NOT ALL EVALUATIONS ARE EQUAL: WHERE TEST-TIME COMPUTE MATTERS IN WORLD-MODEL PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

World-model planners spend their test-time compute scoring populations of imagined action sequences, and they spend it uniformly across the iterations of the search. We find that the planning decision is made early. In cross-entropy-method (CEM) planning with pretrained latent world models, late-round evaluations can be approximated or removed without a detectable change in success on PushT, Reacher, and Two-Room, whereas approximating early rounds can derail the search. This is consistent with the contraction of the proposal: as it narrows, a linear response operator fitted at run time around the proposal mean halves its held-out error on PushT and cuts it more than fourfold on Reacher. Substituting this operator into the early rounds lowers success significantly on PushT under CEM, iCEM, and MPPI and on Two-Room under MPPI, and raises the final cost on both tasks, whereas score noise of the same cost scale shows no such stage dependence on any of the three tasks. Reacher marks the boundary: its success is flat from 5 to 30 rounds, and early substitution does not hurt and even raises success. Whether early approximation hurts goes together with an offline quantity, the cost of stopping the native search early: the two agreed in all eight settings of a test with predictions fixed in advance. The finding can be used in two ways that trade time against fidelity: stopping two rounds early plans 12% faster at a small but significant increase in final cost, and Exact-to-Local CEM, which keeps early rounds exact and approximates late ones, has the same success outcome as native CEM on all 200 paired PushT decisions with 13% fewer exact queries, 3% less time, and no detectable increase in final cost. For the planners and world models we tested, late-round compute can be cut first; the early rounds are where approximation is risky.

1 INTRODUCTION

World models are increasingly used not only to predict but to plan. A visual control decision becomes a search over imagined futures, and the planner spends its test-time compute scoring candidate action sequences with the model (Ha & Schmidhuber, 2018; Hafner et al., 2018; 2019; Schrittwieser et al., 2020). In population-based planners such as the cross-entropy method (CEM) (Rubinstein, 1997), each decision samples hundreds of candidates, scores them with the world model, and refits the sampling distribution to the best ones, repeating this for many rounds. Recent work makes each evaluation cheaper: LeWM learns a compact predictive model for goal-conditioned planning (Maes et al., 2026), and Fast-LeWM shortens candidate rollouts through action-prefix prediction (Gao & Xu, 2026). The planner itself, however, spends the same full evaluation on every candidate in every round, even though its proposal contracts as the search proceeds and later rounds only compare small perturbations of an increasingly settled plan. We find that the decision is made early: late-round evaluations only refine it and can be approximated or removed, whereas early-round evaluations are where approximation can derail the search.

We probe the roles of the rounds with two instruments. The first is a local response operator, a linear model of the world model's prediction around the current proposal mean, fitted at run time from a few exact probes. Its accuracy at each round tells us how locally predictable the search has become. The second is stage-wise substitution: replacing the exact evaluations of one stage of the

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.

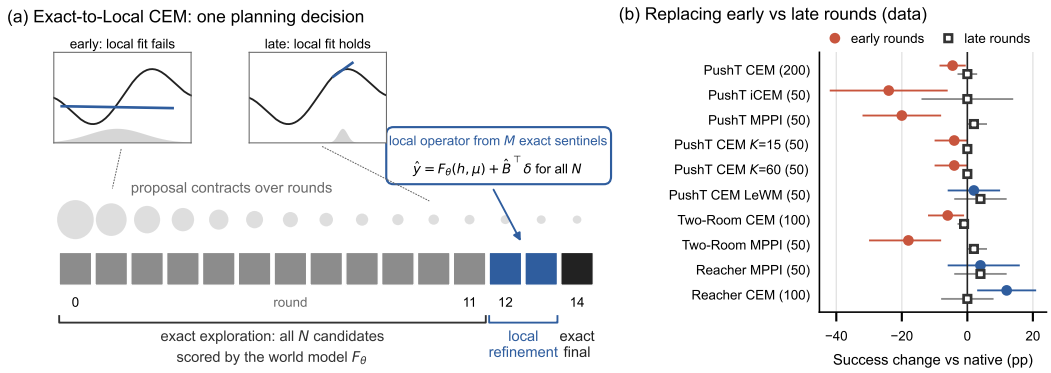


Figure 1: (a) Exact-to-Local CEM for one planning decision (schematic). The CEM proposal contracts over rounds. Early rounds score all candidates with the world model; late refinement rounds score them with a local operator fitted at run time from a few exact sentinels; the final round is exact (the PushT schedule is shown). Insets (synthetic 1-D response): a linear fit around a broad proposal misranks candidates; around a contracted one it holds. (b) Measured change in task success when early or late rounds are replaced by the local operator, paired against the native planner on the same sources (number of sources in parentheses; 95% bootstrap intervals; Table 1). Reacher is the exception.

search with the local operator, or with score noise of matched size, and measuring how far the search and its outcome move. We hypothesize that the stage dependence comes from local predictability: a local description of the world model becomes valid only once the proposal has contracted (Figure 1). This predicts that substituting the local operator should matter more early than late; score noise of the same cost scale, which perturbs every round by the same relative amount, controls for early errors simply propagating further.

The late rounds are dispensable on every task we study: substituting the local operator late does not change success detectably on PushT, Reacher, and Two-Room. The early rounds are not. On PushT, substituting the operator in the first four rounds moves the proposal nearly twice as far from the exact search as substituting it in rounds 8–11 and raises the final planning cost by 11.3 against 0.6–0.9 for later stages; the pattern replicates across search seeds and source sets, on 200 sources early substitution lowers task success significantly, and the same ordering holds when the planner is iCEM or MPPI. On Two-Room, a third task used only for evaluation, early substitution raises the final cost significantly and solves 88 of 100 decisions against 94 ($p = 0.07$), and under MPPI it lowers success from 48 to 39 of 50. Noise of the same cost scale moves the proposal about equally at every stage, although stronger noise also raises the final cost more early than late. Reacher marks the boundary: its success is flat from 5 to 30 rounds, and early substitution does not hurt and even raises success, consistent with a decision made within the first few rounds.

The asymmetry has a practical consequence for test-time compute. We turn it into Exact-to-Local CEM (E2L-CEM), which keeps early rounds exact, replaces later rounds with the run-time local operator, and scores the final population exactly, without changing the objective, elite selection, or update rule. On 200 paired PushT decisions, E2L-CEM has the same success outcome as native CEM on every decision with 13% fewer exact queries and 3% less time, the only time-saving variant with no detectable increase in final cost. The asymmetry also predicts that simply stopping the search early should work, and it does: stopping two rounds early matches native success and plans 12% faster at a small but significant increase in final cost. Where early rounds must stay exact is signaled by the rounds curve: early approximation hurt where stopping the search at about half its rounds raised the final cost, and the two agreed in all eight further settings we tested with predictions fixed beforehand.

Our contributions are:

- **A finding about where the decision is made (Sections 6.1 and 6.2).** The world model becomes locally predictable as the CEM proposal contracts, and replacing exact evaluations

early damages the search far more than replacing them late, on PushT for CEM, iCEM, and MPPI and on Two-Room for CEM and MPPI, while matched score noise shows no stage dependence; the cost of stopping the search early accounts for the Reacher exception and agreed with the substitution outcome in eight of eight prospective tests.

- **A consequence for test-time compute (Section 6.3).** Late-round compute is dispensable on all three tasks, by approximation or truncation, and on two tasks by adaptive stopping; the number of rounds, not the number of queries, sets the planning time.
- **Exact-to-Local CEM (Section 4).** A training-free, drop-in allocation and the only time-saving planner in our comparison with no detectable increase in final cost.

2 Related Work

World models for planning. Learned world models have progressed from compact recurrent latent dynamics (Ha & Schmidhuber, 2018; Hafner et al., 2018; 2019) to discrete, transformer-based, and broad-domain predictors (Hafner et al., 2021; 2023; Micheli et al., 2023; Seo et al., 2023). Recent systems change what the model represents or how an imagined future is produced: DINO-WM predicts pretrained visual features for zero-shot test-time optimization (Zhou et al., 2025), graph and spatial world models add landmarks or geometric structure (Zhang et al., 2021; Kayalibay et al., 2022), and large video models scale action-conditioned generation and prediction (Bruce et al., 2024; Assran et al., 2025). Model-based control couples such predictors to receding-horizon decisions, from ensemble rollouts (Chua et al., 2018; Nagabandi et al., 2018) and visual predictive control (Ebert et al., 2017; Watter et al., 2015) to path-integral trajectory optimization (Williams et al., 2017; 2015), while model-based policy optimization spends model queries during training (Sutton, 1991; Janner et al., 2019; Feinberg et al., 2018; Clavera et al., 2018). Navigation world models plan through generated rollouts (Bar et al., 2025), and a systematic study of JEPA-style world models identifies which planning choices drive physical-planning success (Terver et al., 2025). All of these improve the model, how it is trained, or how the planner is configured. We keep the model and planner fixed and ask which of the planner's queries matter at test time.

Allocating test-time compute. How to spend test-time compute has recently been studied for embodied and world-model systems. DIRECT routes each prompt of a VLM planner to a chain-of-thought depth, model size, or memory length (Dao et al., 2026); adaptive visual test-time scaling decides when and how much to imagine with a world model (Yu et al., 2026); and a compute-value audit of video world models shows that extra samples need not improve selection (Jiang et al., 2026). These works allocate compute across calls, models, or samples. We ask which evaluations inside a single population search shape its decision.

Efficient planning with world models. Planner-side methods cut cost by changing one part of the search. POPLIN plans with policy networks and TAP in a learned latent action space (Wang & Ba, 2020; Jiang et al., 2023); gradient-based visual MPC and CEM-GD change the optimization path (Jyothir et al., 2023; Huang et al., 2021); Sparse Imagination reduces per-rollout visual computation (Chun et al., 2025); LeFlow amortizes planning into a trajectory prior (Huang et al., 2026); hierarchical planners change the temporal abstraction (Zhang et al., 2026; Caselli et al., 2026); and LEAP changes the objective and refinement around a frozen LeWM (Pham & Bera, 2026). Within CEM itself (Rubinstein, 1997; de Boer et al., 2005), iCEM adds temporally correlated noise and population memory (Pinneri et al., 2021), and differentiable CEM makes the optimizer trainable end to end (Amos & Yarats, 2020). Unlike these methods, E2L keeps the candidate distribution, objective, elite selection, and update rule of native CEM, and changes only which rounds receive exact evaluations. It is therefore complementary to them.

Surrogate-assisted and multi-fidelity search. Evolutionary computation has long mixed exact and approximate evaluations. Generation-based evolution control alternates exact and surrogate-scored generations (Jin et al., 2002), and surrogate-assisted CMA-ES variants adapt how many generations use the surrogate or accept its rankings through a rank-correlation test (Loshchilov et al., 2012; Hansen, 2019); surrogate-assisted CEM applies the same idea to the scalar objective (Moss, 2020). More broadly, multi-fidelity optimization treats the choice of information source as part of the search policy (Forrester et al., 2007; Peherstorfer et al., 2018; Kandasamy et al., 2016; Astudillo

& Frazier, 2019; Mikkola et al., 2023; Fan et al., 2024), and value-of-computation analyses ask whether one more evaluation can change a downstream decision (Sezener & Dayan, 2020; Lakshminarasimhan et al., 2024). These methods schedule fidelity by surrogate error or by an explicit value estimate, often using a surrogate built from an accumulated evaluation set. E2L instead allocates fidelity by search stage, keeping the rounds whose decisions later rounds inherit exact, and rebuilds a local response operator around the current proposal at run time. A related reuse pattern appears in training-free diffusion acceleration, where intermediate features are reused across denoising steps (Ma et al., 2024).

3 Problem Formulation

3.1 Population planning with a frozen world model

Let h be the observation history, g the goal, and $a \in \mathbb{R}^q$ a candidate action sequence. A frozen world model $F_\theta(h, a)$ predicts the future representation that the planner scores, and the planning cost is $J_\theta(a; h, g) = \mathcal{J}(F_\theta(h, a), g)$, with lower values preferred.

CEM maintains a diagonal-Gaussian proposal with mean μ_r and scale σ_r at round $r = 0, \ldots, R-1$. The implementation reserves candidate $i = 0$ for the proposal mean and samples the remaining candidates as

$$a_{r,0} = \mu_r, \qquad a_{r,i} = \mu_r + \sigma_r \odot \epsilon_{r,i}, \quad \epsilon_{r,i} \overset{\text{iid}}{\sim} \mathcal{N}(0, I), \quad i = 1, \ldots, N-1.$$

It evaluates each candidate with the world model, $y_{r,i} = F_\theta(h, a_{r,i})$ and $J_{r,i} = \mathcal{J}(y_{r,i}, g)$, keeps the K lowest-cost candidates $\mathcal{E}_r$, and refits

$$\mu_{r+1} = \frac{1}{K} \sum_{i \in \mathcal{E}_r} a_{r,i}, \qquad \sigma_{r+1} = \mathrm{Std}_{K-1}\{a_{r,i} : i \in \mathcal{E}_r\},$$

where the standard deviation is elementwise and uses the sample correction $K - 1$. The planner returns μ_R and executes its leading actions before replanning. A native decision therefore costs RN world-model evaluations, all at full fidelity.

We study the allocation of these evaluations. A planner may replace the exact prediction $y_{r,i}$ with a cheaper approximation $\widehat{y}_{r,i}$ in some rounds, while keeping candidate generation, the cost $\mathcal{J}$, elite selection, and the update above unchanged. The question is which rounds can use $\widehat{y}_{r,i}$ with little effect on the executed action, and how much complete planning time this saves. The simplest alternative is to skip rounds altogether, so any allocation must be compared with running fewer rounds of native CEM at the same cost.

3.2 When is a local approximation valid?

A cheap evaluation affects the decision through two paths. It can change the ranking at the current round, and, if it changes the elite set, it shifts μ_{r+1} and σ_{r+1} and thus every later population. Which of these matters depends on how accurate the approximation is where it is used. We consider approximations that are local around the proposal mean, in the sense that they model $F_\theta(h, a) - F_\theta(h, \mu_r)$ for candidates drawn from the current proposal. Such an approximation is accurate when σ_r is small relative to the scale on which the world model's response bends, and unreliable when the proposal is broad. CEM proposals start broad and contract as the search proceeds, which suggests three roles for evaluations (Figure 1): during global exploration the proposal is broad, a local approximation is unreliable, and its errors are inherited by every later round; during local refinement a local approximation can stand in for the world model; and the final round determines the returned proposal directly. Appendix A formalizes the local approximation error and its propagation through elite selection.

4 Exact-to-Local CEM: Probing and Allocating Evaluations

E2L-CEM divides the R rounds of a decision into three phases that match the roles in Section 3.2: exact exploration, local refinement, and exact final selection. Only the way candidates are evaluated changes between phases; sampling, the cost $\mathcal{J}$, elite selection, and the CEM update are those

of native CEM throughout. We use this construction in two ways. As a probe, the local operator measures how locally predictable the search is at each round, and substituting it into one stage at a time measures how much that stage's exact evaluations matter (Sections 6.1 and 6.2). As an allocation, a fixed schedule of exact and local rounds reduces the number of exact queries per decision (Section 6.3).

Exact exploration. For rounds $r < R_E$, every candidate is evaluated with the frozen world model, $y_{r,i} = F_\theta(h, a_{r,i})$. These rounds are identical to native CEM. They are kept exact because the proposal is still broad, a local approximation would be unreliable, and every later round inherits their elite sets.

Local refinement. At the first local round, indexed by R_E, the proposal has localized around μ_{R_E}. E2L evaluates $M \ll N$ sentinel candidates exactly, including the proposal mean $a^0 = \mu_{R_E}$, and fits a local response operator to them. Writing a candidate in the whitened coordinates of the current proposal, $\delta_i = (a_i - a^0) \oslash \sigma_{R_E}$, the operator predicts

$$\widehat{y}_i \;=\; y^0 + \widehat{B}^{\mathsf{T}}\delta_i, \qquad y^0 = F_\theta(h, a^0),$$

where $\widehat{B}$ is the weighted ridge fit to the exact sentinel responses,

$$\sum_{j=0}^{M-1} \omega_j \big\| y_j - y^0 - \widehat{B}^{\mathsf{T}}\delta_j \big\|_2^2 \;+ \lambda \|\widehat{B}\|_F^2,$$

with weights ω_j that decrease with $\|\delta_j\|$. Here $y_j = F_\theta(h, a_j)$ and $\delta_0 = 0$, so the reference query is part of the M sentinels rather than an additional query. The exact matrix form and ridge scaling are given in Appendix B. For the degree-one operator used in the reported runs, $M < q$ and the implementation evaluates the fit in its dual form. The planner scores all N candidates of this round with $\mathcal{J}(\widehat{y}_i, g)$ and updates the proposal as usual. The same operator is reused, without further world-model queries, for the remaining local rounds $R_E < r < R_E + R_L$.

The operator is fitted at run time from the decision's own queries, so it needs no training data, and it approximates the predicted future rather than the scalar cost: the native cost is applied to the reconstructed future, which keeps about three times the exact elite overlap of a cost surrogate at equal query count (Appendix H).

Exact final selection. The last round $r = R - 1$ evaluates its whole population exactly. Its elite set, and therefore the returned proposal μ_R whose leading actions are executed, is determined by exact world-model scores. Earlier local rounds can still change which population the final round sees; the final exact round guarantees exact selection within that population, not a reproduction of the all-exact search trajectory.

Schedules and cost. A schedule is specified by the numbers (R_E, R_L, R_F, M) of exact-prefix rounds, local rounds, exact-suffix rounds, and sentinel queries, with $R_E + R_L + R_F = R$. The local block includes its fitting round: only that round uses M exact sentinel queries, while the other $R_L - 1$ local rounds use no world-model queries. Thus the search itself uses

$$Q_{\text{search}} = (R - R_L)N + \mathbf{1}\{R_L > 0\}\, M$$

exact queries, compared with RN for native CEM ($R_L = 0$); E2L uses $R_L \geq 1$ and $R_F \geq 1$. The recorded solver additionally evaluates the returned proposal once with the exact model, so the reported count is $Q_{\text{report}} = Q_{\text{search}} + 1$ for every arm.

Both schedules below were fixed on earlier development sources before any source reported in this paper was drawn; neighboring PushT schedules ($R_E = 10$–13) give the same success on fresh sources (Appendix J). For PushT, E12 has $(R_E, R_L, R_F, M) = (12, 2, 1, 24)$, giving $Q_{\text{search}} = 7{,}824$ and $Q_{\text{report}} = 7{,}825$ versus $9{,}000$ and $9{,}001$ for native CEM. E8L4 has $(8, 4, 3, 24)$, giving $Q_{\text{search}} = 6{,}624$ and $Q_{\text{report}} = 6{,}625$. For Reacher, E12 has $(12, 17, 1, 32)$ with $N = 300$, giving $Q_{\text{search}} = 3{,}932$ and $Q_{\text{report}} = 3{,}933$ versus $9{,}000$ and $9{,}001$ for native CEM. Query counts do not include the cost of fitting and applying the operator, synchronization, or sampling; we therefore report complete planning wall-clock per decision, measured over the full solver call, as the primary cost. Appendix B lists the remaining implementation details.

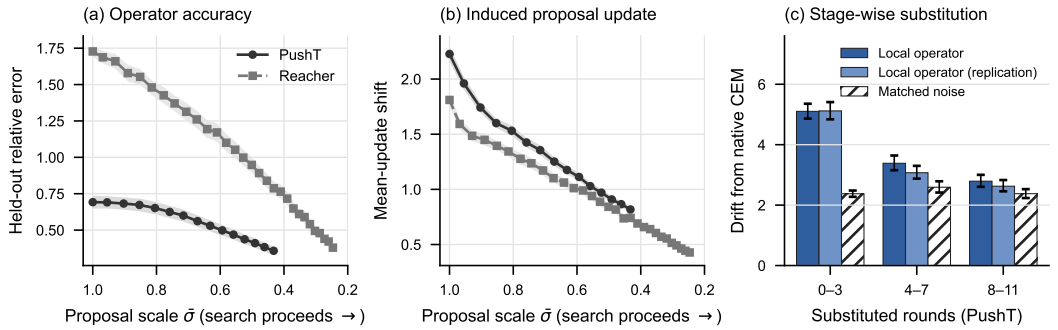


Figure 2: Where a local approximation of the world model is valid. (a) Held-out relative error of the run-time local operator and (b) distance between the updates induced by predicted and exact scores, at every round of native CEM, against the proposal scale (95% intervals over 200 PushT and 100 Reacher decisions). (c) Drift from native CEM after substituting one stage on PushT, for the local operator (two source sets and seeds) and matched score noise (95% bootstrap intervals).

5 Experimental Protocol

Tasks and models. We evaluate on visual PushT, the planar manipulation benchmark popularized by visuomotor diffusion-policy evaluations (Chi et al., 2024), on rendered Reacher from the DeepMind Control Suite (Tassa et al., 2018; Tunyasuvunakool et al., 2020), and on the Two-Room navigation task of the same world-model benchmark suite. Each task uses a frozen pretrained latent world model and the original goal-conditioned controller (Maes et al., 2026; Gao & Xu, 2026). Planning is closed loop: after each decision the leading action block is executed and the planner replans from the new observation. PushT uses the endpoint cost plus a prefix-consistency term with $\beta = 1$; Reacher and Two-Room use the endpoint cost. Two-Room was used only for evaluation: we run the official Fast-LeWM evaluator and planner configuration ($N = 300$, $R = 30$, 30 elites) with our planners in place of its solver, on 50 sources drawn by the evaluator's own generator (seed 42) before any arm was run; the unmodified official solver solves 48/50 of them, in line with the reported success rate.

Sources and pairing. An evaluation source is an (episode, start step) pair from the task's dataset. For every experiment, sources are drawn by an outcome-blind builder that excludes every source identity used in any earlier run and never reads success labels. All arms of a comparison run on the same sources with the same model release, search seed, population size, number of rounds, elite count, and environment initialization, so differences are paired by source. Every schedule is frozen before the sources on which it is evaluated are drawn, so no evaluation source was used to design or select the method it evaluates.

Metrics. We report task success per closed-loop episode (one per source and search seed), and per planning decision the exact final cost of the returned proposal, the number of exact world-model queries per decision, and complete planning time per decision measured over the full solver call (candidate sampling, all world-model queries, fitting and applying the local operator, synchronization, elite selection, and proposal updates). Cold start, weight loading, and environment setup are excluded from all arms. Paired success differences are tested with the exact McNemar test on episodes and bootstrap intervals over sources, clustered where two seeds share sources (per-seed results in Appendix N). For stage-wise analyses we also report the first-action deviation and the distance between the proposal mean of the perturbed search and that of native CEM at the end of the perturbed stage. Timing is compared only within one hardware platform.

Table 1: Early against late substitution of the local operator, paired against the native planner on the same sources. Success is followed by the McNemar p value against native; Δcost is the paired difference in exact final cost with a 95% bootstrap interval. PushT rows use Fast-LeWM with different planners or elite counts (K), except the LeWM row, which uses the original LeWM world model and the endpoint cost. Every arm is listed in Appendix D.

				Early substitution		Late substitution	
Task	Planner	Sources	Native	Success (p)	Δcost	Success (p)	Δcost
PushT	CEM	200	182	173 (0.049)	$+5.50$ [1.87, 9.86]	182 (1.00)	$+0.71$ [-0.73, 1.89]
PushT	iCEM	50	27	15 (0.023)	$+55.70$ [30.48, 81.34]	27 (1.00)	$+8.14$ [-5.91, 21.32]
PushT	MPPI	50	45	35 (0.006)	$+29.68$ [13.98, 48.36]	46 (1.00)	$+1.50$ [0.61, 2.60]
PushT	CEM, $K = 15$	50	46	44 (0.50)	$+4.07$ [1.90, 6.78]	46 (1.00)	$+1.78$ [0.12, 3.80]
PushT	CEM, $K = 60$	50	46	44 (0.50)	$+3.19$ [0.50, 6.55]	46 (1.00)	$+0.64$ [0.09, 1.20]
PushT	CEM, LeWM	50	42	43 (1.00)	$+5.16$ [-1.44, 14.89]	44 (0.63)	$+1.23$ [-0.43, 3.48]
Two-Room	CEM	100	94	88 (0.07)	$+16.73$ [6.17, 29.38]	93 (1.00)	$+1.86$ [-0.05, 4.74]
Two-Room	MPPI	50	48	39 (0.004)	$+81.14$ [49.41, 115.77]	49 (1.00)	-2.60 [-7.77, -0.01]
Reacher	MPPI	50	40	42 (0.73)	$+22.70$ [-0.84, 47.47]	42 (0.63)	-0.26 [-14.73, 14.40]
Reacher	CEM	100	79	91 (0.017)	$+7.34$ [-9.25, 24.80]	79 (1.00)	-11.32 [-23.57, 0.71]

6 RESULTS

6.1 THE WORLD MODEL BECOMES LOCALLY PREDICTABLE AS THE PROPOSAL CONTRACTS

We first measure how well the local operator of Section 4 describes the world model at each round of native CEM. On 100 fresh PushT sources (two decisions each), we run native CEM unchanged and, at every round, fit the operator from 24 exact sentinels on the actual population and compare its predictions for the remaining 576 candidates with their exact predictions. Figure 2(a,b) reports the result (numbers in Table 6, Appendix G).

As the proposal contracts, the operator's representation error roughly halves, and the distance between the updates induced by predicted and exact scores falls by a factor of about 2.7. On Reacher (50 fresh sources), as $\bar{\sigma}$ contracts from 1.00 to 0.25, the held-out error falls from 1.73 to 0.38 and the update distance from 1.81 to 0.43. What improves is representation fidelity, not selection fidelity: the operator's cost ranking correlates only modestly with the exact ranking at every round (Spearman 0.21–0.28 on PushT) and recovers about 14% of the exact elite set. The smaller update distance largely reflects the contraction itself, since a changed elite set shifts the mean by an amount proportional to the proposal scale (Appendix A); relative to the exact step it does not shrink. The same selection error therefore displaces the search far more early, when every later round inherits the displacement, than late (Figure 2b).

6.2 LATE APPROXIMATION PRESERVES SUCCESS; EARLY APPROXIMATION CAN DERAIL THE SEARCH

We next replace the exact evaluations of one stage at a time, either with the local operator, fitted at the first round of the stage and reused for the rest, or with Gaussian score noise whose standard deviation is 0.25 times the within-round standard deviation of the exact costs. On PushT the stages are rounds 0–3, 4–7, and 8–11, with rounds 12–14 exact; on Reacher and Two-Room, with 30 rounds, early and late are rounds 0–7 and 16–23, with rounds 24–29 exact. Table 1 compares early with late substitution in every setting; Table 3 (Appendix D) lists every arm, including the matched-noise controls, and Figure 2(c) shows the drift on PushT.

On PushT, substitution moves the search less the later it is made. In two sets of 50 sources with different search seeds, substitution in rounds 0–3 moves the proposal nearly twice as far from native CEM as substitution in rounds 8–11, with non-overlapping intervals, and in the second set raises the final cost by 11.3 against 0.6–0.9. Matched noise drifts nearly equally at every stage. Noise one to two times stronger, which disturbs elite selection about as much as the operator (elite overlap 0.11–0.27 against 0.14), does raise the final cost more early than late, but it moves the proposal less than the operator (3.6 against 5.1) and leaves success unchanged (Appendix J). On 200 further sources, early substitution lowers success significantly (13 decisions solved only by native CEM, 4 only by the arm), whereas late substitution keeps every success and plans 19% faster. The ordering holds when the planner is iCEM or MPPI with the same world model, where early against late is

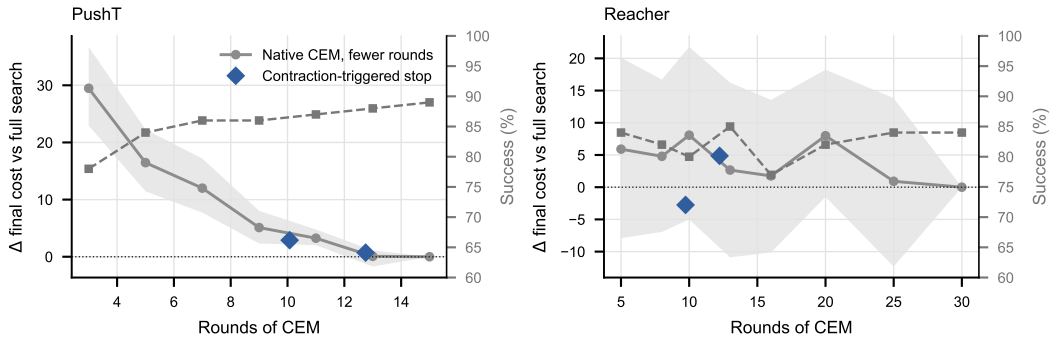


Figure 3: How much each round of native CEM contributes. Paired difference in final cost (solid, left axis; shaded 95% interval) and success rate (dashed, right axis) when native CEM is stopped after fewer rounds, on 100 fresh PushT sources (left, full search 15 rounds) and 100 Reacher decisions (right, 50 fresh sources and two search seeds, full search 30 rounds). Success uses the same scale in both panels. Diamonds: stopping each decision when the proposal-mean step falls below a threshold fixed in advance, plotted at the mean number of rounds used.

significant ($p = 0.017$ and $p = 0.001$) and matched noise does not reproduce it (Appendix J), and when the elite count is 15 or 60, where early substitution moves the proposal 2.9 and 2.4 times as far as late substitution. With a second world model, LeWM, early substitution again drifts much further (6.53 against 2.45), but cost and success do not differ detectably.

Two-Room, on which neither the operator nor the schedules were developed, shows the same asymmetry. Over two disjoint sets of 50 sources, early substitution raises the final cost significantly and solves 88/100 against 94/100 (7 / 1, $p = 0.07$), while late substitution solves 93/100 and plans 24% faster; on the second set, matched noise in rounds 0–7 moves the proposal much less than the operator (3.34 against 5.71) and loses no success. With MPPI on the same second set, early substitution lowers success from 48/50 to 39/50 (9 / 0, $p = 0.004$), and late substitution solves 49/50. Reacher does not. Early substitution still perturbs the search more than late substitution (drift 4.89 against 4.47), but it solves more decisions than native CEM (5 / 17), and an earlier set of 50 Reacher sources pointed the same way, as does MPPI (42/50 early and late against 40/50). On 50 further Reacher sources with matched noise, no arm changes success; early substitution raises the mean final cost ($+28.9$ [6.7, 51.2], per-source median near zero), whereas early noise does not (-2.7 [-16.6, 10.1]) and moves the proposal less (drift 3.57 against 4.79; Appendix J). Reacher's flat rounds curve (Section 6.3) suggests that its search is decided within the first few rounds.

6.3 Late-round compute has little effect on success on all three tasks

If late rounds carry little of the decision, their compute should be removable without loss. Figure 3 shows this directly on 100 fresh PushT sources: stopping native CEM after 13 of 15 rounds leaves the final cost unchanged ($+0.07$ [-1.71, 1.12]), and each further round removed raises it steeply ($+3.3$ at 11 rounds, $+12.0$ at 7, $+29.5$ at 3), while success declines only slowly until the search is cut to three rounds. On Reacher the curve is flat in success: from 5 to 30 rounds native CEM solves between 77 and 85 of 100 decisions against 84 for the full search, and no arm differs significantly (every McNemar $p \geq 0.17$). The final model cost keeps falling with more rounds (per-source median $+0.52$ at 5 rounds, $+0.03$ at 25), but this refinement does not change the outcome. On Two-Room, stopping after 28 of 30 rounds leaves success and final cost unchanged (47/50; $+0.14$ [-0.23, 0.46]), and stopping after 15 keeps success (47/50) at about half the planning time but raises the final cost by 8.0 [5.1, 11.4]. We test this against matched-compute baselines on 100 fresh PushT sources with two search seeds (200 paired decisions). Besides native CEM ($N = 600$, $R = 15$) we run E12 (Section 4; two local rounds), E8L4 (four local rounds, rounds 8–11, rounds 12–14 exact), native CEM with a smaller population matched to either query budget, and native CEM stopped after 13 rounds. Table 2 reports the pooled result.

The comparison confirms what the asymmetry predicts: late-round compute can be removed with little effect on control. For every arm, the 95% interval of the paired success difference from native

Table 2: Matched-compute comparison on 100 fresh PushT sources $\times$ two search seeds (200 paired decisions). McNemar counts: decisions solved only by native CEM / only by the arm. Δcost: paired difference in exact final cost (source-clustered 95% bootstrap interval). Time: mean complete planning time per decision on one RTX 6000D GPU.

Planner	Success	McNemar (p)	Δcost	Exact queries	Time (s)
Native CEM, $R = 15$	178/200	–	–	9,001	0.157
E2L-CEM, E12	178/200	0 / 0 (1.00)	$+0.19$ [-0.46, 0.85]	7,825	0.152
E2L-CEM, E8L4	176/200	5 / 3 (0.73)	$+1.27$ [0.12, 2.77]	6,625	0.131
Native CEM, $N = 522$	176/200	4 / 2 (0.69)	$+0.20$ [-0.42, 0.85]	7,831	0.161
Native CEM, $N = 442$	177/200	4 / 3 (1.00)	$+0.66$ [-0.32, 1.74]	6,631	0.158
Native CEM, $R = 13$	179/200	1 / 2 (1.00)	$+0.69$ [0.07, 1.31]	7,801	0.138

CEM lies between -3.5 and $+2.5$ percentage points, and E12 has the same success outcome as native CEM on every one of the 200 decisions (0 / 0). Shrinking the population to the same query budgets does not save time, because the candidates of a round are evaluated in parallel, so the number of rounds, not the number of queries, is the relevant cost on this hardware. Among the planners that do save time, the choice is a trade-off between time and final cost: E12 is the only one with no detectable increase in final cost, at a 3% time saving, whereas R13 and E8L4 save 12% and 17% at a small but significant increase in final cost. iCEM in the configuration of our earlier development runs, not retuned for this protocol, solves 119/200 (Appendix C).

On Reacher (50 fresh sources, two search seeds), native CEM ($N = 300$, $R = 30$) solves 84/100 in 0.193 s per decision, E12 88/100 in 0.108 s, stopping after 13 rounds 92/100 in 0.096 s, and a population of 131 at the same query budget 85/100 in 0.189 s; over this set and the rounds-curve set, 13 rounds solve 177/200 against 168/200.

Stopping each decision once the proposal-mean step falls below a threshold fixed in advance uses 12.8 of 15 rounds on PushT and 12.2 of 30 on Reacher and lands on the fixed-round curve on both (Figure 3, diamonds); larger early populations at the same query count did not help either (Appendix F).

7 Discussion and Conclusion

We asked where a world-model planner's test-time compute matters. For CEM with pretrained latent world models, late-round compute is dispensable on all three tasks we studied: approximating the late rounds with a run-time local operator or removing them did not change success detectably, nor did stopping adaptively on the two tasks where we tested it. The early rounds are where the decision is made. The world model becomes locally predictable only as the proposal contracts, and approximating the early rounds damaged the search on PushT for three planners and on Two-Room for two, whereas score noise of the same cost scale did not on any task. Across the three tasks, early approximation hurt where stopping the search at about half its rounds raised the final cost significantly (PushT, Two-Room) and not where the rounds curve is flat (Reacher, where it even raised success), which suggests the rounds curve, measured offline, as a diagnostic of where exact evaluations are needed. In a test whose predictions were fixed before running, the two agreed in all eight further settings we checked: iCEM, MPPI, two elite counts, and a second world model on PushT, CEM and MPPI on a second Two-Room source set, and MPPI on Reacher (Appendix J). For the planners and models tested, the practical rule is to cut late-round compute first.

Limitations. Early approximation raised success on Reacher, for reasons we have not tested, and its success loss on Two-Room under CEM falls short of significance ($p = 0.07$), although its cost increase is significant. A second world model, iCEM (untuned), and other elite counts were tested only on PushT, where the second model showed the asymmetry in drift but not in outcomes, and the stage boundaries are fixed by hand. Stronger noise also raises the final cost more early than late, so part of the early sensitivity is generic to perturbing early selection.

Reproducibility Statement

The method is fully specified by Section 4 and Appendix B; it adds no trained component to the planner. Appendix C gives the source-selection procedure, seeds, and settings of every experiment, and Appendix I lists every paired block. The planner code, source manifests, per-episode outcomes, and timing logs will be released upon publication.

AI Use Statement

AI tools were used in four ways. (i) Writing: drafting sections of the paper, polishing text, and organizing the LaTeX source. (ii) Retrieval: searching for related work and checking bibliographic records against primary sources. (iii) Research ideation and execution: proposing and discussing experimental designs and controls, writing and running experiment and analysis code under the fixed protocols of Section 5, and producing figures and tables from the recorded outputs. (iv) Review: critiquing drafts, which informed revisions. All numbers in the paper are read from the recorded experiment outputs; no result was generated by an AI tool without being run. The authors decided the research questions and claims, checked the manuscript, code, data handling, numerical values, protocol conditions, and source attribution, and are responsible for the final submission.

References

Brandon Amos and Denis Yarats. The differentiable cross-entropy method. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 291–302, 2020. URL https://proceedings.mlr.press/v119/amos20a.html.

Mahmoud Assran et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Raul Astudillo and Peter I. Frazier. Bayesian optimization of composite functions. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 354–363, 2019. URL https://proceedings.mlr.press/v97/astudillo19a.html.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, et al. Genie: Generative interactive environments. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 4603–4623, 2024.

Niccolò Caselli, Francesco Massafra, Samuele Punzo, Salvatore Lo Sardo, Ippokratis Pantelidis, and Sathya Kamesh Bhethanabhotla. Mind the gap: Promises and pitfalls of hierarchical planning in LeWorldModel. *arXiv preprint arXiv:2607.12547*, 2026. URL https://arxiv.org/abs/2607.12547.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2024. URL https://arxiv.org/abs/2303.04137.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Junha Chun, Youngjoon Jeong, and Taesup Kim. Sparse imagination for efficient visual world model planning. *arXiv preprint arXiv:2506.01392*, 2025. URL https://arxiv.org/abs/2506.01392.

Ignasi Clavera, Jonas Rothfuss, John Schulman, Yasuhiro Fujita, Tamim Asfour, and Pieter Abbeel. Model-based reinforcement learning via meta-policy optimization. In *Proceedings of the 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pp. 617–629, 2018.

Jadelynn Dao, Milan Ganai, Yasmina Abukhadra, Ajay Sridhar, Mozhgan Nasr Azadani, Katie Luo, Clark Barrett, Jiajun Wu, Chelsea Finn, and Marco Pavone. DIRECT: When and where should you allocate test-time compute in embodied planners? *arXiv preprint arXiv:2606.12402*, 2026. URL https://arxiv.org/abs/2606.12402.

Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134:19–67, 2005.

Frederik Ebert, Chelsea Finn, Alex X. Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. In *Conference on Robot Learning*, pp. 261–270, 2017.

Mingzhou Fan, Byung-Jun Yoon, Edward Dougherty, Nathan Urban, Francis Alexander, Raymundo Arróyave, and Xiaoning Qian. Multi-fidelity bayesian optimization with multiple information sources of input-dependent fidelity. In *Proceedings of the 40th Conference on Uncertainty in Artificial Intelligence*, 2024.

Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I. Jordan, Joseph E. Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning. *arXiv preprint arXiv:1803.00101*, 2018. URL https://arxiv.org/abs/1803.00101.

Alexander I. J. Forrester, András Sóbester, and Andy J. Keane. Multi-fidelity optimization via surrogate modelling. *Proceedings of the Royal Society A*, 463(2088):3251–3269, 2007. doi: 10.1098/rspa.2007.1900.

Yuntian Gao and Xiangyu Xu. Fast LeWorldModel. *arXiv preprint arXiv:2606.26217*, 2026. URL https://arxiv.org/abs/2606.26217.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018. URL https://arxiv.org/abs/1803.10122.

Danijar Hafner, Timothy Lillicrap, Jimmy Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *arXiv preprint arXiv:1811.04551*, 2018. URL https://arxiv.org/abs/1811.04551.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019. URL https://arxiv.org/abs/1912.01603.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. In *International Conference on Learning Representations*, 2021.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023. URL https://arxiv.org/abs/2301.04104.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022. URL https://arxiv.org/abs/2203.04955.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Nikolaus Hansen. A global surrogate assisted CMA-ES. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 664–672, 2019. doi: 10.1145/3321707.3321842. URL https://doi.org/10.1145/3321707.3321842.

Hsiang-Wei Huang, Jianxu Shangguan, Junbin Lu, and Jenq-Neng Hwang. LeFlow: Generative latent flow planning for world models. *arXiv preprint arXiv:2608.24855*, 2026. URL https://arxiv.org/abs/2608.24855.

Kevin Huang, Sahin Lale, Ugo Rosolia, Yuanyuan Shi, and Anima Anandkumar. CEM-GD: Cross-entropy method with gradient descent planner for model-based reinforcement learning. *arXiv preprint arXiv:2112.07746*, 2021. URL https://arxiv.org/abs/2112.07746.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, 2019.

Yuhua Jiang, Junjie Lu, and Feifei Gao. Sampling headroom is not selection gain: A compute-value audit of test-time scaling for video world models. *arXiv preprint arXiv:2609.13257*, 2026. URL https://arxiv.org/abs/2609.13257.

Zhengyao Jiang, Tianjun Zhang, Michael Janner, Yueying Li, Tim Rocktäschel, Edward Grefenstette, and Yuandong Tian. Efficient planning in a compact latent action space. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2208.10291.

Yaochu Jin, Markus Olhofer, and Bernhard Sendhoff. A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on Evolutionary Computation*, 6(5):481–494, 2002.

S. V. Jyothir, Siddhartha Jalagam, Yann LeCun, and Vlad Sobal. Gradient-based planning with world models. *arXiv preprint arXiv:2312.17227*, 2023. URL https://arxiv.org/abs/2312.17227.

Kirthevasan Kandasamy, Gautam Dasarathy, Junier B. Oliva, Jeff Schneider, and Barnabás Póczos. Gaussian process bandit optimisation with multi-fidelity evaluations. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Baris Kayalibay, Atanas Mirchev, Patrick van der Smagt, and Justin Bayer. Tracking and planning with spatial world models. In *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, volume 168 of *Proceedings of Machine Learning Research*, pp. 124–137, 2022. URL https://proceedings.mlr.press/v168/kayalibay22a.html.

Kaushik Lakshminarasimhan, Seren Zhu, and Dora Angelaki. Planning by active sensing. In *Proceedings of the 2nd Gaze Meets ML Workshop*, volume 226 of *Proceedings of Machine Learning Research*, pp. 125–141, 2024. URL https://proceedings.mlr.press/v226/lakshminarasimhan24a.html.

Ilya Loshchilov, Marc Schoenauer, and Michèle Sebag. Self-adaptive surrogate-assisted covariance matrix adaptation evolution strategy. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 321–328, 2012. URL https://arxiv.org/abs/1204.2356.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15762–15772, 2024. doi: 10.1109/CVPR52733.2024.01492.

Lucas Maes et al. LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026. URL https://arxiv.org/abs/2603.19312.

Vincent Micheli, Eloi Alonso, and Francois Fleuret. Transformers are sample-efficient world models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=vhFu1Acb0xa.

Paavo Mikkola, Javier Martinelli, Louis Filstroff, and Samuel Kaski. Multi-fidelity bayesian optimization with unreliable information sources. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 7425–7454, 2023. URL https://proceedings.mlr.press/v206/mikkola23a.html.

Robert J. Moss. Cross-entropy method variants for optimization. *arXiv preprint arXiv:2009.09043*, 2020. URL https://arxiv.org/abs/2009.09043.

Anusha Nagabandi, Gregory Kahn, Ronald S. Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. *arXiv preprint arXiv:1708.02596*, 2018. URL https://arxiv.org/abs/1708.02596.

Benjamin Peherstorfer, Karen Willcox, and Max Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review*, 60(3):550–591, 2018. doi: 10.1137/16M1082469.

Phu Pham and Aniket Bera. Latent energy action planning with world models. *arXiv preprint arXiv:2609.03294*, 2026. URL https://arxiv.org/abs/2609.03294.

Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. In *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 1049–1065. PMLR, 2021. URL https://proceedings.mlr.press/v155/pinneri21a.html.

Reuven Y. Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operational Research*, 99(1):89–112, 1997.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, Chess and Shogi by planning with a learned model. *Nature*, 588:604–609, 2020. doi: 10.1038/s41586-020-03051-4. URL https://doi.org/10.1038/s41586-020-03051-4.

Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and Pieter Abbeel. Masked world models for visual control. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 1332–1344, 2023. URL https://proceedings.mlr.press/v205/seo23a.html.

Eren Sezener and Peter Dayan. Static and dynamic values of computation in mcts. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence*, volume 124 of *Proceedings of Machine Learning Research*, pp. 31–40, 2020. URL https://proceedings.mlr.press/v124/sezener20a.html.

Richard S. Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163, 1991.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. DeepMind control suite. *arXiv preprint arXiv:1801.00690*, 2018. URL https://arxiv.org/abs/1801.00690.

Basile Terver, Tsung-Yen Yang, Jean Ponce, Adrien Bardes, and Yann LeCun. What drives success in physical planning with joint-embedding predictive world models? *arXiv preprint arXiv:2512.24497*, 2025. URL https://arxiv.org/abs/2512.24497.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. doi: 10.1016/j.simpa.2020.100022.

Tingwu Wang and Jimmy Ba. Exploring model-based planning with policy networks. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1906.08649.

Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Grady Williams, Andrew Aldrich, and Evangelos A. Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015. URL https://arxiv.org/abs/1509.01149.

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M. Rehg, Byron Boots, and Evangelos A. Theodorou. Information theoretic mpc for model-based reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation*, pp. 1714–1721, 2017.

Shoubin Yu, Yue Zhang, Zun Wang, Jaehong Yoon, Huaxiu Yao, Mingyu Ding, and Mohit Bansal. When and how much to imagine: Adaptive test-time scaling with world models for visual spatial reasoning. *arXiv preprint arXiv:2602.08236*, 2026. URL https://arxiv.org/abs/2602.08236.

Lunjun Zhang, Ge Yang, and Bradly C. Stadie. World model as a graph: Learning latent landmarks for planning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12611–12620, 2021. URL https://proceedings.mlr.press/v139/zhang21x.html.

Wancong Zhang, Basile Terver, Artem Zholus, Soham Chitnis, Harsh Sutaria, Mido Assran, Randall Balestriero, Amir Bar, Adrien Bardes, Yann LeCun, and Nicolas Ballas. Hierarchical planning with latent world models. *arXiv preprint arXiv:2604.03208*, 2026. URL https://arxiv.org/abs/2604.03208.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79115–79135, 2025. URL https://proceedings.mlr.press/v267/zhou25t.html.

A WHY THE VALIDITY OF A LOCAL APPROXIMATION DEPENDS ON THE SEARCH STAGE

This section makes the argument of Section 3.2 precise under a smoothness assumption. It explains the trend we measure; it is not used to choose the schedule.

Local error scales with the proposal width. Let $z(a) = F_\theta(h, a)$, $D_r = \mathrm{diag}(\sigma_r)$ with $\sigma_{r,k} > 0$, and $G_r = \partial z/\partial a|_{\mu_r}$ (we write G for the Jacobian to avoid a clash with the cost J). For a candidate $a = \mu_r + D_r\epsilon$ with $\epsilon \sim \mathcal{N}(0, I)$, suppose the operator norm of the second derivative of z is at most L on a neighbourhood containing the evaluated candidates and the segments joining them to μ_r (for a Gaussian proposal this is a high-probability region, not the full support). Taylor's theorem gives

$$z(\mu_r + D_r\epsilon) = z(\mu_r) + G_r D_r \epsilon + \rho_r(\epsilon), \qquad \|\rho_r(\epsilon)\| \le \tfrac{L}{2}\|D_r\epsilon\|^2 \le \tfrac{L}{2}\bar{\sigma}_r^2\|\epsilon\|^2, \quad \bar{\sigma}_r = \max_k \sigma_{r,k}.$$

The ideal first-order response is therefore $G_r D_r \epsilon$. The fitted operator has coefficient matrix $\widehat{B}^\mathsf{T}$, so its error decomposes as

$$\big\|z(\mu_r + D_r\epsilon) - \widehat{z}(\epsilon)\big\| \le \tfrac{L}{2}\|D_r\epsilon\|^2 + \big\|G_r D_r - \widehat{B}^\mathsf{T}\big\|\,\|\epsilon\|, \qquad \widehat{z}(\epsilon) = z(\mu_r) + \widehat{B}^\mathsf{T}\epsilon.$$

The first term is the irreducible Taylor error of a linear response; the second is finite-sentinel, weighting, ridge, and numerical fitting error. Proposal contraction therefore predicts a smaller local-linearization error, but does not by itself bound the error of the fitted operator, its cost ranking, or its elite set. This is the mechanism tested empirically in Section 6.1; it is not a schedule-selection guarantee. The bound above describes an operator refitted at each round, which is how the per-round diagnostic of Section 6.1 is computed.

A fixed operator reused across local rounds. E2L fits the operator once, at round $s = R_E$, with anchor $a_\star = \mu_s$, scale $D_\star = D_s$, and reference $y_\star = z(a_\star)$, and reuses $\widehat{y}(a) = y_\star + \widehat{B}^\mathsf{T} D_\star^{-1}(a - a_\star)$ in the later local rounds. With $G_\star = G_s$ and the same smoothness assumption on the segment $[a_\star, a]$,

$$\big\|z(a) - \widehat{y}(a)\big\| \le \tfrac{L}{2}\|a - a_\star\|^2 + \big\|G_\star D_\star - \widehat{B}^\mathsf{T}\big\|\,\big\|D_\star^{-1}(a - a_\star)\big\|.$$

A candidate of a later local round is $a = \mu_r + D_r\epsilon$, so $a - a_\star = (\mu_r - \mu_\star) + D_r\epsilon$ and $\mathbb{E}\|a - a_\star\|^2 = \|\mu_r - \mu_\star\|^2 + \|D_r\|_F^2$. The reused operator therefore has two error sources: the current proposal scale and the drift of the proposal mean away from the anchor. Both are small late in the search, which is where E2L places its local rounds.

Errors that change the elite set can propagate. Let the local round scores be $\widehat{J}_{r,i} = J_{r,i} + e_{r,i}$, sort the exact costs as $J_{(1)} \le \cdots \le J_{(K)} < J_{(K+1)} \le \cdots \le J_{(N)}$, and let $\gamma_r = J_{(K+1)} - J_{(K)}$ be the selection margin. If $\max_i |e_{r,i}| < \gamma_r/2$, then $\widehat{\mathcal{E}}_r = \mathcal{E}_r$ and the CEM update is unchanged. If the elite set changes, the mean shifts by

$$\widehat{\mu}_{r+1} - \mu_{r+1} = D_r\Big(\tfrac{1}{K}\textstyle\sum_{i\in\widehat{\mathcal{E}}_r} \epsilon_{r,i} - \tfrac{1}{K}\sum_{i\in\mathcal{E}_r} \epsilon_{r,i}\Big),$$

which is proportional to the proposal scale, and the next proposal can change:

$$\widehat{\mathcal{E}}_r \neq \mathcal{E}_r \quad\Longrightarrow\quad (\widehat{\mu}_{r+1}, \widehat{\sigma}_{r+1}) \text{ may differ from } (\mu_{r+1}, \sigma_{r+1}).$$

Later populations are then sampled from the perturbed proposal, so the difference can influence later rounds even when those rounds are evaluated exactly; it need not persist or grow monotonically. The Taylor term above grows with the distance from the reference, and a broad proposal places more probability mass at larger such distances. This gives a mechanism for stronger early sensitivity, whereas matched score noise has no distance-dependent structure. The stage ordering in Table 3 is therefore empirical evidence for this mechanism, not a consequence of this argument alone.

B Implementation of the local response operator

Sentinels. At the fit round, the sentinels are the first M candidates of the population, where candidate zero is set to the proposal mean. The sentinel set is therefore fixed by the sampling seed and does not depend on any score.

Fit. The prediction target is exactly the representation consumed by the native task cost: PushT concatenates the terminal latents of the direct and composed-prefix predictions, whereas endpoint-only tasks use the terminal latent. The reference is the sentinel nearest to the proposal mean in whitened coordinates, which is the mean itself. Let $X \in \mathbb{R}^{M\times q}$ have rows δ_j^{T} and let $Y \in \mathbb{R}^{M\times d}$ have rows $(y_j - y^0)^{\mathsf{T}}$. The implementation uses

$$\omega_j = \exp\!\left(-\frac{\|\delta_j\|_2^2}{2q}\right), \qquad D_\omega = \operatorname{diag}(\sqrt{\omega_0}, \ldots, \sqrt{\omega_{M-1}}),$$

$$\widehat{B} = \arg\min_{B\in\mathbb{R}^{q\times d}} \|D_\omega(XB - Y)\|_F^2 + \lambda\|B\|_F^2, \qquad \lambda = 10^{-4}\frac{\|D_\omega X\|_F^2}{q}.$$

For the degree-one fit used in all reported E2L runs, $M < q$, so the implementation evaluates the equivalent dual solution

$$\widehat{B} = (D_\omega X)^{\mathsf{T}}\left[(D_\omega X)(D_\omega X)^{\mathsf{T}} + \lambda I_M\right]^{-1} D_\omega Y.$$

The fit runs in double precision. The intercept is anchored at the exact reference y^0, so the ridge fit estimates only the residual response map, and the square-root weights used in the computation correspond to the objective weights ω_j.

Scoring. Local rounds predict the representation of all N candidates with the fitted operator and apply the native cost, the endpoint term plus the prefix-consistency term with $\beta = 1$. Elite count is $K = 30$. The operator is fitted once per decision and never refitted within a decision.

Timing boundary. Complete planning time is measured per decision over the full solver call and includes candidate sampling, all search queries, fitting and applying the operator, synchronization, elite selection, proposal updates, and the one exact audit query for the returned proposal.

C Experimental details

Source selection. For each experiment, an outcome-blind builder scans the recorded run artifacts of the project, removes every (episode, start step) identity that appears in any earlier run, and draws the required number of sources from the remaining dataset positions with a fixed seed. It never reads success labels. Sources of different experiments are disjoint.

Operator accuracy (Table 6). 100 PushT sources, two decisions each, search seed 51, protocol seed 20260925, native CEM with $N = 600$, $R = 15$, $K = 30$, $\beta = 1$, run on one RTX 6000D GPU. At every round the operator is fitted from the first 24 candidates of the actual native population (candidate zero is the proposal mean) and evaluated on the other 576; the native search itself is not altered. With Y and $\widehat{Y}$ the exact and predicted responses of the held-out candidates (one row each) and $C(Y) = Y - \mathbf{1}\bar{y}^{\mathsf{T}}$, the held-out relative error is $\|\widehat{Y} - Y\|_F / \|Y\|_F$, the centred relative error is $\|C(\widehat{Y}) - C(Y)\|_F / \|C(Y)\|_F$, and the update distance is $d_\mu = \|\widehat{\mu}_{r+1} - \mu_{r+1}\|_2$, where both means are refitted from the same population with predicted and exact scores.

Matched-compute comparison (Table 2). 100 further PushT sources, search seeds 51 and 52 on the same sources, protocol seed 20260925, one RTX 6000D GPU. Confidence intervals resample sources as clusters so that both seeds of a source stay together. iCEM uses the configuration tuned in our earlier comparison ($N = 600$, $R = 15$, elite 30, population decay 1.0, elite reuse 0.3, colored-noise exponent 2, momentum 0.1) and was not re-tuned for this protocol. On the 200 decisions of Table 2 it solves 119/200 (McNemar $p < 10^{-3}$ against native CEM) with a final-cost difference of $+48.95$ [34.18, 63.96]; on the 100 Reacher decisions of the matched comparison it solves 77/100. We report it as a reference configuration, not as a tuned competitor.

Stage-wise substitution (Table 3). First set: 50 PushT sources, two planning decisions each, search seed 51, protocol seed 20260926; replication: 50 further sources, search seed 52, protocol seed 20260925; $N = 600$, $R = 15$, $K = 30$, $\beta = 1$, run on one RTX 6000D GPU. Stages are rounds 0–3, 4–7, and 8–11; rounds 12–14 are exact in every arm. In a substitution arm, the local operator is fitted from 24 exact sentinels at the first round of the stage and reused for its remaining three rounds, so the arm uses $11N + 24 + 1 = 6{,}625$ exact queries per decision, including one exact evaluation of the returned proposal. In a noise arm, every candidate score in the stage receives independent Gaussian noise with standard deviation 0.25 times the standard deviation of the exact costs of that round's population, drawn from a dedicated random stream. The drift of an arm is the L2 distance between its proposal mean and that of native CEM at the last round of its stage, on the same source and decision. The 200-source PushT set uses search seed 51 on sources disjoint from all other sets, with the early and late substitution arms only; stage-end proposal means were not logged there. On Reacher ($N = 300$, $R = 30$, endpoint cost, one A100 GPU), stages are rounds 0–7, 8–15 (first set only), and 16–23, with rounds 24–29 exact and $M = 32$ sentinels, giving $6{,}633$ exact queries per decision; the first set has 50 sources and the second 100 sources with search seed 42. On Two-Room, the arms replace the solver of the official evaluator (Section 5); stages are rounds 0–7 and 16–23 with rounds 24–29 exact and $M = 24$, giving $6{,}625$ exact queries per decision. Stage-end means of native CEM on Two-Room were obtained from a deterministic re-run with stage-end logging, whose success and final costs match the reported native run exactly.

Other planners (Section 6.2). 50 further PushT sources, search seed 51, protocol seed 20260925, endpoint cost with prefix consistency $\beta = 1$, $N = 600$ and 15 iterations, one A100 GPU. iCEM uses the configuration of the matched comparison, with warm start across decisions. MPPI starts each decision from mean 0 and standard deviation 1, with candidate zero set to the current mean; weights are a softmax over all N candidates, $w_i \propto \exp(-(J_i - \min_j J_j)/\lambda)$ with $\lambda = 0.1$ times the standard deviation of that iteration's costs; the mean is the weighted mean and the per-dimension standard deviation the weighted standard deviation around it, floored at 0.05, with no smoothing. The early and late arms fit the operator from 24 exact sentinels at iteration 0 or 8 and reuse it for the next three iterations; the noise arms perturb the same iterations as in the CEM experiments. Timing was measured on a different host from the CEM experiments and is not compared with them.

Component ablation (Table 7). One eight-source PushT block, search seed 51, protocol seed 20260925, same settings as the main comparison, run on one RTX 6000D GPU. The absolute variant fits a linear map from whitened actions to predictions without subtracting the exact reference prediction; the scalar variant fits the planning cost directly around the reference. Elite overlap and drift are computed by replaying each local round of the approximate arm with exact scores on the same population.

E12 blocks (Appendix E). PushT blocks use search seed 51, protocol seed 20260925, $N = 600$, $R = 15$, $K = 30$, $\beta = 1$, with two decisions per source. Reacher blocks use $N = 300$, $R = 30$, and the endpoint cost. Both arms of a block run on the same GPU.

D STAGE-WISE SUBSTITUTION: ALL ARMS

Table 3 lists every arm of the CEM stage-wise experiments summarized in Table 1; the corresponding experiments for iCEM, MPPI, and other elite counts are in Appendix J.

Table 3: Stage-wise substitution and matched-noise controls on paired sources; native CEM's success is given for each source set. Drift: L2 distance between the proposal means of the perturbed search and of native CEM at the end of the perturbed stage (95% bootstrap interval; not logged in the 200-source PushT set, and given per set for Two-Room in Table 10). Δcost: paired difference in exact final cost, arm minus native (95% interval).

Perturbation	Rounds	Success	Drift	Δcost
PushT, first source set, search seed 51 (native 47/50)				
Local operator	0–3	45/50	5.10 [4.86, 5.36]	$+1.02$ [-0.62, 2.51]
Local operator	4–7	47/50	3.38 [3.15, 3.64]	$+1.71$ [-0.55, 4.61]
Local operator	8–11	46/50	2.79 [2.61, 3.00]	$+0.44$ [-0.81, 1.53]
Matched noise	0–3	46/50	2.38 [2.27, 2.48]	$+0.69$ [-0.26, 1.80]
Matched noise	4–7	47/50	2.59 [2.42, 2.79]	$+1.26$ [0.18, 2.86]
Matched noise	8–11	47/50	2.38 [2.23, 2.53]	$+0.61$ [0.09, 1.15]
Matched noise	14	48/50	1.81 [1.67, 1.96]	$+0.05$ [-1.01, 0.81]
PushT, replication: new source set, search seed 52 (native 48/50)				
Local operator	0–3	46/50	5.12 [4.84, 5.41]	$+11.28$ [1.79, 23.98]
Local operator	4–7	47/50	3.07 [2.88, 3.30]	$+0.89$ [0.18, 1.99]
Local operator	8–11	47/50	2.63 [2.46, 2.83]	$+0.60$ [0.09, 1.20]
PushT, 200 new sources, search seed 51 (native 182/200)				
Local operator	0–3	173/200	–	$+5.50$ [1.87, 9.86]
Local operator	8–11	182/200	–	$+0.71$ [-0.73, 1.89]
Two-Room, two sets of 50 sources, $R = 30$ (native 94/100)				
Local operator	0–7	88/100	–	$+16.73$ [6.17, 29.38]
Local operator	16–23	93/100	–	$+1.86$ [-0.05, 4.74]
Two-Room, second set only (native 47/50)				
Local operator	0–7	45/50	5.71 [5.44, 6.01]	$+15.07$ [-1.35, 35.55]
Local operator	16–23	46/50	3.81 [3.44, 4.25]	$+2.58$ [-0.80, 8.19]
Matched noise	0–7	47/50	3.34 [3.15, 3.56]	-3.07 [-8.97, 0.31]
Matched noise	16–23	46/50	3.36 [3.09, 3.67]	$+2.41$ [-0.74, 7.56]
Reacher, 100 new sources, $R = 30$ (native 79/100)				
Local operator	0–7	91/100	4.89 [4.72, 5.08]	$+7.34$ [-9.25, 24.80]
Local operator	16–23	79/100	4.47 [3.90, 5.13]	-11.32 [-23.57, 0.71]

E ADDITIONAL EVALUATION OF THE E12 SCHEDULE

We froze the E12 schedule (Section 4) before evaluating it on five fresh PushT blocks, 99 sources in total, run on A100 and RTX 6000D GPUs, and on three Reacher blocks of eight sources each. Each block pairs E2L with native CEM on the same sources. Table 4 pools the blocks; Appendix I lists each block.

On PushT, E2L and native CEM differ on a single source out of 99, and four of the five blocks have identical success sets. E2L reduces exact queries by 13% and complete planning time by 5.9–8.5% in every block. Because only two of the fifteen rounds are local, the saving is set by the schedule rather than by the cost of the operator. On Reacher the schedule makes seventeen of thirty rounds local and plans 39% faster, with a success difference of one source overall; however, the three blocks disagree (8/8 versus 7/8, 7/8 versus 6/8, and 5/8 versus 8/8), so the Reacher result is less stable than the PushT result. The paired differences are not statistically distinguishable from zero on either task.

Table 4: E2L-CEM against native CEM on fresh sources, pooled over blocks. McNemar counts are sources solved only by native CEM / only by E2L. Speed-up is the reduction in mean complete planning time per decision relative to native CEM, computed within each block (range over blocks for PushT; equal-weight mean for Reacher).

Task	Native CEM	E2L-CEM	McNemar (p)	Exact queries	Speed-up
PushT (99 sources)	92/99	91/99	1 / 0 (1.00)	9,001 → 7,825	5.9–8.5%
Reacher (24 sources)	21/24	20/24	3 / 2 (1.00)	9,001 → 3,933	39.3%

F Reallocating late-round budget to early rounds

On 100 fresh PushT sources (search seed 51, protocol seed 20260925, one RTX 6000D GPU), we compared native CEM ($N = 600$, $R = 15$) with native CEM stopped after 13 rounds and with two 13-round schedules that spend the saved budget on larger early populations at the same total of 9,000 exact queries. Elite count is 30 in every round and the CEM update is unchanged.

Table 5: Front-loaded populations at matched exact queries on 100 PushT sources. McNemar counts: sources solved only by native CEM / only by the arm. Δcost: paired difference in exact final cost (95% bootstrap interval).

Planner	Success	McNemar (p)	Δcost	Time (s)
Native CEM, $R = 15$	93/100	–	–	0.157
Native CEM, $R = 13$	92/100	(1.00)	$+0.63$ [0.09, 1.27]	0.137
$N = 900$ in rounds 0–3, then 600	91/100	2 / 0 (0.50)	$+0.85$ [0.13, 1.72]	0.144
$N = 800$ in rounds 0–5, then 600	91/100	2 / 0 (0.50)	$+0.53$ [-0.10, 1.26]	0.137

G Operator accuracy by round

Table 6: Accuracy of the run-time local operator at each round of native CEM on 100 PushT sources (200 decisions). $\bar{\sigma}$: mean proposal scale. Rel. error: held-out relative L2 error of the predicted representation. Mean shift: distance between the proposal means updated with predicted and with exact scores. ρ: Spearman correlation of predicted and exact costs. Overlap: fraction of the top-30 candidates shared (chance 0.05).

Round	0	3	7	11	14
$\bar{\sigma}$	1.00	0.85	0.67	0.52	0.43
Rel. error	0.69	0.67	0.56	0.44	0.36
Mean shift	2.23	1.60	1.25	0.97	0.82
ρ	0.23	0.21	0.23	0.26	0.28
Overlap	0.14	0.14	0.14	0.13	0.14

On the same 100 sources as the rounds curve in Figure 3, we also stopped each decision once the L2 step of the proposal mean fell below a threshold τ, with at least six rounds. The two thresholds were fixed before the run as the median native step at round 12 ($\tau = 0.631$) and at round 9 ($\tau = 0.775$) on the traces of Table 6. With $\tau = 0.631$ the planner used 12.75 rounds on average (between 8 and 15), solved 89/100 like the full search, and raised the final cost by 0.72 [0.47, 1.00] at 0.140 s per decision; fixed 13 rounds raised it by 0.07 at 0.137 s. With $\tau = 0.775$ it used 10.07 rounds, solved 87/100, and raised the final cost by 2.89 [1.86, 4.28] at 0.113 s, close to the fixed-round curve at ten rounds. On Reacher (the 100 decisions of the rounds curve; at least eight rounds, at most 30), the thresholds were fixed in the same way from the round-12 and round-8 medians of the Reacher native traces ($\tau = 0.790$ and $\tau = 0.931$). With $\tau = 0.790$ the planner used 12.2 rounds on average (median 11) and solved 82/100 against 84/100 for the full search (9 / 7, $p = 0.80$) and 85/100 for fixed 13 rounds ($p = 0.55$), at 0.091 s per decision; with $\tau = 0.931$ it used 9.7 rounds and solved 80/100 ($p = 0.52$ against the full search) at 0.088 s. Because the Reacher rounds curve is flat in success, neither rule can improve on a fixed short budget.

H COMPONENT ABLATION OF THE LOCAL OPERATOR

E2L approximates the world-model prediction around an exact reference and applies the native cost to it. We compare this design with two alternatives under the same E12 schedule and exact-query count: fitting the prediction without the exact reference term (an absolute linear fit), and fitting the scalar planning cost directly around the reference. All arms use one fresh eight-source PushT block. Elite overlap and proposal drift are measured on the same populations by replaying each local round with exact scores.

Table 7: Component ablation of the local response operator on one fresh eight-source PushT block. All approximate arms use 7,824 exact queries per decision. Elite overlap is the fraction of the approximate elite set that coincides with the exact elite set on the same population; drift is the distance between the approximate and exact proposal means after the update. Both are averaged over local rounds.

Operator	Success	Speed-up	Elite overlap	Proposal drift
Native CEM (exact)	5/8	–	–	–
Response with exact reference (E2L)	5/8	**7.05%**	**0.118**	**0.174**
Response without reference	5/8	6.65%	0.088	0.195
Scalar-cost surrogate	5/8	1.49%	0.043	0.255

All arms fail on the same three sources, so success does not separate them on this block; the diagnostics do. Predicting the representation rather than the cost gives the largest gain: the scalar-cost surrogate keeps about a third of the elite overlap, drifts furthest from the exact proposal, and retains only a 1.5% speed-up at the same query count. Anchoring the response at the exact reference further raises overlap and reduces drift. Elite overlap remains low in absolute terms for all operators, which is consistent with limiting local rounds to the part of the search where the proposal has already localized.

I PER-BLOCK RESULTS

Table 8 lists every paired block pooled in Table 4. Each block was drawn by the outcome-blind source builder after excluding all previously used sources, and both arms of a block ran on the same GPU. PushT blocks use search seed 51.

Table 8: Paired blocks for the frozen E12 schedule. Speed-up is the reduction in mean complete planning time per decision relative to native CEM within the block.

Task / block	GPU	Sources	E2L-CEM	Native CEM	Speed-up
PushT B1	A100	25	24/25	24/25	6.44%
PushT B2	A100	25	22/25	22/25	8.22%
PushT B3	A100	8	7/8	7/8	6.59%
PushT B4	RTX 6000D	16	15/16	15/16	8.47%
PushT B5	RTX 6000D	25	23/25	24/25	5.90%
Reacher R1	A100	8	8/8	7/8	–
Reacher R2	A100	8	7/8	6/8	–
Reacher R3	A100	8	5/8	8/8	–

The single PushT source on which the arms differ is in B5, where native CEM succeeds and E2L does not. On Reacher, the equal-weight mean complete planning time over the three blocks is 0.1334 s for E2L and 0.2198 s for native CEM.

J ADDITIONAL RESULTS ACROSS TASKS AND PLANNERS

This appendix lists every arm of the experiments summarized in Sections 6.2 and 6.3 that are not tabulated there. All comparisons are paired by source against the native planner of the same experiment. McNemar counts are decisions solved only by the native planner / only by the arm.

Table 9: Stage-wise substitution for iCEM, MPPI, and CEM with elite counts $K = 15$ and $K = 60$, each on 50 PushT sources with the same world model (Section 6.2; the two CEM variants share one source set, disjoint from the iCEM and MPPI set). Drift is measured for the first decision of each source, where every arm starts from the same history. Native success: iCEM 27/50, MPPI 45/50, CEM $K = 15$ 46/50, CEM $K = 60$ 46/50. The last block replaces Fast-LeWM with the original LeWM world model on a further 50 sources; it uses the endpoint cost only, because LeWM has no prefix rollout (native 42/50).

Planner	Perturbation (iterations)	Success	McNemar (p)	Δcost	Drift
iCEM	Local operator (0–3)	15/50	18 / 6 (0.023)	+55.70 [30.48, 81.34]	17.33 [16.40, 18.29]
iCEM	Local operator (8–11)	27/50	7 / 7 (1.00)	+8.14 [−5.91, 21.32]	5.94 [5.38, 6.51]
iCEM	Matched noise (0–3)	31/50	6 / 10 (0.45)	−5.21 [−16.01, 4.15]	3.83 [3.34, 4.39]
iCEM	Matched noise (8–11)	23/50	9 / 5 (0.42)	+14.13 [0.85, 26.20]	2.30 [1.93, 2.78]
MPPI	Local operator (0–3)	35/50	11 / 1 (0.006)	+29.68 [13.98, 48.36]	8.59 [7.84, 9.32]
MPPI	Local operator (8–11)	46/50	0 / 1 (1.00)	+1.50 [0.61, 2.60]	1.05 [0.92, 1.20]
MPPI	Matched noise (0–3)	47/50	0 / 2 (0.50)	+1.66 [−0.11, 3.57]	3.68 [3.02, 4.33]
MPPI	Matched noise (8–11)	45/50	1 / 1 (1.00)	+0.65 [0.02, 1.45]	1.06 [0.85, 1.28]
CEM, $K = 15$	Local operator (0–3)	44/50	2 / 0 (0.50)	+4.07 [1.90, 6.78]	7.51 [7.07, 8.01]
CEM, $K = 15$	Local operator (8–11)	46/50	0 / 0 (1.00)	+1.78 [0.12, 3.80]	2.61 [2.29, 3.03]
CEM, $K = 60$	Local operator (0–3)	44/50	2 / 0 (0.50)	+3.19 [0.50, 6.55]	5.22 [4.89, 5.58]
CEM, $K = 60$	Local operator (8–11)	46/50	0 / 0 (1.00)	+0.64 [0.09, 1.20]	2.22 [1.93, 2.57]
CEM, LeWM	Local operator (0–3)	43/50	2 / 3 (1.00)	+5.16 [−1.44, 14.89]	6.53 [6.15, 6.92]
CEM, LeWM	Local operator (8–11)	44/50	1 / 3 (0.63)	+1.23 [−0.43, 3.48]	2.45 [2.27, 2.66]

Table 10: Two-Room, $N = 300$. First set: 50 sources drawn by the official evaluator (seed 42). Second set: 50 further sources from the official valid starts in a seed-43 permutation, excluding every episode of the first set, fixed before any arm ran. Pooled: both sets, bootstrap over sources. Time is complete planning time per decision; the official solver is timed per batched call over all environments and is not comparable.

Planner	Success	McNemar (p)	Δcost	Median Δcost	Drift	Time (s)
First set						
Native CEM, $R = 30$	47/50	–	–	–	–	0.319
Native CEM, $R = 28$	47/50	0 / 0 (1.00)	+0.14 [−0.23, 0.46]	+0.03	–	0.290
Native CEM, $R = 15$	47/50	0 / 0 (1.00)	+7.99 [5.11, 11.37]	+3.14	–	0.167
Local operator, rounds 0–7	43/50	4 / 0 (0.13)	+18.39 [5.19, 34.55]	−0.14	5.52 [5.27, 5.84]	0.243
Local operator, rounds 16–23	47/50	0 / 0 (1.00)	+1.14 [0.01, 2.66]	−0.03	3.51 [3.26, 3.79]	0.243
Official solver	48/50	0 / 1 (1.00)	–	–	–	–
Second set						
Native CEM, $R = 30$	47/50	–	–	–	–	0.313
Local operator, rounds 0–7	45/50	3 / 1 (0.63)	+15.07 [−1.35, 35.55]	−0.19	5.71 [5.44, 6.01]	0.244
Local operator, rounds 16–23	46/50	1 / 0 (1.00)	+2.58 [−0.80, 8.19]	−0.03	3.81 [3.44, 4.25]	0.244
Matched noise, rounds 0–7	47/50	0 / 0 (1.00)	−3.07 [−8.97, 0.31]	−0.07	3.34 [3.15, 3.56]	0.315
Matched noise, rounds 16–23	46/50	1 / 0 (1.00)	+2.41 [−0.74, 7.56]	+0.08	3.36 [3.09, 3.67]	0.323
Pooled over both sets						
Native CEM, $R = 30$	94/100	–	–	–	–	–
Local operator, rounds 0–7	88/100	7 / 1 (0.07)	+16.73 [6.17, 29.38]	−0.17	–	–
Local operator, rounds 16–23	93/100	1 / 0 (1.00)	+1.86 [−0.05, 4.73]	−0.03	–	–

Every schedule from E10 to E13 matches native success within one decision and native final cost within the 95% interval; the time saving grows with the number of local rounds, and E13, with a single local round, saves no time on this GPU.

Noise strong enough to disturb elite selection about as much as the local operator also raises the final cost more when it is applied early than late, so part of the early sensitivity is common to any perturbation of early selection. It moves the proposal less than the operator does in the same rounds (drift 3.24–3.62 against 5.10–5.12 for the operator in Table 3) and does not change success.

For the matched-compute comparison of Table 2, the source-clustered 95% intervals of the paired success difference from native CEM, in percentage points, are: E12 [0.0, 0.0]; E8L4 [−3.5, 1.5]; $N = 522$ [−3.5, 1.5]; $N = 442$ [−3.0, 2.0]; $R = 13$ [−1.0, 2.5]; iCEM [−39.0, −20.0].

Table 11: Matched-noise control on Reacher: 50 further sources, search seed 42, $N = 300$, $R = 30$, rounds 24–29 exact. Noise has standard deviation 0.25 times the within-round standard deviation of the exact costs. Median Δcost is over sources. Native CEM solves 41/50.

Perturbation	Success	McNemar (p)	Δcost	Median Δcost	First-block dev.	Drift
Local operator, rounds 0–7	44/50	3 / 6 (0.51)	+28.89 [6.69, 51.19]	+0.005	2.48	4.79 [4.57, 5.06]
Local operator, rounds 16–23	41/50	3 / 3 (1.00)	−2.68 [−18.83, 13.49]	−0.005	0.92	4.36 [3.50, 5.36]
Matched noise, rounds 0–7	42/50	5 / 6 (1.00)	−2.71 [−16.60, 10.09]	−0.012	1.90	3.57 [3.26, 3.91]
Matched noise, rounds 16–23	44/50	3 / 6 (0.51)	+6.67 [−10.23, 24.34]	+0.004	0.81	3.90 [3.09, 4.87]

Table 12: Sensitivity of E2L-CEM to the first local round R_E on 50 further PushT sources (search seed 51, $N = 600$, $R = 15$, $M = 24$). Ek keeps rounds $0, \ldots, k-1$ exact, makes rounds $k, \ldots, 13$ local, and keeps round 14 exact. Time: complete planning time per decision on one RTX 6000D GPU.

Planner	Success	McNemar (p)	Δcost	Exact queries	Time (s)
Native CEM, $R = 15$	47/50	–	–	9,001	0.155
E10	47/50	0 / 0 (1.00)	+0.26 [−0.51, 0.88]	6,625	0.130
E11	46/50	1 / 0 (1.00)	−0.10 [−1.19, 0.62]	7,225	0.140
E12	46/50	1 / 0 (1.00)	+0.09 [−0.38, 0.43]	7,825	0.147
E13	47/50	0 / 0 (1.00)	+0.20 [−0.07, 0.57]	8,425	0.157

K Operator fidelity over the search, by planner and task

Figure 4 extends Table 6 to Reacher and to the two other planners. In every setting the operator is fitted at every round from the first sentinels of the actual native population (candidate zero is the proposal mean) and evaluated on the remaining candidates; the native search is not altered. PushT CEM uses 100 sources (200 decisions, $N = 600$, 24 sentinels); Reacher CEM uses 50 sources (100 decisions, $N = 300$, $M = 32$); iCEM and MPPI use 50 PushT sources each (100 decisions, $N = 600$, 24 sentinels), with the configurations of Appendix C. The centred relative error measures the prediction error relative to the spread of the held-out responses around their mean rather than relative to their norm, so it is not reduced by a large common offset. The mean-update distance is the L2 distance between the proposal means obtained from the same population with predicted and with exact scores. Chance top-30 overlap is $30/N$: 0.05 for PushT and 0.10 for Reacher.

Across all four settings the induced mean-update distance decreases over the search: from 2.23 to 0.82 for PushT CEM, 1.81 to 0.43 for Reacher CEM, 9.47 to 1.43 for iCEM, and 4.21 to 0.52 for MPPI. The held-out relative error falls with the proposal scale for CEM on both tasks and for MPPI. For iCEM it stays between 0.54 and 0.80 over iterations 0–13 and rises to 1.13 at the last iteration, where its cost Spearman correlation also drops (from 0.34 to 0.29). Elite overlap stays above chance but far from one in every setting, so the operator is used for its effect on the update rather than for reproducing the exact elite set. On Reacher the held-out cost correlation rises from 0.09 at round 0 to 0.41 at round 29, and the top-30 overlap from 0.21 to 0.28.

Figure 5 relates the operator error directly to the proposal scale, pooling decisions and rounds. The pooled Spearman correlation between error and scale is 0.36 for PushT CEM, 0.83 for Reacher CEM, and 0.69 for MPPI, but -0.20 for iCEM. The iCEM proposal is warm-started across decisions, uses colored noise, and reuses elites, so its per-dimension scale need not track the spread of the evaluated candidates; its update error nevertheless decreases over the iterations (Figure 4).

L Distribution of paired final-cost differences in stage-wise substitution

The stage-wise tables report means of paired final-cost differences. Because the exact final cost is heavy-tailed, Figure 6 shows every paired decision and Table 21 summarizes each distribution. Pairs share the source, the decision index, and the search seed; the difference is the arm's exact final cost minus that of native CEM. For iCEM and MPPI the reference is each planner's own native run.

Table 13: Stronger score noise on 50 further PushT sources (search seed 51, $N = 600$, $R = 15$, rounds 12–14 exact). Noise has standard deviation k times the within-round standard deviation of the exact costs. Elite overlap: mean top-30 overlap between noisy and exact scores over the perturbed rounds; the local operator's overlap is about 0.14 (Table 6). Native CEM solves 44/50.

Perturbation	Success	McNemar (p)	Δcost	Drift	Elite overlap
Noise $k = 1$, rounds 0–3	44/50	1 / 1 (1.00)	+5.64 [0.36, 14.96]	3.24 [3.09, 3.39]	0.269
Noise $k = 1$, rounds 8–11	43/50	1 / 0 (1.00)	+1.19 [0.49, 1.90]	2.80 [2.63, 3.00]	0.163
Noise $k = 2$, rounds 0–3	43/50	2 / 1 (1.00)	+7.99 [0.87, 19.12]	3.62 [3.46, 3.80]	0.169
Noise $k = 2$, rounds 8–11	42/50	2 / 0 (0.50)	+3.20 [1.10, 6.61]	2.90 [2.69, 3.13]	0.108

Table 14: Prospective test of the rounds-curve diagnostic. Before running, each setting was labelled by whether early substitution had raised the final cost significantly in its stage-wise experiment (for Two-Room, over the pooled 100 sources), and the diagnostic predicted that stopping the native planner at about half its rounds would (or would not) raise the final cost significantly on the same sources. PushT settings stop after 7 of 15 iterations; Two-Room stops after 8 (CEM) or 15 (MPPI) of 30 rounds on its second source set; for Two-Room MPPI, which had no earlier stage-wise experiment, the prediction was registered together with its own stage-wise arms, predicting that early substitution raises the cost more than late substitution (it did: Table 15). Reacher MPPI stops after 15 of 30 iterations; its prediction (no significant cost increase, and no significant success loss from early substitution) was registered with its own stage-wise arms; on 80 of its 100 decisions the shortened search ends at a higher cost, although the mean difference is not significant. The full-length native runs were rerun and matched the stored runs exactly. Significant: 95% bootstrap interval of the paired difference excludes zero.

Setting	Early harm	Predicted	Success (full $\to$ short)	Δcost (short $-$ full)	Held
PushT, MPPI	yes	significant	$45 \to 41$	+13.01 [8.19, 18.87]	yes
PushT, iCEM	yes	significant	$27 \to 16$	+33.81 [18.42, 48.53]	yes
PushT, CEM $K = 15$	yes	significant	$46 \to 44$	+5.04 [2.85, 7.56]	yes
PushT, CEM $K = 60$	yes	significant	$46 \to 45$	+7.00 [4.74, 9.45]	yes
PushT, CEM, LeWM	no	not significant	$42 \to 43$	+2.74 [-2.04, 6.96]	yes
Two-Room, CEM	yes	significant	$47 \to 46$	+38.44 [27.21, 50.82]	yes
Reacher, MPPI	–	not significant	$40 \to 44$	-3.78 [-20.71, 12.28]	yes
Two-Room, MPPI	–	significant	$48 \to 48$	+0.46 [0.28, 0.69]	yes

Differences are per decision; for Reacher their mean equals the mean of the per-source differences reported in the stage-wise results.

On PushT the early arm has a higher final cost than native on 64% of decisions and the late arm on 68%, so the sign of the difference alone does not separate the stages; the separation is in its size. The early arm has the larger median (0.51 against 0.28) and upper quartile (2.96 against 1.39), and after removing its five largest differences its mean remains 2.64, whereas that of the late arm drops to -0.11. For iCEM and MPPI the early arm is higher on 80% and 74% of decisions, with medians of 23.72 and 4.13 (late: 2.01 and 0.20), so these effects are not carried by a few decisions. On Two-Room the median difference is close to zero for both arms (-0.06 and -0.00); the early mean of 18.39 comes from a tail in which the five largest differences contribute 66% of the positive sum, and the early arm is also the one that loses successes on this task (Table 10). On Reacher both arms have medians within ± 0.02 of native and large tails of both signs (early: -346.15 to 366.84; late: -381.25 to 328.38); the means of 7.34 and -11.32 are determined by a small number of decisions.

M ADAPTIVE STOPPING: ROUNDS USED AND ALL ARMS

Appendix F describes the contraction-triggered stopping rule. Figure 7 shows how many rounds each decision used, and Table 22 lists every stopping arm against both references on the same sources: the full search and native CEM stopped after 13 rounds. PushT uses 100 sources with one search seed (minimum 6, maximum 15 rounds); Reacher uses 50 sources with two search seeds (minimum 8, maximum 30 rounds), and its intervals resample sources as clusters.

Table 15: MPPI on Two-Room, second source set (50 sources), $N = 300$, 30 iterations, with the MPPI configuration of the PushT experiments. Local arms fit the operator once per stage from 24 sentinels; iterations 24–29 are exact. Native MPPI solves 48/50.

Arm	Success	McNemar (p)	Δcost	Median Δcost	Drift	Time (s)
Native MPPI, 15 iterations	48/50	0 / 0 (1.00)	+0.46 [0.28, 0.69]	+0.23	–	0.171
Local operator, iterations 0–7	39/50	9 / 0 (0.004)	+81.14 [49.41, 115.77]	+0.11	7.26 [6.88, 7.61]	0.240
Local operator, iterations 16–23	49/50	0 / 1 (1.00)	−2.60 [−7.77, −0.01]	−0.01	1.43 [1.17, 1.74]	0.243

Table 16: MPPI on Reacher, 50 further sources, $N = 300$, 30 iterations, endpoint cost; local arms fit the operator once per stage from 32 sentinels, iterations 24–29 exact. Native MPPI solves 40/50.

Arm	Success	McNemar (p)	Δcost	Median Δcost
Native MPPI, 15 iterations	44/50	1 / 5 (0.22)	−3.78 [−20.71, 12.28]	+0.070
Local operator, iterations 0–7	42/50	3 / 5 (0.73)	+22.70 [−0.84, 47.47]	+0.005
Local operator, iterations 16–23	42/50	1 / 3 (0.63)	−0.26 [−14.73, 14.40]	−0.001

On PushT the looser threshold ($\tau = 0.631$) runs the full 15 rounds on 60 of 200 decisions, and the tighter one ($\tau = 0.775$) on 21; the other decisions stop between 8 and 14 rounds, and between 6 and 14 rounds, respectively. On Reacher the rule stops most decisions far earlier than the 30-round search: with $\tau = 0.790$, 146 of 200 decisions stop within 13 rounds and 4 run all 30; with $\tau = 0.931$, 66 stop at the minimum of eight rounds. The stopping points spread widely on both tasks, yet the resulting success and cost lie on the fixed-round curve (Figure 3): choosing the stopping point per decision from the contraction of the proposal mean did not add value beyond the average number of rounds it implies.

N MATCHED COMPARISONS BY SEARCH SEED

Table 2 and the Reacher comparison of Section 6.3 pool two search seeds on the same sources. Tables 23 and 24 give each seed separately, with every arm. Within a seed, intervals resample sources.

On PushT, E12 solves 89 and 89 of 100 against 89 and 89 for native CEM, with 0 discordant sources over both seeds, and its cost-difference interval contains zero in each seed. The 13-round search takes 0.1379 and 0.1376 s per decision against 0.1566 and 0.1471 s for E12. On Reacher, E12 solves 44 and 44 of 50 against 42 and 42; its exact final cost is above native on 80 of 100 decisions with seed 42 and on 66 of 100 with seed 43, while success does not decrease.

Table 25 gives the Reacher rounds curve for each seed. Across R from 5 to 30, success ranges from 38 to 45 of 50 with seed 42 and from 38 to 44 of 50 with seed 43, so the flat pooled curve is not an average of opposite per-seed trends.

O PLANNING TIME AND MEMORY IN THE PUSHT MATCHED COMPARISON

Figure 8 shows the distribution over episodes of the mean complete planning time per decision for every arm of the PushT matched comparison, and Table 26 lists its quantiles together with the peak GPU memory allocated during planning. All arms of a seed ran consecutively on the same GPU; each episode contributes the mean over its two decisions.

Episode-to-episode variation is small relative to the differences between arms: for native CEM the 5th–95th percentile range is 157.7–159.9 ms with seed 51 and 155.2–156.9 ms with seed 52. The reduced-population arms do not reduce planning time, consistent with parallel evaluation of the population, and they reduce peak memory only from 132.9 to 129.0 and 125.0 MiB. The local arms allocate 137.8 (E12) and 136.9 MiB (E8L4) at peak, slightly more than native CEM, for the sentinel responses and the fitted operator. The mean time of E12 differs between seeds (156.6 and 147.1 ms) by more than its within-seed spread, whereas native CEM differs by 2.5 ms; speed-ups should therefore be read within a seed and GPU session.

Table 17: Native CEM stopped after R rounds (Figure 3): PushT, 100 sources, full search $R = 15$; Reacher, 50 sources $\times$ two search seeds, full search $R = 30$. Δsuccess: paired difference from the full search in percentage points (95% interval). Intervals for Reacher resample sources as clusters.

Task	R	Success	McNemar (p)	Δsuccess	Δcost	Median Δcost	Time (s)
PushT	3	78/100	11 / 0 ($< 10^{-3}$)	[−17.0, −5.0]	+29.48 [22.96, 36.72]	–	0.045
PushT	5	84/100	5 / 0 (0.063)	[−10.0, −1.0]	+16.47 [11.45, 22.26]	–	0.063
PushT	7	86/100	3 / 0 (0.25)	[−7.0, 0.0]	+12.01 [7.73, 17.19]	–	0.087
PushT	9	86/100	3 / 0 (0.25)	[−7.0, 0.0]	+5.11 [2.32, 8.03]	–	0.104
PushT	11	87/100	2 / 0 (0.50)	[−5.0, 0.0]	+3.25 [2.03, 4.77]	–	0.123
PushT	13	88/100	1 / 0 (1.00)	[−3.0, 0.0]	+0.07 [−1.71, 1.12]	–	0.136
PushT	15	89/100	–	–	–	–	0.157
Reacher	5	84/100	9 / 9 (1.00)	[−9.0, 9.0]	+5.91 [−7.91, 20.10]	+0.52	0.052
Reacher	8	82/100	11 / 9 (0.82)	[−12.0, 7.0]	+4.79 [−6.94, 16.70]	+0.45	0.067
Reacher	10	80/100	9 / 5 (0.42)	[−12.0, 4.0]	+8.07 [−5.01, 21.78]	+0.36	0.077
Reacher	13	85/100	7 / 8 (1.00)	[−7.0, 8.0]	+2.66 [−10.89, 16.23]	+0.26	0.098
Reacher	16	77/100	13 / 6 (0.17)	[−15.0, 1.0]	+1.75 [−10.13, 13.55]	+0.17	0.114
Reacher	20	82/100	7 / 5 (0.77)	[−8.0, 4.0]	+7.99 [−1.51, 18.24]	+0.12	0.136
Reacher	25	84/100	6 / 6 (1.00)	[−8.0, 8.0]	+0.92 [−12.32, 13.82]	+0.03	0.166
Reacher	30	84/100	–	–	–	–	0.196

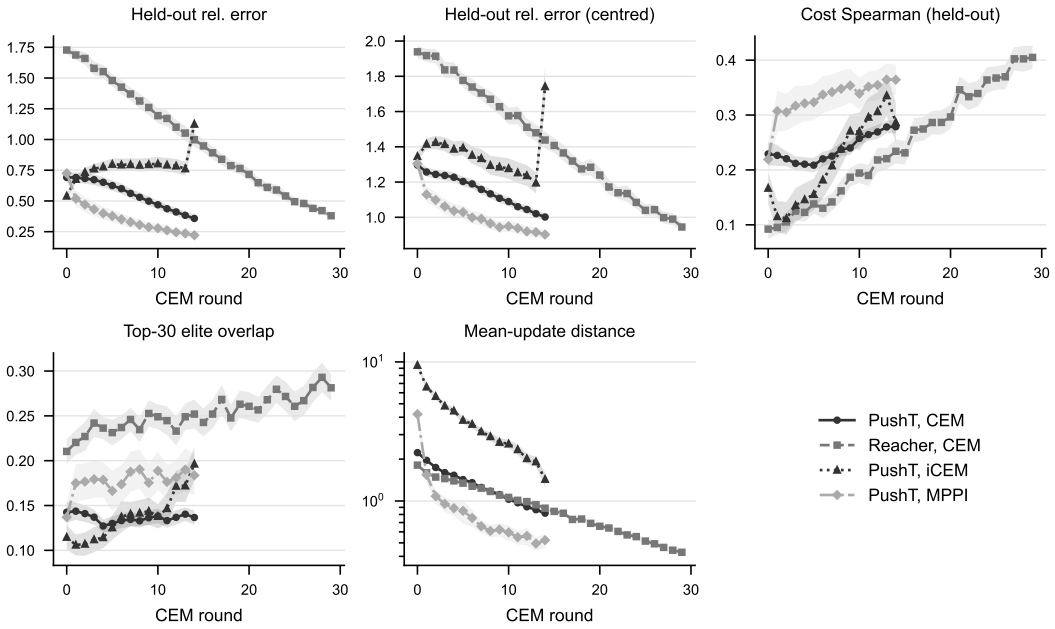


Figure 4: Per-round accuracy of the run-time local operator on the actual native populations of four planner–task pairs. Lines: mean over decisions; bands: ± 1.96 standard errors over decisions. The mean-update distance uses a logarithmic axis.

P Distribution of final-cost differences along the rounds curve

Figure 3 reports mean paired differences when native CEM is stopped after R rounds. Figure 9 shows every paired decision and Table 27 the quantiles. On PushT the pairs are the 200 decisions of the 100 rounds-curve sources; on Reacher, the 200 decisions of 50 sources under two search seeds, each paired with the full search of the same seed.

Truncation raises the final cost on most decisions at every R, and the fraction and size of the increase shrink as R approaches the full search: on PushT the median difference is 11.72 at three rounds, 1.02 at 11, and 0.38 at 13, and 78% of decisions still end higher at 13 rounds. The near-zero mean at 13 rounds (0.07) therefore reflects small increases on most decisions balanced by a few decreases, rather than identical outcomes. On Reacher the median difference falls from 0.60 at five rounds

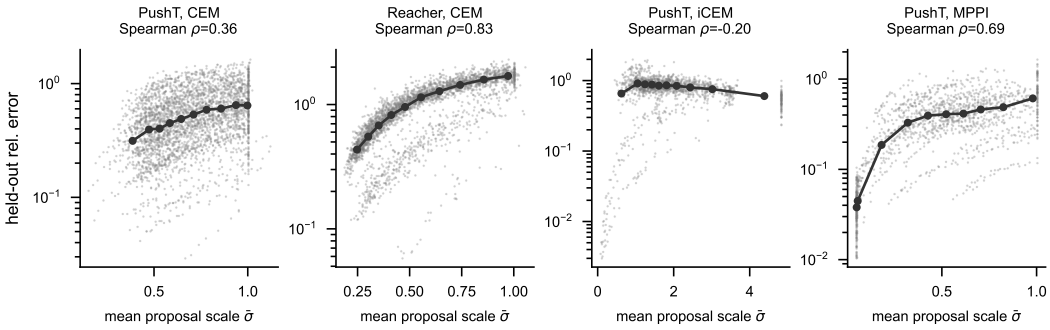


Figure 5: Held-out relative error of the local operator against the mean proposal scale at the round where it is fitted; one point per decision and round. Line: median error within deciles of the proposal scale. ρ: pooled Spearman correlation over all points.

Table 18: Accuracy of the run-time local operator at each round of native CEM on 50 Reacher sources (100 decisions, $N = 300$, $M = 32$ sentinels). Columns as in Table 6; Rel. err. (c) is the centred relative error and Exact step is the L2 step of the exact proposal-mean update, for scale.

Round	$\bar{\sigma}$	Rel. err.	Rel. err. (c)	ρ	Overlap	Mean shift	Exact step
0	1.000	1.728	1.939	0.092	0.210	1.811	1.485
1	0.968	1.687	1.918	0.095	0.220	1.593	1.288
2	0.929	1.660	1.914	0.105	0.227	1.486	1.208
3	0.890	1.579	1.836	0.125	0.242	1.449	1.169
4	0.851	1.553	1.836	0.123	0.236	1.395	1.129
5	0.815	1.480	1.777	0.138	0.231	1.343	1.072
6	0.777	1.426	1.739	0.130	0.237	1.277	1.011
7	0.743	1.370	1.705	0.142	0.246	1.236	0.983
8	0.707	1.313	1.670	0.162	0.235	1.171	0.928
9	0.674	1.261	1.627	0.187	0.253	1.102	0.906
10	0.641	1.194	1.576	0.194	0.249	1.061	0.851
11	0.610	1.171	1.577	0.190	0.245	1.011	0.815
12	0.581	1.100	1.511	0.218	0.233	0.990	0.796
13	0.553	1.052	1.480	0.221	0.249	0.940	0.749
14	0.528	0.998	1.439	0.234	0.252	0.887	0.710
15	0.501	0.948	1.408	0.231	0.243	0.842	0.688
16	0.478	0.892	1.365	0.272	0.252	0.816	0.658
17	0.454	0.839	1.319	0.275	0.268	0.737	0.605
18	0.432	0.787	1.275	0.286	0.248	0.742	0.596
19	0.410	0.764	1.284	0.287	0.263	0.691	0.574
20	0.390	0.716	1.240	0.297	0.261	0.658	0.537
21	0.370	0.648	1.172	0.346	0.257	0.640	0.522
22	0.352	0.610	1.139	0.333	0.268	0.605	0.493
23	0.335	0.589	1.135	0.339	0.280	0.570	0.454
24	0.319	0.541	1.084	0.364	0.272	0.554	0.446
25	0.303	0.494	1.040	0.367	0.261	0.515	0.420
26	0.288	0.479	1.043	0.370	0.267	0.492	0.394
27	0.273	0.440	0.998	0.402	0.282	0.464	0.388
28	0.260	0.421	0.990	0.402	0.293	0.443	0.370
29	0.247	0.379	0.945	0.405	0.281	0.428	0.345

to 0.02 at 25, while success is flat (Table 17): the additional rounds refine the model cost without changing the outcome.

Table 19: Accuracy of the run-time local operator at each iteration of iCEM and MPPI on 50 PushT sources each (100 decisions, $N = 600$, 24 sentinels). Columns as in Table 18. Native success: iCEM 27/50, MPPI 45/50.

	iCEM						MPPI					
Iter.	$\bar{\sigma}$	Rel. err.	ρ	Overlap	Mean shift	Exact step	$\bar{\sigma}$	Rel. err.	ρ	Overlap	Mean shift	Exact step
0	4.817	0.540	0.167	0.115	9.466	8.818	1.000	0.725	0.219	0.137	4.211	3.625
1	3.395	0.676	0.115	0.106	6.613	4.493	0.645	0.519	0.307	0.175	1.541	1.233
2	2.814	0.734	0.112	0.107	5.656	3.558	0.580	0.471	0.305	0.177	1.083	0.825
3	2.444	0.762	0.136	0.112	4.826	2.979	0.538	0.432	0.317	0.179	0.953	0.675
4	2.164	0.778	0.146	0.114	4.449	2.571	0.504	0.399	0.321	0.179	0.888	0.566
5	1.960	0.799	0.156	0.125	3.813	2.371	0.474	0.378	0.323	0.166	0.850	0.552
6	1.787	0.794	0.182	0.137	3.566	2.123	0.442	0.350	0.337	0.174	0.755	0.484
7	1.638	0.797	0.208	0.141	3.153	1.964	0.418	0.329	0.342	0.188	0.658	0.416
8	1.511	0.792	0.238	0.142	2.909	1.786	0.396	0.305	0.348	0.190	0.609	0.384
9	1.396	0.797	0.271	0.144	2.646	1.720	0.377	0.287	0.354	0.175	0.623	0.369
10	1.294	0.803	0.270	0.138	2.577	1.582	0.361	0.278	0.339	0.189	0.594	0.348
11	1.198	0.793	0.296	0.147	2.345	1.485	0.344	0.261	0.352	0.176	0.549	0.339
12	1.111	0.785	0.306	0.172	2.021	1.299	0.328	0.245	0.355	0.181	0.560	0.327
13	1.032	0.762	0.335	0.172	1.918	1.239	0.313	0.236	0.365	0.190	0.494	0.312
14	0.964	1.125	0.287	0.196	1.434	1.139	0.299	0.221	0.364	0.184	0.523	0.306

Table 20: Additional per-round statistics for PushT native CEM (the traces of Table 6; 200 decisions). Rel. err. (c): centred held-out relative error. ρ_{all}: Spearman correlation of predicted and exact costs over all 600 candidates, including the 24 sentinels. Best exact cost: minimum exact cost in the round's population, mean and median over decisions (95% interval of the mean).

Round	Rel. err. (c)	ρ_{all}	Best exact cost, mean	Best exact cost, median
0	1.297	0.253	53.96 [44.46, 64.27]	18.75
1	1.257	0.254	31.09 [24.17, 38.31]	11.23
2	1.244	0.251	22.33 [16.93, 28.55]	8.07
3	1.237	0.244	17.78 [12.80, 23.85]	7.27
4	1.227	0.243	15.64 [10.87, 21.84]	5.90
5	1.204	0.242	14.01 [9.40, 19.81]	5.39
6	1.190	0.253	13.29 [8.61, 19.10]	5.06
7	1.158	0.258	12.57 [7.98, 18.26]	4.58
8	1.133	0.270	12.06 [7.61, 17.76]	4.23
9	1.108	0.272	11.66 [7.10, 17.54]	3.98
10	1.089	0.288	11.30 [6.81, 16.81]	3.64
11	1.060	0.294	11.06 [6.45, 16.67]	3.69
12	1.044	0.300	10.74 [6.28, 16.36]	3.52
13	1.020	0.308	10.50 [6.11, 16.01]	3.36
14	1.002	0.309	10.25 [5.87, 15.77]	3.00

Q Decision-level statistics of the Reacher stage-wise experiments

Table 28 reports, for both Reacher stage-wise sets and every arm, decision-level statistics that complement the success counts: the per-source median of the paired final-cost difference, the number of decisions on which the arm ends below or above native CEM with a two-sided sign test, the deviation of the first executed action block of the first decision from native CEM, and the drift of the proposal mean at the end of the substituted stage.

On both sets the first executed action block moves further from native CEM under early substitution (2.43 and 2.46) than under late substitution (0.93 and 0.91), so early rounds also shape the executed action on Reacher, even though the change raises rather than lowers success there. The per-source median cost differences are close to zero for every arm and the sign tests do not separate the arms at the 0.05 level, in line with the heavy two-sided tails of Figure 6.

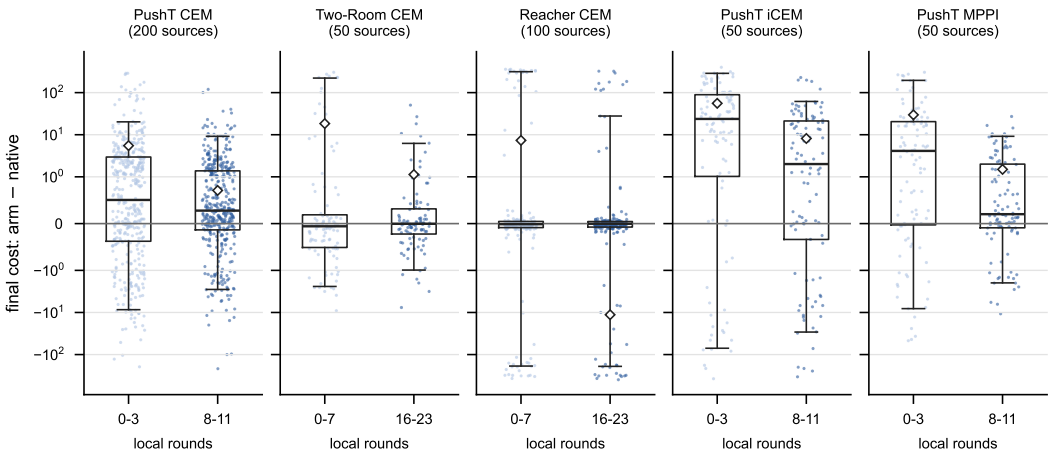


Figure 6: Paired final-cost differences (arm minus native) for early and late local substitution, one point per decision, on a symmetric-log axis (linear within ± 1). Boxes: quartiles; whiskers: 5th and 95th percentiles; open diamonds: means. Stage labels give the rounds (CEM) or iterations (iCEM, MPPI) evaluated with the local operator.

Table 21: Summary of the distributions in Figure 6. Higher / lower: fraction of decisions on which the arm's final cost is above / below native. Trimmed mean: mean after removing the five largest differences. Top-5 share: share of the summed positive differences contributed by the five largest.

Setting	Local rounds	Decisions	Mean	Median	Quartiles	Higher	Lower	Trimmed mean	Top-5 share
PushT CEM (200 sources)	0–3	400	5.50	0.51	[−0.38, 2.96]	0.64	0.36	2.64	0.37
	8–11	400	0.71	0.28	[−0.13, 1.39]	0.68	0.32	−0.11	0.37
Two-Room CEM (50 sources)	0–7	100	18.39	−0.06	[−0.51, 0.19]	0.40	0.60	6.23	0.66
	16–23	100	1.14	−0.00	[−0.22, 0.32]	0.49	0.51	0.01	0.80
Reacher CEM (100 sources)	0–7	200	7.34	−0.01	[−0.09, 0.05]	0.43	0.57	−1.60	0.29
	16–23	200	−11.32	−0.00	[−0.07, 0.04]	0.47	0.53	−18.34	0.63
PushT iCEM (50 sources)	0–3	100	55.70	23.72	[1.02, 89.24]	0.80	0.20	41.98	0.23
	8–11	100	8.14	2.01	[−0.34, 21.22]	0.69	0.29	−1.26	0.47
PushT MPPI (50 sources)	0–3	100	29.68	4.13	[−0.03, 20.47]	0.74	0.26	17.10	0.42
	8–11	100	1.50	0.20	[−0.09, 2.00]	0.68	0.32	0.74	0.41

R INVENTORY OF EXPERIMENTS

Table 29 lists every experiment reported in the paper. Sources are drawn by the outcome-blind builder of Appendix C and are disjoint across experiments, except where an experiment reuses the sources of another, as stated in the table. All results are development evaluations on fresh sources; no source was used to choose a schedule or threshold that is evaluated on it.

The results of each experiment appear in: PushT operator accuracy by round, Fig. 2, Tab. 6, Fig. 4; PushT stage-wise substitution, first set, Tab. 3; PushT stage-wise substitution, replication, Tab. 3; PushT stage-wise substitution, larger set, Tab. 3, Fig. 6; PushT matched-compute comparison, Tab. 2, Tab. 23; PushT front-loaded populations, Tab. 5; PushT rounds curve, Fig. 3, Tab. 17; PushT adaptive stopping, App. F, Tab. 22; Two-Room stage-wise and rounds, Tab. 10, Fig. 6; Reacher operator accuracy by round, Fig. 2, Tab. 18; Reacher stage-wise substitution, first set, Tab. 3; Reacher stage-wise substitution, larger set, Tab. 3, Fig. 6; Reacher matched-compute comparison, Sec. 6.3, Tab. 24; Reacher rounds curve, Fig. 3, Tab. 17; Reacher adaptive stopping, App. F, Tab. 22; PushT other planners, Sec. 6.2, Tab. 9, Tab. 19. PushT and Two-Room experiments ran on one RTX 6000D GPU, and Reacher experiments and the other-planner experiment on one A100 GPU; timings are compared only within an experiment.

S E2L-CEM PROCEDURE

The procedure below restates Section 4 for one planning decision with history h and goal g. Native CEM is the special case $R_L = 0$, with no sentinels; E2L uses $R_L \geq 1$ and $R_F \geq 1$.

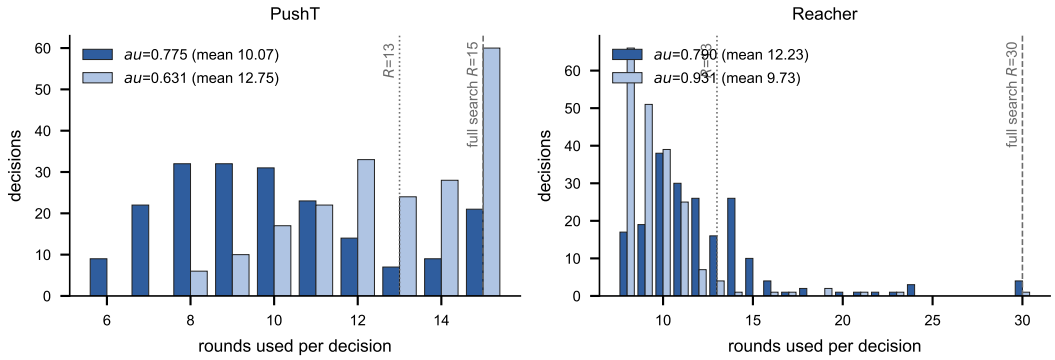


Figure 7: Rounds used per decision by contraction-triggered stopping, for both thresholds fixed in advance (200 decisions per arm). Dashed: full search; dotted: 13 rounds.

Table 22: Contraction-triggered stopping against the full search and against 13 fixed rounds. Success: arm vs. reference. McNemar counts: sources solved only by the reference / only by the stopping arm. Δcost: paired difference in exact final cost (arm minus reference; 95% bootstrap interval). Queries and time are exact queries and complete planning time per decision of the stopping arm.

			vs. full search			vs. 13 rounds				
Task	τ	Rounds (mean)	Success	McNemar (p)	Δcost	Success	McNemar (p)	Δcost	Queries	Time (s)
PushT	0.631	12.75	89 vs 89	0 / 0 (1.00)	0.72 [0.47, 1.00]	89 vs 88	0 / 1 (1.00)	0.65 [−0.38, 2.40]	7651	0.140
	0.775	10.07	87 vs 89	2 / 0 (0.50)	2.89 [1.86, 4.28]	87 vs 88	2 / 1 (1.00)	2.82 [1.22, 5.03]	6046	0.113
Reacher	0.790	12.23	82 vs 84	9 / 7 (0.80)	4.86 [−7.75, 17.06]	82 vs 85	7 / 4 (0.55)	2.20 [−11.00, 15.72]	3670	0.091
	0.931	9.73	80 vs 84	13 / 9 (0.52)	−2.75 [−16.02, 10.70]	80 vs 85	13 / 8 (0.38)	−5.41 [−19.16, 8.41]	2922	0.088

Input: world model F_θ, cost $\mathcal{J}$, history h, goal g, population N, elites K, schedule (R_E, R_L, R_F, M) with $R = R_E + R_L + R_F$, ridge parameter λ, initial proposal (μ_0, σ_0).
Output: proposal mean μ_R; its leading actions are executed.

1. For $r = 0, \dots, R-1$:
   (a) Sample the population $a_{r,i} \sim \mathcal{N}(\mu_r, \operatorname{diag}\sigma_r^2)$, $i = 0, \dots, N-1$, with the sampler of native CEM.
   (b) If $r < R_E$ or $r \geq R_E + R_L$ (exact round): $y_{r,i} \leftarrow F_\theta(h, a_{r,i})$ for all i.
   (c) If $r = R_E$ (fitting round): set $a^0 \leftarrow \mu_r$ and $\sigma^\star \leftarrow \sigma_r$; evaluate M sentinels exactly, $y_j \leftarrow F_\theta(h, a_j)$ for $j = 0, \dots, M-1$, where $a_j = a_{r,j}$ are the first M candidates of the population after replacing $a_{r,0}$ by a^0, so the population keeps N candidates; form $\delta_j = (a_j - a^0) \oslash \sigma^\star$ and fit $\widehat{B}$ by the weighted ridge problem of Section 4, in dual form since $M < q$.
   (d) If $R_E \leq r < R_E + R_L$ (local round): $\widehat{y}_{r,i} \leftarrow y^0 + \widehat{B}^\top\big((a_{r,i} - a^0) \oslash \sigma^\star\big)$ for all i; the only world-model queries are the M sentinels at $r = R_E$.
   (e) Score every candidate with $\mathcal{J}(y_{r,i}, g)$ in exact rounds or $\mathcal{J}(\widehat{y}_{r,i}, g)$ in local rounds, and select the K lowest.
   (f) Update $(\mu_{r+1}, \sigma_{r+1})$ from the elites with the update rule of native CEM.
2. Evaluate $F_\theta(h, \mu_R)$ once (recorded for every arm) and return μ_R.

Exact queries per decision: $(R - R_L)N + \mathbf{1}\{R_L > 0\}M + 1$, against $RN + 1$ for native CEM.

T Extended related work

Sampling-based planners with learned models. The cross-entropy method (Rubinstein, 1997; de Boer et al., 2005) is the default optimizer of many latent world-model planners, from PlaNet (Hafner et al., 2018) and ensemble-based PETS (Chua et al., 2018) to recent predictive-feature models such as DINO-WM (Zhou et al., 2025), LeWM (Maes et al., 2026), and Fast-LeWM (Gao & Xu, 2026). Path-integral control (Williams et al., 2015; 2017) replaces elite selection with exponential weighting, and TD-MPC combines MPPI-style planning with a learned value (Hansen et al.,

Table 23: PushT matched-compute comparison by search seed (100 sources per seed; same sources for both seeds). McNemar counts: sources solved only by native CEM / only by the arm. Δcost: paired difference in exact final cost against native CEM of the same seed (95% bootstrap interval). Time: mean complete planning time per decision; median and 95th percentile are over per-episode means.

Seed	Arm	Success	McNemar (p)	Δcost	Queries	Time (s)	Median (s)	P95 (s)
51	Native CEM	89/100	0 / 0 (1.00)	0.00 [0.00, 0.00]	9001	0.1585	0.1584	0.1599
	E12	89/100	0 / 0 (1.00)	−0.48 [−1.36, 0.20]	7825	0.1566	0.1565	0.1576
	E8L4	89/100	3 / 3 (1.00)	0.58 [−0.74, 1.94]	6625	0.1316	0.1313	0.1338
	Native, $N = 522$	87/100	3 / 1 (0.62)	−0.03 [−1.06, 1.13]	7831	0.1639	0.1637	0.1657
	Native, $N = 442$	90/100	2 / 3 (1.00)	−0.11 [−1.22, 0.83]	6631	0.1585	0.1583	0.1605
	Native, 13 rounds	91/100	0 / 2 (0.50)	0.14 [−0.90, 0.91]	7801	0.1379	0.1384	0.1394
	iCEM	63/100	31 / 5 (<0.001)	44.55 [29.71, 59.59]	9006	0.1614	0.1613	0.1627
52	Native CEM	89/100	0 / 0 (1.00)	0.00 [0.00, 0.00]	9001	0.1560	0.1560	0.1569
	E12	89/100	0 / 0 (1.00)	0.85 [−0.33, 2.23]	7825	0.1471	0.1470	0.1481
	E8L4	87/100	2 / 0 (0.50)	1.96 [0.01, 4.46]	6625	0.1302	0.1297	0.1328
	Native, $N = 522$	89/100	1 / 1 (1.00)	0.44 [−0.34, 1.30]	7831	0.1583	0.1579	0.1610
	Native, $N = 442$	87/100	2 / 0 (0.50)	1.43 [−0.05, 3.23]	6631	0.1568	0.1571	0.1581
	Native, 13 rounds	88/100	1 / 0 (1.00)	1.24 [0.39, 2.25]	7801	0.1376	0.1371	0.1399
	iCEM	56/100	37 / 4 (<0.001)	53.35 [37.29, 70.33]	9006	0.1640	0.1639	0.1654

Table 24: Reacher matched-compute comparison by search seed (50 sources per seed; same sources for both seeds; $N = 300$, $R = 30$ for native CEM). Columns as in Table 23. Median Δcost is over sources; lower / higher: decisions on which the arm's exact final cost is below / above native CEM, with a two-sided sign test. The Reacher evaluator times two synchronized batched policy calls over all sources, so only the mean time per decision is available.

Seed	Arm	Success	McNemar (p)	Δcost	Median Δcost	Lower / higher (p)	Queries	Time (s)
42	Native CEM	42/50	0 / 0 (1.00)	0.00 [0.00, 0.00]	0.00	0 / 0 (1.00)	9001	0.1920
	E12	44/50	3 / 5 (0.73)	0.09 [−18.81, 18.78]	0.50	20 / 80 (<0.001)	3933	0.1078
	Native, $N = 131$	39/50	9 / 6 (0.61)	−7.52 [−31.33, 15.97]	0.06	33 / 67 (<0.001)	3931	0.1893
	Native, 13 rounds	48/50	1 / 7 (0.070)	14.73 [−5.61, 35.47]	0.34	19 / 81 (<0.001)	3901	0.0963
	iCEM	36/50	11 / 5 (0.21)	−17.69 [−37.03, 0.22]	0.08	34 / 66 (0.002)	9006	0.2122
43	Native CEM	42/50	0 / 0 (1.00)	0.00 [0.00, 0.00]	0.00	0 / 0 (1.00)	9001	0.1936
	E12	44/50	5 / 7 (0.77)	4.29 [−17.51, 26.30]	0.23	34 / 66 (0.002)	3933	0.1090
	Native, $N = 131$	46/50	4 / 8 (0.39)	1.89 [−17.91, 21.61]	0.03	37 / 63 (0.012)	3931	0.1879
	Native, 13 rounds	44/50	4 / 6 (0.75)	5.35 [−10.42, 22.06]	0.38	23 / 77 (<0.001)	3901	0.0952
	iCEM	41/50	7 / 6 (1.00)	−14.97 [−33.27, 1.87]	0.06	33 / 67 (<0.001)	9006	0.2130

2022; 2024). iCEM adds temporally correlated sampling, elite reuse, and population decay (Pinneri et al., 2021). These planners share the structure our analysis relies on: a sampling distribution that is refitted to scored candidates and contracts as the search proceeds. We tested CEM, iCEM, and MPPI with the same world model; the stage asymmetry appeared in all three.

Reducing the cost of each evaluation. Much work makes each world-model query cheaper. Fast-LeWM predicts action prefixes to shorten rollouts (Gao & Xu, 2026), Sparse Imagination reduces per-rollout visual computation (Chun et al., 2025), LeFlow amortizes planning into a learned trajectory prior (Huang et al., 2026), POPLIN and TAP search in policy-network or latent action spaces (Wang & Ba, 2020; Jiang et al., 2023), and gradient-based planners and CEM-GD change the optimization path (Jyothir et al., 2023; Huang et al., 2021). These approaches change the evaluation or the optimizer; our analysis keeps both fixed and asks which evaluations of an unchanged search carry the decision, so its conclusions apply on top of any of them.

Surrogates in evolutionary search. Surrogate-assisted evolution strategies fit an approximate model to past evaluations and decide per generation whether to trust it (Jin et al., 2002; Loshchilov et al., 2012; Hansen, 2019). Generation-based evolution control alternates exact and surrogate generations at a fixed or adapted rate; rank-based acceptance tests check whether the surrogate reproduces the ranking of a few exactly evaluated points. Our local operator differs in three ways. It is refitted around the current proposal mean from the current decision's own probes rather than accumulated across generations; it approximates the predicted representation that the native cost consumes rather than the scalar objective; and it is placed by search stage. The component analysis of Appendix H

Table 25: Reacher rounds curve by search seed (50 sources per seed; reference: full search, $R = 30$, of the same seed). Columns as in Table 24; the pooled curve is in Table 17.

Seed	R	Success	McNemar (p)	Δcost	Median Δcost	Lower / higher (p)	Queries	Time (s)
42	5	42/50	5 / 7 (0.77)	2.87 [−17.39, 23.40]	0.50	17 / 83 (<0.001)	1501	0.0512
	8	41/50	7 / 8 (1.00)	1.66 [−17.35, 20.63]	0.43	13 / 87 (<0.001)	2401	0.0677
	10	42/50	2 / 4 (0.69)	8.90 [−5.42, 23.67]	0.38	18 / 82 (<0.001)	3001	0.0773
	13	45/50	2 / 7 (0.18)	5.09 [−14.40, 25.52]	0.31	14 / 86 (<0.001)	3901	0.0966
	16	38/50	8 / 6 (0.79)	−0.94 [−16.49, 14.91]	0.17	26 / 74 (<0.001)	4801	0.1140
	20	39/50	4 / 3 (1.00)	9.63 [−6.79, 26.53]	0.13	26 / 74 (<0.001)	6001	0.1366
	25	41/50	3 / 4 (1.00)	−4.10 [−19.87, 10.99]	0.02	42 / 58 (0.13)	7501	0.1630
	30	40/50	0 / 0 (1.00)	0.00 [0.00, 0.00]	0.00	0 / 0 (1.00)	9001	0.1932
43	5	42/50	4 / 2 (0.69)	8.95 [−10.73, 28.37]	0.53	12 / 88 (<0.001)	1501	0.0532
	8	41/50	4 / 1 (0.38)	7.93 [−9.27, 24.46]	0.50	17 / 83 (<0.001)	2401	0.0665
	10	38/50	7 / 1 (0.070)	7.25 [−14.35, 28.74]	0.34	15 / 85 (<0.001)	3001	0.0772
	13	40/50	5 / 1 (0.22)	0.23 [−19.78, 20.12]	0.16	24 / 76 (<0.001)	3901	0.0993
	16	39/50	5 / 0 (0.062)	4.44 [−16.53, 25.08]	0.17	17 / 83 (<0.001)	4801	0.1135
	20	43/50	3 / 2 (1.00)	6.35 [−9.06, 21.84]	0.09	32 / 68 (<0.001)	6001	0.1359
	25	43/50	3 / 2 (1.00)	5.93 [−12.16, 24.04]	0.04	39 / 61 (0.035)	7501	0.1683
	30	44/50	0 / 0 (1.00)	0.00 [0.00, 0.00]	0.00	0 / 0 (1.00)	9001	0.1985

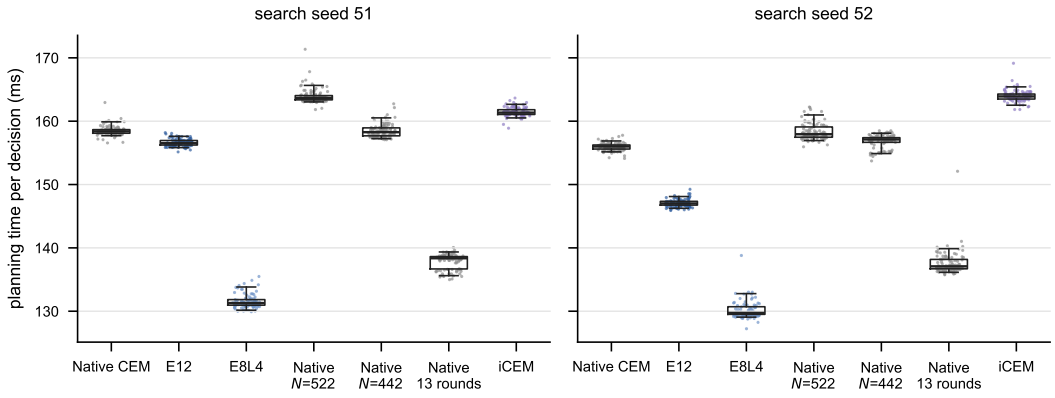


Figure 8: Per-episode mean complete planning time per decision in the PushT matched comparison (100 episodes per arm and seed). Boxes: quartiles; whiskers: 5th and 95th percentiles.

shows that the second choice matters: a surrogate fitted to the scalar cost keeps about a third of the elite overlap of the representation operator at equal query count.

Multi-fidelity optimization and the value of computation. Multi-fidelity Bayesian optimization chooses among information sources of different cost and accuracy (Forrester et al., 2007; Kandasamy et al., 2016; Peherstorfer et al., 2018), including composite objectives (Astudillo & Frazier, 2019), unreliable sources (Mikkola et al., 2023), and input-dependent fidelity (Fan et al., 2024). Value-of-computation analyses ask whether an additional evaluation can change a downstream decision (Sezener & Dayan, 2020; Lakshminarasimhan et al., 2024). Our measurement answers a version of this question empirically for population planning: the late rounds of a contracting search have low value of computation on every task we tested, while the value of early rounds depends on whether the task's decision is still open after the first few rounds.

Test-time compute in world-model and embodied systems. Recent work treats test-time compute as a resource to allocate across prompts (Dao et al., 2026), imagination calls (Yu et al., 2026), or samples of a video world model (Jiang et al., 2026). A common observation is that additional compute yields uneven gains. Our results locate this unevenness inside a single search: in a contracting planner, compute spent after the proposal has settled refines the model cost without changing the executed decision, which is visible in the flat Reacher success curve and in the near-zero final-cost change of the last PushT rounds.

Table 26: Planning time per decision (ms; quantiles over the 100 per-episode means) and peak allocated GPU memory during planning (MiB) in the PushT matched comparison.

Seed	Arm	Mean	Median	P5	P95	Max	Peak memory
51	Native CEM	158.5	158.4	157.7	159.9	162.9	132.9
	E12	156.6	156.5	155.8	157.6	158.2	137.8
	E8L4	131.6	131.3	130.2	133.8	135.5	136.9
	Native, $N = 522$	163.9	163.7	163.0	165.7	171.4	129.0
	Native, $N = 442$	158.5	158.3	157.2	160.5	162.7	125.0
	Native, 13 rounds	137.9	138.4	135.6	139.4	140.1	132.9
	iCEM	161.4	161.3	160.5	162.7	163.6	133.4
52	Native CEM	156.0	156.0	155.2	156.9	157.8	132.9
	E12	147.1	147.0	146.2	148.1	149.2	137.8
	E8L4	130.2	129.7	129.1	132.8	138.8	136.9
	Native, $N = 522$	158.3	157.9	156.9	161.0	162.2	129.0
	Native, $N = 442$	156.8	157.1	154.9	158.1	158.5	125.0
	Native, 13 rounds	137.6	137.1	136.1	139.9	152.1	132.9
	iCEM	164.0	163.9	162.5	165.4	169.1	133.4

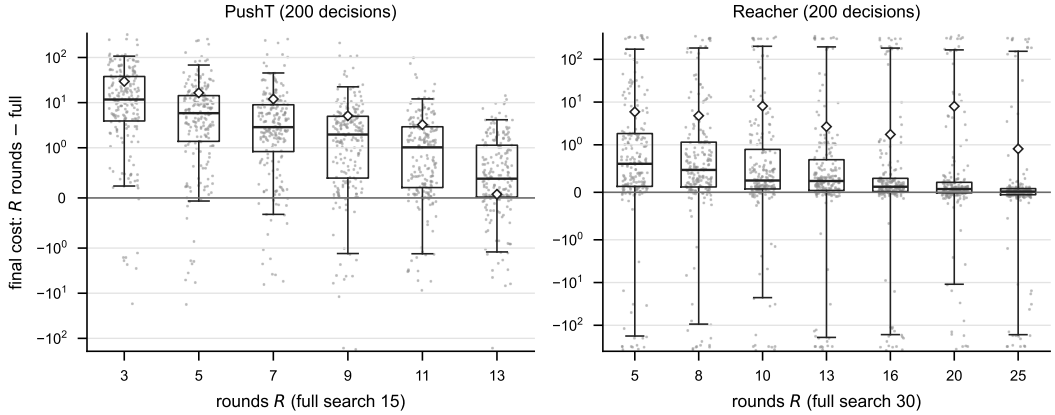


Figure 9: Paired final-cost differences between native CEM stopped after R rounds and the full search, one point per decision, on a symmetric-log axis (linear within ± 1). Boxes: quartiles; whiskers: 5th and 95th percentiles; open diamonds: means.

U Design choices of the local operator

Why fit at run time. The operator is fitted from exact probes of the current decision rather than learned offline. This requires no training data or additional model, keeps the world model and its checkpoint untouched, and adapts the operator to the current history, goal, and proposal. It also makes the operator a measurement instrument: its held-out accuracy on the actual native population at each round (Section 6.1, Appendix K) is a direct probe of how locally linear the world model is around the current proposal.

Why predict the representation. The native cost is a nonlinear function of the predicted representation. Approximating the representation and applying the native cost preserves this nonlinearity, so candidates that differ in predicted outcome are scored by the same rule as in native CEM. Approximating the scalar cost instead flattens this structure into one number per candidate and loses most of the elite overlap (Appendix H).

Why keep the final round exact. The returned proposal is the mean of the final round's elites. Keeping this round exact guarantees that the executed plan is selected by exact world-model scores within the population that the final round samples. It does not guarantee that the population itself matches the all-exact search, which is why we report drift and final cost for every arm.

Table 27: Quantiles of the paired final-cost differences in Figure 9. Higher / lower: fraction of decisions on which the truncated search ends with a higher / lower exact final cost than the full search.

Task	R	Mean	Median	Q1	Q3	Higher	Lower
PushT	3	29.48	11.72	3.92	37.94	0.96	0.04
	5	16.47	5.84	1.39	14.30	0.94	0.06
	7	12.01	2.86	0.92	8.97	0.92	0.09
	9	5.11	1.97	0.40	4.98	0.90	0.10
	11	3.25	1.02	0.21	2.92	0.86	0.14
	13	0.07	0.38	0.02	1.14	0.78	0.23
Reacher	5	5.91	0.60	0.12	1.83	0.85	0.14
	8	4.79	0.47	0.11	1.15	0.85	0.15
	10	8.07	0.25	0.07	0.90	0.83	0.17
	13	2.66	0.24	0.04	0.69	0.81	0.19
	16	1.75	0.12	0.02	0.29	0.79	0.21
	20	7.99	0.07	−0.01	0.21	0.71	0.29
	25	0.92	0.02	−0.05	0.08	0.59	0.41

Table 28: Reacher stage-wise substitution, decision-level statistics. First set: 50 sources; larger set: 100 sources (two blocks). First-block dev.: L2 distance of the first executed action block (5 steps $\times$ 2 dimensions) of the first decision from native CEM. Drift: L2 distance between the arm's and native CEM's proposal means at the last round of the substituted stage (95% bootstrap interval).

Set	Local rounds	Success	Δcost	Median Δcost	Lower / higher (p)	First-block dev.	Drift
First set	0–7	43 vs 41	17.94 [−5.86, 42.07]	0.012	50 / 50 (1.00)	2.43	4.61 [4.41, 4.85]
	8–15	45 vs 41	13.59 [−8.96, 36.44]	0.005	60 / 40 (0.057)	1.46	4.27 [3.75, 4.86]
	16–23	40 vs 41	−3.56 [−24.30, 16.40]	−0.003	55 / 45 (0.37)	0.93	4.17 [3.49, 4.96]
Larger set	0–7	91 vs 79	7.34 [−9.25, 24.80]	−0.011	114 / 86 (0.056)	2.46	4.89 [4.72, 5.08]
	16–23	79 vs 79	−11.32 [−23.57, 0.71]	−0.017	105 / 95 (0.52)	0.91	4.47 [3.90, 5.13]

Why whitened, distance-weighted ridge regression. Whitening by the current proposal scale makes the regression isotropic in the coordinates in which candidates are sampled, so one ridge coefficient applies to all action dimensions. Distance weighting down-weights sentinels far from the proposal mean, where the local linear description is least accurate. Because the number of sentinels is smaller than the action dimension, the fit is underdetermined; the ridge term regularizes it toward operators of small norm, and the dual form keeps it cheap (Appendix B).

Cost accounting. Exact-query counts ignore the time to fit and apply the operator, synchronization, and sampling. On a GPU the candidates of one round are evaluated in parallel, so reducing the population leaves the time of a round nearly unchanged while removing a round removes its full cost (Table 2, Appendix O). We therefore report complete planning time per decision as the primary cost and compare timings only within one experiment on one GPU.

Table 29: Experiments reported in the paper. Sources: distinct (episode, start) pairs; each source contributes two planning decisions. Seeds: search seeds (for Two-Room, the seed of the official evaluator). Arms: number and list of planner configurations, all run on the same sources and GPU.

Experiment	Task	Planner	Sources	Seeds	Arms
Operator accuracy by round	PushT	CEM	100	51	native (diagnostic)
Stage-wise substitution, first set	PushT	CEM	50	51	8: native; local and noise per stage
Stage-wise substitution, replication	PushT	CEM	50	52	4: native; local 0–3 / 4–7 / 8–11
Stage-wise substitution, larger set	PushT	CEM	200	51	3: native; local 0–3 / 8–11
Matched-compute comparison	PushT	CEM	100	51, 52	7: native, E12, E8L4, N=522, N=442, R=13, iCEM
Front-loaded populations	PushT	CEM	100	51	4: native, R=13, two front-loaded
Rounds curve	PushT	CEM	100	51	7: $R \in \{3, 5, \ldots, 15\}$
Adaptive stopping	PushT	CEM	rounds-curve sources	51	2 thresholds
Two-Room stage-wise and rounds	Two-Room	CEM	50	42 (evaluator)	6: native, R=28, R=15, local 0–7 / 16–23, official solver
Two-Room second set	Two-Room	CEM	50	43 (permutation)	5: native; local and noise at 0–7 / 16–23
Operator accuracy by round	Reacher	CEM	50	42	native (diagnostic)
Stage-wise substitution, first set	Reacher	CEM	50	42	4: native; local 0–7 / 8–15 / 16–23
Stage-wise substitution, larger set	Reacher	CEM	100	42	3: native; local 0–7 / 16–23
Matched-noise control	Reacher	CEM	50	42	5: native; local and noise at 0–7 / 16–23
Matched-compute comparison	Reacher	CEM	50	42, 43	5: native, E12, N=131, R=13, iCEM
Rounds curve	Reacher	CEM	50	42, 43	8: $R \in \{5, 8, 10, 13, 16, 20, 25, 30\}$
Adaptive stopping	Reacher	CEM	rounds-curve sources	42, 43	2 thresholds
Other planners	PushT	iCEM, MPPI	50	51	per planner 5: native; local and noise at 0–3 / 8–11
Other elite counts	PushT	CEM, K=15, 60	50	51	per K 3: native; local 0–3 / 8–11
Second world model (LeWM)	PushT	CEM	50	51	3: native; local 0–3 / 8–11
Schedule sensitivity	PushT	CEM	50	51	5: native, E10, E11, E12, E13
Stronger noise	PushT	CEM	50	51	5: native; noise $k = 1, 2$ at 0–3 / 8–11
MPPI stage-wise and diagnostic	Two-Room	MPPI	second-set sources	43 (permutation)	4: native, 15 iterations, local 0–7 / 16–23
MPPI stage-wise and diagnostic	Reacher	MPPI	50	42	4: native, 15 iterations, local 0–7 / 16–23
Diagnostic test	PushT	iCEM, MPPI, CEM	other-planner, elite-count, LeWM sources	51	per setting 2: 15 and 7 iterations
Diagnostic test	Two-Room	CEM	second-set sources	43 (permutation)	2: R=30, R=8