AmbiguousWorld: Benchmarking and Resolving Ambiguous Instructions in Video World Models

Bo Wang[1,*] **Sen Cui**[1,2,3,*,‡,§] **Jin Li**[1] **Mingxi Huang**[1]
Zhikang Chen[1] **Min Zhang**[4] **Guocai Yao**[5,‡] **Changshui Zhang**[1,2,3,‡]
[1]Department of Automation, Tsinghua University
[2]BNRist [3]THUAI
[4]East China Normal University [5]BAAI

Abstract

Translating high-level, under-specified human commands into coherent video is fundamentally challenging. Current video generation models lack explicit reasoning capabilities and typically fail to understand ambiguous instructions, frequently resulting in physically inconsistent and causally disjointed generation. To address this, we introduce **AmbiguousWorld**, an inference-time reasoning framework that resolves visually grounded ambiguity through feedback-driven multi-branch rollout over generated video outcomes. We employ a Vision-Language Model (VLM) both as a semantic planner that decomposes an ambiguous instruction into actionable stages and proposes state-conditioned candidate branches, and as a closed-loop critic that prunes invalid rollouts by evaluating them across multiple physical and functional dimensions. To systematically assess this, we construct AmbiguousDROID, an embodied video generation benchmark of 179 ambiguous instructions built on the DROID dataset, and design a multi-dimensional VLM evaluation framework. Empirical results on three state-of-the-art video generation backbones demonstrate consistent improvements across five evaluation dimensions, raising the overall weighted score by up to 24.4 points, and AmbiguousWorld also outperforms compute-matched prompt-extension baselines. Furthermore, we validate our VLM evaluation framework through comparison with blind human assessment, and conduct comparative and ablation studies on each module of the reasoning framework. Videos are available at `https://ambiguousworld-anon.github.io/`.

1 Introduction

Video world models have emerged as critical components in embodied artificial intelligence, facilitating closed-loop training [34], direct action generation [1], and low-cost policy evaluation [24]. The generative reliability of these models depends heavily on the precision of textual conditioning. Clear instructions ensure object correctness, temporal coherence, and causal rationality in the generated trajectories. However, in real-world human-robot interaction and automated data annotation pipelines, instructions are frequently ambiguous.

Ambiguous instructions are characterized by high-level abstractions that lack explicit physical execution details. They contain subjective human intents while omitting operational specificities. Specifically, they suffer from five dimensions of ambiguity:

1. **High-level abstraction**: Summarizes a multi-stage task rather than listing concrete actions, possibly mixing abstraction levels within one instruction.

[*]Equal contribution: Bo Wang and Sen Cui. Jin Li and Mingxi Huang also contributed equally to each other.
[‡]Corresponding authors.
[§]Project Leader.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

2. **Subject ambiguity**: Fails to specify, or refers ambiguously to, the acting agent.
3. **Goal ambiguity**: Target state and success criteria remain unclear.
4. **Object set ambiguity**: Target objects to be manipulated are undefined or ambiguously referenced.
5. **Contextual ambiguity**: The appropriate realization depends on the observed scene (e.g., which appliance or region is relevant), and must be grounded in the visual context.

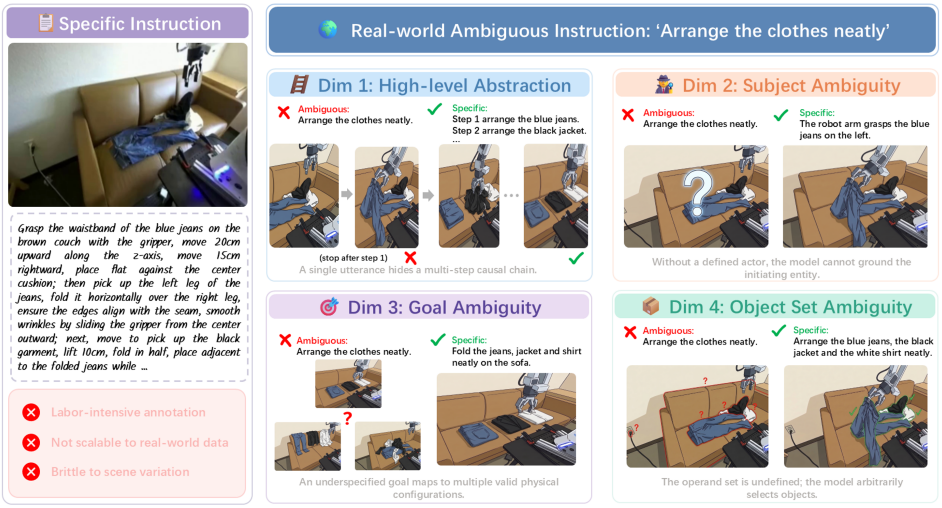


Figure 1: Illustration of multi-dimensional ambiguities in real-world instructions. Compared to labor-intensive, perfectly specified instructions (left), natural human commands are often under-specified. They typically suffer from high-level abstraction, subject ambiguity, goal ambiguity, and object set ambiguity, which cause current video generation models to fail in grounding actors, defining valid physical states, or selecting correct operands. Contextual ambiguity, where the appropriate realization depends on the observed scene, is discussed in Sec. 4.2.

As illustrated in Figure 1, when operating under such under-specified conditions, current video generation models suffer a significant degradation in generative quality, often producing physically inconsistent and causally disjointed outputs. The complexity of this problem extends beyond language; resolving it requires reasoning over visual contexts to ground the missing information, as a single ambiguous instruction can map to multiple valid physical futures in a complex scene.

Prior research addresses ambiguous instructions primarily in the contexts of multi-modal multi-turn question answering [20] and embodied robotic task planning [18]. These approaches leverage external knowledge or explicit reasoning processes to logically parse and disambiguate high-level commands. However, integrating inference-time disambiguation and structured reasoning specifically into the generative pipeline of video world models remains unexplored. The core challenge lies in constructing a framework that can dynamically adjust generative paths based on visual feedback while exploring the combinatorial space of valid physical futures.

To address this, we present AmbiguousWorld, an inference-time reasoning framework built on pre-trained video world models. AmbiguousWorld leverages a VLM as a semantic planner and branch generator. Rather than a direct end-to-end text-to-video mapping, our framework performs a greedy multi-branch rollout: the VLM decomposes the instruction into stages using the initial visual observation and proposes several candidate branches for each stage, and a VLM critic evaluates the rendered rollouts, pruning physically inconsistent or instruction-deviating clips before generation continues. Our goal is not to recover a user's unique latent intention, but to generate one plausible, visually grounded, and executable future for an under-specified instruction whose missing details are inferable from the visible scene.

To systematically assess our approach, we evaluate AmbiguousWorld with state-of-the-art open-weights video generation models, including Cosmos-Predict-2.5, Wan2.2, and CogVideoX, on our curated benchmark. Our reasoning framework consistently improves generative reliability across five physical and semantic dimensions, increasing the weighted total score by up to 24.4 points. Ablation studies isolate the contributions of multi-branch rollout and score-based branch selection, and further comparisons indicate that the gains are not explained by prompt extension or additional sampling

alone. Furthermore, blind human studies validate our automated VLM evaluation framework, with 91% agreement with human judgments.

In summary, our main contributions are:

1. We formalize the challenge of ambiguous instructions in video world models and introduce the Ambiguous-DROID benchmark of 179 ambiguous instructions, alongside a VLM evaluation framework that assesses generation quality across five dimensions.
2. We propose AmbiguousWorld, a training-free reasoning framework that combines prompt-based stage decomposition with greedy multi-branch rollout and VLM-based pruning for multi-step video generation.
3. We provide empirical analysis, including quantitative metrics, ablation studies, comparisons with prompt-extension and reflection-style baselines, and visual case studies across multiple video generation backbones.

2 Related Work

2.1 Reasoning Strategies for Complex Generative Tasks

Early reasoning strategies focus on text-based Chain of Thought [30, 29], enhancing large language models' ability to decompose complex problems into intermediate logical steps. Subsequent self-consistency mechanisms [29] improve reasoning fidelity through multiple sampled trajectories. Building upon these, Tree of Thoughts [36] generalizes language model decoding into deliberate search over a tree of reasoning steps. With multi-modal model advancements, researchers extend these paradigms to visual tasks [31]. VChain [11] uses sparse, causally significant visual thoughts to guide pre-trained video generators, ensuring outputs adhere to physical commonsense. In embodied intelligence, CoT-VLA [37] introduces explicit visual reasoning by autoregressively predicting subgoal images before outputting action sequences.

Leveraging VLMs for dynamic execution, CycleVLA [19] combines proactive self-correction with minimum Bayes risk decoding to anticipate failures and backtrack. Alternatively, VLMs conditionally generate behavior trees [28] for context-aware planning via first-person visual feedback. To resolve physical inconsistencies in long-horizon generation, Embodied Tree of Thoughts [33] and Video Language Planning [5] formulate planning as tree-search, using physical simulators or video models for dynamics alongside VLMs for strategy evaluation and branch pruning. Closest to our setting, RoboDreamer [39] composes language-derived primitives for robot imagination, and RoG [4] refines embodied video predictions through VLM-guided reflection. In contrast, AmbiguousWorld targets under-specified commands with no unique plan: it branches over multiple realized video outcomes and selects among them with visual feedback, without retraining the generator.

2.2 Video World Models

Recent advances in video generation have produced foundation models capable of synthesizing physically plausible world dynamics from text or multi-modal inputs, including Cosmos [22], Wan-2.2 [27], CogVideoX [35], and HunyuanVideo [13]. Building on these priors, specialized action-conditioned and robotic video world models have emerged. For example, CtrlWorld [9] uses video diffusion with action conditioning for faithful manipulation rollouts, while RoboMaster [6] models complex inter-object dynamics and multi-entity interactions in robotic tasks; embodied world models such as GigaWorld-0 [8] and Genie Envisioner [16] further target robot data generation and manipulation. These models enable applications in embodied intelligence, including synthetic data augmentation [22], low-cost policy evaluation [24], and closed-loop imitation learning [34].

Despite this progress, existing models operate as open-loop generative systems lacking an explicit planning layer. When faced with ambiguous instructions, they remain dependent on externally provided detailed prompts and cannot internally resolve underspecification. Prompt extension [27] elaborates a short prompt into a detailed one, but commits to a single continuation without observing its generated outcome. These limitations motivate the inference-time reasoning framework we propose in AmbiguousWorld.

2.3 Generation Task for Ambiguous Instruction

Abstract and under-specified instructions pose a fundamental challenge in realistic human-agent interaction [2, 26]. Prior research addresses this primarily through external knowledge bases or multi-turn dialogues. Visual-O1 [20] demonstrates that even advanced multi-modal models struggle

with ambiguous instructions and proposes simulating human-like multi-turn reasoning and reflection to translate vague commands into definitive steps. In embodied robotics, the EMA framework [18] incorporates external knowledge and dynamic visual perception to logically parse and dynamically allocate subtasks derived from ambiguous verbal commands. Alternatively, the AUTOASK framework [25] trains agents via reinforcement learning to actively generate clarification questions when encountering informational uncertainty. Clarification-based approaches such as CLARA [23] remain necessary when the missing information is not observable. Complementary to them, AmbiguousWorld resolves ambiguity that is inferable from the visual scene, without external dialogue, through visually grounded multi-branch evaluation.

3 Method

AmbiguousWorld is an inference-time reasoning framework designed to tackle the challenge of long-horizon generation in video world models under ambiguous instructions. Direct generation from highly abstract text often leads to physical violations and task failure. By transforming the generative process into a staged multi-branch rollout with visual feedback, AmbiguousWorld produces videos that are physically grounded, causally consistent, and more reliably reach the intended semantic states. Figure 2 presents an overview of our inference-time reasoning framework. More details could be found in Algorithm 1 and Appendix.

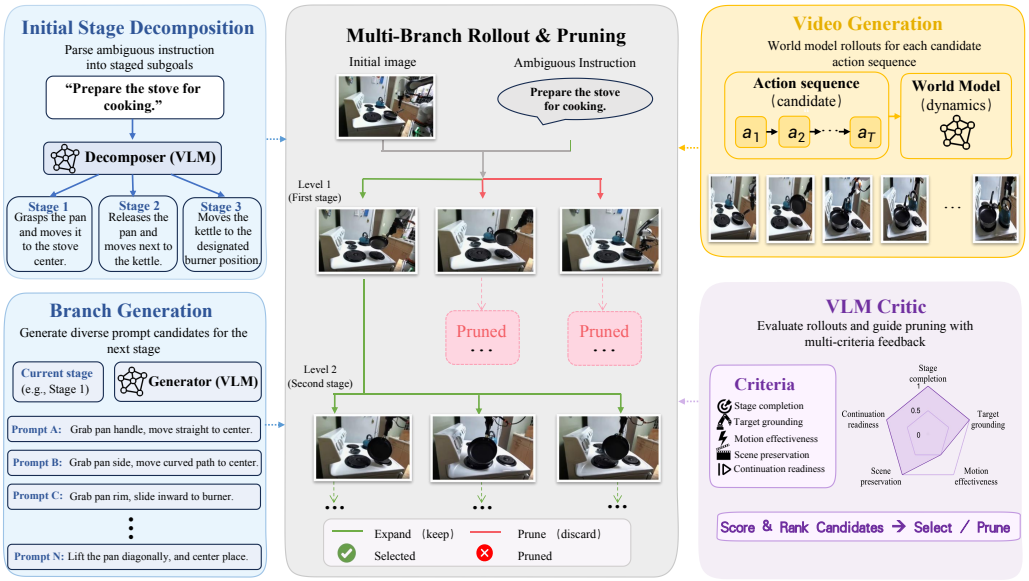


Figure 2: Overview of the proposed AmbiguousWorld framework. A VLM Decomposer first parses the ambiguous instruction into staged subgoals using the initial observation. For the current stage, a VLM Generator produces several prompt candidates (branches) conditioned on the current visual state. The video world model rolls out each branch, and a VLM Critic scores the rollouts across multiple functional criteria (e.g., stage completion, target grounding). Only the highest-scoring valid branch is kept, and its last frame initializes the next stage (greedy multi-branch rollout with VLM-based pruning).

3.1 Problem Formulation

Let x denote the initial state comprising an ambiguous natural language instruction u and an initial visual observation o_0. Ambiguous instructions lack explicit physical execution details. They contain human-readable abstractions but omit critical operational specifics. Consequently, generating a reliable video from x requires inferring the missing context.

We formulate this generative process as a sequential decision problem over a multi-modal thought space $\mathcal{Z}$. A thought $z_t \in \mathcal{Z}$ serves as an intermediate representation containing explicit subtask descriptions, spatial constraints, and generated video clips. The generation state s_t at step t is the cumulative sequence from the initial prompt to the current step defined by $s_t = [x, z_1, z_2, \ldots, z_t]$.

The objective is to find an optimal sequence of visual and semantic thoughts $\mathbf{z}^* = \{z_1, \ldots, z_M\}$ that maximizes joint coherence and task success:

$$\mathbf{z}^* = \arg\max_{\mathbf{z}} \sum_{t=1}^{M} \Big[\log p_\phi^{\text{gen}}(z_t \mid x, c_0, z_{1:t-1}) + \alpha \log V(z_t \mid x, c_0, z_{1:t-1}) \Big], \tag{1}$$

where p_ϕ^{gen} is the thought transition probability, $V \in (0, 1]$ is a value function assessing visual and physical consistency, and α is a temperature scaling factor. Here, c_0 denotes the contextually disambiguated task state derived from the initial configuration. In practice, AmbiguousWorld approximates this objective greedily, stage by stage: the generator only samples candidates, and selection relies on the critic utility (Sec. 3.3).

Assumptions and scope. An ambiguous instruction may admit multiple valid realizations, and the demonstrated trajectory is not treated as the only correct future. We assume that a reasonable interpretation is inferable from the visible scene; private preferences, hidden goals, and safety-critical choices for which a robot should ask the user are outside our scope.

3.2 Stage Decomposition and State-Conditioned Branch Generation

Ambiguous instructions inherently suffer from high-level abstraction and goal ambiguity. Direct end-to-end generation fails because the world model lacks intermediate guidance to bridge the gap between abstract intent and pixel-level physics. To resolve high-level abstraction, the semantic planner must decompose complex tasks into actionable subtasks. To resolve goal ambiguity, it must explicitly define the intermediate objectives and the ultimate target state for each decomposed step.

The VLM divides the generation trajectory based on two primary criteria. First, decomposition relies on identifying distinct task transition boundaries. For instance, the instruction "tidy the desk" is structurally divided into an initial phase of placing scattered objects into a container and a subsequent phase of aligning the container. Second, decomposition occurs at causal divergence points where visual states can evolve into multiple valid physical futures requiring programmatic branching.

The decomposition is prompt-based and requires no task-specific training: given the raw instruction and the first frame, the VLM identifies visible manipulable objects, selects a target region, and outputs at most four chronologically ordered stages in JSON, each operating on one visible object (full prompt in Appendix A). The stage list is produced once; subsequent branch proposals and critic evaluations are conditioned on the visual state selected so far.

Formally, at state s_{t-1}, conditioned on the disambiguated context c_0 and the current stage, the VLM generates K candidate branches that vary the grasp location, approach angle, motion path, or placement position without changing the stage's target object:

$$[z_t^{(1)}, \ldots, z_t^{(K)}] \sim p_\phi^{\text{gen}}(z_t \mid s_{t-1}, c_0). \tag{2}$$

Each generated thought branch $z_t^{(k)}$ contains a rigid generative prompt p_t^k injected with camera constraints and the corresponding video clip rendered by the world model $\mathcal{W}_\theta$:

$$v_t^k = \mathcal{W}_\theta(s_{t-1}, p_t^k). \tag{3}$$

Here, the world model is conditioned on the last frame of the clip selected at the previous stage.

3.3 Greedy Multi-Branch Rollout with VLM-Based Pruning

Real-world dynamics are complex, and generative models frequently hallucinate or violate physical laws when operating over long horizons. Relying on a linear generation path from a decomposed plan is highly fragile. To ensure physical alignment and instruction adherence, the system must evaluate multiple speculative futures and prune invalid ones.

At each stage, the framework expands K candidate clips from the current state, and a VLM critic prunes them in a closed loop, retaining only the most promising branch. This greedy procedure keeps the cost linear in the number of stages, but it does not retain a multi-level frontier, so interactions between choices made at different stages are not explored; a wider beam is a natural extension. We use $K{=}2$ branches, at most four stages, one retry, and one continuation per stage.

For each of the K branches, the critic framework executes a multi dimensional evaluation combining specific functional metrics. The evaluation model assesses the visual clip corresponding to branch

k across a predefined set of dimensions $\mathcal{D}$. These dimensions include stage completion, target grounding, motion effectiveness, scene preservation, and continuation readiness. The model assigns a discrete score $s_{d,k}$ for each dimension $d \in \mathcal{D}$.

Simultaneously, the critic detects fatal geometric or causal violations, outputting a binary failure penalty $\beta_k \in \{0, 1\}$. The overall branch utility V_k is aggregated via a weighted summation modulated by the failure penalty:

$$V_k = [1 - \beta_k] \sum_{d \in \mathcal{D}} w_d s_{d,k}, \tag{4}$$

where w_d is a fixed, hand-set weight and $s_{d,k} \in \{0, \ldots, 4\}$. We use $w = (0.30, 0.20, 0.15, 0.10, 0.25)$ for stage completion, target grounding, motion effectiveness, scene preservation, and continuation readiness, prioritizing functional completion and a usable final frame over minor visual details; severe failures (e.g., wrong target or object disappearance) set β_k=1. We do not claim these weights are optimal (Appendix A).

Furthermore, if a clip is causally correct but halts prematurely with an uncompleted subtask status, we iteratively query the world model to continue the generation. This continuation extends the selected branch in time, resolving early truncation misjudgments without discarding valid causal chains. The framework executes a greedy selection by picking the branch that maximizes the multi dimensional evaluation utility:

$$k^* = \arg\max_k V_k. \tag{5}$$

3.4 Automated Vision-Language Evaluation Framework

Evaluating video generation under ambiguous instructions is challenging: visual metrics lack temporal reasoning, while end-to-end VLMs frequently hallucinate when assessing multi-step trajectories[15, 10]. For rigorous and reproducible comparisons, inspired by WorldGym[24], we introduce an evaluation framework separating visual fact extraction from semantic judgment.

Phase 1: Visual Fact Extraction. Given a generated video v, we uniformly sample discrete frames and prompt a VLM to extract a structured event representation E. This constrains the VLM strictly to factual grounding—identifying actors, tracking entities, and locating spatial anchors—yielding a reliable textual foundation without subjective language hallucination:

$$E = \text{VLM}_{\text{extract}}(v, c_0) \tag{6}$$

Phase 2: Semantic Grading and Rule Override. We apply a deterministic rule engine Φ to the extracted events E to yield binary anomaly flags $A = \Phi(E)$, detecting definitive errors such as unanchored object appearances or actor identity shifts. A text-only reasoning module then outputs a discrete raw score $\tilde{s}_d \in \{0, 0.5, 1\}$ for each evaluation dimension $d \in \mathcal{D}$ (e.g., Task Success, Causal Coherence) conditioned entirely on E and A. Finally, a rigid heuristic filter R acts as a safety net, overriding and capping or nullifying scores if critical visual anomalies are present:

$$s_d = R\left(\tilde{s}_d, E, A\right) \quad \text{where} \quad \tilde{s}_d = \text{VLM}_{\text{score}}(E, A, d) \tag{7}$$

This structural decoupling guarantees that physical violations strictly penalize the final evaluation score without compounding secondary VLM reasoning errors.

4 Experiments

4.1 Baselines

To demonstrate the generalizability of our AmbiguousWorld framework, we select diverse, recent, state-of-the-art open-weights video generation models as our base world models. Specifically, we evaluate on Cosmos-Predict-2.5-2B [22], Wan2.2-TI2V-5B [27], and CogVideoX-1.5-5B [35]. These models represent different architectures and parameter scales, all possessing strong Image-to-Video (I2V) capabilities essential for visual state continuation.

For the reasoning and evaluation modules, we employ advanced VLMs as the semantic planner and critic. The core experiments are driven by the Qwen family [3] (specifically Qwen3.6-plus via API). To demonstrate the robustness of our framework across different reasoning capacities, we also evaluate Cosmos-Reason2-8B, an open-weight, locally deployable multi-modal reasoner. All baseline video models are evaluated zero-shot under their default inference configurations. Unless

stated otherwise, Qwen3.6-plus serves as the planner, branch generator, and critic, and a separate frozen Qwen3.6-plus instance serves as the benchmark evaluator; Qwen3.6-Plus is also used for the first annotation pass of the benchmark (Sec. 4.2).

4.2 Ambiguous-DROID: Ambiguous Instruction Benchmark

Existing embodied video datasets typically provide low-level trajectory annotations. To evaluate video world models on under-specified commands, we introduce the Ambiguous-DROID benchmark, curated from the robotic manipulation dataset DROID [12]. The original DROID task texts (e.g., "grasp the red cup and place it on the white coaster") almost fully specify the object, action, and destination, leaving little to disambiguate; we therefore remove them from model conditioning, retain them as aligned references, and condition generation on high-level intents (e.g., "tidy up the table"). Starting from single-arm episodes, a VLM (Qwen3.6-Plus) checks robot visibility and proposes a candidate ambiguous instruction, keeping only tasks that require multiple steps, admit at least two plausible completions, and include the original trajectory as one valid realization; human reviewers then inspect, edit, replace, or reject each proposal and select the conditioning frame. The final benchmark contains 179 episodes, each pairing one robot-visible conditioning frame with one reviewed ambiguous instruction. Construction details and the complete mapping to the original instructions are given in Appendix B.

We assess the generated video trajectories using our automated VLM evaluation framework across five distinct dimensions, which align with the practical requirements of embodied execution:

1. Task Success: This metric evaluates whether the final state of the video fulfills the underlying semantic intent of the ambiguous instruction.
2. Object-Action Alignment: This metric measures the correctness of the spatial and physical interactions between the agent and the intended objects.
3. Temporal-Causal Coherence: This metric assesses the logical sequencing of events, ensuring the absence of physics-defying jump cuts or object teleportation.
4. Visual Quality: This metric evaluates general visual fidelity, including texture consistency and the absence of severe distortions.
5. Embodiment Consistency: This metric indicates the ability to maintain the structural identity of the acting agent (e.g., preventing a robotic gripper from morphing into a human hand).

Each dimension is scored in $\{0, 0.5, 1\}$ (failure, partial, clear success) following the rubric in Appendix B, and the weighted total $S = 0.35S_{\text{task}} + 0.20S_{\text{object}} + 0.15S_{\text{temporal}} + 0.15S_{\text{visual}} + 0.15S_{\text{embodiment}}$ weights task execution above visual fidelity.

4.3 Experiment Results

Table 1: General video generation quality results on three backbones. Results are reported as mean $\pm$ standard deviation over three random runs. Higher values indicate better results.

Backbone	Method	Task Success	Obj.-Act. Align.	Temp.-Causal Coherence	Visual Quality	Embod. Consist.	Weighted Total
Cosmos-Predict-2.5-2B	Baseline	63.04 ± 1.12	56.52 ± 0.95	79.71 ± 1.24	94.93 ± 0.85	72.46 ± 1.34	70.43 ± 1.10
	AW	$\mathbf{92.03 \pm 0.45}$	$\mathbf{89.13 \pm 0.62}$	$\mathbf{99.28 \pm 0.31}$	$\mathbf{99.28 \pm 0.28}$	$\mathbf{98.55 \pm 0.41}$	$\mathbf{94.60 \pm 0.43}$
	Delta	+28.99	+32.61	+19.57	+4.35	+26.09	+24.17
Wan2.2-TI2V-5B	Baseline	74.24 ± 0.98	78.28 ± 1.05	87.37 ± 0.88	74.24 ± 1.15	9.09 ± 0.55	67.25 ± 0.94
	AW	$\mathbf{89.90 \pm 0.55}$	$\mathbf{81.82 \pm 0.74}$	$\mathbf{100.00 \pm 0.00}$	$\mathbf{98.48 \pm 0.42}$	$\mathbf{93.43 \pm 0.68}$	$\mathbf{91.62 \pm 0.51}$
	Delta	+15.66	+3.54	+12.63	+24.24	+84.34	+24.37
CogVideoX-1.5-5B	Baseline	62.03 ± 1.25	58.86 ± 1.30	82.28 ± 0.95	86.08 ± 1.02	35.44 ± 1.45	64.05 ± 1.21
	AW	$\mathbf{74.05 \pm 0.85}$	$\mathbf{67.72 \pm 0.92}$	$\mathbf{91.77 \pm 0.65}$	$\mathbf{96.84 \pm 0.54}$	$\mathbf{94.94 \pm 0.76}$	$\mathbf{81.99 \pm 0.77}$
	Delta	+12.02	+8.86	+9.49	+10.76	+59.50	+17.94

Table 1 presents the quantitative evaluation of our AmbiguousWorld (denoted as AW) framework against baseline open-loop generation across three foundational world models on the Ambiguous-DROID benchmark. Across all backbones, our inference-time reasoning approach yields substantial and consistent improvements in every evaluation dimension.

Overall, AmbiguousWorld increases the Weighted Total score by 24.17, 24.37, and 17.94 points for Cosmos-Predict-2.5-2B, Wan2.2-TI2V-5B, and CogVideoX-1.5-5B, respectively. These results indicate that stage-wise multi-branch rollout with visual feedback substantially mitigates the effect of instruction ambiguity. Dimension analysis reveals specific benefits:



Figure 3: AW better preserves the embodiment of the acting agent over time.

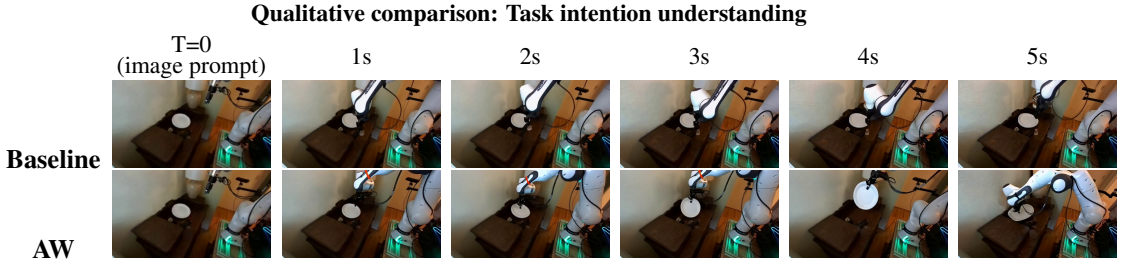


Figure 4: AW makes more effective progress toward the intended manipulation goal, while the baseline tends to produce ineffective or stalled motion.

Semantic Grounding and Execution. AmbiguousWorld significantly improves Task Success and Object-Action Alignment. For instance, on Cosmos-Predict-2.5-2B, Task Success jumps from 63.04% to 92.03% (+28.99%), and Object-Action Alignment improves by 32.61%. This suggests that the VLM Decomposer effectively parses ambiguous intents into actionable subgoals, allowing the world model to manipulate the correct objects.

Physical and Structural Consistency. Our stage-wise closed-loop evaluation enforces physical laws over long horizons. Temporal-Causal Coherence achieves high stability (e.g., 99.28% on Cosmos, 100.00% on Wan2.2). Embodiment Consistency improves substantially, increasing by 84.34% on Wan2.2 and 59.50% on CogVideoX. While open-loop baselines often encounter identity drift, our pruning mechanism rejects candidate branches containing unanchored structural changes.

First, the weighted total scores of the base models follow a ranking consistent with their performance on established world model benchmarks such as PAI-Bench [38], which is consistent with our VLM-based evaluation being a reasonable indicator of generative quality. Second, the strongest baseline backbone, Cosmos-Predict-2.5-2B, also yields the highest overall performance when integrated with AmbiguousWorld. Finally, some dimensions approach the ceiling for some backbone–method pairs (e.g., Temporal-Causal Coherence and Visual Quality), whereas Task Success with AmbiguousWorld (74.05–92.03) still leaves clear headroom; we therefore focus our analysis on the task-facing dimensions.

4.4 Case Study

To provide insights into the multi-branch reasoning pathways, we analyze specific video generation trajectories and the corresponding internal logic states of VLM. The visual outputs demonstrably showcase the ability of the proposed method to navigate branches of causal divergence effectively. We identify two recurring failure modes of the baseline. First, it often exhibits *subject inconsistency*: the generated actor may drift from a robotic arm to a human hand or another robot-like structure, consistent with its lower embodiment consistency in Table 1. Single-pass open-loop generation has no intermediate check on the acting subject, so identity errors accumulate once they emerge. AmbiguousWorld instead generates short stages and selects branches with better scene preservation and continuation quality, which reduces accumulated drift.

Second, the baseline often fails to follow the intended task, producing ineffective local motion or near-stationary behavior that contributes to its weaker task success in Table 1. We attribute this to the difficulty of solving goal inference, motion planning, and visual consistency in one monolithic generation pass. AmbiguousWorld mitigates this by converting the task into smaller subgoals and explicitly selecting branches that make visible progress toward the intended manipulation outcome.

4.5 Ablation and Comparative Studies

We conduct ablation studies to isolate the contributions of our design choices. First, multi-branch rollout outperforms single-path execution by mitigating error accumulation and trajectory drift across multi-stage video generation. Second, score-based branch selection outperforms a status-label heuristic.

4.5.1 Effect of Multi-Branch Rollout

Table 2: Ablation on multi-branch rollout for Cosmos-Predict-2.5-2B. Both settings share the same initial stage decomposition, so the comparison measures branching and pruning rather than repeated decomposition.

Block	Setting	Task Success	Obj.-Act. Align.	Temp.-Causal Coherence	Visual Quality	Embod. Consist.	Weighted Total
AW	Single-Path AW	89.50	87.00	97.50	**100.00**	**99.00**	93.20
	Multi-Branch AW	**92.03**	**89.13**	**99.28**	99.28	98.55	**94.60**
	Delta	+2.53	+2.13	+1.78	-0.72	-0.45	+1.40
Baseline	Single-Path Baseline	**65.50**	**61.50**	**83.50**	**100.00**	**85.50**	**75.57**
	Raw Cosmos Baseline	63.04	56.52	79.71	94.93	72.46	70.43
	Delta	+2.46	+4.98	+3.79	+5.07	+13.04	+5.14

The performance gains of the multi-branch AmbiguousWorld over the single-path AmbiguousWorld are primarily concentrated in the first three dimensions—task success, object-action alignment, and temporal-causal coherence—whereas the last two metrics, visual quality and embodiment consistency, exhibit only marginal fluctuations. AmbiguousWorld explores multiple candidate branches and selects better local continuations through branching and retry, this makes it more likely to choose actions that are applied to the correct objects and arranged in a more causally consistent order, and hence improve success rate. In our setting, the first three metrics are also the most diagnostic for embodied manipulation quality, since they directly measure whether the generated video performs the intended task correctly rather than merely appearing visually plausible.

4.5.2 Effect of Score-Based Branch Selection

Table 3: Ablation on branch selection strategy for Cosmos-Predict-2.5-2B.

Method	Task Success	Obj.-Act. Align.	Temp.-Causal Coherence	Visual Quality	Embod. Consist.	Weighted Total
Heuristic Branch Selection	86.44	84.75	98.31	99.15	98.31	91.57
Score-Based Branch Selection	**92.03**	**89.13**	**99.28**	**99.28**	**98.55**	**94.60**
Delta	+5.59	+4.38	+0.97	+0.13	+0.24	+3.03

We evaluate an ablated variant of AmbiguousWorld without the VLM-scoring component. In this version, branch selection relies only on coarse VLM status labels (`success`, `failed`, `uncompleted`): the search follows a `success` branch if present, or selects the first `uncompleted` branch otherwise. The full AmbiguousWorld ranks candidate branches using VLM evaluation scores, continuing with the highest-scoring one. This principled selection favors task progress, object-action alignment, and temporal-causal coherence, highlighting the advantage of score-based branch selection over a simple rule-based alternative.

4.5.3 Further Comparisons

Prompt extension and additional sampling. On Wan2.2-TI2V-5B (same 100 episodes), the strongest image-aware prompt-extension (PE) baseline reaches a Weighted Total of 87.05 versus 91.62 for

AmbiguousWorld (Task Success +9.60). Generating two PE trajectories and selecting one with an independent critic (compute-matched PE×2+critic) reaches 88.85 at 11.0 min/episode, still below AmbiguousWorld at 9.44 min/episode (H200). PE helps, but does not substitute for visual feedback and branch selection (Appendix C.4). **Reflection- and composition-style baselines.** Simplified reconstructions of RoG-style reflection [4] and RoboDreamer-style composition [39] reach 83.50 and 87.23 Weighted Total, respectively, versus 91.62 for AmbiguousWorld, although RoboDreamer-style is better on Visual Quality and Embodiment Consistency (Appendix C.5). **Other backbones and planners.** On the embodied world model GigaWorld-0 [8], AmbiguousWorld improves Weighted Total by 8.35 points (95% CI [4.53, 12.35]); on Genie Envisioner [16], whose native multi-view input could not be provided, the result is inconclusive (-0.08). Replacing Cosmos-Reason with Qwen3.6-plus as the planner (historical runs, Cosmos-Predict-2.5) improves Weighted Total by 2.53 points on 76 paired episodes, with a 95% CI that includes zero and other configuration differences (Appendix C.6). **Human evaluation and external benchmark.** In a five-annotator blind comparison, multi-branch generation improves human-rated Task Success from 44.0% to 75.0% over single-branch generation (Appendix C.7). On PBench [21] (466 of 1044 cases), whose prompts are detailed rather than ambiguous, AmbiguousWorld is 2.22 points below the baseline in Domain Average, indicating that its benefit is specific to under-specified instructions (Appendix C.8).

4.6 Human-VLM Alignment Study

To verify if VLM-based evaluation reflects human preferences, we conducted a blind user study. For each episode, annotators compared two randomized, blinded videos (Baseline vs. AW), scoring $s \in \{0, 0.5, 1\}$ across five dimensions: Task Success, Object-Action Alignment, Temporal-Causal Coherence, Visual Quality, and Embodiment Consistency. To evaluate baseline improvement, we converted human (H) and VLM (V) scores into binary preference signals:

$$y_H = \mathbb{1}[H_{\text{AW}} \geq H_{\text{baseline}}], \quad y_V = \mathbb{1}[V_{\text{AW}} \geq V_{\text{baseline}}].$$

Agreement is defined as $y_H = y_V$, calculated at the episode-dimension level. VLM evaluation strongly aligns with human judgment, achieving a micro-average agreement of **0.91** (455/500), with per-dimension agreement of 0.97 (Task Success), 0.94 (Object-Action Alignment), 0.90 (Temporal-Causal Coherence), 0.92 (Visual Quality), and 0.82 (Embodiment Consistency) (Table 6 in Appendix C.3.1). This supports our automated framework as a reliable proxy for human preference.

5 Conclusion and Limitations

We presented **AmbiguousWorld**, an inference-time reasoning framework for improving video world models under ambiguous instructions. Experiments across multiple video generation backbones show consistent improvements, especially in task completion and embodiment stability. We also introduced an ambiguous-instruction benchmark **Ambiguous-DROID** and a VLM-based evaluation protocol aligned with human judgments. These results suggest that structured inference-time reasoning is a practical way to make video world models more robust to realistic, underspecified human commands.

Limitations and broader impact. Our evidence is at the video level: we do not execute the generated futures on a real robot, which additionally requires action extraction and closed-loop control (a protocol based on object-centric flow [32] is outlined in Appendix D). Multi-branch rollout takes about 1.9× the time of the strongest prompt-extension baseline, and the hand-set critic weights and the reliance on VLMs for planning and evaluation may limit robustness. Some benchmark dimensions approach the ceiling, and DROID re-annotations cannot cover all real-world contexts, preferences, or physical edge cases. Ambiguity whose resolution depends on information outside the observation, such as user preferences, should be resolved through clarification rather than inference. Finally, controllable video generation could be misused to synthesize misleading action sequences; deployments should add uncertainty estimation, human clarification, and provenance safeguards.

Acknowledgments and Disclosure of Funding

This work is supported by the National Science and Technology Major Project (No. 2022ZD0114903) and the Natural Science Foundation of China (NSFC, No. 62476149). Sen Cui acknowledges the Science and Technology Project of Beijing Municipal Science & Technology Commission (Grant No. Z251100008125030) and the Shuimu Scholar Program at Tsinghua University.

References

[1] 1X World Model Team. 1X World Model: Evaluating bits, not atoms. `https://www.1x.tech/1x-world-model.pdf`, 2025.

[2] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, et al. Do as I can, not as I say: Grounding language in robotic affordances. In *Conference on Robot Learning*, 2022.

[3] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[4] X. Chi, C.-K. Fan, H. Zhang, X. Qi, R. Zhang, A. Chen, C.-M. Chan, W. Xue, Q. Liu, S. Zhang, and Y. Guo. Empowering world models with reflection for embodied video prediction. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 10383–10410, 2025.

[5] Y. Du, M. Yang, P. Florence, F. Xia, A. Wahid, B. Ichter, P. Sermanet, T. Yu, P. Abbeel, J. B. Tenenbaum, L. Kaelbling, et al. Video language planning. In *International Conference on Learning Representations*, 2024.

[6] X. Fu, X. Wang, X. Liu, J. Bai, R. Xu, P. Wan, D. Zhang, and D. Lin. Learning video generation for robotic manipulation with collaborative trajectory control. In *International Conference on Learning Representations*, 2026.

[7] R. Geirhos, Y. Li, T. Wiedemer, N. Kalibhat, Z. Wang, M. Malek, O. Tafjord, K. Swersky, B. Kim, and P. Jaini. Visual prompt engineering for video models. *arXiv preprint arXiv:2607.25537*, 2026.

[8] GigaWorld Team, A. Ye, B. Wang, C. Ni, G. Huang, G. Zhao, H. Li, J. Zhu, K. Li, M. Xu, Q. Deng, et al. GigaWorld-0: World models as data engine to empower embodied AI. *arXiv preprint arXiv:2511.19861*, 2025.

[9] Y. Guo, L. X. Shi, J. Chen, and C. Finn. Ctrl-World: A controllable generative world model for robot manipulation. In *International Conference on Learning Representations*, 2026.

[10] Y. Huang, Y. Zhang, Y. Wang, M. Zhang, L. Shi, H. Zeng, and Y. Fu. Distorted or fabricated? A survey on hallucination in video LLMs. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026.

[11] Z. Huang, N. Yu, G. Chen, H. Qiu, P. Debevec, and Z. Liu. VChain: Chain-of-visual-thought for reasoning in video generation. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026.

[12] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. DROID: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems*, 2024.

[13] W. Kong, Q. Tian, Z. Zhang, R. Min, Z. Dai, J. Zhou, J. Xiong, X. Li, B. Wu, J. Zhang, et al. HunyuanVideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

[14] S. Li, Q. Hao, Y. Shang, and Y. Li. KeyWorld: Key frame reasoning enables effective and efficient world models. *arXiv preprint arXiv:2509.21027*, 2025.

[15] Z. Li, X. Wu, G. Shi, Y. Qin, H. Du, F. Liu, T. Zhou, D. Manocha, and J. L. Boyd-Graber. VideoHallu: Evaluating and mitigating multi-modal hallucinations on synthetic video understanding. In *Advances in Neural Information Processing Systems*, 2025.

[16] Y. Liao, P. Zhou, S. Huang, D. Yang, S. Chen, Y. Jiang, Y. Hu, J. Cai, S. Liu, J. Luo, L. Chen, et al. Genie Envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025.

[17] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, volume 36, 2023.

[18] X. Liu, X. Li, D. Guo, S. Tan, H. Liu, and F. Sun. Embodied multi-agent task planning from ambiguous instruction. In *Robotics: Science and Systems*, 2022.

[19] C. Ma, K. Lu, G. Yang, J. Liu, S. Xu, B. Byrne, I. Havoutis, N. Trigoni, and A. Markham. CycleVLA: Proactive self-correcting vision-language-action models via subtask backtracking and minimum Bayes risk decoding. *arXiv preprint arXiv:2601.02295*, 2026.

[20] M. Ni, Y. Fan, L. Zhang, and W. Zuo. Visual-O1: Understanding ambiguous instructions via multi-modal multi-turn chain-of-thoughts reasoning. In *International Conference on Learning Representations*, 2025.

[21] NVIDIA. PBench: A physical AI benchmark for world models. `https://huggingface.co/datasets/nvidia/PBench`, 2025.

[22] NVIDIA, N. Agarwal, A. Ali, M. Bala, Y. Balaji, E. Barker, T. Cai, P. Chattopadhyay, Y. Chen, Y. Cui, et al. Cosmos world foundation model platform for physical AI. *arXiv preprint arXiv:2501.03575*, 2025.

[23] J. Park, S. Lim, J. Lee, S. Park, M. Chang, Y. Yu, and S. Choi. CLARA: Classifying and disambiguating user commands for reliable interactive robotic agents. *IEEE Robotics and Automation Letters*, 9(2):1059–1066, 2024.

[24] J. Quevedo, A. K. Sharma, Y. Sun, V. Suryavanshi, P. Liang, and S. Yang. WorldGym: World model as an environment for policy evaluation. *arXiv preprint arXiv:2506.00613*, 2025.

[25] R. Ramrakhya, M. Chang, X. Puig, R. Desai, Z. Kira, and R. Mottaghi. Grounding multimodal LLMs to embodied agents that ask for help with reinforcement learning. *arXiv preprint arXiv:2504.00907*, 2025.

[26] A. Z. Ren, A. Dixit, A. Bodrova, S. Singh, S. Tu, N. Brown, P. Xu, L. Takayama, F. Xia, J. Varley, et al. Robots that ask for help: Uncertainty alignment for large language model planners. In *Conference on Robot Learning*, 2023.

[27] Team Wan, A. Wang, B. Ai, B. Wen, C. Mao, C.-W. Xie, D. Chen, F. Yu, H. Zhao, J. Yang, J. Zeng, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[28] N. Wake, A. Kanehira, J. Takamatsu, K. Sasabuchi, and K. Ikeuchi. VLM-driven behavior tree for context-aware task planning. *arXiv preprint arXiv:2501.03968*, 2025.

[29] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

[30] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

[31] C. Wu, S. Yin, W. Qi, X. Wang, Z. Tang, and N. Duan. Visual ChatGPT: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023.

[32] M. Xu, Z. Xu, Y. Xu, C. Chi, G. Wetzstein, M. Veloso, and S. Song. Flow as the cross-domain manipulation interface. In *Conference on Robot Learning*, 2024.

[33] W. Xu, C. Wang, R. Fang, M. Zhang, L. Li, J. Xu, J. Gu, Z. Zeng, and R. Chen. Embodied Tree of Thoughts: Deliberate manipulation planning with embodied world model. *arXiv preprint arXiv:2512.08188*, 2025.

[34] Y. Yang, Y. Liao, J. Mei, B. Wang, X. Yang, L. Wen, J. Zhang, X. Li, L. Lv, H. Chen, B. Shi, et al. SPIRAL: Self-evolving action-conditioned video generation via reflective planning agents. *arXiv preprint arXiv:2603.08403*, 2026.

[35] Z. Yang, J. Teng, W. Zheng, M. Ding, S. Huang, J. Xu, Y. Yang, W. Hong, X. Zhang, G. Feng, et al. CogVideoX: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, 2025.

[36] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 11809–11822, 2023.

[37] Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn, et al. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1702–1713, 2025.

[38] F. Zhou, J. Huang, J. Li, D. Ramanan, and H. Shi. PAI-Bench: A comprehensive benchmark for physical AI. *arXiv preprint arXiv:2512.01989*, 2025.

[39] S. Zhou, Y. Du, J. Chen, Y. Li, D.-Y. Yeung, and C. Gan. RoboDreamer: Learning compositional world models for robot imagination. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 61885–61896, 2024.

This is the supplementary material for the paper *AmbiguousWorld*. We organize the content as follows:

A – Algorithm and Implementation Details

A.1 Decomposer Prompt

A.2 Generator Prompt

A.3 AW Branch-Selection Critic Prompt

A.4 Computational Resources and Runtime

B – Ambiguous-DROID Benchmark Details

B.1 Dataset Construction Pipeline

B.2 Annotation Guidelines

B.3 Annotation Examples

B.4 VLM Evaluation Framework

B.5 Relation to the Ambiguity Taxonomy

B.6 Comparison with the Original DROID Instructions

B.7 Benchmark Release

C – Additional Results

C.1 Extended Qualitative Results

C.2 Failure Mode Analysis

C.3 Consistency of the Automated Evaluation Framework

C.4 Prompt Extension and Compute-Matched Baselines

C.5 Reflection- and Composition-Style Baselines

C.6 Other Backbones and Planners

C.7 Human Evaluation of Multi- vs. Single-Branch Generation

C.8 Evaluation on PBench

C.9 Cross-Context Analysis of Identical Instructions

D – Limitations and Future Work

E – Asset Licenses and Terms of Use

A Algorithm and Implementation Details

This appendix gives the implementation details of **AmbiguousWorld** (AW), our inference-time reasoning pipeline for ambiguous video generation. Given an initial observation image and an ambiguous natural-language instruction, AW first decomposes the instruction into a sequence of executable atomic stages. For each stage, it samples multiple action branches and converts them into prompts for the video generation model. The generated branch videos are then evaluated by a VLM-based critic. The best branch is selected according to a weighted multi-dimensional score, and its last frame is used as the starting image for the next stage. The stage list is produced once by the decomposer; the stages are not re-decomposed after generation starts. In all reported runs, the branch budget is $K_0{=}2$, the number of stages is at most 4, the number of retries is $R{=}1$, and the number of continuations is $C{=}1$.

A.1 Decomposer Prompt

The decomposer takes the first-frame image and the raw instruction as input. Its role is to transform an ambiguous task, such as "Tidy up the table", into a short sequence of visually grounded manipulation

Algorithm 1 AmbiguousWorld

Input: Initial state $x = (u, o_0)$, context c_0, branch budget K_0, max retries R, max continuations C
Initialize: Trajectory $s_0 \leftarrow [x]$, video $\mathcal{Y} \leftarrow []$, $\mathcal{S} \leftarrow \textsc{DecomposeTask}(x, c_0)$
for $t = 1, \ldots, |\mathcal{S}|$ **do**
  for *retry* $= 1 \ldots R$ **do**
    $K_t \leftarrow \textsc{DecideBranchCount}(K_0, s_{t-1}, \mathcal{S}_t)$
    Sample thoughts $\{z_t^{(k)}\}_{k=1}^{K_t} \sim p_\phi^{\text{gen}}(\cdot \mid s_{t-1}, c_0, \mathcal{S}_t)$
    Render clips $v_t^k \leftarrow \mathcal{W}_\theta(s_{t-1}, z_t^{(k)})$ and evaluate utility V_t^k via $C_\psi, \forall k \in [1, K_t]$
    Continue (up to C times) any causally valid branch whose stage is uncompleted, and re-evaluate it ▷ Continuation
    if $\exists k$ satisfying success criteria **then**
      $k^\star \leftarrow \arg\max_k V_t^k$ ▷ Select optimal branch
      Append $v_t^{k^\star}$ to $\mathcal{Y}$; Update state s_t, observation o_t, and history $\mathcal{H}$
      break ▷ Stage successful, proceed to next stage
    end if
  end for
  if stage t incomplete **then break**
  end if ▷ Terminate generation early if all retries fail
end for
return $\textsc{Concatenate}(\mathcal{Y})$

stages. The prompt enforces a two-step reasoning structure: the VLM must first analyze visible objects and plausible target regions, and only then produce the stage decomposition.

The following prompt specifies the scene-analysis step, the JSON output schema, and the visual grounding constraints used by the decomposer.

Prompt of Decomposer

```
You are a professional video-generation task planner. The current task is to generate a video
     according to the user's instruction.

Task description: {prompt}
The task can be decomposed into at most {max_stages} stages.

Step 1: Scene analysis. Before decomposing the task, carefully observe the first-frame image and
     answer:
1. Which manipulable objects are visible? List their color, shape, and current position.
2. What is a reasonable target placement region? Use the following priority:
   - Prefer clearly visible containers, trays, or storage regions.
   - Otherwise use an obvious empty region or edge region on the same visible surface.
   - Do not use a target region that overlaps with the object's current position.
   - Do not use any invisible location.
3. Summarize the reasonable task goal in one sentence.

Step 2: Stage decomposition. Based on the scene analysis, return a JSON object:

{
  "scene_analysis": {
    "objects": ["object description and current position", "..."],
    "target_area": "specific visible target region",
    "task_summary": "one-sentence task goal"
  },
  "num_stages": 3,
  "stages": [
    {"stage_id": 1, "description": "detailed sub-task 1"},
    {"stage_id": 2, "description": "detailed sub-task 2"}
  ]
}

Stage rules:
1. Each stage must be one complete manipulation unit. The gripper must be separated from the target
     object at the end of the stage.
2. Each stage manipulates only one target object. Do not merge actions on two objects into one stage.
3. The placement target must come from the target_area in the scene analysis.
4. The start and end positions must be different. If an object is already in the target region, skip
     it.
5. Stages must be ordered chronologically.
```

```
6. A stage may contain high-level alternatives such as "A or B"; the downstream branch generator will
     choose one concrete option.

Visual grounding constraints:
- The first-frame image is the only scene truth.
- Do not invent invisible shelves, drawers, cabinets, trays, bins, or boxes.
- If an object or location is uncertain, use a conservative phrase such as "visible target object" or
     "visible empty region".
```

The example below illustrates the expected decomposition format for a high-level ambiguous instruction.

Few-shot Example of Decomposer

```
Input instruction:
Tidy up the table.

Possible output:
{
  "scene_analysis": {
    "objects": [
      "a white cloth lying on the center area of the table",
      "a small item near the right side of the table"
    ],
    "target_area": "a visible empty region near the left edge of the same table",
    "task_summary": "move misplaced tabletop objects to a visible empty region on the table"
  },
  "num_stages": 2,
  "stages": [
    {
      "stage_id": 1,
      "description": "Move the white cloth from the center area of the table to the visible empty
     region near the left edge, then release it and separate the gripper."
    },
    {
      "stage_id": 2,
      "description": "Move the small visible item from the right side of the table to the same visible
     empty region, then release it and separate the gripper."
    }
  ]
}
```

A.2 Generator Prompt

For each decomposed stage, the branch generator produces diverse action prompts for the video generation model. Each branch contains a short strategy summary and an English video prompt. Diversity is introduced through grasping location, approach angle, movement path, and placement position. The branch generator is not allowed to create diversity by changing the target object specified by the current stage.

The generator prompt constrains each branch to a concrete object, a deterministic action chain, and a model-ready video prompt.

Prompt of Generator

```
You are an expert AI video director and prompt engineer for image-to-video generation. Analyze the
     provided first-frame image and the user's raw intention, then produce optimized English prompts
     for the video generation model.

User input:
- Raw intention: {prompt}
- Current sub-task: {task_description}

Director rules:
1. The video_prompt must be one flowing paragraph in this order:
   [Camera declaration] + [specific subject] + [time-ordered actions] + [final physical state].
2. The first sentence must be exactly:
   "Fixed camera, no camera movement."
3. Do not describe static background, room layout, walls, lighting, windows, or furniture material.
```

```
4. The action must start from the layout visible in the first-frame image. Do not invent object
     placements.
5. The manipulator must be a robotic arm or parallel gripper, not a human hand.
6. Actions must be strictly chronological. Each sentence should describe one atomic action or one
     tightly coupled action pair.
7. Do not use centimeters, millimeters, inches, degrees, or pixel counts. Use qualitative terms such
     as slightly, fully, downward, horizontally, or toward the edge.
8. Do not use numbered lists, bullet lists, markdown headings, newline characters, or uncertain
     branches such as "either A or B".
9. The final physical state must explicitly state that the gripper is open and no longer in contact
     with the manipulated object.

Object-locking rule:
The current sub-task specifies the object to manipulate. Every branch must operate on the same target
     object. Branches may differ only in:
- grasping position: edge, center, or corner;
- movement path: direct movement or lift-then-translate;
- placement position: left, right, or center of the target region;
- approach angle: top-down or side approach.

Return only valid JSON:
{
  "branches": [
    {
      "direction": "one-sentence strategy summary",
      "video_prompt": "Fixed camera, no camera movement.  The robotic arm ..."
    }
  ]
}
```

The example below shows two branch prompts that differ in grasping and motion strategy while preserving the same target object.

Few-shot Example of Generator

```
Original instruction:
Tidy up the table.

Current sub-task:
Move the white cloth from the center area of the table to the visible empty region near the left edge,
     then release it and separate the gripper.

Possible branches:
{
  "branches": [
    {
      "direction": "Grasp the edge of the white cloth, lift it first, and translate it to the left
     empty region.",
      "video_prompt": "Fixed camera, no camera movement.  The robotic arm moves toward the edge of the
     white cloth on the table.  The gripper closes around the cloth edge and grasps it securely.  The
     arm lifts the cloth from the table surface.  The arm moves horizontally toward the visible empty
     region near the left edge of the table.  The gripper releases the cloth onto the empty region.
     The gripper opens fully and moves away from the cloth, with no contact remaining."
    },
    {
      "direction": "Approach from above, grasp the center of the white cloth, and move it along a
     direct path to the left empty region.",
      "video_prompt": "Fixed camera, no camera movement.  The robotic arm approaches the center of the
     white cloth from above.  The gripper closes around the cloth and holds it securely.  The arm
     moves the cloth along a direct path toward the visible empty region near the left edge of the
     table.  The gripper lowers the cloth onto the empty region.  The gripper opens fully and moves
     away from the cloth, with no contact remaining."
    }
  ]
}
```

A.3 AW Branch-Selection Critic Prompt

This critic is used during the AW generation process to score candidate branch videos at the current stage and select the best continuation for the next stage. It first extracts a compact visual summary from sampled frames, and then scores each branch along five dimensions. The branch-selection score is a weighted average:

$$S = 0.30S_{\text{stage}} + 0.20S_{\text{target}} + 0.15S_{\text{motion}} + 0.10S_{\text{scene}} + 0.25S_{\text{continue}}.$$

Here, each dimension is scored on a 0–4 scale. The branch with the highest internal score is selected for continuation. Severe failure tags, such as wrong target, direction mismatch, object disappearance, or physically implausible interaction, can force a branch to be marked as failed even when some individual scores are non-zero. The weights are fixed, manually specified design choices rather than learned parameters or values fitted to the benchmark labels. We began from equal weighting and prioritized the dimensions most consequential for multi-stage manipulation: stage completion receives the largest weight because a visually clean clip that fails the current atomic action should not advance; continuation readiness is second because the selected final frame becomes the next stage's initial state; target grounding ensures that progress concerns the correct object; motion effectiveness distinguishes causal state change from superficial motion; and scene preservation receives the smallest soft weight because moderate appearance artifacts are less consequential than executing the wrong action. We do not claim that these coefficients are optimal.

The AW critic prompt first asks for a compact visible-fact summary and then scores each branch across five dimensions.

Prompt of AW Critic

```
You evaluate a short robot-manipulation branch video for one atomic action. The input includes the
     current stage description, the branch prompt, and ordered visual evidence. Use visible evidence
     only.

First, summarize visible facts:
Return exactly one JSON object:
{
  "summary": "short visible fact",
  "target_object": "short object name",
  "final_state": "short final state",
  "progress_level": "none|partial|mostly_complete|complete|unclear",
  "anomalies": [],
  "confidence": "low|medium|high"
}

Then evaluate each dimension independently:
1. stage_completion: whether the intended atomic action is visibly completed.
   0 = no relevant progress; 1 = slight relevant motion; 2 = partial action; 3 = mostly completed; 4 =
     clearly completed in the ending frames.
2. target_grounding: whether the robot acts on the correct target object with the correct local
     interaction direction.
   0 = wrong target or wrong direction; 4 = target and interaction are directly visible and
     unambiguous.
3. motion_effectiveness: whether the robot motion causes the intended visible state change.
   0 = no effective change; 4 = intended visible change is clearly caused and established.
4. scene_preservation: whether the scene remains coherent and artifact-free.
   0 = severe corruption or implausibility; 4 = stable and coherent throughout.
5. continuation_readiness: whether the ending frame is a usable start state for the next atomic action.
   0 = unusable ending; 4 = clearly usable ending state.

General judging rules:
- Visual evidence has highest priority. If the summary conflicts with frames, ignore the summary.
- Do not give 4 unless the claimed completion is directly visible in the ending frames.
- If evidence is weak, ambiguous, or mainly inferred from the prompt, keep the score at 2 or below.
- Evidence must be concrete visible facts, not a paraphrase of the task.

Allowed failure tags:
["no_progress", "wrong_target", "direction_mismatch", "insufficient_motion", "scene_drift",
     "object_disappearance", "object_duplication", "actor_inconsistent_with_input",
     "physically_implausible_interaction", "severe_blur", "unclear_final_state"]

Return raw JSON for each dimension:
{
  "score": 0,
  "evidence": "short visual evidence",
  "failure_tags": [],
  "uncertainty": "low|medium|high",
  "notes": ""
}
```

A.4 Computational Resources and Runtime

We used a local video-generation and VLM planning stack. Experiments were run on NVIDIA RTX PRO 6000 GPUs with 96 GB memory and NVIDIA A800 GPUs with 80 GB memory. A single GPU is sufficient to run the full inference pipeline for one episode, including VLM planning/critique and video generation.

For VLM-based planning and critique, we used two implementations: a local Cosmos-Reason2-8B model and an API-based Qwen3.6-Plus model. For video generation, we evaluated Cosmos-Predict2.5-2B, Wan2.2-TI2V-5B, and CogVideoX-1.5-5B. We used the 81-frame setting of the Cosmos standard inference mode as the reference video length, and adjusted the frame settings of the other video-generation models accordingly to keep the generated clips comparable.

In the AW configuration used for the reported runs, the maximum number of stages was set to 4, the maximum number of branches per stage was set to 2, the maximum number of retries was set to 1, and the maximum number of continuations was set to 1. Thus, AW can explore multiple candidate clips for each stage, invoke the VLM critic to select or reject branches, and optionally retry or continue generation when the current stage is incomplete.

The runtime is dominated by video generation. Under our parameter setting, the average video-generation cost for one episode is about 40 minutes. The additional overhead comes from VLM planning, branch proposal, VLM-based video critique, retry/continuation decisions, and GPU cleanup between video-generation subprocesses. Since each episode is assigned to one GPU slot at a time, multiple episodes can be processed in parallel by increasing the number of available GPU slots.

Timing under a unified boundary. For the comparisons in Sec. 4.5.3 and Appendices C.4 and C.5, we additionally measured end-to-end runtime on NVIDIA H200 GPUs under a consistent timing boundary: on Wan2.2-TI2V-5B, AmbiguousWorld takes 9.44 min/episode, versus 4.92 for the strongest prompt-extension baseline, 10.04 for RoG-style, 4.60 for RoboDreamer-style, and 11.0 for Qwen3.6-Plus PE$\times$2+critic. These H200 measurements should not be compared with the approximately 40 min/episode reported above, which was measured on RTX PRO 6000 and A800 GPUs in a different environment. For the embodied-backbone runs (Appendix C.6), we also logged per-episode costs with every upstream call launched as a cold-start process: on GigaWorld-0 (one concurrent job), AmbiguousWorld issues 8.23 video-generator calls per episode versus 2.67 for the duration-matched baseline, with a mean wall time of 1094 s (851 s video generation, 55 s planner, 187 s critic) versus 278 s; on Genie Envisioner, it issues 6.71 versus 2.22 calls, with 363 s versus 62 s wall time. These cold-start ratios (about 3.9–5.9$\times$) upper-bound the overhead of a resident-worker deployment. The overhead could be reduced with adaptive branch budgets, confidence-based early stopping, model reuse, parallel candidate generation, and keyframe-based generation [14].

B Ambiguous-DROID Benchmark Details

This appendix describes the construction of **Ambiguous-DROID**, our benchmark for image-to-video and world-model generation under ambiguous robot-manipulation instructions. Starting from raw DROID trajectories, we select robot-visible manipulation clips and relabel their original low-level task descriptions into high-level instructions with multiple plausible futures. The resulting benchmark pairs an initial observation image with an ambiguous natural-language instruction, and evaluates whether a generated video realizes one reasonable interpretation while preserving physical and visual consistency.

B.1 Dataset Construction Pipeline

Raw DROID annotations are often concrete, low-level action descriptions, such as “Put the blue block in the green bowl” or “Pick the marker and put it in the plastic box”. Such instructions specify the target object, action, and destination almost completely. They are useful for imitation-style evaluation, but they leave little room for assessing whether a world model can reason about underspecified goals, object sets, placements, and action orders. We therefore construct Ambiguous-DROID by converting selected DROID trajectories into high-level ambiguous tasks.

For each candidate episode, we use the original DROID task_text, the first frame, and the midpoint frame from the same camera view. Before annotation, a VLM-based frame selector checks whether

the robot arm is visible in the first and midpoint frames. If neither frame shows the robot arm, the sample is skipped. If both frames contain the arm, the frame with more complete arm visibility, clearer gripper appearance, and less occlusion is preferred as the visual reference. This step ensures that the instruction is grounded in a feasible robot state and that the selected conditioning image is suitable for future rollout generation.

The first annotation pass is performed with Qwen3.6-Plus. The annotator receives the first frame, the midpoint frame, and the original task_text. It first decides whether the trajectory can be lifted into a high-level ambiguous task. A sample is considered labelable only if it satisfies all of the following conditions:

- the task can be abstracted into a high-level intent rather than a single atomic pick/place/open/close action;
- completing the instruction normally requires at least two sub-steps or a non-trivial intermediate process;
- the scene admits at least two plausible completions under the same instruction;
- the observed DROID trajectory can be interpreted as one plausible realization of the high-level task;
- the ambiguity comes from task underspecification, not from missing entities, vague language, or hallucinated context.

If these conditions are not met, the sample is marked as unlabelable and excluded from the automatically selected pool. If the sample is labelable, the model produces a scene summary, a trajectory summary, a high-level task, an ambiguous instruction, ambiguity-type labels, the subgoals realized by the original trajectory, and several plausible interpretations. One interpretation must correspond to the observed trajectory, while the others represent alternative futures that would also be reasonable under the same instruction.

After the automatic pass, we use a local human review interface to inspect the images, original instruction, parsed model annotation, and candidate ambiguous instruction. Annotators may edit or replace the instruction, select the exported conditioning image, and add or remove samples from the final set. Samples not covered by the automatic pass can also be manually labeled by directly browsing the raw manifest. The final benchmark entry is stored in a compact format:

Final Sample Format


```
{
  "id": "episode_000003",
  "image": "images_mid/observation.images.exterior_1_left/episode_000003.png",
  "instruction": "Put the items on the table into the bag."
}
```


B.2 Annotation Guidelines

The goal of annotation is to produce instructions that are natural for a human user, visually grounded in the scene, and ambiguous in a meaningful task-level sense. The original DROID instruction is used only as evidence about what happened in the trajectory; it is not treated as the target instruction. Annotators are explicitly instructed not to merely paraphrase a single low-level action by deleting object names or adding a generic verb.

We use four primary ambiguity dimensions:

- **Goal ambiguity**: the final goal state is underspecified. For example, “Organize the items on the table” may mean clearing the surface, grouping objects, preparing for another task, or placing objects into containers.
- **Object-set ambiguity**: the instruction does not fully specify which visible objects should be manipulated. For example, “Put the items on the table into the bag” leaves open whether all objects, only clutter, or only task-relevant objects should be moved.

- **Placement ambiguity**: the destination or spatial arrangement is underspecified. For example, "Tidy up the bed" does not determine whether clothes should be removed, folded, placed near the pillow, or moved to the side.
- **Ordering ambiguity**: the order of subtasks is underspecified. For example, "Clean up the table" may be realized by first moving papers, first grabbing a cloth, or first collecting small objects.

The implementation also records `preference_ambiguity` when the instruction explicitly involves subjective human standards, such as neatness, convenience, accessibility, or not blocking the workspace. In our main taxonomy, this can be treated as an auxiliary attribute or folded into goal/placement ambiguity depending on the analysis.

The following prompt is used for the first-pass DROID relabeling. It asks the VLM to judge labelability before producing the ambiguous instruction and its supporting evidence.

Prompt of Ambiguous-DROID Annotation

```
You are a robot manipulation data annotation assistant. Your task is to generate a
"reasonable but ambiguous" high-level instruction for a DROID trajectory sample,
suitable for a world-model / future-rollout benchmark.

Input:
1. visual information from the initial frame or key frames
2. the original text annotation

Frame usage rule:
- When the input contains both the first frame and the midpoint frame, prioritize
  the frame where the robot arm is more completely visible.
- If both frames show the robot arm, use both, but treat the more complete frame
  as the main evidence.
- If neither frame shows the robot arm, the sample is skipped before annotation.

Goal:
Generate a high-level ambiguous instruction such that:
- the instruction corresponds to a multi-step task;
- the current scene admits multiple reasonable completion strategies;
- the real trajectory is one plausible realization among them;
- the instruction is suitable as a condition for robot future video/world rollout.

Allowed ambiguity sources:
- underspecified final goal state;
- underspecified object set;
- underspecified placement or organization pattern;
- underspecified subtask order;
- underspecified preference or tidiness standard.

Disallowed ambiguity sources:
- missing necessary entities;
- purely vague utterances such as "handle this area";
- hallucinated objects, invisible storage areas, or unsupported background goals;
- a trivial paraphrase of the original single-step instruction.

First decide whether the sample is labelable:
- Is there more than one actionable object or more than one reasonable goal state?
- Can the trajectory be abstracted into a high-level task rather than a single action?
- Are there at least two plausible completion strategies?
- Can the real trajectory be viewed as one of these strategies?

If not labelable, output:
{
  "labelable": false,
  "reason": "..."
}

If labelable, output:
{
  "labelable": true,
  "scene_summary": "...",
  "trajectory_summary": "...",
  "why_original_text_is_not_used_as_instruction": "...",
  "high_level_task": "...",
  "ambiguous_instruction": "...",
  "ambiguity_type": [
    "goal_ambiguity",
```

```
    "object_set_ambiguity",
    "placement_ambiguity",
    "ordering_ambiguity"
  ],
  "why_multi_step": "...",
  "subgoals_of_real_trajectory": ["...", "..."],
  "plausible_interpretations": [
    {
      "id": "h1",
      "description": "...",
      "why_plausible": "...",
      "is_realized_by_video": true
    },
    {
      "id": "h2",
      "description": "...",
      "why_plausible": "...",
      "is_realized_by_video": false
    }
  ],
  "final_check": {
    "is_high_level": true,
    "is_multi_step": true,
    "has_multiple_valid_futures": true,
    "not_just_a_paraphrase_of_single_step_text": true
  }
}
```

B.3 Annotation Examples

Figure 5 shows representative Ambiguous-DROID annotation examples selected from the final benchmark set. Each row corresponds to one episode and contains the selected conditioning image, the original DROID instruction, the final ambiguous instruction used for generation, and the ambiguity dimensions.

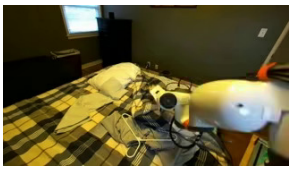
B.4 VLM Evaluation Framework

The benchmark evaluation framework measures the quality of generated videos under ambiguous instructions. We adopt a no-reference VLM evaluation setting: the evaluator observes sampled frames from the generated video together with the ambiguous instruction, extracts a compact evidence trace, and then assigns scores along five independent dimensions.

Before scoring, the evaluator summarizes the generated video into temporally ordered visual events. These events cover the beginning of the clip, the main object interactions, key state changes, and the final state. Each event records the visible subject, action, manipulated object, motion direction, state change, and supporting frame indices. The evaluator also records a final-state summary and the apparent acting embodiment in the first frame, during the main action, and at task completion. All judgments are required to rely on visible evidence; uncertain objects, actions, or actors are marked as unknown rather than inferred from the instruction.

Each dimension is scored using the discrete set $\{0, 0.5, 1\}$. The final reported score is a weighted average:

$$S = 0.35S_{\text{task}} + 0.20S_{\text{object}} + 0.15S_{\text{temporal}} + 0.15S_{\text{visual}} + 0.15S_{\text{embodiment}}.$$



Episode: `episode_000002`
Original: Move the bottom right tip of the duvet to the left.
Ambiguous instruction: Organize the clothes on the bed.
Ambiguity types: object-set ambiguity, placement ambiguity, ordering ambiguity.



Episode: `episode_000060`
Original: Remove the glass lid from the silver pot and put it on the blue towel on the table.
Ambiguous instruction: Get the pot ready for cooking.
Ambiguity types: goal ambiguity, placement ambiguity.



Episode: `episode_000250`
Original: Pick up the pot and put it on the stove.
Ambiguous instruction: Begin preparing popcorn on the stove.
Ambiguity types: ordering ambiguity, goal ambiguity.



Episode: `episode_001065`
Original: Put one white chess piece on the chessboard.
Ambiguous instruction: Set up the chess pieces for a game.
Ambiguity types: object-set ambiguity, placement ambiguity, goal ambiguity.



Episode: `episode_001056`
Original: Remove the sunglasses from the top shelf of the shoe rack and put it in the orange bowl on the bed.
Ambiguous instruction: Arrange the items on the table neatly.
Ambiguity types: goal ambiguity, object-set ambiguity, placement ambiguity.

Figure 5: Representative Ambiguous-DROID annotation examples from the final selected set. Each row shows one episode. The final instruction is intentionally high-level and underspecified while remaining grounded in the visible scene and the original DROID trajectory.

Table 4: Benchmark evaluation rubric for the five scoring dimensions. A score of 0 indicates failure, 0.5 indicates partial correctness, and 1 indicates clear success.

Dimension	0	0.5	1
Task Success	No relevant progress, an outcome inconsistent with the instruction, or an obviously wrong final state.	Meaningful progress is visible, but the final state is incomplete, weakly grounded, or ambiguous.	The final state clearly matches one plausible interpretation of the instruction and is supported by key intermediate events.
Object-Action Consistency	The main object is wrong, the principal action is wrong, or the dominant motion direction conflicts with the task.	Object-action relations are partially correct but contain noticeable local mismatches, missing interactions, or execution errors.	The main object, action type, and motion direction match the instruction semantics.
Temporal-Causal Consistency	Critical causal errors occur, such as teleportation, unexplained object appearance, reversed key order, or state change without visible cause.	The sequence is partly reasonable but contains a minor order ambiguity, weak transition, or causal uncertainty.	Events unfold in a clear, executable, and causally plausible temporal order.
Visual Quality	Frequent flicker, object collapse, severe misalignment, identity swap, heavy occlusion, or corruption makes the video unreliable.	Mild jitter or occasional artifacts are present, but the main interaction remains understandable.	The video is visually stable, geometrically consistent, and free of obvious flicker, ghosting, or severe blur.
Embodiment Consistency	The key action or task completion is performed by an actor inconsistent with the first-frame embodiment.	The actor is mostly consistent, but there is visible intervention, collaboration, takeover, or uncertainty.	The first-frame actor, main-action actor, and task-completion actor remain consistent throughout the video.

In addition to the numerical scores, the evaluator records short textual evidence, an uncertainty note, and failure tags when applicable. Common failure tags include ambiguous goal, wrong object, direction mismatch, order violation, physics violation, hallucination, flicker, identity swap, off-screen action, occlusion, no progress, and actor inconsistency with the input frame.

B.5 Relation to the Ambiguity Taxonomy

The five ambiguity dimensions in Sec. 1 describe how natural instructions are under-specified, whereas the annotation labels above (goal, object-set, placement, and ordering ambiguity) describe which part of a manipulation task is left open. High-level abstraction is a prerequisite for every benchmark sample, since each instruction must require at least two sub-steps. Placement and ordering ambiguity refine goal ambiguity at the level of spatial arrangement and subtask order, and object-set ambiguity includes ambiguous references to visible objects. Contextual ambiguity arises when the appropriate realization depends on the observed scene; we study it with the cross-context analysis in Appendix C.9.

B.6 Comparison with the Original DROID Instructions

The original DROID descriptions are removed only from model conditioning; they are not erased and remain aligned references. We matched all 179 Ambiguous-DROID episodes to the DROID 1.0.1 LeRobot metadata and recovered a non-empty original `tasks` entry for every episode (179/179). The original text specifies a demonstrated action and object, whereas Ambiguous-DROID tests whether a model can infer one plausible execution from an under-specified goal; using the original text as the prompt would expose exactly the operational choices being evaluated. Table 5 lists verified examples. The re-annotations retain the scene, workspace, and functional intent of the demonstration while leaving the object set, action sequence, or terminal state open. For example, putting the brown cup in the sink is one concrete step of getting ready to wash the dishes, and stacking the boxes is one way to arrange the items on the counter for efficient access. These pairs show different strengths of scene and action continuity; they are not paraphrases, and the recorded action is not the only valid future.

Table 5: Verified pairs of original DROID instructions and Ambiguous-DROID instructions.

Episode	Original DROID instruction	Ambiguous-DROID instruction
`episode_001561`	Open the microwave and put the bowl inside	Heat the food in the microwave.
`episode_000250`	Pick up the pot and put it on the stove	Begin preparing popcorn on the stove.
`episode_001272`	Pile the pair of jeans on the couch	Arrange the clothes and items on the couch neatly.
`episode_001309`	Open the cap from the spout of the blue kettle	Prepare the stove for cooking.
`episode_002984`	Put the brown cup in the sink.	Get ready to wash the dishes.
`episode_001022`	Stack the boxes on the counter.	Arrange the items on the counter for efficient access.
`episode_001251`	Put all the objects together on the right side of the table.	Arrange the tools on the table for upcoming use.
`episode_001640`	Pick up the marker and place it in the cup.	Make the table tidy and clear of clutter.
`episode_002807`	Pick up the white cloth and put it on the table.	Organize the items on the table neatly.

B.7 Benchmark Release

We release all 179 episodes together with the complete mapping from each episode to its original DROID instruction. The benchmark (conditioning images, ambiguous instructions, and original-instruction mapping) is available at `https://huggingface.co/datasets/3bai233/Ambiguous-DROID` under the CC BY 4.0 license, with attribution to DROID, and the code is available at `https://github.com/ambiguousworld-anon/ambiguousworld-code`.



Figure 6: Extended qualitative comparison in a kitchen environment. The task requires a temporally extended interaction sequence, including locating the bowl, lifting it, placing it into the microwave, and closing the microwave door. AW executes this multi-step manipulation process in a coherent and goal-consistent manner, ultimately completing the intended heating behavior. In contrast, the baseline remains largely ineffective throughout the rollout, exhibiting limited task progress and failing to realize the required interaction chain. This example highlights the stronger long-horizon interaction stability of AW in complex kitchen scenarios.

C Additional Results

This section provides supplementary figures, tables, and in-depth analyses that were omitted from the main text due to space constraints.

C.1 Extended Qualitative Results

We provide additional qualitative comparisons between the baseline and AW across three representative cases to further illustrate generalization under different physical environments, including kitchen, office, and device-interaction scenarios. Beyond visual plausibility, these examples highlight systematic differences in long-horizon task completion, multi-object coordination, and goal-directed control.

A key distinction emerges in environments that require temporally extended and semantically grounded interactions. In such cases, AW more reliably maintains coherent progress toward the intended goal, whereas the baseline frequently stalls, drifts without effective action, or exhibits behavior that remains weakly coupled to the task.

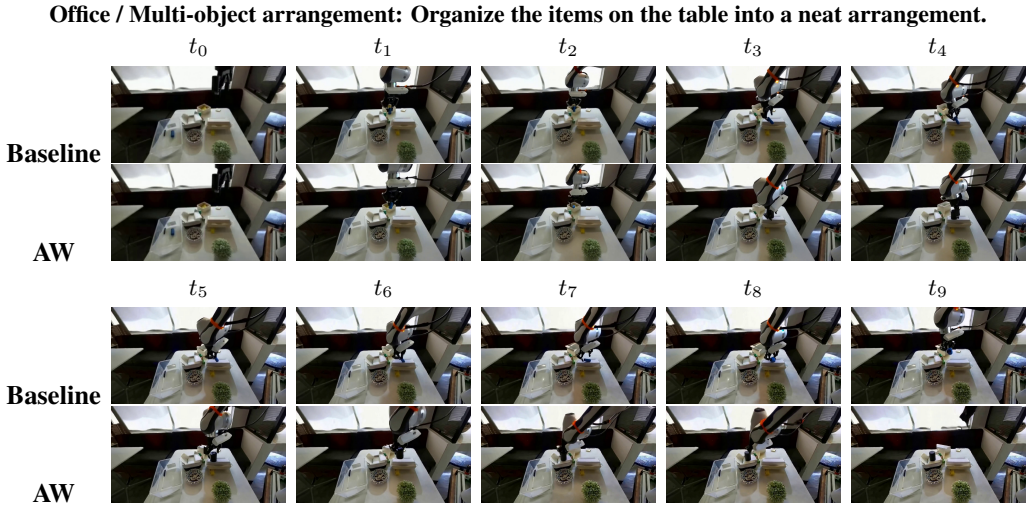


Figure 7: Extended qualitative comparison in an office-like tabletop environment with multiple manipulable objects. The task requires resolving object ambiguity and progressively arranging several items into a cleaner configuration. AW exhibits a more structured execution pattern, suggesting stronger decomposition of the overall objective into intermediate manipulation steps. By contrast, the baseline either stalls without effective interaction or produces weakly grounded motion that does not translate into meaningful rearrangement. This case illustrates the advantage of AW in cluttered multi-object scenes that demand more coordinated sequential behavior.

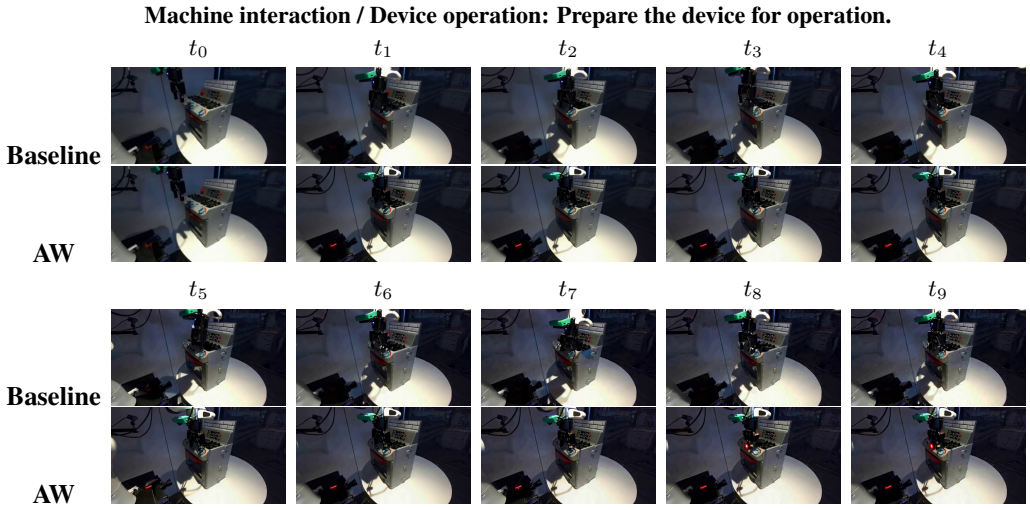


Figure 8: Extended qualitative comparison in a device-operation scenario. The task requires interacting with the machine interface so that the device begins operating, as indicated by the activation of its status light. AW produces a more targeted and semantically appropriate control sequence, successfully engaging the relevant button and triggering the machine. In contrast, the baseline fails to realize a clear control action and does not exhibit a coherent interaction pattern aligned with the operational objective. This example highlights the stronger goal-directed control behavior of AW in machine-centric interaction settings.

C.2 Failure Mode Analysis

we summarize the major failure modes observed in our system. Overall, these failures mainly stem from the limited capability of the VLM, especially in task decomposition, prompt expansion, and understanding highly cluttered scenes.

1. Limited VLM capability in task decomposition. A core assumption of our method is that a complex instruction can be decomposed into a sequence of atomic stages, where each stage corresponds to a simple and executable subtask for video generation, such that each of the stage can be completed as expected. In practice, however, the decomposer does not always produce truly atomic tasks. Although our prompts explicitly encourage the VLM to break down the original instruction into the smallest possible single-step actions, the resulting decomposition can still remain too coarse. As a result, a single stage may implicitly contain multiple coupled actions, making it difficult for the video generator to execute the intended behavior reliably within one stage, resulting in stage failure.

Example 1

Stage descriptions

- *Stage 1*: "The gripper moves to the white rectangular object and grasps it."
- *Stage 2*: "The gripper moves the white rectangular object to the designated area on the table and releases it."
- *Stage 3*: "The gripper moves to the metallic tool and grasps it."

Why it is non-atomic

- Stage 1 combines *approach* and *grasp*.
- Stage 2 combines *transport* and *release*.
- Stage 3 again combines *approach* and *grasp*.

Example 2

Stage descriptions

- *Stage 1*: "The gripper moves to the blue box and grasps it."
- *Stage 2*: "The gripper moves the blue box from the right edge of the desk to the central area."
- *Stage 3*: "The gripper releases the blue box in the center and returns to the initial position."

Why it is non-atomic

- Stage 1 combines *approach* and *grasp*.
- Stage 3 combines *release* and *return/retract*.

Example 3

Stage descriptions

- *Stage 1*: "The gripper moves to the black container, grasps it, and places it on the upper shelf."
- *Stage 2*: "The gripper moves to the green container, grasps it, and places it on the upper shelf."
- *Stage 3*: "The gripper moves to the white container, grasps it, and places it on the upper shelf."

Why it is non-atomic

- Each stage explicitly merges *move* + *grasp* + *place* into a single unit.
- This is a direct violation of the desired atomic-stage formulation.

2. Unintended action insertion during prompt expansion. To make the stage description usable for the video generation model, we need to expand it into a more detailed prompt with sufficient visual and temporal specificity. However, during this expansion process, the VLM may unintentionally introduce new actions that were not present in the original stage description. These added actions are often plausible from a language-modeling perspective, but they alter the intended semantics of the stage and can lead the video generation model to produce trajectories that deviate from the target behavior.

Example 1

Stage description
"The gripper moves the white rectangular object to the designated area on the table and releases it."

Expanded video prompt
"The gripper moves downward toward the edge of the white rectangular object. The gripper closes around the edge of the white rectangular object, securing it firmly. The gripper lifts the white rectangular object upward and moves horizontally toward the center of the target area. The gripper releases the white rectangular object, placing it in the center of the target area. The gripper opens fully and moves away from the object, with no contact remaining."

Why this indicates unintended action insertion
The original stage only specifies *transport* + *release*, but the expanded prompt additionally introduces *approach*, *grasp*, *lift*, and *retract*. These extra actions were not explicitly present in the stage description.

Example 2

Stage description
"The robotic arm moves the red-capped glass bottle from the right side of the stove to the burner."

Expanded video prompt
"The robotic arm moves toward the red-capped glass bottle on the right side of the electric stove. The gripper closes around the middle section of the bottle, securing it firmly. The arm lifts the bottle slightly and moves horizontally toward the right burner of the electric stove. The gripper releases the bottle, placing it on the right burner. The gripper opens fully and moves away from the bottle, with no contact remaining."

Why this indicates unintended action insertion
The original stage only states that the bottle should be moved to the burner. However, the expanded prompt injects a detailed manipulation sequence including *grasping*, *lifting*, and *post-release retraction*, which are not explicitly specified in the stage description.

Example 3

Stage description
"The gripper moves the blue box from the right edge of the desk to the central area."

Expanded video prompt
"The robotic arm moves toward the blue box on the desk. The gripper closes around the top edge of the blue box, securing it firmly. The arm lifts the blue box slightly upward and moves horizontally toward the center of the desk. The gripper releases the blue box, placing it centrally on the desk surface. The gripper opens fully and moves away from the blue box, with no contact remaining."

Why this indicates unintended action insertion
The stage description only refers to *moving* the blue box to the center. During prompt expansion, the VLM adds a much richer action chain including *grasp*, *lift*, *release*, and *move-away*, thereby changing the operational structure of the stage.

Example 4

Stage description
"The gripper moves into the oven, grasps the blue bowl, and transfers it to the wooden tray above the microwave."

Expanded video prompt
"The robotic arm moves toward the blue bowl inside the oven. The gripper closes around the edge of the blue bowl, securing it firmly. The arm lifts the blue bowl vertically and moves horizontally toward the wooden tray above the microwave. The gripper opens, releasing the blue bowl onto the central position of the wooden tray. The gripper fully opens and moves away from the bowl, ensuring no contact remains."

Why this indicates unintended action insertion
Although the original stage is already complex, the prompt expansion still adds extra low-level actions such as *edge-specific grasping*, *vertical lifting*, *precise central placement*, and *explicit no-contact retraction*. These details were not explicitly requested by the decomposed stage.

3. Limited understanding of complex scenes. Another major source of failure comes from limited scene understanding in visually complex environments. The DROID dataset is inherently challenging: many scenes contain multiple objects, cluttered layouts, and ambiguous action targets. In such cases, the VLM may fail to correctly identify the primary object to manipulate or the actual task-relevant region in the scene. When the scene is overly crowded, the model can easily become confused about what should be acted upon, which in turn leads to incorrect decomposition, inappropriate prompt expansion, or entirely off-target video generation.

C.3 Consistency of the Automated Evaluation Framework

Table 6: Agreement rates between VLM evaluation and human preferences across five dimensions.

Dimension	Task Success	Obj.-Act. Align.	Temp.-Causal Coherence	Visual Quality	Embod. Consist.	**Micro Average**
Agreement Rate (VLM / Human)	0.97	0.94	0.90	0.92	0.82	**0.91**

C.3.1 Blind Human Evaluation Setup

Figure 9 shows the interface used in our human evaluation study. For each episode, raters were presented with two candidate videos together with the task prompt. The two videos were anonymized and displayed only as Video A and Video B. The mapping from method identity to displayed label was hidden from raters and randomized across episodes. Both videos were shown with the same layout and playback controls to ensure a symmetric comparison setting.

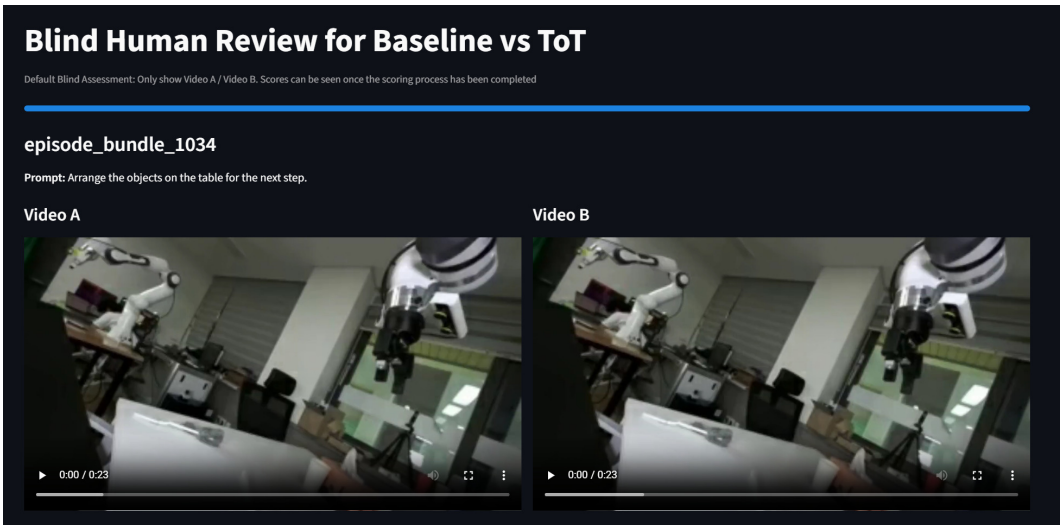


Figure 9: Interface used in the blind human evaluation study. Raters compare two anonymized videos conditioned on the same task prompt.

C.3.2 Scoring Procedure

Raters independently evaluate each video along the same five dimensions used by the benchmark evaluator: task success, object-action consistency, temporal-causal consistency, visual quality, and embodiment consistency. The human scoring protocol is aligned with the rubric in Table 4; namely, each dimension uses the same discrete scale $\{0, 0.5, 1\}$, where 0 denotes failure, 0.5 denotes partial correctness, and 1 denotes clear success. After assigning per-dimension scores, raters also provide an overall pairwise preference label chosen from **A**, **B**, **Tie**, and **Both Bad**, and may optionally record failure reasons and free-form notes.

C.3.3 Fairness and Reliability Considerations

The protocol was designed to improve fairness and reliability in three ways. First, blind presentation prevents raters from being influenced by method identity. Second, randomized A/B assignment and the symmetric interface reduce positional and presentation bias. Third, a standardized instruction manual specifies the criteria for assigning partial and full credit in each dimension, reducing ambiguity in the annotation process. Together, these choices provide a controlled setting for assessing the consistency between automated evaluation outcomes and human judgments.

C.4 Prompt Extension and Compute-Matched Baselines

We compare AmbiguousWorld with image-aware prompt extension (PE), in which a VLM rewrites the ambiguous instruction into a detailed prompt from the first frame, either once for the whole video (one-shot) or once per decomposed stage (per-stage). All methods use Wan2.2-TI2V-5B and the same 100 episodes; runtimes are measured on NVIDIA H200 GPUs under a consistent timing boundary (Table 7). The strongest PE setting is 1.51 points better than AmbiguousWorld on Object-Action Alignment, so PE is clearly useful, but AmbiguousWorld improves Task Success by 9.60, Temporal-Causal Coherence by 4.04, and Weighted Total by 4.57 points: PE elaborates a single continuation, whereas AmbiguousWorld can reject an interpretation after observing its generated outcome.

Table 7: Comparison with image-aware prompt extension on Wan2.2-TI2V-5B (100 episodes). Runtime in H200 minutes per episode.

Method	Task Success	Obj.-Act. Align.	Temp.-Causal Coherence	Visual Quality	Embod. Consist.	Weighted Total	min/ep.
Qwen3.6-Plus per-stage PE	80.30	83.33	95.96	94.44	91.41	87.05	4.92
Qwen3.6-Plus one-shot PE	77.27	79.80	95.45	97.98	92.93	85.96	5.26
Qwen-VL-Max per-stage PE	72.22	71.72	87.88	77.78	48.99	71.82	**4.44**
Qwen-VL-Max one-shot PE	83.84	**83.84**	90.40	74.75	44.95	77.63	5.60
AmbiguousWorld	**89.90**	81.82	**100.00**	**98.48**	**93.43**	**91.62**	9.44

Compute-matched comparison. To test whether the gain comes from additional sampling alone, each per-stage PE baseline generates two complete trajectories with the same PE procedure and Wan2.2-TI2V-5B generator: Branch 0 reuses the single-branch PE output, and Branch 1 repeats the procedure with a paired seed offset of $+10{,}000$, so the seed is the only difference. An independent VLM critic receives only the ambiguous instruction, the initial image, and two anonymous chronological contact sheets (no PE prompts or branch identities), assesses task completion, target grounding, motion effectiveness, scene preservation, and instruction faithfulness, and selects one branch (Critic Best-of-2). We compare it with each branch, a fixed random choice between the two branches (Random-2), and an evaluator-selected Oracle Best-of-2, which is an analysis upper bound rather than a deployable method (Table 8). Paired counts exclude only records rejected by the API safety layer of the evaluator or critic (three in total); no score was imputed. Critic Best-of-2 improves over Random-2 by 3.39 (Qwen3.6-Plus) and 2.43 (Qwen-VL-Max) Weighted Total points, with critic–oracle agreement of 64.29% and 55.56% and mean selection regret of 4.54 and 4.72 points. Even so, it remains 2.77 points below AmbiguousWorld (91.62, $N{=}99$) while taking 11.0 versus 9.44 minutes per episode (two generations at 292 s each plus a 77 s critic call). Informed selection over realized outcomes therefore helps, but selecting among complete trajectories after the fact does not match stage-wise branching with visual feedback.

Table 8: Compute-matched two-branch prompt extension on Wan2.2-TI2V-5B (per-stage PE). Oracle Best-of-2 is an analysis upper bound only.

PE planner	Selection	Task Success	Obj.-Act. Align.	Temp.-Causal Coherence	Visual Quality	Embod. Consist.	Weighted Total	N
Qwen3.6-Plus	Branch 0	80.10	83.16	95.92	94.39	91.33	86.91	98
	Branch 1	78.57	79.08	95.41	95.92	84.69	84.72	98
	Random-2	79.59	79.08	95.41	94.90	88.27	85.46	98
	Critic Best-of-2	**84.18**	**84.18**	**96.94**	**95.92**	90.82	**88.85**	98
	Oracle Best-of-2	91.33	91.33	98.47	97.96	91.33	93.39	98
Qwen-VL-Max	Branch 0	71.72	71.21	87.88	78.79	48.99	71.69	99
	Branch 1	77.27	78.28	89.90	75.25	41.92	73.76	99
	Random-2	74.75	75.25	90.40	76.26	48.48	73.48	99
	Critic Best-of-2	**79.80**	**82.32**	**90.40**	74.24	45.45	**75.91**	99
	Oracle Best-of-2	86.36	85.35	93.43	81.82	46.97	80.63	99
AmbiguousWorld (reference)		89.90	81.82	100.00	98.48	93.43	91.62	99

C.5 Reflection- and Composition-Style Baselines

We compare against simplified RoG-style [4] and RoboDreamer-style [39] procedures on the same 100 episodes and Wan2.2-TI2V-5B backbone (Table 9). Because complete runnable implementations of the required pipelines were unavailable, these are simplified reconstructions from the published descriptions, not claims about the official systems. AmbiguousWorld leads RoboDreamer-style by 7.40 Task Success, 8.32 Object-Action Alignment, and 5.00 Temporal-Causal Coherence points, although RoboDreamer-style is better on Visual Quality and Embodiment Consistency; AmbiguousWorld leads RoG-style by 8.12 Weighted Total points while running 0.60 minutes faster per episode. Our RoG-style procedure refines a single predicted trajectory through VLM-guided reflection, but does not compare competing realized interpretations. Our RoboDreamer-style procedure follows its text

parser, which maps an instruction into verb- and relation-phrase primitives and composes them; it uses no goal images or sketches. An ambiguous command may admit a valid syntactic parse, but syntax alone does not determine a unique executable decomposition or recover omitted goals and steps, which is consistent with RoboDreamer-style being visually coherent (highest Visual Quality and Embodiment Consistency) while scoring lower on Task Success, Object-Action Alignment, and Temporal-Causal Coherence. Because these are simplified reimplementations, we treat this as descriptive evidence consistent with the mechanism difference, not a causal claim about the official systems.

Table 9: Comparison with simplified reflection- and composition-style procedures on Wan2.2-TI2V-5B (100 episodes). *Simplified reconstructions from the published descriptions. Runtime in H200 minutes per episode.

Method	Task Success	Obj.-Act. Align.	Temp.-Causal Coherence	Visual Quality	Embod. Consist.	Weighted Total	min/ep.
RoG-style*	77.50	77.50	94.50	94.50	83.50	83.50	10.04
RoboDreamer-style*	82.50	73.50	95.00	**99.00**	**97.00**	87.23	**4.60**
AmbiguousWorld	**89.90**	**81.82**	**100.00**	98.48	93.43	**91.62**	9.44

C.6 Other Backbones and Planners

Embodied world models. We apply AmbiguousWorld to GigaWorld-0 [8] and Genie Envisioner [16] with duration-matched baselines on the same frozen 100-episode set, 2-FPS sampling, seed 42, and the frozen Qwen3.6-plus evaluator (Table 10). When the internal critic does not accept any branch of a stage, the episode is not discarded; the evaluator scores the complete best-effort AmbiguousWorld trajectory. For GigaWorld-0, Weighted Total improves by 8.35 points (paired 95% bootstrap CI [4.53, 12.35]; 42 wins, 41 ties, 17 losses), mainly from Task Success (+11.00) and Object-Action Alignment (+13.50). Genie Envisioner natively requires four temporally aligned frames from each of three views, whereas our interface provides one single-view image, which we replicated across all 12 memory slots. Under this non-native input, Genie yields mixed changes and an essentially unchanged Weighted Total (-0.08; 95% CI $[-5.28, 5.13]$; 38/27/34); one episode was excluded because both of its videos were rejected by the evaluator's API safety layer after three retries. This result therefore neither demonstrates a gain nor tests Genie under its intended observation conditions. Both intervals use 10,000 episode-level bootstrap resamples.

Table 10: AmbiguousWorld on embodied world models. *Single-view input replicated across Genie Envisioner's multi-view memory slots.

Backbone	Condition	N	Task Success	Obj.-Act. Align.	Temp.-Causal Coherence	Visual Quality	Embod. Consist.	Weighted Total
GigaWorld-0	Baseline	100	82.00	81.00	95.00	96.00	89.50	86.98
	AW	100	**93.00**	**94.50**	**97.00**	**98.50**	**97.00**	**95.33**
	Delta		+11.00	+13.50	+2.00	+2.50	+7.50	+8.35
Genie Envisioner*	Baseline	99	**44.44**	43.94	**66.16**	95.45	91.41	**62.30**
	AW	99	41.92	**46.46**	62.63	**97.98**	**94.44**	62.22
	Delta		-2.53	+2.53	-3.54	+2.53	+3.03	-0.08

Planner VLM. We keep Cosmos-Predict-2.5 as the generator and the same frozen Qwen3.6-plus evaluator, and vary only the planning VLM; the planning and critic prompts are unchanged (Table 11). The paired comparison uses the 76 episodes with identical initial frames (all 176 videos of the common 88-episode pool were re-evaluated; 12 pairs with differing initial frames are excluded). Qwen3.6-plus improves Weighted Total by 2.53 points (paired 95% bootstrap CI $[-0.46, 5.53]$), Task Success by 6.58, and Object-Action Alignment by 2.63, while Temporal-Causal Coherence decreases by 1.97. On all 88 common episodes, including the 12 pairs with different initial frames, the difference is +1.05 (95% CI $[-2.24, 4.18]$). The planning and critic prompts were not re-tuned for Qwen, which may explain why some dimensions improve while others decline. Moreover, these are

recovered historical configurations rather than an isolated planner ablation: the Qwen configuration mostly used three-stage plans without dynamic stage updates or retries, whereas the Cosmos-Reason configuration mostly used single-stage plans with dynamic stage updates and, in 40 of 100 episodes, a retry. Because the interval includes zero and these settings are not controlled, we interpret this as evidence of planner portability and a promising task-facing trend rather than a conclusive comparison of VLMs.

Table 11: Planner comparison on Cosmos-Predict-2.5 (76 paired episodes).

Planner	Task Success	Obj.-Act. Align.	Temp.-Causal Coherence	Visual Quality	Embod. Consist.	Weighted Total
Cosmos-Reason	84.87	89.47	**100.00**	99.34	98.68	92.30
Qwen3.6-plus	**91.45**	**92.11**	98.03	99.34	98.68	**94.84**
Delta	+6.58	+2.63	−1.97	+0.00	+0.00	+2.53

C.7 Human Evaluation of Multi- vs. Single-Branch Generation

Five annotators, blind to method identity, scored paired multi-branch and single-branch outputs on $\{0, 0.5, 1\}$ using the protocol of Appendix C.3.1 (Table 12). The largest gain is in Task Success (+31.0 percentage points), directly reflecting better task completion.

Table 12: Blind human evaluation of multi-branch vs. single-branch generation (five annotators).

Dimension	Multi-branch	Single-branch	Delta
Task Success	**75.0%**	44.0%	+31.0 pp
Object-Action Alignment	74.5%	59.5%	+15.0 pp
Temporal-Causal Coherence	71.0%	68.0%	+3.0 pp
Visual Quality	85.5%	82.0%	+3.5 pp
Embodiment Consistency	73.0%	67.0%	+6.0 pp
Five-dimension mean	**75.8%**	64.1%	+11.7 pp

C.8 Evaluation on PBench

We additionally evaluate on PBench [21], using 466 of its 1044 cases, with Wan2.2-TI2V-5B as the baseline and Cosmos-Reason2-8B as the AmbiguousWorld planner (Table 13). All differences are small and negative (0.93–3.43 points). PBench scores real-world physical-video domains from detailed prompts (mean prompt length 1127.52, versus 43.74 for Ambiguous-DROID), whereas Ambiguous-DROID tests ambiguous, long-horizon embodied goals. PBench is therefore complementary rather than a direct ambiguity evaluation, and we do not claim gains on detailed-prompt benchmarks.

Table 13: PBench results (466 of 1044 cases). Domains: CS, AV, RO, IN, HU, and PH as defined by PBench.

Method	CS	AV	RO	IN	HU	PH	Domain Avg
Baseline	**94.04**	**67.38**	**79.80**	**88.01**	**80.62**	**92.59**	**83.74**
AmbiguousWorld	92.28	66.15	77.13	84.58	79.70	89.32	81.52
Delta	−1.76	−1.23	−2.68	−3.43	−0.93	−3.27	−2.22

C.9 Cross-Context Analysis of Identical Instructions

To test whether AmbiguousWorld adapts to observable context rather than producing one fixed response per instruction, we used strict case-sensitive string matching over the 100 episodes processed by the complete pipeline to find instructions that occur in multiple distinct scenes. Eight instructions qualify, covering 30 episodes. This controls the language while varying the initial visual context. On this subset, AmbiguousWorld achieves a mean Weighted Total of 0.967 versus 0.813 for direct

generation (average paired improvement +0.154); episode-wise, it records 14 wins, 16 ties, and 0 losses, and six of the eight instruction groups improve while two tie. For example, the identical instruction "Arrange the items on the table for better access" occurs in four scenes with different objects and task-relevant regions; AmbiguousWorld scores 0.950 versus 0.500 (+0.450) and improves in all four episodes. This analysis addresses *observable* contextual ambiguity only: whether "cooking" means making tea or avoiding grease may depend on dialogue, preferences, or state outside the camera view, for which a robot should request clarification.

D Limitations and Future Work

While AmbiguousWorld effectively handles ambiguous instructions, we acknowledge several limitations. First, integrating VLM evaluation and multi-branch reasoning introduces additional computational overhead and inference time compared to single-pass generation. Future work will explore generating sparse keyframes instead of dense video sequences during search to mitigate this. Second, our framework relies on the underlying VLM and video generation models. Specifically, current VLMs exhibit limited diversity during candidate generation, restricting the exploration space. Enhancing reasoning and branch diversity via multi-agent frameworks represents a promising direction. Third, while stage-wise pruning mitigates structural drift in moderate horizons, it does not fully resolve temporal coherence issues for extremely long-horizon tasks. Keyframe-based generation could alleviate this by reducing the temporal accumulation of pixel-level errors. Finally, although Ambiguous-DROID highlights challenges in instruction ambiguity, its construction from DROID re-annotations may not comprehensively cover all real-world scenarios, user preferences, and physical edge cases.

From generated videos to robot behavior. Our results establish better generated video trajectories, not better downstream robot behavior; a generated video is not itself an executable control sequence. A controlled test of transfer could use an object-centric flow interface such as Im2Flow2Act [32]: (i) generate AmbiguousWorld and baseline videos from the same initial observation and instruction; (ii) extract object-centric flows using the same tracker and object mask; (iii) condition the same pretrained flow-conditioned policy on each flow; and (iv) compare physical task success over repeated trials with identical initial states. This interface requires embodiment-specific policy training and does not substitute for an experiment on our generated videos; robot execution with action success, intervention rate, and safety failures is a primary future evaluation.

Hidden context. AmbiguousWorld conditions its object set, target region, and stages on the first frame. It does not recover context that is absent from the input, such as private preferences, hidden goals, or state outside the camera view; in such cases, and in safety-critical settings, a robot should ask for clarification [23] rather than infer confidently.

Benchmark difficulty. Temporal-Causal Coherence and Visual Quality approach 99–100 for some backbone–method pairs, so these dimensions have limited resolution on the easiest episodes, while Task Success remains far from saturated. A harder successor benchmark should emphasize longer horizons, denser clutter, multiple plausible objects and targets, context-dependent ordering and goals, and physical edge cases, while retaining per-dimension rather than only aggregate reporting.

Post-training. Ambiguity-aware fine-tuning is likely to help when training data cover the relevant ambiguity patterns. Existing resources describe one realized episode per instruction (DROID [12]), use explicit goals (LIBERO [17]), or provide ambiguity labels without long-horizon videos (CLARA [23]); Ambiguous-DROID provides a common benchmark for developing and comparing post-training and inference-time disambiguation. Our training-free approach remains applicable to frozen or closed generators and to scene-specific ambiguities absent from training data, consistent with evidence that visual test-time reasoning can outperform repeated fixed-prompt sampling at matched cost [7].

E Asset Licenses and Terms of Use

To ensure the ethical and compliant use of external resources, we explicitly declare the licenses and terms of use for all datasets, models, and APIs utilized in this research. We confirm that the creators and original owners of these assets are properly credited, and we strictly adhere to their respective terms. The details are as follows:

- **DROID Dataset** [12]: The dataset is utilized under the Creative Commons Attribution 4.0 International License (CC BY 4.0). We respect the original authors' contribution to large-scale in-the-wild robot manipulation data.
- **Cosmos-Reason2-8B** and **Cosmos-Predict2.5-2B** [22]: The weights and architectures of these models are used in compliance with the NVIDIA Open Model License Agreement.
- **Wan2.2-TI2V-5B** [27]: The model weights and associated code are used under the Apache License 2.0.
- **CogVideoX-1.5-5B** [35]: The model is utilized under the CogVideoX LICENSE.
- **Qwen3.6-Plus API**: As this is a cloud-based service rather than an open-weight model, the API was accessed and utilized in strict compliance with the Alibaba Cloud Terms of Service. No proprietary model weights were downloaded or modified. The same applies to the Qwen-VL-Max API used in the prompt-extension baselines.
- **GigaWorld-0** [8]: The released model weights are used under the Apache License 2.0.
- **Genie Envisioner** [16]: The GE-base checkpoints are used under the license of LTX-Video-2B-v0.9 on which they are built, and the remaining code and data are used under CC BY-NC-SA 4.0, for non-commercial research only.
- **PBench** [21]: The benchmark is used under CC BY-NC 4.0, for non-commercial research and evaluation only.
- **Ambiguous-DROID (ours)**: Released under CC BY 4.0, with attribution to DROID; the original DROID data remain under their own CC BY 4.0 license.

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 paper introduces the AmbiguousWorld framework and the Ambiguous-DROID benchmark, with empirical results in Section 4 demonstrating improvements that align with the claims made in the abstract. The scope (visually inferable under-specification) is stated in Sections 1 and 3.1.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: We discuss limitations (no real-robot execution, inference cost, reliance on VLMs, benchmark ceiling effects, data coverage, and hidden context) in Section 5, with an extended discussion in Appendix D.

   Guidelines:

   - The answer [N/A] 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: [N/A]

Justification: This paper is primarily empirical and does not present formal theoretical theorems or proofs.

Guidelines:

- The answer [N/A] 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 main text section presents the algorithmic framework in Section 3 and the main experimental setup in Section 4. Specific implementation details for reproducing the main experimental results are provided in the appendix.

Guidelines:

- The answer [N/A] 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 release the code for the AmbiguousWorld framework and the Ambiguous-DROID benchmark annotations at https://github.com/ambiguousworld-anon/ambiguousworld-code.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 describes the main experimental setup, including all hyperparameters of AmbiguousWorld (Section 3.3); further details are given in Appendix A and in the released code.

   Guidelines:

   - The answer [N/A] 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: We report the standard deviation over three runs for our main results in Table 1, and paired bootstrap 95% confidence intervals (10,000 resamples) for the additional backbone and planner comparisons in Appendix C.6.

   Guidelines:

- The answer [N/A] 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: See Appendix A.4 for hardware (RTX PRO 6000, A800, and H200 GPUs) and per-episode runtime.

   Guidelines:

   - The answer [N/A] 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: After self-review, the research conducted in our paper complies with the NeurIPS Code of Ethics in all respects.

   Guidelines:

   - The answer [N/A] 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: We discuss positive and negative societal impacts in the limitations and broader impact paragraph of Section 5.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

Answer: [Yes]

Justification: While our work focuses on a reasoning framework, we discuss the potential misuse of video generation and suggested safeguards in Section 5.

Guidelines:

- The answer [N/A] 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 cite the DROID dataset (CC BY 4.0) and all models and benchmarks used (Cosmos-Predict2.5, Cosmos-Reason2, Wan2.2, CogVideoX, GigaWorld-0, Genie Envisioner, Qwen APIs, PBench), with license details in Appendix E.

Guidelines:

- The answer [N/A] 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: The Ambiguous-DROID benchmark annotations and the AmbiguousWorld implementation code are released at https://github.com/ambiguousworld-anon/ambiguousworld-code with a README, data format specification, and license (Appendix B.7).

    Guidelines:

    - The answer [N/A] 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: [Yes]

    Justification: We provide the full instructions, scoring rubrics, and screenshots of the evaluation interface used for the human studies (human–VLM agreement and multi- vs. single-branch comparison) in Appendices C.3.1 and C.7.

    Guidelines:

    - The answer [N/A] 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: [N/A]

Justification: Our human evaluation only involved researchers/volunteers assessing the visual quality and instruction alignment of synthetic videos. No sensitive personal data was collected, posing no risk to participants, which is exempt from formal IRB review under our institutional guidelines.

Guidelines:

- The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: Our core methodology relies on Vision-Language Models (VLMs) as semantic planners, branch generators, and critics, and a VLM is also used as the benchmark evaluator and for the first annotation pass of Ambiguous-DROID. The models and prompts are detailed in Sections 3 and 4 and Appendices A and B.

Guidelines:

- The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.