BRANCH-PROX: JOINT ARRIVAL-TIME AND ACTION OPTIMIZATION WITH WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Goal-conditioned planning with world models often optimizes action sequences by matching the predicted terminal representation to a goal representation at a fixed horizon. However, this objective does not explicitly encourage reaching the goal sooner. We introduce Branch-Prox, a trajectory optimization method that jointly optimizes arrival time and action sequences through a differentiable world model. Each candidate arrival time defines a branch whose objective combines elapsed time with a penalty for exceeding a goal-matching tolerance in latent space. An earlier arrival time may appear unfavorable under the current action sequence yet become preferable after action refinement. Rather than selecting an arrival time based on the current trajectory alone, Branch-Prox solves a convex proximal subproblem for each candidate and jointly selects the arrival time and action update using the optimized local model values. Each box-constrained subproblem reduces to a one-dimensional dual search. Under standard smoothness assumptions on the goal-matching costs and exact branch solves, we establish sufficient decrease of the planning objective with backtracking line search. Branch-Prox integrates into model predictive control without world-model fine-tuning or auxiliary model training. Across three LeWorldModel case studies with 64 matched episodes and three planner seeds, Branch-Prox achieves the lowest mean failure-capped steps among common-horizon reference methods. On Reacher, the mean decreases from the strongest reference baseline's 34.40 steps to 13.57 with 100% native success. On DINO-WM Wall, Branch-Prox reaches all 50 goals with 19.30 mean capped steps, compared with 37.44 for iCEM.

1 INTRODUCTION

A prediction horizon limits how far a planner can look ahead; it need not determine when the planner aims to arrive. Yet a terminal-only goal-matching objective scores a plan solely through its final prediction. Consider two predicted plans with identical final predictions: one reaches the goal at the third action block and stays there; the other arrives at the eighth. A terminal-only objective assigns them the same score. The model can reveal an opportunity for earlier completion, but the objective gives the planner no reason to prefer it.

This gap matters for visual goal planning, where an image specifies the desired outcome (Nair et al., 2018) and a world model predicts the consequences of candidate actions (Finn & Levine, 2017; Hafner et al., 2019). We study the planning layer of frozen predictors such as DINO-WM and LeWorldModel (Zhou et al., 2025; Maes et al., 2026): *can the same predictor support earlier arrival by changing what the planner optimizes?* Receding-horizon execution (Mayne et al., 2000) and stopping at success govern execution; neither adds an early-arrival preference to a terminal-only objective.

We separate a fixed maximum prediction window H from a candidate arrival endpoint $h \leq H$, and optimize this endpoint jointly with the actions. Each endpoint trades elapsed model blocks against a penalty for normalized latent goal mismatch above a tolerance. Their minimum gives a *penalized arrival envelope* that can prefer earlier goal matching without shortening the lookahead. The tolerance specifies relative latent progress; physical success remains an environment-level check.

Choosing an endpoint is not simply finding the best-looking frame in the current rollout. *An endpoint that looks worse now can become preferable after its actions are improved.* A late endpoint may

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.

already satisfy the tolerance and yield no local action update, while an earlier endpoint admits a correction that lowers the arrival objective. Refining only the current winner can miss this opportunity. We therefore compare endpoints through their locally improved plans.

We introduce **Branch-Prox**, which *refines before selecting* (Figure 1). For each endpoint, it uses a prox-linear model (Lewis & Wright, 2016; Drusvyatskiy & Paquette, 2019): the smooth goal-matching cost is linearized while retaining the tolerance hinge and action box. It solves each proximal subproblem before comparing branch values, using coordinatewise clipping and a scalar dual search. Backtracking checks the selected update against the original nonlinear envelope. With Lipschitz endpoint gradients and exact branch solves, sufficiently small proximal step sizes ensure sufficient decrease of this latent objective. The planner requires no additional model, policy or critic training.

We evaluate TwoRoom, PushT and Reacher with LeWorldModel, and Wall with DINO-WM; Section 4 specifies task-selection scope. Main reference conditions use 64 matched episodes and three planner seeds. On Reacher, Branch-Prox reaches 13.57 capped steps and 100% native success, versus 34.40 steps for the strongest reference baseline, GRASP (Psenka et al., 2026). Controls identify contributions from the arrival objective and, most clearly on TwoRoom, the structured update, including against same-objective Projected Adam (Kingma & Ba, 2015). Wall evaluates arrival timing with a second frozen predictor; timing measurements quantify the additional computation.

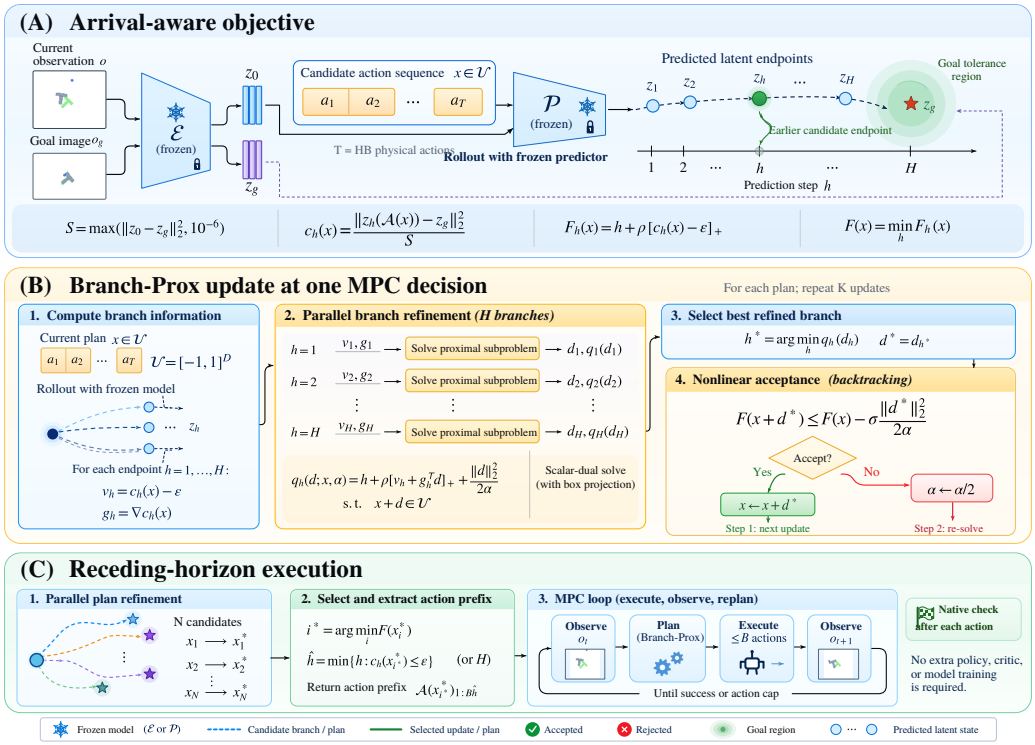


Figure 1: **Branch-Prox: objective, refinement and execution.** (A) Frozen representations define endpoint costs. (B) Refine every branch before selection; rejection reduces the step size and repeats branch solves. (C) Refine N plans for K updates, select by F, and execute at most B actions before replanning. Native success is checked after every action; execution ends at success or the action cap. All imagery is schematic; Algorithm 1 gives the full update.

2 Related Work

Visual dynamics and goal representations. Visual foresight, E2C and PlaNet plan using learned visual dynamics (Finn & Levine, 2017; Watter et al., 2015; Hafner et al., 2019). DINO-WM, V-JEPA 2 and LeWorldModel connect predictive representations to action-conditioned planning (Zhou et al., 2025; Oquab et al., 2024; Assran et al., 2025; Maes et al., 2026). Dreamer and TD-MPC also

learn policies or value functions (Hafner et al., 2025; Hansen et al., 2022; 2024). We optimize actions with released frozen predictors.

Action optimization and learned trajectory scoring. CEM, PETS, iCEM and MPPI support sampling-based planning (de Boer et al., 2005; Chua et al., 2018; Pinneri et al., 2021; Williams et al., 2017). GRASP uses virtual states, soft dynamics constraints and stochastic exploration (Psenka et al., 2026). Traj-LeWM learns a trajectory cost and shapes representations (Huang et al., 2026); Reinforced Planning learns a critic and plan-improvement operator (Sommer & Schilling, 2026). Our arrival score uses the existing latent distance without additional training.

Time and goals in planning. MPC executes plan prefixes (Mayne et al., 2000), while free-final-time control optimizes duration jointly with controls (Szmuk & Acikmese, 2018). Goal- and horizon-conditioned values (Schaul et al., 2015; Pong et al., 2018) and options (Sutton et al., 1999; Huang et al., 2025) connect control to goals and temporal abstraction. We retain primitive-action MPC and optimize discrete endpoints within a bounded rollout.

Structured local optimization. Proximal and prox-linear methods preserve structure in local subproblems (Parikh & Boyd, 2014; Lewis & Wright, 2016; Drusvyatskiy & Paquette, 2019); finite-minimum optimization admits branch enumeration (Van Dessel & Glineur, 2024). We specialize these established ideas to the arrival envelope through box-constrained hinge solves, refinement before selection and nonlinear acceptance. Our contribution is the planning construction and evidence separating objective from update effects.

3 Joint arrival-time and action optimization

We optimize when to match the goal within a bounded prediction window, together with the actions that make that match possible. A penalized arrival envelope defines this choice; branchwise local refinement makes it tractable, and nonlinear backtracking checks each proposed update.

3.1 A penalized arrival envelope

At an MPC decision, let $z_0 = E(o)$ and $z_g = E(o_g)$ encode the current and goal images. A frozen predictor rolls out an action plan to endpoints $h = 1, \dots, H$, each separated by B physical actions ($B = 5$ here). The maximum window H is fixed; the candidate arrival endpoint h is a decision variable. Represent the plan by $x \in \mathcal{U} = [-1, 1]^D$, with $D = HBd_a$ for action dimension d_a, and map it to model actions with A. Define

$$S = \max\{\|z_0 - z_g\|_2^2, 10^{-6}\}, \qquad c_h(x) = \|z_h(A(x)) - z_g\|_2^2 / S, \tag{1}$$

$$F_h(x) = h + \rho\,[c_h(x) - \epsilon]_+\,, \qquad F(x) = \min_{1 \le h \le H} F_h(x), \qquad \min_{x \in \mathcal{U}} F(x). \tag{2}$$

Here $[u]_+ = \max\{u, 0\}$. Each branch trades elapsed model blocks against excess latent mismatch, with $\rho > 0$ and $\epsilon \ge 0$. The scale S is fixed within a solve and recomputed at each MPC decision. Away from its numerical floor, $c_h \le \epsilon$ requires latent distance at most $\sqrt{\epsilon}$ times the current distance (22.4% for $\epsilon = 0.05$). This is a relative progress tolerance; physical success is checked separately. Finite ρ can favor a proxy-infeasible early branch over a feasible later one.

The affine map $A(x) = a_{\text{center}} + a_{\text{radius}} \odot x$ converts the unit box to the model's standardized action bounds. All derivatives and proximal norms use x. The controller reverses standardization before execution: $a_{\text{phys}} = \mu_a + \sigma_a \odot A(x)$, with the dataset action mean and scale. Thus the planner returns model actions; the environment receives physical actions.

3.2 Compare branches after refinement

The update compares endpoints after local action refinement. For each endpoint we linearize the smooth cost, preserve its hinge and add a quadratic proximal term. With $v_h = c_h(x) - \epsilon$, $g_h = \nabla c_h(x)$ and $\alpha > 0$, form

$$q_h(d; x, \alpha) = h + \rho\left[v_h + g_h^\top d\right]_+ + \frac{\|d\|_2^2}{2\alpha}, \qquad x + d \in \mathcal{U}. \tag{3}$$

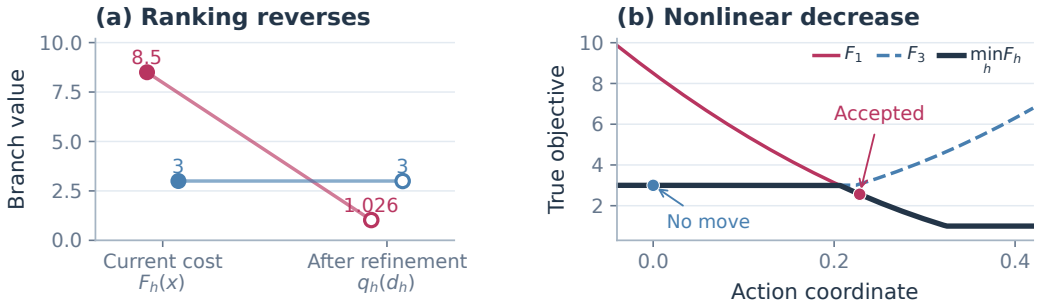


Figure 2: **Why refine before selecting?** Analytic example defined in Appendix A.5. Filled and open markers show current costs and optimized local-model values. At $x = 0$, branch 3 is best and its update is zero. Refining all branches selects branch 1: its optimized local value is 1.0260, while the actual envelope decreases from 3 to 2.5625. Branch 2 is dominated throughout the action box and omitted.

Every branch is strongly convex in d and has a unique minimizer d_h. Branch-Prox selects

$$h_* \in \arg\min_h q_h(d_h; x, \alpha), \qquad d_* = d_{h_*}. \tag{4}$$

Finite enumeration globally minimizes the local finite-branch model $\min_{x+d\in\mathcal{U}} \min_h q_h(d; x, \alpha)$. In particular,

$$q_{h_*}(d_*) \le q_j(d_j) \le q_j(0) = F(x) \quad \text{for any } j \in \arg\min_h F_h(x). \tag{5}$$

This ordering compares locally refined branches. Selecting after refinement exposes an opportunity that choosing the current winner can miss; the nonlinear envelope determines whether to accept the resulting proposal.

Scalar solution of each branch. Each branch needs only coordinatewise clipping and a one-dimensional multiplier search. Using $\rho[u]_+ = \max_{0\le\lambda\le\rho} \lambda u$, define

$$d_h(\lambda) = \Pi_{\mathcal{U}}(x - \alpha\lambda g_h) - x, \qquad r_h(\lambda) = v_h + g_h^\top d_h(\lambda). \tag{6}$$

Here $\Pi_{\mathcal{U}}$ is coordinatewise clipping. The following characterization supplies the branch update; its proof appears in Appendix A.

Proposition 1 (Box-constrained hinge proximal map)**.** *The residual r_h is continuous and nonincreasing. A minimizing displacement is $d_h(\lambda_h)$, where one may choose $\lambda_h = 0$ if $v_h \le 0$, $\lambda_h = \rho$ if $v_h > 0$ and $r_h(\rho) \ge 0$, and otherwise any root of $r_h(\lambda_h) = 0$ in $(0, \rho)$.*

Thus bisection suffices. If the box is inactive and $g_h \neq 0$, the multiplier is $\mathrm{clip}_{[0,\rho]}(v_h/(\alpha\|g_h\|^2))$. We use $M_{\mathrm{dual}} = 28$ bisections, batched over plans and branches, reusing stored gradients without further model evaluations.

3.3 Backtracking and Sufficient Decrease

The local model ranks proposals; the nonlinear envelope decides acceptance. We require

$$F(x + d_*) \le F(x) - \frac{\sigma}{2\alpha}\|d_*\|^2, \qquad 0 < \sigma < 1. \tag{7}$$

Rejection halves α and triggers re-solving and reselection at the same x, reusing its gradients. The selected branch can change with the step size.

Theorem 1 (Sufficient decrease)**.** *Fix the current observation, goal and scale. Suppose every c_h is continuously differentiable with L-Lipschitz gradient on a neighborhood of $\mathcal{U}$. If the local branches are solved exactly, their selected update satisfies*

$$F(x + d_*) \le F(x) - \frac{1 - \alpha\rho L}{2\alpha}\|d_*\|^2. \tag{8}$$

Algorithm 1 Branch-Prox at one MPC decision

Require: Images o, o_g; frozen encoder E and predictor P; action map A
Require: N plans, K updates, H endpoints, B actions/block, J_{bt} trials; ρ, ϵ, σ
Ensure: Standardized-action prefix; controller reverses standardization before execution

$z_0 \leftarrow E(o)$; $z_g \leftarrow E(o_g)$; $S \leftarrow \max\{\|z_0 - z_g\|_2^2, 10^{-6}\}$
Initialize $x_i \in \mathcal{U}$ and $\alpha_i > 0$ independently, $i = 1, \dots, N$
for $k = 1, \dots, K$ **do**
  Roll out P and compute $c_h(x_i), g_h(x_i) = \nabla c_h(x_i)$ for all i, h
  for $i = 1, \dots, N$ **in parallel do**
    $\bar{\alpha} \leftarrow \alpha_i$
    for $j = 1, \dots, J_{\mathrm{bt}}$ **do**
      Solve all d_h by the scalar dual in (6) at $\bar{\alpha}$
      $h_* \leftarrow \arg\min_h q_h(d_h; x_i, \bar{\alpha})$; $d_* \leftarrow d_{h_*}$
      if $F(x_i + d_*) \le F(x_i) - \sigma\|d_*\|_2^2/(2\bar{\alpha})$ **then**
        $x_i \leftarrow x_i + d_*$; $\bar{\alpha} \leftarrow \min\{2\bar{\alpha}, 10^4\}$; **break**
      end if
      $\bar{\alpha} \leftarrow \bar{\alpha}/2$
    end for
    $\alpha_i \leftarrow \bar{\alpha}$ ▷ x_i unchanged if all trials fail
  end for
end for
$i_* \leftarrow \arg\min_i F(x_i)$ using the nonlinear envelope
$\hat{h} \leftarrow \min\{h : c_h(x_{i_*}) \le \epsilon\}$, or H if the set is empty
return $A(x_{i_*})_{1:B\hat{h}}$

For $L > 0$, any $\alpha \le (1 - \sigma)/(\rho L)$ satisfies (7)*; unlimited halving therefore terminates in finitely many trials. If $L = 0$, every positive α satisfies the test.*

The proof is given in Appendix A.

The theorem establishes sufficient decrease of the latent envelope within each MPC solve. The implementation uses $J_{\mathrm{bt}} = 12$ trials and retains x if all fail; after acceptance it doubles the next step size, capped at 10^4. Appendix C.3 examines finite-precision acceptance through a recorded float32 comparison exception.

A current winner can hide an improving earlier branch. Figure 2 shows a current winner with a zero update and an earlier branch whose refinement decreases F; Appendix A.5 specifies every cost and parameter. A recorded LeWorldModel PushT update exhibits the same ranking reversal within a planning solve: the current winner is $h = 6$, while refining $h = 1$ lowers the envelope from 12.671 to 10.418 (Appendix C.3).

3.4 From a local update to receding-horizon execution

Arrival selection acts through the optimized action sequence. Algorithm 1 ranks the final candidates by the nonlinear envelope and returns the prefix ending at the earliest proxy-feasible endpoint, or H if none exists. The controller reverses standardization and executes at most the first block, checking native success after every action, before replanning. The local-model winner, proxy-feasible prefix and physical first-success time are distinct; returning a shorter prefix does not itself change the executed block. Reference Branch-Prox restarts plans at each decision.

Storing endpoint derivatives costs $O(NHD)$ entries; dual arithmetic costs $O(M_{\mathrm{dual}}NHD)$ per backtracking trial. The main expense is obtaining the H endpoint gradients through sequential vector–Jacobian products, batching the N plans. With $H = 8$ and $K = 20$, this requires 160 reverse invocations per solve, plus nonlinear-check rollouts. Although an endpoint is independent of later actions, the implementation still differentiates every endpoint; choosing early arrival does not reduce this work.

4 EXPERIMENTS

We study three questions: does optimizing arrival improve physical execution, what does refinement before selection contribute, and how much computation does it require? We present TwoRoom, PushT and Reacher with LeWorldModel, and Wall with DINO-WM. These tasks were selected after inspecting outcomes from a broader evaluation, so the empirical conclusions concern this selected subset. Complete records are retained in the accompanying research archive.

Shared execution protocol. We use released frozen LeWorldModel checkpoints (Maes et al., 2026) on the three displayed tasks. Each primary condition uses 64 matched independent episodes and planner seeds 42/43/44, with goals 50 recorded actions ahead. Component and same-objective controls use the first 32 and 16 of these episodes, respectively. All methods predict up to $H = 8$ blocks of $B = 5$ actions, execute at most one block per MPC decision, and check native success after every physical action. Every method stops at success or 100 actions. Thus all planners can stop early; the distinction is whether their planning objective rewards early arrival. Models, cases and physical action limits are shared. Native predicates and observation details are in Appendix B.

Comparators and controls. CEM, fixed-population iCEM, elite-truncated MPPI* and AdamW (terminal) (Kingma & Ba, 2015; Loshchilov & Hutter, 2019) optimize terminal latent distance; GRASP uses its released relaxed state/action objective. Reference Branch-Prox uses 16 fresh plans, 20 updates, $\rho = 30$ and $\epsilon = 0.05$; external warm starts are retained. These method-level comparisons include differences in objective, projection and initialization (Appendix B). Component controls and a separate projected optimizer of the same arrival envelope address contribution attribution.

Measuring arrival. We report $T_{\text{cap}} = \min\{T_{\text{native}}, 100\}$, with $T_{\text{native}} = \infty$ for failure and zero for initial success, alongside native success rate. This failure-capped first-arrival metric equals the area under the discrete probability-of-not-yet-arriving curve: $\mathbb{E}[T_{\text{cap}}] = \sum_{t=0}^{99} \Pr(T_{\text{native}} > t)$. It rewards both more reliable and earlier arrival; the full arrival curves show their joint effect. Three seeds are averaged within each episode before 10,000 episode-bootstrap draws. Paired sign-flip tests retain Holm correction over the original 20-comparison reference family (Holm, 1979). Other studies are secondary; reported 95% intervals are marginal.

4.1 PHYSICAL ARRIVAL UNDER A COMMON PREDICTION WINDOW

Table 1: **Common-horizon reference: failure-capped physical steps ↓.** All methods use $H = 8$, 64 matched episodes and three planner seeds. Entries are mean [marginal 95% interval]; bold marks the lowest point estimate, not significance. Initial successes are 1/0/0 across the task columns. Success rates and adjusted primary tests appear in Appendix C.

Method	TwoRoom ↓	PushT ↓	Reacher ↓
CEM	79.21 [71.56, 86.21]	89.18 [84.29, 93.59]	82.98 [78.65, 87.12]
iCEM	79.42 [72.35, 86.10]	95.14 [91.83, 97.92]	81.53 [76.17, 86.58]
MPPI*	63.08 [56.61, 69.60]	93.50 [89.93, 96.62]	80.63 [75.81, 85.12]
AdamW	71.92 [64.73, 78.70]	88.90 [84.58, 92.91]	82.22 [77.18, 86.89]
GRASP	83.32 [75.65, 90.21]	75.43 [66.30, 84.15]	34.40 [25.34, 43.86]
Branch-Prox	**33.43** [26.35, 41.04]	**65.68** [56.44, 74.51]	**13.57** [11.81, 15.48]

Branch-Prox achieves the lowest mean capped steps on all three tasks (Table 1), with native success rates of 80.21%, 44.79% and 100%. Figure 3 shows when these successes occur under the shared stopping rule. Fourteen of the 15 displayed contrasts pass the original-family Holm threshold. The remaining contrast, against GRASP on PushT, is $-9.75\,[-17.77, -1.80]$ steps with adjusted $p = 0.1022$.

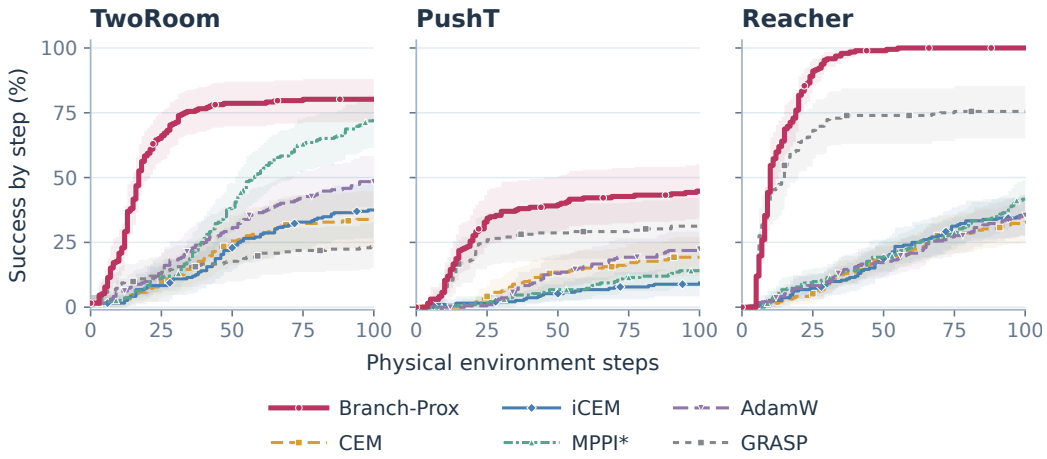


Figure 3: **Native goal arrival over physical execution.** All reference methods share the maximum prediction window and success stopping rule. Curves use 64 episodes and three seeds; shading is pointwise marginal 95% episode-bootstrap uncertainty. Step zero includes initial successes. Higher curves indicate more goals reached by a given step; planning latency is measured separately.

4.2 Comparison with development-selected planners

Each external family selects $H \in \{3, 5, 8\}$ and a fast/reference budget on 16 development episodes (seed 42), then runs that setting on all 64 test episodes and three seeds; Branch-Prox retains its reference configuration. This tests the complete planners after development selection.

Table 2: **Δ steps = Branch-Prox $-$ development-selected baseline; negative favours Branch-Prox.** The same 64 episodes and three seeds yield mean paired differences [marginal secondary 95% intervals]. The final row gives BP absolute means; baseline means equal this reference minus the displayed difference. Selected horizons may differ from $H = 8$; work is unequal.

Method	TwoRoom	PushT	Reacher
CEM	-39.69 [-48.57, -31.04]	-8.03 [-15.98, -0.39]	-21.39 [-27.02, -16.51]
iCEM	-30.33 [-37.55, -23.49]	-8.70 [-16.23, -1.47]	-20.52 [-25.08, -16.62]
MPPI*	-23.23 [-30.66, -16.13]	-13.73 [-22.04, -5.72]	-26.75 [-30.93, -22.80]
AdamW	-22.55 [-28.72, -16.74]	-5.54 [-14.19, 3.13]	-19.52 [-22.93, -16.17]
GRASP	-44.51 [-52.84, -36.30]	-8.55 [-16.98, -0.27]	-22.28 [-30.33, -14.77]
BP absolute steps	33.43	65.68	13.57

Reacher AdamW improves from 82.22 capped steps at $H = 8$ to approximately 33.09 at its selected $H = 3$, versus 13.57 for Branch-Prox. Development selection therefore strengthens the baselines substantially; Branch-Prox retains its clearest advantages on TwoRoom and Reacher.

4.3 Separating the arrival objective from the branch update

On the first 32 test episodes and three seeds, we compare four controlled variants. *Fixed H* optimizes only the terminal branch F_H; *No time* removes the elapsed-time term; *Current branch* chooses the current winning endpoint before refinement; *Fixed step* uses $\alpha = 1$ without line search or adaptive scheduling. All retain the same MPC stopping rule, 16 initial plans and 20 updates. Eight endpoint

derivatives are deliberately computed in each control to match the mechanism study, rather than optimize its runtime.

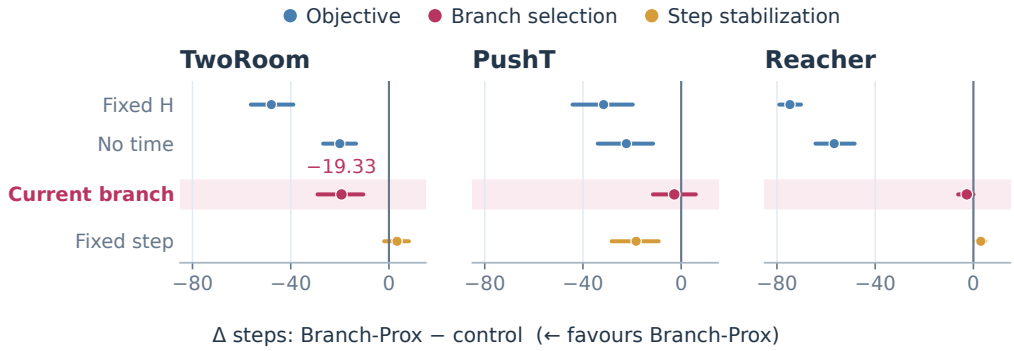


Figure 4: **Separating objective and update effects.** Points show full Branch-Prox minus control in capped steps on 32 episodes and three seeds; lines show marginal secondary 95% paired intervals. Negative favours Branch-Prox. Colors group objective, selection and step-stabilization controls. Fixed step removes both line search and adaptive scheduling. Exact values appear in Table 11.

The arrival formulation contributes across tasks. Full Branch-Prox improves over Fixed H on the three displayed tasks by marginal paired intervals, and over No time on TwoRoom, PushT and Reacher (Figure 4). These contrasts test multiple endpoints and the elapsed-time term, respectively. All controls stop at native success, so their planning objectives drive the contrast.

Refinement before selection contributes most clearly on TwoRoom. Against Current branch, the TwoRoom difference is $-19.33\,[-29.10, -10.43]$ capped steps; the other two task intervals include zero. Fixed step is better on Reacher by $3.00\,[1.23, 4.89]$ steps, so step stabilization is task dependent (Table 11). A recorded earlier-endpoint update illustrates the selection mechanism (Appendix C.3). On TwoRoom, the improvement primarily reflects reliability: Full and Current branch succeed in 85.42% and 61.46% of executions, while their episode-balanced difference on jointly successful case–seed pairs is $+0.04$ steps. Appendix F defines this conditional diagnostic.

Same objective and constraints: Projected Adam (arrival). Projected Adam (Kingma & Ba, 2015) optimizes the identical nonlinear envelope with the same action box, initialization and final ranking. Both use $H = 8$, $N = 16$ and 20 updates; BP differentiates eight endpoints per update, Adam one, so this is not an equal-time comparison. Adam projects every update. Its task-specific learning rate is selected from $\{.001, .01, .1\}$ on all 16 original development episodes before testing the first 16 original episodes with three seeds. This secondary control reuses the corresponding BP records; full settings are in Appendix D.

Table 3: **Same arrival objective, different update.** Mean capped steps on 16 matched episodes per task and three seeds. Δ = Branch-Prox − Adam; brackets are marginal paired 95% intervals. Holm correction uses the original family of four secondary tests.

Task	Branch-Prox	Proj. Adam	Δ steps [95% CI]	Holm p
TwoRoom	24.00	48.60	-24.60 [-40.02, -10.83]	**0.0128**
PushT	60.46	72.94	-12.48 [-22.67, -3.19]	0.0791
Reacher	15.46	17.77	-2.31 [-6.17, $+0.71$]	0.4915

With the objective and constraints matched, Branch-Prox reduces TwoRoom capped steps by 24.60 relative to Projected Adam (Table 3; Holm $p = 0.0128$), isolating an additional contribution from its update. The point estimates also favour Branch-Prox on PushT and Reacher; those two comparisons do not pass the four-test Holm threshold.

4.4 ARRIVAL TIMING IN THE DINO-WM WALL CASE STUDY

DINO-WM (Zhou et al., 2025) Wall evaluates the arrival formulation with a second frozen predictor and a different visual environment. All methods use $H = 5$, five actions per block and seed 42 on 50 independent start–goal pairs. The BP configuration was selected on eight disjoint development cases before these test executions. Intervals measure episode uncertainty at this single seed.

Table 4: **DINO-WM Wall: 50 start–goal pairs, common $H = 5$, one planner seed.** Mean [marginal 95% episode-bootstrap interval]; bold marks the best point estimate. Appendix E gives configuration, interval and timing details.

Method	Capped steps ↓	Success (%) ↑
CEM	41.10 [35.24, 47.82]	92.0 [84.0, 98.0]
iCEM	37.44 [32.44, 43.30]	96.0 [90.0, 100.0]
MPPI*	49.92 [43.40, 57.02]	88.0 [78.0, 96.0]
AdamW	41.26 [35.42, 47.58]	94.0 [86.0, 100.0]
Branch-Prox	**19.30** [16.64, 22.22]	**100.0** [100.0, 100.0]

Branch-Prox reaches all 50 goals with 19.30 mean capped steps; all four paired step comparisons favour it under the original multiplicity correction. At 25 actions, BP has reached 84% of goals versus iCEM's 20%. On their 48 jointly successful cases, it arrives 15.92 steps earlier on average, locating a substantial timing difference within that subset. Appendix F reports the same 25/50/100-action cutoffs and conditional common-success summaries for every comparator.

4.5 COMPUTATIONAL COST AND SENSITIVITY

Endpoint differentiation dominates the scalar-dual solve. A common-$H = 8$ budget sweep pairs physical performance with separately measured exclusive solve latency (Appendix G, Figure 5). The default Branch-Prox median on Reacher is 7.71 s, exceeding every measured iCEM budget. The gain is in physical action use, at additional planning cost; this fixed-horizon sweep measures that trade-off rather than equal-time performance.

Appendix G gives the nine-setting sensitivity grid for the displayed tasks on 16 episodes and three seeds. Penalty and update-budget effects vary across tasks, motivating development selection.

5 DISCUSSION AND CONCLUSION

A fixed prediction window need not impose a terminal-only planning objective. Branch-Prox makes arrival endpoints part of action optimization and compares them after local refinement. The tolerance hinge and action box admit a scalar-dual branch solve; nonlinear backtracking provides sufficient decrease under the stated smoothness and exact-solve conditions. This construction gives frozen predictors an explicit preference for earlier goal matching without additional training.

The experiments distinguish two contributions. Across the three LeWorldModel tasks, the arrival objective improves on terminal-only and time-free controls. On TwoRoom, the structured update improves capped steps over both current-branch refinement and same-objective Projected Adam. Wall demonstrates earlier physical arrival with DINO-WM: 84% of goals are reached within 25 actions. Endpoint differentiation adds planning cost, making efficient derivative computation and allocation of refinement across endpoints natural next steps. Together, the formulation and update make arrival timing an explicit target for planning with frozen world models.

AI USE STATEMENT

Generative AI tools were used solely for language editing and polishing of the Introduction, Related Work, and Experiments sections, with the purpose of improving grammar, clarity, and readability. The scientific content and contributions of the paper remain the work of the authors. All AI-assisted edits were reviewed and verified by the authors. The authors take full responsibility for the final content of the paper.

REFERENCES

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. URL https://web.stanford.edu/~boyd/cvxbook/.

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. URL https://proceedings.neurips.cc/paper/2018/hash/3de568f8597b94bda53149c7d7f5958c-Abstract.html.

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. doi: 10.1007/s10479-005-5724-z. URL https://link.springer.com/article/10.1007/s10479-005-5724-z.

Dmitriy Drusvyatskiy and Courtney Paquette. Efficiency of minimizing compositions of convex functions and smooth maps. *Mathematical Programming*, 178:503–558, 2019. doi: 10.1007/s10107-018-1311-3. URL https://link.springer.com/article/10.1007/s10107-018-1311-3.

Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *IEEE International Conference on Robotics and Automation*, 2017. URL https://arxiv.org/abs/1610.00696.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 2555–2565, 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025. doi: 10.1038/s41586-025-08744-2. URL https://www.nature.com/articles/s41586-025-08744-2.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.16828.

Nicklas A. Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 8387–8406, 2022. URL https://proceedings.mlr.press/v162/hansen22a.html.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. URL https://www.jstor.org/stable/4615733.

Po-Wei Huang, Pei-Chiun Peng, Hung Guei, and Ti-Rong Wu. OptionZero: Planning with learned options. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2502.16634.

Xiaodi Huang, Ziyi Ding, Jingtian Wan, Yuchen Liu, Yuan Zhang, Xiao-Ping Zhang, Jiayu Chen, Zhang Zhang, and Tao Huang. Traj-LeWM: Path-Aware World-Model Planning via Latent Trajectory Cost, 2026. URL https://arxiv.org/abs/2608.14125.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

A. S. Lewis and S. J. Wright. A proximal method for composite minimization. *Mathematical Programming*, 158:501–546, 2016. doi: 10.1007/s10107-015-0943-9. URL https://link.springer.com/article/10.1007/s10107-015-0943-9.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels, 2026. URL https://arxiv.org/abs/2603.19312.

David Q. Mayne, James B. Rawlings, Christopher V. Rao, and Pierre O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000. doi: 10.1016/S0005-1098(99)00214-9. URL https://doi.org/10.1016/S0005-1098(99)00214-9.

Ashvin Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://arxiv.org/abs/1807.04742.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=a68SUt6zFt.

Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2014. URL https://web.stanford.edu/~boyd/papers/prox_algs.html.

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, pp. 1049–1065, 2021. URL https://proceedings.mlr.press/v155/pinneri21a.html.

Vitchyr Pong, Shixiang Gu, Murtaza Dalal, and Sergey Levine. Temporal Difference Models: Model-Free Deep RL for Model-Based Control, 2018. URL https://arxiv.org/abs/1802.09081.

Michael Psenka, Michael Rabbat, Aditi Krishnapriyan, Yann LeCun, and Amir Bar. Parallel stochastic gradient-based planning for world models, 2026. URL https://arxiv.org/abs/2602.00475.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pp. 1312–1320, 2015. URL https://proceedings.mlr.press/v37/schaul15.html.

Armin Sommer and Jannik Schilling. Reinforced Planning with Latent World Models, 2026. URL https://arxiv.org/abs/2608.18669.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211, 1999. doi: 10.1016/S0004-3702(99)00052-1. URL https://doi.org/10.1016/S0004-3702(99)00052-1.

Michael Szmuk and Behcet Acikmese. Successive convexification for 6-dof mars rocket powered landing with free-final-time, 2018. URL https://arxiv.org/abs/1802.03827.

Guillaume Van Dessel and François Glineur. Global minimization of a minimum of a finite collection of functions, 2024. URL https://arxiv.org/abs/2412.04625.

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. URL https://arxiv.org/abs/1506.07365.

Grady Williams, Paul Drews, Brian Goldfain, James M. Rehg, and Evangelos A. Theodorou. Information theoretic model predictive control: Theory and applications to autonomous driving, 2017. URL https://arxiv.org/abs/1707.02342.

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, 2025. URL https://proceedings.mlr.press/v267/zhou25t.html.

A LOCAL SUBPROBLEM AND DESCENT PROOFS

The results in this section concern a single candidate at one MPC decision. The observation, goal, normalization scale S, action box and gradients at the base plan are fixed. Multiple starts repeat the same update independently; they are not needed for the guarantees. Throughout, $\mathcal{U}$ is a nonempty compact box, $x \in \mathcal{U}$, $\alpha > 0$, and $\rho > 0$.

A.1 PROOF OF PROPOSITION 1

Suppress the branch subscript and the constant h. With $C = \mathcal{U} - x$, write the local problem as

$$p_* = \min_{d \in C} \left\{ \rho[v + g^\top d]_+ + \frac{\|d\|^2}{2\alpha} \right\}. \tag{9}$$

The objective is $1/\alpha$-strongly convex on the convex set C. Compactness gives existence and strong convexity gives a unique minimizing displacement. We use the standard primal–dual viewpoint (Boyd & Vandenberghe, 2004) and verify optimality directly below. Define

$$L(d, \lambda) = \lambda(v + g^\top d) + \frac{\|d\|^2}{2\alpha}, \qquad \psi(\lambda) = \min_{d \in C} L(d, \lambda), \quad 0 \le \lambda \le \rho. \tag{10}$$

Completing the square gives the unique minimizer

$$d(\lambda) = \arg\min_{d \in C} \|d + \alpha\lambda g\|^2 = \Pi_{\mathcal{U}}(x - \alpha\lambda g) - x. \tag{11}$$

The dual function is concave and continuously differentiable, with $\psi'(\lambda) = v + g^\top d(\lambda) = r(\lambda)$. This derivative also follows directly by differentiating the coordinatewise clipped quadratic pieces; their derivatives agree at clipping boundaries. Projection is continuous, so r is continuous. For each coordinate with $g_i > 0$, the projection decreases as λ increases and hence $g_i d_i(\lambda)$ is nonincreasing. For $g_i < 0$, the projection increases, again making $g_i d_i(\lambda)$ nonincreasing. Coordinates with $g_i = 0$ contribute zero. Thus r is nonincreasing, including at clipping boundaries.

Because $x \in \mathcal{U}$, $d(0) = 0$ and $r(0) = v$. If $v \le 0$, choose $\lambda_* = 0$; if $v > 0$ and $r(\rho) \ge 0$, choose $\lambda_* = \rho$; otherwise continuity supplies a root in $(0, \rho)$. In each case λ_* maximizes $L(d(\lambda_*), \lambda)$ over $[0, \rho]$: the coefficient of λ is respectively nonpositive, nonnegative or zero. Also $d(\lambda_*)$ minimizes $L(d, \lambda_*)$. These two facts constitute a saddle point and establish primal optimality directly. Multipliers need not be unique when r is flat, but the primal displacement is unique.

If the box is inactive, $d(\lambda) = -\alpha\lambda g$, so $r(\lambda) = v - \alpha\lambda\|g\|^2$. For $g \neq 0$ this yields the clipped formula in the main text. If $g = 0$, the displacement is zero for every multiplier; an endpoint multiplier is sufficient according to the sign of v.

A.2 ENUMERATION AND MODEL DECREASE

For the continuous q_h on the compact set C, minimizing over d and over the finite branch index commutes. Consequently,

$$\min_{d \in C} \min_h q_h(d) = \min_h \min_{d \in C} q_h(d) = q_{h_*}(d_{h_*}). \tag{12}$$

Although each q_h is strongly convex, their pointwise minimum generally is not. Enumeration uses convexity within each branch and does not assume convexity of their minimum. To compare with the original objective, choose any $j \in \arg\min_h F_h(x)$. The feasible choice $d = 0$ gives

$$q_{h_*}(d_*) \le q_j(d_j) \le q_j(0) = j + \rho[v_j]_+ = F(x). \tag{13}$$

This statement also covers ties and a zero update. Selection minimizes the local model $q_h(d_h)$; the minimizing branch of the nonlinear envelope can differ both before and after the update.

A.3 PROOF OF THEOREM 1

Let each c_h have an L-Lipschitz gradient on a neighborhood of $\mathcal{U}$. The segment from x to $x + d$ lies in $\mathcal{U}$, giving

$$c_h(x + d) - \epsilon \le v_h + g_h^\top d + \frac{L}{2}\|d\|^2. \tag{14}$$

For any real u and $e \geq 0$, $[u+e]_+ \leq [u]_+ + e$. Applying this inequality and monotonicity of the hinge yields

$$F_h(x+d) \leq h + \rho[v_h + g_h^\top d]_+ + \frac{\rho L}{2}\|d\|^2 \tag{15}$$

$$= q_h(d) - \frac{1-\alpha\rho L}{2\alpha}\|d\|^2. \tag{16}$$

In particular,

$$F(x+d_*) \leq F_{h_*}(x+d_*) \leq q_{h_*}(d_*) - \frac{1-\alpha\rho L}{2\alpha}\|d_*\|^2 \leq F(x) - \frac{1-\alpha\rho L}{2\alpha}\|d_*\|^2. \tag{17}$$

If $L > 0$ and $\alpha \leq (1-\sigma)/(\rho L)$, the coefficient in the final inequality is at least $\sigma/(2\alpha)$. Starting from $\alpha_0 > 0$, at most

$$\max\left\{0, \left\lceil \log_2 \frac{\alpha_0 \rho L}{1-\sigma} \right\rceil\right\} \tag{18}$$

halvings suffice in exact arithmetic. For $L = 0$ the condition holds at every positive α. The implementation's 12-trial cap does not ensure that this threshold is reached; it retains the base plan when the cap is exhausted. A zero displacement satisfies the test without providing strict decrease. Changing the observation or scale at the next MPC decision changes F, so the theorem does not compare objectives across decisions.

A.4 Approximate dual solutions and floating point

For a feasible multiplier $\widetilde{\lambda}$ and its projected displacement $\widetilde{d} = d(\widetilde{\lambda})$, an exact-arithmetic primal–dual gap for the branch without its constant h is

$$\delta = \rho[r(\widetilde{\lambda})]_+ - \widetilde{\lambda}\, r(\widetilde{\lambda}) \geq 0. \tag{19}$$

Indeed, the primal objective at $\widetilde{d}$ minus $\psi(\widetilde{\lambda})$ equals this expression. Weak duality gives $0 \leq q_h(\widetilde{d}) - q_h(d_h) \leq \delta$. Twenty-eight ideal bisections leave a multiplier bracket of width $\rho/2^{28}$; in float32, representable endpoints can coincide sooner. Equation (19) is a mathematical diagnostic, not an asserted certificate from the saved float32 runs.

If each approximate branch minimizer is within η of its optimum and the smallest approximate model value is selected, the same proof gives

$$F(x+\widetilde{d}) \leq F(x) + \eta - \frac{1-\alpha\rho L}{2\alpha}\|\widetilde{d}\|^2. \tag{20}$$

This additive error explains why approximate model minimization alone need not imply strict descent near a stationary plan. The implementation directly tests the nonlinear envelope, but that comparison itself is subject to rounding. Appendix C.3 summarizes the recorded Armijo deficits and objective increases, including a reproduced rounding exception for the Fixed H control. Those counts do not certify the exact-arithmetic inequality.

A possible model-decrease safeguard. The proof only needs a feasible candidate satisfying $q_h(d; x, \alpha) \leq F(x)$. In exact arithmetic, including $(j, 0)$ for a current winner $j \in \arg\min_h F_h(x)$ alongside approximate candidates guarantees this condition when the smallest local value is chosen, because $q_j(0) = F(x)$. The same descent inequality then follows without exact branch minimization. This optional safeguard is not implemented in the reported experiments; it neither certifies the best branch nor removes rounding error, and a zero step can stall.

A.5 Derivation of the explanatory example

Figure 2 uses the scalar box $\mathcal{U} = [-1, 1]$, $H = 3$, base point $x = 0$, $\alpha = 1$, $\rho = 30$, $\epsilon = 1/20$ and $\sigma = 10^{-4}$. Define the normalized endpoint costs directly by

$$c_1(u) = (u - \sqrt{3/10})^2, \qquad c_2(u) = 100, \qquad c_3(u) = u^2. \tag{21}$$

These functions reproduce the plotted curves and stored figure values. At the base point, $(v_1, g_1, F_1) = (1/4, -2\sqrt{3/10}, 17/2)$ and $(v_3, g_3, F_3) = (-1/20, 0, 3)$, so the current winner's

local minimizer is $d_3 = 0$. Branch 2 has $g_2 = 0$, $d_2 = 0$ and $F_2 = q_2(0) = 6001/2$; it is dominated throughout $\mathcal{U}$.

The branch-1 box constraint is inactive: $\lambda_1 = (1/4)/(6/5) = 5/24$, $d_1 = 1/(8\sqrt{3/10}) \simeq 0.228218$ and $d_1^2 = 5/96$. Its linearized violation is zero, giving $q_1(d_1) = 197/192 \simeq 1.026042 < q_3(0) = 3$. The nonlinear cost is instead $c_1(d_1) = 49/480 > \epsilon$, hence $F_1(d_1) = 41/16 = 2.5625$ and $F_3(d_1) = 49/16 = 3.0625$. Thus $F(0) - F(d_1) = 7/16$, exceeding the Armijo request $\sigma d_1^2/(2\alpha) = 1/384000$. The included `check_toy_example.py` verifies these identities and the plotted values on CPU.

The accepted branch remains proxy-infeasible. Branch 3 wins and is flat near the initial point, making the envelope locally flat; the proposal improves it in this construction. This illustrates refinement before selection, without a general escape or physical-arrival guarantee.

B Experimental protocol and implementation

B.1 Cases, observations and native success

Eligible episodes contain more than 75 actions; whole episodes used in earlier planning experiments are excluded. NumPy SeedSequence (base 20260919, task and split indices) samples 16 development and 64 disjoint test episodes without replacement, then one uniform start from $\{0, \dots, L - 76\}$ per length-L episode. Cases are distinct episodes, not fixed-stride windows. Goals are 50 recorded actions later. A shared manifest fixes cases and simulator seeds.

Planners receive only current/goal RGB images at 224×224, scaled to $[0, 1]$ and normalized by means $(.485, .456, .406)$ and standard deviations $(.229, .224, .225)$. State is used for restoration and native success, not planning. The initial image is recorded; later images are simulated. Restored PushT renders differ from recorded frames, a shared evaluator limitation. Native success differs from latent tolerance.

Table 5: Frozen LeWorldModel checkpoints and native success predicates. All checkpoint identifiers have prefix `quentinll/lewm-`; action dimension is d_a.

Task	**Suffix**	d_a	**Native success**
TwoRoom	tworooms	2	Agent-to-target Euclidean distance < 16.
PushT	pusht	2	Joint planar-coordinate error of pusher and block < 20, and wrapped block-angle error $< \pi/9$.
Reacher	reacher	2	Each absolute joint-position error < 0.05 rad, without angle wrapping.

Models remain frozen without auxiliary training. Physical actions lie in $[-1, 1]$; standardization and transformed bounds use per-coordinate statistics fitted to all finite dataset actions, including data beyond development. Test episodes are held out from planner development, but world-model pretraining membership is unknown. The artifact records normalization vectors and checkpoint/source hashes.

B.2 Solver objectives and settings

Branch-Prox uses Eq. (2) with a summed, unaveraged proximal norm over HBd_a unit-box coordinates. Standardized Gaussian initial plans (first candidate zero) are mapped and clipped to the unit box. Defaults are $(H, B, N, K, \rho, \epsilon) = (8, 5, 16, 20, 30, .05)$. Each plan starts at $\alpha = 1$: Armijo coefficient 10^{-4}, 12 trials, rejection halving, accepted next-step doubling capped at 10^4, and 28 dual bisections on $[0, \rho]$. Exhaustion retains the base plan. The hinge is linear; no temperature is active. Eight endpoint gradients per update give 160 reverse invocations per solve.

CEM, iCEM, MPPI* and AdamW minimize terminal cost $T(a) = \|z_H(a) - z_g\|_2^2$, without normalization by S or a time term. GRASP uses virtual states $v_0 = z_0, v_H = z_g$ and one-step predictions $\hat{z}_{h+1} = P(v_h, a_h)$, summing $\mathrm{MSE}(\hat{z}_{h+1}, v_{h+1}) + \mathrm{MSE}(\hat{z}_{h+1}, z_g)$ over h; synchronization minimizes native full-rollout terminal MSE. Table 6 distinguishes objectives, projection and warm starts.

Table 6: Optimized costs and action handling. External methods use standardized coordinates; Branch-Prox uses the equivalent unit box. "Shifted" moves the previous MPC plan one block and zero-pads its tail.

Method	Optimized cost	Action handling and returned plan	MPC start
CEM	Terminal T	Clip samples before evaluation; final elite mean	Shifted
iCEM	Terminal T	Clip samples before evaluation; final distribution mean	Shifted
MPPI*	Terminal T	Clip samples before evaluation; elite-weighted mean	Shifted
AdamW	Terminal T	Unconstrained updates; project and rescore candidates by T	Shifted
GRASP	Trajectory MSE; terminal sync	Unconstrained updates; project one final plan, no reranking	Shifted
Branch-Prox	Arrival envelope F	Box-feasible updates; rank plans by nonlinear F	Fresh plans

Table 7: External fast/reference presets. Population means samples for CEM/iCEM/MPPI* and initial plans for AdamW and GRASP. Reference comparisons use the second value at common $H = 8$; these settings are not equal computational budgets.

Method	Population	Updates	Elites
CEM / iCEM / MPPI*	128 / 300	10 / 30	16 / 30
AdamW	16 / 100	30 / 30	–
GRASP	1 / 1	200 / 750	–

CEM uses Gaussian standard deviation 1. iCEM uses colored-noise exponent 2, momentum .1 and five reevaluated previous elites within each solve; population stays fixed, so this implementation omits features of original iCEM. MPPI* uses standard deviation 1 and temperature .5 over only the best elites. AdamW uses learning rate .1 and decay .01 without gradient clipping/noise. GRASP uses state/action learning rates .1/.005, state noise .005 and ten native GD synchronization steps (rate .05) every 50 updates; independent one-step predictions are batched.

GD/GRASP may query out-of-box actions before final projection; GD rescores projected candidates, while GRASP returns one projected plan. Diagnostics record GD bound violations, not projection-induced cost changes. External warm starts shift actions only: optimizer states/variances reset, and iCEM elite reuse stays within a solve. Secondary selection crosses both presets with $H \in \{3, 5, 8\}$, ranking 16 development episodes (seed 42) by steps, success, exclusive median latency and name; winners appear in Table 10.

B.3 Statistics, timing and reproduction

The independent unit is an episode. Seeds 42/43/44 are averaged within episode before 10,000 percentile episode-bootstrap resamples (seed 9192026); paired effects match episodes/seeds. Failures receive the action budget, initial successes zero; missing runs are not failures. Budgeted steps combine arrival and failure. Point intervals at 100% do not imply certain population success.

Primary p-values use Holm adjustment over the original full-study family of 20 two-sided paired sign-flip tests, with 100,000 draws and plus-one correction (minimum attainable $p = 1/100001$). Sign flipping assumes a sign-symmetric paired-difference null: conditional on magnitudes, signs are exchangeable. Intervals are marginal; selected-baseline, component, sensitivity and compute analyses are secondary. The complete original analyses remain in the artifact.

The system uses RTX 5090, PyTorch 2.8.0+cu128 and MuJoCo 3.8.0, with TF32 disabled and two CPU threads per worker; complete environment versions accompany the artifact. Exclusive latency times synchronized encoding and solver overhead, excluding preprocessing, simulation and loading. Three historical development observations each receive one warmup and three timed solves without

MPC warm starts, yielding nine-call median/p90 calibration. Concurrent durations are not exclusive latency.

Protocols, hashes, actions and outcomes support recomputation; data fingerprints cover normalization, episode metadata and selected observations, not every image byte. The source package's `evidence_index.json` identifies separate execution records.

Reproduction scope. Native replay checks saved physical actions in the recorded simulation environment. End-to-end replanning and cross-machine reproducibility are separate properties. Replay requires the external datasets and simulation dependencies; the artifact retains the full reproduction diagnostics.

C Additional results and mechanism checks

C.1 Native success and primary tests

Table 8: Native success (%), mean [marginal 95% episode-bootstrap interval], on all 64 episodes with three seeds averaged within episode. Bold marks the highest mean.

Method	TwoRoom	PushT	Reacher
CEM	33.85 [23.44, 44.79]	19.27 [11.98, 27.08]	33.33 [25.52, 41.15]
iCEM	37.50 [26.56, 48.44]	9.37 [4.17, 15.62]	35.42 [27.60, 43.23]
MPPI*	72.40 [62.50, 81.77]	14.06 [7.81, 20.83]	41.67 [34.38, 48.96]
AdamW	48.44 [38.54, 58.33]	22.40 [15.10, 29.69]	35.42 [27.60, 43.23]
GRASP	23.44 [15.10, 32.81]	31.25 [20.83, 42.19]	75.52 [65.10, 85.42]
Branch-Prox	**80.21** [71.35, 88.02]	**44.79** [34.38, 55.21]	**100.00** [100.00, 100.00]

Without its initial success, 63 TwoRoom episodes remain; Branch-Prox success is 79.89 [70.90, 87.83]%. PushT/Reacher have no initial successes. Full noninitial results are archived.

Table 9: Primary step tests: two-sided p-values with Holm adjustment over the original full-study family of 20 comparisons. Values below 0.05 favour Branch-Prox; raw values are archived.

Comparator	TwoRoom	PushT	Reacher
CEM	0.0002	0.0002	0.0002
iCEM	0.0002	0.0002	0.0002
MPPI*	0.0002	0.0002	0.0002
AdamW	0.0002	0.0002	0.0002
GRASP	0.0002	0.1022	0.0002

C.2 Development-selected configurations

Unlike the common-$H = 8$ reference, Table 2 uses these development-selected settings. Full paired records are archived.

C.3 Component success and branch-selection mechanism

The 32-episode component cohorts have no initial successes. Their matched Branch-Prox references are 30.42/65.14/14.05 steps and 85.42/45.83/100% success on TwoRoom/PushT/Reacher, distinct from the 64-episode means.

Table 10: Every external task/family winner, chosen on 16 development episodes, seed 42, before the 64-episode, three-seed test. Branch-Prox retains H=8, 16 plans and 20 updates.

Method	TwoRoom	PushT	Reacher
CEM	H=3 / fast	H=3 / fast	H=3 / reference
iCEM	H=5 / reference	H=3 / fast	H=3 / fast
MPPI*	H=3 / fast	H=3 / fast	H=3 / fast
AdamW	H=5 / reference	H=3 / reference	H=3 / fast
GRASP	H=5 / reference	H=5 / reference	H=5 / reference

Table 11: Component controls: full Branch-Prox minus control in steps, first 32 episodes and three seeds, with marginal secondary 95% paired intervals. Negative favours full; Fixed step removes backtracking and step adaptation.

Method	TwoRoom	PushT	Reacher
Fixed H	-47.88 [-56.26, -38.91]	-31.59 [-44.25, -19.69]	-74.71 [-79.07, -70.20]
Current branch	-19.33 [-29.10, -10.43]	-2.81 [-11.51, 5.97]	-2.69 [-6.23, 0.05]
No time	-20.01 [-26.82, -13.40]	-22.32 [-34.00, -11.41]	-56.68 [-64.31, -48.34]
Fixed step	+3.29 [-1.96, 8.15]	-18.34 [-28.34, -9.16]	+3.00 [1.23, 4.89]

Table 12: Full Branch-Prox minus control in success percentage points [marginal secondary 95% paired interval]; positive favours full. Same 32 episodes and three seeds as Table 11.

Method	TwoRoom	PushT	Reacher
Fixed H	42.71 [27.08, 58.33]	36.46 [21.88, 52.08]	67.71 [57.29, 78.12]
Current branch	23.96 [13.54, 35.42]	5.21 [-7.29, 18.75]	2.08 [0.00, 6.25]
No time	10.42 [1.04, 20.83]	7.29 [-6.25, 20.83]	40.62 [29.17, 52.08]
Fixed step	-2.08 [-9.38, 5.21]	25.00 [13.54, 37.50]	0.00 [0.00, 0.00]

All controls compute eight endpoint derivatives per update; differing stopping times also change total work.

Observed branch changes. The first earlier-branch witness in fixed traversal order (PushT episode 15715, start 37) has current winner $h = 6$ at 12.671239. Branch $h = 1$ starts at 22.112217, attains local value 1.283364, and yields accepted nonlinear envelope 10.418184. Instrumentation preserves the returned actions. This within-solve diagnostic illustrates the branch-selection mechanism.

Floating-point descent. A Fixed H/Reacher Armijo deficit reaches 1.42×10^{-5}: replay confirms that subtracting the requested decrease rounds to the same float32 objective. The archived exception records the finite-precision departure from the exact-arithmetic acceptance condition.

D SAME-OBJECTIVE PROJECTED OPTIMIZER

Projected Adam uses $(\beta_1, \beta_2) = (0.9, 0.999)$, numerical epsilon 10^{-8}, zero weight decay and 20 updates, with no warm start or iterate archive. It differentiates the exact envelope (lowest-index branch at a tie), projects every update and ranks final plans by the same envelope. The selected learning rates are 0.1/0.01/0.1 for TwoRoom/PushT/Reacher. All three rates complete all 16 development episodes

per task with seed 42; choices are frozen before the new tests. The test subset is the predeclared first 16 original episodes, with seeds 42/43/44 and the unchanged 100-action budget.

Table 13: Native success in the same-objective control. Means [marginal 95% intervals]; Δ is Branch-Prox minus Adam in percentage points.

Task	Branch-Prox (%)	Proj. Adam (%)	Δ success (pp)
TwoRoom	93.8 [83.3,100.0]	66.7 [45.8,85.4]	+27.1 [10.4,45.8]
PushT	50.0 [29.2,70.8]	43.8 [22.9,66.7]	+6.2 [−8.3,22.9]
Reacher	100.0 [100.0,100.0]	97.9 [93.8,100.0]	+2.1 [0.0,6.2]

Inference averages seeds within episode, uses 10,000 episode-bootstrap draws and all 2^{16} paired sign flips, and applies Holm over the original full-study family of four step tests, separately from the primary family. This secondary control was designed after the original test outcomes were available; its learning rates were selected on development data. Concurrent timings measure shared-GPU execution, rather than exclusive latency. The source bundle contains native replay and initialization-hash audits, the frozen protocol, selection, manifests, outcome CSVs and `verify_projected_adam.py` for independent CPU reconstruction.

E DINO-WM WALL PROTOCOL AND RESULTS

We evaluate the released frozen DINO-WM Wall checkpoint without additional training (Zhou et al., 2025).

Model and inputs. We use the `wall_single` checkpoint from `showstarpro/DINO-WM-osf`, revision `ddbe627fa09292a9d460d5efde62e929afd9f042`. The frozen DINOv2 ViT-S/14 encoder provides patch features from 224×224 RGB observations; the released six-layer, 16-head predictor uses one observation of history. Current and goal inputs each include RGB and two-dimensional position proprioception, shared by every planner. Images retain the released resize, center crop and channel normalization; actions and proprioception retain the released loader's training-dataset mean and standard deviation. The latent matching cost sums mean squared visual-feature and proprioceptive-feature errors with equal weight. Physical actions are relative displacements, bounded componentwise by $[-1.8, 1.8]$. Checkpoint, encoder and normalization hashes are recorded with the execution artifact.

Cases and native execution. The test uses 50 independent start–goal draws from the native cross-wall sampler, with environment seeds 620200000 through 620200049, wall location 32 and door location 30. Eight separate development draws serve only for configuration selection. These are independent start–goal pairs, not regularly spaced windows from a recorded trajectory. Every method uses planner seed 42, $H = 5$ model blocks and five physical actions per block. MPC executes at most one block before replanning and checks native success after every physical action: Euclidean position error strictly below 4.5 native coordinate units. Execution stops at first success or 100 actions; failures contribute 100 capped steps. All 250 method–case trajectories passed native replay. One planner seed supports episode uncertainty, not planner-seed variability.

Frozen configurations. Branch-Prox uses 32 starts, 20 updates, $\rho = 30$ and $\epsilon = .05$. CEM, AdamW, iCEM and MPPI* retain their common-$H = 5$ reference configurations and respective objectives. GRASP's full development/test matrix was unavailable. Methods share the horizon and physical budget; their computational budgets differ. The source package retains `data/dino_wm_results.json`; execution records and hashes accompany the evaluation artifact.

Paired effects. Branch-Prox minus CEM, AdamW, iCEM and MPPI* is respectively $-21.80\,[-28.32, -15.86]$, $-21.96\,[-28.30, -15.98]$, $-18.14\,[-24.00, -12.80]$ and $-30.62\,[-38.14, -23.54]$ capped steps; all four Holm-adjusted values are 8.0×10^{-5}. Holm adjustment retains the original full-study family of eight paired step tests. Comparisons pair methods

Table 14: DINO-WM Wall: 50 independent start–goal pairs, seed 42. Entries are mean [marginal 95% episode-bootstrap interval]. Time is cumulative planner seconds per episode, including encoding and all solves under concurrent workers; it is not exclusive solve latency.

Method	Steps ↓	Success (%) ↑	Time (s/episode)
CEM	41.10 [35.24,47.82]	92.0 [84.0,98.0]	36.9
iCEM	37.44 [32.44,43.30]	96.0 [90.0,100.0]	34.8
MPPI*	49.92 [43.40,57.02]	88.0 [78.0,96.0]	43.8
AdamW	41.26 [35.42,47.58]	94.0 [86.0,100.0]	40.6
Branch-Prox	**19.30** [16.64,22.22]	**100.0** [100.0,100.0]	138.5

on the same native cases within each task. Intervals use 10,000 episode-bootstrap draws, and two-sided sign-flip tests use 100,000 draws.

F ARRIVAL TIMING AND RELIABILITY DIAGNOSTICS

For this post-hoc review analysis, we fixed common cutoffs of 25/50/100 actions for every method, requiring native success; the execution budget remains 100. For common-success case–seed pairs, arrival differences are averaged first within episode, then across episodes with selected pairs. These descriptive means condition on both outcomes and complement the unconditional capped-step metric; they do not estimate a population or causal effect. Initial successes remain included. Portable records and a verification script reproduce all entries, including noninitial and pair-weighted summaries.

Table 15: Fixed-deadline native success (%) and conditional native-arrival differences. Deadline triples are Success@25/50/100. n_c/E_c counts common-success case–seed pairs/episodes; Δ_c is the episode-balanced arrival difference (BP minus comparator), negative favouring BP. LeWorldModel reference uses 64 episodes and three seeds; its component control uses 32 and three. DINO-WM uses 50 Wall cases and seed 42.

Comparator / task	**BP: 25/50/100**	**Comparator: 25/50/100**	n_c/E_c	Δ_c
LeWorldModel: TwoRoom				
CEM	66.1/78.6/80.2	9.9/25.5/33.9	65/25	-29.74
iCEM	66.1/78.6/80.2	8.3/22.9/37.5	71/29	-31.41
MPPI*	66.1/78.6/80.2	11.5/38.5/72.4	130/50	-32.97
AdamW (terminal)	66.1/78.6/80.2	14.1/30.7/48.4	90/43	-28.13
GRASP	66.1/78.6/80.2	13.5/17.7/23.4	45/22	-22.92
LeWorldModel: PushT				
CEM	34.4/39.6/44.8	4.2/13.5/19.3	26/16	-10.86
iCEM	34.4/39.6/44.8	1.6/5.2/9.4	12/9	-23.33
MPPI*	34.4/39.6/44.8	1.6/6.8/14.1	16/9	-28.96
AdamW (terminal)	34.4/39.6/44.8	1.0/13.0/22.4	27/19	-27.07
GRASP	34.4/39.6/44.8	26.0/28.6/31.2	45/19	-9.41
LeWorldModel: Reacher				
CEM	91.1/99.0/100.0	5.2/19.3/33.3	64/39	-33.87
iCEM	91.1/99.0/100.0	7.3/18.8/35.4	68/41	-37.32
MPPI*	91.1/99.0/100.0	9.9/19.3/41.7	80/51	-41.03
AdamW (terminal)	91.1/99.0/100.0	8.3/17.2/35.4	68/44	-35.98
GRASP	91.1/99.0/100.0	68.2/74.0/75.5	145/49	-2.80
DINO-WM: Wall				
CEM	84.0/100.0/100.0	26.0/76.0/92.0	46/46	-16.91
iCEM	84.0/100.0/100.0	20.0/86.0/96.0	48/48	-15.92
MPPI*	84.0/100.0/100.0	14.0/64.0/88.0	44/44	-23.77
AdamW (terminal)	84.0/100.0/100.0	20.0/78.0/94.0	47/47	-18.83
Current-branch control: Full BP versus Current branch				
TwoRoom	65.6/83.3/85.4	47.9/58.3/61.5	59/24	+0.04
PushT	35.4/39.6/45.8	33.3/38.5/40.6	28/12	-1.78
Reacher	88.5/99.0/100.0	86.5/95.8/97.9	94/32	-0.95

G COMPUTATIONAL COST AND SENSITIVITY

G.1 PHYSICAL ACTIONS VERSUS PLANNING TIME

The common-$H = 8$ sweep uses the first 16 episodes and three seeds, with no initial successes on TwoRoom/PushT/Reacher. Figure 5 pairs their outcomes with the nine-call exclusive latency calibration (Appendix B). The stronger development-selected horizons belong to Table 2.

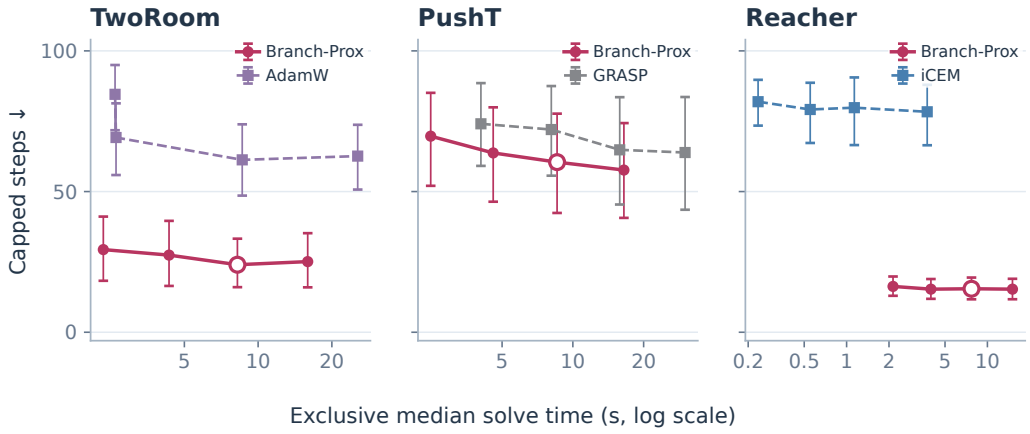


Figure 5: **Common-horizon budget sweep ($H = 8$).** Physical steps versus separately measured exclusive solve time. Branch-Prox is rose; open markers identify its default budget. Points use 16 episodes and three seeds; bars are marginal episode-bootstrap intervals. Lines connect different compute budgets at the fixed horizon.

Per-budget outcomes, paired effects, p90 timings and operation counts remain in the artifact. Reverse invocations and forward transitions are distinct operations.

G.2 ONE-FACTOR SENSITIVITY

Table 16 lists all nine alternatives. Lower ρ helps Reacher but hurts TwoRoom; five updates worsen PushT by $9.23\,[2.29, 18.21]$ steps. The 27 task–setting effects are included in the source bundle. This one-factor design evaluates individual parameter changes around the reference setting; interactions remain untested.

Table 16: One-factor grid: three tasks, first 16 episodes, three seeds; population 16. Each alternative changes one parameter.

	Penalty ρ	Tolerance ϵ	Window H	Updates K
Reference	30	0.05	8	20
Alternatives	10, 100	0.025, 0.1	4, 12	5, 10, 40