HETEROGENEOUS RECURRENCE FOR PARTIALLY OBSERVED CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents for partially observed control usually rely on a single kind of recurrence. Yet the best memory depends on the hidden part of the state. When the hidden quantity is a derivative of the observation, as a velocity is of observed positions, the masked system is observable. Each new observation then carries a correction signal, and a recurrence can recover the hidden quantity by reading the current observation beside its memory. When the hidden quantity is an integral, no observation corrects its initial error, and a plain gated recurrence does as well. One agent may face both cases. In this paper, we therefore propose **DuoCell**, a heterogeneous recurrence for partially observed control. It splits one fixed state budget between a structured prior–innovation path for the first case and a GRU for the second. On masked MuJoCo with velocities hidden, DuoCell leads a width-matched GRU by $+0.52$ normalized IQM with fewer parameters. It also beats eleven other baselines, including an LSTM and a tuned TransformerXL, and its lead over the GRU baselines holds at five times the training budget. Because the regime follows from the sensor layout, it can be read off before training. We fixed this prediction before running eight unseen environments, and it held in both regimes. Where the criterion expected a benefit, the structured recurrence gained $+0.33$ normalized IQM over the GRU.

1 INTRODUCTION

An agent that cannot observe its full state must reconstruct it from history. Recurrent agents usually get one kind of recurrence for this, most often a gated one such as a GRU. Memory architectures are then compared by their average return over a benchmark (Ni et al., 2022; Morad et al., 2023a). This comparison presumes that one architecture is simply better than another.

We argue that the better architecture depends on the hidden part of the state. Consider a body whose positions are observed and whose velocities are not. The velocity is the rate of change of the observed positions, so every new observation carries information about it. A recurrence that predicts the next observation and corrects itself by the error can recover the velocity. Now reverse the mask. The position is the running sum of the observed velocities. No observation reveals it, and no error signal points to it. Only dead reckoning remains, and any recurrence can perform it. Linear systems theory makes this dichotomy exact (Sec. 3.1): the first masked system is observable, and the second is not even detectable. For a mechanical system the first case follows from kinematics alone, whatever the dynamics, so it can be read off a sensor layout before any training. The second case also depends on the dynamics, which we measure.

This view explains a pattern that benchmark averages hide. On masked MuJoCo (Tao et al., 2025), a structured prior–innovation recurrence beats a GRU when positions are observed and loses its edge when velocities are observed. An average over both halves reports a modest gap and conceals the reversal. The view also makes a testable prediction. We assigned eight further environments to a regime from their observation layouts before running any of them. Pooled over each regime, the structured recurrence leads where it is predicted to and trails where it is not, and a GRU given the same reread of the current observation does the same (Fig. 3).

The criterion says when a reread of the current observation pays, and it leaves open which carry to pair it with. Often the sensor layout is not even known, and a platform may carry both kinds of

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.

channel. In this paper, we therefore propose DUOCELL, a heterogeneous recurrence for partially observed control. It runs both constructions side by side under one fixed state budget (Fig. 1). Half of the budget follows a structured prior–innovation update and half follows a GRU. Both read a shared encoder, and the policy receives their normalized concatenation. DUOCELL has 8% fewer parameters than a width-matched GRU, and it trains in $0.59\times$ the wall-clock of the structured recurrence at full width.

Our contributions are:

- **An observability criterion.** Observability of the masked system (Proposition 1) determines whether reading the current observation beside a recurrent memory can help. On eight held-out environments both registered predictions hold, including a reversal in the integral regime, for a structured prior–innovation recurrence and for a GRU given the same reread. A policy-free measurement of how much of the hidden state the observation history reveals tracks the advantage across fourteen environments.
- **DUOCELL, a heterogeneous recurrence.** It composes both constructions under one budget. On masked MuJoCo it improves on the GRU in the derivative regime and on the structured recurrence in the integral regime; comparisons with the other parent remain uncertain.
- **Matched-baseline evaluation.** In the derivative regime of masked MuJoCo DUOCELL leads every baseline we ran, including a GRU with 70% more parameters, an LSTM, a GRU with the same reread and a tuned TransformerXL on held-out seeds (Table 1). The lead over the GRU baselines holds at five times the training budget and for the deployed deterministic controller.
- **Component analysis.** A factorial ablation and training controls locate the gain in the structured carry and its gated reread; at this budget the trained cells do not require online correction. Path lesions show that a trained DUOCELL relies on the construction its regime favours.

2 RELATED WORK

Memory for partially observed control. Gated recurrence is the strongest simple memory on POMDP benchmarks such as POPGym (Ni et al., 2022; Morad et al., 2023a). On masked locomotion a transformer outperforms it (Appendix E.4.1): when the hidden quantity is a derivative, attention can difference the exact recent observations that a GRU must first compress into its state. We use the POBAX benchmark (Tao et al., 2025), which also evaluates the λ-discrepancy auxiliary (Allen et al., 2024). Other memories for RL include stabilized transformers (Parisotto et al., 2020), state-space layers (Smith et al., 2023; Lu et al., 2023; Luis et al., 2025) and fast and forgetful memory (Morad et al., 2023b). Our baselines cover the first two families.

Structured and predictive recurrent states. Recurrent states built from predictive statistics include PSRNNs (Downey et al., 2017), RPSP networks (Hefny et al., 2018) and Predictive-State Decoders (Venkatraman et al., 2017). Learned Kalman-style estimators (Revach et al., 2022; Mortada et al., 2025) and learned belief states (Chen et al., 2022; Nguyen et al., 2020) keep a filter's structure and learn its parts. Our structured path borrows this scaffold without belief supervision. We use it for half of the state only, and we tie its reread of the current observation to the observability of the task rather than treating the path as a general-purpose filter.

Observation shortcuts. Passing the current observation embedding to the policy alongside the recurrent state is common practice (Ni et al., 2022). Morad et al. (2023b) gate this shortcut with a learned sigmoid, close to our readout (Eq. 5). We credit the construction to them. We show that it carries most of the gain inside a structured recurrence, that the same readout over a gated carry does not reproduce that gain in domain, and that out of domain it does (Secs. 4.3 and 4.4).

Multi-part recurrent states. Influence-aware memory runs a feedforward path beside a recurrence that reads only the variables influencing the hidden state (Suau et al., 2025). Clockwork RNNs and Recurrent Independent Mechanisms compose modules of one family by timescale or routing (Koutník et al., 2014; Goyal et al., 2021). The RSSM pairs a deterministic carry with a stochastic latent (Hafner et al., 2019; 2020). Our two paths differ in their update rule, run unconditionally and

meet without a router. The point is a complementary pair, and we test that it is this pair by replacing the structured half with state-space and attention paths. Our auxiliary is the exact-predictor limit of latent self-prediction (Schwarzer et al., 2021; Fujimoto et al., 2023; Ni et al., 2024; Tang et al., 2023; Khetarpal et al., 2024; Voelcker et al., 2024; Tian et al., 2021).

3 METHOD

We work in the standard recurrent actor–critic setting. An encoder maps the observation to an embedding $e_t = \mathrm{Enc}(o_t)$. A recurrent cell updates a carried state from (e_t, a_{t-1}), and actor and critic heads read its output s_t. This section derives the choice of recurrence from the observation map, then builds a cell that carries both choices.

3.1 OBSERVABILITY OF MASKED SYSTEMS

Under partial observability the recurrent state has one job: to estimate the part of the state that the observation leaves out. The classical estimator does this in two steps. It *predicts* the next state from the dynamics and the last action. It then *corrects* the prediction by the innovation, the difference between the new observation and the predicted one. Whether this correction can recover the hidden part depends on how the hidden part couples to the visible one. The point-mass limit makes this exact.

Proposition 1 (Observability dichotomy of masking)**.** *Let $s = (x, v) \in \mathbb{R}^{2n}$ follow the discretized double integrator $x_{t+1} = x_t + \Delta t\, v_t$, $v_{t+1} = v_t + \Delta t\, a_t + w_t$, observed as $y_t = C s_t + \eta_t$. The initial state, the process noise and the observation noise are independent and Gaussian, with $\mathrm{Cov}(x_0) = \Sigma_0$, $\mathrm{Cov}(v_0) = \Sigma_v \succ 0$ and $\mathrm{Cov}(\eta_t) = R$, and each action may depend on past observations. (i) With $C_P = [\, I \;\; 0\,]$ (positions observed), (A, C_P) is observable. For $R \succ 0$ the Kalman filter's error covariance on the hidden block v stays bounded, and the innovation depends on the hidden-block error with gain $\Delta t\, I$. (ii) With $C_V = [\, 0 \;\; I\,]$ (velocities observed), the unobservable subspace is $\{(c, 0)\}$ with unit eigenvalue, so (A, C_V) is not detectable. The error of every causal estimator $\hat{x}_t$ of the hidden block never falls below the prior uncertainty, $\mathbb{E}\,\|x_t - \hat{x}_t\|^2 \geq \mathrm{tr}\,\Sigma_0$. If $R \succ 0$, it also grows linearly in t:*

$$\mathbb{E}\,\|x_t - \hat{x}_t\|^2 \;\geq\; \mathrm{tr}\,\Sigma_0 + \Delta t^2\, t^2\, \mathrm{tr}\big[(\Sigma_v^{-1} + (t{+}1)R^{-1})^{-1}\big]. \tag{1}$$

Appendix B gives the proof. The result is standard linear systems theory, and we claim no novelty for it. It supplies a criterion fixed before any training. In case (i) the hidden variables drive the one-step change of the observation. A state that is corrected by its own prediction error therefore receives, at every step, a signal about the variable the policy lacks. In case (ii) the hidden positions do not enter the prediction of the observed velocities. No correction signal exists, and only dead reckoning remains. Dead reckoning needs no special structure, since any recurrence can integrate its input.

We call case (i) the *derivative regime*, because the hidden variables are derivatives of the observed ones. We call case (ii) the *integral regime*.

Beyond the point mass. The double integrator isolates the observation map, and a MuJoCo body adds dynamics. The derivative regime survives any dynamics. Velocity is the time derivative of the observed configuration, so a difference of two consecutive observations approximates it; this is kinematics, not a property of the forces. The integral regime does not survive unchanged. An observed velocity reveals the configuration only through forces that depend on it, such as gravity and contact. Coordinates on which no force depends stay unobservable, while the others can be partly recovered. The mask therefore fixes the derivative regime, and in the integral regime it states the worst case, which the dynamics can relax. Appendix C measures this directly on every environment.

The rest of this section builds a recurrence for the derivative regime and then combines it with a GRU for tasks whose regime is unknown.

3.2 STRUCTURED PRIOR–INNOVATION RECURRENCE

The criterion suggests a recurrence that carries a prediction and corrects it by its error. The structured path updates its state b_t in three steps. First, a prior advances the previous state under the action,

with a learned retention r_t (Eq. 2). Second, the innovation ε_t compares the observed embedding with the prior's prediction, and a learned gain writes part of this error into the state (Eqs. 3–4). This is the Kalman innovation computed in a learned feature space; because W_p and W_o are learned, it carries none of the whiteness guarantees of the exact filter. Third, the readout re-admits the current embedding through a learned gate (Eq. 5):

$$[c_t;\, \rho_t] = W_f\,[\mathrm{LN}(b_{t-1});\, a_{t-1}], \qquad r_t = r_{\min} + (r_{\max} - r_{\min})\,\sigma(\rho_t), \tag{2}$$

$$\bar{b}_t = r_t \odot b_{t-1} + (1 - r_t) \odot \tanh(c_t), \qquad \varepsilon_t = W_p e_t - W_o\,\bar{b}_t, \tag{3}$$

$$[\delta_t;\, g_t] = \mathrm{MLP}\big([\varepsilon_t;\, \bar{b}_t;\, a_{t-1}]\big), \qquad b_t = \mathrm{LN}\big(\bar{b}_t + \sigma(g_t) \odot \tanh(\delta_t)\big), \tag{4}$$

$$u_t = W_u\,[\mathrm{LN}(b_t);\, \mathrm{LN}(W_p e_t);\, \varepsilon_t], \qquad b_t^{\mathrm{out}} = \mathrm{LN}\big(\mathrm{LN}(b_t) + \sigma(u_t) \odot \mathrm{LN}(W_p e_t)\big). \tag{5}$$

Here W_p projects the shared embedding to the path width, and LayerNorm keeps the scale of the carry fixed so that innovations stay comparable over time. The bound $r_{\max} < 1$ favours slow accumulation.

Connection to observers. Hold the gates constant and keep $\tanh$ in its linear range, as the 0.1-scale initialization arranges. Eqs. 2–4 then reduce to a Luenberger observer, $b_t = \bar{b}_t + K\varepsilon_t$. The function class of the structured path therefore contains the steady-state filter of case (i) of Proposition 1. A GRU has no such template. A reread of the current observation gives either carry the difference signal directly (Sec. 4.3). In the integral regime the template offers nothing: the innovation carries no information about the hidden block, and the structured path falls back on the same dead reckoning a GRU performs.

Gated reread. Early in an episode, and early in training, the carried state is a poor summary of the history. The gated reread lets the policy use the current observation directly until the carry becomes informative, and lets it lean on the carry once it is. In the derivative regime it does more. With the current observation beside a memory of the past ones, the readout can form their difference, which is the hidden velocity; in the integral regime no such difference reveals the hidden block.

3.3 Heterogeneous composition

Proposition 1 says when the current observation corrects the hidden block. When the observation map is not known, neither single construction is safe: the structured path wastes its budget in the integral regime, and a plain GRU lacks the reread in the derivative regime. One platform can also mix both regimes, since a joint encoder reports position while a wheel odometer reports velocity. DuoCell runs both. The second half of the state follows a GRU on the same embedding, and the cell output is the normalized concatenation of the two readouts:

$$h_t = \mathrm{GRU}(h_{t-1}, e_t), \qquad s_t = \mathrm{LN}\big([\, b_t^{\mathrm{out}};\, h_t\,]\big). \tag{6}$$

Algorithm 1 gives the full step. Three design choices follow from the goal.

The total width is fixed. Each path runs at half the budget H, so DuoCell competes at the budget of either single construction. It cannot win by adding capacity.

The encoder is shared. Both paths see the same features, and the composition adds no observation processing of its own.

The readouts are concatenated, not routed. A router would have to infer the regime from the return. Concatenation keeps both readouts available and lets the actor and critic heads weigh them.

Algorithm 1 One DuoCell step

Require: o_t, a_{t-1}, carry (b_{t-1}, h_{t-1})
$e_t \leftarrow \mathrm{Enc}(o_t)$
// Structured path, width $H/2$
$\bar{b}_t \leftarrow \mathrm{prior}(b_{t-1}, a_{t-1})$ Eqs. 2–3
$\varepsilon_t \leftarrow W_p e_t - W_o \bar{b}_t$
$b_t \leftarrow \mathrm{correct}(\bar{b}_t, \varepsilon_t, a_{t-1})$ Eq. 4
$b_t^{\mathrm{out}} \leftarrow \mathrm{reread}(b_t, W_p e_t, \varepsilon_t)$ Eq. 5
// GRU path, width $H/2$
$h_t \leftarrow \mathrm{GRU}(h_{t-1}, e_t)$
// Composition
$s_t \leftarrow \mathrm{LN}([\, b_t^{\mathrm{out}};\, h_t\,])$
return s_t; carry (b_t, h_t)

4 Experiments

We evaluate three claims: DuoCell outperforms matched baselines (Sec. 4.2); the observability criterion predicts, on unseen environments, where reading the current observation beside a memory helps (Sec. 4.3); and the gain has an identifiable source inside the cell (Sec. 4.4).

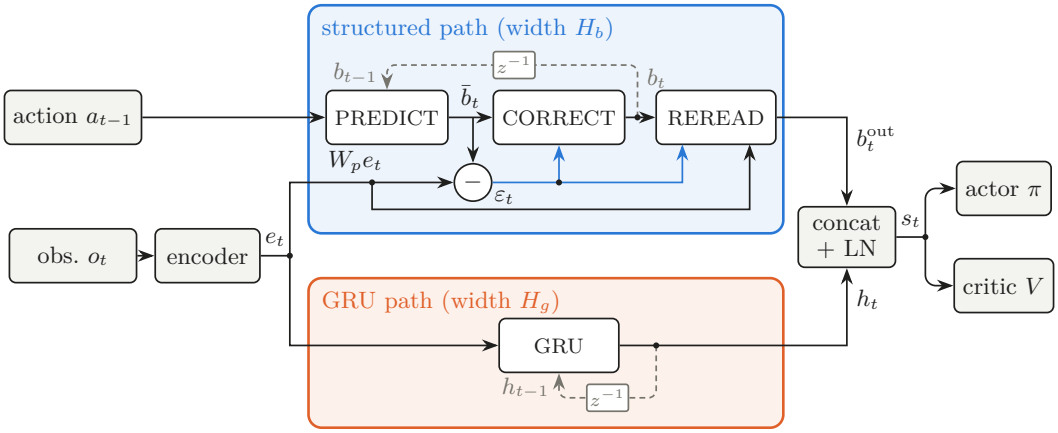


Figure 1: One DUOCELL step. The structured path (blue) predicts its carry, corrects it by the innovation ε_t, and re-reads the current embedding; the GRU path (orange) runs in parallel. Dashed arrows carry each path's state to the next step through a one-step delay z^{-1}.

Table 1: DUOCELL against every baseline, derivative regime, ten paired replications. Positive values favour DUOCELL. *Held-out seeds after a matched learning-rate search.

Baseline	Params	DUOCELL − baseline
GRU, width-matched	2.37M	$+0.520\,[+0.427, +0.598]$
GRU, 70% more parameters	3.70M	$+0.562\,[+0.458, +0.638]$
LSTM, width-matched	1.19M	$+0.562\,[+0.496, +0.623]$
LSTM at width 1024	4.74M	$+0.558\,[+0.464, +0.642]$
two identical GRUs	1.98M	$+0.315\,[+0.228, +0.407]$
GRU + state-space path	1.85M	$+0.416\,[+0.296, +0.517]$
GRU + attention path	1.45M	$+0.381\,[+0.274, +0.468]$
GRU with the same gated reread	3.16M	$+0.282\,[+0.196, +0.364]$
GRU reading $o_t - o_{t-1}$	2.37M	$+0.482\,[+0.395, +0.559]$
structured path alone, with reread	3.43M	$+0.122\,[+0.011, +0.240]$
TransformerXL, tuned*	1.83M	$+0.600\,[+0.416, +0.748]$
TransformerXL, tuned, widened*	2.34M	$+0.625\,[+0.438, +0.779]$

4.1 SETUP

Benchmark. We use the eight masked-MuJoCo environments of POBAX (Tao et al., 2025): four morphologies, each with either the velocities hidden (suffix P, the derivative regime) or the positions hidden (suffix V, the integral regime). All arms use the official recurrent PPO implementation, one million environment steps, and the suite's published per-environment hyperparameters. Every recurrent arm has the same state budget, $H = 512$.

Metric. Following Henderson et al. (2018) and Agarwal et al. (2021), the primary metric is the mean online return over training (AUC). Each comparison is a paired difference over ten replications per arm. We normalize each environment's AUC to $[0, 1]$ over the two arms being compared, report the interquartile mean (IQM) of the difference, and give 95% intervals from a seed-stratified bootstrap. The registered prediction of Sec. 4.3 uses the normalizer fixed in its protocol. Appendix G repeats the headline comparisons in raw return and under other normalizers.

Configurations. The *default* configuration adds a closed-form self-predictive auxiliary on b_t (Appendix A); it has no detectable effect (Sec. 4.4) and every result below is reported without it unless stated. The *tuned* configuration of Table 1 gives DUOCELL and TransformerXL the same per-environment learning-rate search. Appendix H lists every hyperparameter and parameter count.

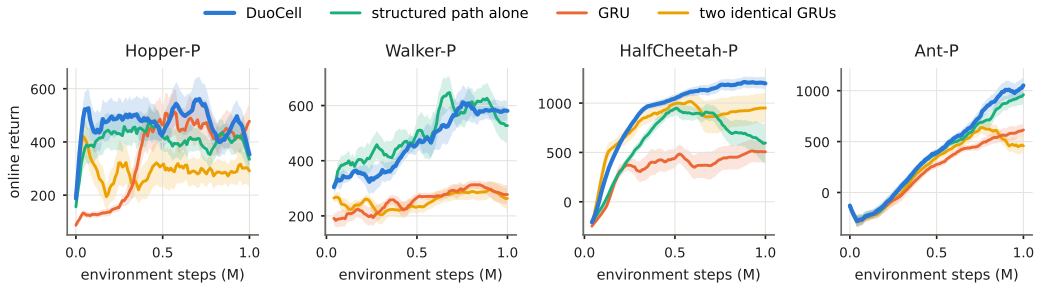


Figure 2: Learning curves in the derivative regime, ten paired replications per arm (mean and standard error).

4.2 Main results

Table 1 compares DuoCell with every baseline in the derivative regime. DuoCell leads all twelve. The rows rule out three alternative explanations. *Capacity:* a GRU with 70% more parameters than DuoCell is no better than the width-matched GRU, so extra parameters do not buy performance in this setting (Appendix E.1). An LSTM does worse than the GRU at either width, by -0.132 $[-0.246, -0.030]$ and -0.160 $[-0.235, -0.083]$. *A second path:* two identical GRUs, and a GRU paired with a state-space or an attention path, all lose. The choice of the two constructions matters, not the number of paths. *Explicit derivatives:* a GRU given the finite difference $o_t - o_{t-1}$ loses by a wide margin, so the recurrent update matters more than access to the derivative. The closest baseline is the structured path alone with the same gated reread. Its gap is small, and it is not detected under the common three-arm normalizer below.

Figure 2 shows the learning curves behind these numbers. DuoCell is at or near the top on all four environments throughout training. On HalfCheetah-P the structured path alone declines after mid-training, while DuoCell keeps its early speed and continues to rise.

The comparison with TransformerXL is the strictest. Both arms receive the same per-environment learning-rate search, the choice is frozen, and both are rerun on fresh seeds. DuoCell leads by 0.60 normalized IQM. The transformer is also the smaller model, so we widened it to 7% more parameters than DuoCell and reran the same confirmation. Its return barely moves, and the gap does not close (Table 1). The advantage is one of sample efficiency: within the same number of environment steps DuoCell reaches higher return, although the transformer trains faster per step on a GPU.

Comparisons across regimes. We compare the composition with both single-path parents on a common per-environment scale pooled over the three arms. In the derivative regime, its difference from the structured parent is $+0.071$ $[-0.008, +0.150]$, while its difference from the GRU is $+0.478$ $[+0.395, +0.551]$. In the integral regime, the corresponding differences are $+0.064$ $[-0.070, +0.215]$ against the GRU and $+0.127$ $[+0.014, +0.241]$ against the structured parent. The intervals support gains over one parent in each regime but do not establish non-inferiority to the other. These regime-level comparisons do not imply a gain on every task (Table 4); their common three-arm normalizer differs from the pairwise normalizers of Table 1.

4.3 Out-of-sample regime prediction

A criterion that only labels the data it came from is a description. To test it as a prediction, we took four morphologies that played no part in any design decision: two balancing tasks and two goal-reaching tasks. We masked each both ways and assigned each variant a regime from its observation layout alone. We committed the assignment and two predictions before the first run. P1: in the derivative regime, the structured recurrence leads the GRU. P2: in the integral regime, it does not.

Both predictions hold (Fig. 3). In the predicted derivative regime the structured recurrence leads by 0.33 normalized IQM. In the predicted integral regime the ordering does not merely flatten; it reverses, to -0.21. Both intervals exclude zero. The reversal is the stronger result. A task with

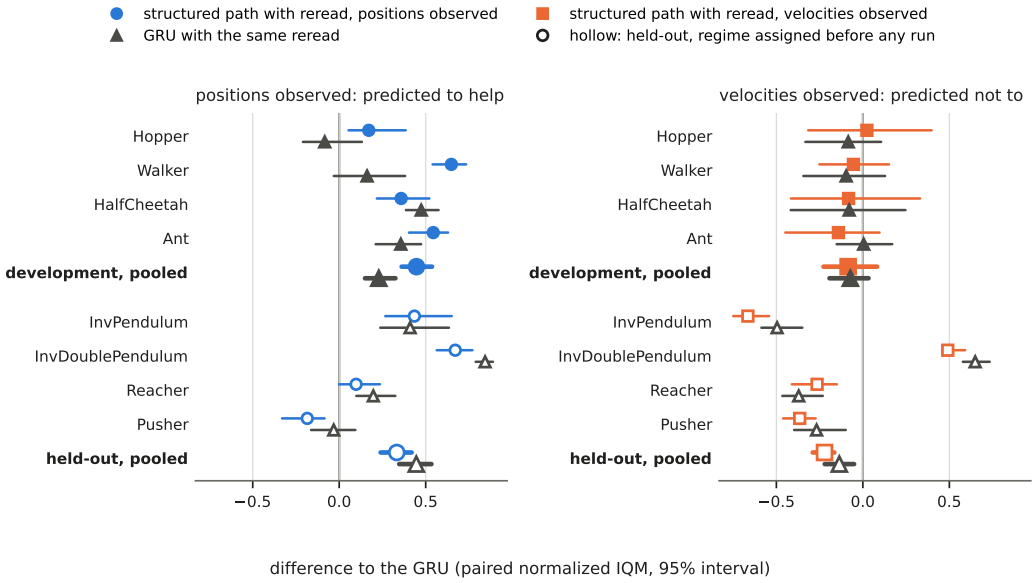


Figure 3: The observation map predicts the sign. Each row is one environment: the paired difference to a GRU of the structured recurrence and of a GRU given the same reread, each normalized over its own two arms, with a 95% interval. Hollow markers are environments whose regime was assigned before they were run.

no hidden state worth recovering would make the two constructions tie. One hypothesis is that a task whose hidden state the observations cannot correct penalizes a construction built to correct it; the GRU with the reread shows a smaller penalty, so the reread itself contributes. The development environments show the same pattern (Fig. 3, filled markers).

What the prediction tests. The structured recurrence differs from the GRU in two ways: its carry and its gated reread of the current observation. Two post-hoc controls separate them. A GRU given the same reread follows the registered pattern, $+0.446$ $[+0.347, +0.533]$ in the predicted derivative regime and -0.136 $[-0.220, -0.051]$ in the predicted integral regime. The prediction therefore concerns reading the current observation beside a memory, as the gated-reread argument of Sec. 3.2 implies, and it holds for either recurrence. The innovation also behaves as predicted. Training the structured recurrence with its innovation zeroed, so that its carry never sees an observation, costs $+0.134$ $[+0.009, +0.253]$ in the predicted derivative regime and nothing detectable in the integral regime, -0.022 $[-0.095, +0.052]$. On the development tasks the same control costs nothing detectable, -0.045 $[-0.160, +0.061]$, as for the full cell (Sec. 4.4), and the innovation-free structured path still leads the GRU with the reread by 0.265 $[+0.182, +0.349]$: in domain the carry's form matters, not its correction. Which carry works best out of sample is a separate question. On the held-out tasks the GRU with the reread leads the structured recurrence in both regimes, by 0.269 $[+0.203, +0.337]$ with positions observed, mostly on InvertedDoublePendulum-P, and by 0.255 with velocities observed; on the development tasks the structured recurrence leads it by 0.218 $[+0.143, +0.286]$.

Per environment, the structured recurrence's sign matches on six of the eight held-out environments, and two contradict it with intervals that exclude zero. Both are informative. On InvertedDoublePendulum-V the structured recurrence leads although velocities are observed. Gravity couples the pole angles into the observed angular accelerations, so the observations do correct part of the hidden block. A direct measurement of recoverability, taken without any policy, finds this: InvertedDoublePendulum-V is the one integral-regime environment whose observation history recovers a substantial part of the hidden block (Appendix C). Across all fourteen measured environments, this recoverability tracks the structured recurrence's advantage (rank correlation $+0.67$). On Pusher-P the structured recurrence trails. Its velocities are recoverable, but the return is dominated

Table 2: Factorial ablation, derivative regime: the cost of removing each component at every setting of the other two.

Component	Measured with	cost of removing it
innovation	auxiliary on, reread on	$+0.101\,[+0.012, +0.185]$
	auxiliary on, reread off	$+0.117\,[+0.050, +0.176]$
	auxiliary off, reread on	$+0.067^{\dagger}\,[-0.013, +0.135]$
	auxiliary off, reread off	$+0.097\,[+0.041, +0.162]$
auxiliary	innovation on, reread on	$-0.024^{\dagger}\,[-0.092, +0.060]$
	innovation on, reread off	$+0.016^{\dagger}\,[-0.064, +0.089]$
	innovation off, reread on	$-0.058^{\dagger}\,[-0.135, +0.029]$
	innovation off, reread off	$-0.004^{\dagger}\,[-0.043, +0.041]$
reread	innovation on, auxiliary on	$+0.203\,[+0.124, +0.287]$
	innovation on, auxiliary off	$+0.242\,[+0.148, +0.322]$
	innovation off, auxiliary on	$+0.219\,[+0.156, +0.290]$
	innovation off, auxiliary off	$+0.272\,[+0.204, +0.347]$

† interval contains zero. No two-way interaction is detected.

by the observed object and goal positions. The criterion states when a filter can reach the hidden block, not whether the task needs it.

4.4 Ablations and analysis

Component ablation. We ablate the innovation, the auxiliary and the gated reread in a full $2 \times 2 \times 2$ factorial, because one-at-a-time removal cannot detect a component that another can replace (Table 2). Removing the reread costs the most, at every setting of the other two components. The innovation matters less and is detected at three of the four settings. The auxiliary has no detectable effect, and no two-way interaction is detected. In domain, a GRU given a gated reread also trails DuoCell (Appendix E.1). Together these results support retaining the reread in this architecture; they do not uniquely identify a structured-carry–reread interaction, since the factorial does not vary the recurrence type.

Inference-time lesion. Setting the innovation input to zero in trained auxiliary-free cells reduces return by an average of 4.2% across the derivative tasks (Appendix E.3.1). This intervention retains the correction network, which still reads the prior and action. The factorial's training-time removal instead disables that correction and changes the readout inputs. The two interventions therefore do not establish that innovation matters only during learning. A training control makes the point sharper. The innovation is the structured carry's only route to the observation, so a cell trained with the innovation zeroed has a carry that never sees an observation: it is driven by the action alone and read out beside the current embedding. This cell shows no detected difference from the full one, $-0.038\,[-0.174, +0.098]$ (Appendix E.1). At this budget the gain therefore does not require online correction; it comes from the structured carry together with the gated reread. The predict–correct derivation of Sec. 3.1 explains when a structured path can help; it does not describe what these trained networks compute.

Path lesion. Zeroing one half of the readout at evaluation shows which path a trained DuoCell relies on (Appendix E.3.2). In the derivative regime the structured path alone keeps most of the return and the GRU alone collapses. In the integral regime the two share the load, and the GRU matters more. The trained cell leans on its reread path where the observations correct the hidden block, as Proposition 1 suggests.

Longer training and deployment. The ordering is not an artifact of the one-million-step budget. At five million steps DuoCell still leads the GRU and two identical GRUs on every environment, including over the last fifth of training. It also leads for the deployed controller: under a checkpoint rule fixed in advance, deterministic evaluation favours DuoCell over the GRU (Appendix E.4.1). A preregistered replication with the model frozen and every baseline retrained passes its acceptance rule (Appendix D).

5 CONCLUSION

What a recurrent agent needs depends on the hidden part of the state. When the hidden quantity is a derivative of the observed one, a recurrence that reads the current observation beside its memory can recover it, and a plain GRU must learn to. When it is an integral, no filter can remove its initial error, and the simpler construction suffices. This criterion is computed from the observation map, and it predicted the sign of the advantage, for a structured recurrence and for a GRU with the same reread, in both regimes on eight environments it had not seen. DUOCELL hedges when the answer is unknown. It splits one state budget between both constructions and has fewer parameters than a width-matched GRU. On masked MuJoCo it outperforms every matched baseline we ran in the derivative regime. In the integral regime it improves on the structured parent, while its difference from the GRU remains uncertain.

Limitations. Our evidence comes from masked locomotion. On masked classic control the regime split does not reproduce, and two identical GRUs outperform DUOCELL there and on InvertedDoublePendulum (Appendices E.5.1 and E.5.2). On the eight held-out environments DUOCELL lands between the two constructions rather than matching the better one: it trails the structured path with positions observed, by -0.119 $[-0.209, -0.026]$, and the GRU with velocities observed, by -0.119 $[-0.206, -0.045]$, while beating the other parent in each regime. A GRU with the same reread trails DUOCELL in domain, by 0.282, but leads it on the held-out tasks in both regimes, by 0.324 with positions observed and 0.156 with velocities observed, so the advantage of the structured carry over that control is established only in domain. Our main metric is online return at one million steps; at five times that budget an untuned TransformerXL closes most of the gap, and at equal wall-clock their difference is not detected (Appendix E.4.1). Finally, the observer interpretation of Sec. 3.2 motivates the architecture, but the present interventions do not establish that the trained policy uses an online filtering mechanism.

The future work would be to take this onto physical robots. There the hidden variables are hidden not by a mask we imposed but by the sensors the machine has. A joint encoder reports position and not velocity. A wheel odometer reports velocity and not position. An inertial unit reports acceleration and neither. Proposition 1 is computed from that sensor suite, so a robot's own hardware would determine the recurrence of its controller before any policy is trained. A platform with channels on both sides of the dichotomy is the natural use case for a cell that carries both constructions.

AI USE STATEMENT

The research idea and the experimental design are the author's. In this work, we used generative AI tools (an LLM-based coding and writing assistant) in a supporting role to implement methods, to give feedback on the experimental design, including suggesting additional controls and analyses that the author then approved, to help formulate and write the proof of Proposition 1, to refine hypotheses in response to results, and to help interpret results. We have not used generative AI tools to generate synthetic data sets; translation, dataset cleaning and reformatting, and qualitative or thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to create and edit software code, create and modify figures, draft and edit every section of the paper, format references, and search for and verify related literature. We have reviewed all AI-assisted work. Every experimental protocol, including all frozen decision rules and prospective predictions, was written before the corresponding runs and is included in the supplementary material. All reported numbers come from those runs and are regenerated from the recorded outputs by scripts; no data was generated by an AI tool, and each citation was checked against its source. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix H gives the training configuration, the per-environment hyperparameters inherited from POBAX, every search grid, and the parameter counts and compute. The anonymized supplementary material carries the cell and every control as runnable code, the launch configuration of every study, the frozen protocol document and decision rule for each suite, the per-seed results behind every

table, and the analysis scripts that turn per-seed extractions into the reported aggregates; the decision rules are applied by those scripts rather than by hand. Every frozen protocol was written before the corresponding runs, and suites postdating them are labelled post hoc wherever they are cited.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, 2021.

Cameron Allen, Aaron Kirtland, Ruo Yu Tao, Sam Lobel, Daniel Scott, Nicholas Petrocelli, Omer Gottesman, Ronald Parr, Michael L. Littman, and George Konidaris. Mitigating partial observability in sequential decision processes via the lambda discrepancy. In *Advances in Neural Information Processing Systems*, 2024.

Xiaoyu Chen, Yao Mu, Ping Luo, Shengbo Li, and Jianyu Chen. Flow-based recurrent belief state learning for POMDPs. In *International Conference on Machine Learning*, 2022.

Carlton Downey, Ahmed Hefny, Byron Boots, Geoffrey J Gordon, and Boyue Li. Predictive state recurrent neural networks. In *Advances in Neural Information Processing Systems*, 2017.

Scott Fujimoto, Wei-Di Chang, Edward Smith, Shixiang Shane Gu, Doina Precup, and David Meger. For SALE: State-action representation learning for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent independent mechanisms. In *International Conference on Learning Representations*, 2021.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Ahmed Hefny, Zita Marinho, Wen Sun, Siddhartha Srinivasa, and Geoffrey Gordon. Recurrent predictive state policy networks. In *International Conference on Machine Learning*, 2018.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *AAAI Conference on Artificial Intelligence*, 2018.

Khimya Khetarpal, Zhaohan Daniel Guo, Bernardo Avila Pires, Yunhao Tang, Clare Lyle, Mark Rowland, Nicolas Heess, Diana Borsa, Arthur Guez, and Will Dabney. A unifying framework for action-conditional self-predictive reinforcement learning. *arXiv preprint arXiv:2406.02035*, 2024.

Jan Koutník, Klaus Greff, Faustino Gomez, and Jürgen Schmidhuber. A clockwork RNN. In *International Conference on Machine Learning*, pp. 1863–1871, 2014.

Chris Lu, Yannick Schroecker, Albert Gu, Emilio Parisotto, Jakob Foerster, Satinder Singh, and Feryal Behbahani. Structured state space models for in-context reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

Carlos E. Luis, Alessandro G. Bottero, Julia Vinogradska, Felix Berkenkamp, and Jan Peters. Uncertainty representations in state-space layers for deep reinforcement learning under partial observability. *Transactions on Machine Learning Research*, 2025.

Steven Morad, Ryan Kortvelesy, Matteo Bettini, Stephan Liwicki, and Amanda Prorok. POPGym: Benchmarking partially observable reinforcement learning. In *International Conference on Learning Representations*, 2023a.

Steven Morad, Ryan Kortvelesy, Stephan Liwicki, and Amanda Prorok. Reinforcement learning with fast and forgetful memory. In *Advances in Neural Information Processing Systems*, 2023b.

Hassan Mortada, Cyril Falcon, Yanis Kahil, Mathéo Clavaud, and Jean-Philippe Michel. Recursive KalmanNet: Deep learning-augmented Kalman filtering for state estimation with consistent uncertainty quantification. In *European Signal Processing Conference (EUSIPCO)*, pp. 885–889, 2025.

Hai Nguyen, Brett Daley, Xinchao Song, Christopher Amato, and Robert Platt. Belief-grounded networks for accelerated robot learning under partial observability. In *Conference on Robot Learning*, 2020.

Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. Recurrent model-free RL can be a strong baseline for many POMDPs. In *International Conference on Machine Learning*, 2022.

Tianwei Ni, Benjamin Eysenbach, Erfan Seyedsalehi, Michel Ma, Clement Gehring, Aditya Mahajan, and Pierre-Luc Bacon. Bridging state and history representations: Understanding self-predictive rl. In *International Conference on Learning Representations*, 2024.

Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, Matthew Botvinick, Nicolas Heess, and Raia Hadsell. Stabilizing transformers for reinforcement learning. In *International Conference on Machine Learning*, 2020.

Guy Revach, Nir Shlezinger, Xiaoyong Ni, Adrià López Escoriza, Ruud J. G. van Sloun, and Yonina C. Eldar. KalmanNet: Neural network aided kalman filtering for partially known dynamics. *IEEE Transactions on Signal Processing*, 70:1532–1547, 2022.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021.

Jimmy T.H. Smith, Andrew Warrington, and Scott W. Linderman. Simplified state space layers for sequence modeling. In *International Conference on Learning Representations*, 2023.

Miguel Suau, Jinke He, Elena Congeduti, Rolf A. N. Starre, Aleksander Czechowski, and Frans A. Oliehoek. Influence-aware memory architectures for deep reinforcement learning in pomdps. *Neural Computing and Applications*, 37(19):13145–13161, 2025.

Yunhao Tang, Zhaohan Daniel Guo, Pierre Harvey Richemond, Bernardo Avila Pires, Yash Chandak, Rémi Munos, Mark Rowland, Mohammad Gheshlaghi Azar, Charline Le Lan, Clare Lyle, András György, Shantanu Thakoor, Will Dabney, Bilal Piot, Daniele Calandriello, and Michal Valko. Understanding self-predictive learning for reinforcement learning. In *International Conference on Machine Learning*, 2023.

Ruo Yu Tao, Kaicheng Guo, Cameron Allen, and George Konidaris. Benchmarking partial observability in reinforcement learning with a suite of memory-improvable domains. In *Reinforcement Learning Conference*, volume 6, pp. 1412–1439, 2025.

Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *International Conference on Machine Learning*, 2021.

Arun Venkatraman, Nicholas Rhinehart, Wen Sun, Lerrel Pinto, Martial Hebert, Byron Boots, Kris Kitani, and J. Andrew Bagnell. Predictive-state decoders: Encoding the future into recurrent networks. In *Advances in Neural Information Processing Systems*, 2017.

Claas A. Voelcker, Tyler Kastner, Igor Gilitschenski, and Amir-massoud Farahmand. When does self-prediction help? understanding auxiliary tasks in reinforcement learning. *Reinforcement Learning Journal*, 2024.

A Closed-Form Self-Predictive Auxiliary

The default configuration adds a latent self-prediction loss on the structured carry. The main results do not use it. It appears as one axis of the factorial ablation (Table 2), where it has no detectable effect at any setting, and in the preregistered replication, which was registered with it on.

Writing $z_t = b_t$ for the carry and $\phi_t = [z_t; a_t]$, the auxiliary solves its linear one-step predictor by ridge regression on detached statistics of each minibatch, instead of learning it:

$$W^* = \widehat{\mathbb{E}}\big[\bar{z}_{t+1}\bar{\phi}_t^\top\big]\Big(\widehat{\mathbb{E}}\big[\bar{\phi}_t\bar{\phi}_t^\top\big] + \lambda I\Big)^{-1}, \tag{7}$$

with $\bar{x} \equiv \mathrm{sg}(x)$ and $\widehat{\mathbb{E}}$ the average over the minibatch's transitions, excluding those that cross an episode boundary. The loss evaluates the frozen predictor on the online features, so gradient reaches the cell through ϕ_t only:

$$\mathcal{L}_{\mathrm{aux}} = \widehat{\mathbb{E}}\big[\|\mathrm{sg}(W^*)\,\phi_t - \bar{z}_{t+1}\|^2\big], \qquad \mathcal{L} = \mathcal{L}_{\mathrm{PPO}} + c\,\mathcal{L}_{\mathrm{aux}}, \tag{8}$$

with $c = 0.1$ and $\lambda = 10^{-2}$. This is the fast-predictor regime of Tang et al. (2023), at the cost of one $(\dim\phi) \times (\dim\phi)$ solve per update.

B Proofs

Proposition 1.

Proof. Write $A = \begin{bmatrix} I & \Delta t I \\ 0 & I \end{bmatrix}$, so $A^j = \begin{bmatrix} I & j\Delta t I \\ 0 & I \end{bmatrix}$.

(i) $C_P A^j = [\,I \quad j\Delta t I\,]$, so the two-step observability matrix $[\,C_P;\, C_P A\,]$ has full rank $2n$ and (A, C_P) is observable. For an observable pair and $R \succ 0$ the Riccati recursion of the Kalman filter converges to a bounded steady-state covariance. For any predicted estimate $\hat{s}_t^-$ with error $e_t = s_t - \hat{s}_t^- = (e_{x,t}, e_{v,t})$ the innovation is $\varepsilon_t = y_t - C_P\hat{s}_t^- = e_{x,t} + \eta_t$. The transition propagates $e_{x,t+1} = e_{x,t} + \Delta t\, e_{v,t}$, so $\partial\varepsilon_{t+1}/\partial e_{v,t} = \Delta t\, I$: the innovation sequence carries the hidden-block error at first order.

(ii) $C_V A^j = [\,0 \quad I\,]$ for every j, so the observability matrix has rank n at every horizon, and the unobservable subspace is $\{(c, 0) : c \in \mathbb{R}^n\}$ with eigenvalue 1. The pair is therefore not detectable.

For the bound, unroll the dynamics: $x_t = x_0 + \Delta t \sum_{k<t} v_k$ and $v_k = v_0 + b_k$, where $b_k = \sum_{j<k}(\Delta t\, a_j + w_j)$. Let $\mathcal{F}_t$ be the information in $(y_{0:t}, w_{0:t})$. Conditioning on more information cannot increase the posterior variance in expectation, so for any estimator $\hat{x}_t$ that is a function of $y_{0:t}$, $\mathbb{E}\|x_t - \hat{x}_t\|^2 \geq \mathbb{E}\,\mathrm{tr}\,\mathrm{Var}(x_t \mid \mathcal{F}_t)$. Given $\mathcal{F}_t$, every action and every b_k is known, so $x_t = x_0 + \Delta t\, t\, v_0 +$ (known). No observation depends on x_0, so x_0 is independent of $\mathcal{F}_t$ and of v_0, and its posterior covariance stays Σ_0. The observations reduce to $y_k - b_k = v_0 + \eta_k$ for $k = 0, \ldots, t$, so the posterior covariance of v_0 is $(\Sigma_v^{-1} + (t{+}1)R^{-1})^{-1}$ when $R \succ 0$. Summing the two independent contributions gives Eq. 1. For $R \succ 0$ the second term behaves as $\Delta t^2\, t\,\mathrm{tr}\,R$ for large t. With exact velocity observations ($R = 0$) the second term vanishes and the error stays at $\mathrm{tr}\,\Sigma_0$: the offset is never corrected. □

C Measured Recoverability of the Hidden Block

Proposition 1 is exact for the double integrator. For a MuJoCo body the dynamics also decide how much of the hidden block the observations reveal (Sec. 3.1). We therefore measure it directly, without any policy or training.

Protocol. For each masked environment we roll out the full simulator state in 64 parallel copies for 1000 steps under smooth random actions (an Ornstein–Uhlenbeck process independent of the state). Termination is disabled and the first 100 steps are discarded, so no window can recover the hidden block by integrating from the known initial state. A two-layer MLP then decodes the hidden block at step t from the last k masked observations and actions. We score it by R^2 on held-out rollouts, averaged over the hidden dimensions that vary, and floor it at zero, since a decoder worse

than the mean recovers nothing. The *history gain* $R^2(k{=}16) - R^2(k{=}1)$ is what memory can add to a single observation. InvertedPendulum is excluded. Without termination its simulation diverges within 25 to 90 steps, and with termination exploratory episodes last 3 to 9 steps, so no window can be separated from the reset. A stabilizing controller would keep it alive, but a controller acting on the full state leaks the hidden block through its actions.

Results. Table 3 gives the result. With positions observed, a window of 16 steps recovers the hidden velocities on every environment, with gains from $+0.39$ to $+0.98$. This is the kinematic argument of Sec. 3.1, and it holds whatever the dynamics. With velocities observed, the gain is at most $+0.18$ on six of the seven environments. The exception is InvertedDoublePendulum-V, at $+0.42$, where gravity couples the pole angles into the observed angular accelerations. It is the one integral-regime environment on which the structured recurrence leads the GRU. Across all fourteen environments the history gain and the structured recurrence's advantage have rank correlation $+0.67$ (permutation $p = 0.011$). The measured gain therefore refines the mask: it agrees with the mask on thirteen environments and corrects it on the one it misassigns. Pusher-P shows the limit of the criterion. Its velocities are recoverable, with a gain of $+0.72$, but the structured recurrence trails the GRU there. Recoverability says that a filter can reach the hidden block, not that the task needs it.

Table 3: Recoverability of the hidden block from the observation history, measured without any policy, against the structured recurrence's paired advantage over a GRU.

		R^2 of the hidden block			
Environment	Observed	1 step	16 steps	History gain	Structured − GRU
Hopper-P	positions	0.03	0.91	+0.89	+0.103
Walker-P	positions	0.00	0.81	+0.81	+0.648
HalfCheetah-P	positions	0.31	0.91	+0.60	+0.261
Ant-P	positions	0.01	0.41	+0.39	+0.479
InvertedDoublePendulum-P*	positions	0.02	0.97	+0.96	+0.671
Reacher-P*	positions	0.01	1.00	+0.98	+0.094
Pusher-P*	positions	0.24	0.95	+0.72	−0.176
Hopper-V	velocities	0.14	0.32	+0.18	+0.020
Walker-V	velocities	0.01	0.17	+0.16	−0.039
HalfCheetah-V	velocities	0.72	0.83	+0.12	−0.073
Ant-V	velocities	0.33	0.23	−0.10	−0.134
InvertedDoublePendulum-V*	velocities	0.08	0.49	+0.42	+0.491
Reacher-V*	velocities	0.14	0.29	+0.14	−0.244
Pusher-V*	velocities	0.00	0.00	+0.00	−0.365

* Held-out environment. Spearman correlation of gain and advantage over the 14 environments: $+0.67$ (permutation $p = 0.011$); $+0.61$ without flooring R^2 at zero. InvertedPendulum is excluded; see the text.

D PER-ENVIRONMENT RESULTS

Main-text tables. Table 4 gives the composition against each single construction on every environment, the per-environment view of Sec. 4.2. Table 5 gives the learning-rate selection behind the TransformerXL row of Table 1. Table 6 gives the numbers plotted in Fig. 3.

Primary baseline. The width-matched GRU is both POBAX's reference architecture and the exact ablation of DUOCELL obtained by removing the structured recurrence. It is therefore the primary comparison. Every other baseline is in Table 1.

Preregistered replication. The primary comparison is a replication on an independent group of runs under a rule frozen in advance: at least six of eight paired wins over the GRU baseline and no environment below -30%. Table 7 reports the outcome beside the development numbers.

Result. We executed the registered replication twice, on independent runs. The second execution meets the criterion: six of eight environments favour the cell, and the worst, Hopper-V, is at -11.4%. The first met the count and missed the floor on Hopper-V alone, at -32.3%. The derivative versus integral grouping was specified in the mechanism hypothesis, but no subgroup endpoint was reg-

Table 4: DUOCELL against each single construction on every environment, ten paired replications (mean AUC).

Environment	GRU	Structured	Dual-GRU	DUOCELL
Hopper-P	347	410	293	**476**
Walker-P	254	**492**	254	458
HalfCheetah-P	325	597	764	**885**
Ant-P	202	380	238	**381**
Hopper-V	**488**	487	371	451
Walker-V	266	259	255	**320**
HalfCheetah-V	885	861	**979**	892
Ant-V	263	231	266	**287**
All, normalized IQM	0.405	0.581	0.446	0.672
vs GRU	reference	$+0.176$ $[+0.111, +0.236]$	$+0.041$ $[-0.014, +0.092]$	$+0.266$ $[+0.198, +0.332]$
POI	reference	0.70	0.55	0.79

Table 5: Learning-rate selection for DUOCELL and TransformerXL on the selection seeds (mean AUC).

Environment	DUOCELL, default	DUOCELL, tuned	Setting chosen	TrXL, tuned
Hopper-P	502	**680**	structured width $1/4$	426
Walker-P	405	**446**	lr 5×10^{-4}	279
HalfCheetah-P	875	**1568**	lr 5×10^{-4}	1255
Ant-P	385	**471**	lr 5×10^{-4}	378
Normalized IQM	0.273	**0.622**	per environment	0.192

Table 6: The registered prediction on eight held-out environments, ten paired replications (mean AUC). The regime was assigned before any run; GRU+reread is the post-hoc control of Sec. 4.3. Gain is the structured recurrence's relative difference to the GRU.

Environment	GRU	GRU+reread	Structured	DUOCELL	Gain
predicted derivative regime					
InvertedPendulum-P	615	757	**778**	746	$+26.5\%$
InvertedDoublePendulum-P	770	**3145**	2058	1485	$+167.4\%$
Reacher-P	-499	-476	-484	-491	$+2.9\%$
Pusher-P	-1309	-1313	-1344	-1326	-2.7%
predicted integral regime					
InvertedPendulum-V	**544**	332	306	456	-43.9%
InvertedDoublePendulum-V	790	**2028**	1236	1055	$+56.5\%$
Reacher-V	-511	-549	-545	-558	-6.7%
Pusher-V	-1277	-1321	-1388	-1353	-8.7%

Table 7: Preregistered replication against an independently retrained GRU (paired gain in mean AUC).

Regime	Paired gain	Environments favouring DUOCELL	Worst environment
P (hidden a derivative)	$+96.3\%$	4/4	Hopper-P $+48.7\%$
V (hidden an integral)	$+3.5\%$	2/4	Hopper-V -11.4%

istered, so the following analysis is post hoc. All four derivative environments favour DUOCELL, whereas the integral subgroup shows no detected difference; this is not an equivalence result.

Hopper-V. Hopper-V is the environment that decides the criterion: it is the worst environment in both executions. Evaluated deterministically, the checkpoints of the registered configuration put the cell ahead of the GRU on it, 725 against 576 (Table 8), and the cell with the auxiliary is the arm whose stochastic and deterministic returns disagree most, by a factor of 1.68 against 1.18 for the GRU. The online metric therefore understates this controller on this environment. The effect belongs to the registered configuration and not to the cell as such: without the auxiliary the cell shows no such gap, and its deterministic return falls below the GRU's.

Table 8: Hopper-V: online AUC and deterministic return of the same final policies.

Arm	Online AUC	Deterministic	Ratio
GRU	488	576	1.18
Dual-GRU	371	294	0.79
DUOCELL+aux	432	**725**	1.68
DUOCELL, no auxiliary	451	362	0.80

Scale-invariant aggregation. Raw suite means weight environments by their reward scale, so Table 9 normalizes each environment over the four arms and reports interquartile means with seed-stratified bootstrap intervals (Agarwal et al., 2021). DUOCELL has the highest score in every group, and it leads both the GRU and two identical GRUs on all eight environments and on the derivative half.

Table 9: Normalized IQM of each arm, ten paired replications, per-environment min–max over the four arms.

	GRU	Dual-GRU	Structured	DUOCELL
All eight	0.405	0.446	0.581	0.672
	[0.360, 0.454]	[0.410, 0.480]	[0.536, 0.621]	[0.623, 0.723]
DUOCELL − GRU		+0.266 [+0.198, +0.332]		
DUOCELL − Dual-GRU		+0.226 [+0.163, +0.290]		
Derivative half	0.331	0.393	0.623	0.710
	[0.297, 0.363]	[0.337, 0.443]	[0.586, 0.659]	[0.650, 0.774]
DUOCELL − GRU		+0.379 [+0.315, +0.445]		
DUOCELL − Dual-GRU		+0.316 [+0.238, +0.396]		
Integral half	0.565	0.491	0.508	0.631
	[0.468, 0.646]	[0.442, 0.545]	[0.425, 0.594]	[0.556, 0.711]
DUOCELL − GRU		+0.067 [−0.047, +0.190]		
DUOCELL − Dual-GRU		+0.140 [+0.041, +0.241]		

Path ablation at full width. The composition exceeds both of its single-path parents on six of eight environments and is the best arm in Table 4 on five. The exceptions are Walker-P, where the structured path alone leads, and Hopper-V, where the GRU leads. Two identical GRUs are the best arm on HalfCheetah-V.

Capacity control. The over-parameterized GRU at width 640 carries 3.70M parameters against DUOCELL's 2.18M, a 70% excess, and trails it on seven of eight environments (Table 10), Hopper-V the exception by 1%. The lead is large on the derivative half and smaller but detected on the integral half.

Partner ablation. Table 11 holds the structured half fixed and varies its partner, at equal width and with the same readout. The GRU is the best partner on three of the four derivative environments. On Ant-P a state-space or attention partner does better.

Observation-difference control. A width-matched GRU reads $[o_t;\ o_t - o_{t-1}]$ at the default hyperparameters (Table 12). The difference feature does not close the gap on any derivative environment, and on HalfCheetah-P it hurts.

Table 10: A GRU with 70% more parameters than DUOCELL, ten paired replications (mean AUC).

Environment	GRU-640	DUOCELL	Relative difference
Hopper-P	364	**476**	+30.8%
Walker-P	262	**458**	+75.2%
HalfCheetah-P	210	**885**	+322.4%
Ant-P	213	**381**	+78.6%
Hopper-V	**456**	451	−1.1%
Walker-V	254	**320**	+25.8%
HalfCheetah-V	880	**892**	+1.4%
Ant-V	268	**287**	+7.3%
derivative half, paired normalized IQM			+0.562 [+0.458, +0.638]
integral half, paired normalized IQM			+0.168 [+0.032, +0.297]

Table 11: Partner ablation: the structured path paired with each partner, five paired replications (mean AUC).

		Structured path +			
Environment	GRU	GRU	attention	state-space	attention
Hopper-P	364	**492**	351	290	254
Walker-P	235	**457**	249	281	283
HalfCheetah-P	272	**783**	488	322	80
Ant-P	218	391	455	**516**	155
Hopper-V	**503**	340	290	387	323
Walker-V	263	**337**	273	276	274
HalfCheetah-V	964	855	**978**	865	760
Ant-V	292	305	321	**347**	279

Table 12: A GRU given the finite difference $o_t - o_{t-1}$, five paired replications (mean AUC).

Environment	GRU	GRU+Δo	DUOCELL+aux
Hopper-P	365	392	**492**
Walker-P	235	268	**458**
HalfCheetah-P	272	151	**783**
Ant-P	219	225	**392**
Hopper-V	**503**	393	340
Walker-V	263	264	**337**
HalfCheetah-V	**964**	810	856
Ant-V	292	267	**305**

Frame-stacking control. A memoryless policy reads the last four or eight observations, at the suite's default hyperparameters (Table 13). At these untuned hyperparameters it barely learns on HalfCheetah-P and Ant-P, so we report it here rather than in Table 1.

Table 13: Frame stacking against DUOCELL, derivative regime, ten paired replications (mean AUC).

Environment	4 frames	8 frames	DUOCELL
Hopper-P	157	153	**476**
Walker-P	232	230	**458**
HalfCheetah-P	12	58	**885**
Ant-P	42	3	**381**
DUOCELL − 4 frames			$+0.700$ $[+0.639, +0.760]$
DUOCELL − 8 frames			$+0.640$ $[+0.569, +0.708]$

Preregistered replication, per environment. Table 14 gives both executions of the registered replication, each against a GRU retrained in the same execution. The second execution passes both parts of the criterion. The first crossed the -30% floor on Hopper-V alone.

Table 14: Preregistered replication per environment: mean AUC of the second execution and the paired gain of each execution.

			Paired gain	
Environment	GRU	DUOCELL+aux	Execution 2	Execution 1
Hopper-P	347	515	$+48.7\%$	$+34.9\%$
Walker-P	254	444	$+74.5\%$	$+94.8\%$
HalfCheetah-P	348	909	$+161.2\%$	$+187.4\%$
Ant-P	202	370	$+82.8\%$	$+79.2\%$
Hopper-V	488	432	-11.4%	-32.3%
Walker-V	266	320	$+20.3\%$	$+28.0\%$
HalfCheetah-V	885	837	-5.5%	-11.2%
Ant-V	263	291	$+10.5\%$	$+4.5\%$

E ABLATIONS AND PROBES

Table 15: Five million environment steps: mean AUC over the full run, and over its final fifth in parentheses.

Environment	GRU	Dual-GRU	TrXL	TrXL 6×	DUOCELL
Hopper-P	534 (600)	262 (291)	505 (566)	414 (459)	**605 (685)**
Walker-P	299 (378)	192 (184)	285 (384)	269 (412)	**584 (715)**
HalfCheetah-P	699 (811)	985 (940)	1585 (1932)	**1777 (2143)**	1264 (1373)
Ant-P	578 (940)	910 (1884)	1629 (2998)	**1670 (2949)**	1193 (2195)

E.1 CONTROL LADDER

Sec. 4.2 names three cheaper explanations of the gain, capacity, a second path and explicit derivatives, and reports that each fails. Table 16 collects every control behind that statement with its interval, so the ladder can be read as one object rather than assembled from the text. Each control runs at the cell's own budget and changes one thing; each contrast is normalized per environment over its own two arms and paired across ten replications.

Three groups deserve comment. The capacity group gives a control *more* parameters than the cell, and the control still loses. This is a stronger argument than a matched parameter count. The parameter-matched rows widen the smaller baselines to the cell's parameter count, and each still loses by a similar margin. The last row is the complement. It keeps every parameter and the whole

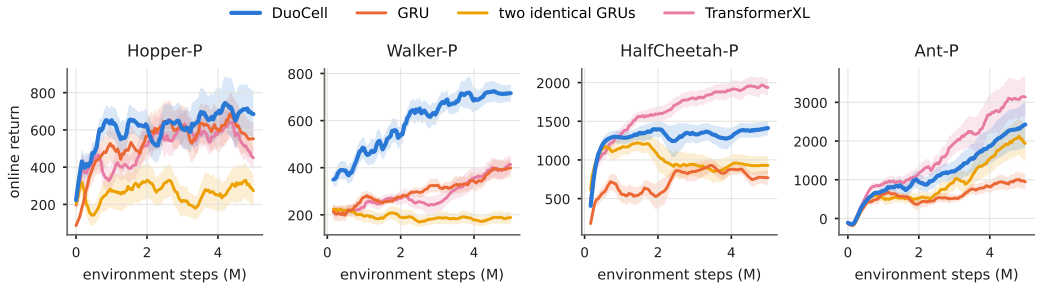


Figure 4: Learning curves at five million steps, ten replications per arm (mean and standard error).

computation graph and zeroes only the innovation's input from initialization. It is the one control with no detected difference from the cell: -0.038 $[-0.174, +0.098]$ for full minus input-zeroed. This interval does not establish equivalence or a benefit of the innovation input, and the unchanged parameter count cannot isolate a capacity effect. The closest architectural control is the structured path alone with the same gated readout. The cell still leads it by $+0.122$ $[+0.011, +0.240]$ at $0.59\times$ its wall-clock.

Table 16: Controls at the cell's budget, derivative regime, ten paired replications. Each contrast is normalized over its own two arms.

Control	DuoCell − control
Capacity	
GRU, width-matched	$+0.520$ $[+0.427, +0.598]$
GRU, 70% more parameters	$+0.562$ $[+0.458, +0.638]$
innovation-free cell, more parameters	$+0.420$ $[+0.300, +0.525]$
LSTM, width-matched	$+0.562$ $[+0.496, +0.623]$
LSTM at width 1024	$+0.558$ $[+0.464, +0.642]$
A second path	
two identical GRUs	$+0.315$ $[+0.228, +0.407]$
two identical GRUs, parameter-matched	$+0.352$ $[+0.250, +0.450]$
Generic heterogeneity	
GRU + state-space path	$+0.416$ $[+0.296, +0.517]$
GRU + state-space path, parameter-matched	$+0.432$ $[+0.329, +0.533]$
GRU + attention path	$+0.381$ $[+0.274, +0.468]$
GRU + attention path, parameter-matched	$+0.379$ $[+0.282, +0.460]$
Hand-built velocity features	
GRU reading $o_t - o_{t-1}$	$+0.482$ $[+0.395, +0.559]$
frame stacking, four frames	$+0.700$ $[+0.639, +0.760]$
frame stacking, eight frames	$+0.640$ $[+0.569, +0.708]$
Components of the cell	
GRU with the gated reread	$+0.282$ $[+0.196, +0.364]$
structured path alone	$+0.620$ $[+0.534, +0.695]$
structured path alone, with reread	$+0.122$ $[+0.011, +0.240]$
innovation input zeroed	$-0.038^{\dagger}$ $[-0.174, +0.098]$

† Interval contains zero.

E.2 Factorial ablation

Removing one component with every other in place cannot detect a component that another stands in for. If two components substitute for each other, removing either one changes nothing, and both look inert. A null one-at-a-time removal of the innovation admits exactly this reading. We therefore ran the full $2 \times 2 \times 2$ design over the innovation, the auxiliary and the gated reread, with ten paired replications per cell on the derivative half and one normalizer per environment across all eight cells.

The reread's estimated cost ranges from $+0.203$ to $+0.272$, with all four intervals above zero (Table 2). The innovation-removal effect is detected at three of the four settings; the auxiliary effect is undetected at all four. All six two-way interaction intervals contain zero, which does not establish additivity, independence, or the absence of substitution. These are individual, not simultaneous, confidence intervals. Removing all three costs $+0.315$ $[+0.236, +0.406]$. Because recurrence type is not an axis of this factorial, it cannot isolate an interaction between structured recurrence and reread.

The no-innovation arm must be the full cell with the innovation disabled, not a separately implemented module. The two compute the same function; we verified this by transplanting weights, with zero numerical difference. A separate module, however, receives a different random initialization. A contrast against it then measures the reimplementation rather than the component, and in our experiments it overstated the innovation's effect about fivefold.

E.3 Probes of trained networks

These ask what a trained network uses, as opposed to what an architecture can represent; a retrained policy may reallocate function across paths, so nothing here is evidence of architectural necessity.

E.3.1 Innovation severed at inference

Scaling the innovation input to zero retains the learned correction network and its prior and action inputs, while the readout retains the current embedding. It therefore removes one information source rather than the correction operation. For auxiliary-free cells, return falls by 4.2% on average across the derivative tasks (Table 17), including 11% on Hopper-P; the integral-task mean change is -3.7%. Paired bootstrap intervals for the raw mean return changes include zero on each derivative task, but do not establish equivalence.

The auxiliary-on probe gives a different picture: mean return falls from 1145.5 to 927.8 on HalfCheetah-P, approximately 19.0%, with an exploratory paired 95% interval of $[-459.2, -20.6]$ for the raw severed-minus-intact difference. The intact Ant-P policies in that probe average only 15.0, so little change there is weak evidence about competent controllers. These probes do not support a configuration-independent claim that innovation is unnecessary at inference.

Each comparison holds trained weights and initial evaluation randomness fixed. Subsequent trajectories and episode counts can diverge, however, and the fixed-horizon probe averages completed episodes. Evaluation variability is therefore not eliminated. Together with the difference between training removal and inference input zeroing, these limitations preclude identifying a unique online filtering mechanism.

Table 17: The innovation severed at inference: the same trained weights evaluated intact and severed (mean return).

Environment	Intact	Severed	Change	Environment	Intact	Severed	Change
Hopper-P	387	344	-11.0%	Hopper-V	323	337	$+4.2\%$
Walker-P	574	568	-1.0%	Walker-V	389	370	-4.7%
HalfCheetah-P	1230	1193	-3.1%	HalfCheetah-V	1135	1080	-4.9%
Ant-P	576	566	-1.8%	Ant-V	324	293	-9.4%

E.3.2 Test-time path lesion

Figure 5 shows how the composition beats both of its parts on HalfCheetah-P. The structured path alone rises fastest but gives return back after its midpoint, falling from 901 to 682 between the third and fourth quartiles of training. The GRU is slower throughout and still rising at the end. DuoCell leads in all four quartiles and does not decline: it keeps the structured path's early speed without its late loss.

A second probe asks which path a trained policy relies on. At evaluation we zero one half of the readout and leave the other untouched (Table 18). Ant-P and Ant-V are omitted: under this evaluation their policies terminate within a few steps in every condition, so there is nothing to compare.

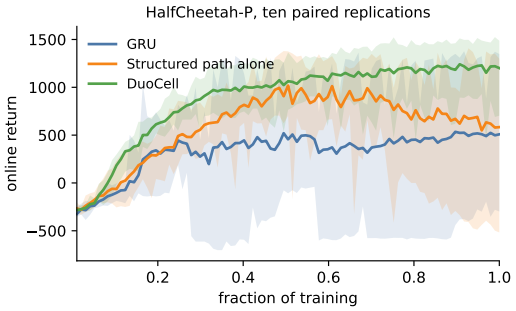


Figure 5: Online return on HalfCheetah-P, ten paired replications (mean and seed range).

Table 18: Test-time path lesion, ten replications (mean return).

Environment	Both paths	Structured only	GRU only
Hopper-P	382	201	62
Walker-P	564	559	88
HalfCheetah-P	1210	842	−10
Hopper-V	306	146	117
Walker-V	388	182	267
HalfCheetah-V	1103	383	473

The dependence follows the regime. In the derivative regime the structured path carries the policy: alone it keeps nearly all of the return on Walker-P and most of it on HalfCheetah-P and Hopper-P, while the GRU alone collapses on all three. In the integral regime the two paths share the load, and the GRU alone keeps more of the return than the structured path alone on Walker-V and HalfCheetah-V. A trained DUOCELL therefore leans on its reread path where the observations correct the hidden block, as Proposition 1 suggests. The probe measures dependence in a trained network, not architectural necessity; a policy retrained without one path could reallocate its function.

E.3.3 UNIFORM ACTION ROUTING

The default configuration passes the previous action directly to the structured transition, while the other recurrences receive it through the encoder input. Under uniform routing the encoder sees only the observation, and every recurrent path receives $[e_t, a_{t-1}]$ at its input. Table 19 repeats the ladder under this routing at the default hyperparameters, with five replications per arm on all eight environments and one normalizer per environment over the four arms.

Table 19: Uniform action routing: mean AUC over five replications, and the paired normalized IQM difference of DUOCELL against each control.

Environment	DUOCELL	Dual-GRU	GRU	GRU+SSM
Hopper-P	645	519	645	670
Walker-P	442	365	336	365
HalfCheetah-P	962	944	504	590
Ant-P	381	260	232	281
Hopper-V	381	463	469	486
Walker-V	285	297	288	285
HalfCheetah-V	854	1007	683	1010
Ant-V	313	251	270	256
Difference, all eight	reference	+0.129 [+0.024, +0.224]	+0.230 [+0.097, +0.347]	+0.135 [+0.016, +0.247]
Difference, derivative half	reference	+0.277 [+0.135, +0.424]	+0.383 [+0.220, +0.510]	+0.260 [+0.145, +0.400]

The ordering of the default ladder is preserved. Every contrast excludes zero on both aggregates, and on the derivative half the cell leads the identical-path control on all four environments. The advantage therefore does not come from the structured path's direct access to the action.

E.4 Design choices

Each of these varies one decision of Sec. 3.3 while holding the budget fixed, and reports how far the result travels with it.

E.4.1 Training budget

The primary metric integrates the online learning curve over one million steps, far below the fifty million at which the suite reports its masked-MuJoCo results (Tao et al., 2025). Three objections follow. A longer budget might erase the ordering. The online metric says nothing about the deployed controller. Equal environment steps are not equal compute. We froze a suite and three decision rules against these objections before running it: five million steps on the four derivative environments, ten paired replications, and the cell against a GRU, two identical GRUs and TransformerXL, all at default hyperparameters.

B1: training budget. Figure 4 shows the curves behind Table 15. Over the full five million the cell leads the GRU by $+0.354$ $[+0.260, +0.418]$ and the identical-path control by $+0.366$ $[+0.259, +0.452]$, four of four environments each. Over the final fifth alone the leads are $+0.307$ $[+0.196, +0.393]$ and $+0.366$ $[+0.235, +0.477]$, so the ordering holds to the end of training and is not an early-learning effect that decays.

B2: deployed controller. Every run stores ten deterministic evaluations, and the reported controller is the best of them under a rule applied identically to every arm. Under it the cell leads the GRU by $+0.288$ $[+0.198, +0.392]$ and the identical-path control by $+0.291$ $[+0.188, +0.405]$. Without the selection rule, the final policies alone still favour the cell over the GRU, $+0.161$ $[+0.011, +0.306]$, while against the transformer the interval contains zero, $+0.006$ $[-0.132, +0.141]$. At this budget the advantage is therefore not only one of learning speed. The transformer trainer does not evaluate periodically, so its runs store no intermediate checkpoints and the selection rule cannot apply to them; they are compared on the final policy alone.

B3: TransformerXL. At equal environment steps the cell leads TransformerXL over the full budget, $+0.118$ $[+0.022, +0.200]$, but in the final fifth the interval contains zero, $+0.065$ $[-0.042, +0.159]$, and the environments split two and two: the cell is ahead on Hopper-P and Walker-P and behind on HalfCheetah-P and Ant-P (Table 15). A further arm gives the transformer six million steps against the cell's five. At $1.2\times$ the steps the transformer is indistinguishable from itself at equal steps, $+0.009$ $[-0.056, +0.071]$, and the cell still leads it by $+0.110$ $[+0.018, +0.189]$. The equal-step lead is therefore not a small shortfall of transformer steps. The large margin of Table 1 is a one-million-step result under a matched learning-rate search; at five times the budget and default hyperparameters, an untuned transformer closes most of it.

Compute is the transformer's advantage. At this budget a cell run takes $3.6\times$ the transformer's wall-clock (249 against 69 minutes), because attention parallelizes over time and a recurrent scan does not. We truncate each cell run at the time a six-million-step transformer run took to finish, between 16% and 42% of the cell's own budget. At this equal wall-clock the two are indistinguishable, -0.070 $[-0.191, +0.075]$: the cell is ahead on Hopper-P and Walker-P and behind on HalfCheetah-P and Ant-P. The advantage of DuoCell is one of sample efficiency, not of compute.

E.4.2 Component ablations and design choices

One-at-a-time removals. Table 20 removes one component at a time from the configuration with the auxiliary on. Removing the reread costs the most, $+0.238$, and removing the innovation costs $+0.144$. Removing the auxiliary has no detectable effect. These are the single-removal cells of the factorial of Table 2, measured against their own reference rather than on the factorial's joint normalizer.

Width split. The even split between the two paths is a hyperparameter, not a derivation. Giving the structured path a quarter or three quarters of the budget changes nothing detectable: $+0.146$

Table 20: One-at-a-time component removals from the configuration with the auxiliary on, derivative regime, ten paired replications (mean AUC).

Environment	DUOCELL+aux	no auxiliary	no innovation	no reread
Hopper-P	515	476	490	476
Walker-P	444	458	438	339
HalfCheetah-P	867	885	648	646
Ant-P	370	381	329	272
Cost of removal	*reference*	-0.032	$+0.144$	$+0.238$

$[-0.003, +0.310]$ and $+0.012$ $[-0.157, +0.190]$ against the even split on the derivative half. A quarter helps on Hopper-P, which is why the tuned configuration selects it there (Table 5).

E.5 BOUNDARY EXPERIMENTS

The three suites below were frozen before they were run and are reported whichever way they came out. Two of them return negatives, and those negatives fix the scope stated in the Limitations.

E.5.1 HELD-OUT MORPHOLOGIES

The replication above holds the environment set fixed. To test the derivative-regime advantage beyond the four development morphologies, we apply the same construction to four simulator morphologies that took no part in any decision in this paper: two balancing tasks (InvertedPendulum and InvertedDoublePendulum) and two goal-reaching tasks (Reacher, Pusher). Observation layouts were read off the simulator source. Every nonvelocity feature is observed and every velocity is hidden. No published environment-specific hyperparameters exist for these tasks, so both arms use the same untuned values. This removes tuning asymmetry at the cost of absolute performance.

Table 21: Held-out morphologies with positions observed, ten paired replications (mean AUC).

Environment	GRU	DUOCELL	Paired gain
InvertedPendulum-P	615	**746**	$+21.3\%$
InvertedDoublePendulum-P	770	**1485**	$+93.0\%$
Reacher-P	-499	-491	$+1.7\%$
Pusher-P	-1309	-1326	-1.3%

The cell leads on three of the four morphologies and on most paired replications (Table 21), consistent with the derivative versus integral split. Balancing a pole depends on the angular velocity withheld by the observation, and these two tasks carry the largest margins outside the development suite. In the goal-reaching tasks the dominant signal is the static target position, and it is observed, so the arms converge. The advantage is therefore not specific to the development morphologies, but it does track the hidden variables rather than the benchmark.

Control ladder on balancing tasks. The two-arm comparison shows that the effect transfers; it does not show that the attribution transfers. We therefore ran the seed-paired controls of Sec. 4.4 on the two balancing tasks at both widths, with the same uniform hyperparameters (Table 23). DUOCELL leads the GRU at both widths, by $+0.533$ at width 256 and $+0.438$ at width 512, and the finite-difference input hurts the GRU rather than helping it. Dual-GRU is the exception. It leads DUOCELL on the pooled pair at both widths, and the lead comes from InvertedDoublePendulum-P, where two identical GRUs reach nearly twice the cell's return. On InvertedPendulum-P DUOCELL is the best arm at both widths. Out of domain, the structured path therefore beats the GRU as in domain, but on one of the two tasks a second identical path helps more.

Specialization in the Dual-GRU. Two identical recurrences do not behave as redundant copies of one another. A test-time lesion of the width-512 checkpoints (Table 22) collapses return on InvertedDoublePendulum whichever half is removed, on almost every seed, so the two paths have specialized even though nothing but their initialization distinguishes them; on the easier InvertedPendulum either half alone suffices, which is what one would expect of a task that does not need

the specialization. Capacity does not explain the effect either: a single GRU at half width performs as the full-width GRU does, an interval containing zero, while the Dual-GRU leads that single path by $+0.563$ $[+0.406, +0.640]$. What the out-of-domain advantage of two identical paths rewards is therefore the presence of two paths, not the width they divide and not an ensemble average over them.

Table 22: Test-time path lesion on the held-out balancing tasks, width 512 (mean return). For DUOCELL the first half is the structured path.

Environment	Arm	Intact	First half only	Second half only
InvertedDoublePendulum-P	Dual-GRU	**4653**	795	2034
	DUOCELL+aux	**1228**	1054	337
InvertedPendulum-P	Dual-GRU	**952**	761	928
	DUOCELL+aux	**315**	256	20

Table 23: The control ladder on the two held-out balancing tasks, ten paired replications (mean AUC), and paired normalized IQM differences.

Environment	Width	GRU	Dual-GRU	GRU+Δo	DUOCELL+aux
InvertedPendulum-P	256	460	715	422	**732**
	512	615	670	563	**753**
InvertedDoublePendulum-P	256	904	**2117**	804	1268
	512	770	**2181**	614	1199

Contrast	Width	Difference
Dual-GRU − GRU	256	$+0.602$ $[+0.430, +0.719]$
Dual-GRU − GRU	512	$+0.447$ $[+0.295, +0.528]$
GRU+Δo − GRU	256	$-0.240^{\dagger}$ $[-0.501, +0.014]$
GRU+Δo − GRU	512	-0.337 $[-0.535, -0.111]$
DUOCELL+aux − GRU	256	$+0.533$ $[+0.368, +0.665]$
DUOCELL+aux − GRU	512	$+0.438$ $[+0.279, +0.626]$
Dual-GRU − DUOCELL+aux	256	$+0.272$ $[+0.036, +0.465]$
Dual-GRU − DUOCELL+aux	512	$+0.239$ $[+0.109, +0.314]$

† Interval contains zero.

E.5.2 MASKED CLASSIC CONTROL

The masked classic-control tasks pose the same two inference problems in a different simulator family. Pendulum-P and CartPole-P observe positions and hide velocities; Pendulum-V and CartPole-V do the reverse. We run the GRU, two identical GRUs, the GRU with Δo, and the cell, ten paired replications each, at uniform untuned hyperparameters.

The regime split does not reproduce (Table 24). The cell shows no detected advantage over the GRU in either regime: -0.191 $[-0.415, +0.029]$ with positions observed and -0.027 $[-0.169, +0.118]$ with velocities observed. Two identical GRUs are the strongest arm in both regimes and lead the cell by $+0.339$ $[+0.202, +0.492]$ and $+0.306$ $[+0.189, +0.408]$. The held-out balancing tasks rule out low dimension as the reason, since the structured path leads there with two or three degrees of freedom. Observability of the masked system is therefore necessary but not sufficient for the structured path to help. The two-identical-path effect seen on the held-out balancing tasks appears here too.

E.5.3 INTEGRAL-REGIME LOSS

The structured recurrence does not help in the integral regime. Two candidate explanations fail their tests. The auxiliary is not the cause: the auxiliary-free structured recurrence remains level with the GRU on the integral environments (Table 4). The retention bound is not the cause either. Raising $r_{\max}$ from 0.995 to 0.99999 turns the prior's explicit skip coefficient into an accumulator and leaves everything else untouched. It leaves the integral regime within noise, with inconsistent signs across its four environments, while costing return on the derivative controls. Neither design choice explains

Table 24: Masked classic control, ten paired replications (mean AUC). The last rows give normalized IQMs and paired differences against the GRU.

Environment	GRU	Dual-GRU	GRU+Δo	DUOCELL+aux
Pendulum-P	-1357	-1307	-1392	-1373
CartPole-P	358	**386**	370	325
Pendulum-V	-1394	-1307	-1401	-1303
CartPole-V	311	**353**	309	280
All, normalized IQM	0.423	0.645	0.347	0.299
vs GRU	reference	$+0.221$ $[+0.129, +0.310]$	-0.077 $[-0.154, +0.008]$	-0.125 $[-0.215, -0.014]$
POI	reference	0.85	0.43	0.38

the integral-regime result. It is what Proposition 1 predicts: when the observations cannot correct the hidden block, reading them beside a memory has nothing to add, and a GRU with the same reread shows the same penalty (Sec. 4.3).

F MEMORY-IMPROVABILITY TEST

A null result for one architecture admits two readings: the architecture is wrong for the task, or the task has no hidden state worth recovering. Tao et al. (2025) separate them with a comparison: a masked task is memory-improvable when an agent with more state information outperforms one with less. We ran the suite this comparison requires and found that it does not apply to these environments.

We built a fully observed twin of each morphology, which observes every quantity that either mask hides, and trained a memoryless policy and a GRU on it at the shared hyperparameters. Two results follow.

First, the memoryless policy barely learns at the shared hyperparameters. On the twins a GRU leads it by $+0.492$ $[+0.436, +0.555]$, and on HalfCheetah and Ant it stays near zero return. A frame-stacking policy, which is memoryless over stacked observations, shows the same failure on HalfCheetah-P and Ant-P (Table 13). At this budget and these untuned hyperparameters the feed-forward policy is the weaker optimizer, so a recurrent advantage is not by itself evidence that a task hides anything.

Second, full observation is not an upper bound on these tasks. On Walker and Ant the best masked agent finishes above the best fully observed agent (Table 25). Averaged over the four morphologies, the integral-mask agents come within 4.5% of the twins, and the derivative-mask agents finish 29% *above* them. Masking changes the observation dimension and with it the optimization problem; it does not simply remove information from an otherwise identical task. The comparison presupposes a nesting that these environments do not provide.

Table 25: Fully observed twins against the best masked agents (mean AUC).

	Fully observed		Best under the mask	
Environment	Memoryless	GRU	Velocities hidden	Positions hidden
Hopper	147	471	410	**488**
Walker	263	258	**492**	266
HalfCheetah	-60	**1017**	597	885
Ant	-52	208	**380**	263

We therefore do not claim the integral half is memory-improvable on this evidence, and we do not use the twins anywhere in the paper's argument. The defence of the integral-regime null rests instead on the sign of the held-out result in Sec. 4.3. On a task with nothing to gain, the structured recurrence and the GRU would tie. On the four registered integral-regime environments they do not: the structured construction is worse by -0.214 $[-0.282, -0.160]$. A GRU with the same reread

shows a smaller penalty in the same regime. At shared untuned hyperparameters a larger, differently parameterized model can also simply optimize worse, so we treat this as supporting evidence only.

G Aggregation Robustness and Final Policies

The suite-level aggregates in the main text normalize each environment by the min–max of all arms and seeds pooled in that comparison, following the released analysis code. This normalizer is data-dependent: it moves when an arm is added or removed, extreme seeds set the weight an environment carries in the pooled score, and the bootstrap holds the estimated scale fixed rather than resampling it. This appendix separates the conclusions depending on that choice from those that do not, and closes the gap between the online metric and deployed performance.

Normalizer sensitivity. A pooled min–max normalizer moves when an arm is added or removed. Table 26 recomputes the two headline contrasts under five normalizers, and Table 27 gives the paired differences in raw return with no normalization at all. DuoCell leads both controls with an interval excluding zero under every normalizer. The magnitudes change with the normalizer, so we read them as ordering evidence within one table rather than as effect sizes that transfer between tables.

Table 26: The two headline contrasts under five normalizers, all eight environments.

Normalizer	DuoCell − Dual-GRU	DuoCell − GRU
min–max over the four arms	$+0.226$ $[+0.163, +0.290]$	$+0.266$ $[+0.198, +0.332]$
min–max, structured path removed	$+0.237$ $[+0.169, +0.306]$	$+0.276$ $[+0.205, +0.343]$
min–max, GRU-640 added	$+0.226$ $[+0.163, +0.289]$	$+0.251$ $[+0.187, +0.313]$
min–max over each pair	$+0.272$ $[+0.194, +0.354]$	$+0.351$ $[+0.260, +0.434]$
per-environment z-score	$+0.989$ $[+0.735, +1.249]$	$+1.124$ $[+0.834, +1.407]$

Table 27: Paired differences in raw return, with 95% seed-resampled intervals.

Environment	DuoCell − Dual-GRU	DuoCell − GRU	DuoCell − Structured
Hopper-P	$+183$ $[+95, +272]$	$+129$ $[+70, +200]$	$+66$ $[-5, +149]$
Walker-P	$+204$ $[+152, +253]$	$+204$ $[+159, +245]$	-33 $[-70, +6]$
HalfCheetah-P	$+121$ $[-50, +308]$	$+560$ $[+407, +689]$	$+288$ $[+187, +387]$
Ant-P	$+143$ $[+95, +188]$	$+179$ $[+134, +226]$	$+2$ $[-25, +28]$
Hopper-V	$+80$ $[-28, +191]$	-37 $[-157, +94]$	-35 $[-108, +38]$
Walker-V	$+65$ $[+40, +89]$	$+54$ $[+32, +78]$	$+61$ $[+35, +90]$
HalfCheetah-V	-87 $[-292, +127]$	$+7$ $[-216, +255]$	$+31$ $[-195, +255]$
Ant-V	$+22$ $[-13, +59]$	$+24$ $[-25, +72]$	$+56$ $[+21, +88]$

Deterministic final-policy evaluation. The training metric is an online stochastic-policy quantity, and Sec. E.3.2's observation that some intact final policies collapse deterministically makes the gap explicit. Each stored checkpoint contains a deterministic evaluation of the final policy; we summarize it per arm, calling a run *collapsed* when its deterministic return falls below half of the lowest median deterministic return among the three arms on its environment. Confirmation-stage checkpoints cover the GRU, Dual-GRU, and cell on all eight environments. We report only this environment-relative rate: pooling raw deterministic returns across tasks with different reward scales would not define a meaningful final-policy aggregate. Transformer checkpoints cover only the derivative half and are therefore omitted from this comparison.

Table 28: Deterministic evaluation of the final policies at one million steps. A run collapses when its return falls below half of the lowest median return among the three arms on its environment.

Arm	Runs	Collapse rate
GRU	40	0.33
Dual-GRU	40	0.30
DuoCell	40	0.28

Collapse is common and similar across all three architectures at one million steps (Table 28), so this check does not establish final-controller superiority there. These are end-of-training snapshots with no checkpoint-selection rule. The five-million-step suite (Appendix E.4.1) stores ten deterministic evaluations per run and applies a selection rule fixed before execution; under it the cell leads the GRU by $+0.288$ $[+0.198, +0.392]$.

H Reproducibility and Cost

Training configuration. All arms train with the official POBAX recurrent PPO implementation. Shared settings, identical across every arm and environment: 4 parallel environments, rollout length 128, 4 minibatches, 4 update epochs per batch, discount $\gamma = 0.99$ (overridden by an environment-specific value where the suite defines one), PPO clip 0.2, entropy coefficient 0.01, value-loss coefficient 0.5, maximum gradient norm 0.5, Adam with a linearly annealed learning rate, one million environment steps, and hidden size 512 for every recurrent arm. Each comparison uses ten paired replications from two seed families of five, one for selection and one for confirmation. A seed fixes all five replications of a study, and the paired arms share it.

Per-environment settings. The published POBAX table supplies GAE λ_0 and learning rate per environment. λ_0: Hopper-P 0.7, Hopper-V 0.7, Walker-P 0.9, Walker-V 0.95, HalfCheetah-P 0.7, HalfCheetah-V 0.9, Ant-P 0.7, Ant-V 0.5. Learning rate: 2.5×10^{-4} everywhere except HalfCheetah-P at 2.5×10^{-5}. Every recurrent arm inherits these values in the default comparisons. TransformerXL uses the POBAX implementation with hidden size 512, embedding size 192, $\lambda_1 = 0.95$, and its per-environment selected learning rate and λ_0 (HalfCheetah-V 2.5×10^{-4} with $\lambda_0 = 0.9$; Walker-V 2.5×10^{-5} with $\lambda_0 = 0.95$; elsewhere the default values above).

Search grids. Tuning suite (Sec. 4.2), all frozen before the runs: the cell searched learning rates $\{10^{-4}, 2.5 \times 10^{-4}\ \text{(default)}, 5 \times 10^{-4}\}$ and structured-path width fractions $\{1/4, 1/2\ \text{(default)}, 3/4\}$; TransformerXL searched learning rates $\{2.5 \times 10^{-5}, 10^{-4}, 2.5 \times 10^{-4}\ \text{(default)}\}$; each control (structured-only, Dual-GRU, GRU+SSM, GRU+attn) received the same three-point learning-rate grid as the cell, without width candidates. The frame-stacking baseline searched stack depth $k \in \{1, 2, 4, 8\}$ and width $\{256, 512\}$, selected per environment on that arm alone. Selection inside every grid was per-environment mean AUC on the selection seeds under a rule frozen in the protocol documents.

Post-hoc suites. Four suites postdate the frozen protocols and are labelled post hoc wherever cited: the uniform-routing ladder (Appendix E.3.3), the full no-innovation arm (Table 20), the observation-difference control (Table 12), and the extension of the transformer's grid to 5×10^{-4} on the derivative half. The no-innovation arm, the retention-bound removal, the reread removal, and the Dual-GRU with the cell's reread readout were also rerun on the confirmation seeds, each under the Table 20 criterion fixed before the runs.

Cell and auxiliary. The auxiliary uses coefficient 0.1 and ridge $\lambda = 0.01$; the tuned configuration of Sec. 4.2 removes it. The structured prior's retention is bounded in $[0.5, 0.995]$; the linear-attention control uses 4 heads with decay bounded in $[0.9, 0.999]$. Parameter counts at full width: GRU 2.37M, structured recurrence alone 3.43M, dual-GRU control 1.98M, DuoCell 2.18M.

Artifacts and compute. The supplementary material carries every frozen protocol document, the launch configuration of every study, the analysis code, the learning curves, and the per-seed extraction behind every table. Compute: eight RTX A5000 GPUs and roughly 650 studies of five replications each, all at one million steps except forty studies at five million, at 12 to 90 minutes per million steps depending on architecture and width. The auxiliary adds one linear solve per update. At matched width DuoCell trains at roughly $0.8\times$ the throughput of a single GRU and $0.16\times$ that of TransformerXL, measured over the held-out confirmation runs, since attention parallelizes over time and a recurrent scan does not.