COMPOSITIONAL DIFFUSION WITH HIERARCHICAL LATENT CASCADES

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models are powerful generative tools across images, video, motion planning and beyond. Compositional diffusion extends a pretrained model to horizons longer than those it was trained on by jointly denoising overlapping windows, and a growing body of work keeps the result **globally coherent**: seamless joints and a consistent layout and style. Yet the composed output often fails the intended **global semantics**: an object requested once appears in several windows, or in the wrong place. We introduce Hierarchical Latent Cascades (HLC), which routes the global condition down a hierarchy of windows. A forward pass of the pretrained model on a coarse view of the whole output reveals how each part of the condition is distributed over local horizons; we use it to reweight the condition for every child window and pass it down the tree. Each child denoises under its own condition and its parent's estimate, and overlapping siblings are aggregated level by level into the long-horizon output. HLC is training-free and uses only the model's own attention, so the same procedure applies to image, video and trajectory models whose condition enters through attention. Experiments on panorama images, long videos and robot motion planning show that it keeps much of the requested global semantics while staying coherent.

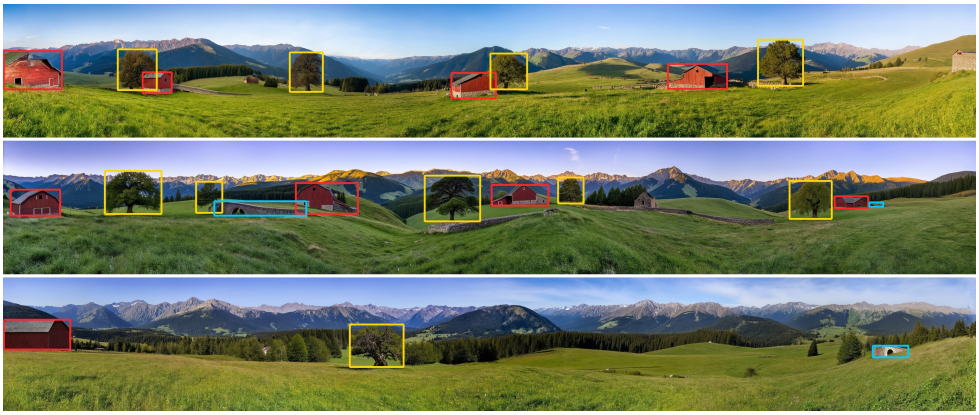


Figure 1: Same prompt, backbone (Stable Diffusion 3 medium), size (5120×704) *a green alpine valley in late afternoon with a red wooden barn on the far left, a single old oak tree in the middle and a small stone bridge on the far right*. SyncDiffusion (top) and CoFi (middle) each produce one seamless scene, but draw several barns and oak-like trees, and the bridge several times or not at all. HLC (bottom) draws one barn at the far left, one oak just left of the middle and one stone bridge at the far right. Boxes mark every instance of a prompted object: barn (red), tree (yellow), bridge (cyan).

1 INTRODUCTION

A pretrained diffusion model draws a sample of fixed size: an image of a given width, a clip of a few seconds, a trajectory of a few hundred steps. Compositional diffusion stretches such a model to horizons it was never trained on by denoising many overlapping windows jointly and fusing them

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

into one output (Bar-Tal et al., 2023). A line of work has made these outputs seamless and globally coherent: windows are synchronised in style (Lee et al., 2023), guided by a low-resolution global structure (Du et al., 2024), searched and pruned for consistency (Mishra et al., 2026), or refined from a shared coarse scaffold (Park et al., 2026). The resulting panoramas and long videos look like one continuous scene.

They rarely mean what the prompt says. Ask for *a red barn on the far left, a single old oak in the middle and a small stone bridge on the far right* and a state-of-the-art sampler returns a coherent valley with four barns, four or five trees and bridges scattered across the width (Figure 1). The failure is not a lack of coherence; it is that every window receives the same condition, so whatever the condition asks for once is asked for in every window. Better fusion, synchronisation or search cannot remove it, because the target distribution itself replicates the condition.

One remedy is to split the condition before sampling: a human or a language model writes one prompt per window (Lin et al., 2024a; Lian et al., 2024). This removes the replication but moves the problem into a planner that never sees the image: independently written window prompts drift in lighting, horizon and scene type (§4.2). Closest to our goal, AccDiffusion (Lin et al., 2024b) observed, for patch-wise image upscaling, that an identical prompt in every patch repeats objects, and keeps for each patch only the words whose thresholded cross-attention on a low-resolution image covers enough of it. This is a step beyond coherence towards semantics, but the selection is a binary choice of single words, which separates a noun from its modifiers (Appendix E), and it is made once, for enlarging one image rather than for horizons many windows long. We ask instead:

Can the pretrained model itself decide which part of the condition each window is responsible for?

Our answer rests on one observation: when the pretrained model draws the whole horizon at once, compressed into a single window, it already decides where each part of the condition goes, and its attention records the decision, because the latent positions that realise a part of the condition attend to the coordinates of the condition embedding $c = (c_1, \dots, c_M)$ that encode it (Hertz et al., 2023; Tang et al., 2023; Helbling et al., 2025). We build *Hierarchical Latent Cascades* (HLC) on this observation with two components: (i) **latent decomposition**, which measures on the coarse view how much attention each coordinate of c receives from every span of the horizon and gives each window a condition whose coordinates are reweighted by their share of the attention on that window's span, so that the scene and the style keep their full weight everywhere while a part used in one span only is attenuated elsewhere; and (ii) **hierarchical cascading**, which grows the output through a tree of native-size windows over the horizon axis, where every window starts from its parent's coarse estimate, receives its decomposed condition, and is aggregated with its siblings on their overlaps. The condition is decomposed once, on the root, and every window of the tree inherits the weights of the leaves it covers, so the global placement is decided at the top and the local detail at the bottom; the depth of the tree rather than the model sets the length, since depth d reaches 2^d native windows (the main experiments use depth two for images and videos and up to five for planning; a length study on panoramas goes to depth four). HLC is training-free, needs no detector or planner, and uses only the model's own denoiser and attention, so it applies to any diffusion model whose condition can be decomposed.

We evaluate HLC where the problem is clearest, on panoramas with Stable Diffusion 3 (Esser et al., 2024), against SyncDiffusion, CoFi and a language-model planner, and show that the same cascade carries over to long videos and to long-horizon manipulation plans. Our contributions are: (i) we identify global semantics as a failure mode of compositional diffusion that coherence-oriented samplers do not address, and measure it with count, position and event metrics across three domains; (ii) we propose HLC, a training-free sampler that decomposes the condition with the model's own attention on a coarse view and refines it through a tree of native windows; (iii) we show on panoramas, long videos and robot plans that it keeps much of the global semantics without a planner, and characterise where it fails.

2 Related work

Diffusion models. Denoising diffusion models (Ho et al., 2020) corrupt data x_0 with Gaussian noise, $x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon$ with $\epsilon \sim \mathcal{N}(0, I)$, and train a network to predict the noise, $\min_\theta \mathbb{E}\,\|\epsilon - \epsilon_\theta(x_t, t, c)\|^2$, under a condition c; sampling inverts the corruption step by step. Flow

matching (Lipman et al., 2023; Liu et al., 2023) instead interpolates linearly, $x_t = (1-t)\,x_0 + t\,\epsilon$, and regresses the velocity, $\min_\theta \mathbb{E}\,\|(\epsilon - x_0) - v_\theta(x_t, t, c)\|^2$, integrating the learned ODE from noise to data. In both cases the model gives, at every step, a clean estimate $\hat{x}_0$ of the output. Models of this kind generate images (Rombach et al., 2022; Esser et al., 2024), videos (Yang et al., 2025; Wan Team, 2025) and robot trajectories (Janner et al., 2022; Chen et al., 2024) of high quality.

Diffusion transformers. Current image and video generators are diffusion transformers (Peebles & Xie, 2023): the latent is split into patch tokens, and the condition enters through attention (Vaswani et al., 2017), either as cross-attention from latent to text tokens or as joint attention over their concatenation. Latent token p attends to condition token m with weight $a_{pm} = \mathrm{softmax}_m(q_p^\top k_m/\sqrt{d})$, which sets how much of token m reaches position p. They localise concepts: the image-to-text attention of multimodal DiTs gives zero-shot segmentation (Helbling et al., 2025; Kim et al., 2025), and in video DiTs it localises concepts in time, although only some heads are temporally informative (Jun et al., 2026). The layout is fixed early in sampling (Choi et al., 2022; Wang & Vastola, 2023), so a coarse sample already carries the model's placement of each part of the condition. Re-weighting or replacing this attention changes the matching content in a controlled way, in U-Nets (Hertz et al., 2023; Chefer et al., 2023) and in DiTs (Shin et al., 2025; Avrahami et al., 2025). We use the same attention in the other direction: we read the spans off a coarse sample instead of imposing them, and re-weight a_{pm} per window accordingly.

Compositional diffusion for long-horizon generation. Outputs beyond a model's native size are usually built by denoising overlapping windows and fusing their updates, for panoramas (Bar-Tal et al., 2023; Lee et al., 2023), high-resolution images (Du et al., 2024; He et al., 2024; Zhang et al., 2024), long videos (Wang et al., 2023; Qiu et al., 2024; Kim et al., 2024) and long-horizon plans (Janner et al., 2022; Chen et al., 2024). Follow-up work improves the global coherence of the composed output, with synchronisation gradients (Lee et al., 2023), population search (Mishra et al., 2026) or a shared coarse scaffold (Park et al., 2026). In all of these, every window receives the same global condition, so they improve how windows agree but not what each window is asked to generate. Two lines of work change the per-window condition: Lin et al. (2024b) reduce object repetition in image upscaling with patch-wise prompts taken from a low-resolution image, and language-model planners write per-region or per-scene prompts and layouts (Lin et al., 2024a; Lian et al., 2024; Long et al., 2024), which avoids replication but cannot see the generated content and loses coherence (§4.2). HLC derives the per-window conditions from the model's own attention on a coarse view, once at the root, and applies the same procedure across domains.

3 METHOD

Problem setup. A pretrained model $p_\theta(x \mid c)$ generates a window x of fixed length along a horizon (image width, video time, trajectory steps) from a condition embedding c, a sequence of token embeddings. We want an output Y over the horizon $[0,1]$ that is many windows long; A *span* $I \subset [0,1]$ is an interval of the horizon, the part of the output that one window covers (a range of image columns, of video frames or of trajectory steps); R_I crops Y to the span I and resizes it to one window. The n leaf windows of the tree split the horizon into spans $I_1, \ldots, I_n$. Compositional samplers give every window the same c. We give each window v its own share of the condition, c_v, and tie it to a coarser estimate of the whole output, $\bar{y}_v$:

$$p(Y \mid c) \;\propto\; \prod_v p_\theta\big(R_{I_v} Y \,\big|\, c_v,\; \bar{y}_v\big) \tag{1}$$

HLC obtains the two factors from the model itself. *Latent decomposition* (§3.1, Figure 2) reads the shares c_v off the attention of one coarse view of the whole output, and *hierarchical cascading* (§3.2, Figure 3) supplies $\bar{y}_v$ and joins the windows into one sample. Figure 4 combines the two.

3.1 LATENT DECOMPOSITION

A diffusion model that sees the whole horizon compressed into one window already decides where each part of the condition acts: the latent positions that draw the barn attend to the coordinates of c that encode the barn. We generate this coarse view, the *root*, read off its attention which coordinates of c each span uses, and reweight c for every window accordingly.

Placement scores. Let $c = (c_1, \ldots, c_M)$ be the coordinates of the condition embedding (for text, the token embeddings). The root covers the whole horizon, $I_{v_0} = [0, 1]$, and is generated under the full c. Its clean estimate $\hat{y}$ is re-noised to a few levels t, and we measure how much the latent positions of each leaf span I_k attend to coordinate m:

$$s_{km} = \mathbb{E}_t \sum_{p \in I_k} a^t_{pm} \qquad \bar{s}_{km} = \frac{s_{km}}{\frac{1}{n}\sum_{k'=1}^{n} s_{k'm}} \tag{2}$$

where a^t_{pm} is the attention from latent position p to coordinate m, summed over layers and heads, and n is the number of leaf spans. $\bar{s}_{km}$ compares span k with the average span. Coordinates that describe the whole output, such as the scene or the style, are used about equally everywhere ($\bar{s}_{km} \approx 1$); a coordinate that the root used in one place scores well above 1 there.

From scores to per-window weights. We turn the scores into weights $w_{km} \in [0, 1]$ in three steps. *(i) Noise level.* Coordinates that describe the whole output, such as the scene or the style, should score 1 in every span; their scores still fluctuate from span to span, and the standard deviation of these fluctuations, σ, measures the noise of the score. *(ii) Standardised score.* We express how far span k exceeds the average span in units of this noise (z_{km} below). *(iii) Weight.* Two thresholds $z_0 < z_1$, both in noise units, turn the standardised score into a weight: below z_0 the weight is 0, above z_1 it is 1, and in between it rises linearly:

$$z_{km} = \frac{\bar{s}_{km} - 1}{\sigma} \qquad w_{km} = \mathrm{clip}\left(\frac{z_{km} - z_0}{z_1 - z_0},\, 0,\, 1\right) \tag{3}$$

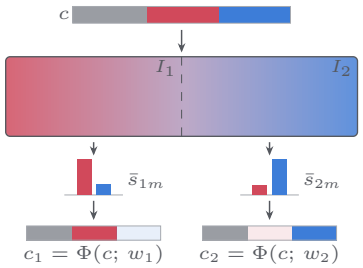


Figure 2: **Latent decomposition.** The coarse view's attention to the coordinates of c, summed over each window, reweight the c of each window w_k.

A span where a coordinate is within the noise ($z_{km} < z_0$) ignores it, a span where the root clearly used it ($z_{km} > z_1$) attends to it fully, and a span in between attends to it partially. Every coordinate keeps $w_{km} = 1$ in its highest-scoring span, so that nothing is dropped from the output, and coordinates that no span singles out (the scene, the style) keep $w_{km} = 1$ everywhere. The two thresholds are set once from score statistics (Appendix A). The weights are computed for the leaf spans; a window higher in the tree covers several leaf spans and receives a coordinate as strongly as the leaf that needs it most, $w_{vm} = \max_{k \subset v} w_{km}$, so that an object placed in any of its leaves is already present in the coarser window that the leaf is refined from.

Reweighted condition. Window k keeps the same embedding c, but sees it through its weights w_k. We write $c_k = \Phi(c; w_k)$ for this reweighted condition, where Φ is our shorthand for the following operation: in every attention layer, the attention a_{pm} from latent position p to coordinate m is multiplied by w_{km} and renormalised, which replaces it by

$$\tilde{a}_{pm} = \frac{w_{km}\, a_{pm}}{\sum_{m'} w_{km'}\, a_{pm'}} \tag{4}$$

where only the latent-to-condition part of the attention is changed. Reweighting attention changes how strongly a part of the condition shapes the output, as observed for image editing (Hertz et al., 2023; Shin et al., 2025). The coordinates need not be interpretable on their own: text encoders contextualise tokens, so content spreads over neighbouring coordinates, and in our experiments we share one weight among the tokens of a phrase (Appendix B).

3.2 Hierarchical Cascading

Windows that share only a condition do not agree on one scene; windows that refine the same coarse picture do. We therefore grow the output through a binary tree of windows, all of native size: each level doubles the resolution of the horizon, every window starts from its parent's estimate of its span, and siblings are denoised together.

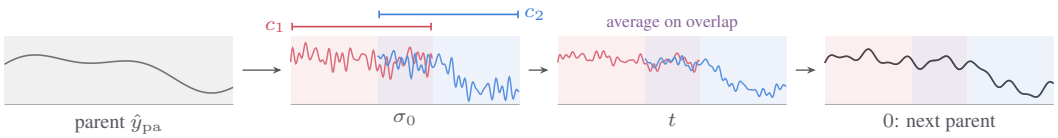


Figure 3: **Aggregation within one level of the cascade**, shown for two windows along a generic horizon. Each window starts from its crop of the parent estimate, low-passed and noised to σ_0; at every step it predicts a clean estimate under its own condition (red: c_1, blue: c_2), and the two estimates are averaged on the overlap, so they converge to one output. This merged output is the parent of the next, finer level, whose windows again start from its low-passed, noised crops.

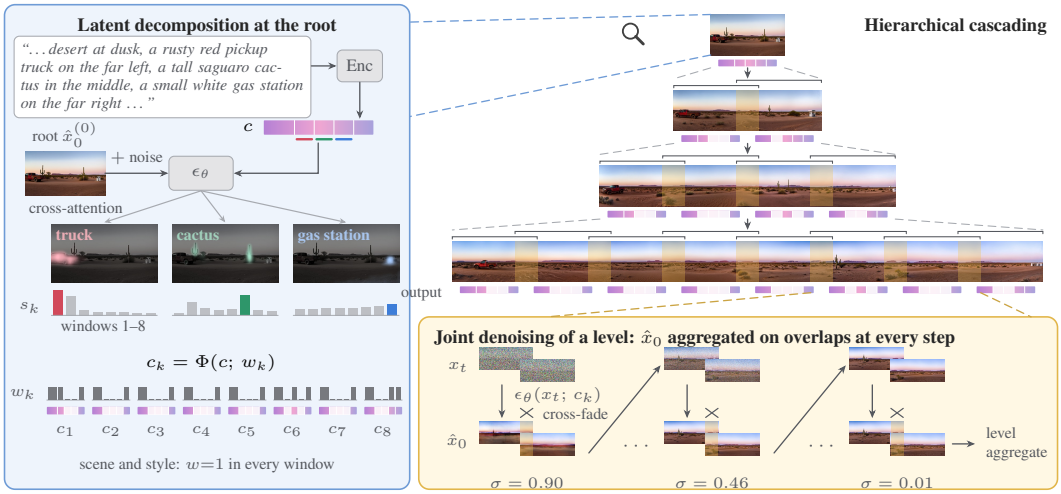


Figure 4: **Overview of HLC** (SD3, desert prompt). *Left:* a forward pass on the root reveals, through its cross-attention, where each part of the condition acts; each window k attends to the coordinates of c with weights w_k, i.e. sees the reweighted condition $c_k = \Phi(c; w_k)$. *Right:* at every level, the windows are denoised jointly from their parent's low-passed estimate, with their $\hat{x}_0$ cross-faded on the shared overlaps at every step; the leaf level's aggregate is the output.

Coarse-to-fine initialisation. A child window v starts from its parent's picture of its span. We take the parent's clean estimate, cut out the child's span, resize it to one window and keep only its low frequencies. We call this coarse picture $\bar{y}_v$. The child starts from it, noised to an intermediate level $\sigma_0 \in (0, 1]$ with Gaussian noise ϵ_v, as in the forward process of flow matching:

$$x_v = (1 - \sigma_0)\, \bar{y}_v + \sigma_0\, \epsilon_v \tag{5}$$

As in SDEdit (Meng et al., 2022), the child keeps the parent's layout and re-creates the detail: the low-pass removes what the stretch has blurred, and σ_0 sets how far the child may depart from its parent. The noise ϵ_v is cropped from one noise map of the level, so overlapping windows start from the same state.

Joint denoising with overlap averaging. At every step, each window predicts its clean estimate $\hat{x}_v$ under its own condition c_v, and the estimates of a level are averaged on their overlaps,

$$\hat{y}^{(\ell)} = \frac{\sum_v R_{I_v}^\top \left(\rho \odot \hat{x}_v\right)}{\sum_v R_{I_v}^\top \rho} \tag{6}$$

where ρ is a per-position weight inside a window that decays towards the window borders, $\odot$ is the element-wise product, and the transpose $R^\top$ of the crop places a window back at its span of the full output (zero elsewhere). The numerator adds up the weighted estimates of all windows at every position, and the denominator adds up their weights, so every position of $\hat{y}^{(\ell)}$ is the weighted mean

of the windows that cover it (Bar-Tal et al., 2023). Each window then takes its solver step from its crop of $\hat{y}^{(\ell)}$. Overlapping windows thus see one estimate of the shared region and cannot drift apart, while their different c_v let them draw different content. The final $\hat{y}^{(\ell)}$ is the parent estimate of the next level, and that of the leaves is Y. Only the horizon, the crop and the low-pass filter depend on the domain.

4 EXPERIMENTS

4.1 SETUP

We test HLC in three domains in the following settings; domain-specific adaptations, hyperparameters, baselines and metrics are given in Appendix B.

- **Panoramas.** Stable Diffusion 3 medium (Esser et al., 2024); windows of 1280×704, outputs of 5120×704 (depth 2); 25 prompts with 2–3 objects at stated positions (Appendix C).
- **Videos.** Wan 2.2 TI2V-5B with its 4-step distilled sampler (Wan Team, 2025); windows of 121 frames at 1280×704, outputs of 493 frames (depth 2); stories of a scene and three events, without time words.
- **Planning.** A 9M-parameter trajectory diffusion model trained on OGBench cube-quadruple (Park et al., 2025) with windows of at most 768 environment steps; lists of 2 and 4 instructions over 1536 and 3072 steps, executed in MuJoCo.

The main measure is *global semantics*: whether each requested object or event appears the requested number of times, at the requested place or time. Coherence and quality metrics are a secondary check; we do not use prompt-alignment scores such as CLIP-S, which reward a window that repeats every object (Lee et al., 2023).

4.2 PANORAMA IMAGE GENERATION

The last two rows of Table 1 are an ablation: the first is our own tree and root with the full prompt in every window. Without decomposition, the cascade never satisfies a prompt, and its count grows with the number of windows (Appendix D); decomposition is what removes the replication, at the cost of a less uniform style (Intra-Style-L 25.4 $\rightarrow$ 33.0). SyncDiffusion and CoFi, with their own flat windows, fail in the same way, although their panoramas are seamless. The LLM planner sits at the other end: it tells each window which objects it contains, so its semantic scores are an upper reference for what per-window conditions can achieve; the price is on the right of the table, where its windows, drawn from unrelated prompts, form collages with a different horizon or sky per window (Intra-Style-L 62.2; Figure 5). HLC approaches the planner on global semantics without being told where anything goes (objects 0.63 / 0.81 vs. 0.84 / 0.87 and counts 0.81 / 0.87 vs. 0.84 / 0.88, detector / VLM), while its coherence and quality stay close to those of the coherence-oriented samplers: slightly below CoFi, but without the collapse of the planner and without a visible loss of quality. More HLC panoramas are in Figure 8.

Table 1: Panoramas (SD3-medium, 5120×704, 25 prompts, 67 objects). Rates by detector / VLM judge; an object is correct if its count and position match; leakage: objects that also appear far from their stated position. "w/o decomposition" is our tree and root with the full prompt in every window, averaged on overlaps as in MultiDiffusion (Bar-Tal et al., 2023); SyncDiffusion and CoFi use their own flat windows. †external language-model planner.

Method	Global semantics (detector / VLM)				Coherence and quality		
	Success ↑	Objects ↑	Count ↑	Leakage ↓	I-LPIPS ↓	I-Style ↓	NIQE ↓
Full prompt in every window (flat windows)							
SyncDiffusion (NeurIPS'23)	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	1.00 / 1.00	0.475	25.3	4.03
CoFi (NeurIPS'26)	0.00 / 0.00	0.00 / 0.00	0.01 / 0.01	1.00 / 1.00	0.493	24.7	3.72
External planner writes one prompt per window							
LLM planner†	*0.56 / 0.64*	*0.84 / 0.87*	*0.84 / 0.88*	*0.00 / 0.01*	0.579	62.2	4.26
HLC w/o decomposition (MultiDiffusion averaging)	0.00 / 0.00	0.00 / 0.00	0.00 / 0.00	0.99 / 1.00	0.473	25.4	4.00
HLC (ours)	0.24 / 0.48	0.63 / 0.81	0.81 / 0.87	0.21 / 0.06	0.482	33.0	4.17



Figure 5: Panoramas for a position prompt (left: *an alpine meadow; a red vintage car on the far left, a white hot air balloon in the middle, a yellow tent on the far right*) and a counting prompt (right: *a clear morning sky over rolling farmland, four hot air balloons*). CoFi repeats every object in every window (about 13 balloons); the planner's windows have different horizons and skies and draw three balloons; HLC keeps one scene, places each object once on the left and draws exactly four balloons on the right.

4.3 Long video generation

Samplers that give every window the full prompt tell the story again in every window (4.8–5.3 appearances per event; Table 2). Sampling all 493 frames in one pass, four times the model's trained length, keeps one scene and scores best on consistency, but four of the six events never happen: without windows, the model has no room to stage the story in time. Given the events with time spans, Gen-L-Video's multi-prompt mode gets the order right but changes the scene at every switch. HLC shows five of the six events exactly once and both stories in order, without time spans, at similar consistency and quality.

Table 2: Long videos (Wan 2.2 TI2V-5B, 493 frames, two stories × three events). Once / Missing: events that appear exactly once / never; Order: stories whose events start in the stated order; Appear.: separate appearances per event, averaged over all six events with a missing event counted as 0 (in one pass the cyclist appears seven separate times). FreeNoise's multi-prompt mode fails the order because its rescheduled noise repeats the first window's content, so later events already appear early. One pass: all 493 frames sampled at once. Right: VBench-style scores. †receives the events in order with time spans.

Method	Once ↑	Missing ↓	Order ↑	Appear. → 1	Subject ↑	Flicker ↑	Aesthetic ↑
One pass, 493 frames	1/6	4/6	0/2	1.33	**0.924**	**0.997**	0.414
Gen-L-Video	0/6	**0/6**	**2/2**	5.33	0.649	0.983	0.410
Gen-L-Video, multi-prompt†	4/6	**0/6**	**2/2**	1.83	0.725	0.984	0.436
FreeNoise (ICLR'24)	0/6	**0/6**	1/2	4.83	0.794	0.986	**0.439**
FreeNoise, multi-prompt†	0/6	**0/6**	0/2	4.33	0.779	0.986	0.434
HLC (ours)	**5/6**	**0/6**	**2/2**	**1.17**	0.752	0.989	0.438



Figure 6: **Long video generation** with HLC on a stronger backbone, LTX-2.5 (HaCohen et al., 2024) (distilled, 8 steps; 481 frames at 24 fps and 1280 × 704; one level of four windows below the root; root chosen from two seeds by eye). Prompt: *A static shot of a wooden park bench on a green lawn, a grey pigeon lands on the bench, a brown dog walks past in front of the bench, a child holding a red balloon walks by, the camera never moves.* The pigeon lands, the dog walks past and leaves, and the child with the red balloon walks by, each once, on one bench. Children are conditioned on the root frames at their boundaries (Appendix B).

4.4 LONG-HORIZON MANIPULATION PLANNING

The same cascade also works for robot plans: the root draws the whole task in one window at a trained stride and the cascade refines it to the execution rate, so HLC plans two and four times beyond the longest trained window (Table 3, Figure 7). It outperforms start–goal conditioning, which gives the windows only the start and the goal state and therefore cannot express intermediate instructions or a cube that must not move, and broadcast, which gives every window the whole list and reorders the instructions (order agreement 0.65 vs. 0.86, mean over the 3072-step cells). Both use the same tree as HLC, so the gap to broadcast isolates the decomposition. CoFi and CDGS (Park et al., 2026; Mishra et al., 2026) condition on the start and goal only, like the start–goal baseline; their coherence mechanisms are orthogonal to what each window is asked to do. The tree does not improve on the root itself here, since these tasks fit one coarse window: executing the root alone gives 0.85 / 0.75, 0.89 / 0.78, 0.69 / 0.30 and 0.70 / 0.28 in the four cells of Table 3. The planning algorithm is given in Algorithm 2 (Appendix B).

Table 3: Manipulation planning beyond the trained horizon (OGBench cube-quadruple; trained windows cover at most 768 steps). Instruction success / all instructions succeeded, executed in MuJoCo; 20 lists × 3 seeds per cell. All methods use the same tree; only the condition of each window differs.

	2 instructions		4 instructions	
Method	1536 steps	3072 steps	1536 steps	3072 steps
Start–goal (same tree)	0.72 / 0.52	0.63 / 0.47	0.36 / 0.03	0.34 / 0.03
Broadcast	0.82 / 0.63	0.72 / 0.55	0.56 / 0.17	0.56 / 0.20
HLC (ours)	**0.84 / 0.68**	**0.84 / 0.70**	**0.70 / 0.30**	**0.67 / 0.23**

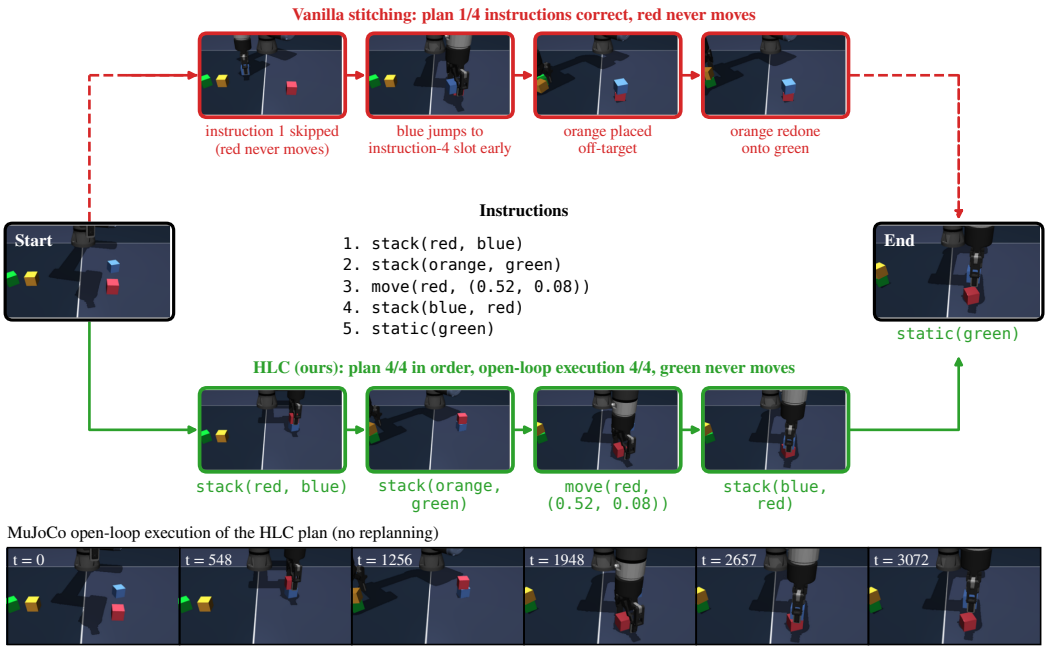


Figure 7: **Instruction-list planning** over 3072 steps, four times the longest trained window (one list of four instructions and a cube that must not move). Start–goal conditioning (red) sees only the start and goal states: it never moves red, moves blue early and places orange twice (1 of 4 instructions). HLC (green) completes the four instructions once each and in order, in the plan and in open-loop MuJoCo execution (bottom), and never moves green.

4.5 Analysis

Where HLC fails. The root cannot generate the final output at its full length, but it does read the condition and decide how each window of the horizon should look, for horizons far longer than it was trained on: two to four times the trained window in planning, four times in video and up to 1:29 in panoramas (Table 5). This reading has limits. When the horizon becomes too long, the coarse view is too compressed for its layout to be refined without visible artifacts (Figure 9); and when the condition is complex, with parts that are strongly entangled (several objects of one kind, events that share a subject), the root's attention does not separate them and the decomposition fails, which shows up as wrong counts and events drawn together.

Inference cost. Compared with sampling only the N leaf windows for T steps, the tree adds the root and the inner levels, and every child starts at $\sigma_0 \leq 1$. If the number of windows at least doubles per level, the inner levels together have fewer windows than the leaves, so

$$\frac{\mathrm{NFE}_{\mathrm{HLC}}}{\mathrm{NFE}_{\mathrm{leaves}}} \;\leq\; \frac{T + \sigma_0 T\,(2N-2)}{N\,T} \;\leq\; 2 - \frac{1}{N} \;<\; 2 \tag{7}$$

plus a few passes on the root to read the scores. The ratio is 1.6 for our panoramas ($N = 4$, $\sigma_0 = 0.9$; 21 s per panorama on one A100, against 67 s for CoFi and 498 s for SyncDiffusion), 1.4 for video and 1.95 for planning, where every level starts from pure noise.

5 Conclusion and Limitations

Work on compositional diffusion has focused on global coherence, and it has become good at it; whether the composed output still means what the condition says has received much less attention. Repeated objects have been noted and reduced in specific settings, such as image upscaling (Lin et al., 2024b), and multi-event videos can follow prompts with user-given time spans (Cai et al., 2025; Xu et al., 2026), but global semantics has not been treated as a goal of compositional sampling in its own right. This paper takes it up as a problem: we measure it across panoramas, long videos and robot plans, find that samplers that give every window the full condition fail it almost completely, and show that a single pretrained diffusion model can already do much better by reading its own coarse view. Its main limitation is the root: when the horizon becomes very long or the parts of the condition are strongly entangled, the coarse view cannot be decomposed reliably, and counts, which a window-level weight cannot divide, are controlled only as far as the root and a single window get them right (§4.5). In addition, although the algorithm is domain-agnostic, each domain needs a few settings to be tuned for good results, such as the start level of the children, the low-pass filter and the way children are tied to their parent (Appendix B). Our evaluation is also small: one seed per prompt, 25 panorama prompts, two video stories and 20 instruction lists per planning cell. HLC shows that a single pretrained diffusion model, without a planner, can already keep much of the global semantics that compositional samplers lose, and we hope that global semantics becomes a standard axis of evaluation next to coherence.

AI use statement

We used generative AI tools (Claude, Anthropic) in supporting roles: surveying prior work at the start of the project, writing part of the experiment code from our specifications, drafting parts of the appendix, and polishing the writing. The key idea of the method and the problem statement are our own. By design, the LLM-planner baseline (§4.2) uses a language model to write its per-window prompts. We checked every suggested reference against the original paper, read and tested all AI-written code, and traced every reported number to the logged outputs; we take full responsibility for the content of this paper.

Ethics statement

This work studies sampling algorithms for publicly released generative models (Stable Diffusion 3 Medium, Wan 2.2 TI2V-5B with a distilled four-step version, and LTX-2.5), used under their licenses, and a trajectory model we trained on the public OGBench data; evaluation uses the public

OWLv2 detector and Qwen2.5-VL-7B. Prompts are synthetic and describe landscapes and everyday scenes. No human subjects or personal data are involved. The method inherits the risks of the underlying models, including misuse for misleading content; it adds no new generative capability beyond longer outputs.

Reproducibility statement

The full algorithm is given in Algorithm 1 (Appendix A); domain-specific choices and all hyperparameters are listed in Appendix B. The image and video models are public checkpoints used without fine-tuning; the planning model is trained by us as described in Appendix B. Code for all three domains is available at `https://anonymous.4open.science/r/anonymous-repo-for-review-3E50/README.md`.

References

Omri Avrahami, Or Patashnik, Ohad Fried, Egor Nemchinov, Kfir Aberman, Dani Lischinski, and Daniel Cohen-Or. Stable flow: Vital layers for training-free image editing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. MultiDiffusion: Fusing diffusion paths for controlled image generation. In *International Conference on Machine Learning (ICML)*, 2023.

Minghong Cai, Xiaodong Cun, Xiaoyu Li, Wenze Liu, Zhaoyang Zhang, Yong Zhang, Ying Shan, and Xiangyu Yue. DiTCtrl: Exploring attention control in multi-modal diffusion transformer for tuning-free multi-prompt longer video generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. *ACM Transactions on Graphics (SIGGRAPH)*, 2023. arXiv:2301.13826.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. arXiv:2204.00227.

Ruoyi Du, Dongliang Chang, Timothy Hospedales, Yi-Zhe Song, and Zhanyu Ma. DemoFusion: Democratising high-resolution image generation with no $$$. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning (ICML)*, 2024.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, Poriya Panet, Sapir Weissbuch, Victor Kulikov, Yaki Bitterman, Zeev Melumian, and Ofir Bibi. LTX-Video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

Yingqing He, Shaoshu Yang, Haoxin Chen, Xiaodong Cun, Menghan Xia, Yong Zhang, Xintao Wang, Ran He, Qifeng Chen, and Ying Shan. ScaleCrafter: Tuning-free higher-resolution visual generation with diffusion models. In *International Conference on Learning Representations (ICLR)*, 2024.

Alec Helbling, Tuna Han Salih Meral, Ben Hoover, Pinar Yanardag, and Duen Horng Chau. ConceptAttention: Diffusion transformers learn highly interpretable features. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2502.04320.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *International Conference on Learning Representations*, 2023. arXiv:2208.01626.

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

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. arXiv:2311.17982.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning (ICML)*, 2022.

Youngjun Jun, Seil Kang, Woojung Han, and Seong Jae Hwang. I'm a Map! interpretable motion-attentive maps: Spatio-temporally localizing concepts in video diffusion transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Chaehyun Kim, Heeseong Shin, Eunbeen Hong, Heeji Yoon, Anurag Arnab, Paul Hongsuck Seo, Sunghwan Hong, and Seungryong Kim. Seg4Diff: Unveiling open-vocabulary segmentation in text-to-image diffusion transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Jihwan Kim, Junoh Kang, Jinyoung Choi, and Bohyung Han. FIFO-Diffusion: Generating infinite videos from text without training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Yuseung Lee, Kunho Kim, Hyunjin Kim, and Minhyuk Sung. SyncDiffusion: Coherent montage via synchronized joint diffusions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Long Lian, Baifeng Shi, Adam Yala, Trevor Darrell, and Boyi Li. LLM-grounded video diffusion models. In *International Conference on Learning Representations*, 2024. arXiv:2309.17444.

Han Lin, Abhay Zala, Jaemin Cho, and Mohit Bansal. VideoDirectorGPT: Consistent multi-scene video generation via LLM-guided planning. In *Conference on Language Modeling*, 2024a. arXiv:2309.15091.

Zhihang Lin, Mingbao Lin, Meng Zhao, and Rongrong Ji. AccDiffusion: An accurate method for higher-resolution image generation. In *European Conference on Computer Vision (ECCV)*, 2024b. arXiv:2407.10738.

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

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.

Fuchen Long, Zhaofan Qiu, Ting Yao, and Tao Mei. VideoStudio: Generating consistent-content and multi-scene videos. In *European Conference on Computer Vision*, 2024. arXiv:2401.01256.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2022.

Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Utkarsh A. Mishra, David He, Yongxin Chen, and Danfei Xu. Compositional diffusion with guided search for long-horizon planning. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2601.00126.

Anish Mittal, Rajiv Soundararajan, and Alan C. Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal Processing Letters*, 20(3):209–212, 2013.

Byoungwoo Park, Utkarsh A. Mishra, Jaemoo Choi, Juho Lee, and Yongxin Chen. Coarse-to-fine compositional diffusion for long-horizon planning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2026.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.20092.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Haonan Qiu, Menghan Xia, Yong Zhang, Yingqing He, Xintao Wang, Ying Shan, and Ziwei Liu. FreeNoise: Tuning-free longer video diffusion via noise rescheduling. In *International Conference on Learning Representations (ICLR)*, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Joonghyuk Shin, Alchan Hwang, Yujin Kim, Daneul Kim, and Jaesik Park. Exploring multimodal diffusion transformers for enhanced prompt-based image editing. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. arXiv:2508.07519.

Raphael Tang, Linqing Liu, Akshat Pandey, Zhiying Jiang, Gefei Yang, Karun Kumar, Pontus Stenetorp, Jimmy Lin, and Ferhan Ture. What the DAAM: Interpreting stable diffusion using cross attention. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023. arXiv:2210.04885.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Wan Team. Wan: Open and advanced large-scale video generative models, 2025. arXiv:2503.20314.

Binxu Wang and John J. Vastola. Diffusion models generate images like painters: an analytical theory of outline first, details later. *arXiv preprint arXiv:2303.02490*, 2023.

Fu-Yun Wang, Wenshuo Chen, Guanglu Song, Han-Jia Ye, Yu Liu, and Hongsheng Li. Gen-L-Video: Multi-text to long video generation via temporal co-denoising, 2023. arXiv:2305.18264.

Qianxun Xu, Chenxi Song, Yujun Cai, and Chi Zhang. SwitchCraft: Training-free multi-event video generation with attention controls. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Xiaotao Gu, Yuxuan Zhang, Weihan Wang, Yean Cheng, Ting Liu, Bin Xu, Yuxiao Dong, and Jie Tang. CogVideoX: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations (ICLR)*, 2025.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Shen Zhang, Zhaowei Chen, Zhenyu Zhao, Yuhao Chen, Yao Tang, and Jiajun Liang. HiDiffusion: Unlocking higher-resolution creativity and efficiency in pretrained diffusion models. In *European Conference on Computer Vision*, 2024. arXiv:2311.17528.

A SCORE STATISTICS AND PSEUDO CODE

Score statistics. On a calibration set of 24 SD3 panoramas (183 object phrases) disjoint from the prompts of Appendix C, the normalised score $\bar{s}_{km}$ (Eq. 2, summed over the tokens of the phrase; 8 leaf windows) is 1.86 at the argmax window (median; range 1.05–3.22), 1.32 at the second window, 0.84 at the median window and 0.65 at the lowest. The scene phrases, which should be uniform, spread with a standard deviation of 0.10 (0.04–0.24) over the windows; this is the noise level σ of Eq. 3. The argmax window therefore lies at a median of 8.6σ above uniform and the second window at about 3σ. With the noise level estimated per scene, the argmax window has a median z of 4.7; we set $z_0 = 2$ (a conventional two-sigma threshold) and $z_1 = 5$ (a window receives the full phrase only with the evidence of a typical object peak), before generating any evaluation image. The thresholds are in units of the noise σ, which is re-estimated for every root and leaf count, so they carry over to the four leaves of the main experiments, where the weights are binary for 64 of 67 objects. Using the raw scores as weights instead would give every window a weight of at least 0.65 and thus request every object everywhere.

Black-box score. When the attention of a model is not accessible, s_{km} can be replaced by the increase of the denoising loss on span I_k when the phrase of coordinate m is removed from the prompt, measured with the same noise for both conditions.

Hard gate. With $w \in \{0, 1\}$ (the argmax window only), window k is conditioned on the prompt with the other windows' phrases removed and re-encoded. Zeroing the phrase's token embeddings instead does not remove it, because the concept survives in the contextualised and padding tokens; re-encoding the shortened prompt conditions the model exactly on the remaining phrases.

Re-weighting in joint attention. In SD3 the text tokens are themselves updated in every joint-attention block, so Eq. 4 is applied to the image-to-text part of the attention in every block (Shin et al., 2025).

Algorithm 1 HLC (full version)

Require: model p_θ, condition $c = (c_1, \ldots, c_M)$, depth d, noise levels $\mathcal{T}$ for scoring, thresholds z_0, z_1, start level σ_0

Sample the root $\hat{y}$ over $[0, 1]$ under the full c
for $t \in \mathcal{T}$ **do**
  Re-noise $\hat{y}$ to level t, run one forward pass and record a^t_{pm} in every attention layer
end for
Compute s_{km} and $\bar{s}_{km}$ for the n leaf spans, $n = 2^d$ (Eq. 2); estimate σ from the scene coordinates
Compute z_{km} and w_{km} (Eq. 3); set $w = 1$ at each coordinate's argmax span and for scene coordinates
$\hat{y}^{(0)} \leftarrow \hat{y}$
for level $\ell = 1, \ldots, d$ **do**
  For every window v of level ℓ: $w_{vm} \leftarrow \max_{k \subset v} w_{km}$; $\bar{y}_v \leftarrow$ low-passed crop of $\hat{y}^{(\ell-1)}$ to I_v; $x_v \leftarrow (1 - \sigma_0)\bar{y}_v + \sigma_0 \epsilon_v$, with ϵ_v cropped from one noise map
  for each sampler step $t \to t'$ from σ_0 to 0 **do**
    For every v: $u_v \leftarrow v_\theta\big(x_v, t, \Phi(c; w_v)\big)$, $\hat{x}_v \leftarrow x_v - t\, u_v$, $\hat{\epsilon}_v \leftarrow x_v + (1 - t)\, u_v$
    $\hat{y}^{(\ell)} \leftarrow \sum_v R^\top_{I_v}(\rho \odot \hat{x}_v) \,/\, \sum_v R^\top_{I_v} \rho$
    For every v: $x_v \leftarrow (1 - t')\, R_{I_v} \hat{y}^{(\ell)} + t'\, \hat{\epsilon}_v$
  end for
end for
return $Y = \hat{y}^{(d)}$

B DOMAIN-SPECIFIC ADAPTATION AND HYPERPARAMETERS

Panoramas (SD3). Stable Diffusion 3 medium with its CLIP and T5 text encoders (T5 up to 256 tokens), 28 Euler steps of the flow-matching sampler, classifier-free guidance 7.0, windows of 1280×704 with a halo of 192 pixels on each shared side, branching 2, depth 2 for 5120×704 (depth 3 and 4 for the scaling study). The attention score (Eq. 2) is read at $t \in \{0.5, 0.75\}$ over all joint-attention blocks; weights are shared within the comma-separated phrases of the prompt, the scene and style phrases are context. For panoramas we use the hard gate (remove the phrase and re-encode, Appendix A): at depth two the weights of Eq. 3 are binary for 64 of the 67 objects, so the soft weights of Eq. 4 and the gate give the same routing. Children start from the parent's low-passed estimate (cut-off at the window height) at $\sigma_0 = 0.9$; siblings average $\hat{x}_0$ on their overlaps at every step with a linear ramp. A single root is used per prompt (seed 0).

Video (Wan 2.2). The two stories are *A static shot of a wooden park bench on a green lawn, a grey pigeon lands on the bench, a brown dog walks past in front of the bench, a child holding a red balloon walks by, the camera never moves* and *A static shot of a quiet street corner with a crosswalk, a red car drives past, a cyclist rides by on a bicycle, a yellow bus pulls up at the curb, the camera never moves*, one seed each. Wan 2.2 TI2V-5B with its 4-step distilled sampler; windows of 31 latent (121 pixel) frames at 1280×704, depth 2. Scores are read per latent frame. For events, Eq. 2 uses only each event's distinguishing tokens (the words of its phrase that occur in no other event phrase); each token's attention is normalised over the frames and the mean profile of the other events is subtracted, a contrastive score, because the plain attention to the three events rises and falls together. A child starts from its parent's frames, stretched in time by linear interpolation in pixel space, colour-matched to the root and re-encoded, at the second sampler level; it is anchored to the parent at its last frame (released after one call) and receives its left neighbour's last decoded frame as the image condition of its first frame; windows are decoded separately and concatenated.

Video (LTX-2.5, Figure 6). LTX-2.5 with its distilled transformer (8 steps, no classifier-free guidance); root and windows of 121 frames at 1280×704 and 24 fps. The root is one clip of the whole story; the scores of Eq. 2 are read from the text cross-attention of all 48 blocks at six noise levels on the re-noised root, with the contrastive per-event score of the Wan setting (above). The four windows each span a quarter of the root and are conditioned on the root frames at their two boundaries (image conditions at the first and last frame) and on their routed text; neighbouring windows share the boundary frame. The root seed was chosen from two by eye. On one A100 the root takes about 35 s and each window about 45 s.

Manipulation planning: model and training. We train our own window model on the OGBench cube-quadruple play data (Park et al., 2025) (train and validation episodes, a few episodes in which a cube leaves the table removed). A plan state is 27-dimensional (effector position and yaw, gripper opening, contact flag and the four cube poses). Events are labelled automatically: a cube that is grasped (effector within 4.5 cm and in contact) and released becomes a *move* atom with its release point, or a *stack* atom if it is released on another cube; a cube that is never grasped in a window may be labelled *static*. Training windows have 97 frames at strides 1, 2, 4 and 8 (96 to 768 environment steps), sampled with probabilities 0.15, 0.15, 0.2 and 0.5. The condition is a set of tokens: a mode token (*complete*: the listed atoms are all releases in the window; *empty*: no release; *partial*: a subset; *null*: condition dropped) and one token per atom, embedding its type, cube(s), target point and region and, optionally, its release time within the window. During training each event atom is dropped with probability 0.2 (mode *partial*), the whole condition with probability 0.1 (*null*), release times with probability 0.3, and each static token is shown with probability 0.5; random frames are given as known frames. The denoiser is a transformer over the 97 frames (width 256, depth 6, 8 heads; 9.1M parameters) with adaLN-Zero modulation by the diffusion step and the stride, cross-attention from frames to condition tokens, and known frames concatenated to the input. It is a DDPM with 512 cosine-scheduled steps and v-prediction, trained with AdamW (learning rate 2×10^{-4}, weight decay 0.01, batch 512, 3000 warm-up steps, cosine decay, gradient clipping at 1.0) for 300k steps (7.5 hours on one A100), and sampled with DDIM ($\eta = 1$, 50 steps) from its EMA weights. Execution uses an inverse-dynamics MLP (three layers of 512) from consecutive stride-1 states to the 5-dimensional action, trained on the same data.

Manipulation planning: how HLC is adapted. We describe the planning instantiation step by step (Algorithm 2).

1. *Condition.* The condition is the list of instructions. Each instruction (*move*, *stack*) is one condition token, and each cube that must not move is a *static* token. Static tokens describe the whole plan, like the scene phrase of a panorama, and are given to every window.
2. *Stride.* A window always has 97 frames; its *stride* is the number of environment steps between two frames. A horizon of L steps is covered at stride $L/96$ by a single window; for $L = 1536$ and 3072 this stride (16 or 32) is longer than any stride the model was trained on (at most 8), and we call it a *virtual* stride.
3. *Root.* The root is one window at the virtual stride. Since the model has no embedding for that stride, it is drawn with the embedding of a trained stride, chosen so that a trained window of that stride contains about as many cube releases as the list has instructions (stride 2 for two instructions, 4 for four); the result is read as a compressed plan of the whole horizon. We draw four such roots and keep the one whose instruction times (next step) follow the list order best, measured by Kendall's rank agreement.
4. *Placement (decomposition).* For each instruction we read the root's cross-attention from all 97 frames to its token; the frame where it peaks is the time at which the root performs the instruction, written as a fraction $u_j \in [0, 1]$ of the horizon. This is the planning counterpart of Eqs. 2–3 with the hard gate: every instruction is given to exactly one window per level, the one whose centre is closest to u_j, together with its time inside that window. A window that receives no instruction is given the mode *empty*, a label the model was trained with that means "no cube is released in this window"; a window may receive several instructions at coarse levels.
5. *Levels.* Each level halves the stride (e.g. $32 \to 16 \to \cdots \to 1$ for 3072 steps, five levels below the root) and lays windows that overlap by half. Because the root's motion is stretched, a level at stride s moves more slowly than data at stride s; it therefore uses the embedding of the trained stride closest to its effective speed $s \cdot s_{\mathrm{root}}/R$, where s_{root} is the root's embedding stride and R the virtual stride.
6. *Coherence.* Instead of starting a child from a noised copy of its parent (Eq. 5), the parent plan is stretched by two (linear interpolation) and every second frame of the child level is fixed to it; the child fills the frames in between. Windows are denoised jointly (DDIM, 50 steps) and their predictions averaged on the overlaps with triangular weights at every step (Eq. 6). With these anchors the plan keeps the root's events, and the levels add the fine motion needed for execution.
7. *Execution and evaluation.* The stride-1 plan is executed open loop in MuJoCo by the inverse-dynamics MLP. An instruction succeeds if its event is found in the executed trajectory: a release of the cube within 4 cm of the target point for *move*, the cube released on top of the other cube for *stack*, and a displacement below 4 cm without ever being grasped for *a cube never moves*; order agreement is the Kendall rank agreement of the matched event times with the list order.

Baselines and metrics. *Panoramas.* SyncDiffusion and CoFi are ported to SD3 from the official code with their default settings. The ablation *HLC w/o decomposition* is our tree and root with the full prompt in every window, averaged on the overlaps as in MultiDiffusion (Bar-Tal et al., 2023). *SyncDiffusion* (Lee et al., 2023) adds, at early steps, a gradient that pulls every window's predicted image towards an anchor window in LPIPS. *CoFi* (Park et al., 2026), the most recent compositional sampler, which reports panorama and long-video results in addition to planning, builds a coarse scaffold by denoising windows pulled towards their mean, then re-noises it with shared noise and refines it. The *LLM planner* has a language model (Claude) write one prompt per window from the full prompt, and the windows are sampled jointly with overlap averaging (as in Lin et al., 2024a). The panorama metrics are defined in Appendix C. *Videos.* Ported to Wan from the official code. *Gen-L-Video* (Wang et al., 2023) denoises overlapping temporal windows jointly and averages them. *FreeNoise* (Qiu et al., 2024) reuses shuffled copies of one window's initial noise and fuses windowed attention. Each receives the full prompt in every window and, where supported, a multi-prompt mode with the events in order over equal thirds of the clip; we also sample all 493 frames in one pass. Per event we count separate appearances over the whole video and check the order of first appearances; VBench-style scores (Huang et al., 2024) use our implementation. *Planning.* Instructions are *move a cube to a point*, *stack one cube on another* and *a cube never moves*; 20 lists per length and horizon, 3 seeds. *Start–goal conditioning* composes windows conditioned only on

Algorithm 2 HLC for instruction-list planning (one list, horizon L environment steps)

Require: start state s_0; instructions $e_1, \ldots, e_J$ (*move*, *stack*) and static cubes; trained window model (97 frames, strides $1, 2, 4, 8$)

1: **Root.** Let $R = L/96$ environment steps per frame. Draw 4 candidate roots, each one 97-frame window with all instructions (mode *complete*) and s_0 as its first frame, using the embedding of the trained stride whose windows hold about J releases
2: For each candidate, read the cross-attention of every instruction token over the 97 frames; its peak frame is where the root performs that instruction
3: Keep the candidate whose peaks follow the list order best (Kendall agreement, ties broken by peak height); record each instruction's peak as its time $u_j \in [0, 1]$
4: **for** stride $s = R/2, R/4, \ldots, 1$ **do**
5: Lay overlapping 97-frame windows (half overlap) over the horizon at stride s
6: **Condition of each window:** the instructions whose time u_j falls in the window (mode *complete*, with their time inside the window), or none (mode *empty*); every window also gets the static cubes
7: **Coherence:** stretch the parent plan by two (linear interpolation) and fix every second frame of the level to it, and the first frame to s_0
8: Denoise all windows jointly (DDIM, 50 steps): at every step, predict each window's clean plan, average the predictions on the overlaps with triangular weights, and re-impose the fixed frames
9: The result is the parent of the next level
10: **end for**
11: **Execute** the stride-1 plan open loop in MuJoCo: an inverse-dynamics model maps each pair of consecutive planned states to an action

the start and the final state implied by the list; *broadcast* gives every window the full list; *root only* executes the single coarse plan, interpolated. HLC places each instruction in the window where the root attends to it most.

C PANORAMA PROMPTS AND EVALUATION

The 25 panorama prompts of §4.2; each is rendered as "wide panoramic photo of *scene*, *objects*, photorealistic, 35mm DSLR photo, natural light, sharp focus".

Table 4: The 25 panorama prompts of Table 1.

ID	Scene	Objects
p01	a green alpine meadow in late afternoon, distant snowy peaks	a red vintage car on the far left; a white hot air balloon in the middle; a yellow tent on the far right
p02	a snowy plain under a pale winter sky, distant pine forest	a brown horse on the far left; a red telephone booth on the right
p03	a flat desert at dusk, dry sand, distant mesas, clear sky	a blue motorcycle on the left; a giraffe in the middle; a wooden windmill on the far right
p04	a quiet sandy beach at golden hour, calm sea, clear sky	a black grand piano on the far left; a yellow school bus in the middle; an elephant on the far right
p05	a calm mountain lake at sunrise, pebble shore, pine-covered hills	a red fire truck on the far left; two zebras on the right
p06	a quiet city street at dusk, brick buildings, wet asphalt	a brown cow on the far left; a green tractor in the middle; a giant teddy bear on the far right
p07	a lavender field in summer, rows of purple flowers, blue sky	a black motorcycle on the left; two brown horses on the far right
p08	a white salt flat under a clear blue sky, distant mountains	a red armchair on the far left; a yellow taxi in the middle; a giraffe on the far right
p09	a country road through an autumn forest, orange and yellow leaves, soft light	an elephant on the far left; a red mailbox in the middle; a white van on the far right
p10	a snowy mountain ridge on a clear day, deep blue sky	a red helicopter on the left; a brown camel on the far right
p11	green rice terraces on a hillside, misty morning	a red telephone booth on the far left; a black horse in the middle; a yellow hot air balloon on the far right
p12	a small fishing harbour on a calm morning, stone quay, gentle sea	a red double-decker bus on the far left; a white horse in the middle; a wooden windmill on the far right
p13	an african savanna at sunset, golden grass, acacia trees	a yellow school bus on the far left; a white lighthouse in the middle; a blue pickup truck on the far right
p14	a city park in spring, green lawn, blooming cherry trees	a red fire truck on the left; a brown camel in the middle; a black grand piano on the far right

p15	a quiet canal in an old european town, stone bridges, calm water	a giraffe on the far left; a yellow hot air balloon on the right
p16	an open prairie under a stormy sky, tall grass	a white lighthouse on the far left; a yellow school bus in the middle; a green tractor on the far right
p17	a rocky coastline with crashing waves, overcast sky	a red vintage car on the far left; a brown horse in the middle; an orange tent on the far right
p18	a vineyard on rolling hills at golden hour, rows of grapevines	an elephant on the left; a red helicopter on the far right
p19	arctic tundra with patches of snow, low grey clouds	a yellow taxi on the far left; a black grand piano in the middle; two zebras on the far right
p20	a cobblestone village square on a sunny afternoon, old stone houses	a brown cow on the far left; a white hot air balloon in the middle; a green tractor on the far right
p21	a long wooden pier over a calm sea at dusk, pastel sky	a red motorcycle on the far left; a giraffe on the right
p22	a black sand beach under a grey sky, dark basalt cliffs	a yellow school bus on the far left; a white horse in the middle; a red telephone booth on the far right
p23	a sunflower field in summer, blue sky with small clouds	a blue vintage car on the far left; two elephants on the right
p24	a frozen lake in winter, snow-covered shore, pale blue sky	a red armchair on the left; a brown camel in the middle; a yellow taxi on the far right
p25	a red rock canyon at midday, layered cliffs, blue sky	a white lighthouse on the far left; a giant teddy bear in the middle; two red umbrellas on the far right

Protocol. Every method is run once per prompt with seed 0, fixed before any image was generated, at 5120×704; HLC uses a single root, without any selection.

Detector. We run OWLv2 (Minderer et al., 2023) (`owlv2-base-patch16-ensemble`) on overlapping square tiles (side $H = 704$, stride $H/2$) with all object phrases of the prompt as queries, map the boxes to panorama coordinates and merge them per phrase by greedy suppression (IoU ≥ 0.3, or intersection over the smaller box ≥ 0.6, so that a box cut by a tile edge yields to the whole box from the neighbouring tile). Boxes with score ≥ 0.25 are counted. Each stated position is the centre of a fifth of the width ($x_p = 0.1, 0.3, 0.5, 0.7, 0.9$ for far left, left, middle, right, far right). For an object with stated count c and n detected instances at horizontal centres x_i: the count is correct if $n = c$; the position is correct if $n \geq 1$ and $|x_i - x_p| \leq 0.2$ for all i; the object leaks if some $|x_i - x_p| > 0.2$; the object is correct if count and position are correct, and a prompt succeeds if all its objects are correct.

VLM judge. Qwen2.5-VL-7B-Instruct (Bai et al., 2025) answers two questions per object: *"This is a wide panoramic photo. How many instances of '{object}' are visible in it? Look carefully across the whole width, including small or partly hidden ones. Answer with a single number."* on the whole panorama, which gives n, and *"How many instances of '{object}' are visible in this image (count partly visible ones too)? Answer with a single number."* on each fifth of the width (1024 pixels). The position is correct if the object appears in the stated fifth or an adjacent one and in no fifth two or more away; it leaks if it appears two or more fifths away.

Validation. We labelled 21 panoramas by hand (57 objects, all methods). The detector matches the hand count exactly for 81% of the objects and within one for 98%, and agrees on the per-object verdict for 91%; the VLM judge gives 77%, 100% and 91%. Detector errors are double boxes on one object, look-alike objects and positions at the ± 0.2 border; the VLM judge is lenient on position (an adjacent fifth passes) and over-counts small animals by one. The detector is therefore the stricter of the two, and we report both.

Coherence and quality. Following the evaluation code of Lee et al. (2023), the panorama is resized to height 512 and cut into non-overlapping 512×512 crops. Intra-LPIPS is the mean LPIPS-VGG (Zhang et al., 2018) over crop pairs, Intra-Style-L the Gram-matrix distance of VGG19 conv1–5 features summed over crop pairs, and NIQE (Mittal et al., 2013) is averaged over the crops.

The planner baseline receives the full prompt and returns one prompt per window (instruction and outputs in the supplementary material).

D Additional results

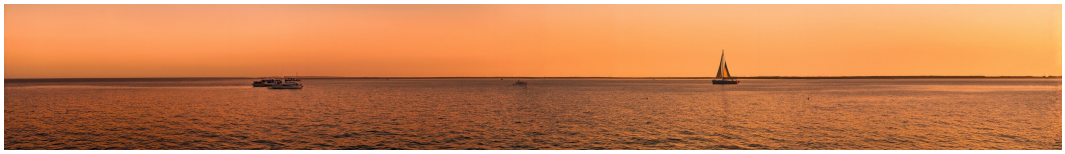
Table 5 shows that the full-prompt cascade draws 4.7, 9.2 and 19.2 instances per object at depths 2–4, while HLC stays close to one. In this study the HLC root was the better of two candidates,



a mountain lake at sunrise, still water, pine forest on the shores, a wooden rowing boat at the left edge of the view, a small red cabin behind the lake, on the far shore



a green savanna plain under a cloudy sky, three zebras



a calm bay at sunset, orange sky reflected in the water, exactly two sailboats

Figure 8: More HLC panoramas (5120×704, SD3-medium) for prompts outside the position template. Each prompt also ends with *photorealistic, 35mm DSLR photo, natural light, sharp focus*.



(a) 1:64 (24576×384)



(b) 1:128 (49152×384)

Figure 9: Where the horizon ends: HLC on very long panoramas (SD3-medium, height 384; each panorama is cut into eight strips, shown top to bottom). Prompt: *a green alpine valley in late afternoon, meadows, distant mountains; a red wooden barn on the far left, a single old oak tree in the middle, a small stone bridge on the far right*. The scene itself stays consistent at both lengths, but from 1:64 window boundaries and small artifacts start to show (seams and bands in the sky), and at 1:128 they become clearly visible.

scored by how strongly each object's phrase is attended in its stated fifth of the canvas; with a single root (as in Table 1) depth 2 gives success 0.30 / 0.60, objects 0.65 / 0.85 and mean count 1.23 on the same ten prompts. At 1:29 the failures of HLC change from extra instances to missing objects: a routed window receives the phrase but sometimes leaves it undrawn.

Table 5: Length study on panoramas (SD3-medium, height 704, prompts p01–p10 of Appendix C, one seed). Rates are detector / VLM judge; mean count is the mean number of detected instances per requested object (1 is correct). "Full prompt in every window" uses the same tree and root as HLC.

	1:7.3 (depth 2)	1:14.5 (depth 3)	1:29 (depth 4)
HLC: prompt success	0.30 / 0.80	0.30 / 0.40	0.40 / 0.40
HLC: objects correct	0.69 / 0.92	0.69 / 0.77	0.58 / 0.58
HLC: mean count	1.19	1.08	0.92
Full prompt in every window: mean count	4.7	9.2	19.2
Full prompt in every window: prompt success	0.00	0.00	0.00



Figure 10: More long videos with HLC on LTX-2.5 (481 frames, 1280×704, 24 fps, root seed 1, no selection). Beach: *A static shot of an empty sandy beach with a calm blue sea, a white seagull flies past, a black dog runs along the shore, a person in a red swimsuit jogs by, the camera never moves.* Snow: *A static shot of a snowy field with a single pine tree, a red fox trots across the snow, a brown deer walks past, a skier in a blue jacket glides by, the camera never moves.* On the beach the jogger passes once while the gulls and the dog stay in the distance throughout; in the snow the root draws the fox and the deer together, so they overlap for much of the clip before the skier passes, a case where the root does not stage the events one after another.

D.1 Figure 1: seed selection

Figure 1 first used seed 11 (seed 12 as a second copy), chosen before the images were seen. In both, HLC drew each object once but put the oak off the middle (at 0.33 and 0.64 of the width), because the model's own score routed it to the second or third of the four leaf windows. We then fixed the following rule before generating further seeds: run HLC on seeds 13 to 24 in order and take the first seed whose panorama has, judged by eye on full-resolution crops, exactly one barn with centre in $[0, 0.25]$ of the width, one oak or broadleaf tree with centre in $[0.375, 0.625]$ and one bridge with centre in $[0.75, 1]$; keep seed 11 if none qualifies. The baselines were run only on the chosen seed and played no part in the choice. Seed 13 qualified (barn 0.03, oak 0.385, bridge 0.91); it is the first seed tried. Of the other eleven seeds, judged on a downscaled overview only, one (seed 24) would probably also pass; the others placed a conifer or put the tree outside the middle quarter.

E Alternatives we tried and found inferior

- **Word-level weights.** Routing single words instead of phrases leaks modifiers: the same boat appeared in all eight windows; phrases fixed it (count error 2.33 $\rightarrow$ 0.72).
- **Zeroing tokens instead of re-encoding.** Zeroing a phrase's embeddings leaves the concept in the contextualised tokens; a window with the other phrases zeroed drew none of its own object.
- **Re-routing at every level.** Intermediate levels are blurred, so re-scoring there mislocalised scene phrases; routing once at the root and inheriting the weights is more stable.
- **Soft weights from raw scores or at the encoder input.** Every window keeps a weight of at least $\sim$0.65, so the object is requested everywhere; scaling input embeddings gives faint copies.

- **Video: one event per window.** Forcing each window to exactly one event contradicts the parent's frames where two events overlap and produces hybrid animals; allowing every event above the threshold removes them.
- **Video: releasing the end anchor.** Without the parent's last frame, colours drift and joints jump; low-band anchors reduced drift but added ghosts.

F COMPUTATIONAL COST

On one A100 at 5120×704: HLC takes about 21 s per panorama (depth 2), CoFi 67 s, and SyncDiffusion 498 s. For 493-frame videos with Wan 2.2, HLC takes about 10 minutes and Gen-L-Video and FreeNoise about 3.5 minutes. The wall-clock ratio (about 2.9) is larger than the ratio of denoiser evaluations (1.4, §4.5) because HLC samples the windows of a level one after another (each window starts from its left neighbour's last frame) and decodes and re-encodes every window in pixel space, whereas the two baselines denoise all windows in one batch and decode once. In planning, HLC uses 3000 and 6150 window evaluations for horizons of 1536 and 3072 steps (about 22 s and 48 s on four CPU threads), the root alone 200.