ORGANIZE, THEN READ: RETHINKING TOKEN MIXING VIA STRUCTURED STATE REPRESENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Self attention has become a central token-mixing operator in embodied policies, yet many alternatives primarily address the quadratic cost of pairwise interaction rather than how source information should be organized before retrieval. We introduce Structured State Representation (SSR), an operator based on the principle of organize first, then read. Within each head, learned state centers shared across samples serve as common addresses for source organization. Source tokens are competitively assigned and aggregated into input-dependent states, after which queries selectively retrieve from the resulting compact state bank. This separation yields linear token-mixing complexity for a fixed number of states without constructing dense query–source interaction matrices. Controlled operator-replacement experiments in ACT and Diffusion Policy show that SSR remains competitive with standard attention and improves performance in selected settings, including human demonstrations. Qualitative latent analyses further suggest more progress-consistent representation geometry while preserving high correspondence with Transformer representations. Together, these findings support structured state organization as a viable alternative to direct pairwise attention in embodied policy learning.

1 INTRODUCTION

Transformers have become a dominant backbone in modern machine learning, with self attention enabling global token interaction without the sequential computation of recurrent architectures (Vaswani et al., 2017). This flexibility suits embodied policies integrating observations, robot states, task conditions, and actions. Transformers now support language-conditioned manipulation, action prediction, and diffusion policies across PerAct, ACT, Diffusion Policy, RT-2, and OpenVLA (Shridhar et al., 2023; Zhao et al., 2023; Chi et al., 2023; Zitkovich et al., 2023; Kim et al., 2025).

However, widespread adoption does not establish direct pairwise interaction as the only useful information flow for embodied policies. Robot demonstrations are costly and training budgets are often limited. Standard attention lets each query directly weight source tokens without first aggregating them into a compact state bank shared across queries. Recent robotics studies explore embodiment-aware structure, selective state-space models, and linear attention (Sferrazza et al., 2025; Jia et al., 2025; Leal et al., 2024). Related architectures organize information through inducing points, latent arrays, or competitive slots (Lee et al., 2019; Jaegle et al., 2021; Locatello et al., 2020). We ask whether organizing sources before query-conditioned retrieval provides a useful inductive bias within existing embodied policy backbones.

We introduce Structured State Representation (SSR), a token-mixing operator built around an *organize first, then read* pathway. Within each head, learned state centers shared across samples serve as persistent organizational addresses. Source tokens are competitively routed toward these centers and aggregated into input-dependent state contents. Queries then retrieve from the resulting state bank: $(\mathbf{K}, \mathbf{V}) \xrightarrow[\mathbf{C}]{\text{organize}} \mathbf{S}, \quad (\mathbf{Q}, \mathbf{S}) \xrightarrow{\text{read}} \mathbf{Y}$. Here, $\mathbf{C}$ provides shared organizational addresses, while $\mathbf{S}$ stores input-dependent contents. The organization is modality agnostic and learned jointly with the policy objective. For a fixed state count, SSR factorizes query-to-source interaction through the compact state bank and scales linearly with source and query sequence lengths without constructing

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.

a dense query-to-source matrix. This is an asymptotic property rather than a guarantee of lower wall-clock latency on short sequences. Our primary interest is the inductive bias introduced by separating source organization from subsequent retrieval.

We evaluate SSR through controlled operator replacement in ACT and Transformer-based Diffusion Policy, retaining the surrounding architectures and training pipelines. Experiments span scripted and human demonstrations, multiple training seeds, and architectural variants, together with analyses of computational characteristics and representation geometry.

Our main contributions are as follows:

1. We introduce Structured State Representation (SSR), which is a new token-mixing operator that separates source organization from query-conditioned retrieval. Learned centers shared across samples provide persistent addresses for constructing input-dependent states, while retaining the standard $(Q, K, V) \to Y$ interface.

2. We show that SSR induces a dynamic state-mediated factorization of query-to-source interaction through M intermediate states. This formulation avoids constructing a dense interaction matrix and yields linear token-mixing complexity in the source and query sequence lengths for fixed M.

3. We validate SSR through controlled replacement in two distinct embodied policy formulations, ACT and Diffusion Policy. Across scripted and human demonstrations, SSR remains competitive with standard attention and improves performance in selected regimes, while complementary analyses characterize its representation geometry, computational behavior, and task-dependent trade-offs.

2 Related Work

Sequence modeling in embodied policies. Sequence models provide a unified interface for observations, conditions, and actions in embodied policies. Prior work has explored multimodal behavior cloning, prompt-conditioned manipulation, generalist robot policies, heterogeneous pre-training, and diffusion-based action generation (Shafiullah et al., 2022; Jiang et al., 2023; Brohan et al., 2023; Ghosh et al., 2024; Haldar et al., 2024; Wang et al., 2024; Liu et al., 2025; Pertsch et al., 2025). Our focus is narrower: we study the token-mixing operator within existing embodied policy backbones rather than introducing a new policy formulation.

Alternatives to pairwise attention. Dense token–token interaction has been reconsidered through kernelized linear attention, random-feature approximations, sparsity, data-dependent gating, and selective state-space models (Katharopoulos et al., 2020; Choromanski et al., 2021; Yang et al., 2024; Han et al., 2023; Fan et al., 2025; Yuan et al., 2025; Gu & Dao, 2024; Dao & Gu, 2024; Lahoti et al., 2026; Han et al., 2026). These approaches motivate alternatives to direct dense pairwise mixing. SSR instead introduces an explicit state bank that separates source organization from subsequent query-conditioned retrieval.

Structured intermediate representations. Compact intermediate representations have also been used through latent arrays, inducing points, learned queries, and slots (Ma et al., 2021; Jaegle et al., 2022; Carion et al., 2020; Jiang et al., 2024; Zhang et al., 2024; Kori et al., 2024). Among the closest examples, Set Transformer's ISAB uses learned inducing points and two attention blocks (Lee et al., 2019), while Slot Attention combines competitive assignment with iterative recurrent slot updates (Locatello et al., 2020). SSR differs by using learned centers shared across samples as persistent organizational addresses, constructing input-dependent state contents in a single distance-based competitive aggregation step, and accessing them through a separate query-conditioned readout while preserving the standard $(Q, K, V) \to Y$ interface.

3 Problem Formulation

3.1 Pairwise Attention and Input-Dependent Addresses

Consider one attention head h with query representations $\mathbf{Q}_h \in \mathbb{R}^{B \times N_q \times d}$ and source representations $\mathbf{K}_h, \mathbf{V}_h \in \mathbb{R}^{B \times N_s \times d}$, where B is the batch size, N_q and N_s denote the query and source sequence lengths, and d is the head dimension. Scaled dot-product attention computes (Vaswani

et al., 2017)

$$\mathbf{A}_h = \mathrm{softmax}\left(\frac{\mathbf{Q}_h\mathbf{K}_h^\top}{\sqrt{d}}\right), \quad \mathbf{Y}_h = \mathbf{A}_h\mathbf{V}_h, \tag{1}$$

where softmax is applied over the N_s source positions and $\mathbf{A}_h \in \mathbb{R}^{B\times N_q\times N_s}$ contains all query–source interaction weights. From a retrieval perspective, each source key acts as an input-dependent address, while its value carries the associated content. Every query compares against all source addresses and constructs its own distribution over the complete source sequence. This enables flexible global interaction, but source organization and query-conditioned retrieval remain coupled within the same pairwise operation: the source is not first transformed into a representation shared across queries.

This coupling is relevant to embodied policies, where heterogeneous token types share a common representation space. Pairwise attention can model their interactions flexibly, but provides no explicit shared structure through which source information is organized before retrieval. We therefore ask whether a shared, task-driven organization of the source can provide a useful structural bias before query-conditioned access.

3.2 Separating Organization from Retrieval

To separate these roles, consider an intermediate state bank formed before query-specific retrieval:

$$\mathbf{S}_h = \Phi_h(\mathbf{K}_h, \mathbf{V}_h), \qquad \mathbf{Y}_h = \Psi_h(\mathbf{Q}_h, \mathbf{S}_h). \tag{2}$$

Here, $\mathbf{S}_h \in \mathbb{R}^{B\times M\times d}$ contains M intermediate states, Φ_h organizes the source sequence, and Ψ_h performs query-conditioned retrieval. In the non-causal setting, the same state bank is shared by all queries associated with an input. For causal self-mixing, the formulation becomes query-position dependent $\mathbf{S}_{h,t} = \Phi_h(\mathbf{K}_{h,\mathcal{V}_t}, \mathbf{V}_{h,\mathcal{V}_t})$, where $\mathcal{V}_t$ denotes the source positions visible to query t. The corresponding query therefore retrieves only from information available within its visible prefix. Both cases preserve the same external $(Q, K, V) \to Y$ interface.

Equation 2 remains agnostic to the parameterization of Φ_h and Ψ_h. A concrete operator must construct input-dependent states and support selective retrieval without recovering dense query–source interaction. SSR instantiates these mappings through persistent learned addresses, competitive source organization, and compact query-conditioned state retrieval, as detailed in Sec. 4.

4 Structured State Representation

In this section, the principle of *organize first, then read* is instantiated as a concrete token mixing operator. A shared bank of learned state addresses coordinates source organization, while the resulting input-dependent states provide a compact interface for subsequent query-conditioned retrieval.

4.1 Overview: Organize, Then Read

Rather than organizing source information independently for each query, SSR equips each head h with M learned state centers,

$$\mathbf{C}_h = [\mathbf{c}_{h1}, \mathbf{c}_{h2}, \ldots, \mathbf{c}_{hM}] \in \mathbb{R}^{M\times d}. \tag{3}$$

Within each SSR layer, the centers in Eq. 3 are learned parameters shared across samples. They do not store the content of a particular input, nor do they represent recurrent memories carried across time. Instead, they serve as persistent organizational addresses around which source information is structured. No address is assigned in advance to a specific modality or semantic category. Its role is learned jointly with the policy objective. Given the source representations $\mathbf{K}_h$ and $\mathbf{V}_h$, SSR constructs a writing matrix $\mathbf{W}_h \in \mathbb{R}^{B\times M\times N_s}$. The relation between $\mathbf{K}_h$ and the center bank determines how the source values are organized into a state bank. Queries subsequently produce a readout matrix $\mathbf{R}_h \in \mathbb{R}^{B\times N_q\times M}$ over the compact state bank, enabling different queries to selectively retrieve from the organized states. For a shared state bank, one head computes

$$\mathbf{S}_h = \mathbf{W}_h\mathbf{V}_h, \quad \mathbf{Y}_h = \mathbf{R}_h\mathbf{S}_h = \mathbf{R}_h\mathbf{W}_h\mathbf{V}_h. \tag{4}$$

Unlike the center bank, $\mathbf{S}_h$ is reconstructed for every input and therefore depends on the current source sequence. The addresses determine where source information is organized, whereas the state

contents contain what has been organized from the current input. From Eq. 4, organization and retrieval become two separate operations mediated by the same compact state bank. Sec. 4.2 defines the construction of $\mathbf{W}_h$, while Sec. 4.3 defines $\mathbf{R}_h$.

4.2 Persistent State Addresses and Source Organization

A central question is how source tokens should be organized around the shared address bank without imposing hard partitions. For state m in head h, SSR associates the center $\mathbf{c}_{hm}$ with a learnable concentration parameter. Given a source key $\mathbf{k}_{bhn}$, the routing geometry is defined as

$$\alpha_{hm} = \alpha_{\min} + \mathrm{softplus}(\rho_{hm}), \quad \delta_{bhmn} = \|\mathbf{k}_{bhn} - \mathbf{c}_{hm}\|_2^2, \quad a_{bhmn} = -\frac{\alpha_{hm}\delta_{bhmn}}{\tau_a}. \tag{5}$$

where $\alpha_{\min} > 0$ prevents the concentration from vanishing and $\tau_a > 0$ is the assignment temperature. A higher concentration sharpens routing around a center, whereas a lower value gives a broader response. Distances are computed without explicit ℓ_2 normalization of the keys or centers, so α_{hm} scales the full key–center distance. To prevent every state from independently collecting the complete source sequence, SSR introduces competition across states. The assignment logits are first normalized over states, followed by normalization over valid source positions:

$$p_{bhmn} = \mathrm{softmax}_m(a_{bhmn}), \quad w_{bhmn} = \frac{g_{bn}p_{bhmn}}{\max\left(\sum_{j=1}^{N_s} g_{bj}p_{bhmj}, \epsilon\right)}, \quad \mathbf{s}_{bhm} = \sum_{n=1}^{N_s} w_{bhmn}\mathbf{v}_{bhn}. \tag{6}$$

where $g_{bn} \in \{0, 1\}$ marks valid source positions and $\epsilon > 0$ provides a denominator floor. The first normalization determines the ownership of the source across states, while the second determines the aggregated content of each state . No fixed semantic interpretation is imposed on an individual state. For causal self-mixing, Eq. 6 is evaluated over the query-specific visible prefix $\mathcal{V}_t$ (equivalently using a query-dependent validity mask); the non-causal case uses the same valid source set for all queries. Prefix states can be constructed with cumulative sums without recomputing previous source positions.

4.3 Query-Conditioned State Readout

Once the state bank has been constructed, each query retrieves from the M organized states rather than directly from the original source tokens. Let $\mathbf{u}_{bhm}$ denote the readout anchor associated with state m. We consider three parameter-free instantiations: $\mathbf{u}_{bhm} = \mathbf{c}_{hm}$, $\mathbf{u}_{bhm} = \mathbf{s}_{bhm}$, and $\mathbf{u}_{bhm} = (\mathbf{c}_{hm} + \mathbf{s}_{bhm})/2$, corresponding to center-based, state-based, and hybrid readout, respectively. In

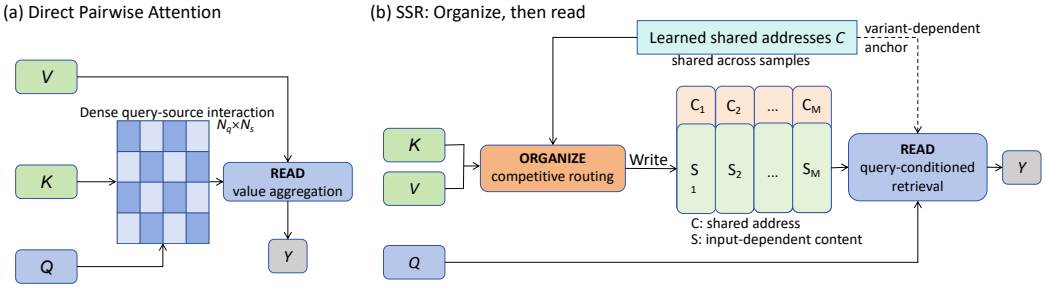

Figure 1: Conceptual comparison between direct pairwise attention and Structured State Representation (SSR). (a) Standard attention (Vaswani et al., 2017) forms a dense query–source interaction before aggregating source values. (b) SSR instead organizes source information into a compact bank of input-dependent states using learned addresses shared across samples, and then performs query-conditioned retrieval over the resulting state bank. The dashed connection indicates that the persistent centers may additionally serve as readout anchors depending on the readout instantiation. This separates source organization from subsequent retrieval while preserving the standard Q, K, V to Y operator interface.

all cases, the structured states provide the retrieved contents:

$$r_{bhtm} = \frac{\mathbf{q}_{bht}^{\top}\mathbf{u}_{bhm}}{\sqrt{d}\,\tau_r}, \quad \beta_{bhtm} = \mathrm{softmax}_m(r_{bhtm}), \quad \mathbf{y}_{bht} = \sum_{m=1}^{M} \beta_{bhtm}\mathbf{s}_{bhm}. \tag{7}$$

Here, $\tau_r > 0$ controls the readout temperature. The three variants share the same state construction and retrieved contents, differing only in their readout anchors. For prefix-causal mixing, state-dependent anchors and retrieved contents use the state bank constructed from the query's visible prefix. The hybrid readout averages address-based and content-based compatibility scores, since $\mathbf{q}^{\top}(\mathbf{c}+\mathbf{s})/2 = (\mathbf{q}^{\top}\mathbf{c} + \mathbf{q}^{\top}\mathbf{s})/2$. In the projected variant, the center and state content are associated with independently learned key and value representations. Their equal-weight fusion is a readout design choice rather than an assumption of coordinate alignment. Organization and readout remain asymmetric: distance-based routing establishes source ownership, whereas scaled inner-product similarity performs query-conditioned retrieval. In the shared-state-bank case, combining Eqs. 6 and 7 recovers the factorized head computation in Eq. 4.

4.4 Multihead SSR and Output Projection

In order to incorporate SSR into existing policy blocks, the operator preserves the input and output interface of multihead attention. Given H heads and $d = D/H$, the representations are divided into $\{\mathbf{Q}_h\}_{h=1}^{H}$, $\{\mathbf{K}_h\}_{h=1}^{H}$, and $\{\mathbf{V}_h\}_{h=1}^{H}$. Each head maintains an independent center bank and constructs its own writing and readout matrices. The head outputs are concatenated along the feature dimension and passed through an output projection, as defined in Eq. 8.

$$\mathbf{Y} = \mathrm{Dropout}\big(\mathrm{Concat}_{h=1}^{H}(\mathbf{Y}_h)\mathbf{W}_O + \mathbf{b}_O\big). \tag{8}$$

where $\mathbf{W}_O \in \mathbb{R}^{D\times D}$ and $\mathbf{b}_O \in \mathbb{R}^{D}$. At the operator level, the canonical SSR formulation introduces no additional learned input projections for $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$, while retaining the output projection in Eq. 8. Variants with learned input projections are treated as architectural ablations. Since the output remains in $\mathbb{R}^{B\times N_q\times D}$, SSR can replace either self attention or cross attention without changing the surrounding policy architecture.

4.5 Structural Properties

Address and content separation. The center bank $\mathbf{C}_h$ is shared across samples and provides a stable address space, whereas the state bank $\mathbf{S}_h$ is reconstructed from the current source sequence and contains input-dependent information. In this way, SSR separates where information is organized from what information is stored. The centers should therefore be interpreted as organizational addresses rather than recurrent memories or state contents.

Competitive organization. A remaining problem is how to prevent multiple states from learning nearly identical collection patterns. To introduce differentiation, the ownership of each source token is normalized across states before state-wise aggregation. Each token therefore distributes a fixed unit of ownership among the M available addresses. This competition may encourage distinct organization patterns, although it does not impose a fixed semantic meaning on any individual state.

Factorized token mixing. For the shared-state-bank formulation, the effective query–source mixing matrix for sample b and head h can be written as

$$\widetilde{\mathbf{A}}_{bh} = \mathbf{R}_{bh}\mathbf{W}_{bh} \in \mathbb{R}^{N_q\times N_s}, \quad \mathrm{rank}\Big(\widetilde{\mathbf{A}}_{bh}\Big) \le M. \tag{9}$$

Thus, SSR does not remove query–source interaction but factorizes it through M structured states. The state count M therefore acts as an explicit capacity parameter: increasing M relaxes the per-head rank bound at a proportional increase in state-mediated computation and memory. This should not be interpreted as a fixed low-rank mixing matrix, since both factors are input dependent: $\mathbf{W}_{bh}$ depends on the source keys, while $\mathbf{R}_{bh}$ depends on the queries and readout anchors. SSR is therefore a dynamic state-mediated factorization.

4.6 Computational Complexity

The construction of assignment scores and the state bank requires $\mathcal{O}(BHMN_sd)$ operations. Across the three readout instantiations considered here, computing query–anchor scores and aggregating the

state contents requires $\mathcal{O}(BHMN_q d)$ operations. Hence,

$$\mathcal{C}_{\mathrm{SSR}} = \mathcal{O}(BHMd(N_s + N_q)), \quad \mathcal{C}_{\mathrm{attn}} = \mathcal{O}(BHN_q N_s d). \tag{10}$$

For fixed M, SSR scales linearly with source and query lengths, and prefix-causal self-mixing preserves this time complexity through cumulative sums. The assignment and readout weights require $\mathcal{O}(BHM(N_s + N_q))$ elements, while materializing all prefix states adds $\mathcal{O}(BHMN_s d)$ elements. In contrast, a materialized dense attention matrix requires $\mathcal{O}(BHN_q N_s)$ elements. These estimates cover token mixing but exclude input and output projections. However, lower asymptotic cost does not guarantee lower wall-clock latency on short sequences, where optimized attention kernels may remain faster. End-to-end FLOPs and module-level memory are evaluated separately in Sec. 5.

5 Experiments

5.1 Experimental Setup

Policy architectures and replacement protocol. We evaluate SSR in two distinct policy formulations: Action Chunking with Transformers (ACT) (Zhao et al., 2023) and the time-series Diffusion Transformer variant of Diffusion Policy (Chi et al., 2023). The ACT experiments use the LeRobot implementation (Cadene et al., 2026), whereas the Diffusion Policy experiments follow its official implementation. For matched operator comparisons, Mamba (Gu & Dao, 2024) and SSR replace the same self-attention modules. In Diffusion Policy, we additionally evaluate a full SSR configuration that replaces all self- and cross-attention modules and compare it with the original Transformer backbone. All remaining network components and training protocols are kept unchanged within each experimental setting.

Tasks and evaluation protocol. We compare Transformer, Mamba, and SSR on the simulated ALOHA Transfer Cube and Bimanual Insertion tasks. ACT is evaluated with both scripted and human demonstrations, whereas Diffusion Policy is evaluated on scripted Transfer Cube demonstrations. Unless otherwise stated, the default SSR configuration uses four states per head, RBF assignment, competitive state organization, hybrid readout, and no learned input QKV projections. ACT and Diffusion Policy are trained with ten and three paired random seeds, respectively. Within each seed, the compared operators use matched data splits and rollout initializations. Each selected policy is evaluated over 50 environment rollouts. Implementation and evaluation details are provided in Appendix C.

5.2 Policy Performance and Ablations

Training dynamics and convergence. Figure 2 compares the Transformer baseline, Mamba with self-attention replacement, SSR-self, and SSR-all on the scripted Transfer Cube task under Diffusion Policy. Including both SSR configurations allows us to examine the effect of replacement scope alongside comparisons with the baseline operators. Mamba improves rapidly during the earliest training stage, while SSR-all reaches a higher-success regime within approximately 200 epochs. Over this early training window, SSR-self and Mamba follow broadly similar trajectories, whereas the Transformer improves more gradually.

During extended training, SSR-all maintains success rates close to saturation over most evaluated checkpoints. SSR-self also improves and approaches SSR-all at later checkpoints, while Transformer and Mamba generally remain at lower mean success rates. All methods use matched data splits and rollout initializations within each paired seed. In this setting, full SSR replacement reaches high success earlier than self-attention replacement alone, although the two SSR configurations attain similar success rates late in training. These results support a convergence advantage for SSR-all on scripted Transfer Cube, while showing that SSR-self can approach its performance with extended training.

Interpretation of task-dependent performance. The performance differences in Tables 1 and 2 motivate examining how state organization balances shared aggregation with selective access to task information. By requiring queries to retrieve from a common state bank, SSR introduces a structural constraint that may reduce sensitivity to irrelevant source variation and provide a useful inductive bias for policy learning. However, aggregation may also attenuate subtle distinctions when action-critical cues are mixed with information that is less relevant to a particular query. This possibility

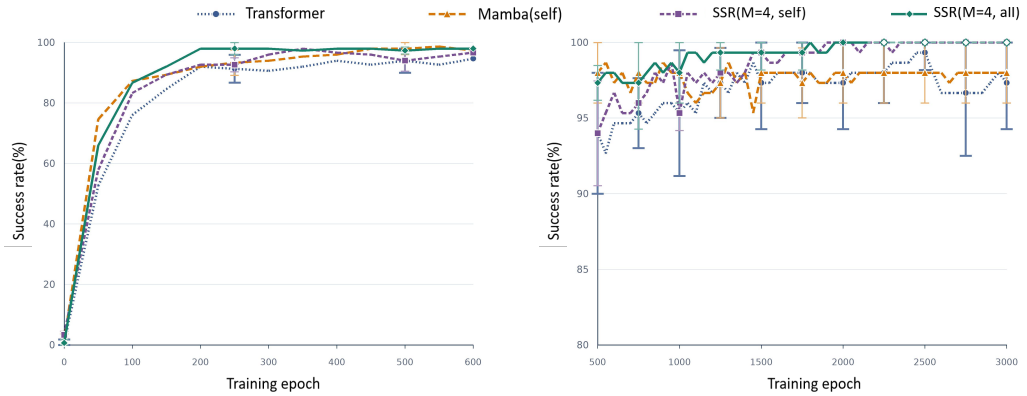

Figure 2: Training dynamics on ALOHA scripted Transfer Cube under Diffusion Policy. Self denotes self-attention replacement, while all denotes replacement of all self and cross attention modules. Both SSR variants use $M = 4$ states per head. The left panel shows epochs 0–600; the right panel shows epochs 500–3000 with the success-rate axis restricted to 80–100%. Curves show mean rollout success rates across three training seeds, and error bars denote one sample standard deviation across seeds.

Table 1: ACT success rates (%, mean ± SD). All SSR variants use $M = 4$ and encoder-only replacement.

Method	Readout	Transfer Cube		Bimanual Insertion		Avg.
		Scripted	Human	Scripted	Human	
Transformer	–	83.8 ± 10.4	52.6 ± 14.8	49.6 ± 9.9	15.8 ± 9.0	50.5
Transformer-S	–	78.2 ± 17.6	39.8 ± 14.7	55.0 ± 12.0	11.2 ± 5.3	46.1
Mamba-v1	–	80.8 ± 16.1	60.8 ± 15.9	50.2 ± 7.8	14.8 ± 5.3	51.7
SSR-P	Hybrid	79.6 ± 22.8	60.8 ± 10.8	47.6 ± 10.2	14.0 ± 8.1	50.5
SSR-P	Center	87.8 ± 7.7	64.8 ± 9.7	46.6 ± 10.6	11.8 ± 4.5	**52.8**
SSR-PF	Hybrid	77.2 ± 17.4	69.4 ± 10.1	42.8 ± 10.3	10.0 ± 6.1	49.9
SSR-PF	Center	85.0 ± 9.3	70.2 ± 6.1	39.2 ± 6.3	10.0 ± 6.4	51.1

P/PF denote projected/projection-free SSR. Avg. reports the unweighted mean across the four settings without SD. Transformer variants and SSR use ten training seeds. Each seed uses a validation-selected checkpoint and 50 continuous-RNG rollouts. Transformer-S changes only encoder self-attention, reducing its internal width/head count from $512/8$ to $64/1$ while retaining the external width of 512. It was selected for its measured module-level peak additional inference memory of 12.92 MiB, between the fused hybrid SSR-PF (9.41 MiB) and SSR-P (14.75 MiB) variants with $M = 4$.

Table 2: Diffusion Policy success rates (%, mean ± SD) on scripted Transfer Cube. All methods use three training seeds and train to epoch 3000. SSR uses projected inputs, hybrid readout, and $M = 4$.

Method	Replacement	Last-10 (%)	Peak (%)
Transformer	–	97.1 ± 3.6	100.0 ± 0.0
Mamba-v1	Self only	97.9 ± 2.0	98.7 ± 1.2
SSR-P	Self only	100.0 ± 0.0	100.0 ± 0.0
SSR-P	Self + cross	100.0 ± 0.0	100.0 ± 0.0

Last-10 averages epochs 2550–3000 within each seed; Peak is the highest checkpoint success rate. Each checkpoint is evaluated with 50 rollouts at 50-epoch intervals.

is especially relevant to precise manipulation, where small differences in relative object pose can require different action corrections. In Table 1, center readout improves mean success over hybrid readout on Transfer Cube under both input parameterizations, while the direction reverses or the means are equal on Insertion. This pattern motivates studying aggregation and retrieval jointly. Success rates alone do not identify the underlying physical failures, so we treat insufficient preservation or selective retrieval of fine geometric cues as a working hypothesis. The current results do not isolate this explanation from optimization or control effects, nor do they establish an intrinsic precision limit of structured state representations.

Effect of state capacity. We vary the number of states per head on ACT Bimanual Insertion with hybrid readout and encoder-only replacement (Table 3). Training and evaluation follow the main ACT

Table 3: State-capacity ablation on ACT Bimanual Insertion (%, mean ± sample SD across ten training seeds). All variants use hybrid readout and encoder-only replacement.

Data	Input	$M = 1$	$M = 4$	$M = 16$	$M = 64$
Scripted	Projected	45.2 ± 7.2	$\mathbf{47.6 \pm 10.2}$	46.0 ± 7.7	42.0 ± 6.0
Scripted	Projection-free	40.8 ± 5.9	42.8 ± 10.3	$\mathbf{48.0 \pm 8.5}$	42.4 ± 14.2
Human	Projected	9.0 ± 7.0	14.0 ± 8.1	13.8 ± 6.9	$\mathbf{15.2 \pm 4.3}$
Human	Projection-free	9.2 ± 4.2	10.0 ± 6.1	$\mathbf{13.0 \pm 6.1}$	9.6 ± 6.1

protocol, using training seeds 1–10, validation-selected checkpoints, and 50 rollouts per trained policy. Compared with $M = 1$, $M = 16$ achieves higher mean success in all four settings, with the largest gain for projection-free SSR under scripted demonstrations, from 40.8% to 48.0%. However, increasing M to 64 reduces mean success in three of the four settings relative to $M = 16$. These results support the effectiveness of the multi-center design while revealing a non-monotonic relationship between state capacity and policy performance.

5.3 Representation Geometry and State Organization

To examine whether SSR changes how policy information is organized, we compare the denoiser representations learned by the Transformer baseline and the full SSR replacement, denoted as SSR-all, in Diffusion Policy. In the t-SNE visualizations in Fig. 3(a), the representations of both models exhibit organization correlated with normalized episode progress. However, the SSR representations form a qualitatively more continuous progression, whereas the Transformer representations occupy more fragmented regions. A related pattern is visible in the episode-wise PCA trajectories in Fig. 3(b): the SSR rollouts follow a more coherent family of low-dimensional paths, while the Transformer rollouts are more dispersed and exhibit more apparent crossings in the two-dimensional projection. Because the dimensionality-reduction mappings are fitted independently, we interpret only the organization within each model's latent space rather than their absolute coordinates across models.

Despite these differences in the low-dimensional projections, linear CKA remains high across denoiser layers and diffusion timesteps (0.90–0.99; Fig. 3(c)). This high correspondence argues against the interpretation that SSR learns an unrelated representation space. Instead, SSR retains a sample-wise representational structure closely aligned with that of the Transformer, while exhibiting a qualitatively more progress-consistent organization in the visualized latent geometry. Taken together,

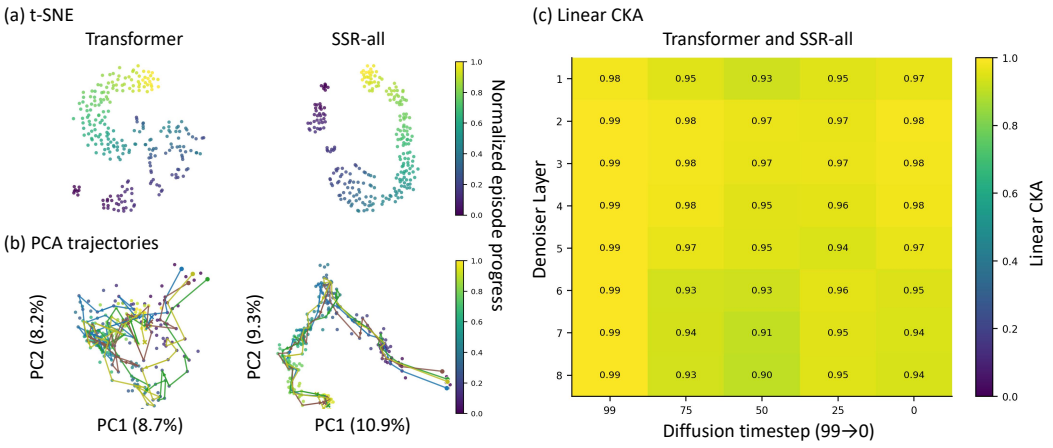

Figure 3: Representation geometry and cross-model correspondence between the Transformer baseline and SSR-all in Diffusion Policy on the scripted Transfer Cube task, using training seed 1. (a) t-SNE embeddings of denoiser representations, colored by normalized episode progress. (b) Episode-wise PCA trajectories, where connected points denote temporally consecutive representations within the same rollout. (c) Linear CKA between Transformer and SSR-all representations across denoiser layers and diffusion timesteps. The t-SNE and PCA mappings are fitted independently for the two models; therefore, their absolute coordinates and orientations are not directly comparable.

Table 4: End-to-end model size and inference FLOPs. All SSR configurations reported here use hybrid readout with $M = 4$ states per head.

Framework	Operator	Replacement	Params (M)	GFLOPs
ACT	Transformer	–	51.614	34.965
ACT	SSR, projected	Encoder only	51.622	34.242
ACT	SSR, projection-free	Encoder only	48.470	32.342
Diffusion Policy	Transformer	–	20.182	17.167
Diffusion Policy	SSR, projected	Self + cross	20.199	17.288
Diffusion Policy	SSR, projected	Self only	20.190	17.233
Diffusion Policy	SSR, projection-free	Self + cross	17.041	12.464
Diffusion Policy	SSR, projection-free	Self only	18.611	14.087

Table 5: Module-level peak additional GPU memory for one ACT attention site. SSR uses fused implementations with hybrid readout and $M = 4$ states per head. Values in parentheses indicate memory reduction relative to the Transformer baseline.

Phase	Transformer	SSR Projected	SSR Projection-free
Inference	63.05 MiB	14.75 MiB (↓76.6%)	9.41 MiB (↓85.1%)
Forward + backward	94.88 MiB	49.53 MiB (↓47.8%)	28.54 MiB (↓69.9%)

these results suggest that structured states reorganize, rather than replace, task-relevant information, consistent with their intended role as an explicit intermediate organization mechanism.

5.4 Computational Characteristics

To characterize the computational profile of SSR, we first compare end-to-end parameter counts and inference FLOPs under matched inputs within each policy framework. As summarized in Table 4, projected SSR remains close to the corresponding Transformer baseline in both model size and computation. Once the input QKV projections are removed, however, the reductions become substantially clearer. This effect is most pronounced in Diffusion Policy, where projection-free SSR replaces both self attention and cross attention.

Beyond end-to-end complexity, the temporary memory required inside the token mixing operator is also important. We therefore measure the peak additional allocation of one ACT attention site under a fixed tensor configuration. As reported in Table 5, fused projected SSR requires substantially less temporary memory during both inference and gradient computation. In contrast to the end-to-end measurements above, this benchmark isolates a single attention module and should not be interpreted as whole-policy peak VRAM.

Taken together, the two analyses reflect complementary computational effects. Removing the input projections reduces model size and FLOPs, while the structured state formulation lowers the temporary memory required for token interaction. Nevertheless, these improvements do not yet imply a wall-clock speedup. For the relatively short sequences considered in this study, highly optimized Transformer kernels may still execute faster than the current SSR implementation.

6 Conclusion

In this work, we introduced Structured State Representation (SSR), a token mixing operator that organizes source information around learned centers before query-conditioned retrieval. Separating persistent addresses from input-dependent contents yields linear token mixing complexity for a fixed state count while preserving the interface of existing attention modules. Controlled replacements in ACT and Diffusion Policy support the viability of this design, with success-rate improvements in some evaluated settings. The fused implementation also substantially reduces peak additional memory during inference in the module-level benchmark. Performance decreases in other settings also reveal the need to understand when shared aggregation is beneficial and which task-relevant distinctions must remain accessible after organization. Preserving and selectively retrieving subtle cues needed for precise action correction is therefore a promising direction for further investigation. SSR provides a concrete alternative operator and an empirical basis for studying how explicit source organization can support embodied policy learning.

AI USE STATEMENT

Large language models were used for minor language editing to improve the clarity and readability of parts of the manuscript. All edits were reviewed by the authors.

REFERENCES

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S Ryoo, Grecia Salazar, Pannag R Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan H Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.025.

Remi Cadene, Simon Alibert, Francesco Capuano, Michel Aractingi, Adil Zouitine, Pepijn Kooijmans, Jade Choghari, Martino Russi, Caroline Pascal, Steven Palma, Dana Aubakirova, Mustafa Shukor, Jess Moss, Alexander Soare, Quentin Lhoest, Quentin Gallouédec, and Thomas Wolf. LeRobot: An Open-Source Library for End-to-End Robot Learning. In *International Conference on Learning Representations*, 2026. URL https://mlanthology.org/iclr/2026/cadene2026iclr-lerobot/.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (eds.), *Computer Vision – ECCV 2020*, pp. 213–229, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58452-8.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.026.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ua6zuk0WRH.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10041–10071. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/dao24a.html.

Qihang Fan, Huaibo Huang, and Ran He. Breaking the low-rank dilemma of linear attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25271–25280, June 2025.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.090.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2024. URL https://arxiv.org/abs/2312.00752.

Siddhant Haldar, Zhuoran Peng, and Lerrel Pinto. Baku: An efficient transformer for multi-task policy learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 141208–141239. Curran Associates, Inc., 2024. doi: 10.52202/079017-4484. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/ff887781480973bd3cb6026feb378d1e-Paper-Conference.pdf.

Dongchen Han, Xuran Pan, Yizeng Han, Shiji Song, and Gao Huang. Flatten transformer: Vision transformer using focused linear attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5961–5971, October 2023.

Dongchen Han, Yining Li, Tianyu Li, Zixuan Cao, Ziming Wang, Jun Song, Yu Cheng, Bo Zheng, and Gao Huang. Vit$ˆ3$: Unlocking test-time training in vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 51–61, June 2026.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4651–4664. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/jaegle21a.html.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J Henaff, Matthew Botvinick, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver IO: A general architecture for structured inputs & outputs. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=fILj7WpI-g.

Xiaogang Jia, Qian Wang, Atalay Donat, Bowen Xing, Ge Li, Hongyi Zhou, Onur Celik, Denis Blessing, Rudolf Lioutikov, and Gerhard Neumann. Mail: Improving imitation learning with selective state space models. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3888–3907. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/jia25b.html.

Jindong Jiang, Fei Deng, Gautam Singh, Minseung Lee, and Sungjin Ahn. Slot state space models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 11602–11633. Curran Associates, Inc., 2024. doi: 10.52202/079017-0370. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/158ac5698e36a01ee5ca9e6732685b34-Paper-Conference.pdf.

Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. VIMA: Robot manipulation with multimodal prompts. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 14975–15022. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/jiang23b.html.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Avinash Kori, Francesco Locatello, Ainkaran Santhirasekaram, Francesca Toni, Ben Glocker, and Fabio De Sousa Ribeiro. Identifiable object-centric representation learning via probabilistic slot attention. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 93300–93335. Curran Associates, Inc., 2024. doi: 10.52202/079017-2960. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/a9a3f0e4a95cb273867931369c8fc3b1-Paper-Conference.pdf.

Aakash Lahoti, Kevin Y. Li, Berlin Chen, Caitlin Wang, Aviv Bick, J. Zico Kolter, Tri Dao, and Albert Gu. Mamba-3: Improved sequence modeling using state space principles, 2026. URL https://arxiv.org/abs/2603.15569.

Isabel Leal, Krzysztof Choromanski, Deepali Jain, Avinava Dubey, Jake Varley, Michael Ryoo, Yao Lu, Frederick Liu, Vikas Sindhwani, Quan Vuong, Tamas Sarlos, Ken Oslund, Karol Hausman, and Kanishka Rao. Sara-rt: Scaling up robotics transformers with self-adaptive robust attention. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6920–6927, 2024. doi: 10.1109/ICRA57147.2024.10611597.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 29982–30009, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/49f80e4d2471ad4f2edf4f5f1ab62339-Paper-Conference.pdf.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 11525–11538. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/8511df98c02ab60aea1b2356c013bc0f-Paper.pdf.

Xuezhe Ma, Xiang Kong, Sinong Wang, Chunting Zhou, Jonathan May, Hao Ma, and Luke Zettlemoyer. Luna: Linear unified nested attention. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 2441–2453. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/14319d9cfc6123106878dc20b94fbaf3-Paper.pdf.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models, 2025. URL https://arxiv.org/abs/2501.09747.

Carmelo Sferrazza, Dun-Ming Huang, Fangchen Liu, Jongmin Lee, and Pieter Abbeel. Body transformer: Leveraging robot embodiment for policy learning. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3407–3424. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/sferrazza25a.html.

Nur Muhammad Shafiullah, Zichen Cui, Ariuntuya (Arty) Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 22955–22968. Curran Associates, Inc., 2022. doi: 10.52202/068431-1668. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/90d17e882adbdda42349db6f50123817-Paper-Conference.pdf.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In Karen Liu, Dana Kulic, and Jeff Ichnowski (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 785–799. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/shridhar23a.html.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena : A benchmark for efficient transformers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qVyeW-grC2k.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Lirui Wang, Xinlei Chen, Jialiang Zhao, and Kaiming He. Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 124420–124450. Curran Associates, Inc., 2024. doi: 10.52202/079017-3952. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/e0f393e7980a24fd12fa6f15adfa25fb-Paper-Conference.pdf.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 56501–56523. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/yang24ab.html.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Yuxing Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, and Wangding Zeng. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23078–23097, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1126. URL https://aclanthology.org/2025.acl-long.1126/.

Yu Zhang, Songlin Yang, Ruijie Zhu, Yue Zhang, Leyang Cui, Yiqiao Wang, Bolun Wang, Freda Shi, Bailin Wang, Wei Bi, Peng Zhou, and Guohong Fu. Gated slot attention for efficient linear-time sequence modeling. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 116870–116898. Curran Associates, Inc., 2024. doi: 10.52202/079017-3710. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/d3f39e51f5f634fb16cc3e658f8512b9-Paper-Conference.pdf.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.016.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and

Kehang Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

APPENDIX

A ADDITIONAL ANALYSES AND BENCHMARKS

Long Range Arena evaluation. Beyond policy learning, we examine whether the proposed organization mechanism remains useful for sequence modeling with different input structures. To this end, Table 6 reports results on five tasks from Long Range Arena (Tay et al., 2021). ListOps evaluates nested symbolic computation, while Text classifies sentiment from byte-level movie reviews. Retrieval assesses document matching through separately encoded representations. Image classifies images presented as pixel sequences, whereas Pathfinder determines whether two marked locations are connected by a path. Together, these tasks probe different demands on hierarchical reasoning, sequence aggregation, and spatial relationships.

Across the reported configurations, SSR obtains higher Retrieval scores than the Transformer, and projected hybrid and state readouts also improve Pathfinder scores. However, Image accuracy remains lower, while differences on ListOps and Text are comparatively small. The variation across tasks and readouts suggests that the usefulness of structured aggregation depends on the information that must remain accessible for prediction. These single-seed results provide preliminary evidence of applicability beyond embodied policies, without establishing a consistent advantage across sequence-modeling tasks.

Table 6: Performance on the Long Range Arena benchmark. All results are percentages from a single training seed (seed 1), evaluated on the official test split using the checkpoint selected by validation performance. P and PF denote projected and projection-free SSR, respectively. M denotes the number of states per head. Best results in each column are shown in bold.

Method	ListOps	Text	Retrieval	Image	Pathfinder	Avg.
Transformer	39.20	**65.61**	83.60	**45.68**	73.83	61.58
SSR-P hybrid ($M = 4$)	38.70	65.58	84.18	44.53	**75.52**	61.70
SSR-P center ($M = 4$)	39.35	65.31	84.33	43.92	73.92	61.37
SSR-P state ($M = 4$)	**39.75**	65.12	84.01	44.79	75.42	**61.82**
SSR-PF hybrid ($M = 4$)	38.80	65.52	84.07	43.99	70.96	60.67
SSR-PF center ($M = 4$)	38.60	65.28	84.53	43.70	73.50	61.12
SSR-PF state ($M = 4$)	38.70	65.43	**84.63**	44.55	71.44	60.95
SSR-P center ($M = 16$)	39.40	65.48	84.14	43.74	74.52	61.46

Additional Diffusion Policy results. To broaden the manipulation evaluation beyond ALOHA, we further compare the operators on Square and Transport using proficient-human (PH) demonstrations. These experiments retain projected SSR with hybrid readout and $M = 4$, while considering both self-only and self-plus-cross replacement (Table 7). On Square, the reported success rates remain close to those of the Transformer and Mamba baselines. On Transport, both SSR configurations exceed the Transformer at the shared evaluation epoch, although SSR does not consistently outperform Mamba across the reported metrics. Moreover, full replacement improves the Last-5 average over self-only replacement on Square, while the ordering reverses on Transport. Taken together, these single-seed observations suggest that the same hybrid configuration can remain competitive on additional manipulation tasks, with the benefit of a broader replacement scope depending on the task.

Additional ACT results on Robomimic. To examine whether the ALOHA findings extend to other manipulation settings, we evaluate ACT on Square and Transport (Table 8). We use projected SSR with $M = 4$ and encoder-only replacement, comparing all three readout variants across ten training seeds. Under the evaluated configuration and training budget, all three variants achieve lower Last-5 mean success rates than the Transformer on both tasks, although state readout matches the final-checkpoint mean on Transport. In the additional Diffusion Policy experiments (Table 7), SSR's Last-5 averages are close to the corresponding Transformer baseline on Square and slightly higher on Transport. These results suggest that SSR's relative performance may depend on the surrounding policy architecture and training setup. However, differences in experimental protocols and the single-seed Diffusion Policy evaluations limit this comparison: the observed contrast cannot be attributed to the policy architecture alone.

Table 7: Additional Diffusion Policy results at matched logged epochs. All values are success rates (%) from training seed 1. SSR-P uses projected inputs, hybrid readout, and $M = 4$.

Method	Replacement	At E	Avg(Last-5)	Peak(last-5)
Square (PH)				
Transformer	–	80.0	80.0	86.0
Mamba-v1	Self only	86.0	81.2	86.0
SSR-P	Self only	80.0	78.4	88.0
SSR-P	Self + cross	84.0	80.8	86.0
Transport (PH)				
Transformer	–	86.0	85.2	88.0
Mamba-v1	Self only	90.0	89.6	92.0
SSR-P	Self only	90.0	87.6	92.0
SSR-P	Self + cross	92.0	86.0	92.0

At E reports the checkpoint at the shared epoch. Last-5 averages the five evaluations from epochs 2800–3000. Peak is the highest checkpoint success within epochs 2800–3000. Each checkpoint uses 50 rollouts. Only one training seed is available; across-seed SD is therefore not reported.

Table 8: Additional ACT results on Robomimic Square and Transport (%, mean $\pm$ sample SD across ten training seeds). SSR uses projected inputs with $M = 4$. Both SSR and Mamba use encoder-only replacement.

	Square (PH)			Transport (PH)		
Method	At E	Last-5 Avg.	Last-5 Peak	At E	Last-5 Avg.	Last-5 Peak
Transformer	61.00 ± 7.62	61.44 ± 6.92	69.60 ± 9.13	37.80 ± 13.74	39.60 ± 6.26	54.80 ± 10.08
Mamba-v1	63.20 ± 6.48	64.32 ± 5.28	71.80 ± 5.37	41.60 ± 14.32	34.16 ± 7.45	47.40 ± 11.89
SSR-P Hybrid	53.20 ± 9.76	53.16 ± 4.07	61.00 ± 6.88	29.40 ± 12.83	29.04 ± 8.35	42.20 ± 12.80
SSR-P Center	56.80 ± 9.94	52.16 ± 7.65	61.80 ± 5.85	34.60 ± 16.06	31.52 ± 10.96	46.00 ± 12.82
SSR-P State	53.80 ± 9.59	53.00 ± 4.03	61.60 ± 4.60	37.80 ± 17.01	33.64 ± 12.73	47.20 ± 14.21

At E reports the final checkpoint at $E = 50{,}000$ training steps. Last-5 uses checkpoints at 40k, 42.5k, 45k, 47.5k, and 50k steps, each evaluated with 50 rollouts. Avg. and Peak are computed within each training seed as the mean and maximum success rate over these five checkpoints, respectively, then summarized across seeds 1–10. Peak is restricted to this window rather than the full training run.

Readout ablation under projected inputs. While the preceding comparisons vary the task or replacement scope, the choice of retrieval anchor may also influence how organized information is used. We therefore compare the three readout variants under projected inputs with $M = 4$ and encoder-only replacement (Table 9). Center and state readouts attain the same highest mean success on scripted Transfer Cube, while state readout yields the highest mean under human demonstrations. By contrast, hybrid readout attains the highest means on both Insertion settings. This change in ordering suggests that persistent addresses and input-dependent contents may provide different benefits when used as retrieval anchors. Accordingly, the results favor treating readout as a configurable part of SSR, rather than assuming that one choice is uniformly preferable.

Table 9: Readout ablation for projected SSR on ACT (%, mean $\pm$ sample SD across ten training seeds). All variants use $M = 4$ and encoder-only replacement.

Readout	Transfer Cube		Bimanual Insertion	
	Scripted	Human	Scripted	Human
Hybrid	79.6 ± 22.8	60.8 ± 10.8	$\mathbf{47.6 \pm 10.2}$	$\mathbf{14.0 \pm 8.1}$
Center	$\mathbf{87.8 \pm 7.7}$	64.8 ± 9.7	46.6 ± 10.6	11.8 ± 4.5
State	$\mathbf{87.8 \pm 9.4}$	$\mathbf{68.8 \pm 9.6}$	45.4 ± 9.8	12.0 ± 6.1

Each seed uses a validation-selected checkpoint and 50 continuous-RNG rollouts. Hybrid and center results are reproduced from the main ACT comparison. Bold indicates the highest mean in each column.

B Efficiency Scaling with Sequence Length

Motivation and comparison. The policy experiments evaluate SSR at the input lengths used by the manipulation backbones, but do not fully characterize its computational behavior as the token workload grows. We therefore complement these experiments with a synthetic sequence-length benchmark, measuring inference latency and peak allocated GPU memory. The comparison includes

Transformer with SDPA, Mamba-v1 with a CUDA scan, and both projected and projection-free SSR with $M \in \{4, 16\}$. This allows us to examine whether the efficiency benefits become more apparent for longer inputs and whether they persist with a larger state bank.

Sequence-length scaling. As shown in Fig. 4, the Transformer has lower latency at short sequence lengths, indicating that the asymptotic advantage of SSR does not immediately translate into faster execution. As N increases, however, Transformer latency grows more rapidly, while the SSR curves rise more gradually and eventually fall below both baselines. At the longest tested sequence, all four SSR configurations have lower measured latency and peak allocated memory than the Transformer and Mamba-v1 implementations. Projection-free SSR also maintains a smaller memory footprint than its projected counterpart. These observations suggest that longer token sequences provide a regime in which the computational benefits of structured state aggregation become more apparent.

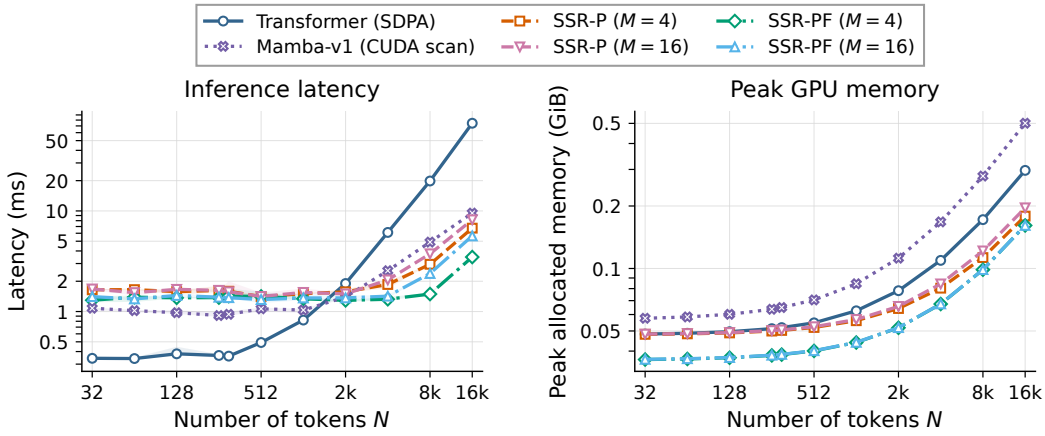

Figure 4: Synthetic sequence-length scaling of Transformer with SDPA, Mamba-v1 with a CUDA scan, and projected (P) and projection-free (PF) SSR with $M = 4$ and $M = 16$ states per head. The left panel reports inference latency in milliseconds, while the right panel reports peak allocated GPU memory in GiB. Both panels use logarithmic axes to show behavior across short and long sequences. Measurements concern the benchmarked token-mixing implementations rather than complete policy execution.

Efficiency with a larger state bank. To examine whether the observed advantages depend on a particularly small state count, Fig. 5 presents the $M = 16$ configurations on linear axes. Although these variants use four times as many states per head as the default $M = 4$ configuration, both retain lower latency and peak allocated memory than the two baselines at the longest tested sequence. The linear-axis view makes the absolute differences at large N easier to compare, particularly the latency gap relative to Transformer and the memory gap relative to Mamba-v1. Thus, the measured long-sequence benefits are not confined to the smallest state bank considered here.

Taken together, these measurements suggest that SSR may offer practical benefits in networks processing long token sequences, including when additional state capacity is required. The extent of any end-to-end improvement nevertheless depends on how much of the surrounding network's runtime and memory is attributable to token mixing.

C Implementation Details

ACT datasets and evaluation. The ACT experiments use four simulated ALOHA datasets released through LeRobot (Cadene et al., 2026): Transfer Cube and Bimanual Insertion, each with scripted expert and human demonstrations collected in simulation. Each dataset contains 50 episodes and 20,000 frames, except human Insertion, which contains 25,000 frames. Observations comprise top-view RGB images and 14-dimensional joint states, while the actions are 14-dimensional and recorded at 50 Hz. Following the ACT protocol (Zhao et al., 2023), 40 episodes are used for training and 10 for validation. The checkpoint with the lowest validation loss is selected and evaluated over 50 fresh environment rollouts.

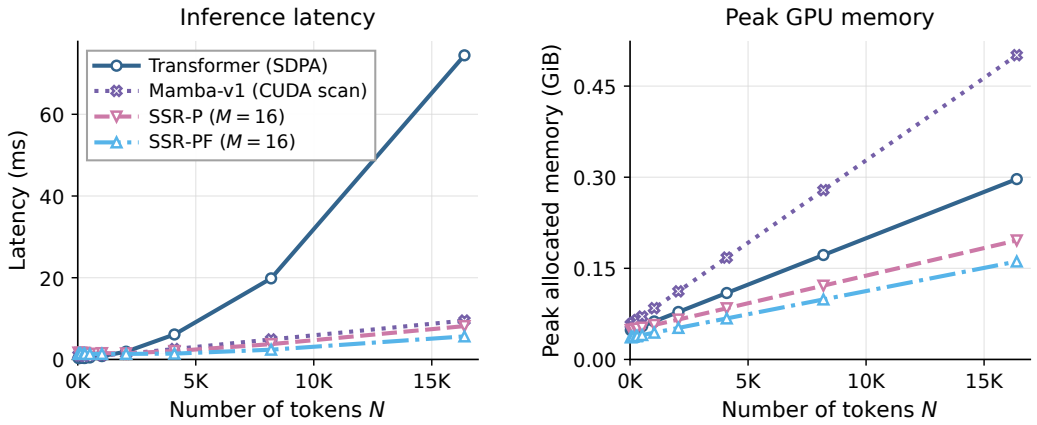

Figure 5: Efficiency comparison with $M = 16$ states per head. The Transformer, Mamba-v1, SSR-P, and SSR-PF measurements from Fig. 4 are replotted on linear axes to show absolute latency and memory differences as the input grows. Increasing the SSR state count to 16 preserves its measured long-sequence efficiency advantages under this benchmark configuration.

Diffusion Policy dataset and evaluation. The Diffusion Policy experiments follow the official implementation released with Diffusion Policy (Chi et al., 2023). Scripted Transfer Cube demonstrations from LeRobot (Cadene et al., 2026) are converted to the format required by the official implementation. For each paired seed, all compared operators use the same split of 49 training episodes and one validation episode, together with matched rollout initializations. Each selected policy is evaluated over 50 environment rollouts.

SSR parameterization and training settings. Table 10 lists the operator-specific parameters. Unless otherwise stated, SSR uses $M = 4$ states per head. The projection and readout configurations are specified in the corresponding result tables. Table 11 summarizes the main ALOHA training and evaluation settings for ACT and Diffusion Policy.

Table 10: SSR operator hyperparameters and parameterization.

Setting	Value / implementation
States per head M	4 by default; state-capacity ablations vary M.
Assignment temperature τ_a	1.0.
Readout temperature τ_r	1.0.
Minimum concentration $\alpha_{\min}$	10^{-3}, with $\alpha_{hm} = \alpha_{\min} + \mathrm{softplus}(\rho_{hm})$.
Numerical stabilizer ϵ	10^{-6}, applied as a denominator floor during state aggregation.
State centers C	Trainable, shape $H \times M \times d$; initialized from $\mathcal{N}(0, 0.02^2)$.
Concentration parameters ρ	Trainable, shape $H \times M$; initialized to zero ($\alpha_{hm} \approx 0.6941$ initially).
Readout	Center, state, or hybrid; hybrid uses $(c_{hm} + s_{bhm})/2$ as the readout anchor.
Input parameterization	Projected SSR uses independent W_Q, W_K, W_V; projection-free SSR directly partitions input features into heads.
Output projection	Retained in both projected and projection-free variants.
Distance normalization	Squared Euclidean distance; no explicit ℓ_2 normalization of Q, K, or C.

Statistical reporting. Main learning curves report the mean and one sample standard deviation across training seeds. For the main Diffusion Policy comparison, Last-10 first averages success rates over checkpoints at epochs $2550, 2600, \ldots, 3000$ within each training seed, then aggregates these per-seed averages across training seeds. Evaluation windows for additional tasks are specified in their respective table captions.

Module-level memory profiling. The module-level memory benchmark uses FP32 with $B = 8$, $D = 512$, $H = 8$, $d = 64$, $N_q = N_s = 301$, and $M = 4$ on an NVIDIA H200 NVL. Peak additional memory is the peak allocated CUDA memory minus the post-warmup allocated-memory baseline. The Transformer reference uses `nn.MultiheadAttention` with `need_weights=True`, whereas SSR uses the TorchInductor-fused implementation. These measurements characterize the

Table 11: Main training and evaluation settings for ACT and Diffusion Policy (DP).

Setting	ACT	Diffusion Policy
Model width / heads	$D = 512$, $H = 8$	$D = 256$, $H = 4$
Transformer depth	4 encoder / 1 decoder	8 decoder layers
Vision encoder	ResNet18, ImageNet pretrained	ResNet18Conv, no pretraining
Temporal setting	100-action chunk	obs. $= 2$, horizon $= 10$
Batch size	8	64
Optimizer	AdamW	AdamW
Learning rate	10^{-5}	10^{-4}
Weight decay	10^{-4}	10^{-3} (denoiser)
Training budget	2000 epochs / 10k updates	3050 configured epochs
LR schedule	None	Cosine, 1000-step warmup
EMA	No	Yes
Precision	FP32	FP32
SSR assign/read/out dropout	$0/0.1/0$	$0.3/0.3/0$
Train/validation split	$40/10$	$49/1$
Main training seeds	10	3
Checkpoint / summary	Minimum validation loss	Fixed-window checkpoint summary
Rollouts	50 per selected policy	50 per evaluated checkpoint

specified module implementations, not whole-policy memory or a comparison against every optimized attention backend.

D PSEUDOCODE OF SSR

Algorithm 1: Structured State Representation (SSR)

Inputs: queries $\mathbf{Q}$, keys $\mathbf{K}$, values $\mathbf{V}$, validity mask $\mathbf{g}$
Hyperparameters: heads H, states M, $\alpha_{\min}$, τ_a, τ_r, and $\epsilon > 0$
Readout configuration: $\mathcal{R} \in \{\text{center}, \text{state}, \text{hybrid}\}$
Parameters: $\mathbf{C} = \{\mathbf{c}_{hm}\}$, $\boldsymbol{\rho} = \{\rho_{hm}\}$, $\mathbf{W}_o$, and $\mathbf{b}_o$
Projected variant only: $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V$ and $\mathbf{b}_Q, \mathbf{b}_K, \mathbf{b}_V$
Input parameterization (choose one):
  Projection-free: $(\widetilde{\mathbf{Q}}, \widetilde{\mathbf{K}}, \widetilde{\mathbf{V}}) \leftarrow (\mathbf{Q}, \mathbf{K}, \mathbf{V})$
  Projected: $(\widetilde{\mathbf{Q}}, \widetilde{\mathbf{K}}, \widetilde{\mathbf{V}}) \leftarrow (\mathbf{Q}\mathbf{W}_Q + \mathbf{b}_Q,\ \mathbf{K}\mathbf{W}_K + \mathbf{b}_K,\ \mathbf{V}\mathbf{W}_V + \mathbf{b}_V)$
$d \leftarrow D/H, \quad (\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h)_{h=1}^{H} \leftarrow \mathrm{SplitHeads}(\widetilde{\mathbf{Q}}, \widetilde{\mathbf{K}}, \widetilde{\mathbf{V}})$
for $h = 1, \ldots, H$ **do**
  $\alpha_{hm} \leftarrow \alpha_{\min} + \mathrm{softplus}(\rho_{hm})$
  $\delta_{bhmn} \leftarrow \|\mathbf{k}_{bhn} - \mathbf{c}_{hm}\|_2^2$
  $a_{bhmn} \leftarrow -\alpha_{hm}\delta_{bhmn}/\tau_a$
  $p_{bhmn} \leftarrow \mathrm{softmax}_m(a_{bhmn})$
  $w_{bhmn} \leftarrow \dfrac{g_{bn}p_{bhmn}}{\max(\sum_{n'} g_{bn'}p_{bhmn'}, \epsilon)}$
  $\mathbf{s}_{bhm} \leftarrow \sum_n w_{bhmn}\mathbf{v}_{bhn}$
  $\mathbf{u}_{bhm} \leftarrow \begin{cases} \mathbf{c}_{hm}, & \mathcal{R} = \text{center}, \\ \mathbf{s}_{bhm}, & \mathcal{R} = \text{state}, \\ (\mathbf{c}_{hm} + \mathbf{s}_{bhm})/2, & \mathcal{R} = \text{hybrid}. \end{cases}$
  $r_{bhtm} \leftarrow \dfrac{\mathbf{q}_{bht}^{\top}\mathbf{u}_{bhm}}{\sqrt{d}\,\tau_r}$
  $\beta_{bhtm} \leftarrow \mathrm{softmax}_m(r_{bhtm})$
  $\mathbf{y}_{bht} \leftarrow \sum_m \beta_{bhtm}\mathbf{s}_{bhm}$
end for
$\widehat{\mathbf{Y}} \leftarrow \mathrm{Dropout}(\mathrm{OutProj}(\mathrm{MergeHeads}(\mathbf{Y}); \mathbf{W}_o, \mathbf{b}_o))$
Return: output $\widehat{\mathbf{Y}}$, structured states $\mathbf{S}$