VideoAgent: Self-Improving Video Generation for Embodied Planning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Video generation has been effective in generating visual plans for controlling robotic systems. Specifically, given an image observation and a language instruction, previous work has generated video plans which are then converted to robot controls to be executed. However, a major bottleneck in leveraging video generation for control lies in the quality of the generated videos, which often suffer from hallucination (e.g., unrealistic physics), resulting in low task success when control actions are extracted from the generated videos. In this work, we propose VideoAgent to ground video generation in the physical world by integrating external feedback. VideoAgent trains a video diffusion model to perform video refinement through a novel objective which we call *self-conditioning consistency*. During inference, VideoAgent samples and refines generated video plans under the guidance of a vision-language model (VLM) as reward, enabling inference-time compute to be turned into better generated video plans. As refined video plans are executed, VideoAgent collects additional data from the environment to further improve video plan generation. Experiments in simulated robotic manipulation from MetaWorld and iTHOR show that VideoAgent drastically reduces hallucination, thereby boosting success rate of downstream manipulation tasks. We further illustrate that VideoAgent can effectively refine real-robot videos, providing an early indicator that robots can be an effective tool in grounding video generation in the physical world.[1]

1 Introduction

Large text-to-video (T2V) models pretrained on internet-scale data enable generation of creative video content (Ho et al., 2022; Hong et al., 2022; Singer et al., 2022), games (Bruce et al., 2024), animations (Wang et al., 2019), and movies (Zhu et al., 2023). Recent work demonstrates their potential as real-world simulators (Yang et al., 2023b; Brooks et al., 2024) and as policies with unified observation and action space (Du et al., 2024; Ko et al., 2023; Du et al., 2023). These advances can lead to internet-scale knowledge transfer and progress toward generalist agents—a single policy for controlling multiple robots across various morphologies, environments and tasks. Nevertheless, T2V models have only had limited success in downstream applications in reality. For instance, in video generation as policy (Du et al., 2024; Ko et al., 2023), when an observation image and a language instruction are given to a video generation model, generated videos often hallucinate (e.g., objects randomly appear or disappear) or violate physical laws (e.g., a robot hand going through an object) (Yang et al., 2023b; Brooks et al., 2024). Such issues have led to low task success rate when generated videos are converted to control actions through inverse dynamics models, goal conditioned policies, or other action extraction mechanisms (Wen et al., 2023; Yang et al., 2024; Ajay et al., 2024). While scaling up dataset and model size can be effective in reducing hallucination in large

[1]See code and examples in supplemental material.

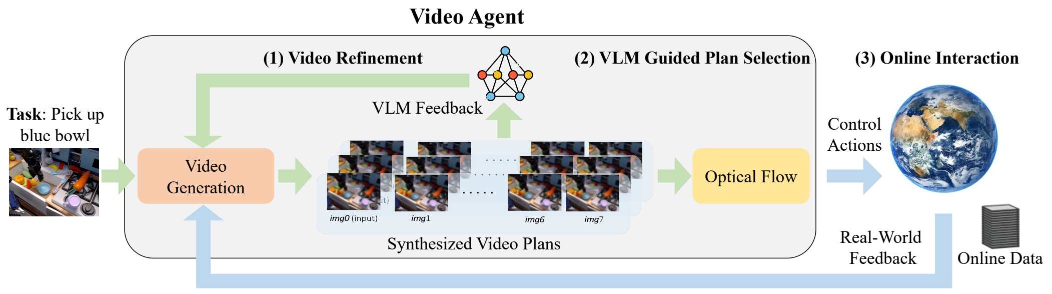


Figure 1: The VideoAgent Framework. VideoAgent generates a video plan conditioned on an initial image and task description, similar to (Du et al., 2023). It then (1) iteratively refines this plan using VLM feedback, (2) the VLM selects the best refined video, converting to control actions via optical flow, and (3) executes these actions in an environment, collects real-world feedback and additional online data to further improve the generation.

language models (LLMs) (Hoffmann et al., 2022), it is more difficult in video generation as language labels for videos are labor intensive to curate and we have not yet converged to an architecture that is more favourable to scaling (Yang et al., 2024). Scaling aside, two major directions to improve generation in LLMs have been to incorporate external feedback (Ouyang et al., 2022b) and to scale search with increased inference time compute (Snell et al., 2024; Wu et al., 2024). It is therefore natural to wonder what kind of feedback and inference-time compute can be leveraged to improve T2V generations.

To answer this question, we explore two types of feedback that are natural to obtain for video generation models, namely AI feedback from a vision-language model (VLM) and real-world execution feedback when generated videos are converted to motor controls. To utilize these feedback for self-improvement, we propose VideoAgent. Different from video generation as policy, which directly turns a generated video into control actions (Du et al., 2023; Ko et al., 2023), VideoAgent is trained to refine a generated video plan iteratively using feedback from a pretrained VLM. During inference, VideoAgent queries the VLM to select the best refined video plan, allowing inference-time compute to be turned into better generated video plans, followed by execution of the plan in the environment. During online execution, VideoAgent observes whether the task was successfully completed, and further improves the video generation model based on the execution feedback from the environment and additional data collected from the environment. The improvement to the generated video plan comes in three folds: First, we propose *self-conditioning consistency* for video diffusion model inspired by consistency models (Song et al., 2023; Heek et al., 2024), which enables low-quality samples from a video diffusion model to be further refined into high-quality samples. Second, VLM feedback combined with more inference-time compute leads to better video plans. Lastly, when online access to the environment is available, VideoAgent executes the current video plan and collects additional successful trajectories to further finetune the video generation model. A visual illustration of VideoAgent is shown in Figure 1.

We first evaluate VideoAgent in two simulated robotic manipulation environments, Meta-World (Yu et al., 2020) and iTHOR (Kolve et al., 2017), and show that VideoAgent improves task success across all environments and tasks evaluated. We additionally provide ablation studies on the effect of different components in VideoAgent, including different types of feedback from the VLM and the amount of inference-time compute spent.Lastly, we illustrate that VideoAgent can iteratively improve real-robot videos, providing early signal that robotics can be an important mean to ground video generation models in the real world.

2 Background

In this section, we provide the background on video generation as policy in a decision making process (Du et al., 2023). We also introduce consistency models (Song et al., 2023; Heek et al., 2024; Daras et al., 2024), which VideoAgent builds upon for self-refinement.

2.1 Video as Policy in Sequential Decision Making

We consider a predictive decision process similar to (Du et al., 2024): $\mathcal{P} := \langle \mathcal{X}, \mathcal{G}, \mathcal{A}, H, \mathcal{E}, \mathcal{R} \rangle$, where $\mathcal{X}$ denotes an image-based observation space, $\mathcal{G}$ denotes textual task description space, $\mathcal{A}$ denotes a low-level motor control action space, and $H \in \mathbb{R}$ denotes the horizon length. We denote

$\pi(\cdot|x_0, g) : \mathcal{X} \times \mathcal{G} \mapsto \Delta(\mathcal{X}^H)$[2] as the language conditioned video generation policy, which models the probability distribution over H-step image sequences $\mathbf{x} = [x_0, ..., x_H]$ determined by the first frame x_0 and the task description g. Intuitively, $\mathbf{x} \sim \pi(\cdot|x_0, g)$ correspond to possible visual paths for completing a task g. Given a sampled video plan $\mathbf{x}$, one can use a learned mapping $\rho(\cdot|\mathbf{x}) : \mathcal{X}^H \mapsto \Delta(\mathcal{A}^H)$ to extract motor controls from generated videos through a goal-conditioned policy (Du et al., 2023), diffusion policy (Black et al., 2023), or dense correspondence (Ko et al., 2023). Once a sequence of motor controls $\mathbf{a} \in \mathcal{A}^H$ are extracted from the video, they are sequentially executed in the environment $\mathcal{E}$, after which a final reward $\mathcal{R} : \mathcal{A}^H \mapsto \{0, 1\}$ is emitted representing whether the task was successfully completed. For simplicity, we only consider finite horizon, episodic tasks. Given a previously collected dataset of videos labeled with task descriptions $\mathcal{D} = \{(\mathbf{x}, g)\}$, one can leverage behavioral cloning (BC) (Pomerleau, 1988) to learn π by minimizing

$$\mathcal{L}_{\text{BC}}(\pi) = \mathbb{E}_{(\mathbf{x},g)\sim\mathcal{D}}[-\log \pi(\mathbf{x}|x_0, g)]. \tag{1}$$

Equation 1 can be viewed as maximizing the likelihood of the videos in $\mathcal{D}$ conditioned on the initial frame and task description.

2.2 Consistency Models

Diffusion models (Ho et al., 2020; Song et al., 2020b) have emerged as an important technique for generative modeling of high-dimensional data. During training, a diffusion model learns to map noisy data (at various noise levels) back to clean data in a single step. Concretely, let $x^{(0)}$ denote a clean image and $x^{(t)}$ denote the noisy image at noise level t, where $t \in [0, T]$, the training objective for a diffusion model $f_\theta(x^{(t)}, t)$ can be written as

$$\mathcal{L}_{\text{diffusion}}(\theta) = \mathbb{E}_{x^{(0)},\epsilon,t}\left[\|f_\theta(x^{(t)}, t) - x^{(0)}\|^2\right], \tag{2}$$

where $\epsilon \in \mathcal{N}(0, I)$ is the added noise, and $x^{(t)} = \sqrt{\alpha_t}x^{(0)} + \sqrt{1-\alpha_t}\epsilon$ where α_t are time-dependent noise levels. Although diffusion models have achieved high-quality image/video generation, they require hundreds or thousands of denoising steps during inference, which induces tremendous computational cost. To overcome the slow sampling speed of diffusion models, *consistency models* (Song et al., 2023; Song & Dhariwal, 2023) were initially proposed by enforcing a consistency loss across different noise levels, i.e.,

$$\mathcal{L}_{\text{consistency}}(\theta) = \mathbb{E}_{x^{(0)},\epsilon,t_1,t_2}\left[\|f_\theta(x^{(t_1)}, t_1) - \texttt{stopgrad}\big(f_\theta(x^{(t_2)}, t_2)\big)\|^2\right], \tag{3}$$

which encourages the output of the single-step map between different noise levels to be similar. In fact, both the diffusion loss in Equation 2 and the consistency loss in Equation 3 can be understood as exploiting the structure of the denoising procedure which corresponds to an ordinary differential equation (ODE). Specifically, as introduced in Song et al. (2023, 2020a), the backward denoising procedure of a diffusion model can be characterized by an ODE, i.e.,

$$\frac{\mathrm{d}x^{(t)}}{\mathrm{d}t} = -t \cdot s(x^{(t)}, t), \tag{4}$$

with $s(x^{(t)}, t)$ is some score function. During the entire path along $t \in (\epsilon, \infty]$, following this ODE should always map $x^{(t)}$ to $x^{(0)}$. If we parametrize the model $f(x^{(t)}, t)$ as the simulation following the ODE governed by $s(x^{(t)}, t)$, we obtain the diffusion loss (2). Meanwhile, for all $t, t' \in (\epsilon, \infty]$, we have $f(x^{(t)}, t) = f(x^{(t')}, t')$ along the simulation path, which induces the consistency loss (3). Therefore, we can combine the diffusion loss and consistency loss together for model training, i.e.,

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{diffusion}}(\theta) + \lambda \cdot \mathcal{L}_{\text{consistency}}(\theta), \tag{5}$$

where λ denotes consistency regularization hyperparameter across different noise levels.

3 Video Generation as An Agent

In this section, we introduce VideoAgent, a framework for improving video plan generation. In Section 3.1, we develop *self-conditioning consistency* to iteratively refine generated video plans. In Section 3.2, we describe how a diffusion model trained with self-conditioning consistency can leverage inference-time compute to select the best video plans. Finally, in Section 3.3, we illustrate how VideoAgent closes the self-improvement loop by collecting additional online data to further enhance video generation and refinement.

3.1 Refinement through Self-Conditioning Consistency

[2]We use $\Delta(\cdot)$ to denote a probability simplex function

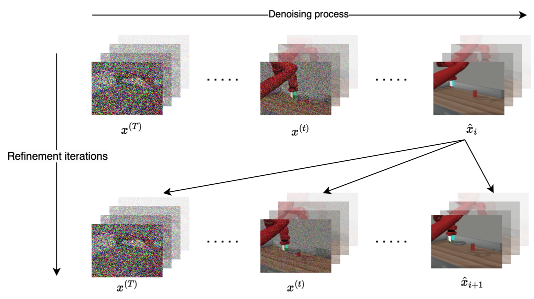


Figure 2: An illustration of Self-Conditioning Consistency. The horizontal direction represents regular denoising process. The two rows represent two refinement iterations. $\hat{\mathbf{x}}_i$ denotes generated video plan at refinement iteration i. We condition the refinement iteration $i+1$ on generated video from the previous iteration $\hat{\mathbf{x}}_i$.

We consider first-frame-and-language conditioned video generation following Du et al. (2023); Ko et al. (2023), which generates a sequence of image frames to complete the task described by the language starting from the initial image. In practice, generated videos often contain hallucinations (Yang et al., 2023b). While such inaccuracies may prevent a video plan from fully completing the task, the generated video may still make meaningful progress towards completing the task. Thus, instead of independently sampling many videos hoping that one may be free from hallucinations, we propose refining previously generated videos iteratively.

Specifically, let $\mathbf{x}^{(0)}$ denote the ground truth video and $\hat{\mathbf{x}}$ a generated video from the original T2V diffusion model. We introduce a *self-conditioning consistency* model, $\hat{f}_\theta(\hat{\mathbf{x}}, \mathbf{x}^{(t)}, t)$, which takes a generated video $\hat{\mathbf{x}}$ and a noisy version of the ground truth $\mathbf{x}^{(t)}$ as inputs to predict the clean video. This formulation enables iterative refinement by conditioning the model on its previous predictions, as illustrated in Figure 2. We denote video samples from the refinement model after the i-th iteration as $\hat{\mathbf{x}}_i$. Self-conditioning is inspired by a reparameterization of the implicit ODE solver for Equation 4 (Song et al., 2020a; Lu et al., 2022; Zhang & Chen, 2022; Chen et al., 2022). For instance, Song et al. (2020a) considered the first-order ODE solver for Equation 4 following:

$$\mathbf{x}^{(t-1)} = \sqrt{\alpha_{t-1}}\mathbf{x}^{(\mathbf{0})} + \sqrt{1 - \alpha_{t-1} - \sigma_t^2} \cdot s(\mathbf{x}^{(t)}, t). \tag{6}$$

In VideoAgent, we adapt Equation 6 by replacing the $\mathbf{x}^{(0)}$ term with $\hat{\mathbf{x}}$, the previously generated video sample, as illustrated in Figure 2. In standard DDIM-based methods (Song et al., 2020a), $\mathbf{x}^{(0)}$ is typically obtained as an intermediate estimate from $\mathbf{x}^{(t)}$ within the *same* iteration. In contrast, our approach reuses $\hat{\mathbf{x}}$ from a *previous* iteration, allowing for a self-conditioning mechanism that improves temporal coherence. By enforcing consistency across iterations, our method enables the denoising process to correct potential failures more effectively.

We learn the ODE solver through self-conditioning consistency by directly predicting the clean video $\hat{\mathbf{x}}_{\mathbf{i+1}}$ using:

$$\begin{aligned}\mathcal{L}_{\text{SC-consistency}}(\theta) = \mathbb{E}_{\hat{\mathbf{x}},\mathbf{x}^{(0)},t}\left[\|\hat{f}_\theta(\hat{\mathbf{x}}, \mathbf{x}^{(t)}, t) - \mathbf{x}^{(0)}\|^2\right] \\ + \mu\mathbb{E}_{\hat{\mathbf{x}}_1,\hat{\mathbf{x}}_2,t}\left[\|\hat{f}_\theta(\hat{\mathbf{x}}_1, \mathbf{x}^{(t)}, t) - \hat{f}_\theta(\hat{\mathbf{x}}_2, \mathbf{x}^{(t)}, t)\|^2\right].\end{aligned} \tag{7}$$

The first term in Equation 7 represents the standard diffusion loss with the additional conditioning on $\hat{\mathbf{x}}$, while the second term regularizes the similarity between different refinement iterations ($\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$) to promote coherence across iterations. This iterative refinement process distinguishes self-conditioning consistency from traditional consistency models. Combined with the standard objective for video diffusion:

$$\mathcal{L}_{\text{video-diffusion}}(\theta) = \mathbb{E}_{\mathbf{x}^{(0)},\epsilon,t}\left[\|f_\theta(\mathbf{x}^{(t)}, t) - \mathbf{x}^{(0)})\|^2\right], \tag{8}$$

the overall objective for training a self-conditioning-consistent video diffusion model thus becomes:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{video-diffusion}}(\theta) + \lambda\mathcal{L}_{\text{SC-consistency}}(\theta). \tag{9}$$

Note that while the video generation model f_θ and the video refinement model $\hat{f}_\theta$ have different input arguments (first frame versus previously generated video), we can share their parameters to train a single unified model for both video generation and refinement tasks. This parameter-sharing approach allows us to leverage the same model architecture for generating initial video plans and iterative refinement. The training process for f_θ and $\hat{f}_\theta$ is detailed in Algorithm 1 in Appendix B.

Feedback Guided Self-Conditioning Consistency. While we can refine videos only from previously generated samples, it may be desirable to condition the refinement process on any additional feedback for the previously generated video that is available (e.g., feedback from humans or vision language models critiquing which part of the generated video is unrealistic). When such feedback is available, we can have the refinement model $\hat{f}$ further take the additional feedback as input, combined with the task description, to guide the refinement process, i.e.,

$$\hat{f}_\theta(\mathbf{x}, \mathbf{x}^{(t)}, t|\text{feedback}), \tag{10}$$

which can be plugged into our framework for learning feedback-guided self-refinement using Equation 9.

3.2 Inference-Time Planning through VLM-Guided Video Generation.

After training the video generation model f_θ and the video refinement model $\hat{f}_\theta$ described in Equation 8 and Equation 7, we can sample from f_θ and iteratively apply $\hat{f}_\theta$ for video refinement. Specifically, let η be the step size for the noise schedule, σ_t be a time dependent noise term, VideoAgent first generates a video plan through

$$\mathbf{x}^{(t-1)} = \mathbf{x}^{(t)} - \eta \cdot \nabla_\theta f_\theta(\mathbf{x}^{(t)}, t) + \sigma_t \cdot \epsilon. \tag{11}$$

The sample $\hat{\mathbf{x}}$ after T denoising steps corresponds to the generated video. Next, we can iteratively apply $\hat{f}_\theta$ to refine the generated video sample

$$\hat{\mathbf{x}}_{i+1} = \hat{f}_\theta(\hat{\mathbf{x}}_i, \mathbf{x}^{(t)}, t), \tag{12}$$

where i denotes the video refinement iteration, with $\hat{\mathbf{x}}_0 = \hat{\mathbf{x}} = \mathbf{x}^{(T)}$. We denote the final video after refinement as $\hat{\mathbf{x}}_{\text{refined}}$. A natural question is when to stop the iterative video refinement process. We use a VLM as a proxy for the environment's reward to assess whether a refined video is likely to lead to successful execution in the environment. Specifically, we denote a VLM as $\hat{\mathcal{R}}$, which takes a refined video $\hat{\mathbf{x}}_i$ and returns a binary value $\{0, 1\}$ to determine whether a video is acceptable based on overall coherence, adherence to physical laws, and task completion (See prompt for VLM in Appendix C). With $\hat{\mathcal{R}}$, the refinement stops when the VLM decides that the refined video is acceptable. Namely, we have

$$\hat{\mathbf{x}}_{\text{refined}} = \hat{\mathbf{x}}_{i^*}, \quad \text{where} \quad i^* = \min \left\{ i : \hat{\mathcal{R}}(\hat{\mathbf{x}}_i) = 1 \right\} \tag{13}$$

Algorithm 2 in Appendix B shows details for video plan generation, refinement, and selection during inference.

3.3 Self-Improvement through Online Finetuning

In addition to video refinement through self-conditioning consistency and inference-time compute as described in Section 3.1 and Section 3.2, we can further characterize the combination of video generation and video refinement as a policy, which can be improved by training on additional data collected from the environment during online interaction. Specifically, the goal is to maximize the expected returns of a policy through trial-and-error interaction with the environment:

$$\mathcal{J}_{\text{online}}(\theta) = \mathbb{E}\left[\mathcal{R}(\mathbf{a}) \mid \pi_\theta, \rho, \mathcal{E}\right], \tag{14}$$

where $\mathcal{R}$ is the true reward function, $\mathcal{E}$ is the interactive environment, and π_θ corresponds to the combination of f_θ and $\hat{f}_\theta$.

A broad array of reinforcement learning methods (Sutton & Barto, 2018) such as policy gradient (Schulman et al., 2017) can be employed to maximize the objective in Equation 14. For simplicity, we consider the setup of first executing the policy in the environment, then filtering for successful trajectories, continuing finetuning the video policy using additional online data, and executing the finetuned policy again to collect more data. Specifically, each online iteration constructs an additional dataset by rolling out the policy π_θ at the current online iteration

$$\mathcal{D}_{\text{new}} = \left\{\hat{\mathbf{x}}_{\text{refined}} \sim \pi_\theta(x_0, g) \mid \mathcal{R}(\rho(\hat{\mathbf{x}}_{\text{refined}})) = 1\right\}, \tag{15}$$

where ρ is the optical flow model that maps the refined video to low-level control actions. See Algorithm 3 in Appendix B for details of online policy finetuning.

4 Experiments

In this section, we evaluate the performance of VideoAgent. First, we measure the end-to-end task success rate of VideoAgent against the baselines in Section 4.1, and study the effect of different components of VideoAgent in Section 4.2. Finally, we show that VideoAgent is effective in improving the quality of real robotic videos in Section 4.3.

Evaluation Setup. We follow the evaluation setup of AVDC (Ko et al., 2023), a method for controlling simulated robots using dense action correspondence extracted from generated videos. We follow AVDC and perform evaluation in three environments: Meta-World (Yu et al., 2020), iTHOR (Kolve et al., 2017), and BridgeData V2 (Walke et al., 2023) (see detailed dataset description in Appendix E). We compare variants of VideoAgent, including VideoAgent with only self-refinement

Table 1: Meta-World Results. Mean success rates of baselines and VideoAgent on 11 simulated robot manipulation environments from Meta-World. VideoAgent consistently outperforms baselines across all tasks.

	door-open	door-close	basketball	shelf-place	btn-press	btn-press-top
AVDC	30.7%	28.0%	21.3%	8.0%	34.7%	17.3%
AVDC-Replan	72.0%	89.3%	37.3%	18.7%	60.0%	24.0%
VideoAgent	40.0%	29.3%	13.3%	9.3%	38.7%	18.7%
VideoAgent-Online (Iter1)	48.0%	40.0%	24.0%	12.0%	42.7%	36.0%
VideoAgent-Online (Iter2)	58.7%	50.7%	28.0%	18.7%	53.3%	41.3%
VideoAgent-Online-Replan	**82.7%**	**97.3%**	**40.0%**	**26.7%**	**73.3%**	**44.0%**

	faucet-close	faucet-open	handle-press	hammer	assembly	**Overall**
AVDC	12.0%	17.3%	41.3%	0.0%	5.3%	19.6%
AVDC-Replan	53.3%	24.0%	81.3%	8.0%	6.7%	43.1%
VideoAgent	46.7%	12.0%	36.0%	0.0%	1.3%	22.3%
VideoAgent-Online (Iter1)	53.3%	28.0%	52.0%	1.3%	5.3%	31.2%
VideoAgent-Online (Iter2)	58.7%	36.0%	64.0%	1.3%	9.3%	38.2%
VideoAgent-Online-Replan	**74.7%**	**46.7%**	**86.7%**	**8.0%**	**10.7%**	**53.7%**

(Section 3.1), VideoAgent-Online (Section 3.3), and VideoAgent-Replan against AVDC and AVDC-Replan (replanning when movement stalls) from Ko et al. (2023). More detailed descriptions of the baselines and the VideoAgent variants are provided in Appendix F.

4.1 End-to-End Task Success

Meta-World. We report the task success rates of baselines and VideoAgent in Table 1. Following Ko et al. (2023), we evaluate performance across three camera poses with 25 seeds per pose. Without online environment access, VideoAgent improves the overall success rate through self-conditioning consistency alone from 19.6% (AVDC) to 22.3%. For certain difficult tasks, e.g., faucet-close, VideoAgent improves performance from 12.0% to 46.7%. With online data collection, VideoAgent-Online further improves success rates with each additional online iteration of rolling out the policy, collecting successful trajectories, and finetuning. VideoAgent-Online can be further combined with replanning, achieving 53.7% overall success, surpassing prior state-of-the-art on this benchmark. Detailed baseline comparisons are provided in Appendix G.2, and qualitative improvements in refined videos are shown in Figure 9 in Appendix K.

Table 2: iThor Success Rates comparing VideoAgent with the AVDC baseline. VideoAgent outperforms AVDC across all four rooms averaged over all three objects in each room.

Room	AVDC	VideoAgent
Kitchen	26.7%	**28.3%**
Living Room	23.3%	**26.7%**
Bedroom	38.3%	**41.7%**
Bathroom	36.7%	**40.0%**
Overall	31.3%	**34.2%**

iTHOR. Next, we evaluate VideoAgent on iThor. Due to the high computational cost of running the iThor simulator(approx. 20 minutes per roll-out), we focus only on evaluating self-conditioning consistency (without online access). We follow the same setup as (Ko et al., 2023), where we measure the average success rate across four rooms each with three objects using 20 seeds. As shown in Table 2, VideoAgent consistently outperforms the AVDC baseline, demonstrating the effectiveness of self-conditioning consistency in producing more plausible video plans for first-person view navigation (i.e., what iThor measures).

4.2 Effect of Different Components in VideoAgent

In this section, we aim to understand the effect of different components of VideoAgent. Specifically, we focus on the effect of (1) different types of feedback given to the refinement model, (2) the number of refinement and online iterations, and (3) the quality of the VLM feedback.

Table 3: Effect of Different Feedback used to train the refinement model. Descriptive feedback from the VLM leads to higher improvement in task success.

	Overall
AVDC	19.6%
VideoAgent	22.3%
VideoAgent-Binary	23.8%
VideoAgent-Suggestive	26.6%

4.2.1 Effect of Different VLM Feedback

In the previous section, we only used VLM during inference to determine when to stop refining a generated video. However, it is natural to wonder if

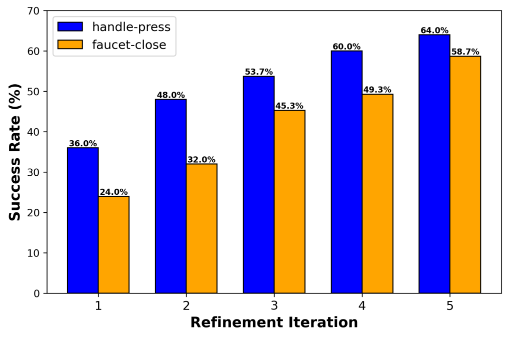


Figure 3: Effect of Refinement Iterations. The accuracy of downstream tasks generally increases as the number of refinement iteration increases.

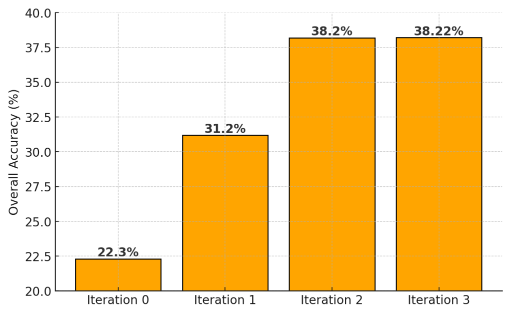


Figure 4: Effect of Online Iterations. The overall task success of VideoAgent increases as the number of online iterations increases.

information-rich feedback from the VLM, such as language descriptions of which part of a generated video to improve, might lead to better refined videos. To answer this question, we propose a few variants of VideoAgent according to the feedback available when training the video refinement model as in Equation 10. Specifically, we use VideoAgent to denote training the video refinement model only conditioned on the original task description. VideoAgent-Binary denotes additionally conditioning on whether a generated video is determined to be successful by the VLM. VideoAgent-Suggestive denotes conditioning additionally on language feedback from the VLM on which part of the video needs improvement and how the video can be improved. We train these three versions of the video refinement model, and report the overall task success from Meta-World in Table 3. We see that VideoAgent-Binary improves upon the base VideoAgent, while training with descriptive feedback in VideoAgent-Suggestive leads to even better performance. This suggests that richer feedback from the VLM can facilitate better training of the video refinement model. Improvement for each individual task can be found in the Appendix I.

4.2.2 Effect of the Number of Iterations.

Next, we want to understand whether more refinement iterations and online finetuning iterations lead to higher task success. We found that while different tasks require a different number of refinement and online iterations to achieve the best performance, VideoAgent does perform better as the number of refinement and online iterations increases, as shown in Figure 3 and Figure 4. During video refinement, specific tasks such as handle-press and faucet-close continue to show improvement even at the fifth refinement iteration. Faucet-close especially benefits from more refinement iterations, bringing success rate from 24.0% to 58.7% after five refinement iterations. The improved task success rates across refinement and online iterations suggests that self-conditioning consistency discussed in Section 3.1 and online interaction discussed in Section 3.3 can indeed effectively reduce hallucination and improve physical plausibility in the generated videos.

4.2.3 Accuracy of VLM feedback

Table 4: VLM Performance measured according to whether a VLM considers a generated video as acceptable using human label as the ground truth.

	Precision	Recall	F1-Score	Accuracy
Unweighted	0.65	0.89	0.76	0.69
Weighted	**0.92**	0.58	0.71	0.75
Without Cam 3	0.91	0.71	0.80	0.79

Since VideoAgent heavily relies on a VLM to select video plans during inference, it is crucial to understand whether the VLM can in fact achieve a reasonable accuracy in providing feedback for video generation. To quantify the performance of a VLM, we use human labels on whether a generated video is acceptable as the ground truth, and measure precision, recall, F1-score, and accuracy based on whether GPT-4 Turbo thinks the generated video is acceptable according to trajectory smoothness (consistent across sequential frames), physical stability, and achieving the goal (See full prompt in Appendix C). We report the average result across 36 generated videos from the Meta-World dataset in Table 4. We see that the original prompt we used (Unweighted, meaning not emphasizing reduction of false positives) achieves 69% accuracy, suggesting that the VLM is able to somewhat judge generated videos but not always accurately. Since VideoAgent uses multiple refinement iterations, we want to avoid false positives where a bad video is accidentally accepted. We can achieve this by penalizing false positives through reweighing its cost in the prompt, which leads to the VLM rejecting videos when the VLM is uncertain about the video's acceptability. This adjustment results in a significant increase in precision as shown in Table 4. This weighted version of the prompt is used in the experiments in Section 4.1.

Partial Observability. In the AVDC experimental setup, center cropping the third camera (what is used in the pipeline) often results in most of the robot arm being outside of the frame. We found that the accuracy of the VLM is affected by such partial observability. As shown in Table 4, removing the third camera from the prompt leads to much higher accuracy.

Descriptive Feedback. While VLM can provide binary feedback on whether a generated video is acceptable, we also measure the accuracy of the VLM in giving more descriptive feedback such as identifying the issue and providing suggestions on how to improve the video. We use three examples with human written language feedback as prompt for in-context learning. GPT-4 Turbo achieves 73.5% accuracy on identification and 86.1% accuracy on suggestion, as evaluated by humans. This result is highly encouraging and opens up future directions of leveraging descriptive feedback from VLMs to improve video generation.

4.3 Evaluating Self-Refinement on Real-World Robot Videos

In this section, we evaluate VideoAgent's ability to refine real-world videos, which often contain higher variability, intricate details, nuanced behaviors, and complex interactions. We study the effect of video refinement using both quantitative metrics and qualitatively for holistic evaluation.

Table 5: BridgeData-V2 Results. Quantitative metrics comparing AVDC and VideoAgent on generated Bridge data. VideoAgent outperforms the baseline according to all except for one metric.

Metrics	AVDC	Video Agent
Clip Score	22.39	**22.90**
Flow Consistency	2.48 ± 0.00	**2.59 ± 0.01**
Visual Quality	1.97 ± 0.003	**2.01 ± 0.003**
Temporal Consistency	1.48 ± 0.01	**1.55 ± 0.01**
Dynamic Degree	**3.08 ± 0.01**	3.07 ± 0.02
Text to Video Alignment	2.26 ± 0.003	**2.30 ± 0.03**
Factual Consistency	2.02 ± 0.004	**2.07 ± 0.01**
Average Video Score	2.16 ± 0.01	**2.20 ± 0.01**
Qualitative Eval on Task Success	42.0%	**64.0%**

We report the average across 2250 videos generated from the AVDC baseline and from VideoAgent in Table 5. VideoAgent performs better according to all metrics except for Dynamic Degree from Video-Score (which shows similar performance between VideoAgent and AVDC). Notably, the gain is significant for metrics critical for instruction following and real-world videos, such as CLIP Score, Factual Consistency, and Text-to-Video Alignment. Improvement in Flow Consistency and Temporal Consistency suggests that VideoAgent produces smoother and more physically plausible videos that adhere better to the physical constraints of the real-world. This directly translates to better performance in real-world robotic tasks in Table 1.

Qualitative Evaluation. Next, we qualitatively evaluate generated videos from the AVDC baseline and from VideoAgent. We collect 50 generated videos from each model and conduct qualitative evaluation on whether a generated video looks realistic. Videos with refinement from VideoAgent improves the acceptance rate by 22% as shown in Table 5. We further show an example video with and without refinement in Figure 8 which we provide in Appendix K, where the baseline (middle row) hallucinates (the bowl disappears) whereas VideoAgent produces the video that completes the task (bottom row). We also present a more fine-grained analysis of Visual Quality, Temporal Consistency, Dynamic Degree, Text to Video Alignment, and Factual Consistency evaluated qualitatively in the Appendix J with the metrics in Table 9, which further echos the results of qualitative evaluations presented in Table 5.

Quantitative Evaluation. Following previous literature on video generation, we consider two reference-free metrics, CLIP Score (Hessel et al., 2021) and Flow Consistency (Teed & Deng, 2020), as well as a set of Video-Scores (He et al., 2024) for evaluation. CLIP Score measures the cosine similarity between frame feature and text prompt, whereas Flow Consistency measure the smoothness and coherence of motion in the videos calculated from the RAFT model. Video-Scores use five sub-metrics with a focus on correlation with qualitative evaluation and real-world videos.

5 Related Work

Feedback and Self-improvement in LLMs. Incorporating feedback and preference signals from feedback into the finetuning process of LLMs, has led to the enormous popularity and practical usability of the current versions of LLMs as chatbots (Casper et al., 2023). Preference feedback from humans or other AI systems (Ouyang et al., 2022a; Lee et al., 2023; Kaufmann et al., 2023) are first collected to train a reward model to guide the LLM's generation or do implicit policy optimization (Schulman et al., 2017; Rafailov et al., 2024). Furthermore LLMs have shown the ability to further improve by iterative refinement during finetuning and inference (Zelikman et al., 2022; Yuan et al., 2024; Tian et al., 2024). We incorporate this reward driven improvement mechanism in our work, but unlike the LLM setting where the feedback came from a reward model or some proxy of this preference model, focus on improving video generation using feedback from action execution.

Image and Video Generation and Editing. With the advent of large scale foundation models pretrained on internet scale data (Bommasani et al., 2021), generation of super realistic multimodal content has become easier. Text generation, image or video generation, and cross-modal generation (OpenAI et al., 2024; Reid et al., 2024; Wu et al., 2021; Ho et al., 2022; Singer et al., 2022; Yang et al., 2023a; Blattmann et al., 2023) has seen major advancements leveraging the autoregressive and diffusion based models architectures. And moving beyond simple generation, these models have been leveraged for guided text, image or video editing and enhancement (Huang et al., 2024) to improve textual and visual aesthetics applied mostly to generative media (Zhang et al., 2023). But none of these existing methods focus on grounding a generative simulator in the real world to perform more complex interactive multi-turn agentic and physical tasks needing both perception and control. To solve this bottleneck, we propose VideoAgent to self-improve or edit generated plan based on grounded feedback from real-world to execute robot manipulation tasks.

Scaling Inference-Time Compute. Beyond pretraining, increasing inference-time compute offers a complementary path to improve model performance. In LLMs, this includes enhanced planning via multiple generations and verifier-guided decoding (Xie et al., 2024; Gandhi et al., 2024; Lightman et al., 2023; Snell et al., 2024). Similar strategies have extended to diffusion models, such as increasing denoising steps to boost generation quality (Karras et al., 2022; Song et al., 2020a,b). Our method combines these test-time refinements with further training the model to self-improve, enabling the model to *learn* to improvement through both gradient-based updates while also leveraging extra compute at inference to further refine video plans.

Video Generation for Robot Learning. Video-based learning for robotics (Nair et al., 2022; Bahl et al., 2022; Shao et al., 2021; Chen et al., 2021; Pari et al., 2022) has enabled visual representation learning, goal extraction, planning (Finn & Levine, 2017; Kurutach et al., 2018), and imitation from expert actions (Fang et al., 2019; Wang et al., 2023; Mani et al., 2024). Recent works reframe decision-making as video generation, enabling policy learning from video predictions (Du et al., 2024; Ko et al., 2023; Wen et al., 2023; Du et al., 2023; Ajay et al., 2024), and use generative models to simulate agent-environment interactions (Yang et al., 2023b). While some methods replan during test time (Bu et al., 2024), VideoAgent refines video plans during training incorporating feedback from failed executions grounded in real-world and further refines the mistakes during test time via self-iteration and replanning.

6 Conclusion, Limitations and Future Work

We have presented VideoAgent, where a video generation model acts as an agent by generating and refining video plans, converting video plans into actions, executing the actions in an environment, and collecting additional data for further self improvement. Through interaction with an external environment, VideoAgent provides a promising direction for grounding video generation in the real world, thereby reducing hallucination and unrealistic physics in the generated videos according to real-world feedback.

Limitations and Future Work. VideoAgent needs to overcome a few limitations. In the online setting, it only considers filtering for successful trajectories for further finetuning, though exploring algorithms such as online reinforcement learning remains promising. VideoAgent currently utilizes optical flow for action extraction, but alternative approaches like inverse dynamics models or image-goal-conditioned diffusion policies may offer improved performance. While we measured end-to-end task success in simulated robotic settings, evaluating VideoAgent in real robotic systems is an important direction for future work. As additional data is collected online, not only the video prediction model but also the action extraction module (flow model) and the VLM feedback model can be finetuned using this data, which we defer to future exploration. Moreover, VideoAgent trades off inference-time compute for better performance by iteratively refining generated video plans under VLM guidance, and investigating alternative inference-time search strategies may further enhance video quality.

References

Anurag Ajay, Seungwook Han, Yilun Du, Shuang Li, Abhi Gupta, Tommi Jaakkola, Josh Tenenbaum, Leslie Kaelbling, Akash Srivastava, and Pulkit Agrawal. Compositional foundation models for hierarchical planning. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 9

Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. Human-to-robot imitation in the wild. In *Robotics: Science and Systems*, 2022. 9

Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023. 3

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. *arXiv preprint arXiv:2304.08818*, 2023. 9

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. 9, 29

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. 2024. *URL https://openai. com/research/video-generation-models-as-world-simulators*, 3, 2024. 1

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. *arXiv preprint arXiv:2402.15391*, 2024. 1

Qingwen Bu, Jia Zeng, Li Chen, Yanchao Yang, Guyue Zhou, Junchi Yan, Ping Luo, Heming Cui, Yi Ma, and Hongyang Li. Closed-loop visuomotor control with generative expectation for robotic manipulation. *2409.09016*, 2024. 9

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023. 9

Annie S Chen, Suraj Nair, and Chelsea Finn. Learning Generalizable Robotic Reward Functions from "In-The-Wild" Human Videos. In *Robotics: Science and Systems*, 2021. 9

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022. 4

Giannis Daras, Yuval Dagan, Alex Dimakis, and Constantinos Daskalakis. Consistent diffusion models: Mitigating sampling drift by learning to be consistent. *Advances in Neural Information Processing Systems*, 36, 2024. 2

Yilun Du, Mengjiao Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Brian Ichter, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, et al. Video language planning. *arXiv preprint arXiv:2310.10625*, 2023. 1, 2, 3, 4, 9

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. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 2, 9

Bin Fang, Shidong Jia, Di Guo, Muhua Xu, Shuhuan Wen, and Fuchun Sun. Survey of Imitation Learning for Robotic Manipulation. *International Journal of Intelligent Robotics and Applications*, 2019. 9

Chelsea Finn and Sergey Levine. Deep Visual Foresight for Planning Robot Motion. In *IEEE International Conference on Robotics and Automation*, 2017. 9

Kanishk Gandhi, Denise Lee, Gabriel Grand, Muxin Liu, Winson Cheng, Archit Sharma, and Noah D Goodman. Stream of search (sos): Learning to search in language. *arXiv preprint arXiv:2404.03683*, 2024. 9

Xuan He, Dongfu Jiang, Ge Zhang, Max Ku, Achint Soni, Sherman Siu, Haonan Chen, Abhranil Chandra, Ziyan Jiang, Aaran Arulraj, et al. Videoscore: Building automatic metrics to simulate fine-grained human feedback for video generation. *arXiv preprint arXiv:2406.15252*, 2024. 8, 20

Jonathan Heek, Emiel Hoogeboom, and Tim Salimans. Multistep consistency models. *arXiv preprint arXiv:2403.06807*, 2024. 2

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021. 8

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020. 3

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. *arXiv preprint arXiv:2210.02303*, 2022. 1, 9

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022. 2

Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022. 1

Yi Huang, Jiancheng Huang, Yifan Liu, Mingfu Yan, Jiaxi Lv, Jianzhuang Liu, Wei Xiong, He Zhang, Shifeng Chen, and Liangliang Cao. Diffusion model-based image editing: A survey. *arXiv preprint arXiv:2402.17525*, 2024. 9

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022. 9

Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. A survey of reinforcement learning from human feedback. *arXiv preprint arXiv:2312.14925*, 2023. 9

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to act from actionless videos through dense correspondences. *arXiv preprint arXiv:2310.08576*, 2023. 1, 2, 3, 4, 5, 6, 9, 17, 18

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. 2, 5, 17, 18

Thanard Kurutach, Aviv Tamar, Ge Yang, Stuart J Russell, and Pieter Abbeel. Learning Plannable Representations with Causal InfoGAN. In *Neural Information Processing Systems*, 2018. 9

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023. 9

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023. 9

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022. 4

Sabariswaran Mani, Sreyas Venkataraman, Abhranil Chandra, Adyan Rizvi, Yash Sirvi, Soumojit Bhattacharya, and Aritra Hazra. Diffclone: Enhanced behaviour cloning in robotics with diffusion-driven policy learning. *arXiv:2401.09243*, 2024. 9

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A Universal Visual Representation for Robot Manipulation. In *Conference on Robot Learning*, 2022. 9

Achiam J OpenAI, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. 2023. *URL: https://arxiv. org/abs/2303.08774*, 2024. 9

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155, 2022a. 9

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022b. 2

Jyothish Pari, Nur Muhammad Shafiullah, Sridhar Pandian Arunachalam, and Lerrel Pinto. The Surprising Effectiveness of Representation Learning for Visual Imitation. In *Robotics: Science and Systems*, 2022. 9

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988. 3

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024. 9

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024. 9

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017. 5, 9

Lin Shao, Toki Migimatsu, Qiang Zhang, Karen Yang, and Jeannette Bohg. Concept2Robot: Learning Manipulation Concepts from Instructions and Human Demonstrations. *IJRR*, 2021. 9

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 1, 9

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024. 2, 9

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020a. 3, 4, 9

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*, 2023. 3

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b. 3, 9

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023. 2, 3

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 5

Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pp. 402–419. Springer, 2020. 8

Ye Tian, Baolin Peng, Linfeng Song, Lifeng Jin, Dian Yu, Haitao Mi, and Dong Yu. Toward self-improvement of llms via imagination, searching, and criticizing. *arXiv preprint arXiv:2404.12253*, 2024. 9

Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pp. 1723–1736. PMLR, 2023. 5, 17, 18

He Wang, Sören Pirk, Ersin Yumer, Vladimir G Kim, Ozan Sener, Srinath Sridhar, and Leonidas J Guibas. Learning a generative model for multi-step human-object interactions from videos. In *Computer Graphics Forum*, volume 38, pp. 367–378. Wiley Online Library, 2019. 1

Hsiang-Chun Wang, Shang-Fu Chen, and Shao-Hua Sun. Diffusion Model-Augmented Behavioral Cloning. *arXiv:2302.13335*, 2023. 9

Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023. 1, 9

Chenfei Wu, Lun Huang, Qianxi Zhang, Binyang Li, Lei Ji, Fan Yang, Guillermo Sapiro, and Nan Duan. Godiva: Generating open-domain videos from natural descriptions. *arXiv preprint arXiv:2104.14806*, 2021. 9

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*, 2024. 2

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024. 9

Mengjiao Yang, Yilun Du, Bo Dai, Dale Schuurmans, Joshua B Tenenbaum, and Pieter Abbeel. Probabilistic adaptation of text-to-video models. *arXiv preprint arXiv:2306.01872*, 2023a. 9

Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023b. 1, 4, 9

Sherry Yang, Jacob Walker, Jack Parker-Holder, Yilun Du, Jake Bruce, Andre Barreto, Pieter Abbeel, and Dale Schuurmans. Video as the new language for real-world decision making. *arXiv preprint arXiv:2402.17139*, 2024. 1, 2

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020. 2, 5, 17, 18

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024. 9

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022. 9

Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, and In So Kweon. Text-to-image diffusion model in generative ai: A survey. *arXiv preprint arXiv:2303.07909*, 2023. 9

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *arXiv preprint arXiv:2204.13902*, 2022. 4

Junchen Zhu, Huan Yang, Huiguo He, Wenjing Wang, Zixi Tuo, Wen-Huang Cheng, Lianli Gao, Jingkuan Song, and Jianlong Fu. Moviefactory: Automatic movie creation from text using large generative models for language and images. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 9313–9319, 2023. 1

Appendix

A Impact Statement

VideoAgent introduces a novel self-conditioning consistency mechanism that enables iterative refinement of generated video plans, significantly improving long-horizon task completion. By leveraging previously generated video segments for refinement, VideoAgent mitigates hallucinations and enhances temporal consistency without requiring extensive interaction with the environment. This reduces the need for costly and time-consuming data collection while still achieving state-of-the-art success rates in simulated robotic environments. Furthermore, VideoAgent 's ability to refine plans without relying on replanning makes it highly adaptable to real-world applications, including robotics, autonomous systems, and video-based reinforcement learning. This work advances scalable and generalizable video generation techniques, contributing to the broader goal of AI agents that can reason and act through visual understanding.

B Algorithms

Algorithm 1: Training of Video Generation and Refinement Models with VLM Feedback

```
Input: Dataset $\mathcal{D}$, learning rate $\gamma$, total training iterations $N$, initial model parameters $\theta$, video
       generation model $f_\theta$, video refinement model $\hat{f}_\theta$, VLM $\hat{\mathcal{R}}$
for iteration $= 1$ to $N$ do
    Sample $\{(\mathbf{x}^{(0)}, g)\} \sim \mathcal{D}$ and $t \sim \text{Uniform}(\{0, 1, \ldots, T\})$;
    Compute vanilla diffusion loss:
        $\mathcal{L}_{\text{video-diffusion}} = \left\| f_\theta(\mathbf{x}^{(t)}, t) - \mathbf{x}^{(0)} \right\|^2$;
    Generate $\hat{\mathbf{x}}$ following Equation 11 and sample feedback $\sim \hat{\mathcal{R}}(\cdot|\hat{\mathbf{x}})$;
    Compute consistency loss:
        $\mathcal{L}_{\text{SC-consistency}} = \left\| \hat{f}_\theta(\hat{\mathbf{x}}, \mathbf{x}^{(t)}, t \,|\, \texttt{feedback}) - \mathbf{x}^{(0)} \right\|^2$;
    Update parameters:
        $\theta \leftarrow \theta - \gamma \nabla_\theta \left( \mathcal{L}_{\text{video-diffusion}} + \mathcal{L}_{\text{SC-consistency}} \right)$;
```

Algorithm 2: VLM Guided Replan

```
Input: Initial frame $x_0$, task description $g$, Reward $\mathcal{R}$, Environment $\mathcal{E}$, VLM $\hat{\mathcal{R}}$,
       max_refine_iterations, max_replans
for replan_count $= 1$ to max_replans do
    $\hat{\mathbf{x}} \leftarrow \pi_\theta(x_0, g)$;
    for $i = 0$ to max_refine_iterations do
        response $\leftarrow \hat{\mathcal{R}}(\hat{\mathbf{x}}_{(i)}, g)$;
        if response == ACCEPT then break;
        $\hat{\mathbf{x}}_{(i+1)} \leftarrow \pi_\theta(\hat{\mathbf{x}}_{(i)}, x_0, g)$;
    success $\leftarrow \mathcal{R}(\rho(\hat{\mathbf{x}}_{\text{refined}}))$;
    if success then break;
    $x_0 \leftarrow \mathcal{E}$.get_state();
```

Algorithm 3: Online Finetuning of Video Generation and Refinement Models

```
Input: Dataset $\mathcal{D}$, policy $\pi_\theta$, Reward $\mathcal{R}$, Environment $\mathcal{E}$
for iteration $i = 1$ to $N$ do
    $\mathcal{D}_{\text{new}} \leftarrow \emptyset$;
    for each $(\cdot, g)$ in $\mathcal{D}$ do
        $x_0 \leftarrow \mathcal{E}$.reset($g$);
        $\hat{x}_{\text{refined}} \sim \pi_\theta(x_0, g)$;
        if $\mathcal{R}(\rho(\hat{x}_{refined}))$ then
            $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{new}} \cup (\hat{x}_{\text{refined}}, g)$;
    $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_{\text{new}}$;
    Finetune $\theta$ using Algorithm 1 on $\mathcal{D}$;
```

C Prompt Structure for VLM Feedback

C.1 Binary Classification

We employ a structured prompting strategy to provide feedback on video sequences for the zero-shot classification. The process consists of one Query-Evaluation Phase, each with distinct sub-goals.

BINARY CLASSIFICATION

Task: You are a video reviewer evaluating a sequence of actions presented as seven consecutive image uploads, which together represent a single video. You are going to accept the video if it completes the task and the video is consistent without glitches.
Query-Evaluation Phase:

- **Inputs Provided:**
  - **Textual Prompt:** Describes the task the video should accomplish.
  - **Conditioning Image:** Sets the fixed aspects of the scene.
  - **Sequence of Images (7 Frames):** Represents consecutive moments in the video to be evaluated.
- **Evaluation Process:**
  - **View and Analyze Each Frame:** Examine each image in sequence to understand the progression and continuity of actions.
  - **Assess Overall Coherence:** Determine if actions transition smoothly and logically from one image to the next.
  - **Check for Physical Accuracy:** Ensure adherence to the laws of physics, identifying any discrepancies.
  - **Verify Task Completion:** Confirm the sequence accomplishes the task described in the textual prompt.
  - **Identify Inconsistencies:** Detect inconsistencies in object movement or overlaps that do not match the conditioning image.
- **Evaluation Criteria:**
  - Accept the sequence if it is a coherent video that completes the task.
  - Reject the sequence if any frame fails to meet the criteria, showing inconsistencies or not achieving the task. Be very strict, rejecting even minor errors.
- **Response Requirement:**
  - Provide a single-word answer: *Accept* or *Reject*. Do not give reasoning.
- **Additional Notes:**
  - No further clarification can be requested.
  - Elements from the conditioning image must match those in each frame of the sequence.

C.2 Identification and Suggestion:

We employ a structured prompting strategy to provide descriptive feedback on video sequences via an in-context few-shot classification setup. The process consists of one Query-Evaluation Phase, each with distinct sub-goals.

IDENTIFICATION AND SUGGESTION

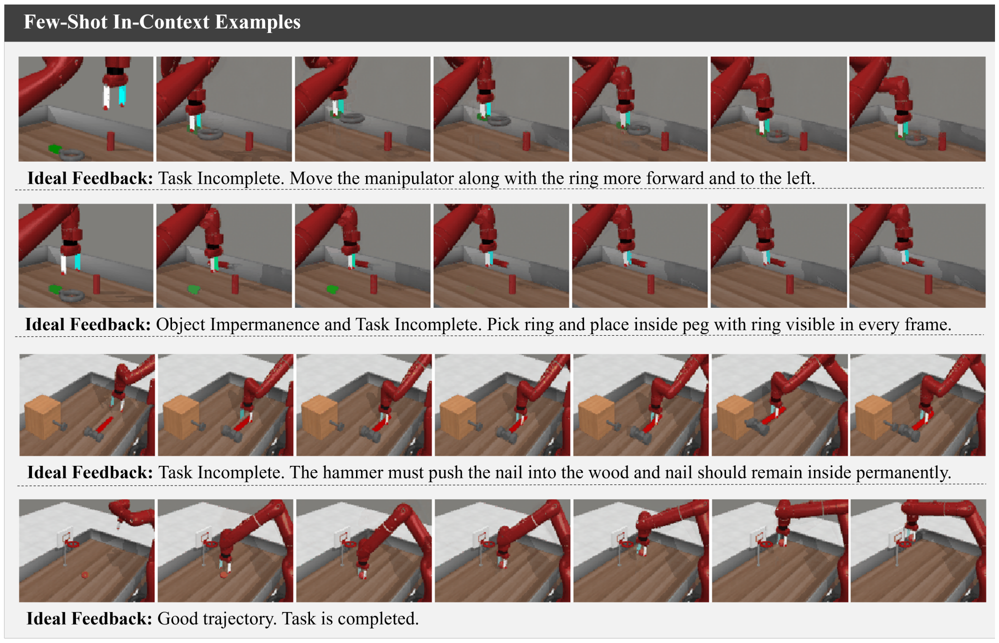
Task: You are a video reviewer tasked with evaluating a series of actions depicted through eight consecutive image uploads. These images together simulate a video. This task is structured as a few-shot learning exercise, where you will first review three examples and then apply learned principles to new queries. **Query-Evaluation Phase:**

- **Inputs Provided:**
  - **Textual Prompt:** Describes the intended outcome or task the video aims to accomplish.
  - **Conditioning Image:** Establishes the fixed elements of the scene.
  - **Sequence of Images (7 Frames):** Illustrates consecutive moments in the video, representing the action sequence.
- **Evaluation Process:**
  - **Frame-by-Frame Analysis:** Carefully examine each of the seven images to understand the progression and continuity of actions.
  - **Assess Overall Coherence:** Evaluate the sequence as a whole to determine if the actions transition smoothly from one frame to the next while maintaining logical progression.
  - **Check for Physical Accuracy:** Ensure each frame complies with the laws of physics, identifying any discrepancies in movement or positioning.
  - **Verify Task Completion:** Confirm if the sequence as a whole accomplishes the task described in the textual prompt.
  - **Identify Inconsistencies:** Detect inconsistencies in object movement or overlaps that contradict the fixed scene elements depicted in the conditioning image.
- **Evaluation Criteria:**
  - **Descriptive Feedback:** Based on your evaluation, provide a concise, constructive sentence suggesting specific improvements. Focus on enhancing physical accuracy and task fulfillment based on identified inconsistencies or discrepancies.
- **Response Requirement:**
  - Feedback must be derived from your observations during the evaluation and not exceed 20 words.
- **Additional Notes:**
  - No further clarification can be requested.
  - Elements from the conditioning image must match those in each frame of the sequence.

D Task Descriptions and In-Context Examples for VLM Feedback

TASK DESCRIPTION AND SUCCESS CRITERIA

- **door-open**: The robot arm has to open the door by using the door handle.
- **door-close**: The robot arm has to close the door by pushing the door or the handle.
- **basketball**: The robot arm has to pick up the basketball and take it above the hoop.
- **shelf-place**: The robot arm has to pick up the blue cube and place it on the shelf.
- **button-press**: The robot arm has to press the red button from the side by pushing it inside.
- **button-press-topdown**: The robot arm has to press the red button from the top by pushing it downward.
- **faucet-close**: The robot arm has to use the red faucet handle and turn it anti-clockwise.
- **faucet-open**: The robot arm has to use the red faucet handle and turn it clockwise.
- **handle-press**: The robot arm has to press the red handle downward.
- **hammer**: The robot arm has to grip and pick up the hammer with a red handle and hit the peg on the box inside.
- **assembly**: The robot arm has to pick up the ring and place it into the red peg.



Ideal Feedback: Task Incomplete. Move the manipulator along with the ring more forward and to the left.

Ideal Feedback: Object Impermanence and Task Incomplete. Pick ring and place inside peg with ring visible in every frame.

Ideal Feedback: Task Incomplete. The hammer must push the nail into the wood and nail should remain inside permanently.

Ideal Feedback: Good trajectory. Task is completed.

Figure 5: Few-Shot Examples given to VLM: We provide some examples to the VLM and corresponding feedback to teach the VLM in-context how to critic the generated videos for task completion and success or failure.

E Dataset Descriptions in Detail

E.1 Datasets and Environments.

We follow the same evaluation setting as (Ko et al., 2023), which considers three datasets: Meta-World (Yu et al., 2020), iTHOR (Kolve et al., 2017), and BridgeData V2 (Walke et al., 2023). Meta-World consists of 11 robotic manipulation tasks performed by a simulated Sawyer arm, with video demonstrations captured from three distinct camera angles. iTHOR is a simulated 2D object navigation benchmark, where an agent searches for specified objects across four room types. BridgeData V2 is a real-world dataset of robotic manipulation.

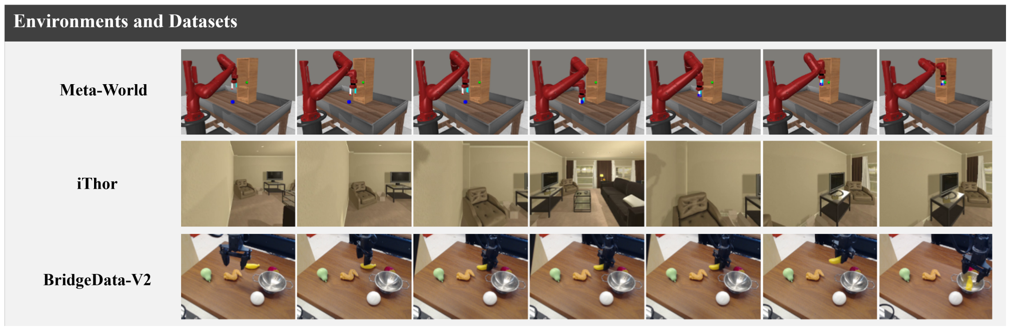


Figure 6: Environments and Datasets that we work with: Meta-World, iThor, and BridgeData-V2

Meta-World (Yu et al., 2020) is a simulation benchmark that uses a Swayer robotic arm to perform a number of manipulation tasks. In our experiments, we make use of 11 tasks as shown in Table 1. We capture videos from three distinct camera angles for each task and use the same camera angles for both the training and testing phases. We gather five demonstration videos per task for each camera angle. During the evaluation, we tested on each of the three camera angles with 25 seeds per camera angle. The position of the robot arm and the object is randomized at the beginning of each seed to ensure variability. A trajectory is considered successful if the Video Agent reaches within a really close threshold of the goal state.

iTHOR (Kolve et al., 2017) is another popular 2D simulated benchmark that focuses on embodied common sense reasoning. We evaluate the Video as Agent framework on the object navigation tasks, where an agent is randomly initialized in a scene and tasked with finding an object of a specified type (e.g., toaster, television). At each time step, the agent can take one of the four possible actions (MoveForward, RotateLeft, RotateRight, or Done), and observes a 2D scene to operate in. We selected 12 objects ((e.g. toaster, television) to be placed in 4 different room types (e.g. kitchen, living room, bedroom, and bathroom). Again, the starting position of the agent is randomized at the start of each episode. During evaluation, we test the agent across 12 object navigation tasks spread across all 4 room types, 3 tasks per room. A trajectory is successful if the agent views and reaches within 1.5 meters of the target object before reaching the maximum environment step or predicting Done.

To test the usefulness of our framework across different videos types, we also use the BridgeData V2 dataset (Walke et al., 2023), a large and diverse dataset of real world robotic manipulation behaviors designed to facilitate research in scalable robot learning. It contains 60,096 trajectories collected across 24 environments using a publicly available low-cost WidowX 250 6DOF robot arm. The dataset provides extensive task and environment variability, enabling skills learned from the data to generalize across environments and domains.

E.2 Additional trajectories per iteration during online training

We collect 15 successful trajectories for each task during every iteration. This standardization helps address task imbalance, as task success rates are higher for certain tasks compared to others. By ensuring a fixed number of successful trajectories per task, we prevent overfitting to easier tasks and maintain balanced model performance across the entire task set. The set of seeds used for training and collecting additional trajectories are different from the seeds used for evaluation.

F Baselines and VideoAgent Variants.

We consider the following methods for comparison:

- **AVDC** (baseline). This is the Actions from Video Dense Correspondences (Ko et al., 2023) baseline, which synthesizes a video and predicts optical flow to infer actions.
- **AVDC-Replan** (baseline). When the movement stalls, AVDC-replan re-runs video generation and action extraction from the flow model to execute a new plan.
- **VideoAgent**. Our proposed video refinement model through self-conditioning consistency as introduced in Section 3.1. VideoAgent generates video and iteratively refines a video plan. We use GPT-4 Turbo for selecting the best video plan during inference (Section 3.2).
- **VideoAgent-Online**. As actions are executed in the online environment, successful trajectories are collected and used to continue training the video generation and refinement model, as described in Section 3.3.

Table 6: Meta-World Results. The mean success rates of baselines and VideoAgent on 11 simulated robot manipulation environments from Meta-World. VideoAgent consistently outperforms baselines across all tasks.

	door-open	door-close	basketball	shelf-place	btn-press	btn-press-top
BC-Scratch	21.3%	36.0%	0.0%	0.0%	34.7%	12.0%
BC-R3M	1.3%	58.7%	0.0%	0.0%	36.0%	4.0%
UniPi (with Replan)	0.0%	36.0%	0.0%	0.0%	6.7%	0.0%
AVDC	30.7%	28.0%	21.3%	8.0%	34.7%	17.3%
VLP	33.3%	28.0%	17.3%	8.0%	36.0%	18.7%
Diffusion Policy	45.3%	45.3%	8.0%	0.0%	40.0%	18.7%
AVDC-Replan	72.0%	89.3%	37.3%	18.7%	60.0%	24.0%
AVDC-IS-Replan	66.7%	93.3%	40.0%	21.3%	65.3%	29.3%
VideoAgent	40.0%	29.3%	13.3%	9.3%	38.7%	18.7%
VideoAgent (Iter2)	48.0%	40.0%	24.0%	12.0%	42.7%	36.0%
VideoAgent (Iter3)	58.7%	50.7%	28.0%	18.7%	53.3%	41.3%
VideoAgent-Replan	**82.7%**	**97.3%**	**40.0%**	**26.7%**	**73.3%**	**44.0%**

	faucet-close	faucet-open	handle-press	hammer	assembly	**Overall**
BC-Scratch	18.7%	22.7%	28.0%	0.0%	0.0%	15.4%
BC-R3M	18.7%	17.3%	37.3%	0.0%	1.3%	16.2%
UniPi (with Replan)	4.0%	9.3%	13.3%	4.0%	0.0%	6.1%
AVDC	12.0%	17.3%	41.3%	0.0%	5.3%	19.6%
VLP	30.7%	10.7%	33.3%	0.0%	1.3%	19.8%
Diffusion Policy	22.7%	**58.7%**	21.3%	4.0%	1.3%	24.1%
AVDC-Replan	53.3%	24.0%	81.3%	8.0%	6.7%	43.1%
AVDC-IS-Replan	48.0%	28.0%	78.7%	10.7%	0.0%	43.8%
VideoAgent	46.7%	12.0%	36.0%	0.0%	1.3%	22.3%
VideoAgent (Iter2)	53.3%	28.0%	52.0%	1.3%	5.3%	31.2%
VideoAgent (Iter3)	58.7%	36.0%	64.0%	1.3%	9.3%	38.2%
VideoAgent-Replan	**74.7%**	**46.7%**	**86.7%**	**8.0%**	**10.7%**	**53.7%**

- **VideoAgent-Replan**. This variant incorporates online filtering of successful trajectories with the replanning mechanism, where replanning is conducted first, and more successful trajectories after replanning are added back to the training data.

G Extended Experiments

G.1 Videos to action conversion

We employ the GMFlow optical flow model to predict dense pixel movements across frames. These predicted flows serve as the foundation for reconstructing both object movements and robot motions depicted in the video. The flow predictions allow us to interpret the temporal evolution of the video in terms of actionable physical dynamics. The optical flow essentially provides a dense correspondence of pixel movements between consecutive frames, which is then used to infer the relative motion of objects and the robot. This mapping bridges the gap between the high-dimensional video representation and the low-level control commands required to execute the tasks in a simulated or real environment.

This method ensures that the generated video plans are actionable and aligned with the task-specific dynamics, making the video generation process directly relevant to downstream policy learning and execution.

G.2 Baseline experiments on Metaworld

We conduct experiments on additional baselines including, Behavioral Cloning (BC), UniPi (with replan), VLP, AVDC-IS-Replan and Diffusion policy. Table 6 consists of these results. **VLP** follows a training setup similar to ours, but does not incorporate the proposed self-consistency loss. **AVDC-IS-Replan** refers to the baseline AVDC model with replanning and a straightforward inference-time scaling strategy, wherein the number of denoising time-steps is increased from 100 to 500 during inference. Our method surpasses all the above baselines considered.

G.3 Further Analysis of VideoAgent-Online

We train VideoAgent-Online for multiple iterations and observe that after 2 iterations, the results start to stabilize. The extra results for iteration 3 are also shown in table 7.

	door-open	door-close	basketball	shelf-place	btn-press	btn-press-top
AVDC	30.7%	28.0%	21.3%	8.00%	34.7%	17.3%
VideoAgent	40.0%	29.3%	13.3%	9.3%	38.7%	18.7%
VideoAgent-Online (Iter1)	48.0%	40.0%	24.0%	12.0%	42.7%	36.0%
VideoAgent-Online (Iter2)	58.7%	50.7%	28.0%	18.7%	53.3%	41.3%
VideoAgent-Online (Iter3)	58.7%	52.0%	26.7%	17.3%	54.7%	40.0%

	faucet-close	faucet-open	handle-press	hammer	assembly	**Overall**
AVDC	12.0%	17.3%	41.3%	0.00%	5.30%	19.6%
VideoAgent	46.7%	12.0%	36.0%	0.0%	1.3%	22.3%
VideoAgent-Online (Iter1)	53.3%	28.0%	52.0%	1.3%	5.3%	31.2%
VideoAgent-Online (Iter2)	58.7%	36.0%	64.0%	1.3%	9.3%	38.2%
VideoAgent-Online (Iter3)	56.3%	36.0%	66.7%	1.3%	10.7%	38.22%

Table 7: Meta-World Result. The mean success rates of VideoAgent combined with successive rounds of data collection via Online Iterations and Replan modules as compared to AVDC baseline.

H Architectural Details of VideoAgent

H.1 Video Diffusion training details

We use the same video diffusion architecture as the AVDC baseline. For all models, we use dropout=0, num head channels=32, train/inference timesteps=100, training objective=predict v, beta schedule=cosine, loss function=l2, min snr gamma=5, learning rate=1e-4, ema update steps=10, ema decay=0.999. All of our models and experiments were run on Nvidia A6000 GPUs.

H.2 Inference time speed

In our current setup, during inference, our video generation model produces a new video within 10 seconds on a single A6000 GPU at a resolution of 128×128 for Meta-World. The process of mapping this generated video to an action takes, on average, an additional 25 seconds. This action-mapping stage involves calculating optical flow, receiving feedback from the vision-language model (VLM), and converting the video into an action sequence based on the computed flow.

I VLM Feedback for Correction

	door-open	door-close	basketball	shelf-place	btn-press	btn-press-top
AVDC	30.7%	28.0%	21.3%	8.00%	34.7%	17.3%
VideoAgent	40.0%	29.3%	13.3%	9.3%	38.7%	18.7%
VideoAgent-Binary	46.7%	32.0%	14.7%	6.7%	38.7%	21.3%
VideoAgent-Suggestive	46.7%	33.3%	18.7%	12.0%	41.3%	22.7%
VideoAgent-Online-Suggestive	52.0%	28.0%	21.3%	16.0%	46.7%	22.7%

	faucet-close	faucet-open	handle-press	hammer	assembly	**Overall**
AVDC	12.0%	17.3%	41.3%	0.00%	5.30%	19.6%
VideoAgent	46.7%	12.0%	36.0%	0.00%	1.3%	22.3%
VideoAgent-Binary	46.7%	17.3%	32%	0.00%	5.3%	23.8%
VideoAgent-Suggestive	48.7%	17.3%	46.7%	0.00%	5.3%	26.6%
VideoAgent-Online-Suggestive	45.3%	20.0%	48.0%	2.7%	5.3%	27.4%

Table 8: Meta-World: VideoAgent-Feedback Guided Results The mean success rates for various tasks, comparing different VideoAgent-Feedback Guided variants and the AVDC baseline.

J Details of Qualitative Evaluation on BridgeData V2

Qualitative Evaluation. Next, we qualitatively evaluate video generation quality using the five Video-Score dimensions: Visual Quality (VQ) for clarity and resolution, Temporal Consistency (TC) for smooth frame transitions, Dynamic Degree (DD) for capturing accurate object/environment changes, Text-to-Video Alignment (TVA) for matching the video to the prompt, and Factual Consistency (FC) for adherence to physical laws and real-world facts. Videos are rated on a 4-point scale based on the metric in He et al. (2024): 1 (Bad), 2 (Average), 3 (Good), and 4 (Perfect). Our evaluation is based on 50 generated videos from a held-out set.

Table 9: Task Success and Other Fine-grained Qualitative Evaluation Metrics on BridgeData-V2

Metrics		**AVDC**	**Video Agent**
Task Success via Qualitative Eval		42.0%	**64.0%**
Holistic Assessment via Qualitative Eval	Visual Quality	1.74	1.84
	Temporal Consistency	1.58	1.76
	Dynamic Degree	3.14	2.98
	Text to Video Alignment	2.66	3.04
	Factual Consistency	3.22	3.30
	Qualitative Eval Average	2.47	**2.98**

In terms of VQ and TC, both the baseline AVDC and our VideoAgent generate average quality videos (graded 2), with AVDC hallucinating more and generating some choppy jumps in videos temporally (we grade such videos as 1) and Video Agent fixing some of these upon video conditioned iterative refinement. The reason for AVDC baseline having higher DD is attributed to unruly movements that cause higher DD scores compared to VideoAgent, where movements are smoother. This also explains the result in fifth row of Table 5, and upon closer examination of the generated videos and their corresponding individual scores, we observed similar traits in videos having higher DD due to unnatural robot arm movements and object impermanence. TVA shows trends similar to ClipScore in Table 5 due to the better instruction following ability of VideoAgent leading to more controlled generation. FC is a very crucial metric for deployment of video generation agents as policy for task completion in robotics, scene navigation, and so on. Improved visual quality does not imply adherence to correct physical laws and real-world constraints, FC particularly checks for this aspect and due to video conditioned self-refinement, VideoAgent has better FC compared to AVDC.

K Examples

Additional video examples are provided in the supplementary material.

K.1 Zero-shot generalization on real-world scenes

VideoAgent trained on Bridge dataset demonstrates strong performance on zero shot video generation for natural distribution shifts and longer language instructions. Some examples of the synthesized videos can be found in Fig. 7.

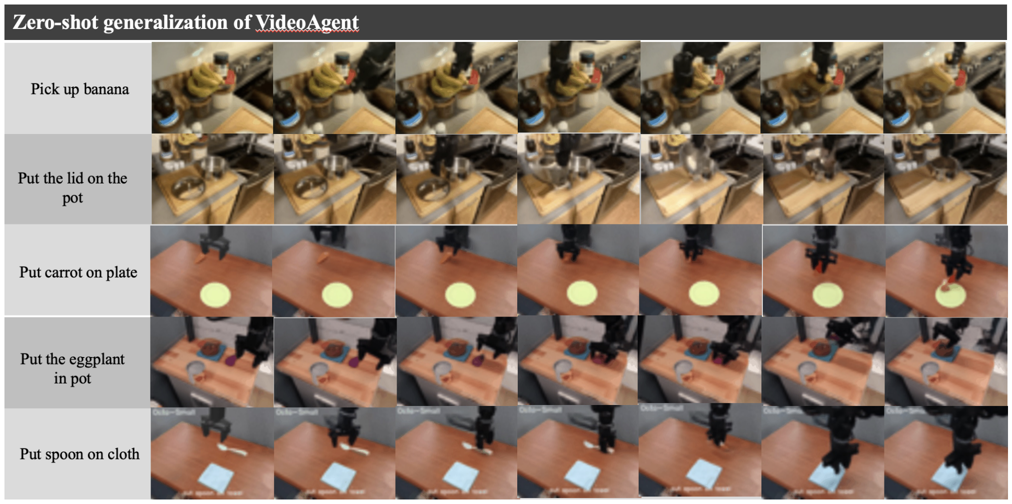


Figure 7: Zero-shot generalization of VideoAgent: VideoAgent generalizes fairly well to natural distribution shifts and is able to generate successful trajectories on data it has not been trained on.

K.2 Improvements on real-world scenes

We show an example video with and without refinement in Figure 8, where the baseline (middle row) hallucinates (the bowl disappears) whereas VideoAgent produces the video that completes the task (bottom row).

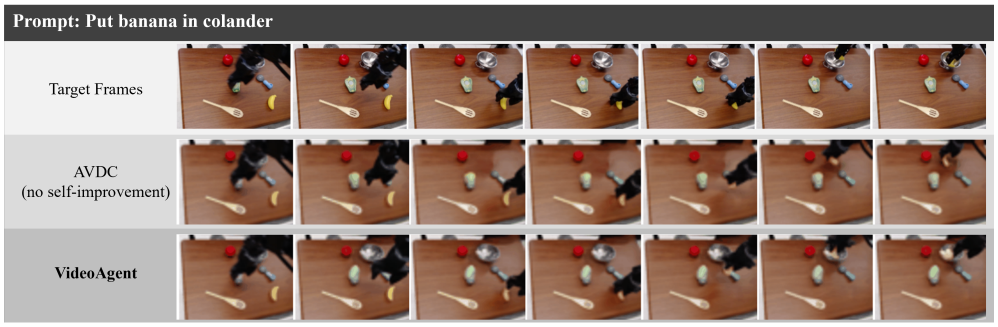


Figure 8: Correcting Hallucinations in Video Generation: The AVDC model hallucinates after the second frame, removing the colander and placing the banana on the table. In contrast, VideoAgent accurately retains the colander's position and correctly places the banana inside.

K.3 Improvements in Meta-World

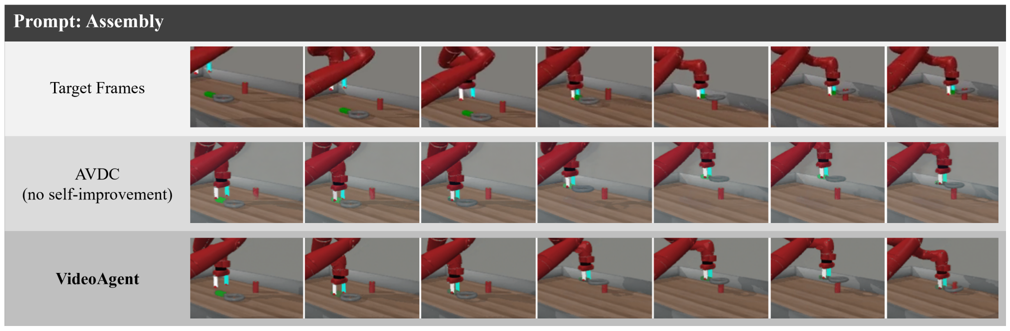


Figure 9: Correcting Hallucinations in Video Generation: The goal prompt is "Assembly" as shown in the Target Video. The AVDC model has problem of object permanence and action incomplete in last frame. In contrast, our VideoAgent model accurately object permanence and correctly places the inside the peg properly.

K.4 Improvements in iThor

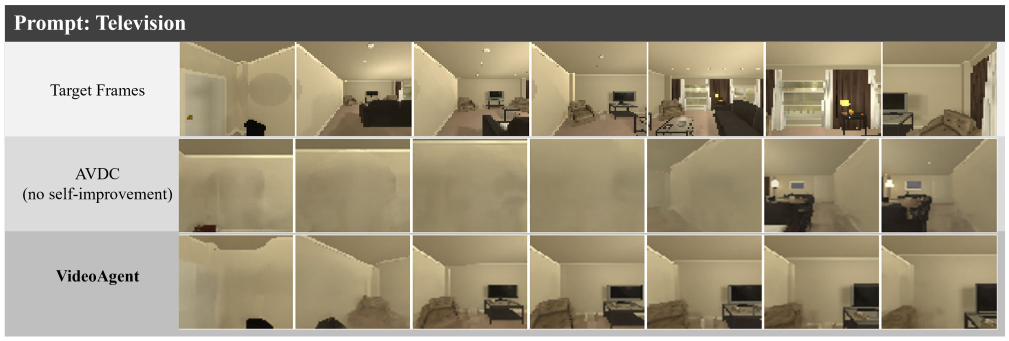
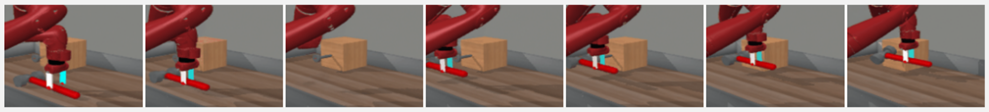


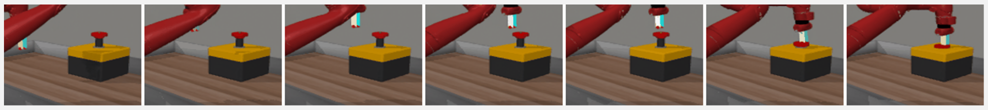
Figure 10: Correcting Hallucinations in Video Generation: The goal prompt is "Television" as shown in the Target Video, the goal is for the navigator to locate the object and reach near it. The AVDC model has difficulty reconstructing and navigating in the livingroom to find the television. In contrast, our VideoAgent model solves the initial frame hallucinations and accurately reaches near the television correctly.

K.5 Identification and Suggestive Feedback Examples

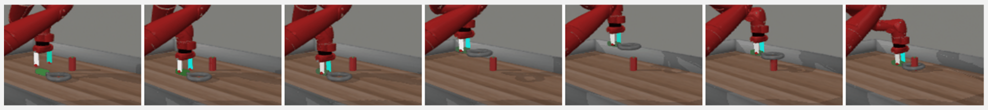
Identification and Suggestive Feedback Examples



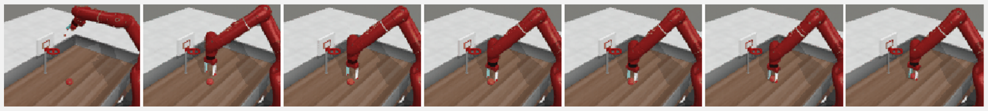
Feedback: The hammer is not aligned to hit the peg; adjust the position for effective striking and contact with nail.



Feedback: The robot arm should not be retracted after pressing the button down, it should hold button down for task completion.



Feedback: The robot arm needs to move the ring directly down into the red peg and hold it in place.



Feedback: The robot arm cannot move and fails to pick up basketball and take it to the hoop.

Figure 11: Detailed VLM Feedback: We show the efficacy of VLMs to provide useful feedback even in the absence of access to a simulator or real-world execution environment. The VLM acts as a proxy reward model to condition VideoAgent on useful corrective signals, leading to improved performance as described in Table 3.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims in the abstract and introduction are that (1) current video generation models for robotic planning suffer from hallucinations and unrealistic physics, (2) external feedback can be used to refine these plans, and (3) the proposed method, VideoAgent, introduces self-conditioning consistency as a self-improvement operator and VLM-guided refinement to iteratively improve video plans, leading to improved downstream task performance. These claims are supported by both the technical methods in Sections 3 and the experimental results in Sections 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a dedicated discussion of limitations in Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: The paper does not contain formal theoretical results or theorems requiring assumptions and proofs. Instead, we propose a novel self-improvement framework for video generation and support it through mathematical formulations (e.g., consistency losses and refinement updates) and empirical evaluations.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides detailed descriptions of the model architecture, training objectives (Equations in Sections 2, 3), experimental setup (Section 4), baselines, and evaluation metrics (Tables 1–5). Algorithms for training, inference, and online finetuning and hyperparameters are explicitly presented in Appendix B, and prompt structures used with the VLM are outlined in Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We include all the code to generate data, train models and perform inference to reproduce our experimental results in the supplemental material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: All details are provided in Section 3 and Section 4 with further details in Appendix B and Appendix C.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: In our work the primary metric is discrete and its the success rate (count of number of successful plans out of all performed rollouts). We thus do not report error bars following previous related works evaluation setup and in the case of the fine-grained metrics for bridge dataset the errors have been shown in Table 5. Also for our setup there isn't a training and testing distribution per say and its just evaluates different seed in these environments to make sure there in no overlap between the generated data for training the models and the test time environment rollouts.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: In the paper we clearly mention the details of compute resources used for all our experiments, models used and api calls made for feedback analysis. All details are present in Section 4 and in Appendix G and H.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The paper presents VideoAgent, a framework to self improve video generation by refining video plans using grounded external feedback, significantly reducing hallucinations and enhancing task success. We show specific applications in simulated robotic manipulation tasks. All the models, datasets and environments we consider are publicly available and widely used in academic research. There are no human participants used in this study. The LLM feedback we used is available via the OpenAI API. While the specific approach and application considered in this paper does not hold potential for negative societal impact, there are still many nefarious ways in which the underlying video models and LLMs can be used ways which have been well documented in the previous papers (Bommasani et al., 2021), hence we do not specifically highlight them here.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The primary goal of this paper is a research study of using a novel self-conditioned consistency loss as a self-improvement operator to improve pre-trained video generation models for real world robot manipulation tasks by grounding them in feedback. While the specific application we consider does not demonstrate any potential for negative societal impact, the use of video generation models and LLMs/VLMs for feedback can have some positive and negative impacts which have been highlighted at length in various pervious works (Bommasani et al., 2021). We have mentioned our impact statement in Appendix A.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

Justification: We generate simulated robot manipulation data from academic open-source environments and use them to train small scale task specific video models that do not pose any particular risk of misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have used publicly available simulation environments and robot manipulation video datasets which are free to use. We have cited the original work for each of these and the license and terms are respected.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We are providing the code for data generation, training and inference with details on how to re-run experiments in our supplementary material. Model checkpoints will also be open-sourced soon.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve any crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve any crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: In the paper we use LLMs/VLMs as proxy reward and feedback models to improve our video generations.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.