Video Models Reason Early: Exploiting Plan Commitment for Maze Solving

Kaleb Newman, Tyler Zhu, and Olga Russakovsky

Princeton University

Abstract. Video diffusion models exhibit emergent reasoning capabilities like solving mazes and puzzles, yet little is understood about *how* they reason during generation. We take a first step towards understanding this and study the internal planning dynamics of video models using 2D maze-solving as a controlled testbed. Our investigations reveal two findings. Our first finding is **early plan commitment**: video diffusion models commit to a high-level motion plan within the first few denoising steps, after which further denoising alters visual details but not the underlying trajectory. Our second finding is that **path length**, not obstacle density, is the dominant predictor of maze difficulty, with a sharp failure threshold at 12 steps. This means video models can only reason over long mazes by **chaining** together multiple sequential generations. To demonstrate the practical benefits of our findings, we introduce **Ch**aining with **Ea**rly **P**lanning, or ChEaP, which only spends compute on seeds with promising early plans and chains them together to tackle complex mazes. This improves accuracy from 7% to 67% on long-horizon mazes and by 2.5× overall on hard tasks in Frozen Lake and VR-Bench across Wan2.2-14B and HunyuanVideo-1.5. Our analysis reveals that current video models possess deeper reasoning capabilities than previously recognized, which can be elicited more reliably with better inference-time scaling.

Keywords: Video Models · Reasoning · Test-Time Inference

1 Introduction

Video generation models generate high-fidelity, temporally coherent videos that capture complex motion dynamics for creative applications or demonstrate intuitive world physics as synthetic data engines [1, 33, 36]. Recent works have discovered that these models exhibit emergent *general-purpose vision understanding*, from basic perception and manipulation to maze and symmetry solving [35,38]. Unlike vision-language models (VLMs), which map visual inputs to linguistic space, video models simulate reasoning directly in pixel space. This makes video models particularly fit for spatial tasks like maze solving, object tracking, and robot navigating, which require a type of spatial imagination that some refer to as *chain-of-frames* reasoning, i.e., using the frames as a visual scratchpad [35]. Yet despite growing interest in these capabilities, we lack a basic understanding of how such chain-of-frames reasoning *emerges* during generation and how reliably we can elicit these latent capabilities for solving reasoning tasks.

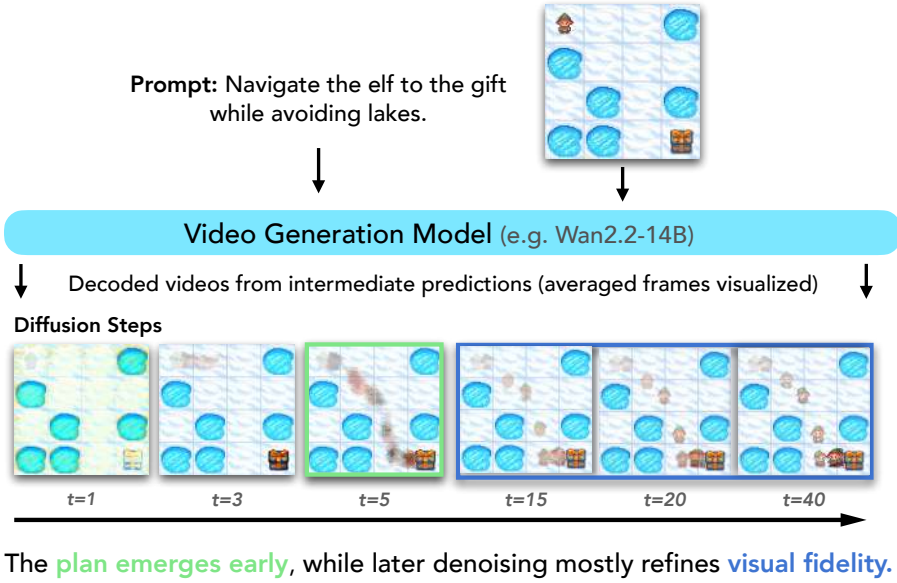

Fig. 1: Video diffusion models plan early. Decoded intermediate $\hat{x}_0$ predictions reveal that the model commits to a trajectory within the first few denoising steps (green box); later steps refine visual details but rarely alter the path (blue box).

Studying reasoning in open-ended video tasks, however, is challenging. Video models fabricate arbitrary details to produce diverse outputs, making them difficult to control. Without a defined end goal, the outputs are even harder to verify automatically. A controlled setting is much more tractable.

Maze solving provides exactly this. Mazes have served as a canonical testbed for studying planning since Tolman's cognitive map theory [29], through model-based reinforcement learning (RL) [28], to modern deep RL benchmarks [7, 21]. They require sequential, constraint-satisfying action planning, and conditioning on an input frame holds the environment constant, separating reasoning failures from rendering failures. Ground-truth solutions exist via BFS, enabling automatic verification, and difficulty can be systematically varied through grid size, path length, and obstacle placement. Using mazes as a controlled testbed, we analyze the internal dynamics of video diffusion models during maze solving and uncover a phenomenon we call **early plan commitment** (Fig. 1). This means the model commits to a high-level motion plan within the first few denoising steps, which remains stable for the remainder of sampling.

This observation has immediate practical consequences. If the plan is visible early, then standard best-of-N sampling—which fully denoises every seed—wastes most of its compute polishing unsuccessful trajectories. Instead, compute should be spent *exploring more candidate plans* rather than refining each one. We apply this insight with *Early Planning Beam Search* (EPBS), which partially denoises a

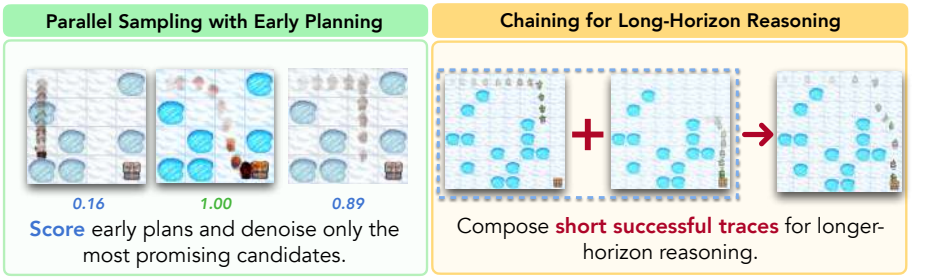

Fig. 2: Overview of ChEaP. (Left) *Early Planning Beam Search* scores early plans from partially denoised predictions and selects the most promising candidates for full generation. (Right) *Chaining* reconditions on the last frame of successful traces to extend reasoning beyond the single-generation horizon.

large pool of seeds, scores their early plans with a lightweight verifier, and reserves full denoising for only the most promising candidates. We find this strategy is successful up until a sharp failure cliff at 12-step trajectories, which are too long for video models to solve in a single video. This further motivates *chaining*: decomposing long-horizon tasks into shorter sub-problems to solve sequentially. Together, we call this **Ch**aining with **Ea**rly **P**lanning, or ChEaP (Fig. 2).

Evaluated on the Frozen Lake [30] and VR-Bench [38] datasets across the Wan2.2-14B [33] and HunyuanVideo-1.5 [36] video models, ChEaP matches best-of-N accuracy in 0.3× the diffusion steps, achieves up to 2.5× accuracy gains on hard tasks, and boosts long-horizon maze accuracy from 7% to 67%. Our analysis across two state-of-the-art models and over 480 mazes demonstrates the efficacy of ChEaP for enhancing maze solving capabilities, and that existing video models possess substantially deeper reasoning abilities than previously recognized. Our project page is at `video-maze-reasoning.github.io`.

2 Related Work

Visual reasoning in video models. Video diffusion models have recently demonstrated surprising emergent capabilities, solving mazes, puzzles, and physical reasoning tasks without task-specific training [35]. This has spurred new benchmarks and datasets for systematic evaluation [5, 34, 38], as well as calls for process-aware metrics that go beyond final-frame accuracy [16]. Yet zero-shot generation still fails under strict long-horizon constraints [32]. He *et al.* [11] address this by fine-tuning the model for native image-to-image reasoning, but this requires retraining. Rather than training the model, we show that stronger reasoning already exists within off-the-shelf video models and can be elicited through better *inference-time* compute allocation.

Phase transitions in diffusion models. Understanding *why* inference-time strategies work requires examining the internal dynamics of diffusion. It is

now well established that the reverse diffusion process exhibits a coarse-to-fine hierarchy: global semantic structure crystallizes in the earliest denoising steps, while later steps refine only low-level detail [2, 3, 8, 24, 25]. Empirically, cross-attention maps [12] and internal activations [14] confirm that spatial layout and semantic identity are fixed early, and stage-specialized denoisers exploit this by training separate experts per noise level [2]. Theoretically, this behavior has been formalized as sharp phase transitions [3, 24] with bounded critical windows [15], and studied through geometric [37], spectral [31], and information-theoretic [22] lenses. However, these analyses focus on image generation. We show that the same early-commitment behavior holds for *video* diffusion, and that it applies not just to appearance, but to the model's motion plan.

Inference-time scaling for diffusion models. Given this internal structure, a natural question is how to exploit it training-free. Inspired by the success of test-time compute scaling in language models [27], a growing line of work applies similar ideas to diffusion: searching over noise seeds with verifier feedback [13, 20], Feynman-Kac particle resampling [26], tree search with MCMC refinement [39], and noise-trajectory optimization for visual quality [18]. These methods improve output quality by generating and evaluating more candidates, but they treat the diffusion process as a black box, allocating compute uniformly across timesteps without exploiting *when* the model commits to its plan. Our EPBS exploits early plan commitment to prune unpromising seeds after only a few denoising steps, as we find that mid/late-stage refinement contributes little to reasoning diversity.

3 Mazes as a Controlled Testbed for Studying Reasoning

We focus on mazes as a controlled proxy for *action planning*—a setting that is verifiable, visually grounded, and sufficiently challenging to require multi-step decision-making. Crucially, conditioning on an input frame fixes the maze layout, so all task-relevant structure is concentrated in the agent's *motion trajectory*—making failures diagnostic and solutions automatically verifiable.

Research questions. Mazes are well positioned for studying the *internal dynamics* of generation, because trajectories can be extracted from *intermediate* denoising predictions and compared against exact solution structure. Recent work has established *that* video diffusion models can solve mazes [35, 38], and that sampling more candidates improves reliability by 10–20% [38], but these results describe only the *outputs* of the process. We do not know when the model commits to a plan during denoising, what structural properties of a maze make it difficult, or why brute-force sampling plateaus on hard instances. We ask:

1. **When does the model decide its answer when tasked with a reasoning problem?**
2. **Are there any signals early in the denoising process that are predictive of success?**
3. **What are the common failure modes and what structural properties of the task drive them?**

Experimental setup. To answer these questions with precision, we need control over our dataset. We evaluate on **Frozen Lake** mazes [30], which has an elf in the top left cell whose goal is to reach a gift in the bottom right while avoiding falling into frozen lakes along the way. We vary grid size (4×4 to 10×10), obstacle density (20–80%), and goal placement—distinguishing **norm** mazes (goal at the far corner, maximizing path length) from **vary** mazes (randomly placed goal, often admitting shorter solutions). We complement this with **VR-Bench** [38], which tests whether the same planning behaviors hold across different visual textures and constraint types (maze navigation and trap avoidance). Across these two benchmarks, we evaluate over 480 maze environments on two state-of-the-art video diffusion models: Wan2.2-14B [33] and HunyuanVideo-1.5 [36], using the standard image+text-to-video paradigm [4, 9]. We provide examples in Supp. D.

Evaluation. We evaluate *task success* rather than exact path match. Exact-match evaluation, as used in prior VR-Bench work [38], is overly strict: many mazes admit multiple valid solutions, and small trajectory-extraction errors can invalidate an otherwise correct video. Instead, we extract the agent trajectory from the generated video using SAM2 [23] and mark a sample as successful if the agent reaches the goal without violating task constraints, while rejecting degenerate cases such as goal drift. Full extraction details and success criteria are provided in Supp. A.1 and Supp. A.2.

4 Early Plan Commitment in Video Diffusion Models

Using mazes as a controlled testbed, we can analyze *how* solutions form during denoising. Because the answer to a maze is expressed primarily through the agent's motion path, we study how this trajectory evolves across intermediate $\hat{x}_0$ predictions. This analysis reveals a striking pattern: for the vast majority of seeds, the model commits to a coarse route within the first few denoising steps, and later computation primarily refines visual fidelity rather than changing the underlying plan. We refer to this phenomenon as ***early plan commitment***.

4.1 Flow matching

Both video models we study are built on *flow matching* [10, 17, 19], a generative framework that learns a velocity field transporting noise ϵ to data x_0 along straight paths. During training, an interpolant constructs noisy samples $x_t = (1-t)\, x_0 + t\, \epsilon$ for $t \in [0, 1]$ and $\epsilon \sim \mathcal{N}(0, I)$, and a network v_θ is trained to predict the velocity $\frac{dx_t}{dt} = \epsilon - x_0$. At inference, one integrates v_θ from $t = 1$ (pure noise) to $t = 0$ (clean video) using a discrete schedule of T steps.

Crucially, at any intermediate step t the velocity prediction can be rearranged to recover a *clean-sample estimate*:

$$\hat{x}_0^{(t)} = x_t - t \cdot v_\theta(x_t, t). \tag{1}$$

This $\hat{x}_0$ prediction is the model's current best guess of the final video given only the partially denoised state. Early in sampling, $\hat{x}_0^{(t)}$ is blurry and coarse, but as

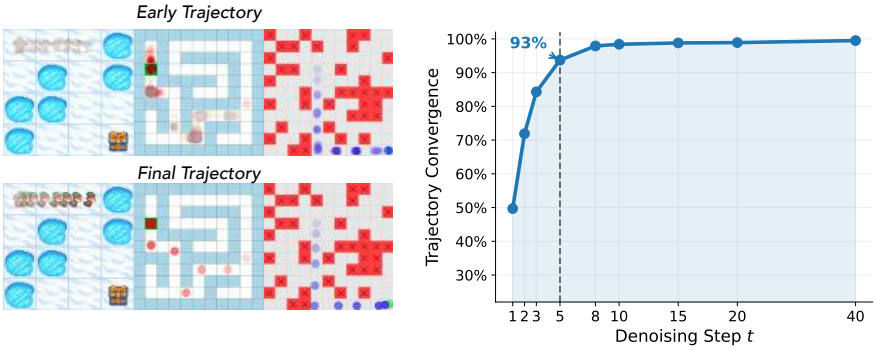

Fig. 3: Early plans stay consistent. (Left) Across multiple settings, the early trajectories emerging from decoded $\hat{x}_0$ predictions at step 5 match the final trajectory. (Right) Mean trajectory convergence throughout the denoising process. Step 5 already reaches 93%, *i.e.* trajectories stay converged (over 163 4×4 mazes).

$t \to 0$ it converges to the final output. We decode these intermediate predictions throughout denoising to study *when* the model's plan takes shape.

4.2 What is an early trajectory?

For a fixed random seed, let $\hat{x}_0^{(t)}$ denote the model's decoded prediction of the clean video at denoising step t into pixel space. We define an *early trajectory* $\mathcal{T}^{(t)} = (c_0^{(t)}, c_1^{(t)}, \dots)$ as the sequence of cells this intermediate prediction visits.

This is the natural object to study in mazes because the trajectory carries almost all of the task-relevant structure. The maze layout, obstacles, and goal are meant to be fixed according to the conditioning image and prompt; only the agent moves. Thus, to understand planning in this setting, we do not need every intermediate prediction to be visually sharp or pixel-accurate. We need only ask whether the model has already committed to the *route* it will ultimately follow.

4.3 Trajectories converge early during denoising

To quantify when trajectories converge, we calculate the *trajectory convergence* $\mathcal{C}$ of a step t prediction to the final one at time step T. We first extract a spatial *motion energy map* from each video: for every grid cell, we count the total number of pixels whose color deviates from the estimated background across all frames (with the goal cell masked to suppress its idle animation). This yields an $N \times N$ matrix $\mathbf{M}^{(t)}$ summarizing where motion occurs, with high values along the elf's path and near-zero values elsewhere. We then measure how well the intermediate energy pattern matches the final one via cosine similarity:

$$\mathcal{C}(\text{step } t) = \frac{\mathbf{m}^{(t)} \cdot \mathbf{m}^{(T)}}{\|\mathbf{m}^{(t)}\| \, \|\mathbf{m}^{(T)}\|},$$

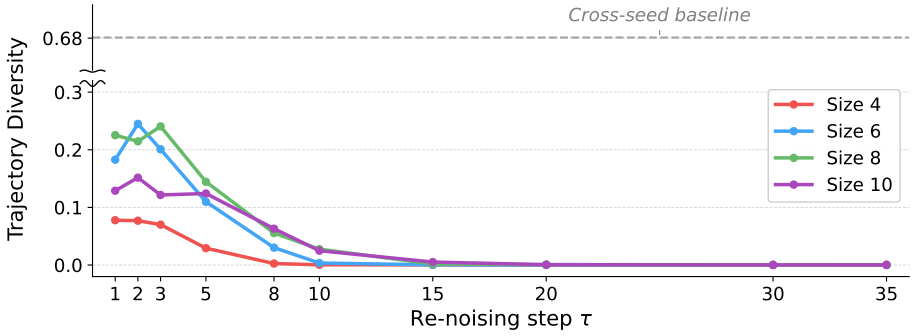

Fig. 4: Stepwise refinement. Mean pairwise trajectory IoU among K=5 re-noised completions at each step τ, across grid sizes 4–10. Even at τ=1, branch trajectories are far more similar to each other than trajectories from different seeds (dashed line), indicating that the route is largely encoded in the initial noise sample.

where $\mathbf{m}^{(t)}$ and $\mathbf{m}^{(T)}$ are the flattened energy vectors from the step-t prediction and the final video, respectively. A convergence of 1.0 indicates that motion energy is distributed identically across grid cells; 0.0 indicates no agreement. This metric is scale-invariant (robust to brightness differences between intermediate and final renderings) and requires no binarization threshold, avoiding the sensitivity to energy normalization that affects discrete cell-overlap measures on small grids (see supplement for a detailed comparison).

Figure 3 shows that trajectory structure emerges surprisingly early. On 4×4 Frozen Lake mazes, the step 5 trajectories for Wan2.2-14B are at 93% mean convergence. By step 10, convergence is nearly perfect. In other words, the model usually decides its route within the first quarter of denoising; the remaining steps primarily improve visual fidelity instead of altering the plan.

Because the metric operates on continuous energy values rather than binary cell sets, it is robust even on larger grids where background-difference extraction can introduce low-level noise in unvisited cells: such noise contributes zero to the numerator (since the reference energy is zero there) and only marginally affects the denominator. We find the same pattern holds across sizes and models in Supp. C.1, and we include visual examples of early plan commitment in Supp. D.1.

4.4 Early trajectories are diverse across seeds, not refinement

This observation suggests that reasoning-relevant structure is not uniformly distributed across denoising. A common technique for sample diversity in flow matching is *refinement*, where the step t noise is added back to $\hat{x}_0^{(t)}$ before continuing to denoise to explore other denoising paths. We suspect that trajectory diversity benefits more from *sampling different seeds*, not from refining an existing one. To test this, we perform a refinement ablation in which we renoise at a chosen step during the denoising process to sample a different trajectory (Fig. 4).

Algorithm 1 Early Planning Beam Search (EPBS)

Require: video model with denoising steps T, budget B, probe step τ, beam size K
Compute the number of initial candidates: $N = \left\lfloor \frac{B-KT}{\tau} \right\rfloor + K$
Sample N random seeds
for each seed **do**
    Partially denoise for τ steps
    Decode the intermediate $\hat{x}_0$ prediction
    Score the decoded prediction with the verifier
Select the top-K seeds under the verifier score
Fully denoise only these K candidates to $t = 0$
Return the highest-scoring final sample

We measure their diversity as $1 - \mathcal{C}(t', T)$ calculated between the new resulting trajectory from renoising at step t and the old final trajectory at step T.

We find that refinement branches from the same seed are nearly identical in trajectory with at most 25% trajectory diversity. Refining earlier in the denoising process or for larger mazes results in higher diversity, but none are as high as the 68% diversity between different seeds. **If early trajectories are predictive of final success, then inference-time scaling for reasoning should prioritize screening more candidate trajectories** rather than fully decoding every seed.

5 Trajectory Screening for Efficient Sampling

The plan commitment phenomenon suggests a straightforward strategy: instead of fully denoising every seed, screen trajectories from early timesteps and discard unpromising ones. We formalize this as *Early Planning Beam Search* (EPBS).

5.1 Early Planning Beam Search

EPBS reallocates inference-time compute from full denoising to early candidate exploration. Rather than fully denoising every seed, we first partially denoise many candidates for τ steps, score their intermediate $\hat{x}_0$ predictions with a lightweight verifier, and reserve full decoding only for the top-K seeds (Algorithm 1). Under a fixed number of function evaluations (NFEs), this allows EPBS to explore substantially more candidate trajectories than standard best-of-N sampling.

For Wan2.2-14B ($T = 40$), EPBS with $\tau = 5, K = 1$ evaluates 73 candidates at $B = 400$, compared to only 10 for best-of-N. The gain comes from terminating most seeds after only a small fraction of the denoising schedule, relying on planning commitment to ensure the τ-step trajectory predicts the final output. All that remains is to score the $\hat{x}_0$ predictions with a verifier.

Lightweight trajectory verifier. The verifier requires only minimal privileged information: the locations of the agent, the goal, and obstacle cells (lakes, traps, or walls depending on the environment). We argue that this is a reasonable setup; the goal is fully observable in a 2D setting, only the path to it must be discovered.

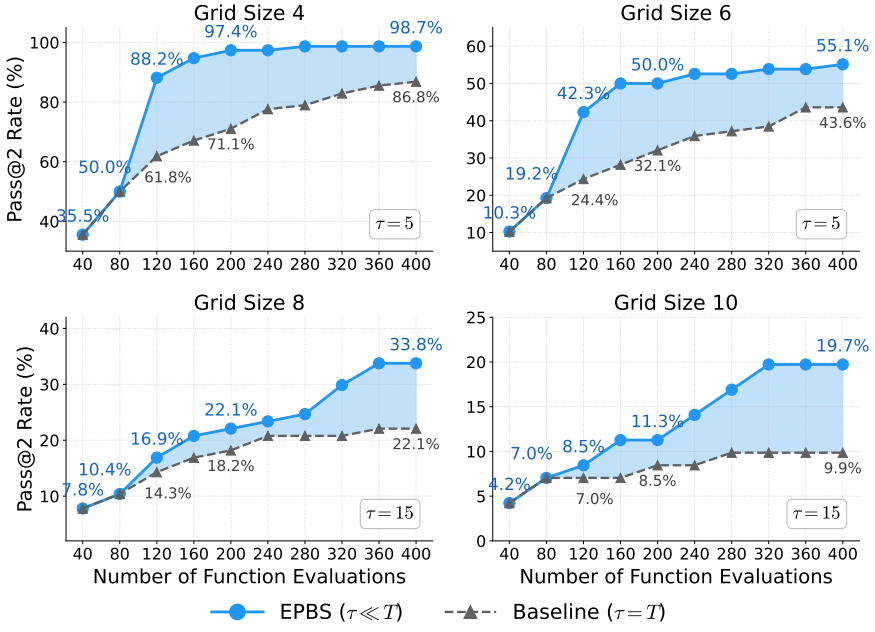

Fig. 5: EPBS finds solutions much more efficiently than best-of-N. Accuracy vs Function Evaluations (NFEs) on Frozen Lake mazes across four sizes with Wan2.2-14B. EPBS consistently dominates standard best-of-N, with large gains on larger mazes.

To score intermediate $\hat{x}_0$ predictions, we track the agent's position across frames and compute a confidence score that rewards goal progress while penalizing time spent in obstacle cells. Only the top-K seeds ranked by this score are fully decoded. Full verifier details and the scoring formula are provided in Supp. A.3.

5.2 Results

We compare EPBS to best-of-N sampling (equivalent to $\tau = T$) with $N = 1, \ldots, 10$. We use beam size $K = 2$ for both methods with the pass@K metric [6], following prior work [35], and test on Wan2.2-14B ($T = 40$) and HunyuanVideo-1.5 Step-Distilled ($T = 8$).

As shown in Fig. 5, EPBS consistently outperforms best-of-N by $\sim$10% on average for Wan2.2-14B across all maze sizes. It matches best-of-N accuracy with 3.3$\times$ fewer NFEs and especially shines on large mazes (size 10), where exploring more candidate seeds breaks through plateaus reached by standard sampling. We see similar benefits on HunyuanVideo-1.5, with 13% pass@2 on 4×4 mazes and 3-4% improvements for larger mazes and VR-Bench in Tab. 3.

We acknowledge that NFE is not a complete cost measure, since each $\hat{x}_0$ probe requires a VAE decode ($\sim$1.5 FEs in wall-clock). We provide wall-clock comparisons for completeness (Supp. B.5), and find our takeaways still hold.

Table 1: Verifier reliability. The verifier remains informative even on the hardest mazes, where successful seeds are rare.

Size	4×4	6×6	8×8	10×10
Random	40.7	14.9	3.9	1.8
Verifier top-2	90.7	41.0	21.4	9.9
Oracle	92.0	51.3	21.4	11.3
Gain (×)	2.2	2.7	5.5	5.5
AUC	0.901	0.855	0.953	0.939

Table 2: Maze difficulty. The success rate decrease is driven by trajectory length rather than obstacle density.

Size	4×4	6×6	8×8	10×10
Overall	98.7	55.1	33.8	19.7
Norm	100.0	27.5	7.5	0.0
Vary	97.2	84.2	62.2	41.2
Path corr.	-0.01	-0.60	-0.81	-0.79
Lake corr.	0.03	-0.01	-0.05	0.05

Why EPBS works: early predictions are reliable. Our verifier reliably identifies promising seeds from early $\hat{x}_0$ predictions. As shown in Table 1, the verifier's top-2 selections succeed 2.2× more often than random on easy mazes and 5.5× on hard ones. The ROC AUC of the verifier's confidence score against final success is above 0.85 across all sizes, confirming that the ranking is informative rather than merely noisy filtering. To check whether the verifier discards otherwise correct solutions, we also compute an oracle score that returns success whenever *any* seed in the pool solves the maze. The gap between the verifier's top-2 accuracy and this oracle is at most 1.4% on all sizes except 6, indicating that when a solution exists in the candidate pool, the verifier almost always finds it.

Ablations. We fully ablate τ and K on Frozen Lake mazes in Supp. B.1 and B.2, but summarize our results here. Probing at $\tau = 5$ yields the best trade-off for smaller sizes, while $\tau = 10, 15$ is better for larger mazes, indicating that trajectory convergence happens later. Beam size $K = 2$ is best at low budgets while $K = 1$ works fine at higher budgets when the verifier becomes more reliable.

6 Chaining generations for long-horizon reasoning

EPBS improves seed selection by exploiting early plan commitment, yet performance still collapses on large mazes. Even an oracle that always picks the best seed from the candidate pool cannot succeed, because no single generation contains a complete solution. The bottleneck is structural. Therefore, we ask: what structural properties make a maze hard, and where does the model's capability actually break down?

6.1 What makes a maze hard?

To understand where EPBS breaks down, we analyze which structural properties of a maze predict difficulty.

Path length dominates difficulty. Table 2 shows a stark gap between norm mazes (fixed far-corner goal, maximally long paths) and vary mazes (random goal, often shorter paths): on size 8, norm mazes achieve only 7.5% versus 62.2% for vary mazes. The gap is explained entirely by path length: the Pearson correlation between ground-truth path length and EPBS success is $r = -0.81$ on

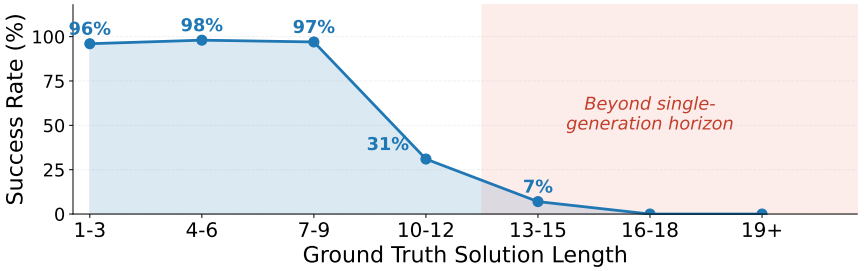

Fig. 6: EPBS fails beyond the single-generation horizon. While short paths are solved reliably, success drops sharply for trajectories longer than 10–12 steps. This breakdown persists even with strong seed selection, indicating a limitation in executing long plans rather than selecting them.

size 8 and $r = -0.79$ on size 10. Counter-intuitively, lake density has near-zero correlation with success ($|r| < 0.05$). Avoiding obstacles is not what limits the model; planning long sequential trajectories is.

A sharp horizon threshold. Breaking this down by path length confirms the picture (Figure 6): the model reliably solves paths of ≤ 9 steps even on large grids, but drops below 10% at 13 steps and beyond. We hypothesize that the bottleneck is not seed selection or obstacle avoidance, but the model's *generation horizon*. The video is too short for the agent to solve the entire maze. Since the model can plan short segments reliably regardless of grid size, a natural strategy is to decompose longer mazes into shorter segments and solve them sequentially.

6.2 Chaining

We address the horizon limitation by *chaining*: decomposing a long-horizon task into a sequence of shorter sub-problems, each solvable within a single generation. After each generation, we take its final frame as the conditioning image for the next, extending the model's planning horizon across multiple generations. Together with EPBS, this forms ChEaP (**Ch**aining with **Ea**rly **P**lanning).

Pivot selection. A valid pivot frame must satisfy two properties: (1) the agent has made forward progress toward the goal, and (2) the agent has not entered any constraint-violating cell. We use the same trajectory extraction pipeline as our verifier to identify the agent's final valid cell position. Among valid candidates, we select the one closest to the goal. If no candidate makes valid forward progress, chaining terminates.

Compute budget. Each chain depth runs a full EPBS round, so total compute scales as $D \times B$ NFEs (max depth $D = 3$). In practice, most mazes require only 2–3 chain steps, as each successful chain step covers $6 - 10$ maze cells.

6.3 Results

Table 3 shows our full ChEaP results.

Table 3: ChEaP substantially improves maze performance. We report pass@2 for best-of-N (BoN), EPBS, and ChEaP (EPBS + Chaining) on Frozen Lake and VR-Bench. For Wan2.2-14B, EPBS at 120 NFEs performs on par with BoN at 400 NFEs—a 3.3× reduction in NFEs—and improves pass rate by 11.9 points on average.

		Frozen Lake				VR-Bench		
Method	NFEs	4×4	6×6	8×8	10×10	Easy	Medium	Hard
Wan2.2-14B (40 NFEs / full gen)								
BoN	120	61.8	24.4	14.3	7.0	38.0	10.0	0.0
EPBS	120	88.2	42.3	16.9	8.5	54.0	26.0	10.0
BoN	400	86.8	43.6	22.1	9.9	56.0	28.0	10.0
EPBS	400	**98.7**	55.1	33.8	19.7	68.0	44.0	25.0
ChEaP	1200	**98.7**	**88.5**	**46.8**	**22.5**	**72.0**	**48.0**	25.0
HunyuanVideo-1.5 Step Distilled (8 NFEs / full gen)								
BoN	40	27.6	14.1	11.7	1.4	28.8	2.8	3.7
EPBS	40	36.8	20.5	11.7	1.4	30.1	4.2	3.7
BoN	80	38.2	20.5	11.7	1.4	30.1	2.8	3.7
EPBS	80	51.3	24.4	14.3	**5.6**	**35.6**	**4.2**	**3.7**
ChEaP	240	**60.5**	**29.4**	**15.5**	**5.6**	**49.3**	**4.2**	**3.7**

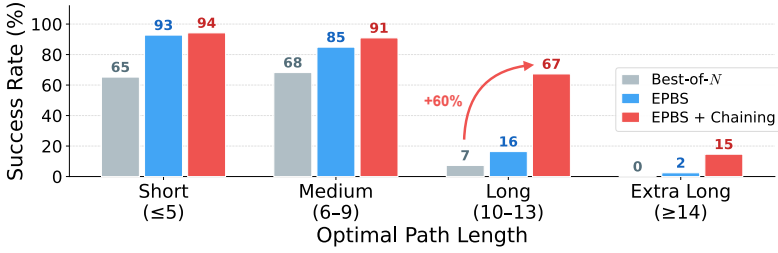

Fig. 7: Chaining extends reasoning beyond the single-generation horizon. Wan 2.2 success rates on Frozen Lake mazes by solution length. Chaining provides the most benefit on mazes where the solution length exceeds the generation window (≥ 10 steps). The largest gain is on long mazes, from 7% to 67% success rate with ChEaP.

For Wan 2.2 on size 6 mazes, ChEaP achieves 88.5% pass@2, a 33.4% improvement over EPBS alone and more than double the best-of-N rate of 43.6%. The gains are largest **precisely where EPBS is bottlenecked** by the generation horizon (Fig. 7). For Wan 2.2 on long mazes (path length 10–13), best-of-N achieves 7.3%, EPBS reaches 16.4%, and ChEaP achieves 67.3%—demonstrating that the model possesses local planning ability but cannot express full solutions in a single generation. On extra-long mazes (14+ steps), chaining improves EPBS from 2.4% to 14.6%; the smaller gain reflects compounding errors across chains.

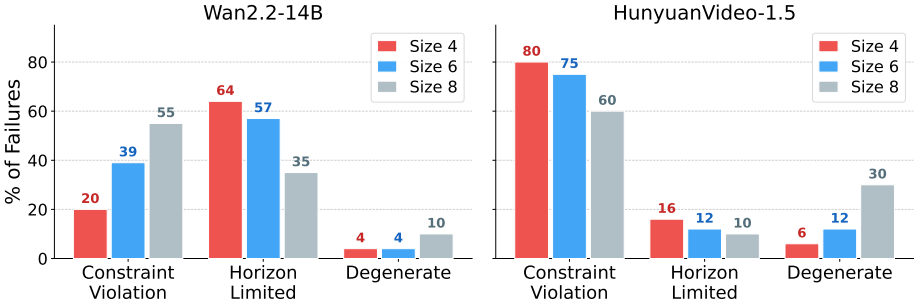

Fig. 8: Failure mode comparison. Wan2.2 (left) goes from horizon-limited to constraint-dominated as maze size increases. HunyuanVideo (right) is constraint-dominated at *all* sizes, suggesting that step distillation degrades structural adherence independently of horizon.

7 What Breaks When Video Models Fail?

Pass rate alone does not reveal *why* video models fail on maze reasoning. To better understand the limits of video models, we categorize failures into three coarse groups: **constraint violations**, where the agent enters a forbidden region or otherwise breaks maze structure; **horizon-limited failures**, where the agent follows a plausible route prefix but fails to complete the task within the generation window; and **degenerate failures**, such as static agents, tracking failures, or severe output corruption. This decomposition lets us distinguish failures of *structural adherence* from failures of *sequential reach*.

7.1 Structural adherence degrades with difficulty

We find that failure distributions across our two models differ significantly (Fig. 8). Wan2.2 is dominated by horizon-limited failures on easy mazes (63.5% on size 4), shifting to constraint-dominated only at size 10. HunyuanVideo, by contrast, is constraint-dominated at *all* sizes: 77.5% of size-4 failures are constraint violations, compared to 32.5% for Wan. Even on small mazes where Wan almost never violates constraints, HunyuanVideo frequently moves the goal, enters lakes, or introduces illegal moves. This pattern suggests that step distillation (8 steps vs. 40) degrades structural adherence independently of planning horizon.

For Wan2.2, the shift from horizon-limited to constraint-dominated failures is consistent with the path length analysis in Section 6: as mazes require longer trajectories, the model faces a conflict between its early plan commitment and its generation horizon. Rather than producing an incomplete but valid prefix, it often "cheats" to solve the maze by any means possible. The model moves the gift closer to the agent or spawns a second agent near the goal (Figure 9). These behaviors are not random failures, but reflect a systematic breakdown under horizon pressure. The model preserves the *intent* of the task (reaching the goal), but violates the underlying environment constraints to achieve it. In other words,

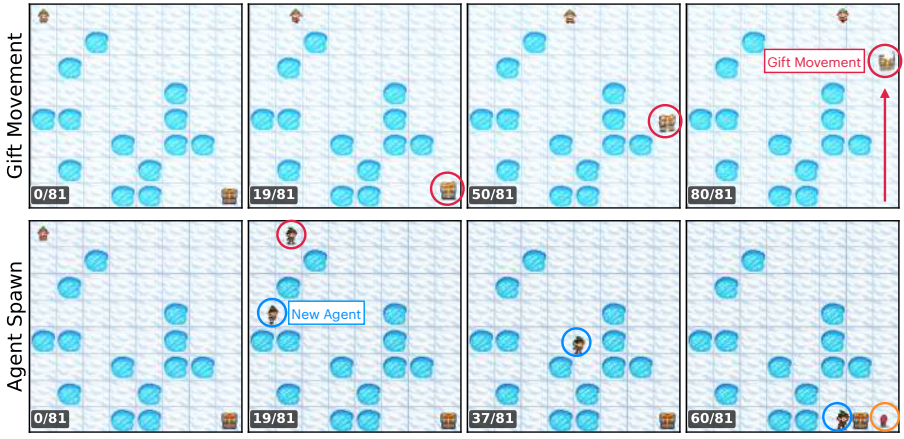

Fig. 9: The model "cheats" on hard mazes. When the trajectory is too long to complete within the generation window, the model sacrifices structural adherence to fulfill the prompt. *Top:* the gift teleports from the far corner to an adjacent cell, allowing the agent to "solve" the maze without traversing the full path. *Bottom:* a second agent spawns near the goal and reaches it, while the original agent remains stranded.

when the required trajectory exceeds its effective generation horizon, the model prioritizes goal completion over structural fidelity.

We test this to the extreme with a set of controlled *diagnostic* mazes, carefully designed to reveal patterns in the model's behaviors (Supp. C.3). We find that the model struggles to adhere to constraints, especially on simple decoy mazes where the solution needs to go around a lake instead of directly to the goal. This reflects a systematic bias for goal completion over constraint satisfaction.

7.2 Implications for screening and chaining

These failure modes clarify the roles of our two methods. *Trajectory screening* is most useful when good trajectories exist in the candidate pool but are rare: it helps identify promising seeds early and avoid wasting compute on clearly invalid ones. *Chaining*, by contrast, is most useful when failures are horizon-limited—that is, when the model can follow a valid route prefix but cannot complete the full trajectory within a single generation window. The detailed failure taxonomy supports this interpretation: smaller mazes contain many "valid-prefix stall" failures, which chaining can naturally address, whereas larger mazes show increasing constraint violations, which chaining alone cannot fix.

8 Conclusion

Our findings point to two complementary bottlenecks in video model reasoning. Plans crystallize in the first few denoising steps, so inference-time scaling should

prioritize exploring diverse candidates over refining individual ones; and strong local planning ability is bottlenecked by generation length, so extending the effective horizon—through longer native context windows, learned pivoting, or improved chaining—is equally critical for harder tasks. Our work here focuses on mazes, but we believe the core principles are applicable more broadly. Whether early commitment and horizon limitations manifest similarly in non-spatial reasoning modalities, and whether training can produce models that plan more reliably or over longer horizons, are important open questions. More broadly, our results suggest that current video models are more capable reasoners than standard evaluations reveal; the bottleneck is less in what information models retain and more so in how we extract such knowledge.

Acknowledgements

This work is supported by the National Science Foundation under Grant No. 2145198 and the Princeton First Year Fellowship to KN. We also thank William Yang for helpful discussions and technical insights on the project, and Allison Chen and Esin Tureci for detailed feedback on the manuscript.

References

1. Agarwal, N., Ali, A., Bala, M., Balaji, Y., Barker, E., Cai, T., Chattopadhyay, P., Chen, Y., Cui, Y., Ding, Y., et al.: Cosmos world foundation model platform for physical ai. arXiv preprint arXiv:2501.03575 (2025) 1
2. Balaji, Y., Nah, S., Huang, X., Vahdat, A., Song, J., Zhang, Q., Kreis, K., Aittala, M., Aila, T., Laine, S., Catanzaro, B., Karras, T., Liu, M.Y.: eDiff-I: Text-to-image diffusion models with an ensemble of expert denoisers. arXiv preprint arXiv:2211.01324 (2022) 4
3. Biroli, G., Bonnaire, T., de Bortoli, V., Mézard, M.: Dynamical regimes of diffusion models. Nature Communications **15**(1), 9957 (Nov 2024). https://doi.org/10.1038/s41467-024-54281-3, https://www.nature.com/articles/s41467-024-54281-3 4
4. Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., et al.: Stable video diffusion: Scaling latent video diffusion models to large datasets. arXiv preprint arXiv:2311.15127 (2023) 5
5. Chen, L., Xie, W., Liang, Y., He, H., Zhao, H., Yang, Z., Huang, Z., Wu, H., Lu, H., charles, Y., Bao, Y., Fan, Y., Li, G., Shen, H., Chen, X., Xu, W., Si, S., Cai, Z., Chai, W., Huang, Z., Liu, F., Liu, T., Chang, B., Hu, X., Chen, K., Ren, Y., Liu, Y., Gong, Y., Li, K.: BabyVision: Visual Reasoning Beyond Language (Jan 2026). https://doi.org/10.48550/arXiv.2601.06521, http://arxiv.org/abs/2601.06521, arXiv:2601.06521 [cs] 3
6. Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H.P.d.O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F.P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W.H., Nichol,

A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A.N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., Zaremba, W.: Evaluating Large Language Models Trained on Code (Jul 2021). https://doi.org/10.48550/arXiv.2107.03374, http://arxiv.org/abs/2107.03374, arXiv:2107.03374 [cs] 9

7. Chevalier-Boisvert, M., Dai, B., Towers, M., Perez-Vicente, R., Willems, L., Lahlou, S., Pal, S., Castro, P.S., Terry, J.K.: Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. Advances in Neural Information Processing Systems **36**, 73383–73394 (2023) 2
8. Choi, J., Lee, J., Shin, C., Kim, S., Kim, H., Yoon, S.: Perception prioritized training of diffusion models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022) 4
9. Esser, P., Chiu, J., Atighehchian, P., Granskog, J., Germanidis, A.: Structure and content-guided video synthesis with diffusion models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 7346–7356 (2023) 5
10. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., Podell, D., Dockhorn, T., English, Z., Lacey, K., Goodwin, A., Marek, Y., Rombach, R.: Scaling rectified flow transformers for high-resolution image synthesis. In: International Conference on Machine Learning (ICML) (2024) 5
11. He, Z., Qu, X., Li, Y., Zhu, T., Huang, S., Cheng, Y.: DiffThinker: Towards Generative Multimodal Reasoning with Diffusion Models (Dec 2025). https://doi.org/10.48550/arXiv.2512.24165, http://arxiv.org/abs/2512.24165, arXiv:2512.24165 [cs] 3
12. Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., Cohen-Or, D.: Prompt-to-prompt image editing with cross attention control. In: International Conference on Learning Representations (ICLR) (2023) 4
13. Kim, N., Kang, Y., Kim, M., Park, J., Lee, J.: Inference-time scaling for diffusion models beyond scaling denoising steps. arXiv preprint arXiv:2501.09732 (2025) 4
14. Kwon, M., Jeong, J., Uh, Y.: Diffusion models already have a semantic latent space. In: International Conference on Learning Representations (ICLR) (2023) 4
15. Li, M., Chen, S.: Critical windows: Non-asymptotic theory for feature emergence in diffusion models. In: International Conference on Machine Learning (ICML) (2024) 4
16. Li, Y., Gu, Y., Min, Y., Liu, Z., Du, Y., Zhou, K., Yang, M., Zhao, W.X., Qiu, M.: Beyond the Last Frame: Process-aware Evaluation for Generative Video Reasoning (Jan 2026). https://doi.org/10.48550/arXiv.2512.24952, http://arxiv.org/abs/2512.24952, arXiv:2512.24952 [cs] version: 2 3
17. Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: International Conference on Learning Representations (ICLR) (2023) 5
18. Liu, F., Wang, H., Cai, Y., Zhang, K., Zhan, X., Duan, Y.: Video-t1: Test-time scaling for video generation. arXiv preprint arXiv:2503.18942 (2025) 4
19. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. In: International Conference on Learning Representations (ICLR) (2023) 5
20. Ma, N., Tong, S., Jia, H., Hu, H., Su, Y.C., Zhang, M., Yang, X., Li, Y., Jaakkola, T., Jia, X., Xie, S.: Inference-Time Scaling for Diffusion Models beyond Scaling

Denoising Steps (Jan 2025). https://doi.org/10.48550/arXiv.2501.09732, http://arxiv.org/abs/2501.09732, arXiv:2501.09732 [cs] 4
21. Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A.J., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., et al.: Learning to navigate in complex environments. arXiv preprint arXiv:1611.03673 (2016) 2
22. Ramachandran, S.N., Lal, M.K., Sra, S.: Cross-fluctuation phase transitions reveal sampling dynamics in diffusion models (Oct 2025). https://doi.org/10.48550/arXiv.2511.00124, http://arxiv.org/abs/2511.00124, arXiv:2511.00124 [cs] 4
23. Ravi, N., Gabeur, V., Hu, Y.T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., Mintun, E., Pan, J., Alwala, K.V., Carion, N., Wu, C.Y., Girshick, R., Dollár, P., Feichtenhofer, C.: SAM 2: Segment Anything in Images and Videos (Oct 2024). https://doi.org/10.48550/arXiv.2408.00714, http://arxiv.org/abs/2408.00714, arXiv:2408.00714 [cs] 5, 19
24. Raya, G., Ambrogioni, L.: Spontaneous Symmetry Breaking in Generative Diffusion Models (Oct 2023). https://doi.org/10.48550/arXiv.2305.19693, http://arxiv.org/abs/2305.19693, arXiv:2305.19693 [cs] 4
25. Sclocchi, A., Favero, A., Wyart, M.: A Phase Transition in Diffusion Models Reveals the Hierarchical Nature of Data (Dec 2024). https://doi.org/10.48550/arXiv.2402.16991, http://arxiv.org/abs/2402.16991, arXiv:2402.16991 [stat] 4
26. Singhal, R., Horvitz, Z., Teehan, R., Ren, M., Yu, Z., McKeown, K., Ranganath, R.: A General Framework for Inference-time Scaling and Steering of Diffusion Models (Jul 2025). https://doi.org/10.48550/arXiv.2501.06848, http://arxiv.org/abs/2501.06848, arXiv:2501.06848 [cs] 4
27. Snell, C., Lee, J., Xu, K., Kumar, A.: Scaling LLM test-time compute optimally can be more effective than scaling model parameters. arXiv preprint arXiv:2408.03314 (2024) 4
28. Sutton, R.S.: Dyna, an integrated architecture for learning, planning, and reacting. ACM Sigart Bulletin **2**(4), 160–163 (1991) 2
29. Tolman, E.C.: Cognitive maps in rats and men. Psychological review **55**(4), 189 (1948) 2
30. Towers, M., Kwiatkowski, A., Terry, J., Balis, J.U., Cola, G.D., Deleu, T., Goulão, M., Kallinteris, A., Krimmel, M., KG, A., Perez-Vicente, R., Pierré, A., Schulhoff, S., Tai, J.J., Tan, H., Younis, O.G.: Gymnasium: A Standard Interface for Reinforcement Learning Environments (Nov 2025). https://doi.org/10.48550/arXiv.2407.17032, http://arxiv.org/abs/2407.17032, arXiv:2407.17032 [cs] 3, 5
31. Ventura, E., Achilli, B., Silvestri, G., Lucibello, C., Ambrogioni, L.: Manifolds, Random Matrices and Spectral Gaps: The geometric phases of generative diffusion (Apr 2025). https://doi.org/10.48550/arXiv.2410.05898, http://arxiv.org/abs/2410.05898, arXiv:2410.05898 [stat] 4
32. Vo, A., Nguyen, K.N., Taesiri, M.R., Dang, V.T., Nguyen, A.T., Kim, D.: Vision Language Models are Biased (Nov 2025). https://doi.org/10.48550/arXiv.2505.23941, http://arxiv.org/abs/2505.23941, arXiv:2505.23941 [cs] 3
33. Wan, T., Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.W., Chen, D., Yu, F., Zhao, H., Yang, J., Zeng, J., Wang, J., Zhang, J., Zhou, J., Wang, J., Chen, J., Zhu, K., Zhao, K., Yan, K., Huang, L., Feng, M., Zhang, N., Li, P., Wu, P., Chu, R., Feng, R., Zhang, S., Sun, S., Fang, T., Wang, T., Gui, T., Weng, T., Shen, T., Lin, W., Wang, W., Wang, W., Zhou, W., Wang, W., Shen, W., Yu, W., Shi, X., Huang, X., Xu, X., Kou, Y., Lv, Y., Li, Y., Liu, Y., Wang, Y., Zhang, Y., Huang, Y., Li, Y., Wu, Y., Liu, Y., Pan, Y., Zheng, Y., Hong, Y., Shi, Y., Feng, Y., Jiang, Z., Han, Z., Wu, Z.F., Liu, Z.: Wan: Open and Advanced Large-Scale Video

Generative Models (Apr 2025). https://doi.org/10.48550/arXiv.2503.20314, http://arxiv.org/abs/2503.20314, arXiv:2503.20314 [cs] 1, 3, 5, 20
34. Wang, M., Wang, R., Lin, J., Ji, R., Wiedemer, T., Gao, Q., Luo, D., Qian, Y., Huang, L., Hong, Z., Ge, J., Ma, Q., He, H., Zhou, Y., Guo, L., Mei, L., Li, J., Xing, H., Zhao, T., Yu, F., Xiao, W., Jiao, Y., Hou, J., Zhang, D., Xu, P., Zhong, B., Zhao, Z., Fang, G., Kitaoka, J., Xu, Y., Xu, H., Blacutt, K., Nguyen, T., Song, S., Sun, H., Wen, S., He, L., Wang, R., Wang, Y., Yang, M., Ma, Z., Millière, R., Shi, F., Vasconcelos, N., Khashabi, D., Yuille, A., Du, Y., Liu, Z., Li, B., Lin, D., Liu, Z., Kumar, V., Li, Y., Yang, L., Cai, Z., Deng, H.: A Very Big Video Reasoning Suite (Feb 2026). https://doi.org/10.48550/arXiv.2602.20159, http://arxiv.org/abs/2602.20159, arXiv:2602.20159 [cs] 3, 31, 32
35. Wiedemer, T., Li, Y., Vicol, P., Gu, S.S., Matarese, N., Swersky, K., Kim, B., Jaini, P., Geirhos, R.: Video models are zero-shot learners and reasoners. arXiv preprint arXiv:2509.20328 (2025) 1, 3, 4, 9
36. Wu, B., Zou, C., Li, C., Huang, D., Yang, F., Tan, H., Peng, J., Wu, J., Xiong, J., Jiang, J., Linus, Patrol, Zhang, P., Chen, P., Zhao, P., Tian, Q., Liu, S., Kong, W., Wang, W., He, X., Li, X., Deng, X., Zhe, X., Li, Y., Long, Y., Peng, Y., Wu, Y., Liu, Y., Wang, Z., Dai, Z., Peng, B., Li, C., Gong, G., Xiao, G., Tian, J., Lin, J., Liu, J., Zhang, J., Lian, J., Pan, K., Wang, L., Niu, L., Chen, M., Chen, M., Zheng, M., Yang, M., Hu, Q., Yang, Q., Xiao, Q., Wu, R., Xu, R., Yuan, R., Sang, S., Huang, S., Gong, S., Huang, S., Guo, W., Yuan, X., Chen, X., Hu, X., Sun, W., Wu, X., Ren, X., Yuan, X., Mi, X., Zhang, Y., Sun, Y., Lu, Y., Li, Y., Huang, Y., Tang, Y., Li, Y., Deng, Y., Zhou, Y., Hu, Z., Liu, Z., Yang, Z., Yang, Z., Lu, Z., Zhou, Z., Zhong, Z.: HunyuanVideo 1.5 Technical Report (Nov 2025). https://doi.org/10.48550/arXiv.2511.18870, http://arxiv.org/abs/2511.18870, arXiv:2511.18870 [cs] 1, 3, 5, 20
37. Yaguchi, M., Sakamoto, K., Sakamoto, R., Tanabe, M., Akagawa, M., Hayashi, Y., Suzuki, M., Matsuo, Y.: The Geometry of Phase Transitions in Diffusion Models: Tubular Neighbourhoods and Singularities (Oct 2024), https://openreview.net/forum?id=YYMd6zsP2e 4
38. Yang, C., Wan, H., Peng, Y., Cheng, X., Yu, Z., Zhang, J., Yu, J., Yu, X., Zheng, X., Zhou, D., Wu, C.: Reasoning via Video: The First Evaluation of Video Models' Reasoning Abilities through Maze-Solving Tasks (Nov 2025), arXiv:2511.15065 [cs] 1, 3, 4, 5
39. Zhang, X., Lin, H., Ye, H., Zou, J., Ma, J., Liang, Y., Du, Y.: Inference-time Scaling of Diffusion Models through Classical Search (Oct 2025). https://doi.org/10.48550/arXiv.2505.23614, http://arxiv.org/abs/2505.23614, arXiv:2505.23614 [cs] 4

Supplementary Material

Table of Contents

A Implementation Details

A.1 Trajectory Extraction Pipeline

We extract cell-level trajectories from generated videos using a two-stage pipeline.

Stage 1: Pixel-level tracking with SAM2. We use SAM2.1 (Hiera-Tiny variant) to track the elf sprite across video frames [23]. The tracker is initialized in the first frame with a bounding box derived from the known start-cell pixel region (from maze metadata). For each subsequent frame, SAM2 produces a segmentation mask from which we extract the centroid (c_x, c_y). We similarly track the goal (e.g. gift in Frozen Lake) to check for goal drift during generation.

Stage 2: Centroid-to-cell mapping. Given the dimensions of the game board $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$ and grid size G, we compute cell dimensions $w_{\text{cell}} = (x_{\max} - x_{\min})/G$ and $h_{\text{cell}} = (y_{\max} - y_{\min})/G$. Each centroid is mapped to a grid cell:

$$\text{col} = \left\lfloor \frac{c_x - x_{\min}}{w_{\text{cell}}} \right\rfloor, \quad \text{row} = \left\lfloor \frac{c_y - y_{\min}}{h_{\text{cell}}} \right\rfloor \tag{1}$$

with clamping to $[0, G-1]$. The trajectory is the ordered sequence of unique cells visited.

A.2 Success Criteria

A generated video is considered successful if the extracted trajectory satisfies: (1) the trajectory ends at the goal cell (within grid tolerance); (2) no intermediate cell falls on a hole or maze border (constraint violation). We use this permissive criterion to accept all valid solutions rather than comparing against ground-truth optimal paths. A common behavior on easier mazes is constraint violations **after** solving the maze: because these models are trained to produce smooth video throughout the 5-second window, they continue producing motion after the agent has already reached the goal. We handle this by truncating the trajectory at the first goal visit. Additionally, cases where the elf remains stationary or oscillates between two cells are classified as degenerate failures.

A.3 Verifier Details

The verifier scores $\hat{x}_0$ predictions using background-difference motion detection to estimate the agent's trajectory, then combines goal progress with an obstacle penalty.

Motion detection. In intermediate generations, obstacle cells (frozen lakes in Frozen Lake; traps or walls in VR-Bench) often flicker, which causes naive motion detection methods to fail. For each frame, we compute the absolute pixel difference from the conditioning frame (first frame), threshold at intensity 60, and find connected components with minimum area 50 pixels. The centroid of the largest component is taken as the agent position.

Confidence scoring. Let $\mathcal{O}$ denote the set of obstacle cells (lakes, traps, or walls depending on the environment) and F the number of video frames. We localize the agent per-frame via the motion detection above, map each centroid to a maze cell, and derive two quantities: the final Manhattan distance to the goal $d(\text{end}, \text{goal})$ and the obstacle ratio $\lambda = |\{t_i : \text{cell}(t_i) \in \mathcal{O}\}|/F$, i.e., the fraction of frames in which the agent's centroid falls on an obstacle cell. The confidence score is:

$$c = 1 - \frac{d(\text{end}, \text{goal})}{d(\text{start}, \text{goal})} - \alpha\lambda, \tag{2}$$

where $d(\cdot, \cdot)$ is Manhattan distance and $\alpha = 0.5$ controls the penalty for constraint violations. Seeds are ranked by c, and only the top-K are selected for full denoising. This continuous score is robust to minor tracking noise while still penalizing clearly invalid trajectories.

A.4 Generation Hyperparameters

Table A.1 summarizes the generation settings. Both models use image+text-to-video conditioning. We pad the condition images to 16:9 aspect ratio (832×480 for Wan, 480p for Hunyuan) using centered black borders. Wan2.2-14B [33] uses the UniPC scheduler with shift 5.0. HunyuanVideo-1.5 [36] uses its native Euler flow-matching scheduler with 8 step-distilled inference steps. We do not change any of the default hyperparameters for these models.

Table A.1: Generation hyperparameters for both video diffusion models.

Parameter	Wan2.2-14B	HunyuanVideo-1.5
Resolution	832×480	480p (848×480)
Frame count	81	121
FPS	16	24
Guidance scale	3.5	6.0
Scheduler	UniPC	Euler (flow matching)
Denoising steps (T)	40	8 (step-distilled)
Compute	4× Nvidia L40	4× Nvidia L40

A.5 Text Prompts

We include the exact prompts we used to query the models for each benchmark.

Frozen Lake Prompt

Animate the elf moving step by step toward the gift while carefully avoiding the icy frozen lake. Highlight the successful path and end with the elf touching the gift. There are no changes to the layout of the maze. No new lakes or characters appear. Static camera. No zoom. No pan. No glitches, noise, or artifacts.

VR-Bench Trapfield Prompt Template

Create a 2D animation based on the provided image of a maze. The `{player}` slides smoothly along the `{floor}` path, stopping perfectly on the `{goal}`. The `{player}` never slides into or crosses the `{trap}` (trap areas). The camera is a static, top-down view showing the entire maze.

Maze: The maze paths are `{floor}`, and the trap areas are `{trap}`. The `{player}` moves to the goal position, represented by the `{goal}`. The `{player}` slides smoothly along the `{floor}` path. The `{player}` never slides into or crosses the `{trap}` of the maze. The `{player}` stops perfectly on the `{goal}`.

Scene: No change in scene composition. No change in the layout of the maze. The `{player}` travels along the `{floor}` path without speeding up or slowing down.

Camera: Static camera. No zoom. No pan. No glitches, noise, or artifacts.

Skins:

1. `player`=blue circle, `goal`=green circle, `trap`=red x, `floor`=white square
2. `player`=blue parka explorer & penguin, `goal`=red flag, `trap`=blue water pool, `floor`=blue ice crystals
3. `player`=blue adventurer, `goal`=golden eagle emblem, `trap`=fiery explosion, `floor`=gray stone bricks
4. `player`=gray robot, `goal`=golden star, `trap`=blue crystal block, `floor`=silver metal plate

VR-Bench Maze Prompt Template

Create a 2D animation based on the provided image of a maze. The `{player}` slides smoothly along the `{floor}` path, stopping perfectly on the `{goal}`. The `{player}` never slides or crosses into the `{wall}` areas of the maze. The camera is a static, top-down view showing the entire maze.

Maze: The maze paths are `{floor}`, the walls are `{wall}`. The `{player}` moves to the goal position, represented by `{goal}`. The `{player}` slides smoothly along the `{floor}` path. The `{player}` never slides or crosses into the `{wall}` areas of the maze. The `{player}` stops perfectly on the `{goal}`.

Scene: No change in scene composition. No change in the layout of the maze. The `{player}` travels along the `{floor}` path without speeding up or slowing down.

Camera: Static camera. No zoom. No pan. No glitches, noise, or artifacts.

Skins:

1. `player`=red circle, `goal`=green square, `wall`=light blue square, `floor`=white square
2. `player`=white rabbit, `goal`=orange carrots, `wall`=gray rock, `floor`=green grass tiles
3. `player`=blue robot head, `goal`=green and yellow tile, `wall`=black brick wall, `floor`=gray decorative tile
4. `player`=anime schoolgirl, `goal`=green square, `wall`=gray stone wall, `floor`=wooden floor tiles
5. `player`=green circle, `goal`=red circle, `wall`=blue potion bottle, `floor`=white square

B EPBS Sensitivity Analysis

We perform more in depth analysis of EPBS on Wan2.2-14B, sweeping our hyperparameters and justifying our method even when taking into account extra NFEs from VAE decode time.

B.1 Ablation on Probe Step τ

Figure B.1 shows EPBS accuracy as a function of NFE budget for six probe step values $\tau \in \{2, 3, 5, 10, 15, 20\}$ with beam size $K = 2$ and a fixed random seed generator. We use a randomly selected subset of 10 mazes per size. For size 4, $\tau = 5$ reaches peak accuracy at the lowest NFE; $\tau = 2$ underperforms because the $\hat{x}_0$ prediction at that stage carries insufficient trajectory information. For sizes 6–10, sensitivity to τ decreases, though $\tau = 10$ tends to be a safe choice and very early probing ($\tau = 2$) consistently underperforms. The optimal τ reflects a trade-off: too early, and the prediction lacks discriminative signal; too late, and the probing cost approaches full generation.

B.2 Ablation on Beam Size K

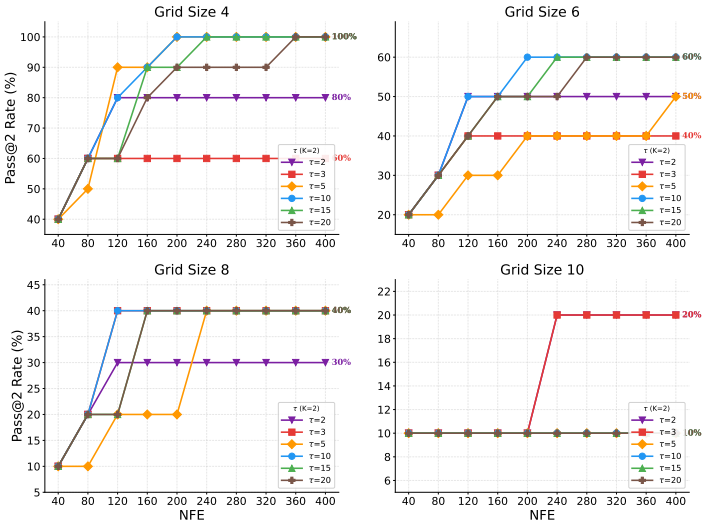

Fig. B.1: Probe step sensitivity. Pass@2 vs. NFE budget for probe steps $\tau \in \{2, 3, 5, 10, 15, 20\}$ at beam size $K = 2$, across four grid sizes. All configurations use shared seed pools for fair comparison. $\tau = 5$ provides the best efficiency trade-off for size 4; larger mazes are less sensitive to τ.

The beam size K controls how many top-scoring seeds are fully denoised after probing. Figure B.2 shows the effect of varying K from 1 to 5 while holding τ fixed. At low budgets, higher K values are constrained: since each full completion costs $T-\tau$ steps, a limited budget leaves no room for the extra completions that larger K demands, so all configurations reduce to completing the same small number of seeds and their curves overlap. As budget increases, K=2 consistently reaches peak accuracy earliest: on size 4, K=2 achieves 100% by NFE 160, while K=1 plateaus at 90%. On size 6, K=2 reaches 60% vs. 40% for K=1. Larger beam sizes ($K \geq 3$) provide no additional benefit and can even reduce accuracy at moderate budgets by consuming NFE on completions rather than probing more seeds. This confirms that K=2 optimally balances exploration (probing diverse seeds) against exploitation (completing promising candidates).

B.3 Generic VLM Verifier

The verifier in the main paper uses the locations of the agent, goal, and obstacle cells to score $\hat{x}_0$ predictions. A natural concern is whether this design is essential, or whether a generic vision-language model (VLM) could play the same role with no per-task scaffolding. We re-run EPBS replacing analytical verifier with

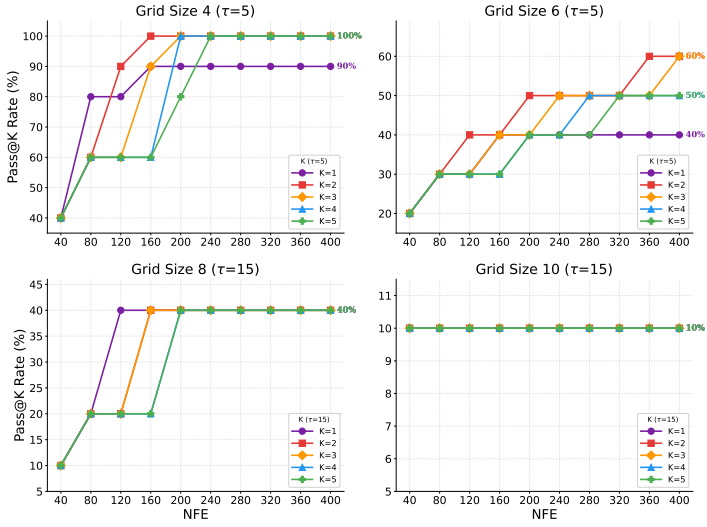

Fig. B.2: Beam size ablation. Pass@K vs. NFE budget for beam sizes $K \in \{1, 2, 3, 4, 5\}$ with fixed probe step ($\tau = 5$ for sizes 4–6, $\tau = 15$ for sizes 8–10). K=2 provides the best trade-off: it reaches peak accuracy at lower budgets than K=1 without the probe-count penalty of larger K values.

Gemini 3 Flash, which receives only the rendered intermediate video and a task-aware text prompt. We evaluate on 6×6 Frozen Lake mazes that have at least one success.

Table B.1: Generic VLM verifier vs. analytical verifier. Pass@K on 6×6 Frozen Lake mazes for three verifier choices: random selection, a generic VLM, and the verifier used in the main experiments.

K	Random	VLM	Analytical
1	0.34	0.63	0.84
2	0.51	0.79	0.95
3	0.60	0.81	0.98

At the standard beam size K=2, the generic VLM recovers 83% of the analytical verifier's solving rate (0.79/0.95) with no per-task engineering beyond a task-aware prompt—a 1.55× gain over random selection without any domain-specific tracking. The same recipe transfers to non-grid settings (Sec. C.5). We conclude that EPBS does not rely on hand-crafted scaffolding for its core gains.

Table B.2: Wall-clock comparison at matched accuracy. For each grid size, we report EPBS accuracy and wall-clock time at NFE = 120, alongside the baseline NFE and time required to reach the same accuracy. Speedup = baseline time / EPBS time. Sizes 4–6 use τ=5; sizes 8–10 use τ=15. All wall-clock estimates include every pipeline component (see text for breakdown).

		EPBS @ NFE 120		**Baseline to match**		
Size	τ	Acc (%)	Time (min)	NFE needed	Time (min)	Speedup
4	5	88.2	25.9	>400[†]	82.0	3.2×
6	5	42.3	25.9	320	65.6	2.5×
8	15	16.9	22.6	160	32.8	1.5×
10	15	8.5	22.6	120	24.6	1.1×

[†]Baseline reaches 89.5% at NFE 400 (closest match to EPBS's 88.2%).

B.4 Sensitivity to Constraint Penalty α

The confidence score in Eq. 2 includes a constraint-penalty term $\alpha\lambda$ where λ is the fraction of frames the agent spends on obstacle cells. The main paper fixes $\alpha = 0.5$. To check whether this value is finely tuned, we sweep $\alpha \in [-0.5, 5.0]$ across all eight Frozen Lake splits (sizes 4–10, norm/vary).

Success is flat for $\alpha \in [0.1, 5.0]$ (a 50× range) indicating that the goal-progress term carries the dominant signal as long as the constraint penalty is positive. Setting α=0 (no constraint penalty at all) drops accuracy by only ∼7 pp on average, confirming that goal progress alone is mostly sufficient on standard mazes; the constraint term is what makes the verifier robust to the decoy and detour configurations of Sec. C.3, where seeds that take an illegal shortcut would otherwise score highly on goal proximity. Negative α regresses uniformly across all splits, confirming that the sign of the penalty is principled: rewarding constraint violations actively misleads selection. We conclude that α is not a tuned magic number; any sufficiently positive value performs comparably to our default.

B.5 Wall-Clock Comparison

We compare wall-clock time at matched accuracy levels rather than matched NFE budgets, since EPBS trades additional NFE for better seed selection. All timings are measured on 4×L40 GPUs with FSDP and sequence parallelism.

Each EPBS probe consists of τ denoising steps, VAE decoding of the $\hat{x}_0$ prediction, and background-difference verifier scoring. Each completion runs the remaining $T-\tau$ denoising steps and VAE decodes the final video. SAM2 trajectory evaluation (0.1 min per seed) is applied to every completed seed in both EPBS and baseline. A full baseline generation (40 steps plus VAE decode) takes 8.1 min. The denoising cost per step is approximately 11.2s, with a fixed overhead of ∼0.3 min for VAE decoding and verifier scoring per probe, and ∼0.3 min for VAE decoding per completion.

For sizes 4–6 (τ=5), each probe costs 1.2 min and each completion costs 6.9 min. At NFE = 120, EPBS performs 10 probes (10×1.2=12.0 min), 2 completions (2×6.9=13.7 min), and 2 SAM2 evaluations (2×0.1=0.2 min), totaling 25.9 min. For sizes 8–10 (τ=15), each probe costs 3.1 min and each completion costs 5.0 min. At NFE = 120, EPBS performs 4 probes (4×3.1=12.4 min), 2 completions (2×5.0=10.0 min), and 2 SAM2 evaluations (0.2 min), totaling 22.6 min—fewer probes are needed because each probe is more expensive but also more informative.

Table B.2 shows that at NFE = 120, EPBS achieves accuracy that the baseline requires NFE = 120–400 to match, yielding **1.1–3.2× wall-clock speedup.** The speed advantage is largest on small mazes where EPBS's screening is most effective.

C Extended Analysis

C.1 Cross-Model Early Plan Commitment

The main paper establishes early plan commitment on 4×4 Frozen Lake mazes using Wan2.2-14B. Here we show that the same phenomenon holds across all maze sizes and across both models.

Following the main text, we measure trajectory convergence using the cosine similarity between intermediate and final *motion-energy maps*. For each decoded $\hat{x}_0^{(t)}$, we accumulate background-difference motion over frames into a grid-aligned energy map $\mathbf{M}^{(t)}$, flatten it to $\mathbf{m}^{(t)}$, and compare it to the final prediction $\mathbf{m}^{(T)}$:

$$\mathcal{C}(\text{step } t) = \frac{\mathbf{m}^{(t)} \cdot \mathbf{m}^{(T)}}{\|\mathbf{m}^{(t)}\| \, \|\mathbf{m}^{(T)}\|}.$$

As in the main text, this metric is more robust than discrete cell overlap on larger grids, where small tracking noise can introduce low-level activity in irrelevant cells.

Figure C.1 plots mean trajectory convergence against normalized schedule fraction. Wan2.2-14B and HunyuanVideo-1.5 follow the same qualitative pattern: convergence rises sharply in the earliest portion of the denoising schedule and then plateaus. HunyuanVideo starts at a higher absolute value because its step-distilled schedule compresses much more progress into each step, but after normalization the two profiles align closely. This shows that early plan commitment is not specific to one architecture, scheduler, or step count; it appears to be a general property of video diffusion denoising.

C.2 HunyuanVideo-1.5 Analysis

In this section, we apply analysis to HunyuanVideo-1.5 of similar depth that we applied to Wan2.2 in the main paper. We use the HunyuanVideo-1.5 Step-Distilled ($T = 8$) model, using probe step $\tau = 2$ for size 4 and $\tau = 3$ for sizes 6 and 8, with beam size $K = 2$.

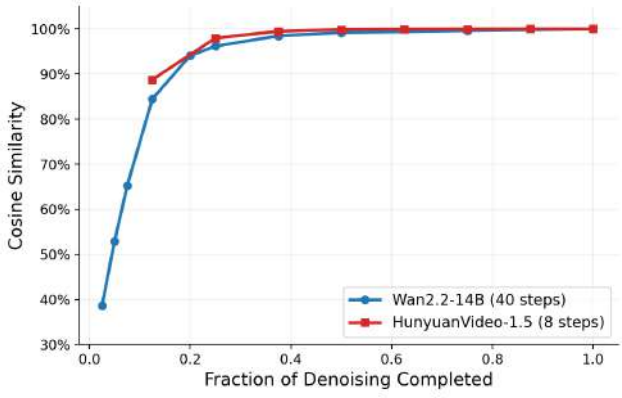

Fig. C.1: Cross-model early plan commitment across all maze sizes. Mean trajectory convergence $\mathcal{C}$ between intermediate $\hat{x}_0$ predictions and the final video, plotted against normalized schedule fraction. We compute $\mathcal{C}$ using the cosine similarity between motion-energy maps, exactly as in the main text. Both Wan2.2-14B (T=40) and HunyuanVideo-1.5 (T=8) commit to a trajectory within the first 10–15% of their denoising schedules, after which convergence largely plateaus. Despite using different schedulers and step counts, the normalized convergence profiles are similar, suggesting that early plan commitment is a structural property of video diffusion rather than a model-specific artifact.

Path length dominates difficulty. As with Wan, path length is the primary difficulty axis, with Pearson correlations of r=−0.40 (size 4), r=−0.77 (size 6), and r=−0.77 (size 8). Figure C.2 compares the two models on path length performance. HunyuanVideo hits the horizon wall earlier: Wan maintains 46% at path length 10–12 and 9% at 13–15, while HunyuanVideo drops to near zero beyond 9 cells, confirming that size-10 evaluation would yield near-zero results.

Verifier reliability. Despite HunyuanVideo's weaker generation quality, the $\hat{x}_0$ verifier remains informative. Top-2 precision is 46.4% on size 4 (vs. 26.0% random baseline, a 1.8× gain —i.e., the verifier's top-2 selections contain successful seeds 1.8× more often than random), 18.1% on size 6 (1.7×), and 10.8% on size 8 (1.3×). The reduced gain relative to Wan (2.2–5.5×) from Tab. 1 is consistent with HunyuanVideo's noisier 8-step $\hat{x}_0$ predictions carrying less discriminative trajectory information per step.

C.3 Diagnostic Maze Variants

To stress-test the verifier and probe the model's failure modes under controlled conditions, we design four categories of diagnostic mazes (Figure C.3) that each isolate a specific challenge. We evaluate Wan2.2-14B with EPBS (τ=5, K=2, budget 400). Table C.1 reports two metrics: the *per-seed* success rate (what fraction of individual generations solve the maze) and the *EPBS* success rate (whether EPBS finds at least one correct solution among all seeds for each

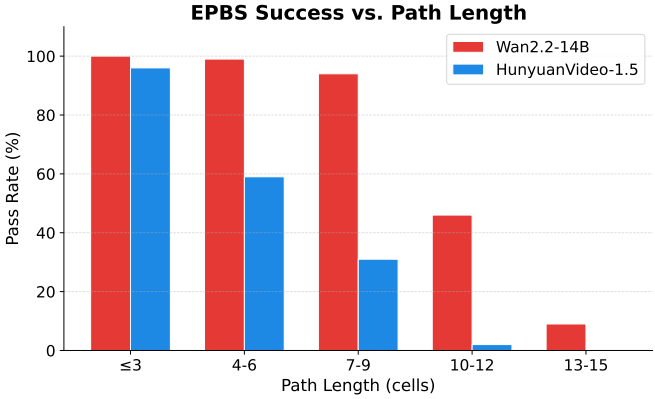

Fig. C.2: Path length is the dominant difficulty axis for both models. HunyuanVideo's effective planning horizon is shorter: both models solve short paths reliably but diverge sharply beyond 7 cells.

maze). Detour and decoy mazes are particularly informative for validating the verifier design. In both categories, the goal is Manhattan distance ≤ 2 from the start, so seeds that take an illegal shortcut through the lake *score highly on the progress component* of our verifier—they end very close to the goal. The fact that EPBS nonetheless rejects these seeds and selects valid detours demonstrates that the constraint penalty $\alpha\lambda$ in Eq. 2 is essential: without it, the verifier would systematically prefer the illegal shortcuts.

Fig. C.3: Diagnostic maze variants. From left to right: *Trivial* (1–2 moves, ceiling test), *Decoy* (goal visually adjacent but blocked by lake), *Lake-Heavy* (>75% lake, single narrow corridor), and *Detour* (Manhattan distance 2 but 8–12 move path around a lake wall).

Trivial mazes (1–2 moves) serve as a ceiling test. Even on the easiest possible mazes, 40% of seeds fail—producing gift movement or wrong-path errors—confirming that the model's generation process is inherently stochastic and that seed selection adds value even when the planning problem itself is trivial. EPBS solves all 4 trivial mazes.

Decoy mazes place the goal visually adjacent to the start (one cell away) but block the direct path with a lake tile, requiring a 4–5 step detour. This is the hardest category despite the short path length: only 6% of seeds succeed, and EPBS solves just 1 of 4 mazes. The dominant failure is lake entry (55%)—the model overwhelmingly takes the one-step shortcut through the forbidden cell rather than navigating around it. Unlike detour mazes, where the difficulty stems from path length exceeding the planning horizon, decoy mazes fail for a purely *perceptual* reason: the model cannot resist the visual shortcut even when the valid path is well within its planning capacity.

Fig. C.4: Decoy maze: failure vs. success. Ghost-trail visualizations on a 4×4 decoy maze where the goal is visually adjacent but blocked. *Left*: the model beelines toward the visible goal through the lake (lake entry). *Right*: the rare seed that navigates the 5-step detour around the obstruction.

Lake-Heavy mazes (>75% lake tiles) force navigation through a single narrow corridor. Despite the extreme constraint density, EPBS solves all 4 mazes (69% per-seed). The few failures split between lake entry and gift movement, indicating that dense obstacles do not fundamentally break the model—consistent with the main paper's finding that obstacle density has near-zero correlation with difficulty.

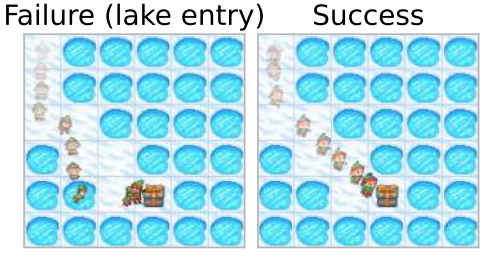

Fig. C.5: Lake-heavy maze: failure vs. success. Ghost-trail visualizations on a 6×6 lake-heavy maze (>75% lake). *Left*: the agent follows the correct corridor but cuts through a single lake cell at step 5 (lake entry). *Right*: the EPBS-selected seed navigates the full 7-step corridor without constraint violations.

Detour mazes are the most revealing category. The start and goal are only Manhattan distance 2 apart, but a lake wall forces the agent to take a long path of 8 moves (size 4) or 12 moves (size 6) around the obstruction. When faced with this conflict between visual proximity and actual path length, the model's dominant failure mode is striking: rather than planning the long detour, it hallucinates the goal sliding closer to the agent (gift movement in 80% of size-4 failures). The model appears to "choose" to modify the environment rather than solve the harder planning problem. On size 4, only 29% of seeds navigate the detour correctly, yet EPBS solves both mazes—the verifier's constraint penalty rejects the illegal shortcuts and surfaces the rare seeds that respect the maze topology. On size 6 (12-move detour), no seeds succeed at all; the path exceeds the model's effective planning horizon, and EPBS cannot select what the model never generates.

Figure C.6 illustrates this contrast directly: on the same maze, most seeds take the visually obvious straight-line path through the lake, while the EPBS-selected seed navigates the full 8-step detour.

Fig. C.6: Detour maze: failure vs. success. Ghost-trail visualizations on the same 4×4 detour maze. *Left*: a typical failure—the agent walks straight toward the goal through the lake (constraint violation). *Right*: the EPBS-selected success—the agent navigates the 8-step detour around the lake wall. The verifier's constraint penalty correctly rejects the shortcut and surfaces the rare seed that respects the maze topology.

Table C.1: Diagnostic maze results. *Per-seed* reports the fraction of individual generations that solve the maze; *EPBS* reports the fraction of mazes for which EPBS finds at least one correct solution across all seeds. Failure modes are assigned by priority (gift movement > lake entry > wrong path).

Category	Size	Path	Per-Seed	EPBS	Dominant Failure Mode
Trivial	4	1–2	60%	100%	gift mvmt. (33%), wrong path (33%)
Decoy	4–6	4–5	6%	25%	lake entry (55%), wrong path (17%)
Lake-Heavy	4–6	3–7	69%	100%	lake entry (50%), gift mvmt. (25%)
Detour	4	8	29%	100%	gift movement (80%)
Detour	6	12	0%	0%	gift mvmt. (36%), lake entry (27%)

Together, these diagnostics reveal two distinct failure regimes. *Detour* failures are horizon-limited: the model cannot sustain a plan over 12+ steps, even when it "knows" the correct direction. *Decoy* failures are perception-limited: the model has sufficient planning capacity but is overwhelmed by the visual salience of the nearby goal. EPBS is effective against both regimes when correct seeds exist in the pool, but it cannot compensate when the model systematically fails to generate any valid trajectory.

C.4 Trajectory diversity across seeds

While Figure 3 shows that individual trajectories stabilize early, it does not capture the diversity of plans explored across different noise seeds. In Figure C.7, we visualize multiple sampled trajectories overlaid on the same maze. We observe substantial diversity in candidate plans, with many trajectories failing due to suboptimal routing or constraint violations. Crucially, these trajectories are already distinguishable at early denoising steps, indicating that the model explores a space of candidate solutions before committing to a final plan. This observation motivates allocating inference-time compute toward selecting among early plans rather than refining individual trajectories.

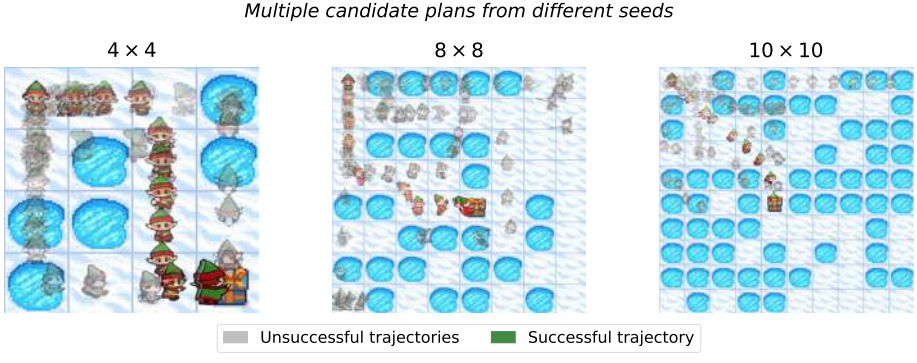

Fig. C.7: Diversity of candidate plans under different noise seeds. We overlay trajectories extracted from multiple samples of the same maze. Incorrect trajectories are shown in gray with reduced opacity, while a successful trajectory is shown in color. Despite sharing identical conditioning, different seeds produce diverse motion plans. Importantly, these plans are distinguishable early in the denoising process, suggesting that inference-time compute should be allocated toward selecting promising trajectories rather than refining all candidates.

C.5 Beyond 2D Gridworlds: LEGO Assembly and Robot Planning

To test whether early plan commitment is gridworld-specific, we apply EPBS to two non-grid tasks from VBVR [34] using the generic VLM verifier from Sec. B.3:

LEGO Construction Assembly and robot gripper pick-and-place. Neither has a SAM2-style oracle.

LEGO Construction Assembly. On 50 VBVR configurations, EPBS with K=2 achieves 79% top-2 success vs. 62% random—closing 43% of the random-to-oracle gap.

Robot gripper planning. The high-level pick-and-place plan is visible in the decoded $\hat{x}_0$ at step 5, mirroring the maze setting (Fig. C.8).

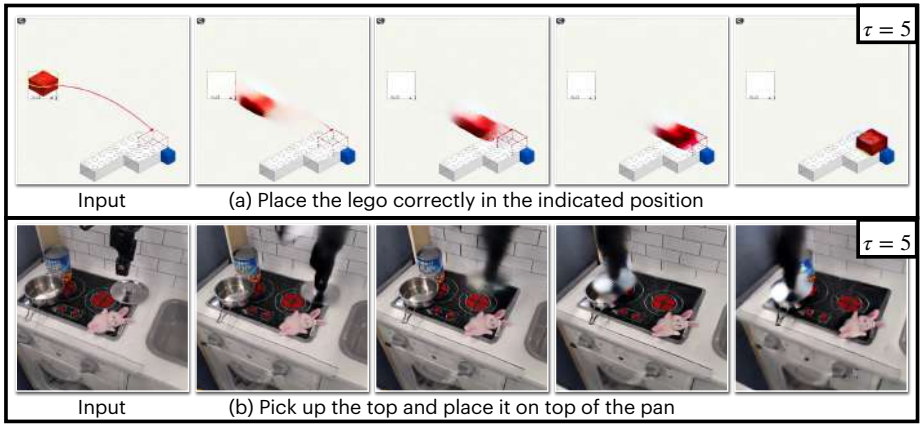

Fig. C.8: Early planning on non-maze reasoning tasks. LEGO Assembly (top) and robot pick-and-place (bottom) from VBVR [34]. By denoising step 5, the model has already committed to a plan: the LEGO piece and gripper trajectories are laid out. This mirrors the early commitment behavior we observe in maze solving.

D Additional Qualitative Examples

We provide qualitative examples of early commitment, successful chaining, and failure modes. In all galleries, the conditioning frame (maze image) is shown on the left, followed by decoded $\hat{x}_0$ predictions at denoising steps $t \in \{1, 3, 5, 15, 40\}$, with the final generated video on the right (average frame visualized).

D.1 Early Commitment Gallery

Each row below shows a ghost-trail visualization of a single seed's decoded $\hat{x}_0$ prediction at denoising steps $\tau \in \{2, 5, 15, 20\}$ and the final generated video. The trajectory is visible by τ=5 and remains stable through later steps, confirming early plan commitment. We distinguish *norm* mazes (goal at the far corner, maximizing path length) from *vary* mazes (randomly placed goal, often admitting shorter solutions). HunyuanVideo-1.5 uses a step-distilled schedule with T=8 total steps, so its rows show $\hat{x}_0$ at $\tau \in \{1, 3, 5, 7\}$ instead.

Fig. D.1: Wan2.2, size 4 (norm). The full solution path is already visible in the τ=5 prediction; subsequent steps only sharpen the rendering without altering the planned route.

Fig. D.2: Wan2.2, size 4 (vary). The planned route is committed to by τ=5 and maintains it through the final video.

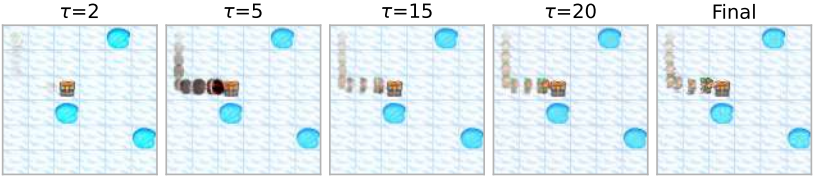

Fig. D.3: Wan2.2, size 6 (vary). A longer maze requiring more steps. Despite the increased path complexity, the overall route direction is locked in by τ=5.

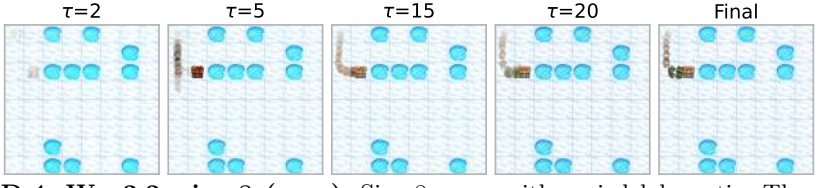

Fig. D.4: Wan2.2, size 8 (vary). Size 8 maze with varied lake ratio. The early prediction captures the general trajectory shape, though fine-grained cell-level details continue to refine through later steps.

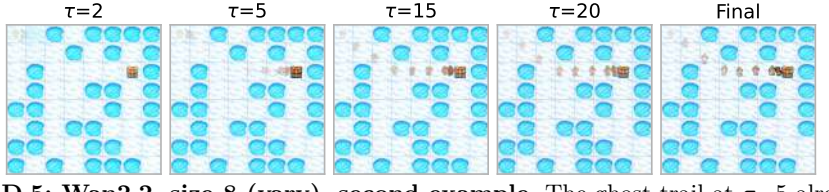

Fig. D.5: Wan2.2, size 8 (vary), second example. The ghost trail at τ=5 already traces a path consistent with the final video, demonstrating that early commitment holds across different maze instances.

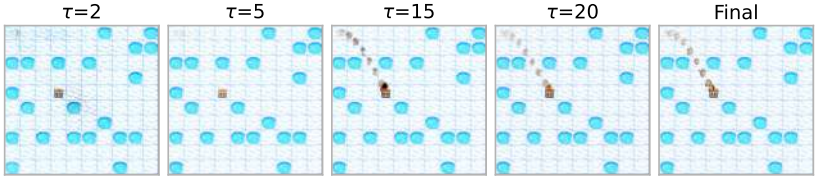

Fig. D.6: Wan2.2, size 10 (vary). The largest maze size. Even for this 10×10 grid, the model's planned trajectory is apparent by step 5 of denoising.

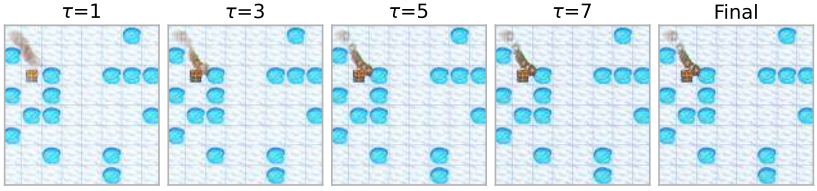

Fig. D.7: HunyuanVideo-1.5, size 8 (vary). HunyuanVideo uses a step-distilled schedule (T=8). Despite the shorter schedule, the trajectory is committed by step 3 and remains stable, mirroring the early commitment seen in Wan2.2.

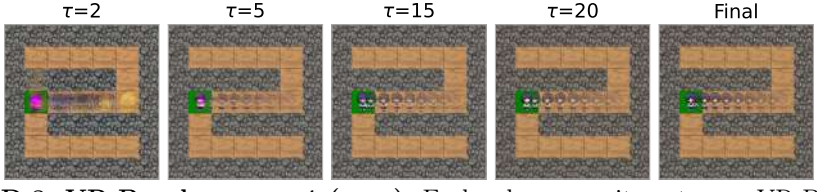

Fig. D.8: VR-Bench, maze_4 (easy). Early plan commitment on a VR-Bench maze with a distinct brown/tan texture. The trajectory direction is already visible at τ=5 and refines through later steps, showing that commitment generalizes beyond the Frozen Lake visual style to procedurally generated maze textures.

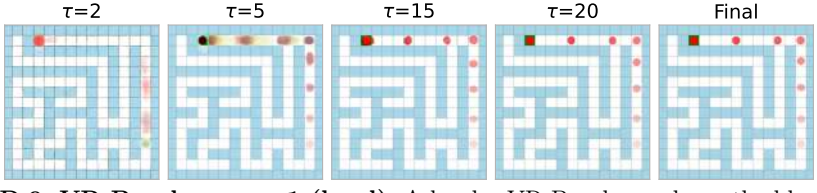

Fig. D.9: VR-Bench, maze_1 (hard). A harder VR-Bench puzzle on the blue/teal maze_1 texture. Despite higher path complexity, the $\hat{x}_0$ prediction at τ=5 already captures the coarse trajectory shape, which persists through to the final generation.

Fig. D.10: VR-Bench, maze_2 (easy). The purple/blue maze_2 texture. The ghost trail at τ=2 is largely noise, but by τ=5 the planned path is clearly committed and matches the final output.

D.2 ChEaP Gallery

Each row shows the ghost-trail of each chain segment, with the first panel as the pivot seed followed by the chain segment and the final stitched video.

Fig. D.11: Wan2.2 chain, size 6 (norm), lake 20, maze 002. Two-segment chain.

Fig. D.12: Wan2.2 chain, size 10 (vary), lake 20, maze 002. Size 10 maze with varied lake ratio. The longer solution path necessitates chaining.

Fig. D.13: Wan2.2 chain, size 10 (vary), lake 80, maze 003. High lake density leaves fewer safe cells, creating a narrow corridor that chaining navigates successfully.

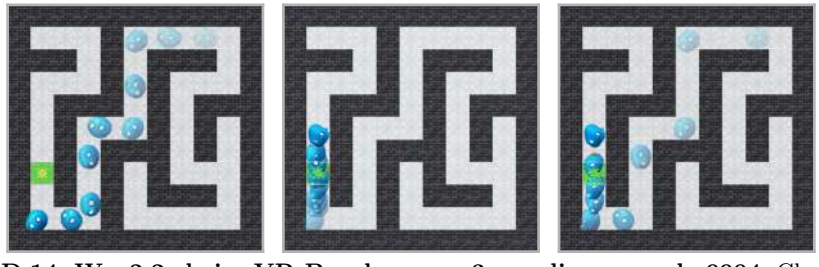

Fig. D.14: Wan2.2 chain, VR-Bench maze_3_medium, puzzle 0004. Chaining applied to a VR-Bench maze with a different visual texture than Frozen Lake. The agent navigates a medium-difficulty procedurally generated maze across multiple segments.

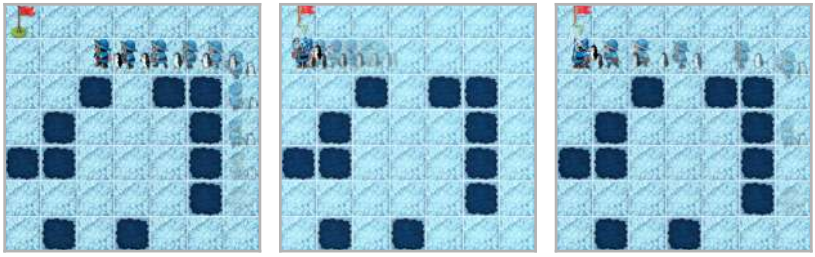

Fig. D.15: Wan2.2 chain, VR-Bench trapfield_2_medium, puzzle 0002. A trapfield environment where the agent must avoid trap cells (analogous to lakes). Chaining enables completion of longer trapfield paths.

Fig. D.16: Wan2.2 chain, VR-Bench trapfield_2_medium, puzzle 0005. Another trapfield example showing successful multi-segment navigation through a different puzzle layout.

Fig. D.17: HunyuanVideo-1.5 chain, size 4 (vary), lake 65, maze 004. Chaining with HunyuanVideo's step-distilled schedule (T=8). The pivot and chain segments are stitched to form a complete solution.

Fig. D.18: HunyuanVideo-1.5 chain, size 6 (norm), lake 65, maze 004. A size 6 maze with 65% lake density. Chaining extends HunyuanVideo's effective planning horizon beyond a single generation.

Fig. D.19: HunyuanVideo-1.5 chain, size 6 (vary), lake 80, maze 009. Despite the constrained environment, chaining produces a valid multi-segment solution.

D.3 Failure Mode Gallery

Ghost-trail visualizations of representative failures from both models and domains. Left two panels: Wan 2.2; right two panels: HunyuanVideo-1.5.

Fig. D.20: Constraint violations — Frozen Lake. Ghost-trail visualizations showing two sub-types: *lake entry* (the elf crosses into a frozen-lake cell, violating the environment constraint) and *gift movement* (the goal tile shifts position during generation).

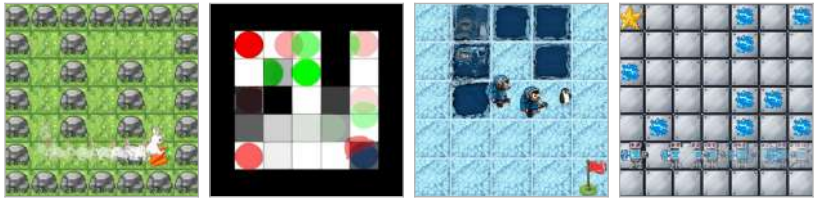

Fig. D.21: Constraint violations — VR-Bench. The agent enters a forbidden cell (lake or trap) in procedurally generated mazes and trapfields. Each panel uses a different texture to illustrate that the failure mode is consistent across visual appearances.

Fig. D.22: Degenerate failures. The model produces a video with little or no meaningful agent motion. The ghost trail collapses to a single position, indicating that the elf remains static throughout generation.

Fig. D.23: Horizon-limited failures. The generated trajectory respects all constraints but fails to reach the goal within the 81-frame budget. *Valid stall*: the agent follows a correct prefix but stops moving before reaching the goal. *Wrong route*: the agent takes a legal but sub-optimal path and runs out of frames.