation demonstrations	230 episodes, 15,138 action steps
Training retrieval memory	100 context episodes
Observation source	corner camera, 256×256, SAM3 robot-arm mask
Visual representation	precomputed 768-D LILA features
State / action dimensions	39 / 4
Action chunk length H	8
Goal offset h	8
Retrieved snippets N	4
Transformer width	512
Transformer depth / heads	4 layers / 8 heads
MLP ratio / dropout	4 / 0.1
Position and type encoding	RoPE and learned type embeddings
Batching	batch size 64, at least 5 tasks per batch
Optimizer	AdamW, learning rate 1×10^{-4}, weight decay 0.01
Training length	500 epochs
Gradient clipping	1.0
flow matching solver	Euler sampling, 8 integration steps
Continuity weight	0.01
Seed	42

A.2 MANISKILL

A.2.1 TASKS AND TRAINING DATASETS

We conduct simulation experiments in ManiSkill, a Franka Emika Panda manipulation benchmark with higher-dimensional control and more complex contact dynamics than Meta-World. For VG-IDM training, we use four representative tabletop tasks: PickCube-v1, LiftPegUpright-v1, PushCube-v1, and StackCube-v1. In the closed-loop evaluation suite, PokeCube-v1 and PullCube-v1 are held out from IDM training and used as unseen tasks. This split follows the same purpose as the Meta-World split: it tests execution stability on the training task distribution and generalization to new manipulation tasks under the same video-planner setting.

Observation Space. Each collected episode stores a fixed third-person RGB stream and a compact robot state. We render the RGB observation from the left-rear-close camera at 512×512 resolution. As in Meta-World, VG-IDM uses the current RGB frame, a future RGB goal frame, and the current robot state as the query condition. For ManiSkill, we also use SAM3 to obtain robot-centric masked observations from the RGB frames, where the robot pixels are retained and the background is suppressed. LILA features are then precomputed from these SAM3-masked observations.

Action Space. ManiSkill uses a seven-dimensional continuous action space in our experiments. The action follows the `pd_ee_delta_pose` control mode: the first three dimensions control end-effector translation, the next three dimensions control end-effector rotation, and the final dimension controls the gripper. For ManiSkill, each IDM target is an eight-step action chunk in this seven-dimensional action space.

Data Collection. We build the ManiSkill IDM dataset by replaying official demonstration trajectories with the fixed camera and control configuration used at evaluation time. During replay, we record RGB observations, robot states, seven-dimensional actions, and success labels. We then apply SAM3 to the RGB observations to obtain robot-centric masked frames, which are subsequently encoded using LILA. As in Meta-World, an IDM training sample is formed by choosing a timestep t, using (I_t, s_t) as the current observation and state, using I_{t+8} as the visual goal, and supervising the model with the action chunk $\{a_t, \ldots, a_{t+7}\}$.

The main ManiSkill IDM is trained on the four source tasks summarized in Table 4. These tasks provide 900 successful training episodes and 36,634 training frames. We use 160 held-out suc-

Table 4: Source demonstrations used for VG-IDM training on ManiSkill.

Task	Episodes	Frames
PickCube-v1	250	12,448
LiftPegUpright-v1	200	8,311
PushCube-v1	200	5,999
StackCube-v1	250	9,876
Total	900	36,634

Table 5: VG-IDM architecture and training hyperparameters on ManiSkill.

Parameter	Value
Training demonstrations	900 episodes, 36,634 action steps
Validation demonstrations	160 episodes, 6,391 action steps
Retrieval memory	32 trajectories (8 per task)
Observation source	left-rear-close camera, 512×512, SAM3 robot mask
Visual representation	precomputed 768-D LILA features
State / action dimensions	54 / 7
Action chunk length H	8
Goal offset h	8
Retrieved snippets N	4
Transformer width	512
Transformer depth / heads	4 layers / 8 heads
MLP ratio / dropout	4 / 0.1
Position and type encoding	RoPE and learned type embeddings
Batching	batch size 64, at least 4 tasks per batch
Optimizer	AdamW, learning rate 1×10^{-4}, weight decay 0.01
Training length	500 epochs
Gradient clipping	1.0
flow matching solver	Euler sampling, 8 integration steps
Continuity weight	0.01
Seed	42

cessful episodes for offline validation, with 40 validation episodes for each training task. During closed-loop evaluation, the trained IDM is reused without task-specific parameter updates on the six-task ManiSkill suite reported in the main text. PickCube-v1, LiftPegUpright-v1, PushCube-v1, and StackCube-v1 are seen IDM tasks, while PokeCube-v1 and PullCube-v1 are held out from IDM training.

A.2.2 VG-IDM TRAINING

The ManiSkill IDM uses the same retrieval-conditioned flow matching Transformer architecture described in Section A.1.2. We keep the same tokenization scheme, context encoder, query-based top-N retrieval, action-causal Transformer mask, flow matching objective, continuity regularizer, and Euler sampling procedure. The benchmark-specific differences are the state and action dimensions, camera configuration, data split, and diverse-task batching constraint.

Training Protocol and Hyperparameters. Training samples are constructed with the same goal offset and action horizon as Meta-World: I_{t+8} is used as the visual goal and $\{a_t, \ldots, a_{t+7}\}$ is used as the supervised action chunk. In the main ManiSkill configuration, we use a fixed Context32 retrieval memory during training, consisting of 32 demonstration trajectories sampled from the four seen training tasks, with eight trajectories per task. The main architecture and training hyperparameters are summarized in Table 5.

B Experimental Evaluation Details

B.1 Video Model

We use Wan2.2-TI2V-5B as the video planner in all closed-loop experiments. Wan2.2-TI2V-5B is a text-and-image-conditioned video generation model: it takes an initial image and a language prompt as input, and predicts a short future video. This interface matches our decoupled control setting because the video model specifies the desired visual evolution of the scene without directly predicting robot actions. We adapt the pretrained model with LoRA rather than full-model fine-tuning. The base video model is kept fixed, and low-rank adapters are trained in the DiT modules with rank 32 on the `q`, `k`, `v`, `o`, `ffn.0`, and `ffn.2` layers.

For both benchmarks, the video training data are stored as 49-frame H.264 MP4 windows paired with task prompts. Windows are sampled from successful robot trajectories at 15 FPS with stride 8, using at most two windows per episode. We train the LoRA adapters with learning rate 1×10^{-4}, dataset repeat 2, two epochs, batch size 1 per process, and gradient checkpointing. During closed-loop evaluation, the planner receives the current simulator RGB frame and the task prompt, generates a 9-frame future video, and uses the final generated frame as the visual goal for the downstream IDM. The same benchmark-specific video planner is shared by VG-IDM and all video-to-action baselines, so the closed-loop comparison isolates the action-translation module as much as possible. In all reported rollouts, the LoRA scale is 1.0. The negative prompt is "overexposed, static, blurry, low quality, malformed robot arm, broken gripper, distorted object, inconsistent geometry, duplicate arm, text, watermark".

Meta-World video planner. For Meta-World, we fine-tune a Wan2.2-TI2V-5B LoRA planner on the ten-task video suite used in the main evaluation. The raw simulator frames are rendered from the Meta-World corner camera at 256×256 resolution, and the Wan training pipeline resizes and pads each video window to 480×832 while preserving aspect ratio. The training split contains 500 successful episodes and 913 video windows, with up to 50 training episodes per task. At rollout time, the planner is queried with the current corner-camera frame and the task prompt, generates 9 frames at 480×832, and the last generated frame is converted back into the IDM visual-processing pipeline. The reported Meta-World rollouts use 8 Wan inference steps.

ManiSkill video planner. For ManiSkill, we train a separate Wan2.2-TI2V-5B LoRA planner on a six-task video suite containing PickCube-v1, LiftPegUpright-v1, PushCube-v1, StackCube-v1, PullCube-v1, and PokeCube-v1. The videos are rendered from the left-rear-close camera at native 512×512 resolution and are used by Wan at the same spatial resolution. The training split contains 600 successful episodes and 1,076 video windows, with up to 100 training episodes per task. At rollout time, the planner receives the current 512×512 RGB observation and the task prompt, generates a 9-frame future video, and passes the last generated frame to the IDM after the same goal-image masking and feature-extraction pipeline used in the rest of the ManiSkill evaluation. The reported ManiSkill rollouts use 50 Wan inference steps.

B.2 Closed-Loop Rollout

Closed-loop evaluation uses the same receding-horizon pipeline for both benchmarks. Each rollout starts from a fresh simulator reset with the evaluation task seed. At the beginning of every planning round, the system renders the current full RGB observation from the simulator and reads the current robot state. The video planner then generates a 9-frame future video conditioned on the current RGB frame and the task prompt. We use the last generated frame as the visual goal for action translation. The current frame and the generated goal frame are then passed through the benchmark-specific IDM preprocessing pipeline, which applies the robot-mask and visual-feature extraction steps used by the corresponding IDM. VG-IDM encodes this current-goal-state query, retrieves the nearest in-context demonstrations from the fixed retrieval memory, and samples an action chunk with the flow matching IDM.

We execute only the first L actions of each sampled action chunk, with $L = 4$ in our experiments. After executing the first L actions, the next video plan is generated from the true simulator observation rather than from the previously generated video. This prevents long-horizon execution from

Algorithm 2 Closed-loop rollout with video planning and action translation.

Require: simulator environment $\mathcal{E}$, task prompt p, video planner $\mathcal{V}_\psi$, action translator π_θ, retrieval memory $\mathcal{M}$, maximum planning rounds R, maximum environment steps T, execution horizon L
Ensure: rollout success flag

reset $\mathcal{E}$ with the evaluation seed
$n \leftarrow 0$, $success \leftarrow$ **false**
for $r = 0, \ldots, R-1$ **do**
  if $success$ or $n \geq T$ **then**
    break
  end if
  render current RGB observation I_n and read robot state s_n
  $\hat{Q}_n \leftarrow \mathcal{V}_\psi(I_n, p)$ ▷ generate a 9-frame video plan
  $I_g \leftarrow \mathrm{LastFrame}(\hat{Q}_n)$
  $(\tilde{I}_n, \tilde{I}_g) \leftarrow \mathrm{PreprocessForIDM}(I_n, I_g)$
  $q_n \leftarrow \mathrm{EncodeQuery}(\tilde{I}_n, \tilde{I}_g, s_n)$
  $C_n \leftarrow \mathrm{RetrieveTopN}(\mathcal{M}, q_n)$
  $A_{n:n+H-1} \leftarrow \pi_\theta(\tilde{I}_n, \tilde{I}_g, s_n, C_n)$
  $t_0 \leftarrow n$
  for $j = 0, \ldots, \min(L, H) - 1$ **do**
    if $success$ or $n \geq T$ **then**
      break
    end if
    $a \leftarrow \mathrm{Clip}(A_{t_0+j}, \mathcal{E}.\,\mathrm{action_space})$
    step $\mathcal{E}$ with action a
    $n \leftarrow n + 1$
    update $success$ from the simulator info dictionary
    if $success$ or the environment terminates or truncates **then**
      break
    end if
  end for
  if $success$ or the environment terminates or truncates **then**
    break
  end if
end for
return $success$

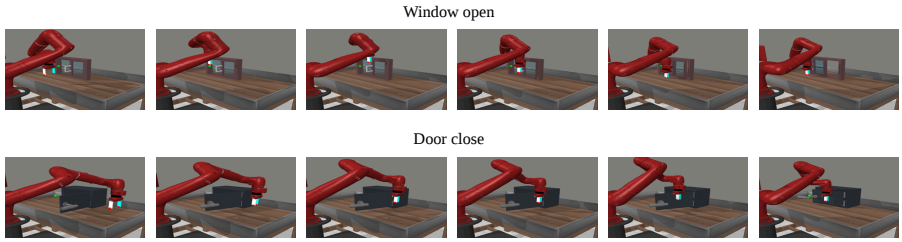


Figure 6: Representative successful rollouts of GICAT on two unseen Meta-World tasks. These examples show the robot opening a window and closing a door, demonstrating that VG-IDM can follow the plans generated by the video generation model and translate visual goals into executable actions, enabling successful completion of previously unseen manipulation tasks.

depending entirely on an open-loop visual prediction. Actions are clipped to the simulator action bounds before execution. A rollout terminates when the task succeeds, the environment terminates or truncates, the maximum number of environment steps is reached, or the maximum number of planning rounds is reached. The same outer loop is used for the baseline action models, with the GICAT action-translation call replaced by the corresponding baseline IDM inference. The complete closed-loop rollout procedure is summarized in Algorithm 2.

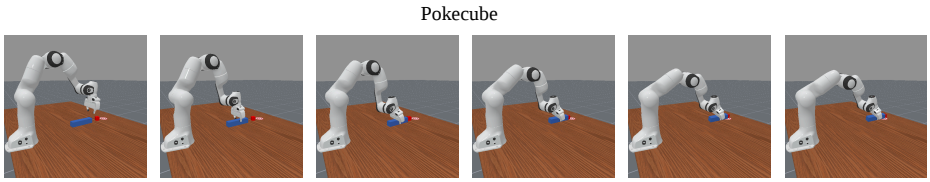
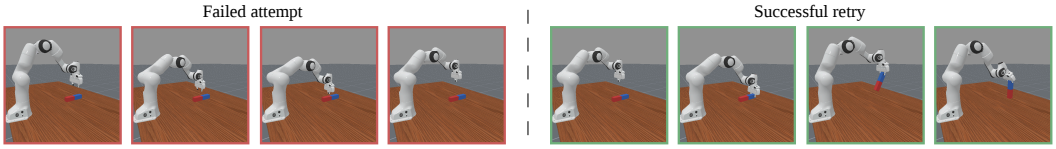


Figure 7: Representative successful rollout of GICAT on an unseen ManiSkill task. This example shows the robot pushing the cube toward the target, demonstrating that VG-IDM can follow the plan generated by the video generation model and translate the visual goal into executable actions, enabling successful completion of a previously unseen manipulation task.



(a) **Meta-World recovery:** After an unsuccessful initial execution, GICAT replans a new visual goal from the updated observation and successfully completes the task.



(b) **ManiSkill recovery:** The first grasp attempt misses the peg. GICAT then replans from the updated scene, lowers the gripper again, and successfully grasps the peg on the second attempt.

Figure 8: Closed-loop recovery behaviors of GICAT in Meta-World and ManiSkill. After an unsuccessful initial attempt, GICAT replans from the updated observation and successfully completes the task in a subsequent attempt.

B.3 Qualitative Analysis of Closed-Loop Recovery

Figs. 6 and 7 show representative successful rollouts of GICAT on tasks held out from VG-IDM training. These examples illustrate that GICAT can generalize beyond the manipulation tasks seen during action-model training. We attribute this capability to the complementary roles of the two stages: the video planner predicts task-specific visual goals that describe the desired scene evolution, while VG-IDM translates the corresponding robot-centric visual changes into executable control actions. By focusing on the motion and configuration changes of the robot rather than task-irrelevant appearance variations, VG-IDM can reuse the learned visual-to-action correspondence across different tasks and scenes.

Fig. 8 illustrates representative closed-loop recovery behaviors of GICAT in Meta-World and ManiSkill. In both examples, the initial execution does not successfully complete the intended manipulation, leaving the robot in an intermediate state that differs from the nominal successful trajectory. Rather than continuing to follow the previous plan, GICAT replans from the updated environment observation and generates a new action sequence adapted to the resulting scene. In Meta-World, the replanned execution corrects the unsuccessful initial attempt and completes the task. In ManiSkill, after the first grasp misses the peg, the model replans from the new scene, lowers the gripper again, and successfully grasps the object on the second attempt. These examples qualitatively demonstrate that GICAT can adapt its action generation to previously encountered intermediate states, rather than relying on a fixed open-loop execution pattern.