SPACE: SPARSE PREDICTIVE ATTRACTOR VIA COUNTERFACTUAL EVICTION FOR STREAMING VIDEO MEMORY

Anonymous authors
Paper under double-blind review

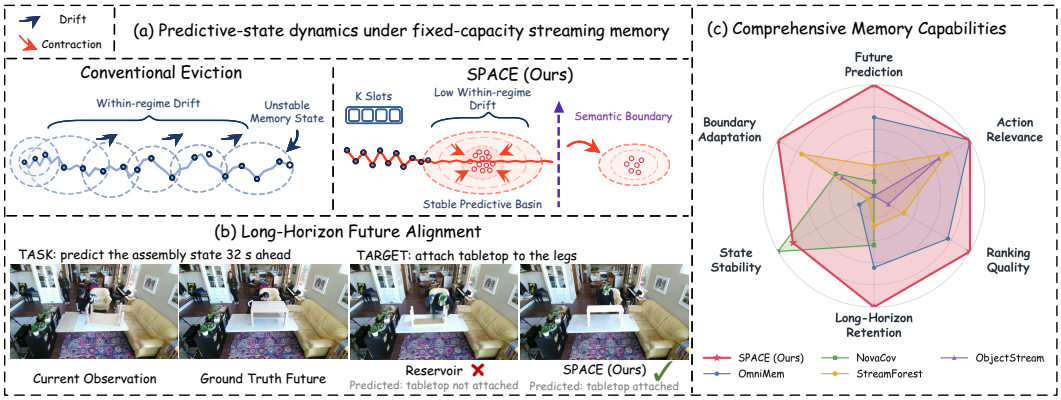


Figure 1: **Predictive stability of SPACE.** Unlike conventional streaming memory, which drifts under repeated updates, SPACE forms a stable yet adaptive predictive attractor, improving long-horizon future alignment while preserving broad memory capabilities. In (c), the six metrics are independently min–max normalized to $[0, 1]$ across all methods, with directions aligned so that higher is better; radial rings are spaced by 0.2. Predicted futures are nearest-neighbor RGB visualizations of future latents from a participant-disjoint RGB retrieval bank, not generated pixels.

ABSTRACT

Fixed-capacity streaming video memory requires repeated eviction decisions whose effects accumulate over time. Yet existing policies are evaluated primarily in terms of retained information or downstream accuracy, leaving how repeated updates alter the futures supported by memory largely unexamined. We define a memory's predictive state as the future representations supported by its retained history and formulate eviction as counterfactual control over transitions in this space. We introduce **SPACE** (Sparse Predictive Attractor via Counterfactual Eviction), which uses a frozen multi-horizon JEPA to predict the future representations induced by alternative eviction actions. Counterfactual utility identifies future-useful alternatives, while slow predictive-basin geometry determines when to correct avoidable drift and when to adapt to sustained predictive change, without online parameter updates. We further introduce **MABS-Bench**, which evaluates future-task sufficiency, within-regime predictive stability, transition responsiveness, and perturbation recovery under matched causal streams and memory budgets. Across multiple video datasets, SPACE yields consistent improvements in dataset-native task performance while reducing predictive-state drift.

1 INTRODUCTION

Streaming video models must summarize an indefinitely growing observation stream using a memory of fixed capacity (Xu et al., 2021; Wu et al., 2022; He et al., 2024; Qian et al., 2024; Kim et al., 2025). Once the memory reaches capacity, each new observation requires a decision about

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.

which event to discard, and the effects of these local decisions accumulate over time. Existing eviction policies typically prioritize recency, novelty, coverage, or an estimate of the importance of an individual event (Vitter, 1985; Aljundi et al., 2019b; Zeng et al., 2025; Duan et al., 2026; Dong et al., 2026; Ge et al., 2026). However, evaluations centered on retained information or downstream accuracy may leave unclear how successive evictions reshape the future supported by memory. The resulting drift can misalign long-horizon predictions with the unfolding activity even under routine input variation, as illustrated in Fig. 1(b). Yet preventing such drift by simply limiting replacement risks retaining outdated history when the activity genuinely shifts. This creates a stability-plasticity tension, also central to continual learning (Rolnick et al., 2019): memory must resist drift under routine variation while remaining responsive to genuine semantic change.

We argue that resolving this tension requires characterizing video memory through the futures it supports. Different observed states may lead to similar futures, whereas similar observed states may evolve into different future activities. Following predictive representation learning (Oord et al., 2018; LeCun et al., 2022; Assran et al., 2023; Bardes et al., 2024), we define the predictive state as the future representations supported by the retained history. As shown in Fig. 1(a), repeated local evictions can cause this state to drift within a stable semantic regime. To resist such drift, predictive states should remain near a persistent reference within a semantic regime, while allowing that reference to shift when future semantics change. This motivates an adaptive predictive attractor (Hopfield, 1982; Ramsauer et al., 2020; Susman et al., 2019). Because a large departure may result either from an avoidable eviction or from a genuine shift shared by all feasible updates, determining when to stabilize requires comparing the counterfactual futures induced by alternative evictions.

To this end, we propose **SPACE** (**S**parse **P**redictive **A**ttractor via **C**ounterfactual **E**viction), a controller for fixed-capacity streaming video memory. Rather than scoring stored events independently, SPACE evaluates feasible evictions according to how they alter the future supported by memory. Counterfactual predictions identify utility-preserving alternatives, while the adaptive predictive attractor provides a reference for comparing their predicted states and determining whether drift can be avoided without leaving the current predictive regime. SPACE therefore intervenes selectively to stabilize predictive dynamics when a useful, state-consistent alternative exists and otherwise allows the memory to adapt. As summarized in Fig. 1(c), this approach improves predictive stability and long-horizon retention without sacrificing broader memory capabilities.

To evaluate how well streaming memory preserves predictive utility while remaining stable under routine variation and adaptive to genuine semantic change, we introduce **MABS-Bench**, a controlled benchmark for fixed-capacity memory under matched causal streams. It measures four complementary properties: future-task sufficiency, within-regime predictive stability, responsiveness to genuine semantic transitions, and recovery toward a paired clean trajectory after controlled memory corruption. Together, these dimensions help distinguish useful stability from mere inertia.

The main contributions of this paper are summarized as follows:

- We formulate fixed-capacity streaming video memory as a counterfactual control problem over transitions in a predictive state space and identify uncontrolled slot churn as a source of predictive-state drift under routine streaming variation.
- We propose SPACE, a sparse eviction controller that combines JEPA-based counterfactual future utility with slow predictive-basin geometry to form an adaptive and recoverable predictive attractor through sparse, basin-aware interventions.
- We introduce MABS-Bench, a matched-stream benchmark that jointly measures downstream utility, predictive-state dynamics, and recovery after controlled memory perturbations under a shared causal-stream protocol. Across multiple video datasets, SPACE improves the dataset-native task metric while reducing predictive-state drift.

2 Related Work

Streaming and long-form video modeling. Video models designed for long or continuous streams must preserve information from an expanding history while keeping inference and storage costs bounded (Song et al., 2024; Diko et al., 2025; Zhang et al., 2025; Huang et al., 2025). Recent systems address this constraint by organizing observations into persistent event hierarchies (Zeng et al., 2025; Ge et al., 2026), maintaining object-centric anchors (Dong et al., 2026), compressing

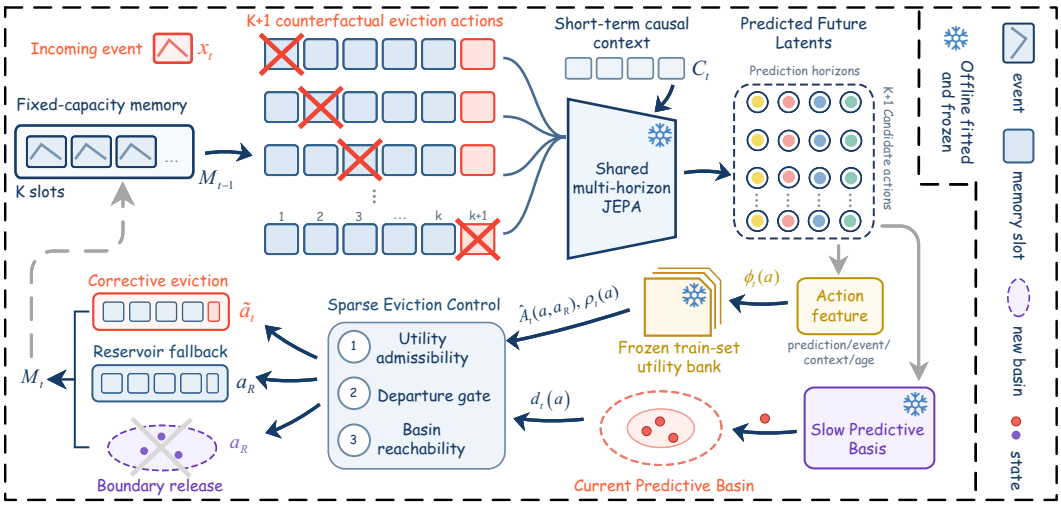


Figure 2: **SPACE overview.** Candidate eviction actions are evaluated by a frozen multi-horizon JEPA and selected using complementary future-utility and predictive-basin signals.

multimodal key-value states (Sun et al., 2026), or constructing historical event memories for long-horizon prediction and control (Wang et al., 2026). These approaches demonstrate the value of structured history for preserving task-relevant information under a bounded memory budget.

Memory selection and eviction. Fixed-capacity memory requires a policy for deciding which observations to retain. Classical strategies include FIFO and reservoir sampling (Vitter, 1985), while learned or training-free policies rank items according to recency, novelty, redundancy, attention, or influence (Aljundi et al., 2019b;a). More structured criteria preserve complementary events, perturbation-sensitive states (Sun et al., 2026), persistent objects (Dong et al., 2026), or set-level coverage (Duan et al., 2026). These methods therefore determine what to retain based primarily on properties of individual items or of the retained set.

Predictive representation learning and memory dynamics. Predictive coding and joint-embedding predictive architectures learn representations by predicting future or target embeddings from observed context rather than reconstructing pixels (Oord et al., 2018; LeCun et al., 2022; Assran et al., 2023). V-JEPA extends feature prediction to video and demonstrates that this objective captures both appearance and motion semantics (Bardes et al., 2024). Separately, slow feature analysis extracts temporally persistent coordinates by suppressing rapid variation (Wiskott & Sejnowski, 2002). SPACE connects these ideas: a frozen multi-horizon JEPA evaluates the futures supported by alternative memories, while slow coordinates derived from its prediction trajectories define the predictive state space used for counterfactual eviction control.

3 METHOD

SPACE treats memory eviction as sparse control over the predictive state supported by memory. Future utility alone may permit avoidable state drift, whereas indiscriminately stabilizing the current state may prevent adaptation to genuine regime change. SPACE uses Reservoir as the nominal transition and intervenes only when a future-useful alternative can correct an avoidable departure from the current predictive basin. Figure 2 summarizes the framework. At each full-memory update, the incoming event and K stored events define $K + 1$ counterfactual eviction actions. A shared frozen multi-horizon JEPA predicts the futures supported by the resulting memories. These predictions feed complementary utility and predictive-basin branches, which assess departure, utility admissibility, and basin reachability. SPACE then chooses a corrective eviction or Reservoir fallback, triggering boundary release when the previous basin becomes unreachable. During deployment, all components remain frozen; only the memory, basin prototype, and causal statistics evolve online.

3.1 MEMORY AS PREDICTIVE DYNAMICS

The effect of removing an event depends on which other events remain. We therefore evaluate each feasible post-eviction memory by the future representations it supports. This makes the candidate-induced predictive state, rather than an isolated stored event, the object of control in SPACE. A streaming model maintains K timestamped long-term events $M_{t-1} = \{(m_i, \tau_i)\}_{i=1}^{K}$. Here, t indexes memory updates, x_t is the event that has just left the short-term window, and the next L observed tokens form the more recent, non-overlapping causal context $C_t = (c_{t,1}, \ldots, c_{t,L})$. Events are inserted directly while free slots remain. Once the memory is full, an action $a \in \mathcal{A}_t = \{0, \ldots, K\}$ removes one item from $M_{t-1} \cup \{x_t\}$, inducing

$$M_t^{(a)} = \mathrm{Evict}(M_{t-1} \cup \{x_t\}, a). \tag{1}$$

Thus, one action rejects x_t, while the other K actions replace an existing slot.

A shared multi-horizon JEPA predictor (Assran et al., 2023; Bardes et al., 2024) evaluates every candidate memory:

$$\hat{z}_{t,h}^{(a)} = f_\theta(M_t^{(a)}, C_t, h), \qquad h \in \mathcal{H}. \tag{2}$$

Because all actions share f_θ and C_t, differences among their predicted future latents isolate the effect of eviction. SPACE uses these differences to estimate both future utility and movement in the predictive state space; no ground-truth action labels or boundaries enter the controller.

3.2 PREDICTIVE ATTRACTOR VIA COUNTERFACTUAL EVICTION

Future utility and predictive stability address different failure modes and should not be optimized in isolation. Utility alone may favor an action that preserves predictive accuracy but causes avoidable state departure, whereas minimizing state change may make memory too inert to follow a genuine regime shift. SPACE therefore combines the signals asymmetrically rather than as a single score: future utility determines admissible alternatives, whereas predictive-basin geometry determines whether intervention is necessary and the current regime remains reachable. We denote the counter-based Reservoir action by a_R and use it as the nominal transition.

Counterfactual predictive utility. The utility branch first determines whether there is evidence that a candidate preserves future predictions better than the nominal Reservoir action. On training streams, the multi-horizon future-prediction cost of candidate a is

$$u_t(a) = \sum_{h \in \mathcal{H}} w_h \left[1 - \cos(\hat{z}_{t,h}^{(a)}, z_{t+h})\right], \tag{3}$$

where w_h are normalized horizon weights. We retain only anchors whose prediction and rollout targets all remain within the video. To account for delayed consequences, we first execute a and then follow Reservoir under the same subsequent observations and random draws. The resulting rollout cost is $q_t^{(R)}(a) = R^{-1} \sum_{j=0}^{R-1} u_{t+j}(a \to a_R)$. The utility bank stores the standardized relative target

$$\Delta C_t(a, a_R) = \beta \frac{u_t(a) - u_t(a_R)}{\sigma_{\mathrm{short}}} + (1 - \beta) \frac{q_t^{(R)}(a) - q_t^{(R)}(a_R)}{\sigma_{\mathrm{roll}}}, \tag{4}$$

where each scale is estimated from the training bank. Negative values favor a over Reservoir; SPACE uses the sign-reversed quantity as a positive utility advantage. Computing this target requires future observations and is therefore restricted to offline preparation. We store each training target together with the corresponding action feature

$$\phi_t(a) = \mathrm{concat}\Big(\mathrm{vec}_h(W_\theta \hat{z}_{t,h}^{(a)}), W_\theta v_t^{(a)}, W_\theta \bar{c}_t, \log(1 + \gamma_t^{(a)})\Big), \tag{5}$$

where W_θ is the frozen JEPA input projector, $v_t^{(a)}$ and $\gamma_t^{(a)}$ denote the evicted event and its age, respectively, and $\bar{c}_t$ is the mean context. We collect the bank by enumerating actions along train-only FIFO and Reservoir trajectories. At deployment, SPACE computes the same feature from the observed prefix and retrieves similar training states and actions to estimate the counterfactual cost, which is no longer observable online. The utility branch therefore provides a support-aware admissibility test; it does not by itself select the executed action.

Predictive basin dynamics. A utility-admissible action may still cause an unnecessary state transition, whereas a large state change may instead reflect a genuine change in the underlying predictive regime. Distinguishing these cases requires a temporal reference that suppresses routine local variation while retaining persistent predictive change. We therefore construct a second signal from the temporal structure of the same JEPA predictions. Let $y_{t,h}^{(a)} = \mathrm{norm}(W_\theta \hat{z}_{t,h}^{(a)})$, and write $y_{t,h}$ for the corresponding prediction realized along a Reservoir trajectory. From train-only Reservoir trajectories, we compute the lag-δ displacement covariance

$$\Sigma_\delta = \mathbb{E}_p \, \mathbb{E}_{t,h|p} \left[(y_{t,h} - y_{t-\delta,h})(y_{t,h} - y_{t-\delta,h})^\top \right], \tag{6}$$

where participants p receive equal weight. The r leading generalized eigenvectors of

$$\Sigma_{\delta_l} v = \eta(\Sigma_{\delta_s} + \epsilon I)v, \qquad \delta_l > \delta_s, \tag{7}$$

ordered by decreasing η, form P_{slow}. These directions exhibit low short-lag change relative to long-lag displacement, thereby retaining persistent predictive evolution while suppressing routine local variation. Applying the shared basis to every horizon yields the action-conditioned predictive state

$$s_t^{(a)} = \mathrm{norm}\Big(\mathrm{vec}_h[P_{\mathrm{slow}}^\top y_{t,h}^{(a)}] \Big). \tag{8}$$

The slow predictive state provides coordinates for the dynamics, but the controller also requires a causal reference for the regime currently supported by memory. SPACE represents this regime by a unit-norm basin prototype μ_t. For candidate a, its distance to the current basin is $d_t(a) = 1 - \langle s_t^{(a)}, \mu_{t-1} \rangle$; smaller values indicate a more conservative transition. After executing a_t,

$$\mu_t = \mathrm{norm}\Big((1 - \alpha_t)\mu_{t-1} + \alpha_t s_t^{(a_t)} \Big). \tag{9}$$

During normal operation, $\alpha_t = \alpha_{\mathrm{slow}}$, so the prototype tracks only persistent predictive change. The Reservoir-selected state at the first full-memory update initializes μ. SPACE continues to follow Reservoir during the calibration warm-up period while accumulating the causal histories used by the controller. We write $\mathrm{RZ}(v; \mathcal{D})$ for a causal robust z-score computed from prior values in $\mathcal{D}$ using their median and median absolute deviation.

Sparse eviction control. At test time, SPACE combines the two branches using utility-support, departure, and reachability tests. A two-stage cosine retrieval first identifies similar training states and then matches candidate actions within that neighborhood. Let $\widehat{A}_t(a, a_R)$ be the retrieved mean advantage after subtracting λ times the combined standard error of a and a_R. Retrieval support $\rho_t(a)$ is the mean cosine similarity of the matched training actions. SPACE admits only candidates for which both the candidate and the Reservoir reference are sufficiently supported,

$$\mathcal{A}_t^U = \{a_R\} \cup \{a : \widehat{A}_t(a, a_R) \geq \tau_U, \ \min(\rho_t(a), \rho_t(a_R)) \geq \tau_S\}. \tag{10}$$

Utility admissibility alone is not a sufficient reason to override Reservoir, because ordinary updates may cause small and harmless predictive-state changes. We therefore next test whether the nominal transition induces a departure that another feasible action could have avoided. Across all $K + 1$ actions, the restoration score $r_t = [d_t(a_R) - \min_a d_t(a)]_+$ measures whether the default Reservoir transition produces an avoidable departure. Let $\mathcal{R}_{t-1}$ contain the preceding restoration scores. The corresponding causal gate is $g_t^{\mathrm{dep}} = \sigma((\mathrm{RZ}(r_t; \mathcal{R}_{t-1}) - \tau_D)/T_D)$. The candidate considered for intervention is $\tilde{a}_t = \arg\min_{a \in \mathcal{A}_t^U} d_t(a)$. Let $\mathcal{D}_{t-1}$ contain the previously realized basin distances and define $z_t^{\mathrm{reach}} = \mathrm{RZ}(d_t(\tilde{a}_t); \mathcal{D}_{t-1})$. Both histories exclude the current update. SPACE selects

$$a_t = \begin{cases} \tilde{a}_t, & g_t^{\mathrm{dep}} \geq \tau_G \ \wedge z_t^{\mathrm{reach}} \leq \tau_R, \\ a_R, & \text{otherwise.} \end{cases} \tag{11}$$

This rule distinguishes three cases. When the departure gate is inactive, SPACE retains the nominal Reservoir transition. When the gate is active and a utility-admissible candidate remains reachable, SPACE executes the candidate closest to the current basin. If $z_t^{\mathrm{reach}} > \tau_R$, SPACE follows a_R and triggers boundary release by setting $\alpha_t = \alpha_{\mathrm{boundary}}$; otherwise, $\alpha_t = \alpha_{\mathrm{slow}}$. Thus, SPACE corrects supported departures within a reachable basin and rapidly relocates the prototype only when that basin becomes unreachable.

3.3 Learning and Deployment

Offline preparation separates predictive representation learning from controller construction. First, we train the causal Transformer JEPA on sampled training-stream memory configurations using a weighted multi-horizon cosine prediction loss. This stage optimizes θ using only future-prediction targets and requires no counterfactual action supervision. Second, after freezing the predictor, we replay train-only Reservoir trajectories and fit P_{slow} from their projected prediction dynamics. Third, at sampled full-memory states from train-only FIFO and Reservoir trajectories, we enumerate all actions. For each candidate memory $M_t^{(a)}$, the same prediction loss defines the action-conditioned cost $u_t(a)$, from which we construct the counterfactual utility target and store it together with the corresponding action feature in the utility bank. At deployment, the encoder, JEPA, slow basis, and utility bank remain frozen. Each full-memory update evaluates all $K+1$ candidates in a single JEPA batch, retrieves their utility estimates, applies Eq. 11, and updates only the causal basin prototype and its histories through Eq. 9. Future observations are used only to construct offline training targets; every deployment decision depends solely on the observed prefix and causal statistics.

4 Experiments

4.1 Implementation Details

Unless specified otherwise, all methods use a memory capacity of $K = 16$ and prediction horizons $\mathcal{H} = \{1, 4, 16, 64\}$. SPACE uses a rank-32 slow predictive basis with $(\delta_s, \delta_l) = (1, 32)$, an $R = 32$ rollout, and equal standardized weights $\beta = 0.5$ for the one-step and rollout costs. We set $(\alpha_{\mathrm{slow}}, \alpha_{\mathrm{boundary}}) = (0.05, 1)$ and $(\tau_U, \tau_S, \tau_G, \tau_R) = (0.05, 0.5, 0.5, 3.0)$. The structural settings were fixed in advance. Controller thresholds were selected once on Assembly101 using the prespecified procedure in Appendix A and were then fixed across all cross-dataset experiments. Full details of predictor optimization, retrieval and calibration settings, and hyperparameter sensitivity are provided in Appendix A and Appendix B.

4.2 MABS-Bench

Conventional downstream metrics measure what a fixed-capacity memory retains, but not how its state evolves under continual updates or responds to disruptions. We therefore introduce **MABS-Bench** (**M**emory **A**daptation and **B**ehavioral **S**tability Benchmark), which jointly evaluates future-task sufficiency, within-regime predictive stability, transition responsiveness, and perturbation recovery. Together, these dimensions characterize whether a memory remains useful and stable under routine variation, adapts to genuine semantic changes, and recovers from controlled disruptions.

Dataset and evaluation protocol. We conduct the main evaluation on Assembly101 (Sener et al., 2022) using the participant-disjoint validation split and 24-way multi-label verb prediction. Official TSM (Lin et al., 2019) representations are sampled every 0.5 seconds, so the prediction horizons $\mathcal{H} = \{1, 4, 16, 64\}$ correspond to 0.5, 2, 8, and 32 seconds, respectively. The predictor, slow basis, utility bank, and downstream action decoder are fitted on the training split. The same task–stability trend holds on Breakfast Actions (Kuehne et al., 2014), IKEA ASM (Ben-Shabat et al., 2021), and EPIC-KITCHENS-100 (Damen et al., 2022) (Appendix D).

All policies share the same causal stream, encoder, predictor, memory capacity, and downstream evaluator. At time t, each receives the observed prefix, current memory, and corresponding predictor outputs. Measurements are averaged within recordings and equally weighted across participants. Gains are computed against Reservoir on the same recording. The main table reports mean $\pm$ sample standard deviation over three complete predictor-and-policy seeds. Participant-level paired inference is detailed in Appendix A.2.

To evaluate recovery from controlled disruptions, we intervene within an Assembly101 semantic regime by replacing either 25% or 50% of the memory slots with representations from an earlier, semantically different state in the same video. The intervention preserves timestamps and slot metadata. The clean and perturbed branches subsequently receive the same video stream, and their predictive-state trajectories are compared at offsets $h \in \{0, 1, 2, 4, 8, 16\}$. Multiple interventions are first averaged within each recording and then aggregated across participants.

Compared methods. We compare SPACE with standard eviction policies, predictor-guided policies, recent memory-selection methods, and component variants of SPACE. Standard policies include FIFO, Reservoir sampling (Vitter, 1985), Surprise, and Similarity, whereas predictor-guided baselines include Attention and Self-Influence. We further adapt StreamForest (Zeng et al., 2025), OmniMem (Sun et al., 2026), ObjectStream (Dong et al., 2026), and NovaCov (Duan et al., 2026) to the shared MABS-Bench interface and K-slot budget. Adaptation details and scope are provided in Appendix A.3; these are controlled common-interface adaptations and are not intended to establish the relative performance of the original systems in their native settings. Utility-only applies counterfactual predictive utility without basin geometry, whereas State-only applies departure and reachability control without utility filtering.

4.3 Evaluation Metrics

Future-task sufficiency. We measure whether the retained memory supports future-action prediction. **Action NLL** is the negative log-likelihood of the ground-truth future labels under the action decoder applied to JEPA predictions. We average NLL across the four prediction horizons and report horizon-specific gains at H1, H4, H16, and H64. We additionally report **mAP**, **LRAP**, and **Recall@5** for multi-label prediction.

Basin stability and boundary adaptation. Let s_t denote the predictive state produced by a memory policy. For Assembly101, let $q_t \in \{0,1\}^{24}$ denote the evaluation-only multi-hot vector of annotated verbs active at time t. We restrict evaluation to foreground regimes. Within-regime transitions retain the same nonempty active-label set, whereas semantic boundaries comprise foreground onsets and changes in the active-label set. Transitions from foreground to background are excluded because background denotes the absence of an annotated action rather than a coherent destination regime. **Basin drift** measures the expected cosine distance between consecutive predictive states within the same annotated regime:

$$D_{\text{within}} = \mathbb{E}[1 - \cos(s_t, s_{t+1}) \mid q_t = q_{t+1} \neq \mathbf{0}]\,. \tag{12}$$

Lower basin drift indicates greater within-regime stability. For adaptation, **boundary selectivity** compares predictive-state displacement across semantic boundaries with within-regime drift:

$$S_{\text{boundary}} = \log\left(\frac{D_{\text{boundary}} + \epsilon}{D_{\text{within}} + \epsilon}\right),$$

where D_{boundary} is the mean cosine displacement over foreground onsets and foreground label-set changes as defined above. Higher selectivity indicates that the predictive state changes more strongly at semantic boundaries than within a regime.

Perturbation recovery. For a controlled memory perturbation, let d_h denote the cosine distance between the perturbed predictive state and the paired clean state after h subsequent updates. **Predictive recovery** is $\mathrm{Rec}(h) = (d_0 - d_h)/\max(d_0, \epsilon)$, where $\epsilon = 10^{-8}$ is used only as a numerical safeguard. Larger values indicate stronger contraction toward the clean trajectory. **Recovery AUC** summarizes the trajectory as the normalized area under $\mathrm{Rec}(h)$ across the evaluated offsets. We further report **memory recovery** based on the convergence of the perturbed and clean memory sets, and **task recovery** based on the convergence of their future-action predictions.

5 Results

5.1 Main Results on MABS-Bench

Table 1 shows that SPACE achieves the strongest utility–stability trade-off among the compared methods. It achieves the best or tied-best performance on every task metric, with its clearest advantage at H64, while reducing within-regime drift relative to Reservoir in all three seeds and achieving the highest boundary selectivity. The paired gains are consistent across participants: Action NLL improves for 8 of 10 participants (95% CI $[0.0010, 0.0039]$, $p = 0.0059$), and H64 improves for 9 of 10 participants ($p = 0.0029$; Appendix C). Although Attention and State-only achieve lower absolute drift, both degrade Action NLL and H64 relative to Reservoir, indicating that minimizing state variation alone does not preserve useful future information. An independent audit evaluator shows similar task and drift trends under an independent predictive geometry (Appendix G).

Table 1: MABS-Bench results on Assembly101 (Sener et al., 2022) over three complete predictor-and-policy seeds. Values are mean ± sample standard deviation. H64 gain is paired against Reservoir within the same seed. Best mean values are **bold** and second-best mean values are underlined.

Method	Action NLL ↓	mAP ↑	LRAP ↑	Recall@5 ↑	H64 gain ↑	Basin drift ($\times 10^{-5}$) ↓	Boundary selectivity ↑
FIFO	3.6820 ± 0.0091	0.1339 ± 0.0005	0.4743 ± 0.0017	0.7030 ± 0.0012	-0.0561 ± 0.0246	8.7180 ± 2.343	0.8229 ± 0.0220
Reservoir	3.6315 ± 0.0209	0.1466 ± 0.0012	0.4875 ± 0.0021	0.7123 ± 0.0026	0.0000 ± 0.0000	8.9420 ± 2.429	0.8177 ± 0.0206
Surprise	3.6633 ± 0.0218	0.1443 ± 0.0008	0.4797 ± 0.0026	0.7080 ± 0.0019	-0.0590 ± 0.0075	9.3870 ± 2.486	0.8135 ± 0.0180
Similarity	3.6451 ± 0.0201	0.1459 ± 0.0010	0.4841 ± 0.0022	0.7109 ± 0.0020	-0.0179 ± 0.0022	8.8410 ± 2.488	0.8178 ± 0.0232
Attention	3.6632 ± 0.0264	0.1375 ± 0.0022	0.4793 ± 0.0019	0.7060 ± 0.0012	-0.0296 ± 0.0162	$\mathbf{8.1840 \pm 2.344}$	0.8254 ± 0.0187
Self-Influence	3.6391 ± 0.0218	$\mathbf{0.1468 \pm 0.0013}$	0.4870 ± 0.0021	0.7115 ± 0.0011	0.0033 ± 0.0022	8.3530 ± 2.492	0.8155 ± 0.0203
StreamForest	3.6557 ± 0.0205	0.1446 ± 0.0005	0.4814 ± 0.0023	0.7093 ± 0.0028	-0.0401 ± 0.0034	9.1000 ± 2.492	0.8247 ± 0.0188
OmniMem	3.6398 ± 0.0205	$\mathbf{0.1468 \pm 0.0009}$	0.4860 ± 0.0023	0.7110 ± 0.0024	-0.0161 ± 0.0015	9.0510 ± 2.460	0.8177 ± 0.0233
ObjectStream	3.6657 ± 0.0218	0.1438 ± 0.0014	0.4798 ± 0.0023	0.7084 ± 0.0026	-0.0577 ± 0.0036	9.1320 ± 2.514	0.8208 ± 0.0197
NovaCov	3.6610 ± 0.0244	0.1376 ± 0.0014	0.4783 ± 0.0021	0.7073 ± 0.0006	-0.0291 ± 0.0167	8.6080 ± 2.392	0.8214 ± 0.0222
Utility-only	3.6403 ± 0.0294	0.1437 ± 0.0019	0.4858 ± 0.0013	0.7119 ± 0.0007	-0.0051 ± 0.0227	8.8590 ± 2.387	0.8123 ± 0.0176
State-only	3.6522 ± 0.0299	0.1401 ± 0.0012	0.4822 ± 0.0019	0.7094 ± 0.0003	-0.0172 ± 0.0224	8.3400 ± 2.347	0.8263 ± 0.0172
SPACE	$\mathbf{3.6291 \pm 0.0209}$	$\mathbf{0.1468 \pm 0.0011}$	$\mathbf{0.4883 \pm 0.0018}$	$\mathbf{0.7131 \pm 0.0028}$	$\mathbf{0.0068 \pm 0.0016}$	8.6900 ± 2.421	$\mathbf{0.8270 \pm 0.0219}$

Table 2: Recovery from controlled memory perturbations on Assembly101. Predictive recovery measures contraction toward the paired clean trajectory, and recovery AUC summarizes offsets H1–H16. Best values within each perturbation level are **bold** and second-best values are underlined.

Corruption	Method	Predictive AUC ↑	H16 recovery ↑	Memory AUC ↑	Task AUC ↑
25%	Reservoir	**0.0328**	**0.1209**	0.0417	0.0399
	Utility-only	−0.3954	−0.8692	0.0191	−0.2098
	State-only	−0.3740	−0.8425	**0.0533**	−0.3587
	SPACE	0.0309	0.0727	0.0426	**0.0517**
50%	Reservoir	0.0376	0.0770	0.0239	0.0308
	Utility-only	−0.4916	−0.7216	0.0082	−0.3305
	State-only	−0.2861	−0.7355	**0.0534**	−0.7077
	SPACE	**0.0525**	**0.0881**	0.0218	**0.0485**

5.2 Perturbation Recovery and Future Semantics

Recovery from controlled perturbations. Table 2 evaluates recovery after replacing 25% or 50% of the memory slots with representations from earlier, semantically different states. SPACE yields positive predictive, memory, and task recovery at both corruption levels. At 25%, Reservoir achieves higher predictive AUC and H16 recovery, whereas SPACE achieves the highest task AUC. Because this milder intervention produces smaller initial separations d_0, the normalized ordering is more sensitive to the denominator; we therefore interpret this result as evidence of positive recovery rather than as evidence that SPACE outperforms Reservoir. At 50%, SPACE achieves the highest predictive AUC, H16 recovery, and task AUC. The predictive advantage of SPACE also remains positive under the minimum-d_0 sensitivity analysis in Appendix C.

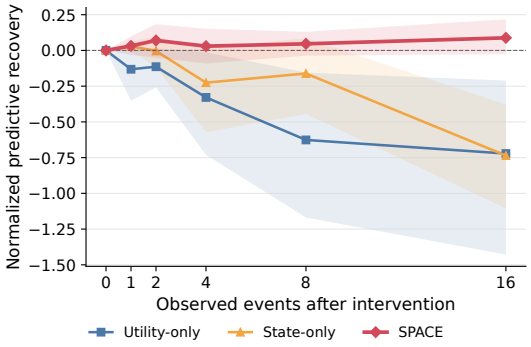


Figure 3: Predictive-state recovery after 50% perturbations. Positive values denote contraction toward the paired clean trajectory; shading shows 95% participant-bootstrap confidence intervals.

Figure 3 shows the recovery trajectories under 50% corruption. Utility-only and State-only progressively diverge from their paired clean trajectories, whereas SPACE maintains positive contraction. This contrast supports combining counterfactual future utility with predictive-basin control.

Future-semantic visualization. Figure 4 visualizes the future representations supported by memories produced by Reservoir and SPACE on EPIC-KITCHENS-100 (Damen et al., 2022). At H64, the ground-truth activity is *stir aubergine*. SPACE retrieves *stir food*, preserving the long-term action semantics, whereas Reservoir retrieves *put down celery*. This semantic difference is consistent with the lower H64 action-prediction loss of SPACE for the same observed context.

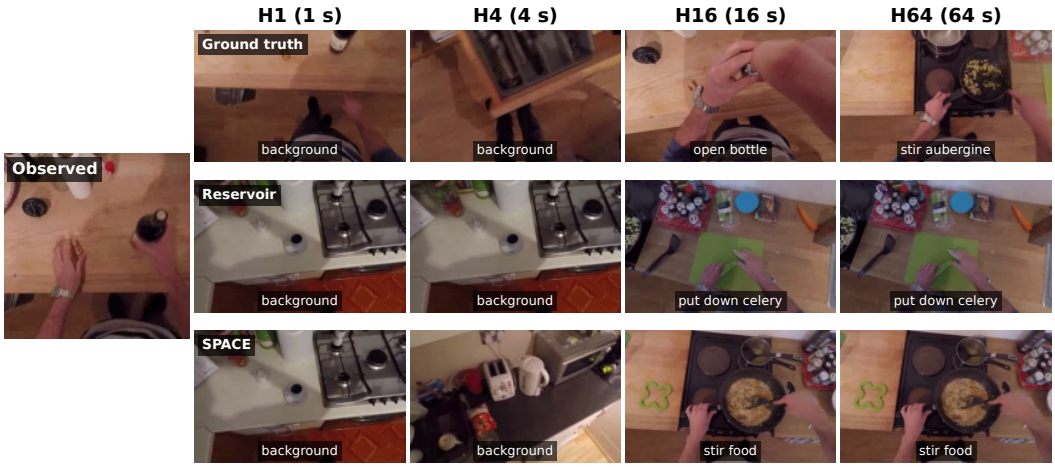


Figure 4: Future-semantic visualization on EPIC-KITCHENS-100. The left column shows the observed context, followed by predictions at H1, H4, H16, and H64. The rows show the ground-truth future and the nearest training frames retrieved using future representations predicted from Reservoir and SPACE memories.

Table 3: Component ablation on Assembly101 over predictor-and-policy seeds. Within each seed, all variants share the same stream, predictor, utility bank, memory capacity, and downstream decoder. Horizon-specific NLL and drift gains are paired against Reservoir. Values are mean ± sample standard deviation. Best mean values are **bold** and second-best mean values are underlined.

Variant	Action NLL ↓	H1 gain ($\times 10^{-2}$) ↑	H4 gain ($\times 10^{-2}$) ↑	H16 gain ($\times 10^{-2}$) ↑	H64 gain ($\times 10^{-2}$) ↑	Drift gain ($\times 10^{-6}$) ↑	Boundary selectivity ↑
SPACE	$\mathbf{3.6291 \pm 0.0209}$	$\mathbf{0.0189 \pm 0.0131}$	$\mathbf{-0.0007 \pm 0.0015}$	$\mathbf{0.2574 \pm 0.0029}$	0.6844 ± 0.1626	2.5203 ± 0.0804	$\mathbf{0.8270 \pm 0.0219}$
w/o Departure gate	3.6434 ± 0.0153	-2.1477 ± 0.0318	-2.2982 ± 1.6290	-1.7154 ± 0.0896	$\mathbf{1.4013 \pm 0.7037}$	0.7356 ± 0.1854	0.8249 ± 0.0003
w/o Utility gate	3.6522 ± 0.0299	-1.6018 ± 0.5118	-2.6354 ± 0.7932	-2.3207 ± 0.2242	-1.7221 ± 2.2370	$\mathbf{6.0203 \pm 0.8612}$	0.8263 ± 0.0172
w/o Reachability gate	3.6314 ± 0.0207	-0.2293 ± 0.0042	-0.2801 ± 0.0028	0.0713 ± 0.0120	0.4780 ± 0.4729	0.5623 ± 0.0147	0.8199 ± 0.0164
Full-latent predictive state	3.6431 ± 0.0210	-1.4174 ± 0.0020	-1.4993 ± 0.0873	-1.1734 ± 0.0199	-0.5559 ± 0.0716	0.5138 ± 0.0085	0.8261 ± 0.0097
w/o Boundary release	3.6437 ± 0.0209	-1.4370 ± 0.0057	-1.5063 ± 0.0991	-1.1458 ± 0.0380	-0.7908 ± 0.0203	0.3192 ± 0.0084	0.8268 ± 0.0172

5.3 Ablation Analysis

Table 3 isolates the roles of predictive utility and state control. Removing the utility gate produces the clearest failure mode: task performance and H64 deteriorate despite substantially greater drift reduction, indicating that contraction alone does not preserve useful future information. The remaining ablations primarily redistribute this trade-off: some improve an individual horizon or drift measure, but SPACE retains the lowest mean Action NLL and the most balanced multi-horizon stability profile. The comparison with the full-latent predictive state is also consistent with the slow basis suppressing local variation without sacrificing aggregate utility. SPACE is therefore designed to balance future utility and stability rather than maximize contraction or performance at any single horizon. Because several differences are small relative to their across-seed standard deviations, we interpret fine-grained differences among variants descriptively.

6 Conclusion

We introduced SPACE, which treats fixed-capacity streaming memory eviction as counterfactual control over future-predictive dynamics. By combining counterfactual future utility with predictive-basin control, SPACE is designed to preserve useful future information while limiting avoidable predictive-state drift and remaining responsive to genuine regime changes. Together with MABS-Bench, our experiments show that, among the compared policies, this formulation provides a better balance between downstream utility and predictive-state dynamics, while the resulting memory dynamics remain recoverable after controlled perturbations. More broadly, these results support evaluating streaming memory not only by what it retains, but also by how its predictive state evolves under continual updates. SPACE currently relies on a predictor and utility bank trained offline and frozen during deployment; adapting these components under larger distribution shifts remains an important direction for future work on adaptive streaming memory.

AI USE STATEMENT

In this work, we used generative AI tools to assist with the interpretation and consistency checking of experimental results. We did not use generative AI tools to develop the core research idea or conceptual framework, propose or refine research hypotheses, or conceive the proposed method. Additionally, we used generative AI tools to assist with polishing parts of the paper and with literature retrieval and discovery. All AI-assisted analyses and text were reviewed and verified by the authors, and literature identified with AI assistance was checked against the original sources. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the methodological and experimental details required to reproduce our results in the main text and appendices. Section 3 describes SPACE and its learning and deployment procedure, while Section 4 specifies the evaluation protocol and main experimental settings. Appendix A provides the complete implementation configuration, parameter-selection protocol, and statistical procedures. The remaining appendices report sensitivity analyses, robustness and validation studies, cross-dataset evaluations, and runtime measurements. An anonymized implementation, including the training pipeline and reported configurations, is provided in the supplementary material.

REFERENCES

Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. *Advances in neural information processing systems*, 32, 2019a.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019b.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629. IEEE, 2023.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. Featured Certification.

Yizhak Ben-Shabat, Xin Yu, Fatemeh Saleh, Dylan Campbell, Cristian Rodriguez-Opazo, Hongdong Li, and Stephen Gould. The ikea asm dataset: Understanding people assembling furniture through actions, objects and pose. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 846–858. IEEE, 2021.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Evangelos Kazakos, Jian Ma, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. *International Journal of Computer Vision*, 130(1):33–55, 2022.

Anxhelo Diko, Tinghuai Wang, Wassim Swaileh, Shiyan Sun, and Ioannis Patras. Rewind: Understanding long videos with instructed learnable memory. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13734–13743. IEEE, 2025.

Mingkang Dong, Muxin Pu, Jie Li, Bohan Guo, Songruo Chen, Bin Ren, Xu Zheng, Chen Zhao, Tianwen Qian, Mohamed Elhoseiny, et al. Objectstream: Latent objects as memory anchors for streaming video understanding. *arXiv preprint arXiv:2607.28312*, 2026.

Moxu Duan, Jingwen Fu, and Yuwang Wang. Think in sets for streaming video token compression. *arXiv preprint arXiv:2608.01169*, 2026.

Haonan Ge, Yiwei Wang, Hang Wu, and Yujun Cai. What should a streaming video model remember? *arXiv preprint arXiv:2606.16353*, 2026.

Bo He, Hengduo Li, Young Kyun Jang, Menglin Jia, Xuefei Cao, Ashish Shah, Abhinav Shrivastava, and Ser-Nam Lim. Ma-lmm: Memory-augmented large multimodal model for long-term video understanding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13504–13514. IEEE, 2024.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Zhenpeng Huang, Xinhao Li, Jiaqi Li, Jing Wang, Xiangyu Zeng, Cheng Liang, Tao Wu, Xi Chen, Liang Li, and Limin Wang. Online video understanding: Ovbench and videochat-online. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3328–3338. IEEE, 2025.

Minsoo Kim, Kyuhong Shim, Jungwook Choi, and Simyung Chang. Infinipot-v: Memory-constrained KV cache compression for streaming video understanding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 780–787, 2014.

Yann LeCun et al. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 7083–7093, 2019.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Rui Qian, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Shuangrui Ding, Dahua Lin, and Jiaqi Wang. Streaming long video understanding with large language models. *Advances in Neural Information Processing Systems*, 37:119336–119360, 2024.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*, 2020.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in neural information processing systems*, 32, 2019.

Fadime Sener, Dibyadip Chatterjee, Daniel Shelepov, Kun He, Dipika Singhania, Robert Wang, and Angela Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21064–21074. IEEE, 2022.

Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, et al. Moviechat: From dense token to sparse memory for long video understanding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18221–18232. IEEE, 2024.

Guangzhi Sun, Yixuan Li, Yudong Yang, and Chao Zhang. Omnimem: Perturbation-aware memory compression for streaming audio-visual llms. *arXiv preprint arXiv:2606.07577*, 2026.

Lee Susman, Naama Brenner, and Omri Barak. Stable memory with unstable synapses. *Nature communications*, 10(1):4441, 2019.

Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985.

Kai Wang, Zhaopeng Gu, Yixiang Chen, Yuan Xu, Qisen Ma, Jiabing Yang, Zhaowen Li, Yan Huang, Liang Wang, and Peng Su. Dim-wam: World-action modeling with diverse historical event memory. *arXiv preprint arXiv:2606.27677*, 2026.

Laurenz Wiskott and Terrence J. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002. doi: 10.1162/089976602317318938.

Chao-Yuan Wu, Yanghao Li, Karttikeya Mangalam, Haoqi Fan, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Memvit: Memory-augmented multiscale vision transformer for efficient long-term video recognition. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13577–13587. IEEE, 2022.

Mingze Xu, Yuanjun Xiong, Hao Chen, Xinyu Li, Wei Xia, Zhuowen Tu, and Stefano Soatto. Long short-term transformer for online action detection. *Advances in Neural Information Processing Systems*, 34:1086–1099, 2021.

Xiangyu Zeng, Kefan Qiu, Qingyu Zhang, Xinhao Li, Jing Wang, Jiaxin Li, Ziang Yan, Kun Tian, Meng Tian, Xinhai Zhao, Yi Wang, and Limin Wang. Streamforest: Efficient online video understanding with persistent event memory. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, and Xiaojie Jin. Flash-vstream: Efficient real-time understanding for long video streams. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21059–21069. IEEE, 2025.

A Implementation and Reproducibility Details

This appendix provides the full implementation configuration, dataset-specific evaluation protocols, statistical aggregation and inference procedures, hyperparameter sensitivity analyses, and robustness audits.

A.1 Complete Configuration and Selection Protocol

Table 4 summarizes the default configuration. Structural choices, including the prediction horizons, slow-basis construction, and rollout definition, were fixed in advance. Controller thresholds were selected once on the Assembly101 validation split using a prespecified feasibility-first protocol and were then frozen for all cross-dataset experiments. Feasibility was assessed before performance ranking, and no weighted composite score was used. The slow basis and utility bank were constructed exclusively from training streams.

Table 4: Default implementation configuration. Unless stated otherwise, the same settings are used across datasets.

Component	Parameter	Value
JEPA predictor	Hidden dimension	512
	Transformer layers / heads	$4/8$
	Feed-forward dimension	2048
	Recent-context length	8
	Prediction horizons $\mathcal{H}$	$\{1, 4, 16, 64\}$
	Horizon weights w_h	uniform $(1/4)$
	Dropout / memory dropout	$0.1/0.15$
	Training epochs / batch size	$30/128$
	AdamW learning rate / weight decay	$3 \times 10^{-4}/0.05$
	Learning-rate schedule / warmup	cosine / 5% linear
	Numerical precision	bfloat16
Slow basis	Rank r	32
	Short / long lag (δ_s, δ_l)	$(1, 32)$
	Covariance regularization	10^{-3}
	Participant weighting	equal
	Basis sharing across horizons	yes
Utility retrieval	Memory capacity K	16
	Rollout length R	32
	One-step / rollout weight β	0.5
	State / action neighbors	$32/32$
	Retrieval similarity	cosine
	Uncertainty multiplier λ	1.0
Controller	Prototype rates $(\alpha_{\text{slow}}, \alpha_{\text{boundary}})$	$(0.05, 1.0)$
	Utility / support thresholds (τ_U, τ_S)	$(0.05, 0.5)$
	Departure threshold / temperature (τ_D, T_D)	$(1.0, 0.5)$
	Restoration / reachability thresholds (τ_G, τ_R)	$(0.5, 3.0)$
	History length / warmup	$32/16$
	Minimum robust scale	10^{-5}

A.2 Statistical Aggregation and Inference

For each seed, measurements are first averaged within each recording and then across recordings within each participant. Participants receive equal weight regardless of the number of recordings. A seed-level estimate is obtained by averaging the participant-level values, and the main tables report the mean and sample standard deviation of these estimates across three seeds.

For paired inference, each method and Reservoir are evaluated on the same recordings. Their recording-level measurements are first differenced within each seed, then averaged within each participant, and finally averaged across seeds. We resample the 10 participants with replacement for

10,000 bootstrap replicates and report the 2.5th and 97.5th percentiles as the 95% confidence interval. We additionally apply an exact one-sided sign-flip test to the 10 participant-level paired gains, with the alternative that their mean is greater than zero. Because controller thresholds are selected on the Assembly101 validation split, these intervals and tests are reported as post-selection measures of participant-level consistency rather than as fully held-out confirmatory inference.

A.3 Adaptation of Baselines

We implement literature-inspired memory adapters under the shared MABS-Bench interface. All adapters process the same frozen visual features and causal stream, maintain at most K long-term vectors, and are evaluated with the same JEPA predictor and downstream decoder. Incoming events fill empty slots until capacity is reached. Thereafter, OmniMem and NovaCov select K vectors from the $K+1$ available candidates, whereas StreamForest and ObjectStream may either merge an incoming event into an existing slot or replace a stored vector. Thus, all methods share the same slot budget, although merging adapters may retain aggregated representations rather than individual observed events.

StreamForest-style event merging. We approximate persistent event memory with a flat collection of K event representatives. The incoming event is matched to the representative with the highest cosine similarity weighted by exponential recency decay. We use a temporal scale of 64 updates and merge when this affinity is at least 0.75. Otherwise, the representative with the lowest weighted combination of uniqueness, recency, and persistence is replaced, using weights $(0.5, 0.25, 0.25)$. Uniqueness is one minus the maximum cosine similarity to any other stored representative, and persistence is the logarithm of one plus the accumulated count. This adapter does not reconstruct the original hierarchical event forest.

OmniMem-style perturbation-aware selection. We enumerate all $K+1$ single-item evictions and obtain their multi-horizon forecasts from the shared frozen JEPA. Each candidate is scored by its mean horizon-wise cosine distance from the average candidate forecast, plus 0.1 times the retained-set redundancy. Redundancy is the mean, over retained vectors, of the maximum cosine similarity to another retained vector. The candidate with the lowest score is selected. This adapter uses a visual-only forecast-distortion surrogate rather than the original attention-output distortion criterion, audio–visual budget allocation, or KV-cache compression.

ObjectStream-style persistent anchors. We treat stored vectors as latent event anchors. An incoming event is merged into its most similar anchor when their cosine similarity is at least 0.85. Otherwise, it replaces the anchor with the lowest retention score, which combines persistence, distinctiveness, and recency with weights $(0.5, 0.3, 0.2)$. Distinctiveness uses the same nearest-neighbor uniqueness measure as above, and recency has a half-life of 64 updates. These anchors operate on event-level features; the adapter does not perform patch-level latent-object discovery or maintain object-conditioned temporal residuals.

For both merging adapters, an anchor with count n is updated with coefficient $\min(0.25, 1/(n+1))$. Its timestamp is refreshed to that of the incoming event, and its count is incremented; new anchors start with count one.

NovaCov-style set coverage. For each possible eviction, we score the retained set using weighted facility-location coverage over the $K+1$ candidate vectors and recent context. Each reference contributes its maximum cosine similarity to a retained vector. Candidate-reference weights decay with an age half-life of 64 updates, whereas recent-context references receive unit weight. We select the eviction that maximizes total coverage. This adapter retains a set-based coverage criterion but does not implement the original separate historical reference bank or dual-branch novelty-aware objective.

These comparisons evaluate the specified adapters under a common representation, causal stream, and memory budget. They should not be interpreted as establishing the relative performance of the original systems in their native settings.

Table 5: Single-seed capacity and one-factor threshold sensitivity on Assembly101. Gains are paired against Reservoir with the same memory capacity. Best values within each block are bold.

Setting	ΔNLL ↑	H16 gain↑	H64 gain↑	Drift gain ↑
$K = 8$	0.000818	0.000811	0.002413	8.833×10^{-8}
$K = 16$	**0.002387**	**0.002552**	**0.007412**	$\mathbf{2.5101\times10^{-6}}$
$K = 32$	0.001384	0.001633	0.004260	4.303×10^{-7}
Default $(\tau_U, \tau_G, \tau_R) = (0.05, 0.5, 3.0)$	**0.002387**	**0.002552**	0.007412	$\mathbf{2.5101\times10^{-6}}$
$\tau_U = 0.00$	0.002239	0.002413	**0.008183**	7.153×10^{-7}
$\tau_U = 0.10$	0.001963	0.001522	0.006591	4.254×10^{-7}
$\tau_G = 0.25$	0.002247	0.002218	0.007858	6.721×10^{-7}
$\tau_G = 0.75$	0.001334	0.000459	0.005246	3.049×10^{-7}
$\tau_R = 2.0$	0.001683	0.001544	0.005759	4.622×10^{-7}
$\tau_R = 4.0$	0.001850	0.001646	0.007291	4.625×10^{-7}

Table 6: Participant-level paired inference for SPACE relative to Reservoir on Assembly101. Gains are averaged across seeds within each participant, and positive values favor SPACE.

Metric	Mean gain	Positive	95% bootstrap CI	p
Action NLL	0.002400	8/10	$[0.00105, 0.00392]$	0.0059
H16 NLL	0.002574	8/10	$[0.00115, 0.00422]$	0.0059
H64 NLL	0.006844	9/10	$[0.00296, 0.01190]$	0.0029
Basin drift	2.5203×10^{-6}	10/10	$[1.97\times10^{-6}, 3.56\times10^{-6}]$	0.0010

B Capacity and Hyperparameter Sensitivity

Table 5 reports a single-seed diagnostic across memory capacities $K \in \{8, 16, 32\}$ and one-factor variations in the utility, departure, and reachability thresholds. SPACE retains positive NLL, H16, H64, and drift gains at every tested capacity and threshold setting. The default configuration yields the largest NLL, H16, and drift gains, whereas several alternatives yield larger H64 gains. We therefore interpret these results as evidence of local robustness rather than universal hyperparameter optimality.

C Robustness and Statistical Audits

C.1 Participant-Level Consistency

Table 6 evaluates whether the gains of SPACE over Reservoir are consistent across Assembly101 validation participants. Within each seed, recording-level paired gains are averaged within each participant; the resulting participant-level gains are then averaged across the three seeds. All metrics are oriented so that positive values favor SPACE. Action NLL and H16 gains are positive for 8 of 10 participants, the H64 gain is positive for 9 of 10, and the basin-drift gain is positive for all 10. All participant-bootstrap confidence intervals exclude zero. Equal participant weighting prevents participants with more recordings from dominating the estimates, while the positive counts show that the gains are broadly distributed across participants.

C.2 Recovery Normalization Audit

Predictive recovery is normalized by the initial distance d_0, so very small initial separations could amplify small absolute changes. Table 7 reports the intervention-level distribution of d_0 before recording- and participant-level aggregation. The analysis includes 62 valid interventions per policy at 25% corruption and 59 at 50% corruption. The numerical safeguard $\epsilon = 10^{-8}$ was never active because the minimum observed distance was 8.34×10^{-7}. Stronger corruption generally produces larger initial separations. For SPACE, the median d_0 increases from 1.37×10^{-5} at 25% corruption to 3.25×10^{-5} at 50%, while the fraction below 10^{-5} decreases from 35.5% to 5.1%.

Table 7: Distribution of the intervention-level initial predictive distance d_0. Intervals denote the 5th–95th percentiles.

Corruption	Method	Median [5th, 95th]	Minimum	Fraction with $d_0 < 10^{-5}$
25%	Reservoir	1.36×10^{-5} [2.15×10^{-6}, 6.73×10^{-5}]	1.07×10^{-6}	35.5%
25%	Utility-only	1.22×10^{-5} [2.40×10^{-6}, 7.78×10^{-5}]	1.01×10^{-6}	33.9%
25%	State-only	9.30×10^{-6} [1.14×10^{-6}, 6.81×10^{-5}]	8.94×10^{-7}	51.6%
25%	SPACE	1.37×10^{-5} [2.39×10^{-6}, 7.48×10^{-5}]	1.07×10^{-6}	35.5%
50%	Reservoir	2.74×10^{-5} [3.22×10^{-6}, 9.89×10^{-5}]	1.67×10^{-6}	13.6%
50%	Utility-only	2.36×10^{-5} [2.82×10^{-6}, 9.56×10^{-5}]	1.61×10^{-6}	22.0%
50%	State-only	1.60×10^{-5} [2.95×10^{-6}, 7.88×10^{-5}]	8.34×10^{-7}	35.6%
50%	SPACE	3.25×10^{-5} [1.03×10^{-5}, 9.96×10^{-5}]	3.16×10^{-6}	5.1%

We additionally repeat the paired SPACE–Reservoir comparison after retaining only interventions for which both methods exceed the same minimum-d_0 threshold. At 50% corruption, the SPACE-minus-Reservoir recovery-AUC gain remains positive at thresholds $2 \times 10^{-6}, 5 \times 10^{-6}, 10^{-5}$, with corresponding gains of $0.0145, 0.0276, 0.0322$, compared with 0.0150 without filtering. At 25% corruption, the paired gain changes sign across thresholds. We therefore interpret the milder setting as evidence of absolute contraction rather than a robust improvement over Reservoir.

D Cross-Dataset Results

We extend the matched causal-stream evaluation to Breakfast Actions, IKEA ASM, and EPIC-KITCHENS-100. Within each dataset, all policies process the same observed streams with a shared memory capacity, frozen predictor, and frozen decoder. Online decisions never access future frames or action labels. These comparisons isolate the effect of changing the memory policy within the fixed evaluation pipeline.

The tables report the same task and predictive-dynamics measurements as Table 1. For the single-label Breakfast and IKEA tasks, Top-1 and Top-5 accuracy replace the multi-label mAP, LRAP, and Recall@5 metrics used for Assembly101 and EPIC-KITCHENS-100. H64 gain is the paired reduction in H64 Action NLL relative to Reservoir.

Measurements are first averaged within each recording and then within each participant, with equal weight assigned to participants. Gains are computed relative to Reservoir on the same recording before aggregation. Across the evaluated configurations, SPACE improves task performance relative to Reservoir while reducing within-regime predictive drift. These participant-balanced point estimates provide complementary cross-dataset evidence, but do not quantify uncertainty from participant sampling or training randomness.

Breakfast Actions. Table 8 shows that SPACE achieves the highest Top-1 and Top-5 accuracy, the lowest basin drift, and the highest boundary selectivity. It also improves Action NLL and H64 relative to Reservoir, although Attention retains the lowest Action NLL and the largest H64 gain. These results indicate a favorable trade-off between future-task utility and predictive stability under the evaluated setting.

IKEA ASM. On IKEA ASM, SPACE achieves the lowest Action NLL, highest Top-1 accuracy, largest H64 gain, lowest basin drift, and highest boundary selectivity, while ranking second in Top-5 accuracy (Table 9). These results show that the task and predictive-dynamics benefits of SPACE transfer to a distinct single-label streaming regime rather than being specific to Assembly101.

EPIC-KITCHENS-100. Table 10 further confirms this pattern on the larger and more diverse EPIC-KITCHENS-100 benchmark. SPACE achieves the lowest Action NLL, highest LRAP, largest H64 gain, and lowest basin drift, while ranking second in mAP, Recall@5, and boundary selectivity. SPACE improves future-task utility and within-regime stability relative to Reservoir, while individual baselines remain stronger on some metrics.

Table 8: Results on Breakfast Actions under the matched causal-stream protocol. H64 gain is paired against Reservoir. Best mean values are **bold** and second-best mean values are underlined.

Method	Action NLL ↓	Top-1 ↑	Top-5 ↑	H64 gain ↑	Basin drift ($\times 10^{-3}$) ↓	Boundary selectivity ↑
FIFO	2.6320	0.3151	0.6256	0.0057	2.7040	0.3538
Reservoir	2.6474	0.3173	0.6152	0.0000	3.2163	0.2991
Surprise	2.7044	0.3046	0.6007	−0.1049	3.6405	0.3293
Similarity	2.6827	0.3063	0.6049	−0.0686	3.3748	0.3335
Attention	**2.6126**	0.3212	0.6305	**0.0571**	2.8566	0.3160
Self-Influence	2.6332	0.3145	0.6261	0.0342	2.7257	0.3426
Utility-only	2.6445	0.3185	0.6141	0.0173	3.1083	0.3233
State-only	2.6423	0.3221	0.6147	0.0140	2.9754	0.3285
SPACE	2.6198	**0.3299**	**0.6353**	0.0322	**2.5013**	**0.3730**

Table 9: Results on the IKEA ASM sealed test split under the matched causal-stream protocol. H64 gain is paired against Reservoir. Best mean values are **bold** and second-best mean values are underlined.

Method	Action NLL ↓	Top-1 ↑	Top-5 ↑	H64 gain ↑	Basin drift ($\times 10^{-3}$) ↓	Boundary selectivity ↑
FIFO	2.5686	0.3176	**0.6489**	0.0062	1.0576	0.1914
Reservoir	2.5786	0.3146	0.6482	0.0000	1.0782	0.2718
Surprise	2.6211	0.2883	0.6332	−0.0591	1.1446	0.2654
Similarity	2.6365	0.2686	0.6269	−0.0696	1.0874	0.2688
Attention	2.5206	0.3506	0.6412	0.0549	1.0085	0.2133
Self-Influence	2.5947	0.3078	0.6445	−0.0341	0.9963	0.2439
Utility-only	2.5626	0.3226	0.6451	0.0049	1.0825	0.2666
State-only	2.5604	0.3274	0.6466	0.0082	1.0263	0.2516
SPACE	**2.4889**	**0.3736**	0.6484	**0.0873**	**0.9600**	**0.2730**

E Runtime Cost and Memory Footprint

We profile the deployment-time costs of representative policies on Assembly101 with memory capacity $K = 16$. Measurements are obtained on an NVIDIA A800 80 GB GPU. Disk I/O, downstream metric computation, and trajectory logging are excluded from the timed region. Wall-clock measurements are synchronized with the GPU before and after each update.

For each of three predictor-and-policy seeds, we profile 15 recordings from five validation participants. For each recording, we measure one full-memory update after the 16-update controller calibration period, so that SPACE is profiled under its normal deployment-time decision rule rather than during calibration. Each measurement uses three untimed CUDA warm-up repetitions followed by ten timed repetitions. Repetitions are first averaged within each recording, recordings are then averaged within each participant, and participants receive equal weight. The reported standard deviation is computed over the five participant-level means after averaging across seeds.

Table 11 summarizes the deployment-time latency and memory footprint of the evaluated policies. At each full-memory update, SPACE evaluates all $K+1 = 17$ feasible actions in one batched predictor pass. It requires 9.87 ms per update, approximately $10.1\times$ the cost of Reservoir but comparable to Attention and $6.25\times$ faster than Self-Influence, which performs 18 separate predictor calls. At the 0.5-second sampling interval used on Assembly101, the measured SPACE update occupies approximately 2% of the interval. The State-only and Utility-only variants require 5.37 ms and 8.95 ms, respectively, indicating the additional cost of combining the two branches.

SPACE reaches a peak allocated GPU footprint of approximately 1.76 GB, including 828.82 MB of update-time allocation. The utility bank occupies approximately 1.47 GB of CPU memory and 740.24 MB of GPU memory in the profiled implementation. Absolute peak values include resident allocations from the common benchmarking process and therefore should not be interpreted as the standalone memory requirements of baselines that do not use the utility bank.

Batching reduces the practical cost by replacing $K + 1$ separate predictor calls with one forward pass, but it does not remove the underlying $\mathcal{O}(K)$ candidate computation or activation-memory

Table 10: Results on EPIC-KITCHENS-100 under the matched causal-stream protocol. H64 gain is paired against Reservoir. Best mean values are **bold** and second-best mean values are underlined.

Method	Action NLL ↓	mAP ↑	LRAP ↑	Recall@5 ↑	H64 gain ↑	Basin drift ($\times 10^{-3}$) ↓	Boundary selectivity ↑
FIFO	4.3424	0.2715	0.4478	0.6198	0.0125	8.5610	0.6106
Reservoir	4.3469	0.2749	0.4473	0.6247	0.0000	9.2249	0.5950
Surprise	4.3844	0.2753	0.4426	0.6172	−0.0879	10.6138	0.5888
Similarity	4.3832	0.2746	0.4432	0.6147	−0.0821	9.6320	0.5975
Attention	4.3448	0.2714	0.4467	**0.6261**	0.0162	7.4006	**0.6337**
Self-Influence	4.3519	0.2720	0.4458	0.6228	0.0037	7.5461	0.6257
StreamForest	4.3615	0.2733	0.4454	0.6202	−0.0337	9.0522	0.6016
OmniMem	4.3635	0.2742	0.4448	0.6187	−0.0366	9.8807	0.5871
ObjectStream	4.3673	0.2738	0.4431	0.6181	−0.0414	9.3291	0.5942
NovaCov	4.3619	**0.2756**	0.4463	0.6182	−0.0368	8.5487	0.5993
Utility-only	4.3415	0.2748	0.4454	0.6241	0.0182	8.8777	0.6041
State-only	4.3459	0.2743	0.4467	0.6248	0.0082	8.3987	0.6175
SPACE	**4.3245**	0.2755	**0.4530**	0.6254	**0.0433**	**7.3193**	0.6289

Table 11: Deployment-time cost on Assembly101 with $K = 16$. Wall time is the participant-balanced mean ± sample standard deviation per full-memory update. "Candidates" counts candidate-equivalent evaluations, whereas "calls" counts predictor forward passes after batching. ×Res. is computed relative to Reservoir within each matched participant and seed before aggregation; it therefore need not equal the ratio of the displayed mean wall times. ΔGPU denotes the peak memory allocated during the update above the loaded pre-update state.

Method	Candidates	Calls	Wall time (ms) ↓	×Res.	ΔGPU (MB) ↓	Peak GPU (MB) ↓
FIFO	0	0	1.686 ± 0.003	1.67	0.78	925.68
Reservoir	0	0	1.006 ± 0.006	1.00	0.65	925.55
Attention	2	2	9.866 ± 1.337	9.89	4.28	931.89
Self-Influence	18	18	61.698 ± 0.068	63.05	4.63	932.24
Utility-only	17	1	8.950 ± 0.042	9.15	828.82	1756.43
State-only	17	1	5.367 ± 0.188	5.47	12.63	940.24
SPACE	17	1	9.870 ± 0.038	10.08	828.82	1756.43

scaling. The measurements therefore establish that the overhead is practical at the evaluated capacity $K = 16$, rather than claiming cost independent of memory capacity.

F Retrospective Utility Validation

We retrospectively evaluate whether the train-only utility bank identifies beneficial candidate evictions. Future observations used to construct these retrospective oracle targets are accessed only after online decisions are fixed; they are not used to train the utility bank or alter the evaluated online decisions. Oracle advantage is measured relative to the same Reservoir action using the train-standardized combination of one-step and $R = 32$ rollout costs, with positive values indicating lower future cost.

Table 12 shows that the estimator exceeds the majority sign baseline by 10.69 percentage points. Its Hit@1 and Hit@3 rates are approximately $2.30\times$ and $1.60\times$ their corresponding uniform-ranking levels. The utility bank therefore provides informative candidate screening rather than a precise ranking over all 17 actions, consistent with its role as an admissibility filter rather than the final action selector.

SPACE overrides Reservoir on only 3.44% of full-memory updates. Among the 209 overrides with valid oracle targets, 58.85% reduce retrospective future cost. Although not every intervention improves utility in hindsight, the mean combined oracle advantage remains positive at $+0.4166$, indicating that beneficial corrections outweigh detrimental interventions on average. This result is consistent with the asymmetric role of utility in SPACE: utility screens supported alternatives, while basin geometry determines whether and when to intervene.

Table 12: Retrospective utility validation on Assembly101. Estimator metrics use 960 full-memory states with 17 candidate actions, and intervention metrics use 209 SPACE overrides with complete oracle targets. Hit@1 and Hit@3 are compared with uniform ranking. Combined gain is reported in train-standardized cost units.

Metric	Observed	Reference
Estimator ordering		
Pairwise sign accuracy ↑	61.98%	51.29% majority
Hit@1 ↑	13.53%	5.88% random
Hit@3 ↑	28.29%	17.65% random
Executed interventions		
Positive-gain overrides ↑	58.85%	50.00% parity
Mean combined oracle gain ↑	+0.4166	0 (Reservoir)

Table 13: Evaluation with an independently trained audit predictor on Assembly101. Values are participant-balanced means ± sample standard deviations across the ten participant-level means after averaging policy seeds. Gains are paired with Reservoir on the same recording. Best values are **bold**, and second-best values are underlined.

Method	Audit Action NLL ↓	H16 gain ($\times 10^{-4}$) ↑	H64 gain ($\times 10^{-4}$) ↑	Audit drift gain ($\times 10^{-6}$) ↑	Boundary selectivity ↑
Reservoir	$\underline{3.5763 \pm 0.1073}$	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	$\underline{0.8568 \pm 0.0864}$
Utility-only	3.5768 ± 0.1082	$\underline{2.5895 \pm 10.7770}$	5.0917 ± 10.9680	2.0749 ± 1.1209	0.8564 ± 0.0868
State-only	3.5777 ± 0.1081	-0.1472 ± 9.1888	$\underline{5.2427 \pm 11.9840}$	$\mathbf{22.2150 \pm 14.2020}$	0.8563 ± 0.0881
SPACE	$\mathbf{3.5761 \pm 0.1070}$	$\mathbf{4.6237 \pm 4.6086}$	$\mathbf{6.0321 \pm 4.5149}$	$\underline{3.0900 \pm 4.5634}$	$\mathbf{0.8569 \pm 0.0872}$

G Independent-Evaluator Analysis

The primary stability metrics are computed from predictions of the JEPA model that also supports the SPACE controller. To test whether the observed behavior is specific to this predictive geometry, we construct a separate audit evaluator with a different architecture and disjoint parameters. The audit predictor and its action decoder are trained only on the Assembly101 training split and remain frozen throughout evaluation. Neither component accesses the utility bank, slow predictive basis, basin prototype, controller thresholds, or SPACE decisions.

We replay the frozen policy trajectories through this evaluator without altering their eviction actions. The independent predictive state is formed directly from the multi-horizon outputs of the audit predictor, without applying P_{slow}, and is used to recompute within-regime drift and boundary selectivity. The accompanying action decoder provides an independent measurement of future-task utility.

As shown in Table 13, SPACE retains the lowest Audit Action NLL and the largest positive gains at both H16 and H64 under the independent evaluator. It also reduces within-regime drift while maintaining boundary selectivity under the independent predictive geometry. State-only produces substantially greater drift reduction but weaker future-task utility, consistent with the main finding that stronger state contraction alone does not improve predictive utility.

Although the differences under the audit evaluator are small, their consistent direction across future-task and state-dynamics metrics suggests that the behavior of SPACE is not specific to the JEPA geometry used by the controller. Because both evaluators operate on the same frozen TSM input representation, we interpret these results as robustness to an independently trained evaluator with a different architecture and disjoint parameters, rather than as complete representation independence.

H Stability Beyond the Slow Predictive Space

To assess whether the observed drift reduction depends on P_{slow}, we re-evaluate fixed policy trajectories using the original JEPA predictor across three evaluation spaces. In addition to the slow space, we consider the pre-slow space, which retains the JEPA input projection but omits P_{slow}, and the raw JEPA output space, which omits both projections. The predictor and eviction decisions remain unchanged; only the representation used to measure drift varies.

Table 14: SPACE drift gains beyond the slow predictive space. Gain is defined as $D_{\text{within}}^{\text{Reservoir}} - D_{\text{within}}^{\text{SPACE}}$; positive values indicate lower within-regime drift. Means and 95% participant-level confidence intervals are reported in units of 10^{-6}. Positive counts indicate participants with positive paired gains.

Evaluation space	Mean drift gain ↑	95% CI	Positive
Slow	0.4958	[0.2818, 0.7513]	5/5
Pre-slow	3.6922	[2.4701, 5.5429]	5/5
Raw JEPA output	7.8954	[5.5802, 11.5720]	5/5

Table 14 shows positive drift gains across all three spaces, with confidence intervals excluding zero and positive gains for all five evaluated participants. In particular, the pre-slow result directly isolates the effect of removing P_{slow}, while the raw-output result provides further evidence that the drift reduction is not confined to the slow predictive geometry used by the controller. Gain magnitudes are not directly comparable across representation spaces. Given the limited participant coverage, we interpret this audit as complementary evidence of stability beyond the slow projection rather than as evidence of representation-independent stability.