Sensory Compression as a Unifying Principle for Action Chunking and Time Coding in the Brain

Sreejan Kumar[1,2,6 *], Matthieu B. Le Cauchois[3,4 *], Alexander Mathis[5], Lea Duncker[6,7], Jonathon R. Howlett[8,9 **], and Marcelo G. Mattar[2,**]

[1]Princeton Neuroscience Institute, Princeton University
[2]Department of Psychology, New York University
[3]Department of Cognitive Science, University of California, San Diego
[4]School of Engineering, École polytechnique fédérale de Lausanne (EPFL)
[5]School of Life Sciences, École polytechnique fédérale de Lausanne (EPFL)
[6]Zuckerman Mind, Brain, Behavior Institute, Columbia University
[7]Department of Neuroscience, Columbia University
[8]VA San Diego Healthcare System
[9]Department of Psychiatry, University of California, San Diego

[*]Joint first authors.
[**]Joint senior authors.

Abstract

The brain seamlessly transforms sensory information into precisely-timed movements, enabling us to type familiar words, play musical instruments, or perform complex motor routines with millisecond precision. This process often involves organizing actions into stereotyped "chunks". Intriguingly, brain regions that are critical for action chunking, such as the dorsolateral striatum (DLS), also exhibit neural dynamics that encode the passage of time. How such brain regions support both task-specific motor habits and task-invariant internal timing, two seemingly distinct functions, remains a fundamental question. Here we show, using recurrent neural network models, that these two functions emerge from a single computational principle: sensory compression, the functional compression of high-dimensional sensory information into a low-dimensional representation. We find that a sensory bottleneck forces the network to develop stable internal dynamics that implicitly encode time, which in turn serve as a scaffold upon which the brain learns action chunks in response to predictable environmental regularities. This mechanism unifies task-invariant time coding and sensory-guided motor timing as two outcomes of the same process of sensory compression, providing a general principle for how the brain mirrors environmental regularities in both internal stable neural trajectories and external consistent motor habits.

1 Introduction

Every day, we perform complex motor sequences with remarkable precision and without conscious effort—from playing a piano sonata to typing a familiar password. This is due to the brain's remarkable ability to leverage repeated temporal structure in our environments into complex sensorimotor behaviors that maximize reward while minimizing cognitive effort. A classic example of this is the development of action "chunks"—stereotyped sequences that are executed together as cohesive units and reused across similar contexts (Dezfouli & Balleine, 2012; Graybiel, 1998). Critically, action chunks are defined not only by their sequential structure, the specific order of actions (e.g., playing musical notes C, D, then E in a musical phrase), but also by the temporal structure in the precise timing and rhythm of each action and the intervals between them (e.g., the relative duration and spacing between notes). This precise timing is reliably reproduced across repeated performances of a chunk (Hidalgo-Balbuena et al., 2019) and is flexible in that time representations can persist regardless of task relevance or stimulus properties (Toso et al., 2021). While certain computational frameworks have separately addressed

how the brain learns action sequences (Lai & Gershman, 2021) and how it represents time (Tsao et al., 2022), we still lack a unifying principle that explains how the temporal structure of behavior is learned and encoded alongside its sequential structure.

Brain regions that support chunking, such as the dorsolateral striatum (DLS, Graybiel (1998)), also display neural dynamics that reliably encode elapsed time (Toso et al., 2021; Tsao et al., 2022). The DLS is essential for acquiring and executing rigid, stereotyped action chunks (Graybiel, 1998), while simultaneously, DLS neurons show stable neural trajectories that encode elapsed time during repetitive behaviors, functioning as an internal clock even when timing is task-irrelevant (Toso et al., 2021; Tsao et al., 2022). The co-existence of these two functionalities raises a fundamental question: are there computational principles that can explain why we see both stereotyped motor sequences and flexible temporal encoding in the same neural circuits? The coexistence of these two distinct functions in the same brain regions suggests an underlying shared mechanism that has yet to be identified.

In existing computational frameworks, the neural mechanisms of action chunking and time-tracking have been addressed separately. For action chunking, the policy compression framework posits that the brain simplifies complex behaviors by compressing its decision-making policy to operate within its capacity limits (Lai & Gershman, 2021). This compression leads to the formation of habitual action chunks, where frequently co-occurring states and actions are bundled into efficient, reusable units (Lai et al., 2025). Models of prospective neural timing posit that stable dynamics that evolve predictably over time serve as an implicit internal clock (Tsao et al., 2022). These stable neural trajectories are common when repeated temporal patterns in the environment, whether through external sensory cues or internally generated motor routines, provide predictable structure. Both perspectives highlight the role of environmental predictability, but neither directly explain why chunking and time coding would co-occur in the same brain regions.

Here, we hypothesize that action chunking and time coding are two manifestations of a single, unifying principle: *sensory compression.* In particular, after repeated exposure to an environment with predictable structure, the brain's sensorimotor system can leverage this predictability to compress sensory information into a minimal representation that can trigger the appropriate motor response. The brain's sensorimotor pathway has itself been interpreted as an information bottleneck (Tishby et al., 2000) from stimulus to action. As proposed by Zareian et al. (2023), the sensorimotor pipeline consists of incoming sensory information as input and motor action as output, with a recurrent loop between DLS and motor cortex at its center. There is considerable evidence of converging sensory inputs to both DLS and motor cortex, producing favorable anatomical conditions for compression. In the case of motor cortex, an *in vitro* slice electrophysiology study where connectivity $S1$ somatosensory cortex and $M1$ motor cortex was preserved, Petrof et al. (2015) found that many axons in $S1$ can project to individual $M1$ cells. There is a massive dimensional reduction from sensory cortex to striatum—where millions of cortical neurons project onto orders of magnitude fewer striatal cells (Hintiryan et al., 2016; Paxinos & Franklin, 2019). The work of Morris et al. (2003), which found uncorrelated neural activity in basal ganglia output nuclei despite this massive convergence, provides direct evidence that this dimensional reduction is involved in an optimal compression of cortical activity.

We suggest that sensory compression *intrinsically* generates stable neural trajectories that track time as an inevitable byproduct of dimensionality reduction, not as a learned feature. When exposed to environments with temporal regularities, these same constraints enable policy compression (Lai & Gershman, 2021), allowing frequently co-occurring actions to be bundled into chunks (Lai et al., 2025). Thus, sensory compression provides a single mechanism that explains both phenomena: creating an intrinsic temporal scaffold that can be flexibly utilized for organizing behavior.

To formalize and test this hypothesis, we developed a recurrent neural network (RNN) model constrained by bottlenecks of varying dimensionality (Figure 1A). Our architecture implements the hypothesized sensory compression through a bottleneck layer connecting sensory encoding to recurrent dynamics, analogous to converging sensory cortical inputs that feed into DLS-motor cortex loops (Grillner, 2025; Zareian et al., 2023). We then study the model's behavior and neural representations on a sensorimotor chunking task with temporal regularities. We show that networks with stronger sensory bottlenecks spontaneously develop both target phenomena: they chunk action sequences in patterns matching human behavior (Lai et al., 2025) and generate

smooth, time-encoding neural trajectories. Critically, we then show that our model reproduces seemingly paradoxical neurophysiological experimental findings in rodents concerning temporal coding in the DLS. In particular, we show that bottlenecked networks maintain stimulus-invariant time encoding regardless of task demands (as in Toso et al. (2021)), and that sensory perturbations affect behavioral timing while preserving sequence structure (as in Hidalgo-Balbuena et al. (2019)). Networks without sensory bottlenecks fail to show either phenomenon. These results reveal sensory compression as a unifying computational principle for how the brain translates environmental regularities into predictable internal trajectories and precisely-timed, automatic external actions.

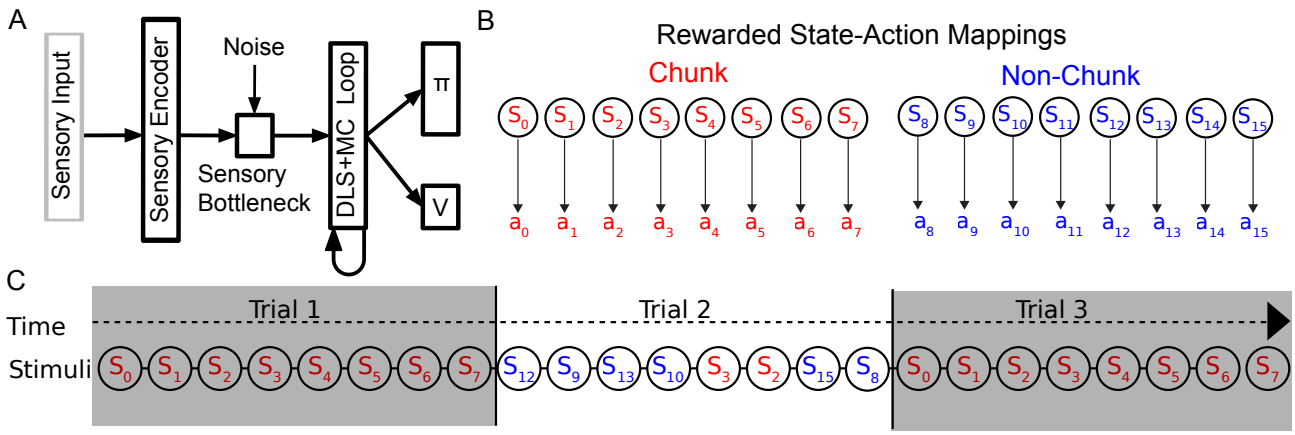

Fig. 1. Model Architecture and Sensorimotor Chunking Task. (A) Architecture of the Sensory Bottleneck RNN model. The model consists of a sensory encoder (which has 64 units) and an LSTM (also has 64 units) linked by a stochastic bottleneck layer. We vary the number of units in the bottleneck layer to find the effect of having a sensory bottleneck on downstream behavior and neural representations. The grey module denotes the raw sensory input. (**B**). **Sensorimotor chunking task.** An episode consists of a sequence of sensory inputs. The number of possible actions equals the number of possible stimuli and each stimulus has one rewarded action. (**C**). **High-level task Structure.** A sequence of T sensory inputs S_i is sampled with either a structured or random scheme. In structured episodes, a specific set of stimuli are shown with a fixed order (i.e. a "chunk"). In random episodes, stimuli at each timestep are chosen randomly, with the stimuli that are a part of the chunk sequence having a lower probability. Here, shaded trials 1 and 3 represent example structured episodes and non-shaded trial 2 represents an example random episode.

Results

Sensory Bottlenecks Recover Human-like Performance Patterns in an Action Chunking Task

To test the hypothesis that sensory compression drives action chunking, we first studied a sensorimotor task specifically designed to elicit and study this behavior. In this task, inspired by Lai et al. (2025), participants observe sequences of stimuli drawn from a set of 16 possible items and must select the corresponding action for each stimulus to receive a reward (Figure 1B,C). The key manipulation is that eight of these stimuli (S0-S7, "chunk stimuli") frequently appear in a fixed temporal order, forming a “chunk” (structured episodes). The remaining "non-chunk" stimuli always appear in random order. In particular, when a trial initiates, there is a $P_{init} = 0.45$ that a chunk sequence initiates and all chunk stimuli are shown in a fixed temporal order. On non-chunk "random episodes", stimuli are drawn randomly, with chunk stimuli having a lower probability ($P_{aux} = 0.09$) than non-chunk stimuli. This means that, although chunk stimuli can appear individually in random contexts, they more frequently co-occur together in a chunk via structured episodes. This design creates an opportunity for behavioral policy compression: rather than treating each stimulus independently, an efficient learner can recognize and exploit the predictable chunk sequence.

Human performance on this task reveals two signatures of capacity-limited processing that any valid model of chunking must reproduce. First, human performance decreases as the total number of stimuli increases, reflecting the cognitive cost of maintaining more stimulus-response mappings (Figure 2A, light vs. dark bars). Second, humans perform significantly better on structured trials containing the chunk than on random trials, demonstrating that they successfully exploit the temporal regularity (Figure 2A, red vs. blue bars). These two effects—degraded performance with increased memory load and improved performance with predictable

structure—are hallmarks of policy compression in capacity-limited systems (Lai & Gershman, 2021).

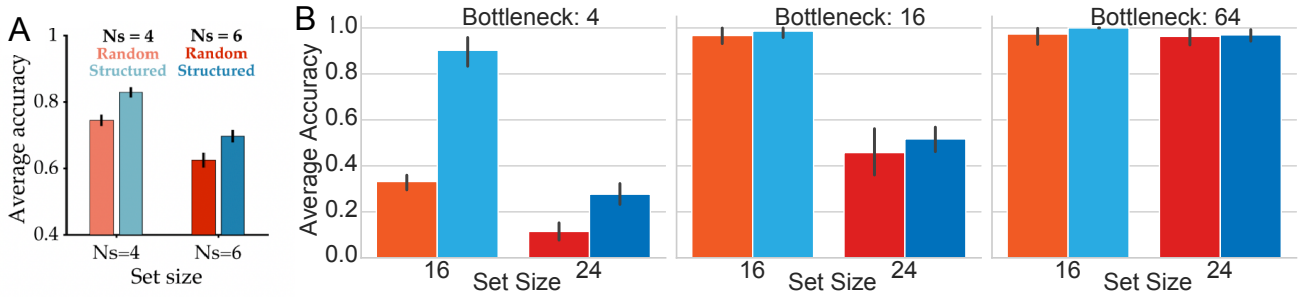

Fig. 2. Human and model performance on the chunking task. (A) Human behavioral signatures of chunking. Data from Lai et al. (2025) showing two key effects: performance decreases with larger set sizes (Ns=4 vs. Ns=6 stimuli), and performance improves on structured trials containing chunks versus random trials (red vs. blue). (**B**). **Only strong sensory bottlenecks reproduce both human signatures.** Models with varying bottleneck dimensions tested on the same task. The 4-dimensional bottleneck (left) shows both effects, the 16-dimensional (middle) shows only the set size effect, and the 64-dimensional (right) shows neither. Error bars indicate 95% CI across training seeds.

To test whether sensory compression could reproduce these human behavioral signatures, we designed a neural network architecture mirroring the anatomical organization of cortico-striatal circuits. Our model consists of three key components: (1) a sensory encoder that processes input stimuli (64 units), (2) a bottleneck layer that compresses these representations from high to low dimensions (varied from 4 to 64 units), and (3) an LSTM recurrent layer (64 units) that generates behavioral policies (Figure 1A). The bottleneck layer compresses sensory input before passing it to an LSTM that generates action policies. This architecture parallels the convergence that happens from sensory cortex to motor cortex and striatum (Hintiryan et al., 2016; Paxinos & Franklin, 2019; Petrof et al., 2015), followed by recurrent loops between striatum and motor cortex that generate habitual behaviors (Grillner, 2025). After training, we evaluated each model's performance on both structured and random trials with different set sizes, matching the conditions used in human experiments.

We first asked whether sensory bottlenecks produce the capacity limitations seen in human chunking. The 64-dimensional model—effectively lacking compression—performed equally well across all conditions, showing neither the set size effect nor the structured trial advantage (Figure 2B, right panel). This ceiling performance indicates that without sensory constraints, the model simply memorizes all stimulus-response mappings independently. The 16-dimensional model showed intermediate behavior: performance decreased with set size but showed no advantage for structured trials (Figure 2B, middle). Only the 4-dimensional bottleneck reproduced both human signatures: performance decreased with set size and improved on structured versus random trials (Figure 2B, left). These results demonstrate that strong sensory compression produce human-like chunking behavior—i.e., only under sufficient constraints will models solve the task by exploiting temporal regularities, rather than memorizing stimulus-response relationships.

Sensory Bottlenecks Lead to Behaviors Consistent with Policy Compression

Having established that sensory bottlenecks reproduce human performance patterns, we next asked whether they also capture the specific behavioral signatures predicted by policy compression theory. The policy compression framework (Lai & Gershman, 2021) conceptualizes action chunking as an information-theoretic problem where capacity-limited agents must balance reward maximization against the cost of maintaining complex policies. This framework predicts specific behavioral phenomena that distinguish truly chunked behavior from simple memorization, including action slipping, perseveration, and enhanced generalization (Gershman, 2020; Lai et al., 2025). If sensory compression drives chunking through policy compression, bottlenecked models should exhibit all these signatures.

First, we tested whether sensory bottlenecks creates simpler behavioral policies. Following Lai & Gershman (2021), we quantified policy complexity using mutual information between states and actions, a measure of how much stimulus information is preserved in action selection (Lai & Gershman, 2021; Lai et al., 2025). We found that decreasing bottleneck dimension resulted in lower policy complexity (Figure 3A). These results corroborate

previous work relating dimensionality of state representations to policy complexity (Liu et al., 2025). Plotting the reward–complexity trade-off (Gershman, 2020) revealed the expected pattern: models with more complex policies achieve higher returns, but with diminishing gains at high complexity (Fig. 3B). These results confirm that our model operates under the core principles of policy compression.

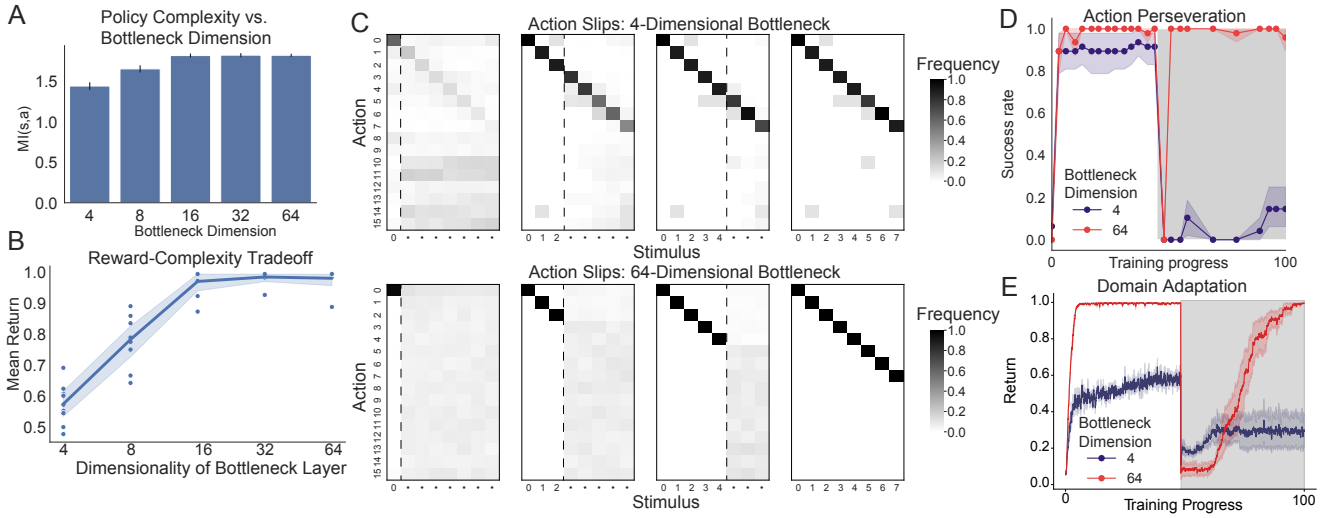

Fig. 3. Sensory bottlenecks induce behavioral signatures of policy compression. (**A**) **Policy complexity decreases with bottleneck strength.** Policy complexity, measured by the mutual information between states and actions, across bottleneck dimensions. Error bars are CIs across training seeds. (**B**) **Reward-complexity tradeoff.** Higher-complexity policies achieve better returns with diminishing benefits (Gershman, 2020). (**C**) **Bottlenecked models exhibit action slipping.** Action distributions following partial chunk initiations (· represents marginalization over future states, dashed lines represent end of the partial chunk initiation). The low-dimensional bottleneck model (top) inflexibly executes the chunk sequence even when only partially cued, whereas the high-dimensional model (bottom) does not. (**D**) **Bottlenecked models exhibit action perseveration.** Success rate for state S3 when its rewarded action changes mid-training (gray region). Low-dimensional model perseverates on the old policy for longer than the high-dimensional model. (**E**) **Bottlenecked models show faster domain adaptation.** Mean return when input features change from one-hot to random vectors mid-training. Bottleneck models show faster initial recovery. Error bars: 95% CI across seeds.

To further validate our model, we tested whether it exhibited the classic behavioral errors associated with compressed, chunked policies. A hallmark of chunking is "action slipping," where a familiar sequence is executed inflexibly even when inappropriate (Norman, 1981). We tested this by presenting partial chunk initiations (e.g., stimuli S0-S2) followed by random stimuli requiring different actions. We found that low-dimensional models inflexibly completed the entire sequence, ignoring subsequent, non-chunk stimuli (Figure 3C, top). In contrast, non-bottleneck models remained responsive to the actual stimuli presented (Figure 3C, bottom). Similarly, compressed policies are known to be less flexible, leading to "perseveration" on old strategies (Dickinson, 1985). To test this, we switched the rewarded action for stimulus S3 mid-training. We found that low-dimensional models perseverated on the old, incorrect action for significantly longer than non-bottleneck models (Figure 3D). These results show that the sensory bottleneck induces the same kinds of inflexible, habitual errors that characterize human action chunking.

We then asked whether this trade-off for inflexibility conferred any benefits, such as enhanced generalization. Previous work suggests that capacity constraints can improve a system's ability to adapt (Igl et al., 2019; Lerch & Sims, 2019). To test this, we performed a domain adaptation experiment where we changed the sensory features of the stimuli halfway through training (replacing the input vector for each stimulus with a new random vector) while keeping the task structure the same. The bottleneck model, having learned a compressed, abstract policy, adapted to the new sensory inputs significantly faster than the non-bottleneck model, which had to re-learn the entire stimulus-response mapping (Figure 3E). This demonstrates that policy compression via a sensory bottleneck provides a key advantage: faster generalization to novel sensory environments.

Finally, we investigated the mechanism by which the bottleneck enable chunking. We hypothesized that reducing bottleneck size decreases the sensory information available to the policy, forcing the model to rely more on internally generated temporal dynamics. Consistent with this, we found that the ability to decode the current

stimulus from the network's hidden state was significantly worse in models with smaller bottlenecks (Figure S1). To confirm that the bottleneck models relied more on internally generated temporal dynamics, we froze the model's recurrent weights during training, preventing the development of internal dynamics. This manipulation severely impaired the bottleneck model but had little effect on the non-bottleneck model (Figure S5). Moreover, high-dimensional models trained exclusively on structured trials still failed to chunk (Fig. S4). Together, these results suggest that sensory compression, not exposure alone, is required for the emergence of time-sensitive chunking, as the absence of stimulus-specific information forces models to exploit temporal structure.

Sensory Bottlenecks Produces Stable and Smooth Recurrent Trajectories that Implicitly Encode Time

Having shown that sensory bottlenecks force a reliance on recurrent dynamics, we next investigated the structure of these dynamics and their relationship to time encoding. To address this, we examined hidden state trajectories using Principal Components Analysis (PCA) for models with either a 4-dimensional (strong compression) or 64-dimensional (no compression) bottleneck, run in both the chunked training environment and a control IID environment where all stimuli (both chunk and non-chunk) were drawn fully independently (IID environment, Fig. 4). Note that the size of the LSTM's hidden state in both models have the same dimensionality (64 neurons); the two models only differ in the bottleneck's dimensionality (or, equivalently, the dimensionality of the LSTM's input space). In the 4-dimensional bottleneck model, chunk stimuli evolved along a consistent, smooth trajectory, while non-chunk stimuli collapsed into a single cluster. These trajectories were stable across trials and arranged such that position along the trajectory corresponded to elapsed time. By contrast, the 64-dimensional bottleeck model showed fragmented, stimulus-specific clusters and less smooth, less temporally structured trajectories. In the IID environment, the 4-dimensional bottleneck model still produced stable, smooth, time-ordered trajectories, whereas the 64-dimensional model showed highly variable, stimulus-driven paths—consistent with its higher sensory decodability (Fig. S1).

To quantify representational structure, we performed representational similarity analysis (RSA, Figure 4) separately across stimuli ("state RSA") and across timepoints ("time RSA"). State RSA revealed that the 4-dimensional model represented chunk stimuli in a temporally ordered manner and collapsed non-chunk stimuli into a common representation, a structure absent in the IID environment. The 64-dimensional model instead represented each stimulus distinctly, consistent with memorization. Time RSA showed strong temporal autocorrelation in the 4-dimensional model's chunk environment trajectories, and this autocorrelation persisted in the IID environment despite the absence of external temporal structure. These results suggest that sensory compression gives rise to an internal temporal scaffold, reflected in the geometry of its neural trajectories.

To rigorously test this observation, we developed quantitative metrics to test whether bottleneck models systematically produced *stable*, *smooth*, and *time-encoding* trajectories. Time encoding was measured by training a linear decoder to predict elapsed time from hidden state; stability was measured as trial-to-trial variability; smoothness was quantified by mean acceleration and curvature of trajectories. Across both chunked and IID environments, the low-dimensional bottleneck model produced trajectories that were significantly more stable (lower variability), smoother (lower acceleration and curvature), and contained more linearly decodable information about elapsed time (higher R^2) than the non-bottleneck model (Figure 5). This held true even when trial lengths were varied randomly, dissociating absolute time from task progress (Figure S2), and was robust to architectural changes, such as using a vanilla RNN instead of an LSTM (Figure S3). These results quantitatively confirm that sensory bottlenecks produce stable, smooth, time-encoding neural dynamics.

Finally, we asked whether these time-encoding dynamics are an intrinsic property of the compression process or if they depend on learning. To distinguish these possibilities, we analyzed the dynamics of untrained, randomly initialized (i.e. uniform weights) networks. We found that random-weight models with a sensory bottleneck already exhibited strong time-encoding trajectories, whereas non-bottleneck random-weight models did not (Figure S6). This is a particularly striking result because both models use the exact same RNN architecture and only differ in the dimensionality of their inputs. The bottleneck model's sensory encoder randomly projects the input to the RNN into a lower-dimensional space before feeding into the RNN, whereas the non-bottleneck model's sensory encoder retains the input's original dimensionality. Through this result, we see that when a recurrent network's input dimensionality is sufficiently dimensionally-constrained, the variance of the hidden

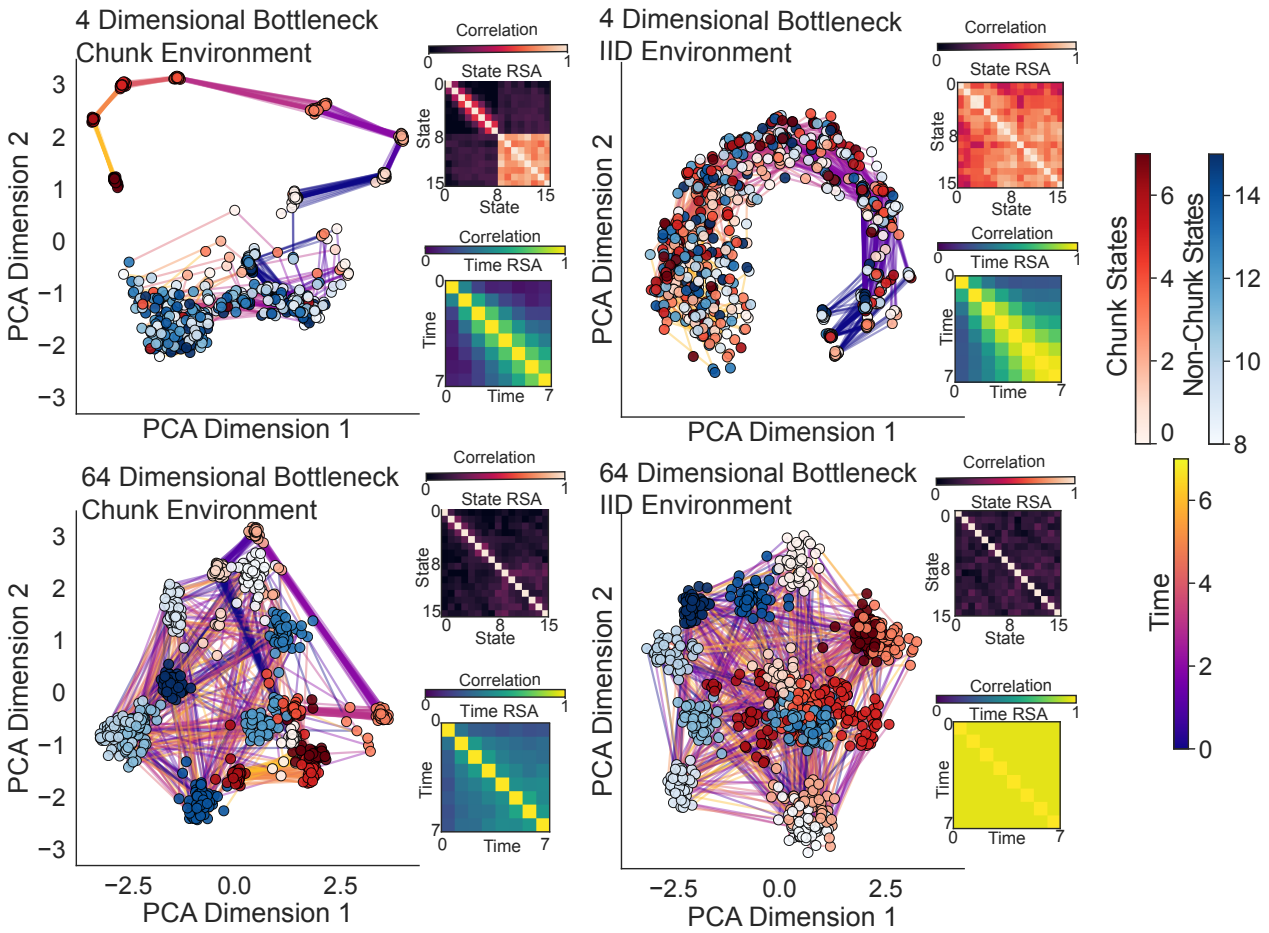

Fig. 4. Sensory bottlenecks create smooth, time-encoding neural trajectories. Principal components analysis (PCA) of hidden state evolution for low-dimensional (bottleneck dim=4, top) and high-dimensional (bottleneck dim=64, bottom) models. (**Left**) In the chunking task, the bottleneck model develops a smooth, stereotyped trajectory for the chunked sequence. (**Right**) In a control task with random (IID) stimuli, the bottleneck model still produces a smooth trajectory, while the non-bottleneck model does not. Points are colored by stimulus identity (chunk vs. non-chunk) and lines by timestep. Representational Similarity Matrices (RSM) of each condition are shown as insets to each PCA plot. The state RSM shows all pairwise correlations of the mean activity evoked for each stimuli. The time RSM shows all pairwise correlations of the mean activity evoked for each timestep. Note that both models have the same dimensionality in the recurrent layer (64 units) but differ in the bottleneck layer's dimensionality, which is the *input* to the recurrent layer (4-dimensional vs. 64-dimensional).

state overtime is dominated by the network's internal dynamics rather than the variance of the input, leading to highly stable trajectories that can implicitly encode time. This critical finding demonstrates that the capacity for temporal encoding can arise from the process of sensory compression itself, existing independently of and prior to task-specific learning. Through this result, we provide a resolution to the observation of how brain regions like the DLS can exhibit robust time coding even when timing is irrelevant to the task (Toso et al., 2021): the temporal scaffold is an intrinsic property created by sensory compression independently of task demands, and this scaffold is then refined and utilized by task-specific learning to produce precisely timed, habitual behaviors.

Sensory Bottlenecks Explain Task-Invariant Time Coding in Dorsolateral Striatum

Our finding that sensory compression creates an intrinsic temporal scaffold offers a direct explanation for the observation that the DLS encodes time even when it is task-irrelevant. To test this, we simulated the key experiment from Toso et al. (2021). In their study, rats performed a stimulus comparison task where they had to judge either the duration or intensity of two sequential stimuli (Figure 6A). The critical finding was that DLS population activity encoded elapsed trial time with high fidelity, and this encoding was invariant to the specific properties of the stimuli (their duration or intensity) and to which task the animal was performing (Figure 6B).

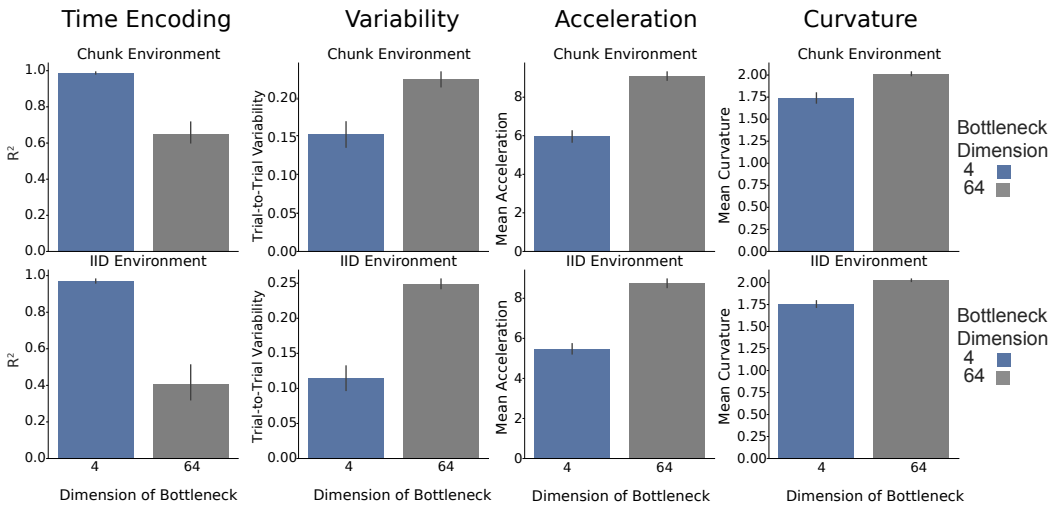

Fig. 5. Bottlenecked trajectories are quantitatively more stable, smooth, and time-encoding. (**Top row**) In the chunking task, the low-dimensional model shows higher time decoding performance (R^2), lower trial-to-trial variability, and lower state space acceleration and curvature. (**Bottom row**) The same effects hold even when stimuli are presented in a random (IID) order. Error bars are CIs over training seeds.

Note that timing is irrelevant to the intensity task since the animals only have to judge the stimuli's magnitude. This suggests the DLS maintains an internal clock that runs independently of specific sensory events.

We asked whether our sensory bottleneck model could reproduce this stimulus-invariant time coding. We trained our bottleneck (dim=4) and non-bottleneck (dim=64) models on both the duration and intensity comparison tasks. We then decoded elapsed time from the hidden state activity, replicating the analysis of Toso et al. (2021). We found that the low-dimensional model successfully reproduced their key finding: time could be decoded accurately, and this decoding was invariant to the intensity or duration of the presented stimuli (Figure 6C). In contrast, the non-bottleneck model's time code was less accurate and was contaminated by stimulus features. Furthermore, the bottleneck model's time code was robust across correct and incorrect trials, another key finding from the original study (Figure S8). We conducted the same neural trajectory analysis that we performed in Figure 5 and found that, across both tasks (duration and intensity comparison), the 4-dimensional model produces stable and smooth trajectories than the 64-dimensional non-bottleneck baseline model (Figure 6D). Figure S9 shows PCA trajectories of each model on both tasks. These results demonstrate that the intrinsic temporal scaffold created by sensory compression can explain the task- and stimulus-invariant nature of time coding in the DLS.

This finding explains stimulus-invariant time encoding, but raises a paradox: if the time code of DLS is intrinsic and stimulus-invariant, how is it used to guide precisely-timed, sensory-dependent behaviors? The work of Hidalgo-Balbuena et al. (2019) shows that the DLS actively uses somatosensory input to time motor habits. Although these findings may at first seem at tension with each other, our framework accommodates both findings: sensory compression creates an intrinsic time-encoding scaffold (explaining the results of Toso et al. (2021)), and learning then maps specific sensory inputs onto this scaffold to guide behavior (in line with results of Hidalgo-Balbuena et al. (2019)). In the next section, we directly test this second prediction.

Sensory Bottlenecks Enable Sensory-Guided Timing of Motor Habits

Having shown that sensory compression creates intrinsic time-encoding dynamics, we next tested whether these dynamics can be flexibly guided by sensory input to produce precisely-timed behaviors. As a critical test case, we considered a recent study showing that, while DLS maintains stable motor sequences during habit execution, somatosensory input causally controls the timing of these sequences (Hidalgo-Balbuena et al., 2019). Using a novel treadmill task, rats learned a precisely timed motor habit, developing a stereotyped "Front-Back-Front" (FBF) strategy: they first allow the treadmill to carry them backward, then jog steadily, and finally sprint to a goal zone at the correct time (Figure 7A). Optogenetic perturbation of somatosensory input to the DLS disrupted the timing of this habit without altering the sequence of actions, demonstrating that sensory input provides a

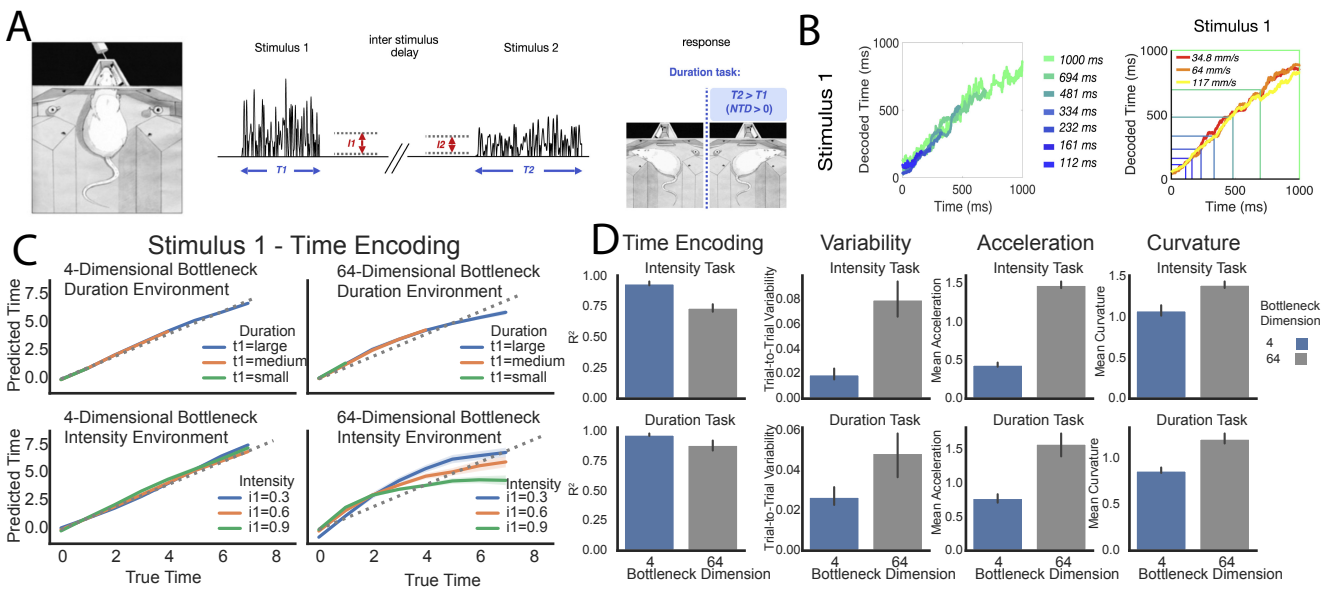

Fig. 6. Sensory bottlenecks reproduce stimulus-invariant time coding in the DLS. (**A**) **The stimulus comparison task from Toso et al. (2021)**. Rats were trained either on duration comparison (T1 vs. T2) or intensity comparison (I1 vs. I2). (**B**) **Experimental data showing that time decoding from DLS population activity is invariant to stimulus properties**. Both plots show true time vs. decoded time from DLS neural activity. Each line represents different stimulus durations (left plot) or different stimulus intensities (right plot). The fact that these lines overlap so much suggests that the DLS' encoding of time does not change much across different stimulus properties. (**C**) **Model replication of the experiment.** The low-dimensional bottleneck model reproduces this stimulus-invariant time coding, whereas the high-dimensional model's time code is stimulus-dependent. Plots show decoding accuracy across stimulus conditions. (**D**) **Quantitative comparison of stability, smoothness, and time-coding strength** for bottleneck vs. high-dimensional models, replicating the key features of DLS activity. Error bars are 95% confidence intervals across training seeds.

critical temporal reference. The DLS also exhibited smooth, time-encoding neural trajectories during this behavior (Figure 7B).

We first set out to instill the complex, multi-phase FBF strategy in our networks by developed a reinforcement learning environment that mimicked the treadmill task designed to produce this behavior. On this task, agents control their position on a virtual treadmill that moves backward at constant velocity and must reach a goal zone at a specified target time. The agents have three actions—hold (passive drift backward), jog (partial resistance), and sprint (forward movement)—corresponding to the three phases of the rat FBF strategy. Critically, we provided temporal reference information only during the jog action, mimicking Hidalgo-Balbuena et al. (2019)'s finding that somatosensory timing signals are most prominent during consistent motor phases. This design allows us to test whether sensory bottlenecks can both generate stereotyped action sequences and use sensory input to guide their timing. We found that both models learned to reach the goal near the target time, both using qualitatively different strategies. The 64-dimensional model (no compression) achieved higher temporal precision but alternated rapidly between actions throughout each trial (Figure 7C-D). In contrast, the 2-dimensional model (strong compression) spontaneously organized its behavior into three distinct phases that closely matched the rat FBF strategy: an initial hold phase, a sustained jog phase, and a final sprint (Figure 7D).

Next, we tested if the model's timing was dependent on sensory input in the same way as the DLS. To mimic the optogenetic experiments, we took the action-gated temporal somatosensory cue (available only during the 'jog' phase) and perturbed this signal mid-trial by artificially adding 10 timesteps to the temporal reference. In the low-dimensional model, this perturbation significantly disrupted the final arrival time while preserving the three-phase sequence structure. In the non-bottleneck model, the perturbation had minimal effect (Figure 7F). This result directly replicates the core finding of Hidalgo-Balbuena et al. (2019) that sensory input serves as a temporal reference for the timing of motor habits.

Finally, we examined the model's neural dynamics to see if they explained this behavioral pattern. The bottleneck model, like the rat DLS, produced stable, smooth, and time-encoding neural trajectories during the FBF habit

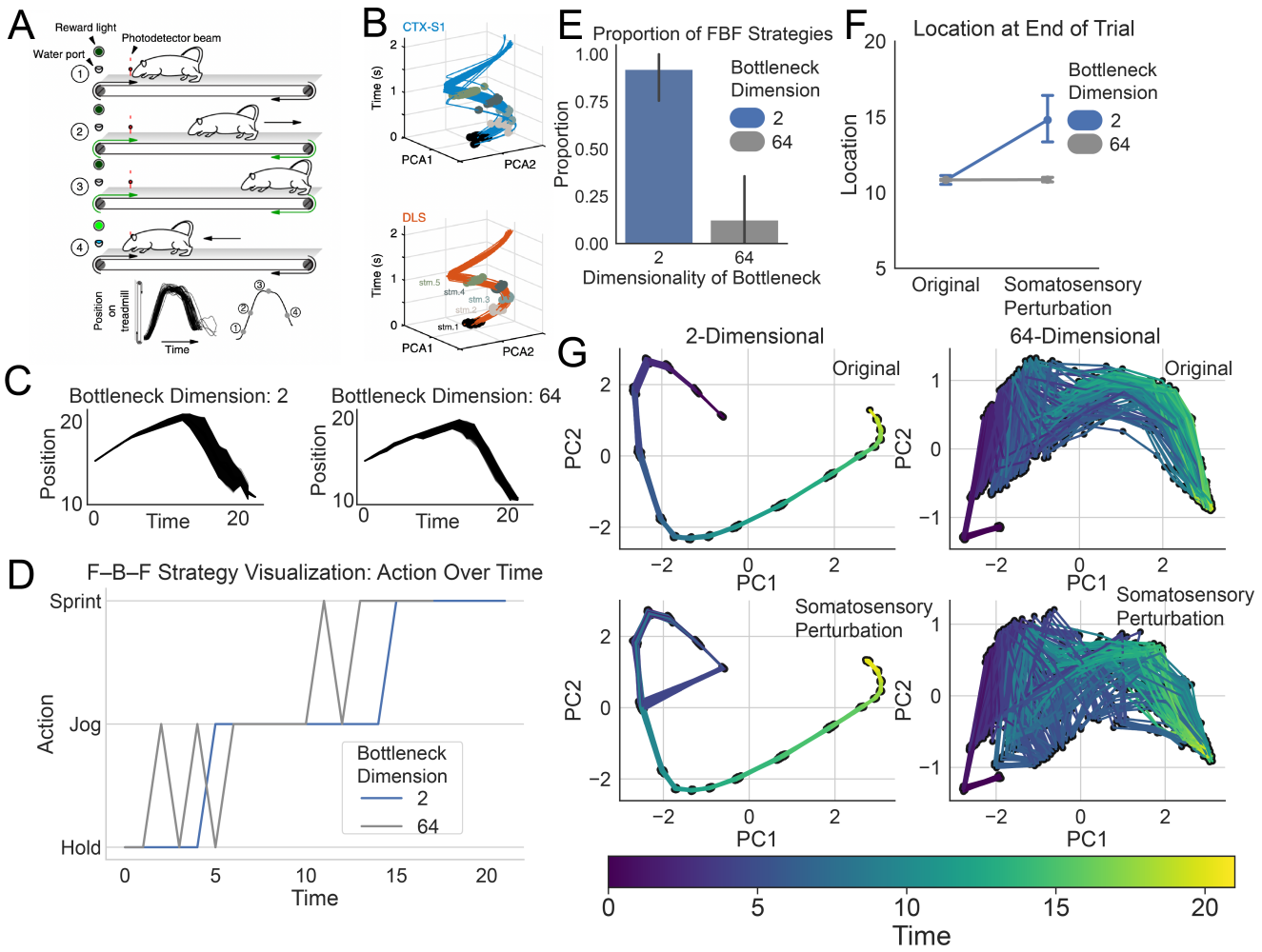

Fig. 7. Sensory bottlenecks reproduce sensory temporal reference effects in DLS. (**A**) **Treadmill navigation task.** Rats develop a stereotyped Front-Back-Front (FBF) strategy to reach a goal at a specified time. (**B**) **DLS shows consistent, smooth neural trajectories during FBF execution** (data from Hidalgo-Balbuena et al. (2019)). (**C**) **Bottleneck models achieve precise timing through chunked actions.** Agent position over time showing goal arrival. (**D**) **Action sequences reveal chunking in bottleneck models.** Example action trajectories for each model. The lower-dimensional bottleneck develops consistent FBF trajectories by chunking primitive actions together (hold → jog → sprint) whereas the higher-dimensional model does not. (**E**) **Quantification of FBF strategy emergence.** Proportion of trials showing clear FBF structure. The lower-dimensional model produce significantly more action trajectories consistent with an FBF strategy. Errorbars are CI's over training seeds. (**F**) **Sensory perturbation affects timing in bottleneck models.** Perturbing the sensory temporal reference input causes larger timing errors in 2-dimensional models, matching experimental findings. (**G**) **Neural trajectories remain stable under perturbation.** PCA trajectories of each model on original trials (top) vs. somatosensory-perturbed trials (bottom). The low-dimensional model produces consistent trajectories, like the DLS trajectories shown in B, even when the somatosensory temporal reference input is perturbed. The high-dimensional model does not.

(Figure 7G, top). Critically, when we perturbed the sensory input, the underlying neural trajectory remained intact; the model simply traversed the same trajectory it would have without the perturbation (Figure 7G, bottom). This explains why the timing was disrupted while the action sequence was preserved. The non-bottleneck model, in turn, showed much more variable and unstable trajectories that did not have any clear consistent effect from the somatosensory perturbation as the bottleneck model did. Quantitative analysis confirmed that 2-dimensional model trajectories were more stable, smooth, and time-encoding (Figure S10), matching the DLS recordings from Hidalgo-Balbuena et al. (2019) (Figure 7B). These results demonstrate that sensory compression creates a stable neural scaffold for action sequences, with sensory input controlling progression speed along this scaffold rather than the scaffold structure itself. Together, these results resolve the second half of the DLS time encoding paradox, demonstrating that the intrinsic stimulus-invariant temporal scaffold created by sensory compression is the substrate upon which learning anchors sensory cues to produce precisely timed, stimulus-specific habitual actions.

Discussion

Our work identifies sensory compression as a fundamental principle that unifies two key functions of the sensorimotor system: the formation of stereotyped motor habits via action chunking and the encoding of time. We propose that compression during sensory processing intrinsically generates stable, smooth neural trajectories that serve as a temporal scaffold. When an agent is exposed to environmental regularities, this scaffold enables the learning of temporally structured behaviors like action chunking, where predictable action sequences are compacted into cohesive, reusable units. To test this principle, we developed a recurrent neural network model with a sensory bottleneck (Figure 1A) and found it successfully reproduces human-like chunking performance (Figure 2) and the behavioral signatures of policy compression (Figure 3). Critically, these behaviors were supported by the emergence of stable, smooth, time-encoding neural trajectories (Figure 4, Figure 5), demonstrating that both behavioral habits and their underlying neural time code can arise from the single process of sensory compression.

The Dorsolateral Striatum (DLS) exemplifies this dual role in biological systems. Experimental studies show that DLS activity is crucial for acquiring and executing rigid, stereotyped action chunks (Graybiel, 1998) and simultaneously exhibits stable, time-encoding neural trajectories (Hidalgo-Balbuena et al., 2019; Toso et al., 2021; Tsao et al., 2022; Yang et al., 2024). Our model explains findings by Toso et al. (2021), who showed that DLS time coding is invariant to stimulus properties and task demands (Figure 6). While such invariance may seem inconsistent with frameworks in which temporal codes emerge only from learned structure (Tsao et al., 2022), sensory compression offers a simple explanation: it naturally produces stable, time-encoding dynamics independent of task relevance. These intrinsic dynamics then provide the foundation for task-specific behavior.

On the other hand, DLS time representations are also critically dependent on sensory input for guiding behavior. Previous work suggests that sensory feedback is causally required for the precise timing of motor habits (Hidalgo-Balbuena et al., 2019). Our model reconciles this finding by showing how sensory inputs are mapped onto the intrinsic temporal scaffold. In a simulation of their treadmill task, our bottleneck model spontaneously developed the stereotyped "Front-Back-Front" strategy by chunking primitive actions (Figure 7DE). Mimicking their optogenetic experiments, we found that perturbing a sensory timing signal disrupted timing of behavior but left the chunked action sequence intact. The neural trajectories in our bottleneck model were stable and smooth, matching the DLS recordings from Hidalgo-Balbuena et al. (2019), and these trajectories persisted in smoothness and stability even under perturbation (Figure 7G). Together, these results illustrate our framework's dual role: sensory compression creates an *intrinsic*, stimulus-invariant scaffold (explaining the Toso et al. result; Figure S6), while learning determines how *extrinsic* sensory inputs guide traversal along this scaffold to produce precisely timed, stimulus-dependent habits (explaining the Hidalgo-Balbuena et al. result).

Our model, while capturing key functional phenomena, is necessarily an abstraction of the underlying biological systems. We used discrete time and action spaces to model experiments that involved continuous motor outputs and timing (Hidalgo-Balbuena et al., 2019; Toso et al., 2021). For instance, the original task in Hidalgo-Balbuena et al. (2019) involved continuous running speed, whereas our model used discrete actions corresponding to the observed phases of movement. This approach was designed to isolate the high-level computational principles rather than to replicate spike-level neural data. Consequently, our simulations should be interpreted as demonstrations of how a general principle can account for high-level structure in neural and behavioral data across different paradigms. A valuable direction for future work will be to develop more detailed biophysical models that can capture the sensorimotor compression process at the spike level within high-fidelity, continuous-time environments.

Our framework further predicts that time encoding should serve as a signature of active compression, appearing in different brain regions depending on the stage of skill acquisition. This may explain the observed migration of temporal representations from prefrontal cortex to the striatum during learning (Tunes et al., 2022). Early in learning, prefrontal circuits may engage compression to handle working memory demands, whereas after extensive training, the computational load shifts to the striatum as behaviors become automatized (Mizes et al., 2024). The striatum's highly convergent anatomy (Zareian et al., 2023) and compact volume relative to its cortical inputs (Hintiryan et al., 2016; Paxinos & Franklin, 2019) provides a natural substrate for compression (Morris et al., 2003), which, when engaged, gives rise to the robust time encoding seen in well-trained animals (Toso et al., 2021). Both motor cortex and striatum can exhibit time-encoding dynamics (Hidalgo-Balbuena et al., 2019; Rodriguez et

al., 2024; Yang et al., 2024; Zhou et al., 2020), with striatum showing stronger encoding in well-trained animals (Yang et al., 2024; Zhou et al., 2020), consistent with it becoming the primary site of sensorimotor compression after automatization. This perspective suggests that time encoding is not the function of a single, dedicated clock, but an emergent property of circuits actively engaged in compressing information for the task at hand.

Although our work focuses on sensory bottlenecks, compression also occurs at the motor output of the basal ganglia, which receives convergent input from motor cortex and produces substantia nigra pars reticulata (SNr) output that is relatively low-dimensional compared to its cortical targets (Humphries, 2025). This "motor bottleneck" has been hypothesized to provide control signals that shift cortical states, a function thought to be critical for the learning of new, complex actions (Humphries, 2025). Lesion studies support this view by showing a stronger role for motor cortex in learning than in the execution of motor habits (Hikosaka et al., 2002; Kawai et al., 2015), and cortico-basal ganglia loops have long been emphasized in theories of complex action learning (Miller & Buschman, 2008). In our own modeling, we briefly explored the role of motor bottlenecks by modifying the LSTM (Figure S11), but found that a motor bottleneck alone was insufficient to reproduce the key effects. Importantly, our study is focused on the *execution* of well-learned behaviors, where we propose sensory bottlenecks play a dominant role. A crucial avenue for future research will be to investigate how sensory and motor bottlenecks interact during the learning process, and how neuromodulators such as dopamine—known to regulate both habit formation and the internal clock (Ambrosi & Lerner, 2022; De Corte et al., 2019)—coordinate these compression mechanisms across striatum and motor cortex.

Finally, while our investigation centered on cortico-striatal circuits for motor habits, sensory compression is a ubiquitous feature of neural architecture and may therefore apply broadly across the brain. For example, hippocampal place cells integrate highly multimodal sensory information from diverse sources (Jeffery, 2007) and also exhibit rich temporal coding relevant for episodic memory (Eichenbaum, 2014). Similarly, the cerebellum transforms converging, compressed sensory inputs into temporally precise neural dynamics that support time-sensitive behaviors such as conditioned responses and motor coordination (Medina et al., 2000; Wagner et al., 2019). These examples support our proposal that sensory compression may be a canonical computation, offering a unifying explanation for how diverse brain regions transform environmental structure into temporally precise neural codes and behaviors.

Methods

Algorithm and training procedure

The models were trained on tasks using the RL algorithm named Proximal Policy Optimization (PPO) (Schulman et al., 2017). PPO is a policy gradient algorithm. For traditional algorithms of the sort, the loss function from Eq 1 is optimized, where θ are the parameters of the ANN, the expectation is taken over the current policy π_θ, a_t is the action taken at state s_t, and $\hat{A}_t$ is the estimated advantage function $A_t(s_t, a_t) = Q_t(s_t, a_t) - V_t(s_t)$. The parameters of the ANN are updated using backpropagation through time.

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t \left[\log \pi_\theta \left(a_t | s_t \right) \hat{A}_t \right] \tag{1}$$

Multiple steps of gradient descent are performed over the same sampled trajectories, which can cause destructively large policy updates. PPO was devised as a simple approach to limit policy updates. Namely, the scale of the updates is directly clipped in the objective. The PPO objective is described in Eq 2, where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$.

$$L^{PPO}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta) \hat{A}_t, \text{clip} \left(r_t(\theta), 1 - \epsilon, 1 + \epsilon \right) \hat{A}_t \right) \right] \tag{2}$$

To unpack this objective, first consider the case where the advantage is positive (i.e. $r_t(\theta)$ must be made greater than 1). The original objective is clipped if the policy update is greater than $1 + \epsilon$. In the case where the advantage is negative (i.e. $r_t(\theta)$ must be made less than 1), the original objective is clipped if the policy update is smaller than $1 - \epsilon$.

The LSTM was initialized at the beginning of each training episode. We repeated each training run with three

different seeds to ensure that the results were robust to parameter initialization and environment stochasticity. Training was implemented using RLlib (Liang et al., 2018) The default training hyperparameters we used for the experiment are compiled in Table 1. The chunking task used these default hyperparameters, and the different hyperparameters that we used for the other tasks are mentioned in their relevant sections within Methods. Weight optimization used Adam (Kingma & Ba, 2014).

Table 1: **PPO training default hyperparameters.**

Hyperparameter	**Value**
Epochs	1500
Learning rate	1e-3
Learning rate schedule	5e-4 after 1200 epochs
Batch size	128
Discount factor γ	0.99

Action Chunking Task

We used a simple serial decision-making task inspired by Lai et al. (2025), and presented in Fig 1B. This task was specifically designed to highlight chunking. At each timestep $t \leq T$, a stimulus s_t is drawn from a fixed set S and a corresponding observation is presented to the agent. These stimuli were presented in the form of one-hot vectors. Each state is associated with a single rewarded action. The agent receives a penalty equal to $-\frac{1}{T}$ if an incorrect action is picked, 0 if it behaves correctly.

Importantly, the statistics of the task are tuned such that a subset $F \subset S$ of states appears more frequently in a pre-specified fixed order than not. The rationale is that capacity-limited ANNs should rely on chunking of this frequent sequence (i.e. learning the corresponding sequence of actions almost independently of the presented observation), and unconstrained networks should not. The task implementation is such that at each timestep that is not already part of the frequent sequence, there is a probability p_{init} of initiating the sequence. When the sequence is initiated, the first state is presented, and subsequent states are presented in following timesteps. If the sequence is not initiated, there is a probability p_{aux} that a state $s \in F$ is drawn independently, and conversely $1 - p_{\text{aux}}$ for a state $s \notin F$. The probabilities p_{init} and p_{aux} can be tuned to obtain a specific probability p_F of observing a state $s \in F$, and p_{chunk} of observing a state embedded in the sequence. Note that the sequence can be initiated only if it can be fully executed.

To reduce complexity of the task, the number of timesteps and frequent sequence size were chosen such that an episode can witness at most a single sequence. We relaxed this in supplementary experiments, however (Figure S2). The baseline values of the parameters of the task are given in Table 2. The simulation results that we present in this work are obtained after averaging over 100 randomly drawn episodes and over 9 training seeds, with error bars and significance being calculated over different training seeds.

Table 2: **Task baseline parameters.**

S	F	T	p_{init}	p_{aux}	p_{chunk}	p_F
$\{0, ..., 15\}$	$\{0, ..., 7\}$	8	0.45	0.09	0.45	0.5

Stimulus Comparison Task

To simulate the duration discrimination task described in Toso et al. (2021), we implemented a custom environment using the OpenAI Gym framework. Each trial consisted of two sequential stimuli ($S1$ and $S2$), separated by a silent delay period. The durations of $S1$ and $S2$ ($T1$ and $T2$ respectively) were randomly sampled integers within a predefined range (2-8 time steps), with the constraint that they were not equal. Stimuli were represented as sequences of 16-dimensional observation vectors. Each vector combined a 10-dimensional signal component—centered around a stimulus-specific intensity value ($I1$ and $I2$ respectively, chosen from 0.3, 0.6, or 0.9) and combined with a 6-dimensional Gaussian noise centered on 0.1. The signal and noise dimensions

were concatenated and randomly shuffled at each time step to prevent position-specific encoding. Note that the bottleneck dimension of the model added more noise on top of this.

The delay period consisted of zero-valued observations and lasted for a fixed number of steps (4 timesteps). At the end of each trial, the agent was required to choose whether S1 or S2 had the greater duration ($T1$ vs. $T2$) or greater intensity ($I1$ vs. $I2$), depending on the task mode). The environment included a discrete action space with two options: selecting S1 or S2. A reward of 1.0 at the end was given for correct responses, and 0.0 otherwise.

Default hyperpameters were used as in Table 1, except we trained only on 500 epochs due to the task being significantly simpler and used a default noise parameter for the bottleneck layer of 0.6. We modified the learning rate to start at $5e-4$ and linearly decay to $1e-4$.

Treadmill Task

To get our model to reproduce the FBF strategy in Hidalgo-Balbuena et al. (2019), we constructed a custom reinforcement learning environment in which an agent navigates along a one-dimensional track while being continuously displaced by a virtual treadmill moving backward at a fixed velocity of 0.6 units per timestep. The agent begins near the front of the treadmill at position $x = 85.0$ and must reach a goal zone $x \geq 90.0$ at a specific target time TT, which is randomly sampled from a uniform distribution between 8 and 23 time steps on each trial. At each timestep, the agent selects from three discrete actions, each associated with a different self-generated velocity: **hold** results in no self-motion, and the agent moves backward with the treadmill at -0.6; **jog** results in the agent passively resisting the treadmill and slowing down the effect of the treadmill to -0.3, and **sprint** results in the agent exerting maximum effort to counteract the treadmill, producing a net forward motion of $v = +1.7$.

The agent receives an observation vector consisting of its current position, a normalized countdown to the target time that is revealed only during the jog action, one-hot encoding of the previous action, and the previous reward. Following (Wang et al., 2018), which modeled PFC-DMS circuits using explicit temporal context for meta-learning, observations included previous action and reward to enable adaptation across varying arrival times for model. Importantly, since this observation is effectively 4-dimensional, we used 2-dimensional bottleneck models in this task to force a compression of this observation like in our other two tasks. This gating of temporal information is designed to mirror the insight of Hidalgo-Balbuena et al. (2019) that the rats use the consistent jogging phase of the FBF strategy (Figure 7A, number 3) to gauge how much time they have. By providing a temporal estimate gated to a specific mode of movement, we can reproduce a similar strategy in the models.

The reward function strongly favors reaching the goal zone precisely at time T; if the agent arrives after the target time, it receives a positive reward that decays linearly with temporal error. If the agent arrives early, it receives a fixed negative reward, thus establishing a clear preference for being slightly late over being early (just as in the original task in Hidalgo-Balbuena et al. (2019)). To facilitate learning FBF, we introduced small, one-time shaping rewards that encourage appropriate motor transitions: one for an early hold action, one for a jog action that follows a hold action, and one for a sprint action after the jog action.

Each of these shaping rewards is only granted once per episode and is small in magnitude (≤ 0.15) compared to the maximum terminal reward of 1.0. We used the same default hyperparameters as in Table 1, but modified the learning rate to start at $5e-4$ and linearly decay to $1e-4$. We also reduced the default bottleneck dimension noise to 0.1 due to the added temporal complexity of the task.

Statistics on Neural Trajectories

To assess the neural trajectories of the model's recurrent layer, we computed time encoding and smoothness metrics from the hidden states recorded during task execution. Time encoding was quantified via a time decoding analysis. Specifically, we trained a linear Ridge Regression model to predict the timestep within a trial from the hidden state. We employed a "Leave One Trial Out" procedure where we trained on a set of 99 training trials, evaluated on a held-out test trial, and repeated this process until all trials have been a test trial. We gridsearched over possible regularization parameter values using a nested training loop within the training set. The measure of decoding performance was the coefficient of determination R^2. High R^2 values indicate that time is linearly decodable from the hidden state, demonstrating that the neural trajectories encode temporal information.

Smoothness was evaluated by computing two geometric properties of the hidden state trajectories: acceleration and curvature. For each trial, we extracted the full trajectory of hidden states and computed both the mean acceleration magnitude and mean curvature. Acceleration was estimated by taking the second finite derivative of the hidden state vectors over time, and its magnitude was computed as the Euclidean norm of these acceleration vectors. Curvature was estimated as the average angular deviation between successive velocity vectors, calculated using the arccosine of their normalized dot product. Lower values of both metrics indicate smoother trajectories in state space.

We also quantified trial-to-trial variability to assess the consistency of neural trajectories. For each timestep, we computed the mean squared deviation of individual trial activations from the average trajectory across trials. When trials had different lengths, we grouped trials of the same length and computed variability within each group before averaging across groups weighted by the number of trials. Lower variability indicates more stereotyped and consistent neural dynamics across repeated task executions.

For each experimental condition and architecture, we reported the mean time encoding R^2, mean acceleration, mean curvature, and mean trial-to-trial variability across trials. These metrics jointly characterize the representational geometry of the RNN, with high time encoding, low acceleration, low curvature, and low variability suggesting stable, smooth, and consistent trajectories that implicitly encode time.

Supporting information

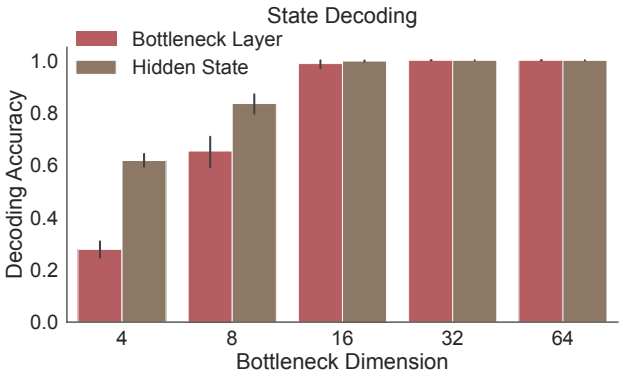

Fig. S1. Decoding of Stimulus from Bottleneck Layer and RNN Hidden State Classification accuracy when decoding state in the chunking task from the bottleneck layer and hidden state of the model. We used Logistic Regression, where the regularization term was optimized in a nested training loop via grid search. We performed 3-fold cross validation, where in each fold, the classifier was evaluated on trials that are held-out from training. Errorbars denote 95% confidence intervals over training seeds. Models with stronger bottlenecks will contain less information about the current timestep's stimulus.

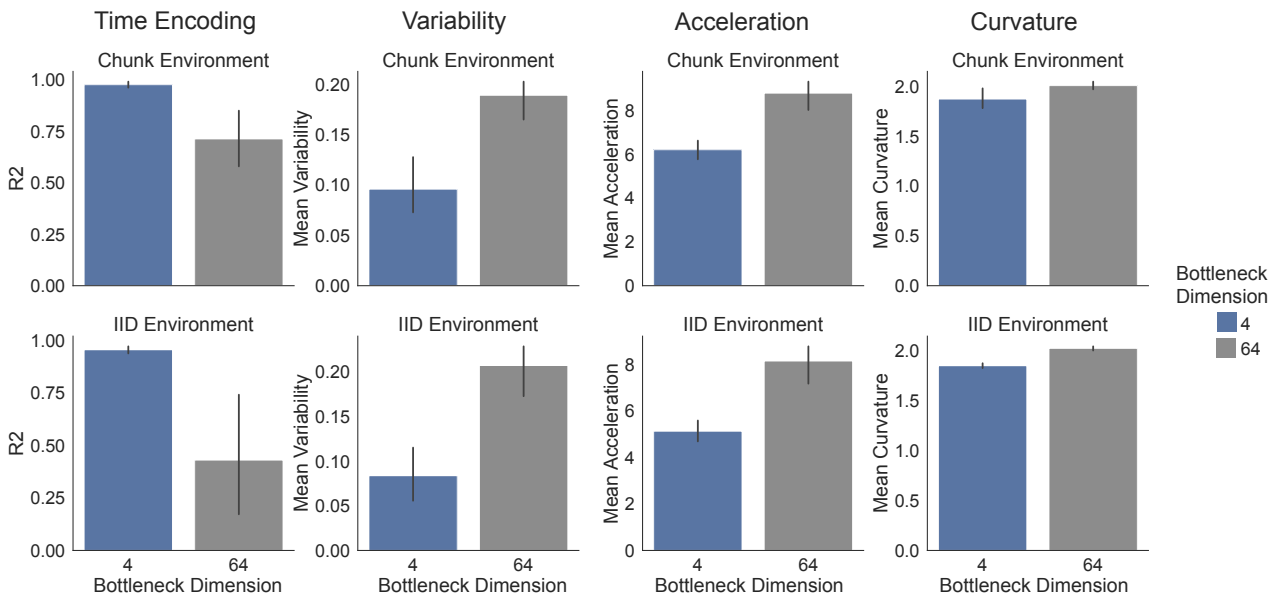

Fig. S2. Reproduction of chunking task results with variable trial lengths We trained a version of the model on chunking task where trial lengths varied randomly from 8 to 12. In this case, the chunk can appear anytime from $t = 0$ to $t = 3$, so timestep and index within the chunk are not totally aligned. Even in this case, bottleneck models exhibit strong time encoding and hidden state trajectories that are stable and smooth.

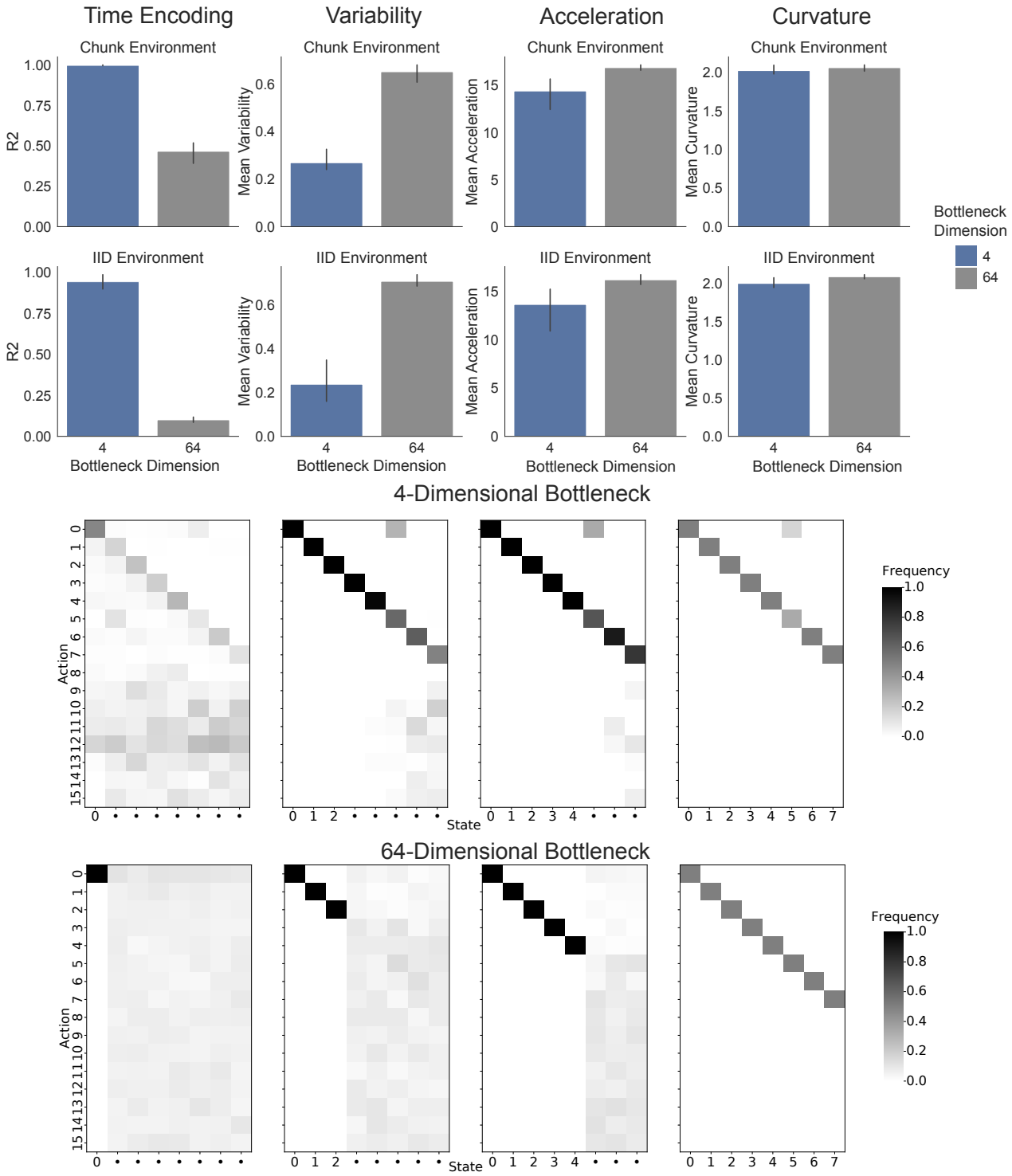

Fig. S3. Reproduction of chunking task results with vanilla RNNs We trained a version of the model replacing the LSTM layer with a vanilla RNN and show that a vanilla RNN with a sensory bottleneck also (**A**). produces stable, smoooth, and time-encoding neural trajectories and (**B**). employs chunking.

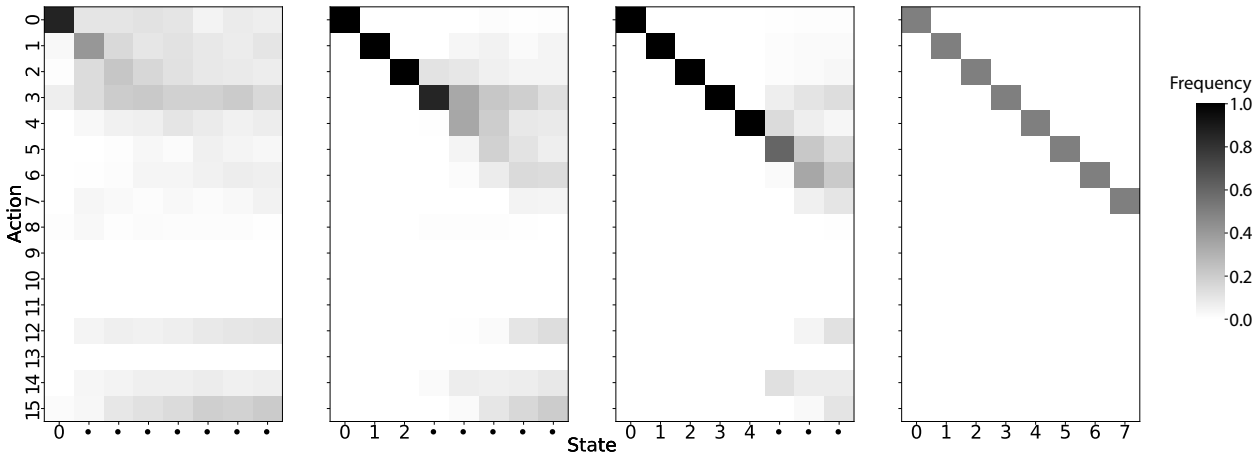

Fig. S4. No Chunking for a Non-Bottleneck Model Trained Only on Structured Trials We trained a baseline non-bottleneck model *only* on structured trials (so every single trial in its training experience were the same stimuli in the same order) and conducted the same action slipping analysis in Figure 3C to show this model still does not employ chunking. This gives further evidence that the sensory bottleneck is necessary to produce chunking.

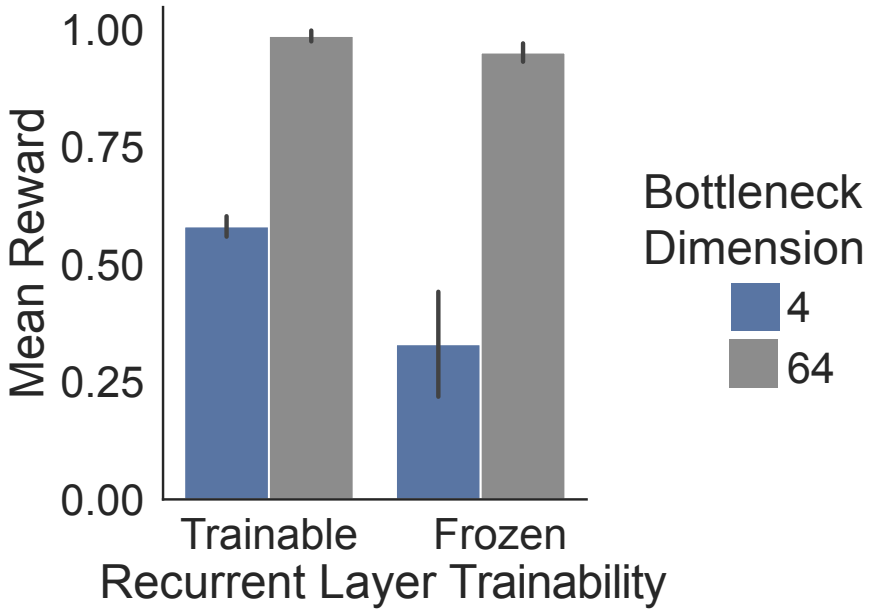

Fig. S5. Results from Freezing Recurrent Weights in the Chunking Task We show mean performance across bottleneck vs. non-bottleneck models when the recurrent weights are frozen. The bottleneck model's performance is greatly affected, while the non-bottleneck model's performance is not.

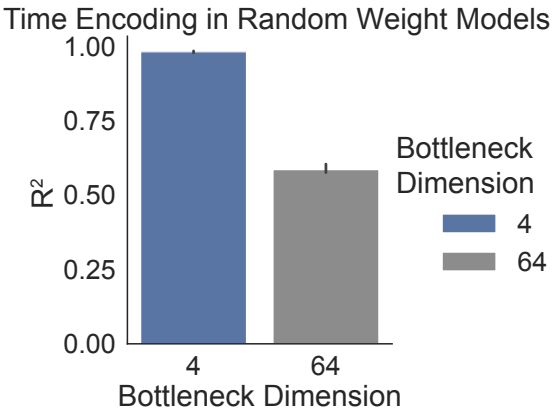

Fig. S6. Random-Weight Models with Sensory Bottlenecks Exhibit Time Encoding. We compare untrained models with and without sensory bottlenecks. For the non-bottleneck model, the sensory encoder has no compression, so the sensory input's stays at 64-dimensions. For the bottleneck model, the sensory encoder does a random compression down to 4-dimensions before passing it to the recurrent layer. For both models, the recurrent layer is 64-dimensional. The weights were initialized from a uniform distribution $\mathcal{U}(-\sqrt{k}, \sqrt{k})$, where $k = \frac{1}{hidden_size} = \frac{1}{64} = 0.125$. Since the input is low-dimensional in the bottleneck model, the hidden state's variance is dominated by the recurrent layer's internal dynamics, which naturally unfold stably with time. These results show that sensory compression intrinsically generates time-encoding dynamics, independent of task-specific learning (see main text for details).

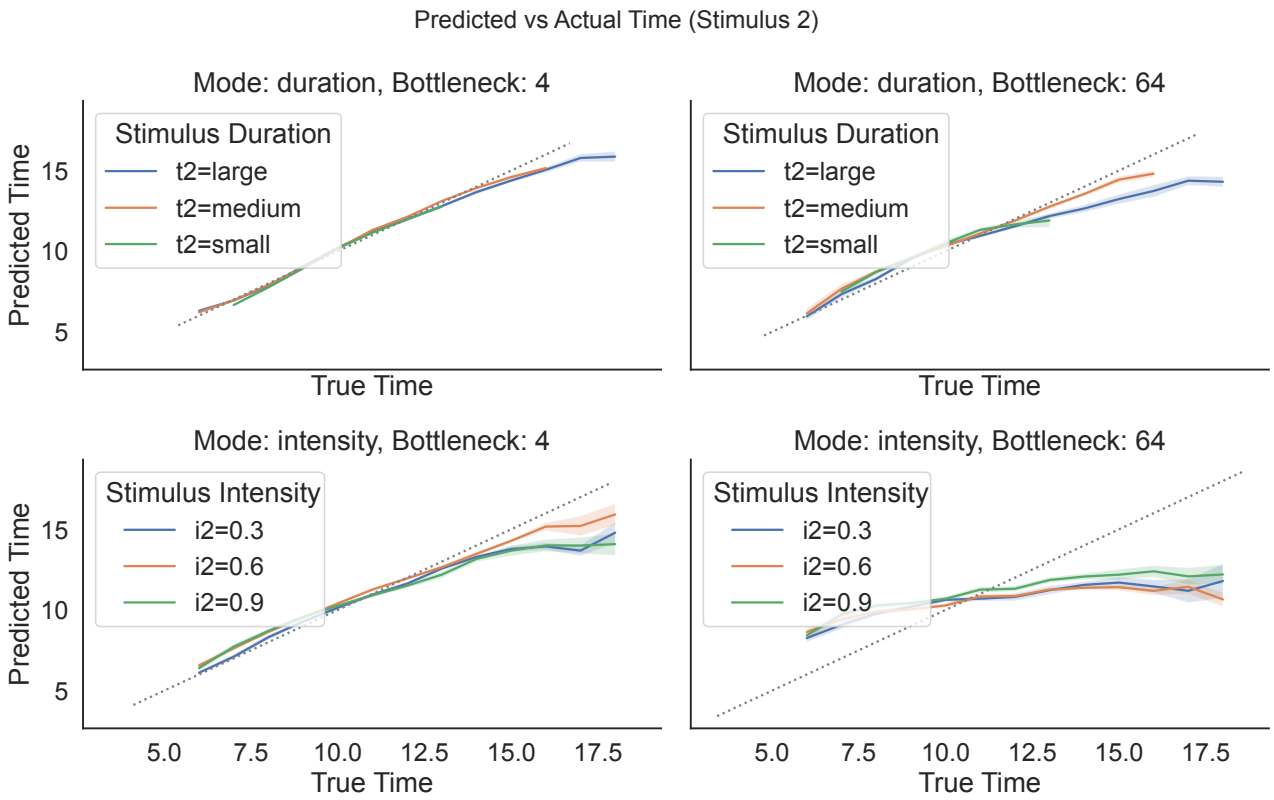

Fig. S7. Time encoding in stimulus comparison task during second stimulus Same as Figure 6C, but for the second stimulus in the trial. The bottleneck model has stronger time coding. Although one can see a small effect of stimulus intensity on time decoding in the bottleneck model towards the end of the stimulus (bottom left, where lines of different colors start diverging), it is important to note that this effect was also seen in DLS by Toso et al. (2021), as detailed in their author correction.

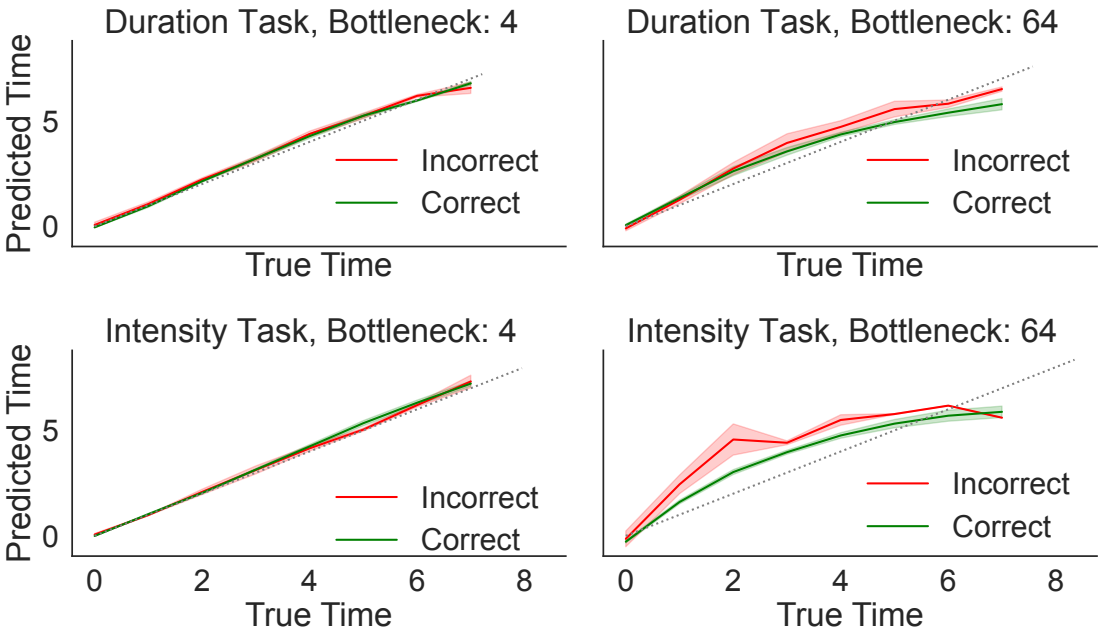

Fig. S8. Difference in time encoding between correct and incorrect trials during Stimulus 1 Toso et al. (2021) found time encoding in Rat DLS does not differ across correct and incorrect trials (Figure 5D in their paper). We replicate this analysis here and show bottleneck models have this property, but non-bottleneck models do not.

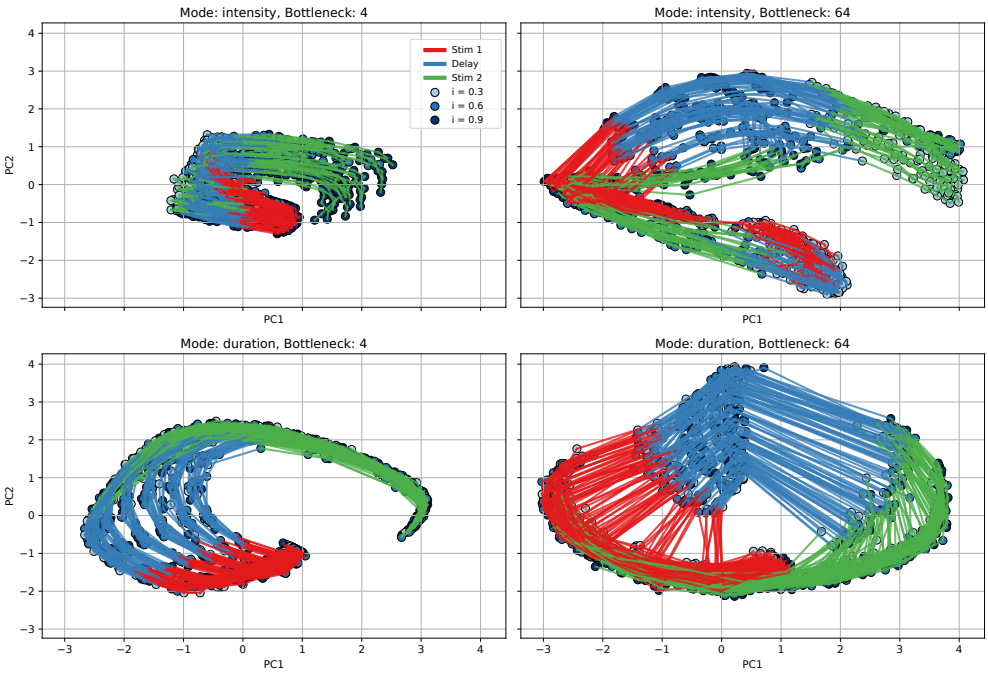

Fig. S9. PCA trajectories of hidden state during stimulus comparison task Top row is for networks trained to do intensity comparison, bottom for duration comparison. Left column is for bottleneck model and right is for non-bottleneck baseline. Color between points represent stage of the task (stimulus 1, delay, stimulus 2) and colors for points themselves represent stimulus intensity. 100 trajectories are shown in each plot. In bottleneck models, hidden states from different timepoints are consistently more clearly seperated (indicated by a greater lack of overlap in lines with different colors than the non-bottleneck model trajectories).

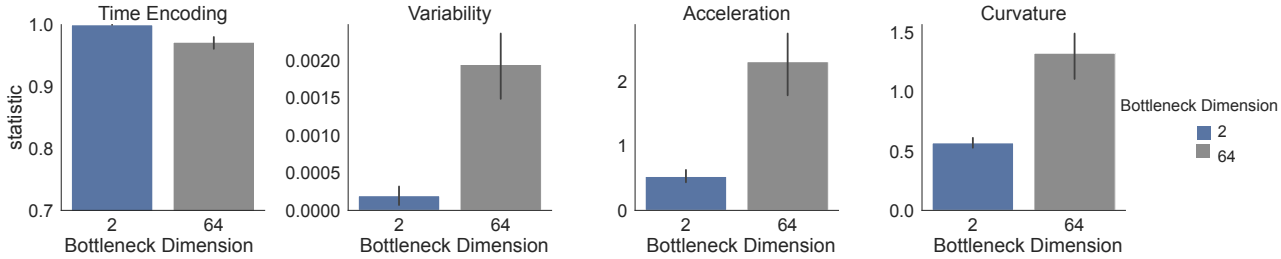

Fig. S10. Neural Trajectory Statistics for Treadmill Task Same neural trajectory statistics we show in Figure 5 for the treadmill task. We see stronger time encoding along with stable and smooth trajectories in the sensory bottleneck models.

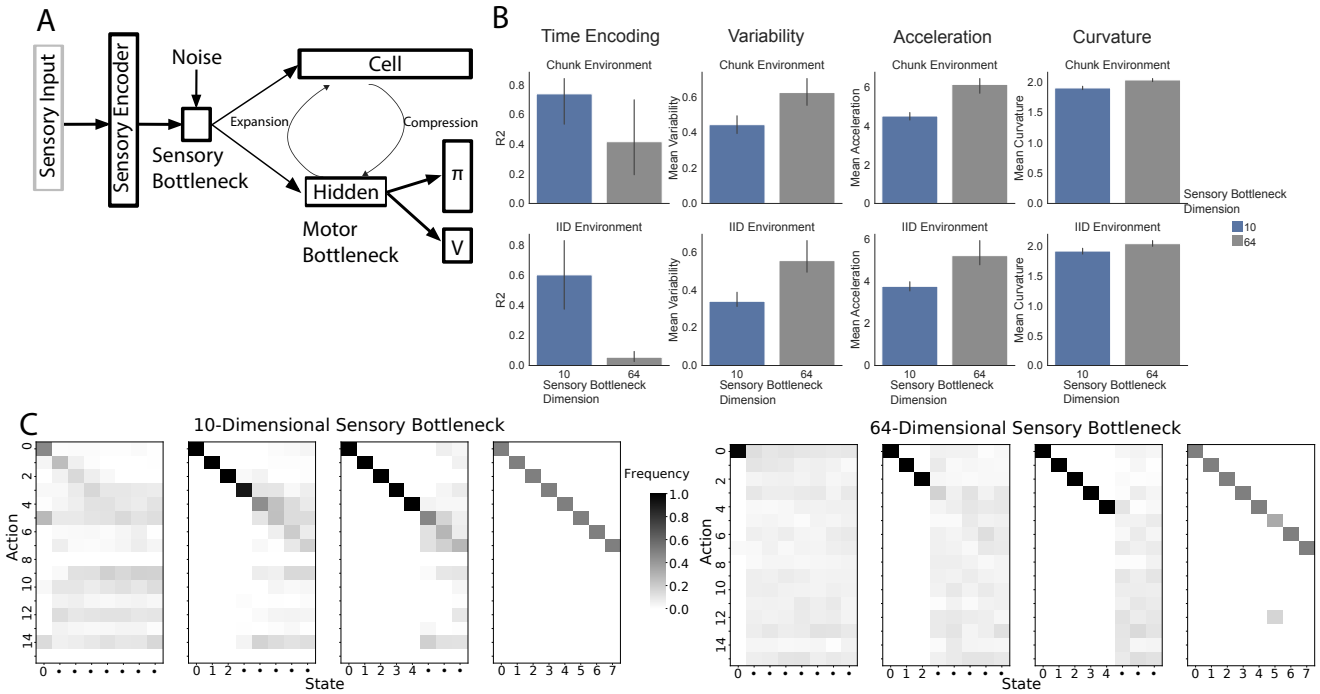

Fig. S11. Modified architecture with motor bottleneck (**A**). Along with a sensory bottleneck from sensory regions, the DLS also has a bottleneck in its projections to and from motor cortex. To investigate the effect of this bottleneck, we created a version of our model in which the LSTM cell state (64 dimensions) is higher-dimensional than the LSTM hidden state (10 dimensions). We used 10 dimensions instead of 4 dimensions like in the main text, because we found a 4-dimensional motor bottleneck severely limits the network's ability to learn the task regardless of the sensory bottleneck dimension. The compression from cell state to hidden state and expansion from hidden state to cell state can be thought of as analogous to the loop interactions between motor cortex and DLS. (**B**). Results from models with a motor bottleneck on the chunking task. Blue indicates a model with *both* a sensory bottleneck (10 dimensions) and a motor bottleneck and gray indicates a model with *only* a motor bottleneck and not a sensory bottleneck. Inducing a sensory bottleneck produces time-encoding stable and smooth neural trajectories like the ones we see in Figure 5, whereas inducing only a motor bottleneck does not produce neural trajectories different from the non-bottleneck baseline model in Figure 5. (**C**). Action slipping analysis from Figure 3C on motor bottleneck models, indicating that only the model with a sensory bottleneck does chunking and the model with only a motor bottleneck does not do chunking.

Acknowledgements

S.K was previously supported by a Google PhD Fellowship and currently supported by a Leon Levy Fellowship by the New York Academy of Sciences. J.H is supported by VA CSR&D award 1I01 CX002872.

References

Ambrosi, P., & Lerner, T. N. (2022). Striatonigrostriatal circuit architecture for disinhibition of dopamine signaling. *Cell reports*, *40*(7).

De Corte, B. J., Wagner, L. M., Matell, M. S., & Narayanan, N. S. (2019). Striatal dopamine and the temporal control of behavior. *Behavioural brain research*, *356*, 375–379.

Dezfouli, A., & Balleine, B. W. (2012). Habits, action sequences and reinforcement learning. *European Journal of Neuroscience*, *35*(7), 1036–1051.

Dickinson, A. (1985). Actions and habits: the development of behavioural autonomy. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, *308*(1135), 67–78.

Eichenbaum, H. (2014). Time cells in the hippocampus: a new dimension for mapping memories. *Nature Reviews Neuroscience*, *15*(11), 732–744.

Gershman, S. J. (2020). Origin of perseveration in the trade-off between reward and complexity. *Cognition*, *204*, 104394.

Graybiel, A. M. (1998). The basal ganglia and chunking of action repertoires. *Neurobiology of learning and memory*, *70*(1-2), 119–136.

Grillner, S. (2025). How circuits for habits are formed within the basal ganglia. *Proceedings of the National Academy of Sciences*, *122*(13), e2423068122.

Hidalgo-Balbuena, A. E., Luma, A. Y., Pimentel-Farfan, A. K., Peña-Rangel, T., & Rueda-Orozco, P. E. (2019). Sensory representations in the striatum provide a temporal reference for learning and executing motor habits. *Nature communications*, *10*(1), 4074.

Hikosaka, O., Nakamura, K., Sakai, K., & Nakahara, H. (2002). Central mechanisms of motor skill learning. *Current opinion in neurobiology*, *12*(2), 217–222.

Hintiryan, H., Foster, N. N., Bowman, I., Bay, M., Song, M. Y., Gou, L., … others (2016). The mouse cortico-striatal projectome. *Nature neuroscience*, *19*(8), 1100–1114.

Humphries, M. D. (2025). The computational bottleneck of basal ganglia output (and what to do about it). *eneuro*, *12*(4).

Igl, M., Ciosek, K., Li, Y., Tschiatschek, S., Zhang, C., Devlin, S., & Hofmann, K. (2019). Generalization in reinforcement learning with selective noise injection and information bottleneck. *Advances in neural information processing systems*, *32*.

Jeffery, K. J. (2007). Integration of the sensory inputs to place cells: what, where, why, and how? *Hippocampus*, *17*(9), 775–785.

Kawai, R., Markman, T., Poddar, R., Ko, R., Fantana, A. L., Dhawale, A. K., … Ölveczky, B. P. (2015). Motor cortex is required for learning but not for executing a motor skill. *Neuron*, *86*(3), 800–812.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lai, L., & Gershman, S. J. (2021). Policy compression: An information bottleneck in action selection. In *Psychology of learning and motivation* (Vol. 74, pp. 195–232). Elsevier.

Lai, L., Huang, A. Z., & Gershman, S. J. (2025). Action chunking as conditional policy compression. *Cognition*, *264*, 106201.

Lerch, R., & Sims, C. (2019). Rate-distortion theory and computationally rational reinforcement learning. *Proceedings of Reinforcement Learning and Decision Making (RLDM) 2019*, 7–10.

Liang, E., Liaw, R., Nishihara, R., Moritz, P., Fox, R., Goldberg, K., … Stoica, I. (2018). Rllib: Abstractions for distributed reinforcement learning. In *International conference on machine learning* (pp. 3053–3062).

Liu, S., Kikumoto, A., Badre, D., & Gershman, S. J. (2025). Neural and behavioral signatures of policy compression in cognitive control. *bioRxiv*, 2025–05.

Medina, J. F., Garcia, K. S., Nores, W. L., Taylor, N. M., & Mauk, M. D. (2000). Timing mechanisms in the cerebellum: testing predictions of a large-scale computer simulation. *Journal of Neuroscience*, *20*(14), 5516–5525.

Miller, E. K., & Buschman, T. J. (2008). Rules through recursion: how interactions between the frontal cortex and basal ganglia may build abstract, complex rules from concrete, simple ones. *Neuroscience of rule-guided behavior*, 419–440.

Mizes, K. G., Lindsey, J., Escola, G. S., & Ölveczky, B. P. (2024). The role of motor cortex in motor sequence execution depends on demands for flexibility. *Nature neuroscience*, *27*(12), 2466–2475.

Morris, G., Nevet, A., & Bergman, H. (2003). Anatomical funneling, sparse connectivity and redundancy reduction in the neural networks of the basal ganglia. *Journal of Physiology-Paris*, *97*(4-6), 581–589.

Norman, D. A. (1981). Categorization of action slips. *Psychological review*, *88*(1), 1.

Paxinos, G., & Franklin, K. B. (2019). *Paxinos and franklin's the mouse brain in stereotaxic coordinates*. Academic press.

Petrof, I., Viaene, A. N., & Sherman, S. M. (2015). Properties of the primary somatosensory cortex projection to the primary motor cortex in the mouse. *Journal of neurophysiology*, *113*(7), 2400–2407.

Rodriguez, A. C., Perich, M. G., Miller, L. E., & Humphries, M. D. (2024). Motor cortex latent dynamics encode spatial and temporal arm movement parameters independently. *Journal of Neuroscience*, *44*(35).

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Tishby, N., Pereira, F. C., & Bialek, W. (2000). The information bottleneck method. *arXiv preprint physics/0004057*.

Toso, A., Reinartz, S., Pulecchi, F., & Diamond, M. E. (2021). Time coding in rat dorsolateral striatum. *Neuron*, *109*(22), 3663–3673.

Tsao, A., Yousefzadeh, S. A., Meck, W. H., Moser, M.-B., & Moser, E. I. (2022). The neural bases for timing of durations. *Nature reviews neuroscience*, *23*(11), 646–665.

Tunes, G. C., de Oliveira, E. F., Vieira, E. U., Caetano, M. S., Cravo, A. M., & Reyes, M. B. (2022). Time encoding migrates from prefrontal cortex to dorsal striatum during learning of a self-timed response duration task. *Elife*, *11*, e65495.

Wagner, M. J., Kim, T. H., Kadmon, J., Nguyen, N. D., Ganguli, S., Schnitzer, M. J., & Luo, L. (2019). Shared cortex-cerebellum dynamics in the execution and learning of a motor task. *Cell*, *177*(3), 669–682.

Wang, J. X., Kurth-Nelson, Z., Kumaran, D., Tirumala, D., Soyer, H., Leibo, J. Z., … Botvinick, M. (2018). Prefrontal cortex as a meta-reinforcement learning system. *Nature neuroscience*, *21*(6), 860–868.

Yang, Z., Inagaki, M., Gerfen, C., Fontolan, L., & Inagaki, H. K. (2024). The frontal cortex adjusts striatal integrator dynamics for flexible motor timing. *bioRxiv*, 2024–06.

Zareian, B., Lam, A., & Zagha, E. (2023). Dorsolateral striatum is a bottleneck for responding to task-relevant stimuli in a learned whisker detection task in mice. *Journal of Neuroscience*, *43*(12), 2126–2139.

Zhou, S., Masmanidis, S. C., & Buonomano, D. V. (2020). Neural sequences as an optimal dynamical regime for the readout of time. *Neuron*, *108*(4), 651–658.