IT'S THE MODEL, NOT THE MACRO-ACTIONS: WHAT HIERARCHY BUYS IN LATENT-SPACE PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

A hierarchical latent world model raises two abstraction levels at once: it rolls plans through a model that predicts ten steps at a time, and it searches over learned temporally extended macro-actions. The pair beats flat planners on long-horizon tasks, and the standard explanation credits the *action abstraction*. Using a released model pair whose flat and hierarchical checkpoints have *bitwise-identical* level-1 weights — so arms differ only in the planner and in which of the two predictors rolls the plan — we find that, on this pair, the credit belongs elsewhere. We first characterize what the learned macro-action space contains: a $\sim$3.5-dimensional code whose variance the two lowest temporal modes of a ten-step action chunk reach to 98% and plain constant-hold chunking to 65.5%. We then run the same hand-designed action classes through each of the two models. Over the fine model, *every* restriction scores below the unrestricted planner in the same harness — $5/20$, $2/20$, $0/20$, $2/20$ across classes spanning 66% to 99% coverage, against $9/20$ (three of the four gaps significant). Over the coarse model, the *crudest* class — two numbers per macro-action — reaches $\mathbf{19/20}$, matching the full released hierarchy, with 14 discordant trials to 0 against the identical class over the fine model ($p = 1.2 \times 10^{-4}$); searching the entire learned 8-D space does no better ($17/20$). Model-side measurements taken with no planner in the loop suggest why: at long lookahead the fine model's cost stops ordering plans the way the environment does (latent cost, $\rho = 0.20 \rightarrow 0.05$ from 100 to 250 steps; position cost, -0.07 at 500), while the coarse model's does not, and the gap is significant at 500 steps ($+0.15$, $p = 0.0012$, 17/20 layouts, $K = 256$ plans). At 250 steps an earlier $K = 64$ estimate of that gap ($+0.14$, $p = 0.007$) did not survive a $4\times$ larger plan sample ($+0.05$, $p = 0.26$); we report it as within noise. We also report what failed: truncating the flat planner's lookahead to where its model is still trustworthy does not help, monotonically, across a $10\times$ range ($1/20$, $5/20$, $10/20$, $12/20$), because the task's own objective is uninformative at short horizons ($\kappa = 0.007$ at 50 steps). Of this paper's preregistered predictions three were confirmed, three refuted and two null: the confirmations are two model-side measurements and the prediction that constant hold over the coarse model reaches $\geq 17/20$; every refutation is a prediction about the flat planner at its own timescale.

1 INTRODUCTION

A world model can be trained to predict one environment step at a time or many; that choice — its *prediction stride* — fixes the temporal abstraction level at which a planner rolls candidate plans forward. At which level should a world model predict, and can the level be chosen from the dynamics rather than learned? Hierarchical latent world models answer the first question by hand, twice: a fine model at stride one and a coarse model at stride ten, the latter paired with a learned space of temporally extended macro-actions. The resulting hierarchical planner solves long-horizon navigation problems that a flat planner over the same latent space does not, and the natural reading, the one the literature encourages, is that the *action abstraction* does the work: macro-actions compress the horizon, shrink the search space, and give the optimizer a better-conditioned problem.

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

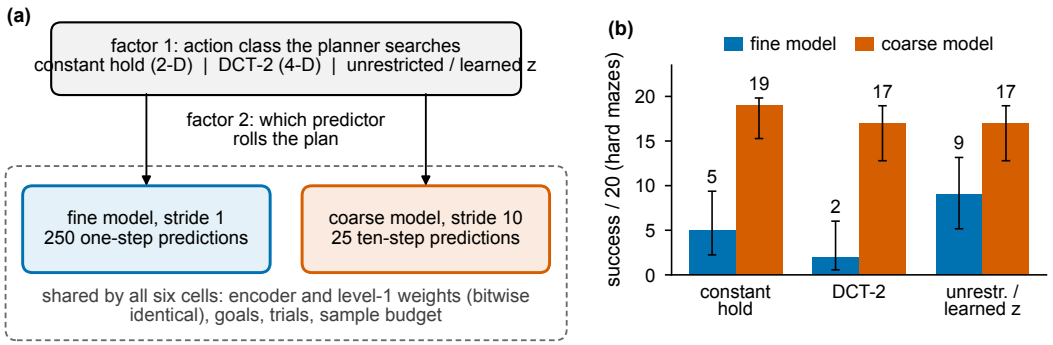


Figure 1: **The design and the headline result.** (a) A hierarchical planner changes two things at once; we cross them on a released pair whose flat and hierarchical checkpoints share bitwise-identical encoder and level-1 weights (schematic). (b) Success on the same 20 hard trials for each action class through each model, all six cells from one planning harness (Table 2); error bars: 95% Wilson intervals ($n = 20$). On this pair, the model decides; the action class does not.

That reading conflates two abstraction levels. A hierarchical planner differs from a flat one in *what it searches over* — a learned macro-action space — and in *what it rolls the plan through* — a model that predicts ten steps at a time. Because the two always change together, the field's evidence cannot say which level matters; this paper asks that question in its controlled form.

We separate the two levels on a released model pair whose flat and hierarchical checkpoints have bitwise-identical level-1 weights, so that no comparison in this paper involves retrained weights: arms differ only in the planner and in which of the two predictors rolls the plan. Three prior accounts have already fallen on this pair (Section 3): coarse-scale transition structure is a measurement artifact; the gap survives giving the flat planner the hierarchy's entire sample budget; and replacing learned macro-actions with plain action chunking collapses *below* the flat baseline and degrades further with search. What survived is the two-way confound above.

Our contribution is to split it, and the answer is one-sided (Figure 1). We first measure what the learned macro-action space actually contains, then hand the same hand-designed action classes to each of the two models. Over the fine model every restriction does worse than the unrestricted planner, regardless of how much of the learned space it spans. Over the coarse model the crudest class available — two numbers per macro-action — matches the full released hierarchy. The learned macro-action space, characterized in Section 4, turns out to be something the planner does not need the details of; what it needs is a model that still orders plans correctly at long range, which we measure directly and model-side in Section 5. The ingredient, on this evidence, is the abstraction level of the *model* — the stride at which it predicts — and not the abstraction level of the action space searched over it.

We also report a failure, because it is the part most likely to be useful to others. The obvious engineering consequence of our measurements — plan no further ahead than your model can be trusted — does not work here, monotonically, across a $10\times$ range of lookahead, because at short horizons the task's own objective carries almost no information about eventual progress. A planner in this environment must choose between an objective it can evaluate and an objective worth evaluating, and the hierarchy is the only configuration we tested that escapes the choice.

Contributions.

1. **On this pair, the model's abstraction level, not the action class, decides success.** A 2×2 that separates action class from rollout model on a released pair with identical level-1 weights (Section 7): the same constant-hold class is worth 4–$5/20$ through the stride-one (fine) model and $19/20$ through the stride-ten (coarse) one (discordant 14–0 in one harness, $p = 1.2 \times 10^{-4}$), and the full learned 8-D macro-action space is no better than that 2-D slice.

2. **The action-space abstraction is not the ingredient.** A characterization of the learned macro-action space (Section 4) and a coverage ladder over it (Section 6.1) showing that, through the fine model, success does not track coverage in any direction: no hand-designed action class does what changing the model's stride does.
3. **The model's abstraction level is visible without a planner.** Rank-fidelity measurements with no planner in the loop (Section 5): at 500 steps the stride-ten model orders plans the way the environment does and the stride-one model does not ($17/20$ layouts, $p = 0.0012$ at $K = 256$). We also report that an earlier $K = 64$ estimate of a significant gap at 250 steps did not replicate at $K = 256$ (Appendix A).
4. **The fine model's failure is not repaired at its own abstraction level.** Two negative results: truncating the flat planner's lookahead does not help at any of four horizons, and neither does enriching its action class; a simulator-only measurement (κ) says why the first cannot work here.

All predictions were preregistered and their failures are recorded alongside their successes (three confirmed, three refuted, two null). The pattern is worth stating up front: *the confirmations are two model-side measurements and one planner arm over the coarse model; every refuted prediction is about the flat planner at its own timescale.*

2 THE INSTRUMENT

Model pair. Released PLDM (Sobal et al., 2025) (flat) and HWM (hierarchical) checkpoints (Zhang et al., 2026) for diverse-maze navigation. Two facts, both established by inspection and both load-bearing. First, the hierarchical checkpoint has *no level-2 encoder*: its level-2 tensors are all predictor tensors, its configured level-2 encoder is the identity, and both levels share one $18 \times 43 \times 43$ latent space. "Information content per scale" is therefore a tautology on this model, which is why the structure account had to be tested as a transition-graph question rather than a representation question. Second, the level-1 weights are *bitwise identical* across the two checkpoints (verified tensor by tensor after normalising parameter names; max absolute difference 0.0 over all 28). Every arm below therefore shares one encoder and one level-1 predictor; arms differ in the planner and in whether the plan is rolled through the level-1 or the level-2 predictor.

Evaluation. The release's own planning stack; goal-reaching success on held-out start/target pairs binned by geodesic distance; identical trials across arms, so all comparisons are **paired exact McNemar** tests. The 20 maze layouts used for evaluation are **disjoint from the 25 layouts the world model was trained on** (checked: zero overlap), so every number here is on unseen mazes. Figure 2 shows the gap on three trials chosen by rule, not by look.

3 THREE ACCOUNTS THAT FAIL

Structure. The 10-step abstract-action transition graph does not carry more structural information (Li and Pan, 2016) than the 1-step graph. The apparent effect ($+0.494$, $20/20$ maps, $p < 10^{-4}$) tracks off-diagonal mass at $r = +0.987$ and disappears under a degree-matched null with self-loops removed ($+0.021$, $p = 0.45$). The ground-truth gate — position decodable at $R^2 = 1.000$ on every map — passed *first*, yet the result was an artifact: gates validate axes, not comparisons.

Compute. On hard mazes the hierarchical planner reaches 0.925 against the flat planner's 0.500 ($n = 40$, paired $p = 0.0002$). Granting the flat planner the hierarchy's entire sample budget leaves $19/20$ against $13/20$ ($p = 0.031$, discordant 6–0), and a $20\times$ sample increase buys it nothing ($11/20 \to 13/20$, $p = 0.73$). The hierarchical planner, conversely, barely needs its budget: at 4000, 2000, 1000, 500 and 250 level-2 samples it scores 20, 17, 17, 17 and $16/20$ (none differs from 4000, $p \geq 0.125$), with steps-to-goal rising only from 140 to 202.

Horizon compression. Giving the flat planner macro-actions with no learned model — each of $250/k$ decision variables held for k steps inside a faithful 250-step level-1 rollout — does not merely fail to imitate the hierarchy. It collapses below the flat baseline at every k ($1/20$, $3/20$, $4/20$ for $k = 2, 5, 10$ against flat's $11/20$) and gets *worse* with more compute ($0/20$ at $10\times$ samples, discordant 0–11 against flat, $p = 0.0010$). Compressing the horizon with the wrong action class is worse than not compressing — and its degradation with compute foreshadows Section 5.

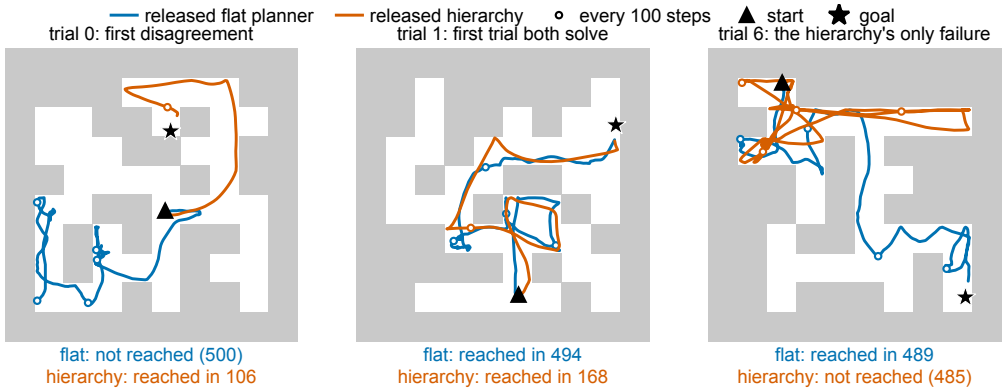


Figure 2: **Case study: the released flat planner and the released hierarchy on the same held-out mazes.** Executed paths on three of the 20 shared hard trials, chosen by rule: the first trial (in index order) on which the two planners disagree, the first both solve, and the only one the hierarchy fails. Circles mark every 100 steps; parenthesised counts are episode lengths at the step cap. The flat planner oscillates in dead ends or reaches the goal near the cap; the hierarchy's typical path is direct. Paths are from the released runs whose first 20 trials are the $11/20$ and $19/20$ arms of Table 5.

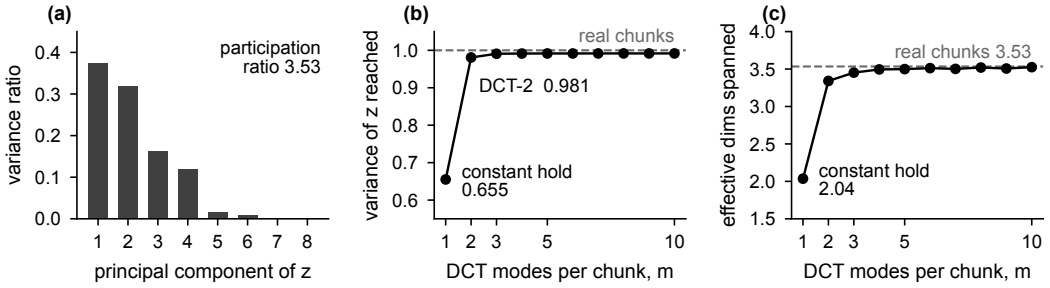


Figure 3: **The learned macro-action space is a $\sim$3.5-dimensional, two-mode object.** (a) PCA spectrum of the 8-D code z over 10,000 real chunks. (b) Share of the variance of z reached, and (c) effective dimensions spanned, when chunks are restricted to their first m DCT modes (dashed: real chunks). Constant-hold chunking (m=1), the class that collapsed in Section 3, reaches two of the three-and-a-half dimensions; two modes reach 98%. This turns "try richer macro-actions" into a one-parameter ladder with a known coverage axis (numbers in Table 4, Appendix D).

4 What the learned macro-action space is

The level-2 macro-action is $z = \mathrm{LayerNorm}(\mathrm{MLP}(a_{1:10}))$, a learned encoding of a ten-step chunk of level-1 actions; at plan time MPPI samples z directly inside the percentile box of that encoder's outputs on real data. Encoding 10,000 non-overlapping real chunks:

- the eight-dimensional code is effectively $\sim$3.5-dimensional (PCA variance ratios .375, .318, .163, .119, .016, .008, .000, .000; participation ratio 3.53);
- it is a *low-frequency* code: DCT modes 0–3 of the chunk are linearly recoverable from z ($R^2 \approx 0.43$ each), modes 4–9 are annihilated ($R^2 \approx 0$), and modes 2–3, though encoded, add only 1% of z's variance beyond the first two (Table 4);
- it is not merely net displacement ($R^2 = 0.65$ from the chunk mean alone).

Figure 3 asks which hand-designed chunk classes can reach it. We stress in advance that Section 7 will show this characterization, though accurate, does not govern planning; we report it because it made the experiments of Sections 6.1 and 7 possible to design.

lookahead T (level-1 steps)	10	20	50	100	250	500
main run: latent cost, plan length $S = 250$, $K = 64$ plans per layout						
level-1 model	0.216	0.187	0.200	0.196	**0.046**	—
level-2 model	−0.020	0.137	0.111	0.137	**0.157**	—
extended run: position cost, $S = 500$, $K = 64$ (superseded by the $K = 256$ rows below)						
level-1 model	—	—	0.151	0.150	0.048	**−0.102**
level-2 model	—	—	0.088	**0.228**	0.189	0.091
same plans, $K = 256$ (nested draw; the estimate used for all claims)						
level-1 model	—	—	0.148	0.176	0.072	**−0.065**
level-2 model	—	—	0.100	**0.234**	0.126	0.089
coarse − fine	—	—	−0.048	+0.058	+0.054	**+0.154**
exact Wilcoxon p	—	—	0.73	0.96	0.26	**0.0012**
layouts with coarse > fine	—	—	8/20	8/20	11/20	**17/20**

Table 1: Rank fidelity: does the model order plans the way the environment does? Spearman ρ between model and true cost over K plans per layout, 20 held-out layouts. The fine model's ranking signal collapses at the horizon the released flat planner plans over and is negative at 500 steps; the coarse model decays more slowly but also runs out (the released hierarchical planner looks 470 steps ahead). **Correction at $K = 256$:** the $K = 64$ gap at $T = 250$ ($+0.140$, $p = 0.0073$, 15/20 layouts) shrinks to $+0.054$ ($p = 0.26$, 11/20) and is not claimed; the gap at $T = 500$ survives ($+0.154$, 95% CI $[+0.08, +0.23]$, $p = 0.0012$, Holm-adjusted over the four lookaheads 0.0048, 17/20). Bold: entries the text discusses; full table and test in Appendix A.

5 How far ahead is each model worth anything?

All of Section 4 is about the action space. The measurements here are about the model, and they are taken *without a planner*: perturb a plan, execute it in the simulator, and compare the model's rollout against the encoder's embedding of the real observation, in the latent space of the planner's cost.

A null we recorded first. At the planner's own sampling amplitude over its own 250-step horizon, the model's response to a plan perturbation is uncorrelated with the environment's at *every* temporal mode ($\cos \in [-0.017, +0.006]$, 1200 perturbations, 20 maps), with unit gain. We had preregistered a fidelity spectrum falling with frequency; there is no frequency structure to find. That prediction was withdrawn in a dated amendment before any behavioural arm reported.

The axis that does carry structure is the lookahead. Rolling both models over the same plans from the same states (Table 7, Appendix E), the two levels agree to within 0.03 out to 50 steps ($\cos$ to the truth 0.675 vs. 0.679 at 50), and then only the fine model breaks (0.340 vs. 0.613 at 250 steps, paired $t = +71$). The fine model's tracking of a plan *change* — the quantity a cost ranking depends on — falls from 0.363 at 10 steps to 0.009 at the 250 steps the released flat planner plans over (largest of three perturbation amplitudes; all three end at ≤ 0.014).

What a planner actually consumes is an order. This cosine asks whether the model gets the *direction* of a plan change right in the full latent space, which is harder than what MPPI needs: only the *order* its cost induces over the plans it drew. Measuring that order directly — Spearman ρ between the released objective (squared distance from the rollout's terminal latent to the encoded goal) and the true cost, over K plans per layout, paired over 20 layouts — gives Table 1.

An organizing frame. Appendix B writes one MPPI replan as a score-function step whose signal-to-noise behaves like $\alpha(T)\,\varphi(T)/\sqrt{d(T)}$ — the fraction of the task's cost decided within T steps, times the model's fidelity at that lookahead, over the square root of the searched dimension. It predicts an interior optimum in the lookahead, hierarchy acting through $\varphi(T)$ at large T, and compute hurting outside the trust horizon (the chunking signature of Section 3). Its lookahead form survives because $\varphi(T)$ was measured to vary, after a frequency form was preregistered, found false and withdrawn. It is a frame, not a predictive model: it fails twice below.

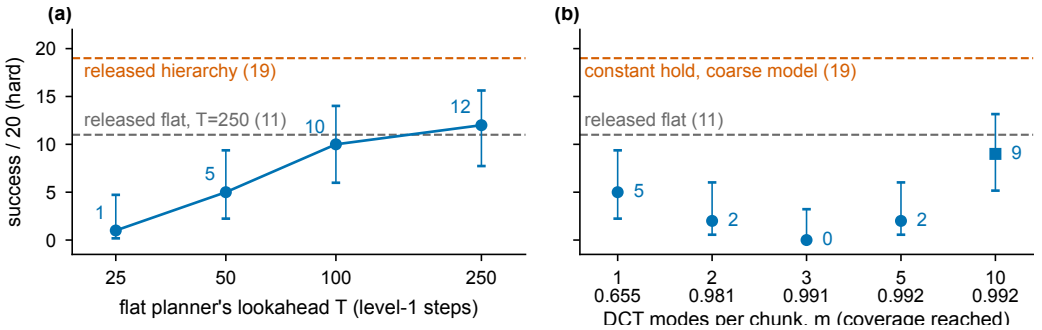


Figure 4: **Two fine-timescale interventions, neither of which helps.** Success of the flat planner on the same 20 hard trials; error bars are 95% Wilson intervals ($n = 20$); dashed lines are the released flat planner (grey) and the coarse-model reference (vermillion). (a) Truncating the lookahead (Table 5): success is monotone non-decreasing in T. (b) Restricting each ten-step chunk to its first m DCT modes (Table 6; square: m=10, unrestricted): coverage of the learned macro-action space saturates at m=2 while success does not track it.

6 The intervention this implies, and why it fails

If the flat planner's trouble is that it optimizes past its model's trust horizon, then truncating its lookahead should help — a one-parameter change, no new action space, no hierarchy, no retraining. We preregistered that prediction, with $T^\star = 100$ computed from the model-side measurements alone, and it is **wrong**. Success is monotone non-decreasing in lookahead across a $10\times$ range (Figure 4a; $1/20$, $5/20$, $10/20$, $12/20$ at $T = 25$, 50, 100, 250; 25 vs. 250: discordant 0–11, $p = 0.001$). The 250-step arm is this runner's reproduction of the released baseline ($12/20$ vs. $11/20$, $p = 1.00$; steps 482 vs. 478), so the negative result is not a harness artifact (Table 5).

What was missing. Our reach term was the *unsigned* fraction of start-to-goal distance a T-step plan closes. In a maze that is the wrong quantity: closing Euclidean distance in 50 steps often means entering a dead end. The signed version — does progress at T predict progress at the end? — is measurable from the simulator with no model in it:

$$\kappa(T) = \mathrm{Spearman}\big(\text{true cost at } T,\ \text{true cost at } 500\big)$$
$$= 0.040,\ 0.007,\ 0.089,\ 0.249,\ 1.000 \quad (T = 25, 50, 100, 250, 500).$$

A 50-step objective in this environment carries essentially no information about eventual progress. This term was added after seeing the T=50 result and is reported as post-hoc.

The scissors. Reach rises with lookahead; fine-model fidelity falls (Figure 5). Their product $\kappa\varphi/\sqrt{d}$ is small at *every* horizon for the flat planner — its best, 0.0011 at T=100, is 0.40 of what the coarse model reaches at 500 (at $K = 256$; 0.00094 and one third at $K = 64$). At horizons the fine model can evaluate, the objective is meaningless; at horizons where the objective is meaningful, the fine model cannot evaluate it, and at 500 steps its cost is *anti*-correlated with the truth. The hierarchy is the only configuration we measure that has both at once, which is why neither a horizon knob (Table 5) nor action chunking (Section 3) imitates it.

We are explicit about how much the product formula earns: it predicted the T=50 collapse (its product is $11\times$ smaller than T=100's at $K = 256$; $9\times$ at $K = 64$) and it ranks T=100 above T=250, which the data contradict (10 vs. 12, $p = 0.75$). It identifies when a horizon is *unusable*; it does not resolve differences among usable ones.

6.1 Nor does a richer action class

Section 4 showed the learned macro-action space is a two-mode, $\sim$3.5-dimensional object of which constant-hold chunking reaches only 65.5%. If the *action class* is what the hierarchy buys, then giving the flat planner a class that reaches 98% of it should recover the gap. It does the opposite. DCT-2 reaches 98% of the learned class and does no better than chunking's 65% ($2/20$ vs. $4/20$, $p = 0.63$); DCT-3 and DCT-5 are no better (Figure 4b, Table 6). The three DCT classes are significantly

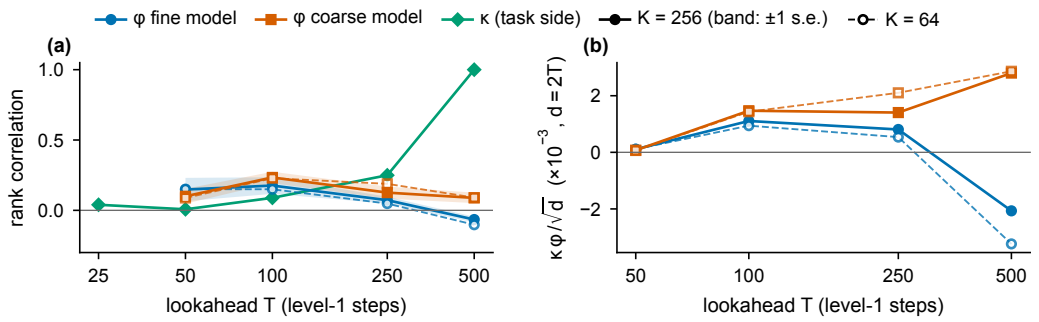


Figure 5: (a) The task-side blade $\kappa(T)$ (simulator only, no model) and the model-side blade $\varphi(T)$ (rank fidelity, position cost; no planner) have opposite slopes; filled markers: $K = 256$ (bands ± 1 s.e. over the 20 layouts), hollow: the earlier $K = 64$. (b) Their product $\kappa\varphi/\sqrt{d}$, $d = 2T$: small everywhere and negative at 500 for the fine model, largest at 500 for the coarse one. At $K = 256$ the coarse product's 250-step value is within noise of its 100-step value, consistent with Table 1.

		success on 20 hard trials ↑, by rollout model	
action class (coverage of learned z)	dim.	fine (250×1 step)	coarse (25×10 steps)
constant hold (0.655)	2	5/20 (published: 4/20)	**19/20**
DCT-2 (0.981)	4	2/20	17/20
unrestricted / free learned z	20 — 8	9/20 (released: 11/20)	17/20

Table 2: The columns decide; the rows do not. All six cells were run in one patched planning harness on the same 20 trials; dimensions are per ten-step macro-action (fine — coarse). Parenthesised: the published runs of the same two fine-model planners, which the harness replicates ($p = 1.00$ and 0.73). Bold: best cell. Constant hold scores $5/20$ through the fine model and $19/20$ through the coarse one (discordant 14–0; 15–0 against the published arm; both $p \leq 1.2 \times 10^{-4}$, and $p = 0.0018$ after Bonferroni correction over all 15 pairwise comparisons among the six cells), and beats the unrestricted fine planner 10–0 (8–0 against the released one, $p = 0.0078$). Steps-to-goal: 147 against the released hierarchy's 143 and the flat planner's 478.

below the unrestricted planner in the same harness ($p \leq 0.04$, paired McNemar); constant hold is below it but not significantly ($5/20$ vs. $9/20$, $p = 0.34$). Restricting the plan to a smooth low-frequency basis hurts *regardless of how well that basis spans the learned macro-action space*.

Together with the horizon sweep this is the third fine-timescale intervention to fail. Compute parity, horizon truncation and action-class enrichment all leave the flat planner where it started, and two of the three make it worse. What none of them changes is the model the plan is rolled through — which is the one thing Section 5 identifies as different.

7 The Decisive Comparison

Every intervention so far changed what the flat planner searches. None changed the model its plan is rolled through. The arms below change exactly that: the planner searches m DCT coefficients of a ten-step action chunk, the chunk is bounded in action space exactly as before, and it is passed through the *released* posterior encoder to give the macro-action the level-2 model is rolled with; as in the released hierarchy, the level-2 plan's subgoals are then tracked by the release's level-1 sub-planner. Same horizon, same trials, same sample budget, same encoder.

Two conclusions follow, and the second retracts something we established earlier in this paper.

The hierarchy's advantage is the temporally-abstract model. The identical action class, at the identical horizon, through the identical encoder, is worth 4–$5/20$ or $19/20$ depending on which model rolls the plan (and, over the coarse model, the released subgoal loop that model drives).

The learned macro-action space is not what the hierarchy buys. Searching the full learned 8-D space is no better than searching a 2-D constant-hold slice of it ($17/20$ vs. $19/20$, $p = 0.63$), and both match the released planner. Section 4's characterization of that space is accurate and, as far as planning is concerned, beside the point: the planner does not need its details (replication of published arms: Appendix E).

Is the coarse planner steering with its macro-actions at all? If the level-2 rollout ignored the macro-action it is fed, its subgoals would not depend on the plan, the level-1 sub-planner would be doing the work, and $19/20$ would say nothing about temporal abstraction. We tested this with a control whose decision rule was committed before it ran: $\geq 17/20$ would mark the result as degenerate and withdraw it; $\leq 12/20$ would mark the macro-actions as load-bearing. The constant-hold arm is re-run with its macro-actions randomly permuted across the sampled population after synthesis, so the sample budget, the action bounds and the marginal distribution over macro-actions are unchanged and only the correspondence between the plan being scored and the macro-action rolled out is broken. It scores $2/20$ (discordant 0–17 against the intact arm, $p = 1.5 \times 10^{-5}$; below the flat planner's $11/20$, 1–10, $p = 0.012$; steps-to-goal 456.5). The macro-actions are load-bearing. A companion arm meant to scramble the free-z planner never engaged — the permutation sat in a code path only the chunk-parameterized arms take — and returned per-trial outcomes bit-identical to the intact free-z arm; we have no scramble control for free z and draw no conclusion from it.

At $n = 40$. Re-run on all 40 hard trials and paired with the stored 40-trial chunking arm (the same class over the fine model), the constant-hold arm over the coarse model scores $37/40$ against $7/40$: discordant 30–0, $p = 1.9 \times 10^{-9}$. The released hierarchy also scores $37/40$ there (discordant 1–1).

8 Related work

Temporal abstraction and hierarchy. The options framework (Sutton et al., 1999) and its successors motivate macro-actions primarily through exploration and credit assignment during *learning*. Nachum et al. (2019) isolate the claimed benefits of hierarchical RL across locomotion, navigation and manipulation and conclude that most of the measured benefit is attributable to improved exploration rather than to easier policy learning or to the imposed hierarchical structure. Our setting is complementary and unusually clean: the two planners here share bitwise-identical level-1 weights and are compared at *deployment*, so no learning-time mechanism can be responsible for the gap we measure. Jumpy (Buesing et al., 2018) and clockwork (Saxena et al., 2021) world models predict at a coarser, hand-fixed stride, as the pair we study does. Hierarchical Diffuser (Chen et al., 2024) plans jumpy subgoals at a coarse stride and credits the larger receptive field at lower cost; our 2×2 isolates the rollout stride in a latent dynamics model at a fixed action class.

Action chunking. Chunked action prediction is now standard in imitation learning: ACT (Zhao et al., 2023) predicts an action sequence rather than a single action, explicitly to reduce the effective horizon of the task, and ensembles across overlapping chunks. Our chunking result concerns something different — chunking as a plan parameterization *inside a sampling planner over a learned model* — and points the other way: at matched horizon and compute it collapses below the flat baseline and degrades further with search (Section 3).

The model we study. HWM (Zhang et al., 2026) learns world models at multiple temporal scales in a shared latent space, uses the long-horizon model's predictions as subgoals for the short-horizon one via latent matching, and compresses primitive action chunks into latent macro-actions with a learned action encoder. Our paper takes its released diverse-maze checkpoints as an instrument rather than as a baseline to beat: the flat and hierarchical releases share bitwise-identical level-1 weights, which is what makes a clean planner-only comparison possible at all.

Concurrent work reaching part of this conclusion. Caselli et al. (2026) freeze a pretrained low-level latent world model, add high-level planning over latent subgoals, and report that hierarchy does not automatically help: at longer horizons they find a *mismatch between the learned high-level action space and the inference-time search distribution*, and that unconstrained search selects latent macro-actions that score well under the model but make poor control targets. Constraining search to macro-actions encoded from training trajectories recovers $+11.3$ and $+14.7$ points on PushT. This corroborates our measurements from a different model and task, and the mechanism is the same one: a cost that ranks plans well under the model and badly under the environment (Section 5).

Our arms sit on both sides of their intervention: our free-latent arm samples the latent macro-action freely (their unconstrained case, $17/20$); our constant-hold arm reaches it only through the released encoder, on-manifold by construction (their constrained case, $19/20$). The direction agrees, though at $n = 20$ we report the two as indistinguishable, as our preregistration anticipated. Where we differ is what the finding is *about*. Caselli et al. (2026) locate the bottleneck in high-level subgoal generation — the learned action space. Our 2×2 (Section 7) holds the action class fixed and swaps only the model that rolls the plan, and finds the entire gap there: the same crude class is worth $4/20$ or $19/20$ depending on the model alone, and the learned macro-action space can be replaced by two numbers at no cost. On our reading their constrained-search result works *because* it keeps the coarse model's inputs in distribution, not because the learned action vocabulary carries representational work the planner needs. Appendix C covers rollout length, adaptive chunking and test-time search.

9 Limitations

One environment family (diverse mazes) and one released model pair. The evaluation layouts are disjoint from the 25 training layouts, so the results are on mazes the model has never seen, but they are drawn from the same distribution; a second environment family remains the main open item, and the release ships only the PLDM implementation.

$n = 20$ paired trials on every behavioural cell. This is enough for the decisive contrast, which is 15–0 discordant (30–0 at $n = 40$), and not enough for the comparisons we explicitly decline to make from it — notably $19/20$ vs. $17/20$ between action classes over the coarse model, which we report as indistinguishable rather than as an ordering.

The rank-fidelity table is one measurement seed at $K = 256$ plans per layout, and one of its entries did not survive the move from $K = 64$: the coarse model's advantage at $T = 250$ (Appendix A). We therefore locate the mechanism at the longest lookahead we measured (500 steps) and do not claim it at the flat planner's own 250. More measurement seeds and a finer grid between 250 and 500 steps are the next measurements.

The coarse column of Table 2 is the released hierarchical control loop: its subgoals are tracked by the level-1 sub-planner. The scramble control shows the coarse plan is load-bearing ($2/20$ when its macro-actions are permuted), but we ran no coarse-model arm without that sub-planner, so "the model" in our claim is the coarse predictor together with the subgoal loop it drives.

MPPI is the sampler throughout; we do not know how much of the fine model's failure is specific to sampling-based optimization, though the falsified compute-parity arms suggest the limitation lies in the objective rather than the search. Fidelity is measured around plans drawn from the planner's own prior — what MPPI samples at its first replan — rather than around converged plans, so it describes the regime the planner starts in, not the one it ends in.

The decomposition in Appendix B motivated our experiments and twice failed to predict them; it is an organizing frame, not a quantity to optimize. A later set of arms capping the hierarchy's own lookahead (Appendix E) is monotone too; we do not count it as a confirmation.

Choosing the stride is left open: the released pair fixes ten by hand (Appendix C).

10 Conclusion

On a released model pair whose flat and hierarchical checkpoints share bitwise-identical level-1 weights, the hierarchical planner's advantage on hard mazes is not the learned macro-action space: the crudest hand-designed macro-action — two numbers held for ten steps — matches the full released hierarchy when the plan is rolled through the coarse model ($19/20$) and collapses when the identical plan is rolled through the fine one (4–$5/20$), while the full learned 8-D space buys nothing over that slice. Model-side measurements with no planner in the loop are consistent with this: at 500 steps the coarse model still orders plans the way the environment does and the fine model does not; at 250 steps the two are not distinguishable once the estimator is given enough plans. What the hierarchy buys, on this evidence, is a model whose cost is still worth optimizing at long range; shorter lookahead and richer action classes both fail; which stride to predict at, and whether it can be read off the dynamics, is what this leaves open.

REPRODUCIBILITY STATEMENT

Every number in this paper can be regenerated from the logged experimental outputs. The per-trial outcomes of every behavioural arm, the rank-fidelity measurements at $K = 64$ and $K = 256$, the preregistrations with their dated amendments (including the refutations), the patches to the released planning code, and the analysis scripts that regenerate every table — including the exact test used for the $K = 256$ correction (Appendix A) — are included as anonymous supplementary material. All references were verified against a primary source (arXiv listing, publisher page, or DOI record) rather than reproduced from memory.

AI USE STATEMENT

We used a generative AI assistant (Claude, Anthropic), under the authors' direction, to give feedback on experimental design, to implement analysis and plotting code, to help interpret results, to draft and edit the text, to create and edit figures and tables, to identify and summarise related literature, and to format references. We did *not* use generative AI tools to generate synthetic data reported as measurements, or for translation. We have reviewed all AI-assisted work: every reported number was checked by the authors against the logged experimental outputs, and every reference was checked against its published record. We take responsibility for the final content of this work, including all text, claims and artifacts produced with the aid of generative AI.

REFERENCES

W. Zhang, B. Terver, A. Zholus, S. Chitnis, H. Sutaria, M. Assran, R. Balestriero, A. Bar, A. Bardes, Y. LeCun, and N. Ballas. Hierarchical Planning with Latent World Models. *arXiv:2604.03208*, 2026.

V. Sobal, W. Zhang, K. Cho, R. Balestriero, T. G. J. Rudner, and Y. LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. *arXiv:2502.14819*, 2025.

C. Chen, F. Deng, K. Kawaguchi, C. Gulcehre, and S. Ahn. Simple hierarchical planning with diffusion. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2401.02644.

N. Caselli, F. Massafra, S. Punzo, S. Lo Sardo, I. Pantelidis, and S. K. Bhethanabhotla. Mind the Gap: Promises and Pitfalls of Hierarchical Planning in LeWorldModel. *arXiv:2607.12547*, 2026.

R. S. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211, 1999.

O. Nachum, H. Tang, X. Lu, S. Gu, H. Lee, and S. Levine. Why does hierarchy (sometimes) work so well in reinforcement learning? *arXiv:1909.10618*, 2019.

T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems (RSS)*, 2023. arXiv:2304.13705.

M. Janner, J. Fu, M. Zhang, and S. Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1906.08253.

G. Williams, A. Aldrich, and E. A. Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40(2):344–357, 2017.

T. Iiyama, M. Suzuki, and Y. Matsuo. SUNTA: Hierarchical Video Prediction with Surprise-based Chunking. *arXiv:2607.02087*, 2026.

L. Buesing, T. Weber, S. Racanière, S. M. A. Eslami, D. Rezende, D. P. Reichert, F. Viola, F. Besse, K. Gregor, D. Hassabis, and D. Wierstra. Learning and querying fast generative models for reinforcement learning. *arXiv:1802.03006*, 2018.

V. Saxena, J. Ba, and D. Hafner. Clockwork variational autoencoders. *arXiv:2102.09532*, 2021.

A. Li and Y. Pan. Structural information and dynamical complexity of networks. *IEEE Transactions on Information Theory*, 62(6):3290–3339, 2016.

A THE $K = 256$ RE-RUN OF THE RANK-FIDELITY MEASUREMENT

The extended-run rows of Table 1 were first measured with $K = 64$ plans per layout. Because a 64-plan Spearman is noisy, and because the two cost estimators (position-space and latent-space) disagreed in sign at $T = 100$ at $K = 64$, the measurement was repeated at $K = 256$ on the same 20 held-out layouts, the same checkpoints, plan length $S = 500$ and the same plan-draw seed (seed 0); the 64-plan draw is the first 64 of the 256-plan draw, so the two estimates are nested and differ only in sample size. One measurement seed is on record.

Test. For each lookahead T the per-layout difference $D_i = \rho_{\text{coarse}}(i) - \rho_{\text{fine}}(i)$, $i = 1, \ldots, 20$, is tested with a two-sided Wilcoxon signed-rank test using the exact null distribution (all 2^{20} sign assignments; the $|D_i|$ are distinct, so there are no ties); this is the same test applied to the $K = 64$ estimate. "Held-out selection" splits the layouts at random in half 500 times, picks the winning model on one half and scores that choice on the other; the number reported is the fraction of splits in which the held-out gain is positive. The table below is regenerated directly from the logged per-layout measurements.

lookahead T	50	100	250	500
position cost ρ_{pos}, $K = 64$ plans per layout (published extended run)				
fine model	0.151	0.150	0.048	-0.102
coarse model	0.088	0.228	0.189	0.091
coarse $-$ fine	-0.063	$+0.078$	$+0.140$	$+0.193$
layouts coarse $>$ fine	6/20	11/20	15/20	16/20
exact Wilcoxon p	0.4749	0.4749	0.0073	0.0006
position cost ρ_{pos}, $K = 256$ plans per layout (re-run, nested draw, seed 0)				
fine model	0.148	0.176	0.072	-0.065
coarse model	0.100	0.234	0.126	0.089
coarse $-$ fine	-0.048	$+0.058$	$+0.054$	$+0.154$
s.e. of the difference	0.082	0.064	0.040	0.035
layouts coarse $>$ fine	8/20	8/20	11/20	17/20
exact Wilcoxon p	0.7285	0.9563	0.2611	0.0012
held-out selection, frac. > 0	0.44	0.68	0.79	1.00
latent cost ρ_{lat}, $K = 256$ (estimator agreement check)				
coarse $-$ fine	-0.082	$+0.004$	$+0.049$	$+0.053$

Table 3: Rank fidelity at $K = 64$ and $K = 256$, position cost, $S = 500$, 20 held-out layouts. At $T = 250$ the $K = 64$ gap ($+0.140$, $p = 0.0073$) shrinks to $+0.054$ ($p = 0.26$, $11/20$ layouts, held-out selection 0.79) and is not claimed. At $T = 500$ the gap is $+0.154$ ($p = 0.0012$, $17/20$, held-out 1.00). At $K = 256$ the latent-cost estimator agrees in sign with the position-cost estimator at every T (last row), so the $T = 100$ sign disagreement seen at $K = 64$ was sampling noise. The main-run rows of Table 1 (latent cost, $S = 250$) were not re-measured and no claim is made about them at $K = 256$.

B WHY THIS SHOULD GOVERN PLANNING: THE ORGANIZING FRAME

Consider one replan of MPPI (Williams et al., 2017). The plan over a lookahead of T steps is a vector $c \in \mathbb{R}^{d(T)}$; the planner draws K samples $c_k = \bar{c} + \varepsilon_k$ with $\varepsilon_k \sim \mathcal{N}(0, \sigma^2 I)$, scores them with the *model* cost $\hat{J}$, and updates $\bar{c}^+ = \bar{c} + \sum_k \omega_k \varepsilon_k$, with $\omega_k \propto \exp(-\hat{J}(c_k)/\lambda)$. What we care about is the *true* cost of the plan it commits to.

Signal. For a locally linear cost and Gaussian samples the update is the familiar score-function estimate, $\mathbb{E}[\sum_k \omega_k \varepsilon_k] \to -(\sigma^2/\lambda)\hat{g}$ where $\hat{g}$ is the *model's* gradient. The realised improvement in the true cost is therefore

$$\mathbb{E}[\Delta J] \;\approx\; \tfrac{\sigma^2}{\lambda}\, g^\top \hat{g} \;=\; \tfrac{\sigma^2}{\lambda}\, \|\nabla J\|\, \|\hat{g}\|\, \alpha(T)\, \varphi(T), \tag{1}$$

with $\alpha(T)$ the fraction of the task's cost that is *decided* within T steps (rising with T: one cannot plan around a wall one never reaches) and $\varphi(T)$ the model's fidelity at that lookahead — the quantity Table 7 measures, and the quantity that separates the two levels.

Curse. That is the $K \to \infty$ limit. At finite K the softmax does not average, it *selects*: writing the model's cost error over the sampled set as η with scale σ_η, the selected plan's true cost is inflated by roughly $\sigma_\eta\sqrt{2\log K}$, and σ_η grows with the number of searched directions along which the model is wrong, i.e. with $d(T)(1-\varphi(T))$. This term gets *worse* with more compute, which is exactly the signature Section 3 reports: chunking at $10\times$ samples goes $4/20 \to 0/20$.

Consequence. Collecting terms, the planner's signal-to-noise per replan behaves like

$$\mathrm{SNR}(T) \;=\; \frac{\alpha(T)\,\varphi(T)}{\sqrt{d(T)}}, \tag{2}$$

rising while lookahead still buys relevance and falling once it buys mostly model error. Three consequences, each testable: (i) there is an interior optimum in the lookahead, so the released 250-step flat planner is mis-specified; (ii) hierarchy raises $\varphi(T)$ at large T rather than abstracting the action space — Table 7 is that claim measured directly; (iii) more compute helps inside the trust horizon and hurts outside it.

We stress what this account does *not* say. The preregistered version of this account used the action-subspace dimension as the design variable, on the hypothesis that fidelity falls with temporal frequency. That hypothesis was preregistered, measured, and found false — there is no frequency structure in φ at all (Section 5) — and withdrawn in a dated amendment before any behavioural arm reported. The lookahead form survives because $\varphi(T)$ was then measured to vary sharply, not because it was assumed to. In Section 6 the frame is used with $\kappa(T)$ in the role of $\alpha(T)$ and the position-cost rank fidelity in the role of $\varphi(T)$, with $d(T) = 2T$.

C Additional related work

Rollout length and compounding error in model-based RL. That model error compounds with rollout length, and that short rollouts are therefore safer, is long-standing practice: Janner et al. (2019) show that short model-generated rollouts branched from real data obtain the benefits of model-based methods without their usual pitfalls. Two things are new here. First, the trust horizon is *measured* rather than tuned: we report the lookahead at which a released model's cost stops ordering plans the way the environment does, from the model alone, with no planner in the loop. Second, the obvious prescription that follows — plan no further than that — **does not work** (Section 6), because the task's own objective is uninformative at those horizons ($\kappa = 0.007$ at 50 steps). The interesting object is the interaction between the two, not either factor alone.

Hierarchical sequence models. A parallel line puts hierarchy in the predictor rather than the planner. Iiyama et al. (2026) argue that fixed-length chunking misaligns with a sequence's intrinsic temporal structure and choose chunk boundaries by prediction error instead, holding video prediction accurate over 250 timesteps where fixed-chunk baselines degrade within 10. That work has no actions and no planner, but it raises the sharpest alternative reading of our chunking results: perhaps fixed 10-step chunks are simply the wrong boundaries. Boundary placement cannot be the explanation here, because the *identical* fixed 10-step chunking fails through the fine model and matches the full hierarchy through the coarse one.

Choosing the stride. If the ingredient is the model's prediction stride rather than the action space searched over it, what this paper leaves open is how to choose it: the released pair fixes ten by hand, and nothing here says ten is right (the coarse model, too, runs out before the hierarchical planner's own lookahead, Table 1). Two criteria are candidates. The first is model-based: train predictors at several strides and keep the one whose cost still orders plans the way the environment does at the lookahead the task needs — the rank-fidelity measurement of Section 5, taken here at two strides as a diagnostic, not a selector. The second is model-free: read the stride off the transition statistics of the dynamics before any model is trained, in the spirit of temporal abstraction as a semi-Markov process (Sutton et al., 1999) and of jumpy and clockwork world models (Buesing et al., 2018; Saxena et al., 2021); the structural information of a transition graph (Li and Pan, 2016) is one candidate quantity, with the caution that Section 3 found a naive version of it to be an estimator artifact on this pair. This paper measures the first criterion and only conjectures the second.

Test-time search. A companion analysis of the same model family treats test-time search as iterative refinement and finds refinement's value governed by prior–task mismatch. The present paper

concerns a third axis — how far ahead the objective is evaluated — and the two meet at the question of what makes a planner's optimization problem easy.

D DETAIL TABLES FOR SECTIONS 4 AND 6

Figures 3 and 4 plot these tables; the tables add steps-to-goal and the paired tests.

chunk class	dof / macro-action	var. of z reached	eff. dims spanned
constant hold (= action chunking)	2	0.655	2.04
DCT-2	4	**0.981**	3.34
DCT-3	6	0.991	3.45
DCT-10 (full chunk space)	20	0.992	3.52
real action chunks	—	1.000	3.53

Table 4: The learned macro-action space is a $\sim$3.5-dimensional, two-mode object. Constant-hold chunking — the class that collapsed in Section 3 — reaches two of its three-and-a-half dimensions; two DCT modes reach 98%. This turns "try richer macro-actions" into a one-parameter ladder with a known coverage axis.

flat planner lookahead	success ↑ (hard, n=20)	steps ↓	paired vs. this runner's 250
25	1/20	336	discordant 0–11, $p = 0.001$
50	5/20	362	discordant 2–9, $p = 0.065$
100	10/20	418	discordant 4–6, $p = 0.75$
250 (this runner)	12/20	482	—
250 (released, published)	11/20	478	discordant 5–6 vs. 12/20, $p = 1.00$
hierarchical (released)	19/20	143	discordant 7–0 vs. 12/20, $p = 0.016$

Table 5: Truncating the horizon does not help. Success is monotone non-decreasing in lookahead across a $10\times$ range. The 250-step row is this runner's reproduction of the released baseline ($12/20$ vs. $11/20$, $p = 1.00$; steps 482 vs. 478), so the negative result is not a harness artifact.

plan parameterization	coverage of learned z	dof	success ↑	steps ↓
constant hold (m=1; = chunking), published	0.655	50	4/20	391
constant hold (m=1), this runner	0.655	50	5/20	404
DCT-2 (m=2)	**0.981**	100	**2/20**	338
DCT-3 (m=3)	0.991	150	0/20	—
DCT-5 (m=5)	0.992	250	2/20	438
unrestricted (m=10; patch gate)	0.992	500	9/20	492
unrestricted (released flat)	—	500	11/20	478
hierarchical (released)	—	—	19/20	143

Table 6: Bold marks the DCT-2 row the text discusses, not a best result. At the fine timescale, restricting the plan to a smooth low-frequency basis hurts *regardless of how well that basis spans the learned macro-action space*. DCT-2 reaches 98% of the learned class and does no better than chunking's 65% ($2/20$ vs. $4/20$, $p = 0.63$); DCT-3 and DCT-5 are no better. The three DCT classes are significantly below the unrestricted planner in the same harness ($p \leq 0.04$, paired McNemar); constant hold is below it but not significantly ($5/20$ vs. $9/20$, $p = 0.34$). The m=10 row is the patch gate: an orthogonal change of coordinates over the full action block, which must and does behave like the stock planner ($9/20$ vs. the published $11/20$, $p = 0.73$).

E SUPPLEMENTARY FIGURES, TABLES AND A REPLICATION NOTE

A note on replication. Three arms of Section 7 re-derive published numbers through an independent path — new machine, rewritten patches, different RNG stream: the flat planner at $12/20$ and $9/20$ against a published $11/20$, and chunking at $5/20$ against a published $4/20$ ($p = 1.00$). We did not tune anything to make these agree.

Capping the hierarchy's lookahead. Arms limiting the hierarchical planner's own lookahead to 10 or 25 macro-steps against the released 47 ran after the main experiments: $10/20$, $16/20$ and $17/20$ (10 vs. 47: discordant 1–8, $p = 0.039$). Longer is again better. The ordering is the one the frame predicts for the coarse model, but we had recorded before these arms ran that a monotone result cannot separate the frame from "longer is better", which every arm here has shown, so we do not count it as a confirmation.

lookahead (level-1 steps)	10	20	50	100	250
level-1 model (fine, 1 step)	0.664	0.652	0.675	0.528	0.340
level-2 model (coarse, 10 steps)	0.644	0.630	0.679	0.643	**0.613**
paired t (l2 − l1)	-5	-5	$+0.5$	$+16$	$+$**71**
tracking a plan *change*, level-1	0.363	0.343	0.262	0.085	**0.009**

Table 7: cos between each model's rollout and the truth, in the planner's own latent space, 3100 records over 20 held-out layouts. Rolling coarsely costs nothing at short range and buys a great deal at long range. The last row is the quantity a cost *ranking* depends on: at the horizon the released flat planner actually plans over, its own model tracks the effect of changing the plan at cos $= 0.009$ (largest of the three perturbation amplitudes of Figure 6).

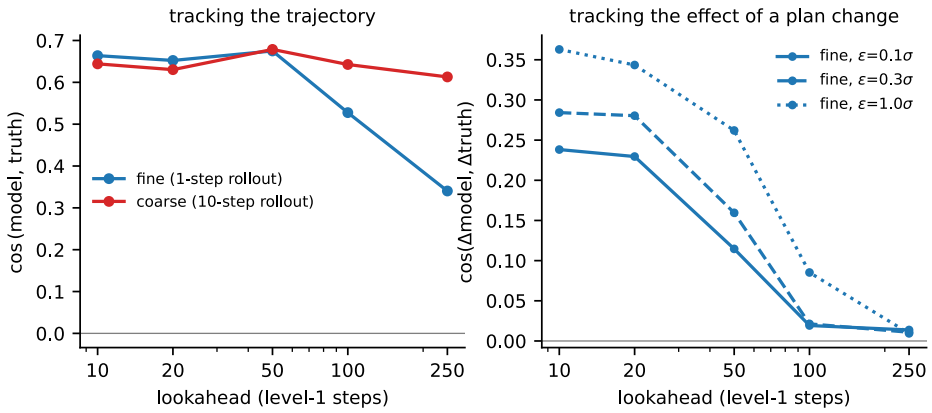


Figure 6: The data of Table 7. Left: cos between each model's rollout and the encoded true observation, fine (1-step) and coarse (10-step) rollouts, as a function of lookahead. Right: the fine model's tracking of the *effect* of a plan change at three perturbation amplitudes; all three fall to ≈0 by 250 steps.