Flexible mapping of abstract domains by grid cells via self-supervised extraction and projection of generalized velocity signals

Abhiram Iyer[1, 3, 4], **Sarthak Chandra**[2, 3], **Sugandha Sharma**[2, 3], and **Ila Fiete**[2, 3, 4]

[1]Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA
[2]Department of Brain and Cognitive Sciences, MIT, Cambridge, MA
[3]McGovern Institute for Brain Research, MIT, Cambridge, MA
[4]K. Lisa Yang Integrative Computational Neuroscience (ICoN), MIT, Cambridge, MA ,
{abiyer,sarthakc,susharma,fiete}@mit.edu

Abstract

Grid cells in the medial entorhinal cortex create remarkable periodic maps of explored space during navigation. Recent studies show that they form similar maps of abstract cognitive spaces. Examples of such abstract environments include auditory tone sequences in which the pitch is continuously varied or images in which abstract features are continuously deformed (e.g., a cartoon bird whose legs stretch and shrink). Here, we hypothesize that the brain generalizes how it maps spatial domains to mapping abstract spaces. To sidestep the computational cost of learning representations for each high-dimensional sensory input, the brain extracts self-consistent, low-dimensional descriptions of displacements across abstract spaces, leveraging the spatial velocity integration of grid cells to efficiently build maps of different domains. Our neural network model for abstract velocity extraction factorizes the content of these abstract domains from displacements within the domains to generate content-independent and self-consistent, low-dimensional velocity estimates. Crucially, it uses a self-supervised geometric consistency constraint that requires displacements along closed loop trajectories to sum to zero, an integration that is itself performed by the downstream grid cell circuit over learning. This process results in high fidelity estimates of velocities and allowed transitions in abstract domains, a crucial prerequisite for efficient map generation in these high-dimensional environments. We also show how our method outperforms traditional dimensionality reduction and deep-learning based motion extraction networks on the same set of tasks. This is the first neural network model to explain how grid cells can flexibly represent different abstract spaces and makes the novel prediction that they should do so while maintaining their population correlation and manifold structure across domains. Fundamentally, our model sheds light on the mechanistic origins of cognitive flexibility and transfer of representations across vastly different domains in brains, providing a potential self-supervised learning (SSL) framework for leveraging similar ideas in transfer learning and data-efficient generalization in machine learning and robotics.

1 Introduction

Grid cells in the medial entorhinal cortex are of paramount importance for navigating and representing spatial domains. Interestingly, a series of recent experiments have shown that the brain still uses the same cells to represent non-spatial environments that are continuously traversed. These include free

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

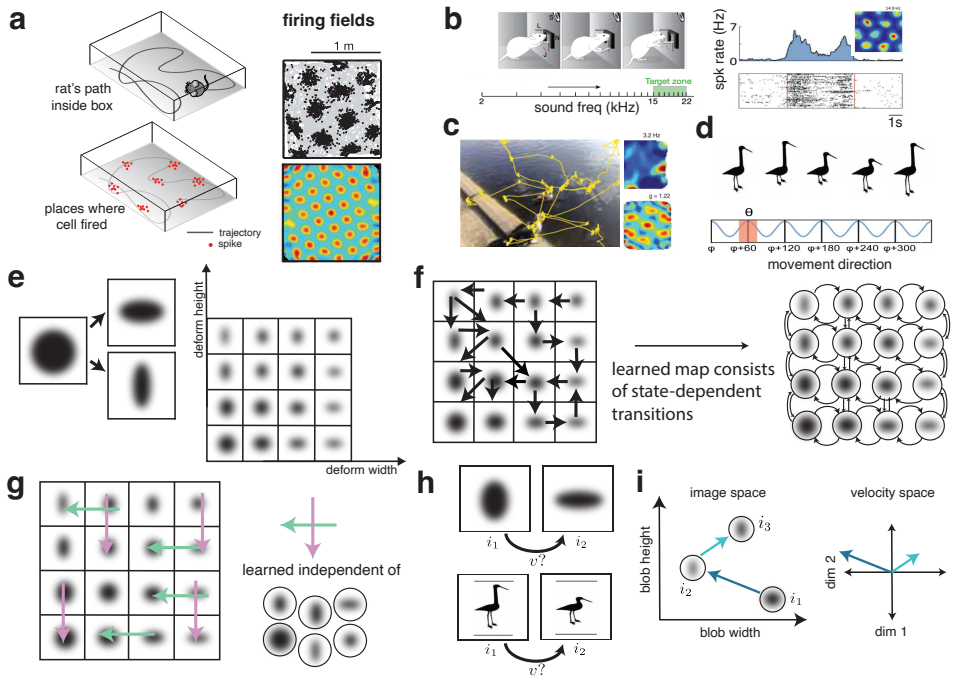

Figure 1: **Conceptual understanding of learning velocities in abstract cognitive environments.** **a.** Grid cells generate hexagonal tuning in spatial navigation experiments (illustration borrowed from [1]). **b.** Similar grid-like tuning has been found in auditory frequency space in rodents [2], (**c.**) in visual space in monkeys [3], and (**d.**) in an abstract cartoon bird space in human experiments. **e.** Example of an abstract, non-spatial domain called 'Stretchy Blob', a 2D Gaussian that can either stretch or shrink along two axes. **f.** SR, CSCG, and TEM [4–7] learn transition structures of this cognitive domain by traversing it. These models require one to build a set of representations for these encountered states as well as structures enabling transitions between them. **g.** Our approach learns a self-consistent movement (velocity) signal that is independent of the states traversed and encodes the global transition structure of the environment in a minimally low-dimensional representation. **h.** How can the brain initially extract a general notion of velocity within abstract, non-spatial domains? Solving this important prerequisite challenge can show how grid cells flexibly map and organize various abstract spaces. **i.** The actual movement signals in this particular example are low-dimensional. These velocities are independent from the states they connect.

viewing of naturalistic images, listening to and modifying a sound changing in pitch, navigating a conceptual 'Stretchy Bird' space (images of a bird that stretch or shrink via joystick input), or even traversing an 'odor' space among many others [8–14, 3, 15, 2] (Fig. 1a-d).

How does the brain transfer its ability to represent space to these non-spatial environments? More specifically, how does the brain infer a metric layout in these abstract domains? Answering this question involves understanding how the brain perceives and processes structure in different domains and modalities, and integrates them into coherent cognitive maps.

How the brain maps abstract cognitive environments has been a question of intense interest, with several proposed approaches [4, 5, 7, 6]. These works propose that when the brain learns the transition structure of a cognitive domain (such as the 'Stretchy Blob' space in Fig. 1e), it simultaneously builds a set of representations for these states as well as structures enabling transitions between them, Fig. 1f.

Our approach investigates how the brain exploits an existing scaffold of structured states, provided by grid cells, to represent new cognitive domains by projecting them onto the invariant grid coding space. We hypothesize that this projection involves anchoring a state in the cognitive domain to a grid phase and then extracting a self-consistent movement signal to measure displacements in the cognitive domain (Fig. 1g). This abstract velocity signal serves as the input to the grid cell network, updating grid phases and encoding transitions between high-dimensional states using the existing

low-dimensional transition mechanism within the grid cell circuit. Crucially, rather than building a *new set of continuous stable states* for each explored cognitive domain — a process that is both difficult and slow — this mechanism allows for the *efficient reuse of a canonical scaffold of cognitive states* for the memory and coding of continuous variables.

In spatial domains, continuous attractor models of the grid cell circuit [16] efficiently reuse prestructured grid cell states to encode transitions between high-dimensional states. However, using such models to map non-spatial, abstract domains (and by extension, explaining how grid cells can be efficiently reused) requires the extraction of faithful representations of velocity in these domains. Thus, the fundamental, prerequisite challenge lies in extracting a general notion of a minimal dimensional velocity signal from high-dimensional, time-varying data from various abstract domains, Fig. 1h. Crucially, the representations of this extracted velocity must be independent of the specific states, and must be self-consistent with a net-zero velocity corresponding to a net identity transformation in the abstract space. These constraints point towards a self-supervised learning paradigm, wherein neither the coordinate system of the space nor the dimensionality of the estimated velocity are known a priori.

The following are the key contributions of the paper:

- **First neural network model for abstract velocity extraction.** We present the first neural network model explaining how grid cells flexibly represent different abstract spaces through providing a learning framework for velocity estimation in arbitrary spaces. This reuse of grid cells across domains leads to cognitive flexibility in mapping spaces, providing insights towards transfer learning and data-efficient generalization in machine learning and robotics.
- **State-independent velocity extraction.** Our framework for abstract velocity extraction generates state-independent, self-consistent low-dimensional velocity estimates by separating the content of abstract domains from their displacements (Fig. 1g,i).
- **Self-supervised geometric consistency constraint.** We show how consistent metric representations of velocity can be learned through a self-supervised geometric consistency constraint requiring displacements along closed loop trajectories to sum to zero — an operation facilitated by the downstream grid cell circuit.
- **Superior dimensionality reduction performance.** Our method surpasses traditional dimensionality reduction and deep learning-based motion extraction networks, specifically in environments characterized by underlying low-dimensional transitions and motions between states.
- **Preservation of cell-cell relationships.** The model predicts that cell-cell relationships between grid cells are preserved *across* different spatial and non-spatial domains. For example, if two grid cells are co-active in a spatial task, they should remain co-active in a non-spatial task. This prediction critically relies on the same continuous attractor-based grid module performing integration across domains — a capability that can only be realized through extraction of a state-independent velocity from abstract domains. This prediction aligns with the observed invariance of internal neural representations across different spatial environments and brain states [17–23]. This forms a testable hypothesis for future experimental studies on neural representations in abstract cognitive domains.

1.1 Related work

Our work on learning velocities and transition operators in abstract cognitive spaces can be compared to two key areas: neuroscience models of spatial mapping and dimensionality reduction models for high-dimensional data. We first discuss cognitive space mapping models and then relate them to dimensionality reduction work.

The Tolman-Eichenbaum Machine (TEM) [7] learns a map of a 'cognitive domain' via a recurrent neural network, predicting sensory observations based on given actions. However, TEM does not infer transition structures or affordances, relying instead on predefined actions at each time step. When presented with a new environment, TEM requires re-learning these affordances along with grid cell-like representations from scratch. Similarly, the successor representation (SR) [4, 5] predicts future states as a weighted sum of expected future occurrences and uses an underlying discrete action space. State-action variants of SR are dependent on these discrete action inputs to construct

cognitive maps. Clone-structured cognitive graphs (CSCG) [6] learn hidden Markov models of sensory representations from observational inputs without prior domain structure assumptions but are limited to only discrete domains. By construction, these models are not guaranteed to preserve cell-cell correlations across domains and modalities and are unable to efficiently reuse the prestructured states and transition operators provided by grid cells.

More generally, these models posit that the brain simultaneously learns representations of the external states *and* the transition structures to form cognitive maps. Here, we propose that approaching the cognitive mapping problem from a velocity-extraction-first perspective offers key benefits. By learning a low-dimensional velocity that captures transitions between external inputs, mapping the environment to reusable grid cell states can be performed simply by a continuous attractor network, instead of something more complex like TEM. More generally, via our self-supervised learning framework that infers velocity solely from sensory inputs in continuous domains, we can entirely avoid the computationally expensive task of representation and transition learning through capturing the minimal low-dimensional structure of these abstract observations.

Our work thus bridges the tasks of building cognitive maps of abstract domains and dimensionality reduction. Previous dimensionality reduction methods [24–29] focus on preserving proximity in high-dimensional spaces on low-dimensional manifolds but do not take advantage of the structure imposed on the tangent spaces of these manifold (i.e., velocity spaces) through transitions between states. As a result, while these methods learn low-dimensional mappings, they often fail to reproduce the underlying metric space of abstract domains. Unlike these methods, our framework learns a global velocity operator applicable to any input within the environment's state manifold. Our work also relates to the task of motion decomposition and next-frame prediction of temporally varying inputs [30–35]. However, these works have not attempted to decipher a minimally low dimensional description of velocity, making their learned representations unsuitable for grid cells to effectively map observed environments.

2 Self-supervision for velocity extraction

Learning action primitives purely through self-supervision, without access to true velocities in the space, requires careful consideration of the data generation process, the loss terms to extract meaningful neural representations, and the chosen neural network architecture. We will discuss each of these aspects in detail.

2.1 Task setup

The initial step in our methodology involves the construction of tasks designed to facilitate the model's inference of the lowest-dimensional representations of velocity within cognitive domains.

We revisit the 'Stretchy Blob' environment that we previously introduced. This environment can be procedurally generated where state transitions are characterized by changes in the blob's width and height. As seen in Fig. 2a, given two images, i_1 and i_2, we aim to learn a function f that infers a low-dimensional velocity $\hat{v}$ representing the transition from i_1 to i_2 (Fig. 2b).

To ensure that $\hat{v}$ is a useful and faithful metric representation of the transition between two inputs, we demand that it satisfy two properties.

First, the estimated transition velocity $\hat{v}$ can be used to transform a given input, i.e., there exists a function g that uses $\hat{v}$ to transform i_1 to predict i_2. In practice, to ensure that f does not merely memorize image features of i_2, we demand that g predict an unseen i_3 obtained by transforming i_2 using the same transition $\hat{v}$. More generally, this ensures that the estimates of displacements, $\hat{v}$, in this space must be independent of the route taken, ensuring both path and state independence. We refer to this requirement as 'next-state prediction'.

Second, the estimated velocities in the abstract space, $\hat{v}$, must be geometrically self-consistent, i.e., the start and end point of a trajectory in the abstract space are identical if, and only if, the sum of estimated velocities is zero (Fig. 2c). (Note that this also ensures that the identity transformation should be represented by the zero-vector.) We refer to this requirement as 'loop-closure'. This summation of velocities must be performed by a neural integrator, such as grid cells, which results in a calibrated velocity input for grid cells to then themselves use to map out the input abstract domain.

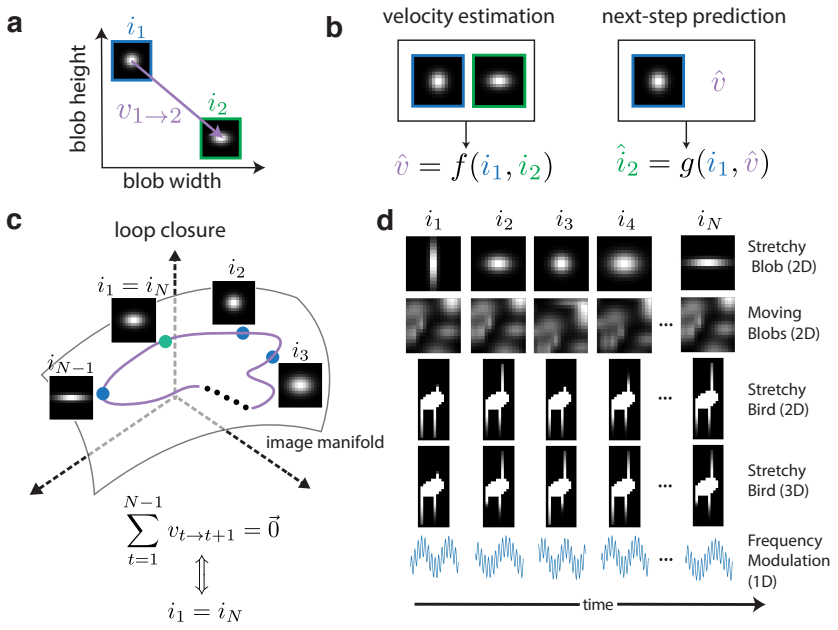

Figure 2: **Task and fundamental problem setup. a.** Two states in the 'Stretchy Blob' task that have a low-dimensional velocity representing the transition between them. **b.** If two consecutive images in this space i_1 and i_2 are separated by a velocity $v_{1\to2}$, can we learn a function f that estimates this velocity and a function g that 'applies' this quantity to i_1 to predict i_2? **c.** The key self-consistency constraint we introduce called 'loop-closure': estimated velocities along a closed-loop trajectory must sum to zero. This computation is performed by a neural integrator, such as grid cells. **d.** Our various procedurally generated abstract cognitive domains: Stretchy Blob (2D), Moving Blobs (2D), Stretchy Bird (2D), Stretchy Bird (3D), and Frequency Modulation (1D).

These self-consistency constraints on $\hat{v}$ suggests a *self-supervised learning paradigm, obviating the need for a specific coordinate system within the explored abstract space*.

To rigorously evaluate our system, we procedurally generate random trajectories across five different "abstract cognitive domains." These trajectories are generated by starting from an initial random state on the image manifold and then taking random velocities to determine the subsequent states on the same manifold. We assume for simplicity that states are unique at each point in space within these domains. We visualize a randomly sampled trajectory of each of these domains in Fig. 2d:

1. A 2D Stretchy Blob environment (discussed above) where a blob in the center of the visual field stretches or shrinks in height and/or width.
2. A 2D Stretchy Bird environment where a bird's legs and neck stretch and shrink. This environment is specifically constructed to mimic its experimental counterpart [15].
3. A 3D variant of Stretchy Bird where the two bird legs can independently transform.
4. A 2D Moving Blobs environment where a set of Gaussian blobs uniformly translate, moving in and out of the visual field.
5. A 1D Frequency Modulation task that emulates the experimental sound modulation task created by [2] with a sum of sine waves uniformly changing in frequency.

Detailed information regarding the generated data is provided in SI Sec. C.

2.2 Encoder-Decoder architecture

We construct an encoder-decoder architecture, wherein the encoder f and decoder g are both multi-layer perceptrons (MLPs) (Fig. 3a). The encoder f processes two adjacent inputs i_t and i_{t+1} from any of our procedurally generated trajectories and maps them to a low-dimensional velocity space. It

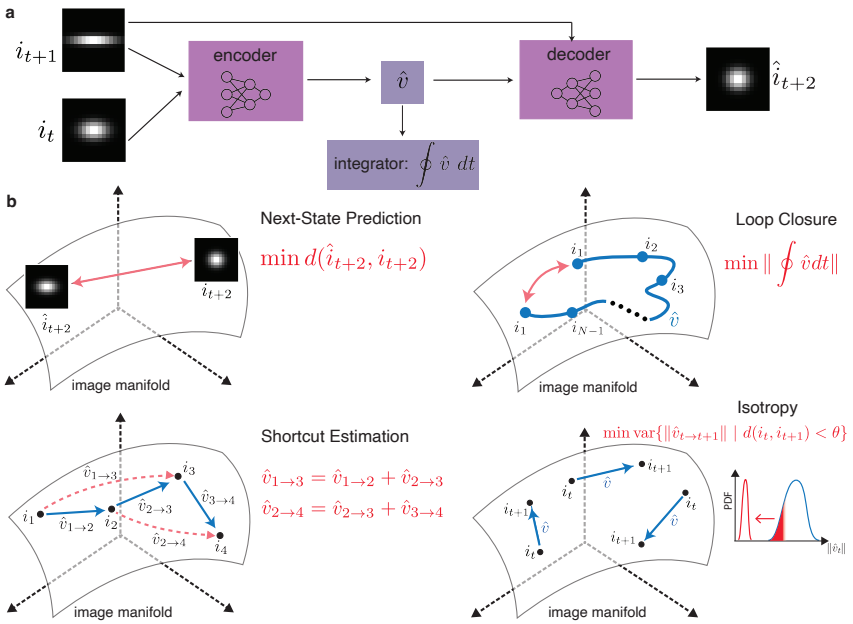

Figure 3: **Self-supervised learning framework. a.** Model diagram consisting of an encoder, decoder, and an integrator which acts on the low-dimensional velocity latent space. The model takes in two consecutive input frames and predicts an unseen frame with the learned velocity. **b.** Our various self-supervised loss terms. The two critical loss terms (*top*) are 'next-state prediction' and 'loop-closure'. The auxiliary loss terms (*bottom*) which further refine the solution space are 'shortcut estimation' and 'isotropy'.

is important to note that the low-dimensional velocity signal is not known a priori. The decoder g then upsamples this latent velocity representation and combines it with the last input to generate the subsequent $\hat{i}_{t+2}$ which remains unseen to the model.

A single sample in a batch of data for this model comprises a trajectory of states $\mathcal{T} = \{i_t \mid 1 \leq t \leq N\}$ and velocities $\mathcal{V} = \{v_{t\to t+1} \mid 1 \leq t \leq N-1\}$ (individual velocities abbreviated as v_t for ease). The model is trained through triplets of frames of the input trajectory, using i_t and i_{t+1} to predict i_{t+2} for $1 \leq t \leq N-1$. As discussed above, we train our models on triplets of states solely to ensure that the encoder does not memorize features of the image to be predicted. Training on pairs of images instead of triplets does not affect any of our results (SI Sec. B.2).

Detailed experimental procedures regarding the training of this model across various constructed domains are provided in SI Sec. C. We note that the same architectural motif was employed for training in all our experiments.

2.3 Loss functions

We formalize our two requirements of the extracted low-dimensional velocity signal into two critical loss terms, a next-state prediction loss, and a loop-closure loss. These losses from the core of our self-supervised learning framework. To further refine the solution space, we also employ auxiliary losses in addition to our primary constraints. Our loss terms are visualized in Figure 3b and are described in detail in SI Sec. C.

- **Next-State Prediction Loss.** We quantify this based on the difference between the next-state prediction $\hat{i}_{t+2}$ and the ground-truth frame i_{t+2}. This loss term operates on individual samples of the generated trajectory, and ensures decodability of the estimated velocity $\hat{v}$.
- **Loop-Closure Loss.** We quantify this as a norm of the sum of velocities along a closed loop trajectory $\mathcal{T}$, i.e., the model must produce velocity estimates such that $\sum_{\hat{v}\in\hat{\mathcal{V}}} \hat{v} = \vec{0}$. The error signal for this loss operates at the scale of the entire generated trajectory. For convenience, we construct all trajectories of our training data as random loops in the

considered abstract spaces, such that the start and end state of trajectories are identical. See SI Sec. B.2 for training data that is not solely loops.

- **Shortcut Estimation Loss.** The first of our two auxiliary loss terms is a shortcut estimation loss, which further tests the generalization ability of our decoder g. From i_{t+1}, we predict future states i_{t+3} or i_{t+4} by directly modifying $\hat{v}$. Specifically, if $\hat{v}_{2\to3}$ is inferred from i_{t+2} to i_{t+3} and $\hat{v}_{3\to4}$ is inferred from i_{t+3} to i_{t+4}, then $\hat{i}_{t+4}$ should be $\hat{v}_{2\to3} + \hat{v}_{3\to4}$ away from i_{t+2}. This loss is important for further refining our velocity estimates and ensuring their validity during generalization.
- **Isotropy Loss.** The loss terms considered so far do not ensure isotropy in the inferred velocity space, allowing differential scaling factors for transformations in different input space directions. To induce isotropy, we introduce an auxiliary isotropy loss term that acts on the norm of the velocities, independent of direction. Since we don't assume access to the global velocity distribution in the training data, the isotropy loss is applied only near zero velocity. In particular, we minimize the variance of $\{\|\hat{v}_{t\to t+1}\| \mid d(i_t, i_{t+1}) < \theta\}$, where $d(i_t, i_{t+1})$ is a similarity metric in the input image space, and θ is some small threshold. In practice, we use $1-$ cosine similarity as our distance metric d.

There is no direct supervision signal regressing the model outputs onto a known distribution of velocities; instead, the velocities are extracted and inferred automatically. Further, we do not a priori assume knowledge of the dimensionality of the underlying transition structure of the training domains.

Regardless of the training environment, the relative weighting ratio of the two critical loss terms remains consistent, with loop-closure loss always weighted ten times higher than next-state prediction loss. To stabilize the training of models with three-dimensional latent spaces, we include a small L_1 regularization during training. All models are evaluated on an unseen testing dataset consisting of random walks within the domain. To prevent overfitting, all models are deliberately underparameterized relative to the training dataset. We conduct ablation studies on our loss terms, which can be found in SI Sec. B.1.

3 Experimental results

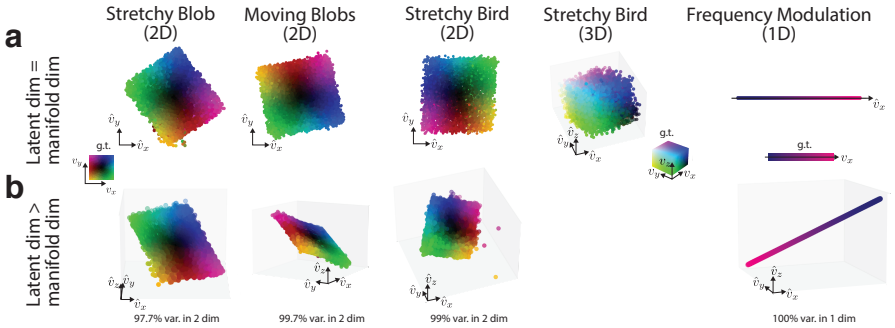

Figure 4: **Model results. a.** Our model produces low-dimensional velocity latents that are similar to the ground truth (g.t.) distribution without knowing this distribution across a variety of cognitive environments. **b.** In cases where the model's latent dimensionality is higher than the intrinsic velocity dimensionality of the environment, our model still identifies the lowest-dimensional representations embedded in higher-dimensional space.

3.1 Single learning framework infers geometrically consistent representations of velocity across cognitive domains

For each abstract environment, we compare the model-inferred velocity representations (Fig. 4a) to the true distribution of velocities. Note that the self-supervised framework does not result in an exact identity mapping between the ground truth velocities and the model outputs — it suffices for the obtained output to be a linear transformation of the ground-truth velocity space. Correspondingly, we

Task	Our Model	MCNet	Autoencoder	PCA	Isomap	UMAP
Stretchy Blob (2D)	**0.05 ± 0.01**	1.95 ± 0.14	(1.59 ± 3.40) $\times 10^3$	0.63	0.42	0.79
Stretchy Bird (2D)	**0.07 ± 0.03**	2.01 ± 0.01	20.90 ± 38.04	0.21	0.36	0.46
Stretchy Bird (3D)	**0.07 ± 0.02**	2.96 ± 0.10	(2.66 ± 5.08) $\times 10^2$	0.31	0.64	1.05
Moving Blobs (2D)	**0.02 ± 0.01**	2.00 ± 0.01	2.03 ± 0.05	1.94	0.66	0.62
Freq. Modulation (1D)	**0.02 ± 0.02**	2.01 ± 0.01	2.00 ± 0.01	1.97	2.00	2.00

Table 1: **Mean and standard deviation of errors for various tasks and models.** Mean and standard deviation of errors are computed across 6 different runs for each experiment. Each run was seeded to ensure reproducibility. The 6 seeds were picked randomly and are the same seeds used across different experiments where multiple runs were run.

construct an error metric on the inferred velocities as the mean squared error in mapping the predicted velocities to the ground-truth via a single linear transformation, after removing a small number of outliers via the DBSCAN clustering algorithm [36]. More details about this error metric can be found in SI Sec. C.

In all cases, irrespective of the dimensionality of the input manifold space or the detailed statistics and structure of the environment states, we see that the inferred velocities are faithful metric representations of the ground-truth velocities, quantified in Table 1.

While we primarily consider cases where the latent dimensionality of the encoder output matches the underlying dimensionality of transitions in the input space, this is not a necessity. In Fig. 4b, we set up our framework to have latent dimensionalities larger than the true data manifold dimensionality. In all cases, the model outputs *automatically* occupy a subspace of dimensionality that corresponds to the actual input manifold transition space, with a PCA of the inferred velocities capturing greater than 97% of the variance within the correct number of dimensions (cf. Fig. 4b). Thus, our model can effectively identify the appropriate low-dimensional structure within the high-dimensional embedding space of the inputs. Results from the other synthetic cognitive domains can be found in SI Sec. A.

3.2 Comparison to existing dimensionality reduction methods

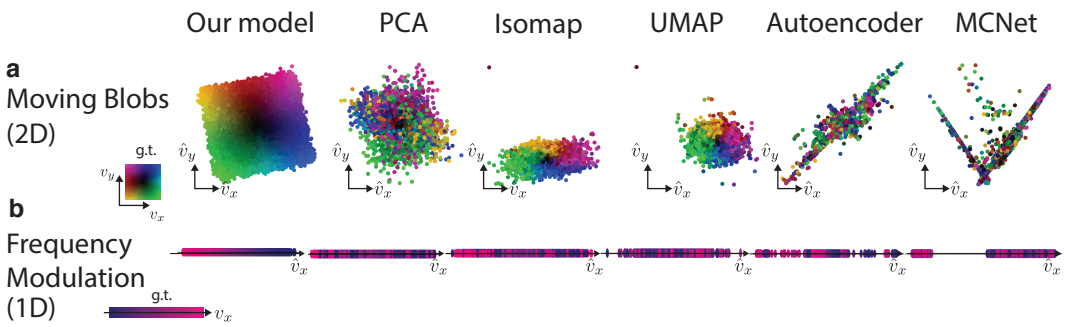

Figure 5: **Comparison to baselines.** We show our model comparisons to various dimensionality reduction and motion-prediction baselines in the **a.** 2D Moving Blobs and **b.** 1D Frequency Modulation tasks. Existing baselines cannot identify the low-dimensional velocity signals between arbitrary transitions in this space, even failing to do so in a simple one-dimensional domain. Meanwhile, our model produces results that closely match the true, underlying velocity distribution.

Our model estimates low-dimensional velocities between successive high-dimensional states. These velocities can then be integrated to determine a low-dimensional representation for each state. In this sense, it is possible to view our work as a dimensionality reduction method for continuously varying inputs. Traditional dimensionality reduction methods rely on the *statistics of distances between points* across an ensemble of states. In contrast, our approach finds a *structured tangent manifold* around each state that captures the low-dimensional transitions to successive states.

We can compare standard dimensionality reduction techniques such as PCA, Isomap, UMAP, and deep autoencoders with our method. To do so, we use these techniques to embed the inputs into the known

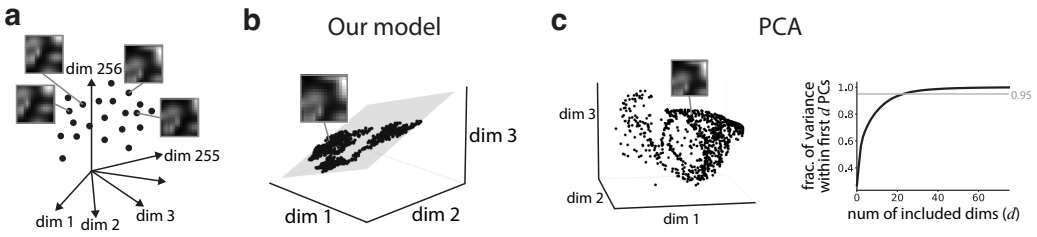

Figure 6: **Dimensionality reduction through our model, versus PCA. a.** Schematic of the raw input states from a random trajectory of the 2D Moving Blob task, showing the states as points in a 16x16 dimensional space. **b.** We estimate 3-dimensional velocities between states, and integrate these estimated velocities to obtain a low-dimensional representation of the initial input states. A 2-dimensional plane is shown in gray for perspective, demonstrating our model-produced low-dimensional representations are approximately in a 2D subspace. **c.** *Left*: Computing PCA on the same dataset shows representations occupying a volume in a 3-dimensional space. *Right*: Around 24 dimensions are required for PCA to capture 95% of the variance in the data, indicating that PCA is unable to find a low-dimensional space describing the dataset.

latent dimensionality of the data manifold, then compute an estimated low-dimensional velocity between states by taking the difference of their corresponding low-dimensional representations. Fig. 5 compares our model's estimated velocities with those produced by these standard baselines. Since our self-supervised framework uses a next-step prediction component, we also compared our results to MCNet [30] (a flexible deep network designed specifically for future frame prediction) and constrained the model to use low-dimensional representations of the transitions between frames. In all cases, our model significantly outperformed other baseline models (cf. Table 1) and produced velocities that were closely aligned with the true velocity distributions. Remarkably, for even one-dimensional manifolds embedded in high-dimensional spaces (as in our 1D Frequency Modulation task), existing dimensionality reduction techniques struggle to find a coherent representation for velocity and produce errors that are two orders of magnitude larger than our model.

We also examine dimensionality reduction through constructing representations of the original data via integrating the model velocity estimates, Fig. 6. For a sample trajectory through the high-dimensional states of the 2D Moving Blob environment, our models representation collapses onto a two-dimensional plane (consistent with the data, since the states can be minimally described through transitions in $\mathbb{R}^2$). PCA of the same set of states fails to capture this low-dimensional description, with ~ 24 principal components necessary to capture $> 95\%$ of the variance.

3.3 Model outputs allow reuse of grid cells in mapping abstract domains

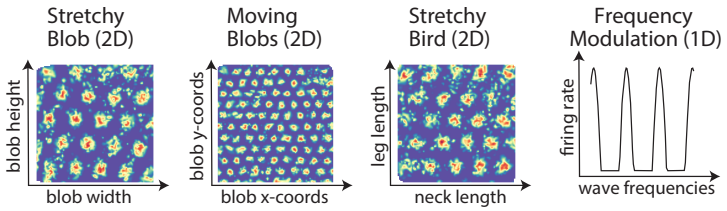

Figure 7: **Model outputs can be used to generate grid-like firing fields.** Our model's outputs can be used as input to a synthetic grid cell network across a variety of cognitive domains. We predict a clear hexagonal-like firing field when traversing these environments, illustrating how the grid cell circuit is crucial to building these cognitive maps.

We hypothesize that the brain can map abstract cognitive domains to the manifold of grid cell states by learning low-dimensional velocity representations of the input space. Extracting this low-dimensional signal allows the brain to reuse continuous attractor dynamics and path integration functionalities to stably represent and traverse along the manifold of inputs in the mapped domain.

To verify that the model inferred velocities are faithful enough for accurate path integration, we construct the tuning curve of a random neuron from a synthetic grid cell module (details of which are briefly described in SI Sec. C) that is fed these inferred velocities (up to a best-fit linear transformation; for simplicity we choose our transformation to map to aligned velocities across environments, leading to orientation-aligned grid tuning curves — this need not hold in experimentally observed tuning curves) as the path integration inputs. As seen in Fig. 7, our model results in hexagonal tuning curves in the abstract cognitive domains, consistent with previous work on grid representations [8–14, 3, 15, 2] (grid firing fields from baseline models explored in SI Sec. A). Through extracting faithful low-dimensional representations of velocities across abstract domains, the *same* continuous attractor-based grid modules can be used across tasks. As a result, if two grid cells fire in an overlapping (or non-overlapping) way in one mapped domain, they continue to be overlapping (or non-overlapping) in all other domains. Thus, cell-cell relationships are preserved between grid cells across abstract domains. This forms a testable hypothesis for future experimentation, that may be falsified if the brain were to use distinct, independent grid cell modules to organize information from different modalities, or if grid cells were not prestructured networks that function independently of the nature of the inputs.

We also note that while each grid cell module is a two-dimensional toroidal manifold, enabling integration of two-dimensional velocities, the grid cell system consisting of multiple modules can integrate velocities in higher dimensions[37]. Thus, higher dimensional velocities extracted by our model (e.g., 3D Stretchy Bird) do not pose a problem for integration by grid cells.

4 Discussion

Our work introduces the first neural network model that can infer velocities within abstract cognitive domains. This enables circuits like grid cells to encode transitions between high-dimensional sensory states through low-dimensional path integration, mapping sensory inputs to prestructured states instead of learning independent representations for each state. Our velocity extraction mechanism itself requires path-integration (via the 'loop-closure' loss), necessitating a grid-cell-like neural model, highlighting how grid cells form the foundation of mapping abstract spaces.

Future Work. Our research offers new perspectives for neuroscientists on the flexible utilization of grid cells to organize and map non-spatial domains. Future neuroscience research may test our hypothesis on the conservation of cell-cell relationships across cognitive domains and identify brain regions that generate velocity signals, along with their experimental signatures. Future machine learning research directions include scaling our framework for naturalistic environments, learning non-Euclidean spaces like family trees, and possibly using our framework to augment existing cognitive space-mapping models like TEM or CSCG.

Limitations. While our core SSL loss terms (next-state prediction and loop-closure) are biologically plausible and may align with sensory prediction error and neural integration, the auxiliary losses (shortcut and isotropy) are less biologically supported. Additionally, we assume velocity vectors in our latent space commute, which prevents them from directly representing tangent vectors in non-Euclidean spaces like a sphere. However, non-Euclidean spaces can be represented by embedding them in a higher-dimensional Euclidean space where velocities commute (e.g., a sphere embedded in three-dimensional space)[38, 39].

Broader Impact to AI. Our novel SSL framework can be applied for invertible dimensionality reduction (by virtue of a generative decoder which generates high-dimensional states corresponding to points in a low-dimensional latent) and manifold learning tasks. Our model significantly outperforms non-invertible dimensionality reduction baselines on datasets that contain relatively lower-dimensional transitions (suggesting applications to video data, for example). Our method also naturally lends itself to manifold alignment-related challenges, which is particularly effective when the data exhibits a small number of continuous modes of variability. Moreover, with a small number of "gluing" points, our method allows for building one-to-one correspondences between different domains. Our work also shows how a fixed integrator circuit can leverage common velocity representations to navigate between abstract spaces efficiently. For example, in a complex maze where learning action strategies is costly, our model maps transitions and actions to the grid-cell representational space, enabling strategies learned in a simpler, topologically similar space to be effectively applied to the complex domain — a relevant challenge in robotics.

5 Acknowledgements

We thank the McGovern Institute and the K. Lisa Yang Integrative Computational Neuroscience (ICoN) Center for supporting and funding this research. AI was supported by the K. Lisa Yang Integrative Computational Neuroscience (ICoN) Fellowship.

References

[1] Rylan Schaeffer, Mikail Khona, Tzuhsuan Ma, Cristobal Eyzaguirre, Sanmi Koyejo, and Ila Fiete. Self-supervised learning of representations for space generates multi-modular grid cells. *Advances in Neural Information Processing Systems*, 36, 2024.

[2] Dmitriy Aronov, Rhino Nevers, and David W Tank. Mapping of a non-spatial dimension by the hippocampal-entorhinal circuit. *Nature*, 543(7647):719–722, 2017.

[3] Nathaniel J Killian, Michael J Jutras, and Elizabeth A Buffalo. A map of visual space in the primate entorhinal cortex. *Nature*, 491(7426):761–764, November 2012.

[4] Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural computation*, 5(4):613–624, 1993.

[5] Kimberly L Stachenfeld, Matthew M Botvinick, and Samuel J Gershman. The hippocampus as a predictive map. *Nature neuroscience*, 20(11):1643–1653, 2017.

[6] Dileep George, Rajeev V Rikhye, Nishad Gothoskar, J Swaroop Guntupalli, Antoine Dedieu, and Miguel Lázaro-Gredilla. Clone-structured graph representations enable flexible learning and vicarious evaluation of cognitive maps. *Nature communications*, 12(1):2392, 2021.

[7] James CR Whittington, Timothy H Muller, Shirley Mark, Guifen Chen, Caswell Barry, Neil Burgess, and Timothy EJ Behrens. The tolman-eichenbaum machine: unifying space and relational memory through generalization in the hippocampal formation. *Cell*, 183(5):1249–1263, 2020.

[8] Niklas Wilming, Peter König, Seth König, and Elizabeth A Buffalo. Entorhinal cortex receptive fields are modulated by spatial attention, even without movement. *Elife*, 7, 2018.

[9] Seongmin A Park, Douglas S Miller, Hamed Nili, Charan Ranganath, and Erie D Boorman. Map making: Constructing, combining, and inferring on abstract cognitive maps. *Neuron*, July 2020.

[10] Seongmin A Park, Douglas S Miller, and Erie D Boorman. Inferences on a multidimensional social hierarchy use a grid-like code. *Nature neuroscience*, 24(9):1292–1301, 2021.

[11] Matthias Nau, Tobias Navarro Schröder, Jacob LS Bellmund, and Christian F Doeller. Hexadirectional coding of visual space in human entorhinal cortex. *Nature neuroscience*, 21(2):188–190, 2018.

[12] Xiaojun Bao, Eva Gjorgieva, Laura K Shanahan, James D Howard, Thorsten Kahnt, and Jay A Gottfried. Grid-like neural representations support olfactory navigation of a two-dimensional odor space. *Neuron*, 102(5):1066–1075, 2019.

[13] Joshua B Julian, Alexandra T Keinath, Giulia Frazzetta, and Russell A Epstein. Human entorhinal cortex represents visual space using a boundary-anchored grid. *Nature neuroscience*, 21(2):191–194, 2018.

[14] Simone Viganò, Valerio Rubino, Antonio Di Soccio, Marco Buiatti, and Manuela Piazza. Grid-like and distance codes for representing word meaning in the human brain. *NeuroImage*, 232:117876, 2021.

[15] Alexandra O Constantinescu, Jill X O'Reilly, and Timothy E J Behrens. Organizing conceptual knowledge in humans with a gridlike code. *Science*, 352(6292):1464–1468, June 2016.

[16] Yoram Burak and Ila R Fiete. Accurate path integration in continuous attractor network models of grid cells. *PLoS computational biology*, 5(2):e1000291, 2009.

[17] Mikail Khona and Ila R Fiete. Attractor and integrator networks in the brain. *Nat. Rev. Neurosci.*, 23(12):744–766, December 2022.

[18] Kijung Yoon, Michael A Buice, Caswell Barry, Robin Hayman, Neil Burgess, and Ila R Fiete. Specific evidence of low-dimensional continuous attractor dynamics in grid cells. *Nat. Neurosci.*, 16(8):1077–1084, August 2013.

[19] K J Yoon, S Lewallen, A A Kinkhabwala, D W Tank, and I R Fiete. Grid cell responses in 1D environments assessed as slices through a 2D lattice. *Neuron*, 89(5):1086–1099, March 2016.

[20] Richard J Gardner, Li Lu, Tanja Wernle, May-Britt Moser, and Edvard I Moser. Correlation structure of grid cells is preserved during sleep. *Nat. Neurosci.*, 22(4):598–608, 2019.

[21] Sean G Trettel, John B Trimper, Ernie Hwaun, Ila R Fiete, and Laura Lee Colgin. Grid cell co-activity patterns during sleep reflect spatial overlap of grid fields during active behaviors. *Nat. Neurosci.*, 22(4):609–617, April 2019.

[22] Richard J Gardner, Erik Hermansen, Marius Pachitariu, Yoram Burak, Nils A Baas, Benjamin A Dunn, May-Britt Moser, and Edvard I Moser. Toroidal topology of population activity in grid cells. February 2021.

[23] Ling L Dong and Ila R Fiete. Grid cells in cognition: Mechanisms and function. *Annual Review of Neuroscience*, 47, 2024.

[24] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[25] Joshua B Tenenbaum, Vin de Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[26] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[27] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[28] S Velliangiri, SJPCS Alagumuthukrishnan, et al. A review of dimensionality reduction techniques for efficient computation. *Procedia Computer Science*, 165:104–111, 2019.

[29] Weikuan Jia, Meili Sun, Jian Lian, and Sujuan Hou. Feature dimensionality reduction: a review. *Complex & Intelligent Systems*, 8(3):2663–2693, 2022.

[30] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. *arXiv preprint arXiv:1706.08033*, 2017.

[31] Yufei Zhang, Jeffrey O Kephart, and Qiang Ji. Incorporating physics principles for precise human motion prediction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6164–6174, 2024.

[32] Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. Learning trajectory dependencies for human motion prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9489–9497, 2019.

[33] Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and S Yu Philip. Predrnn++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. In *International conference on machine learning*, pages 5123–5132. PMLR, 2018.

[34] Judith Butepage, Michael J Black, Danica Kragic, and Hedvig Kjellstrom. Deep representation learning for human motion prediction and classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6158–6166, 2017.

[35] Sergiu Oprea, Pablo Martinez-Gonzalez, Alberto Garcia-Garcia, John Alejandro Castro-Vargas, Sergio Orts-Escolano, Jose Garcia-Rodriguez, and Antonis Argyros. A review on deep learning techniques for video prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(6):2806–2826, 2020.

[36] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[37] Mirko Klukas, Marcus Lewis, and Ila Fiete. Efficient and flexible representation of higher-dimensional cognitive variables with grid cells. *PLoS computational biology*, 16(4):e1007796, 2020.

[38] John Nash. Analyticity of the solutions of implicit function problems with analytic data. *Annals of Mathematics*, 84(3):345–355, 1966.

[39] Robert E Greene and Howard Jacobowitz. Analytic isometric embeddings. *Annals of Mathematics*, 93(1):189–204, 1971.

A Additional experiments

A.1 Additional results across domains

Fig. 8 includes further experiments of our framework across different domains and also compare them to more baselines. Fig. 9 visualizes the grid firing fields using our model outputs as input to a synthetic grid cell network and compares them to the same baselines. Our model produces the most faithful representations of velocity across various domains.

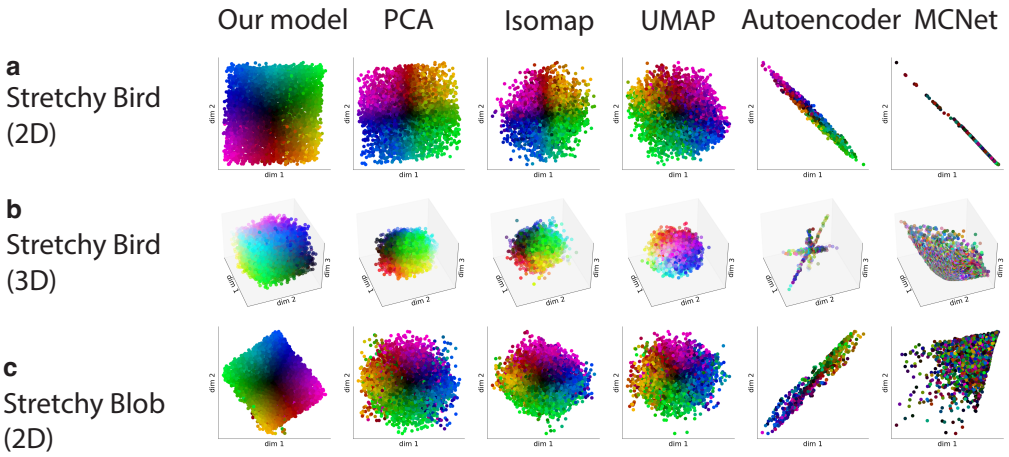

Figure 8: **Continued, comparison to baselines. a.** Model inferred velocity space in the 2D Stretchy Bird environment compared with baselines. **b.** Model inferred velocity space in the 3D Stretchy Bird environment compared with baselines. **c.** Model inferred velocity space in the 2D Stretchy Blob environment compared with baselines.

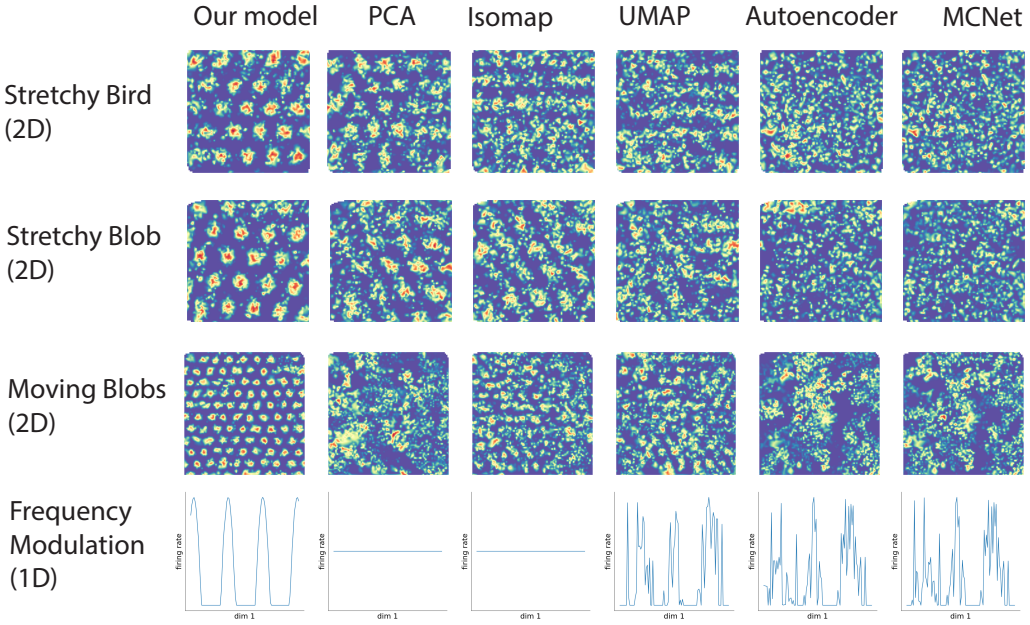

Figure 9: **Can baseline models produce faithful representations of velocity?** We pipe the outputs of our baselines into the same synthetic grid cell network to observe grid firing rates along a trajectory. Qualitatively, our model produces the most hexagonal grid firing field in comparison to other baselines.

A.2 Decoder implicitly learns boundaries of training data manifold

While the encoder is state-independent and infers a generalized notion of velocity between any two high-dimensional states within our abstract domains, the decoder is state dependent by construction as it outputs a predicted state given a state and velocity. We investigate how the decoder performs at boundaries in the 2D Stretchy Bird environment, as illustrated in Fig. 10. We find that the decoder implicitly understands the boundaries of the training data's underlying manifold. That is, once a velocity produces a bird state that is unseen in the training distribution, i.e. a bird that cannot further shrink its legs or extend its neck, further transitions in the same direction do not produce any changes in the predicted state. Thus, the decoder understands the boundaries observed in the training data.

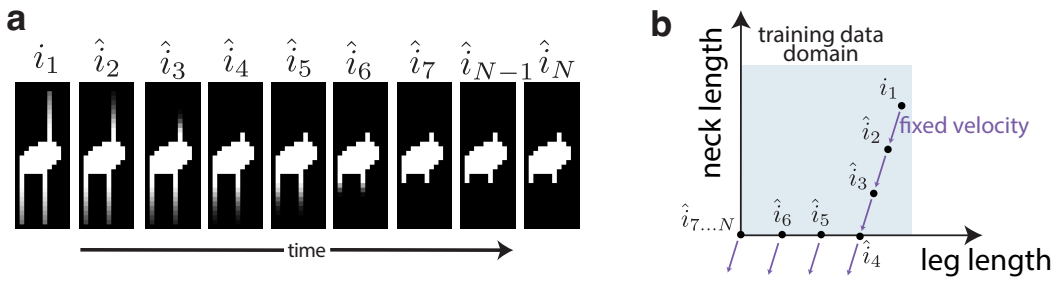

Figure 10: **Decoder respects boundaries inferred from the extent of the training data. a.** Starting from a random point in the 2D Stretchy Bird environment, we use the decoder to estimate the state after applying a velocity v. We then iteratively apply this same velocity v to the generated state estimate from the previous step, generating a series of states obtained by traversing the environment while following this fixed velocity. We find that once the neck and leg length have maximally shrunk to the extent observed in the training data, the decoder arrives at a fixed point. Thus, the decoder performs state-dependent transformations: after reaching an inferred boundary of the training data, further velocities along the same direction do not continue to transform the data along that dimension.

B Ablation studies

B.1 Loss Ablations

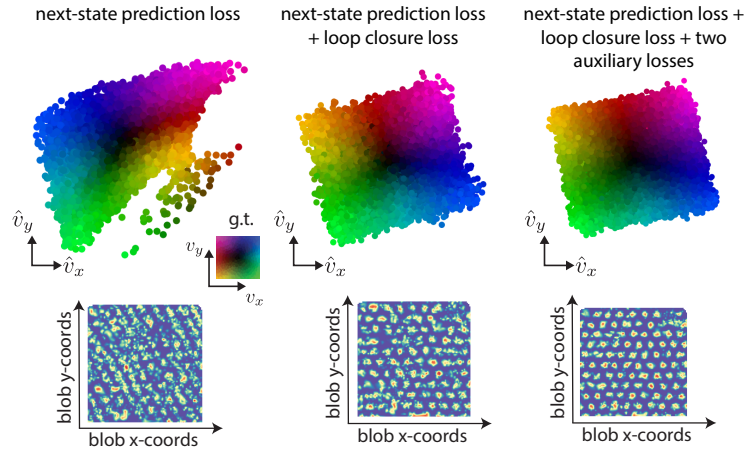

Figure 11: **Loss Ablation studies.** Loss ablation studies in the 2D Moving Blobs environment show that most of velocity estimation learning process comes from our two critical loss terms, 'next-state prediction' and 'loop-closure'. Further loss terms refine the solution space. Grid cell firing rates show that model generated velocities are faithful representations of the true, underlying velocity distribution.

We ablate our various loss terms in order to identify which losses are critical for faithful velocity extraction. As seen in Fig. 11 *left*, if we are interested in extracting some nonlinear low-dimensional

velocity between states in any abstract domain, the next-step prediction loss is enough (representation error of 0.134). However, since grid cells are doing path integration through linear addition of velocity vectors, constraining our latent space to also be a linear function of the true velocity signals is essential. Thus, to demand that the estimated velocities be linear functions of the ground truth velocities, we require the loop-closure loss, the addition of which is visualized in Fig. 11 *middle* (representation error of 0.044). If we require further refinement and guaranteed isotropic representations of velocities, we include our isotropy and shortcut losses, the addition of which is visualized in Fig. 11 *right* (representation error of 0.02). Examining the grid cell tuning curve within this domain reveals that using only the next-step prediction loss results in non-hexagonal firing fields. In contrast, incorporating the loop-closure loss results in hexagonal tuning curves, with further refinement achieved by adding auxiliary losses.

For completeness, we prove that the loop-closure loss exacts a strong linearity constraint on its inputs.

Proof: Let the function $e : v_{t\to t+1} \to \hat{v}_{t\to t+1}$ represent the mapping from the true velocity to the model estimated one. The loop-closure loss, applied on trajectories that form loops, applies the following constraint:

$$\sum_{0\le t\le T-1} v_{t\to t+1} = 0 \implies \sum_{0\le t\le T-1} e(v_{t\to t+1}) = 0.$$

We wish to show that this constraint implies that e must be a linear function over the reals. To show this, we first prove homogeneity, i.e., $e(\alpha v) = \alpha e(v)$ for all $\alpha \in \mathbb{R}$. Then we will prove additivity, i.e., $e(v_1 + v_2) = e(v_1) + e(v_2)$.

1. Homogeneity:

Consider a trajectory where $T = 2$, such that $v_{0\to1} = -v_{1\to2} = v$, and hence $v_{0\to1} + v_{1\to2} = 0$. Loop-closure implies:

$$e(v) + e(-v) = 0 \tag{1}$$
$$\implies -e(v) = e(-v). \tag{2}$$

This shows that e is an odd function, so $e(0) = 0$.

Now, consider a trajectory where $T = n + 1$ for any $n \in \mathbb{N}$:

$$v_{t\to t+1} = \begin{cases} v & \text{for } 0 \le t \le n-1, \\ -nv & \text{for } t = n. \end{cases}$$

Loop-closure implies:

$$ne(v) + e(-nv) = 0 \implies -ne(v) = e(-nv).$$

Using Eq. 2, we thus obtain

$$e(nv) = ne(v), \quad \forall n \in \mathbb{Z} \tag{3}$$

Thus, e is homogeneous over the integers.

To show that e is homogenous over the rationals, consider the loop with $T = 2$ given by $v_{0\to1} = mv$ for $m \in \mathbb{Z}$, and $v_{1\to2} = -nw$ for $n \in \mathbb{Z}^+$ with $w = \frac{mv}{n}$. The loop-closure condition implies to:

$$e(mv) + e(-nw) = 0 \tag{4}$$
$$\implies me(v) = ne\left(\frac{m}{n}v\right) \tag{5}$$
$$\implies \frac{m}{n}e(v) = e\left(\frac{m}{n}v\right). \tag{6}$$

Thus $e(\alpha v) = \alpha e(v)$ for $\alpha \in \mathbb{Q}$. Assuming that e is a continuous function, and since the rationals are dense in the reals, $e(\alpha v) = \alpha e(v)$ for $\alpha \in \mathbb{R}$, i.e., e is homogeneous over the reals.

2. Additivity:

Assume $T = 3$ with $v_{0\to1} = v$, $v_{1\to2} = w$, and $v_{2\to3} = -(v + w)$. Loop-closure implies:

$$e(v) + e(w) + e(-(v + w)) = 0 \tag{7}$$
$$\implies e(v) + e(w) = -e(-(v + w)) \tag{8}$$
$$\implies e(v) + e(w) = e(v + w), \tag{9}$$

for all velocity vectors v and w.

Conclusion: Since e is both homogeneous and additive, e is linear over the reals. Thus, the loop-closure loss constrains the estimated velocities to be a linear function of the true velocities, thereby inducing a global metric structure in this estimated space.

B.2 Data Ablations

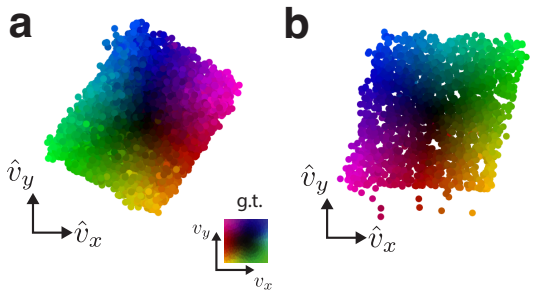

Figure 12: **Data Ablation studies. a.** Data ablation studies in the 2D Stretchy Bird environment show that training data does not need to consist only of loops. We created a dataset with 50% of trajectories as loops and 50% as independent random walks. The loop-closure loss applies only on loops while other loss terms apply on all trajectories. We report a transformation error of 0.035, comparable to models trained on only closed-loop trajectories. **b.** The training paradigm we previously employed asked the model to predict $\hat{i}_{t+2}$ given $\hat{v}_{t\to t+1}$ and i_{t+1}, assuming that $v_{t+1\to t+2} = v_{t\to t+1}$. This was done by construction to ensure that image features of i_{t+2} were not memorized in the velocity latent space. This paradigm is not necessary. We retrain our network on the 2D Moving Blobs task with trajectories that vary in velocity randomly at each time-step. We predict an estimated $\hat{v}_{t\to t+1}$ from i_t and i_{t+1}, and then predict $\hat{i}_{t+1}$ from $\hat{v}_{t\to t+1}$ and i_t. We report a transformation error of 0.02, comparable to models trained under our earlier training paradigm.

We conduct two ablation studies on our data generation process.

First, we note that while our models are trained with data consisting of closed-loop trajectories, this training paradigm is for convenience and not a necessity of our framework. If arbitrary random walks were selected instead for training samples, self-intersections would automatically lead to loops within subsequences of the walks. These loops could then be used for the loop-closure loss, with all other losses applied on all loop and non-loop trajectories. As a simple proof of concept, we generated a dataset of 2D Stretchy Bird trajectories with 50% of the trajectories as loops and the remainder as independent random walks. While the next-state prediction, isotropy, and shortcut losses applied to all trajectories, the loop-closure loss only applied to trajectories that formed loops. Training on this modified dataset achieves a transformation error of 0.035 (cf. Fig. 12a), comparable to the error reported in Table 1 (corresponding to models trained on only closed-loop trajectories). In greater generality, we note that the loop closure loss can also work on trajectories that are "almost loops" – i.e., a loss whose coefficient scales with how close a trajectory forms a closed loop.

Second, to ensure that our model velocity latent space did not memorize image features, we estimate a velocity $v_{t\to t+1}$ from i_t to i_{t+1}, and predict an unseen i_{t+2} that is $v_{t\to t+2} = v_{t\to t+1}$ away from i_{t+1}. Fig. 12b shows that this training paradigm is not a necessity. We train on trajectories from the Moving Blob environment whose velocities vary at each timestep. Training our model to estimate a velocity $v_{t\to t+1}$ from i_t to i_{t+1} and predicting i_{t+2} from $v_{t\to t+1}$ from i_t also results in similar performance. We obtain a transformation error of 0.02, again comparable to the error reported in Table 1 (corresponding to models trained under our earlier paradigm).

C Experimental Details

Code All experiments were run on a single NVIDIA Titan RTX GPU. Each experiment took anywhere from 1-5 hours to train. Code and all experimental runs can be found here: `https://github.com/abhi-iyer/velocity_extraction`.

Dataset construction. Details about our dataset construction can be found in Table 2. Each sample in the dataset is described as $\mathbf{x} \in \mathbb{R}^{\mathrm{T} \times \text{frame size}}$, which is a state consisting of two parts where T is the trajectory length:

- $\mathbf{x}_1 \in \mathbb{R}^{\frac{\mathrm{T}}{2} \times \text{frame size}}$, is a random walk.
- $\mathbf{x}_2$, which is a negative permutation of $\mathbf{x}_1$ such that the velocities between states satisfy $\int_0^T v(t)\, dt = 0$.

Loss function details. The loss terms, whose specific coefficients are also described in Table 2, are explicitly written out here:

- **Next-state prediction loss.** Given two states i_t and i_{t+1}, the next-state prediction loss minimizes the distance between the predicted state and the true state: min $||i_{t+2} - \hat{i}_{t+2}||_2$.
- **Loop-Closure Loss.** The loop-closure loss ensures that all the predicted velocities in a given trajectory sum to zero given that the trajectory is a loop: min $\sum_{0 \le t \le T-1} \hat{v}_{t \to t+1} =$ min $|| \oint \hat{v} dt||$.
- **Shortcut Loss.** The shortcut loss ensures that the decoder g can generalize given a state and a velocity. For instance, min $||g(i_{t+2}, \hat{v}_{2 \to 3} + \hat{v}_{3 \to 4}) - \hat{i}_{t+4}||$.
- **Isotropy loss.** Finally, the isotropy loss induces an isotropy in the inferred velocity space: min var$[\, ||\hat{v}_{t \to t+1}|| \;\mid\; d(i_t, i_{t+1}) < \theta \,]$, where d is a similarity function in the input image space and θ is some small threshold.

Best-fit linear transform as an error metric. We aim for the estimated velocities $\hat{\mathcal{V}}$ to be a linear function of the true velocities $\mathcal{V}$. Correspondingly, we first estimate the best-fit linear transformation T from $\hat{\mathcal{V}}$ to $\mathcal{V}$ via a pseudoinverse, $T = \mathcal{V}\hat{\mathcal{V}}^{\dagger}$. Then, we compute our error metric as a normalized mean-squared error between the transformed points and the true distribution: $e = \frac{||T\hat{\mathcal{V}} - \mathcal{V}||_2}{N \text{var}\{\mathcal{V}\}}$. The presence of outliers, particularly in some of the poorly performing baseline methods, can lead to particularly poor best-fits T. To alleviate this, we find the transformation T after removing a small number of outliers in $\hat{\mathcal{V}}$ via the DBSCAN clustering algorithm (any outlier rejection tool will suffice); however, we report the error e evaluated on the entire dataset including outliers.

Grid cell model. To visualize grid firing fields given inputs from our model, we use an approximation of a continuous attractor model for a module of grid cells: we simulate patterned activity on a lattice of neurons as the sum of three plane waves, resulting in a hexagonal pattern of activity. Input velocities are used to update the state of the activity on the lattice of neurons through updating the phases of the plane waves, leading to accurate integration of the input velocities.

Hyperparams	**Stretchy Blob (2D)**	**Stretchy Bird (2D)**	**Stretchy Bird (3D)**	**Moving Blobs (2D)**	**Frequency Modulation (1D)**
Frame size	16×16	32×12	32×12	16×16	1×100
Trajectory length	81	81	81	81	81
Velocity distribution	$U^2(0.05, 0.6)$	$U^2(-1.5, 1.5)$	$U^3(-1.5, 1.5)$	$U^2(-20, 20)$	$U(0.1, 10)$
Max velocity step	0.08	1.5	1.5	1.0	0.05
Optimizer	Adam	Adam	Adam	Adam	Adam
Learning Rate	5e-4	5e-4	5e-4	5e-4	5e-4
Epochs Trained	800	800	800	800	1200
Batch size	256	192	192	256	256
Learnable Parameters	536e3	622e3	622e3	536e3	544e3
State Prediction Weight	1	1e1	1e1	1	1
Loop-Closure Weight	1e1	1e2	1e2	1e1	1e1
Shortcut Estimation Weight	1	1e1	1e1	1	1
Isotropy Weight	1e2	1e2	1e2	1e2	1e2
Isotropy Threshold	1e-4	6e-3	6e-3	1e-2	1e-4
Training Set Size	800e3	800e3	800e3	800e3	800e3
Testing Set Size	200e3	200e3	200e3	200e3	200e3

Table 2: Hyperparameters used for generating the datasets and training the networks.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, the scope of what our paper tries to solve is clearly explained in several sections. We are clear about what challenges our paper solves and what future challenges can be solved as a result of this work.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, limitations to this paper, including proposals to address them in future directions of research, have been explained in the discussion section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.

- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Theoretical motivations for one of our SSL loss terms ('loop-closure') has been presented in the Supplementary Section. Assumptions for empirical experiments have been provided in various sections of the paper.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes, all necessary information (code, experimental runs, etc.) has been provided to reproduce the main results. Please see the Supplementary Material section for hyperparameters related to model training and dataset generation.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Link to the code, Wandb runs, and hyperparameters have been provided in the Supplementary Section.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Hyperparams for running the experiments and generating the data we have used are in the Supplementary Materials section. Experiments are fully reproducible.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Results include a table with mean and std of errors across a variety of experiments. These experiments, as mentioned in the paper, were across 6 random runs that are seed controlled (and thus fully reproducible).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Discussion about the computer resources used to run these experiments are discussed in the Supplementary Section.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, we have taken into account these ethics considerations. Our work follows the NeurIPS Code of Ethics fully.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This work does not have any immediate societal impact. We build a theoretical model of how grid cells can be used to map abstract cognitive spaces. Our model does not interact with any users nor does it collect real-world data.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our models pose no such risks. We do not train large models like LLMs that do pose risks to its users, nor do we collect real data for training our models.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Yes, we implemented baselines described by other papers and have cited them in our work properly.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We provide the code for generating the synthetic tasks/datasets that we trained our models on. This generation process is well documented.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper did not crowdsource experiments nor collect nor train on real-world or human data.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper did not crowdsource experiments nor collect or train on real-world or human data.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.