On Path Integration of Grid Cells: Group Representation and Isotropic Scaling

Ruiqi Gao[1]* ruiqigao@ucla.edu
Jianwen Xie[2] jianwen@ucla.edu
Xue-Xin Wei[3] weixx@utexas.edu
Song-Chun Zhu[1,4,5] sczhu@stat.ucla.edu
Ying Nian Wu[1] ywu@stat.ucla.edu

[1]Department of Statistics, UCLA [2]Cognitive Computing Lab, Baidu Research
[3]Department of Neuroscience, UT Austin [4]Department of Computer Science, UCLA
[5]Beijing Institute for General Artificial Intelligence (BIGAI)

Abstract

Understanding how grid cells perform path integration calculations remains a fundamental problem. In this paper, we conduct theoretical analysis of a general representation model of path integration by grid cells, where the 2D self-position is encoded as a higher dimensional vector, and the 2D self-motion is represented by a general transformation of the vector. We identify two conditions on the transformation. One is a group representation condition that is necessary for path integration. The other is an isotropic scaling condition that ensures locally conformal embedding, so that the error in the vector representation translates conformally to the error in the 2D self-position. Then we investigate the simplest transformation, i.e., the linear transformation, uncover its explicit algebraic and geometric structure as matrix Lie group of rotation, and explore the connection between the isotropic scaling condition and a special class of hexagon grid patterns. Finally, with our optimization-based approach, we manage to learn hexagon grid patterns that share similar properties of the grid cells in the rodent brain. The learned model is capable of accurate long distance path integration. Code is available at https://github.com/ruiqigao/grid-cell-path.

1 Introduction

Imagine walking in the darkness. Purely based on the sense of self-motion, one can gain a sense of self-position by integrating the self motion - a process often referred to as path integration [10, 15, 22, 16, 28]. While the exact neural underpinning of path integration remains unclear, it has been hypothesized that the grid cells [22, 18, 42, 25, 24, 12] in the mammalian medial entorhinal cortex (mEC) may be involved in this process [21, 31, 23]. The grid cells are so named because individual neurons exhibit striking firing patterns that form hexagonal grids when the agent (such as a rat) navigates in a 2D open field [19, 22, 17, 6, 35, 5, 7, 11, 30, 1]. The grid cells also interact with the place cells in the hippocampus [29]. Unlike a grid cell that fires at the vertices of a lattice, a place cell often fires at a single (or a few) locations.

The purpose of this paper is to understand how the grid cells may perform path integration calculations. We study a general optimization-based representational model in which the 2D self-position is

*The author is now a Research Scientist at Google Brain team.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

represented by a higher dimensional vector and the 2D self-motion is represented by a transformation of the vector. The vector representation can be considered position encoding or position embedding, where the elements of the vector may be interpreted as activities of a population of grid cells. The transformation can be realized by a recurrent network that acts on the vector. Our focus is to study the properties of the transformation.

Specifically, we identify two conditions for the transformation: a group representation condition and an isotropic scaling condition, under which we demonstrate that the local neighborhood around each self-position in the 2D physical space is embedded conformally as a 2D neighborhood around the vector representation of the self-position in the neural space.

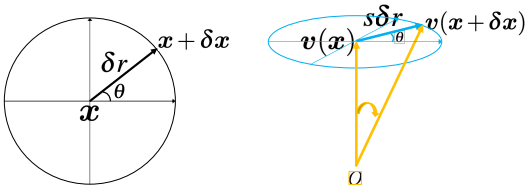
We then investigate the simplest special case of the transformation, i.e., linear transformation, that forms a matrix Lie group of rotation, under which case we show that the isotropic scaling condition is connected to a special class of hexagonal grid patterns. Our numerical experiments demonstrate that our model learns clear hexagon grid patterns of multiple scales which share observed properties of the grid cells in the rodent brain, by optimizing a simple loss function. The learned model is also capable of accurate long distance path integration.

Contributions. Our work contributes to understanding the grid cells from the perspective of representation learning. We conduct theoretical analysis of (1) general transformation for path integration by identifying two key conditions and a local conformal embedding property, (2) linear transformation by revealing the algebraic and geometric structure and connecting the isotropic scaling condition and a special class of hexagon grid patterns, and (3) integration of linear transformation model and linear basis expansion model via unitary group representation theory. Experimentally we learn clear hexagon grid patterns that are consistent with biological observations, and the learned model is capable of accurate path integration.

2 General transformation

2.1 Position embedding

Consider an agent (e.g., a rat) navigating within a 2D open field. Let $\boldsymbol{x} = (x_1, x_2)$ be the self-position of the agent. We assume that the self-position $\boldsymbol{x}$ in the 2D physical space is represented by the response activities of a population of d neurons (e.g., $d = 200$), which form a vector $\boldsymbol{v}(\boldsymbol{x}) = (v_i(\boldsymbol{x}), i = 1, ..., d)^\top$ in the d-dimensional "neural space", with each element $v_i(\boldsymbol{x})$ representing the firing rate of one neuron when the animal is at location $\boldsymbol{x}$.



(a) physical space (b) neural space

Figure 1: The local 2D polar system around self-position $\boldsymbol{x}$ in the 2D physical space (a) is embedded conformally as a 2D polar system around vector $\boldsymbol{v}(\boldsymbol{x})$ in the d-dimensional neural space (b), with a scaling factor s (so that δr in the physical space becomes $s\delta r$ in the neural space while the angle θ is preserved).

$\boldsymbol{v}(\boldsymbol{x})$ can be called position encoding or position embedding. Collectively, $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$ forms a *codebook* of $\boldsymbol{x} \in \mathbb{R}^2$, and $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$ is a *2D manifold* in the d-dimensional neural space, i.e., globally we embed $\mathbb{R}^2$ as a 2D manifold in the neural space. Locally, we identify two conditions under which the 2D local neighborhood around each $\boldsymbol{x}$ is embedded *conformally* as a 2D neighborhood around $\boldsymbol{v}(\boldsymbol{x})$ with a scaling factor. See Fig. 1. As shown in Section 3.3, the conformal embedding is connected to the hexagon grid patterns.

2.2 Transformation and path integration

At self-position $\boldsymbol{x}$, if the agent makes a self-motion $\Delta\boldsymbol{x} = (\Delta x_1, \Delta x_2)$, then it moves to $\boldsymbol{x} + \Delta\boldsymbol{x}$. Correspondingly, the vector representation $\boldsymbol{v}(\boldsymbol{x})$ is transformed to $\boldsymbol{v}(\boldsymbol{x} + \Delta\boldsymbol{x})$. The general form of the transformation can be formulated as:

$$\boldsymbol{v}(\boldsymbol{x} + \Delta\boldsymbol{x}) = F(\boldsymbol{v}(\boldsymbol{x}), \Delta\boldsymbol{x}). \tag{1}$$

The transformation $F(\cdot, \Delta\boldsymbol{x})$ can be considered a representation of $\Delta\boldsymbol{x}$, which forms a 2D additive group. We call Eq. (1) the *transformation model*. It can be implemented by a recurrent network to

derive a path integration model: if the agent starts from $\boldsymbol{x}_0$, and makes a sequence of moves $(\Delta \boldsymbol{x}_t, t = 1, ..., T)$, then the vector is updated by $\boldsymbol{v}_t = F(\boldsymbol{v}_{t-1}, \Delta \boldsymbol{x}_t)$, where $\boldsymbol{v}_0 = \boldsymbol{v}(\boldsymbol{x}_0)$, and $t = 1, ..., T$.

2.3 Group representation condition

The solution to the transformation model (Eq. (1)) should satisfy the following condition.

Condition 1. *(Group representation condition)* $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$ *and* $(F(\cdot, \Delta \boldsymbol{x}), \forall \Delta \boldsymbol{x})$ *form a representation of the 2D additive Euclidean group* $\mathbb{R}^2$ *in the sense that*

$$F(\boldsymbol{v}(\boldsymbol{x}), 0) = \boldsymbol{v}(\boldsymbol{x}), \ \forall \boldsymbol{x}; \tag{2}$$

$$F(\boldsymbol{v}(\boldsymbol{x}), \Delta \boldsymbol{x}_1 + \Delta \boldsymbol{x}_2) = F(F(\boldsymbol{v}(\boldsymbol{x}), \Delta \boldsymbol{x}_1), \Delta \boldsymbol{x}_2), \ \forall \boldsymbol{x}, \Delta \boldsymbol{x}_1, \Delta \boldsymbol{x}_2. \tag{3}$$

$(F(\cdot, \Delta \boldsymbol{x}), \forall \Delta \boldsymbol{x})$ is a Lie group of transformations acting on the codebook manifold $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$.

The reason for (2) is that if $\Delta \boldsymbol{x} = 0$, then $F(\cdot, 0)$ should be the identity transformation. Thus the codebook manifold $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$ consists of fixed points of the transformation $F(\cdot, 0)$. If $F(\cdot, 0)$ is furthermore a contraction around $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$, then $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$ are the attractor points.

The reason for (3) is that the agent can move in one step by $\Delta \boldsymbol{x}_1 + \Delta \boldsymbol{x}_2$, or first move by $\Delta \boldsymbol{x}_1$, and then move by $\Delta \boldsymbol{x}_2$. Both paths would end up at the same $\boldsymbol{x} + \Delta \boldsymbol{x}_1 + \Delta \boldsymbol{x}_2$, which is represented by the same $\boldsymbol{v}(\boldsymbol{x} + \Delta \boldsymbol{x}_1 + \Delta \boldsymbol{x}_2)$.

The group representation condition is a necessary self-consistent condition for the transformation model (Eq. (1)).

2.4 Egocentric self-motion

Self-motion $\Delta \boldsymbol{x}$ can also be parametrized egocentrically as $(\Delta r, \theta)$, where Δr is the displacement along the direction $\theta \in [0, 2\pi]$, so that $\Delta \boldsymbol{x} = (\Delta x_1 = \Delta r \cos \theta, \Delta x_2 = \Delta r \sin \theta)$. The egocentric self-motion may be more biologically plausible where θ is encoded by head direction, and Δr can be interpreted as the speed along direction θ. The transformation model then becomes

$$\boldsymbol{v}(\boldsymbol{x} + \Delta \boldsymbol{x}) = F(\boldsymbol{v}(\boldsymbol{x}), \Delta r, \theta), \tag{4}$$

where we continue to use $F(\cdot)$ for the transformation (with slight abuse of notation). $(\Delta r, \theta)$ form a polar coordinate system around $\boldsymbol{x}$.

2.5 Infinitesimal self-motion and directional derivative

In this subsection, we derive the transformation model for infinitesimal self-motion. While we use $\Delta \boldsymbol{x}$ or Δr to denote finite (non-infinitesimal) self-motion, we use $\delta \boldsymbol{x}$ or δr to denote infinitesimal self-motion. At self-position $\boldsymbol{x}$, for an infinitesimal displacement δr along direction θ, $\delta \boldsymbol{x} = (\delta x_1 = \delta r \cos \theta, \delta x_2 = \delta r \sin \theta)$. See Fig. 1 (a) for an illustration. Given that δr is infinitesimal, for any fixed θ, a first order Taylor expansion of $F(\boldsymbol{v}(\boldsymbol{x}), \delta r, \theta)$ with respect to δr gives us

$$\begin{aligned} \boldsymbol{v}(\boldsymbol{x} + \delta \boldsymbol{x}) &= F(\boldsymbol{v}(\boldsymbol{x}), \delta r, \theta) = F(\boldsymbol{v}(\boldsymbol{x}), 0, \theta) + F'(\boldsymbol{v}(\boldsymbol{x}), 0, \theta) \delta r + o(\delta r) \\ &= \boldsymbol{v}(\boldsymbol{x}) + f_\theta(\boldsymbol{v}(\boldsymbol{x})) \delta r + o(\delta r), \end{aligned} \tag{5}$$

where $F(\boldsymbol{v}(\boldsymbol{x}), 0, \theta) = \boldsymbol{v}(\boldsymbol{x})$ according to Condition 1, and $f_\theta(\boldsymbol{v}(\boldsymbol{x})) \coloneqq F'(\boldsymbol{v}(\boldsymbol{x}), 0, \theta)$ is the first derivative of $F(\boldsymbol{v}(\boldsymbol{x}), \Delta r, \theta)$ with respect to Δr at $\Delta r = 0$. $f_\theta(\boldsymbol{v}(\boldsymbol{x}))$ is the *directional derivative* of $F(\cdot)$ at self-position $\boldsymbol{x}$ and direction θ.

For a fixed θ, $(F(\cdot, \Delta r, \theta), \forall \Delta r)$ forms a one-parameter Lie group of transformations, and $f_\theta(\cdot)$ is the generator of its Lie algebra.

2.6 Isotropic scaling condition

With the directional derivative, we define the second condition as follows, which leads to locally conformal embedding and is connected to hexagon grid pattern.

Condition 2. *(Isotropic scaling condition) For any fixed* $\boldsymbol{x}$, $\|f_\theta(\boldsymbol{v}(\boldsymbol{x}))\|$ *is constant over* θ.

Let $f_0(\boldsymbol{v}(\boldsymbol{x}))$ denote $f_\theta(\boldsymbol{v}(\boldsymbol{x}))$ for $\theta = 0$, and $f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))$ denote $f_\theta(\boldsymbol{v}(\boldsymbol{x}))$ for $\theta = \pi/2$. Then we have the following theorem:

Theorem 1. *Assume group representation condition 1 and isotropic scaling condition 2. At any fixed $\boldsymbol{x}$, for the local motion $\delta\boldsymbol{x} = (\delta r\cos\theta, \delta r\sin\theta)$ around $\boldsymbol{x}$, let $\delta\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{x}+\delta\boldsymbol{x}) - \boldsymbol{v}(\boldsymbol{x})$ be the change of vector and $s = \|f_\theta(\boldsymbol{v}(\boldsymbol{x}))\|$, then we have $\|\delta\boldsymbol{v}\| = s\|\delta\boldsymbol{x}\|$. Moreover,*

$$\delta\boldsymbol{v} = f_\theta(\boldsymbol{v}(\boldsymbol{x}))\delta r + o(\delta r) = f_0(\boldsymbol{v}(\boldsymbol{x}))\delta r\cos\theta + f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))\delta r\sin\theta + o(\delta r), \tag{6}$$

where $f_0(\boldsymbol{v}(\boldsymbol{x}))$ and $f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))$ are two orthogonal basis vectors of equal norm s.

See Supplementary for a proof and Fig. 1(b) for an illustration. Theorem 1 indicates that the local 2D polar system around self-position $\boldsymbol{x}$ in the 2D physical space is embedded conformally as a 2D polar system around vector $\boldsymbol{v}(\boldsymbol{x})$ in the d-dimensional neural space, with a scaling factor s (our analysis is local for any fixed $\boldsymbol{x}$, and s may depend on $\boldsymbol{x}$). Conformal embedding is a generalization of isometric embedding, where the metric can be changed by a scaling factor s. If s is globally constant for all $\boldsymbol{x}$, then the intrinsic geometry of the codebook manifold $(\boldsymbol{v}(\boldsymbol{x}), \forall\boldsymbol{x})$ remains Euclidean, i.e., flat.

Why isotropic scaling and conformal embedding? The neurons are intrinsically noisy. During path integration, the errors may accumulate in $\boldsymbol{v}$. Moreover, when inferring self-position from visual image, it is possible that $\boldsymbol{v}$ is inferred first with error, and then $\boldsymbol{x}$ is decoded from the inferred $\boldsymbol{v}$. Due to isotropic scaling and conformal embedding, locally we have $\|\delta\boldsymbol{v}\| = s\|\delta\boldsymbol{x}\|$, which guarantees that the ℓ_2 error in $\boldsymbol{v}$ translates proportionally to the ℓ_2 error in $\boldsymbol{x}$, so that there will not be adversarial perturbations in $\boldsymbol{v}(\boldsymbol{x})$ that cause excessively big errors in $\boldsymbol{x}$. Specifically, we have the following theorem.

Theorem 2. *Assume the general transformation model (Eq. (4)) and the isotropic scaling condition. For any fixed $\boldsymbol{x}$, let $s = \|f_\theta(\boldsymbol{v}(\boldsymbol{x}))\|$, which is independent of θ. Suppose the neurons are noisy: $\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{x}) + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \tau^2\boldsymbol{I}_d)$ and d is the dimensionality of $\boldsymbol{v}$. Suppose the agent infers its 2D position $\hat{\boldsymbol{x}}$ from $\boldsymbol{v}$ by $\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}'} \|\boldsymbol{v} - \boldsymbol{v}(\boldsymbol{x}')\|^2$, i.e., $\boldsymbol{v}(\hat{\boldsymbol{x}})$ is the projection of $\boldsymbol{v}$ onto the 2D manifold formed by $(\boldsymbol{v}(\boldsymbol{x}'), \forall\boldsymbol{x}')$. Then we have*

$$\mathbb{E}\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|^2 = 2\tau^2/s^2. \tag{7}$$

See Supplementary for a proof.

Connection to continuous attractor neural network (CANN) defined on 2D torus. The group representation condition and the isotropic scaling condition appear to be satisfied by the CANN models [2, 6, 7, 30, 1] that are typically hand-designed on a 2D torus. See Supplementary for details.

3 Linear transformation

After studying the general transformation, we now investigate the linear transformation of $\boldsymbol{v}(\boldsymbol{x})$, for the following reasons. (1) It is the simplest transformation for which we can derive explicit algebraic and geometric results. (2) It enables us to connect the isotropic scaling condition to a special class of hexagon grid patterns. (3) In Section 4, we integrate it with the basis expansion model, which is also linear in $\boldsymbol{v}(\boldsymbol{x})$, via unitary group representation theory.

For finite (non-infinitesimal) self-motion, the linear transformation model is:

$$\boldsymbol{v}(\boldsymbol{x}+\Delta\boldsymbol{x}) = F(\boldsymbol{v}(\boldsymbol{x}), \Delta\boldsymbol{x}) = \boldsymbol{M}(\Delta\boldsymbol{x})\boldsymbol{v}(\boldsymbol{x}), \tag{8}$$

where $\boldsymbol{M}(\Delta\boldsymbol{x})$ is a matrix. The group representation condition becomes $\boldsymbol{M}(\Delta\boldsymbol{x}_1 + \Delta\boldsymbol{x}_2)\boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{M}(\Delta\boldsymbol{x}_2)\boldsymbol{M}(\Delta\boldsymbol{x}_1)\boldsymbol{v}(\boldsymbol{x})$, i.e., $\boldsymbol{M}(\Delta\boldsymbol{x})$ is a matrix representation of self-motion $\Delta\boldsymbol{x}$, and $\boldsymbol{M}(\Delta\boldsymbol{x})$ acts on the coding manifold $(\boldsymbol{v}(\boldsymbol{x}), \forall\boldsymbol{x})$. For egocentric parametrization of self-motion $(\Delta r, \theta)$, we can further write $\boldsymbol{M}(\Delta\boldsymbol{x}) = \boldsymbol{M}_\theta(\Delta r)$ for $\Delta\boldsymbol{x} = (\Delta r\cos\theta, \Delta r\sin\theta)$, and the linear model becomes $\boldsymbol{v}(\boldsymbol{x}+\Delta\boldsymbol{x}) = F(\boldsymbol{v}(\boldsymbol{x}), \Delta r, \theta) = \boldsymbol{M}_\theta(\Delta r)\boldsymbol{v}(\boldsymbol{x})$.

3.1 Algebraic structure: matrix Lie algebra and Lie group

For the linear model (Eq. (8)), the directional derivative is: $f_\theta(\boldsymbol{v}(\boldsymbol{x})) = F'(\boldsymbol{v}(\boldsymbol{x}), 0, \theta) = \boldsymbol{M}'_\theta(0)\boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x})$, where $\boldsymbol{B}(\theta) = \boldsymbol{M}'_\theta(0)$, which is the derivative of $\boldsymbol{M}_\theta(\Delta r)$ with respect to Δr at 0. For infinitesimal self-motion, the transformation model in Eq. (5) becomes

$$\boldsymbol{v}(\boldsymbol{x}+\delta\boldsymbol{x}) = (\boldsymbol{I} + \boldsymbol{B}(\theta)\delta r)\boldsymbol{v}(\boldsymbol{x}) + o(\delta r), \tag{9}$$

where $\boldsymbol{I}$ is the identity matrix. It can be considered a linear recurrent network where $\boldsymbol{B}(\theta)$ is the learnable weight matrix. We have the following theorem for the algebraic structure of the linear transformation.

Theorem 3. *Assume the linear transformation model so that for infinitesimal self-motion $(\delta r, \theta)$, the model is in the form of Eq. (9), then for finite displacement Δr,*

$$\boldsymbol{v}(\boldsymbol{x}+\Delta\boldsymbol{x}) = \boldsymbol{M}_\theta(\Delta r)\boldsymbol{v}(\boldsymbol{x}) = \exp(\boldsymbol{B}(\theta)\Delta r)\boldsymbol{v}(\boldsymbol{x}). \tag{10}$$

Proof. We can divide Δr into N steps, so that $\delta r = \Delta r/N \to 0$ as $N \to \infty$, and

$$\boldsymbol{v}(\boldsymbol{x}+\Delta\boldsymbol{x}) = (\boldsymbol{I} + \boldsymbol{B}(\theta)(\Delta r/N) + o(1/N))^N \boldsymbol{v}(\boldsymbol{x}) \to \exp(\boldsymbol{B}(\theta)\Delta r)\boldsymbol{v}(\boldsymbol{x}) \tag{11}$$

as $N \to \infty$. The matrix exponential map is defined by $\exp(A) = \sum_{n=0}^{\infty} A^n/n!$. □

The above math underlies the relationship between matrix Lie algebra and matrix Lie group in general [39]. For a fixed θ, the set of $\boldsymbol{M}_\theta(\Delta r) = \exp(\boldsymbol{B}(\theta)\Delta r)$ for $\Delta r \in \mathbb{R}$ forms a *matrix Lie group*, which is both a group and a manifold. The tangent space of $\boldsymbol{M}_\theta(\Delta r)$ at identity $\boldsymbol{I}$ is called *matrix Lie algebra*. $\boldsymbol{B}(\theta)$ is the basis of this tangent space, and is often referred to as the *generator matrix*.

Path integration. If the agent starts from $\boldsymbol{x}_0$, and make a sequence of moves $((\Delta r_t, \theta_t), t = 1, ..., T)$, then the vector representation of self-position is updated by

$$\boldsymbol{v}_t = \exp(\boldsymbol{B}(\theta_t)\Delta r_t)\boldsymbol{v}_{t-1}, \tag{12}$$

where $\boldsymbol{v}_0 = \boldsymbol{v}(\boldsymbol{x}_0)$, and $t = 1, ..., T$.

Approximation to exponential map. For a finite but small Δr, $\exp(\boldsymbol{B}(\theta)\Delta r)$ can be approximated by a second-order (or higher-order) Taylor expansion

$$\exp(\boldsymbol{B}(\theta)\Delta r) = \boldsymbol{I} + \boldsymbol{B}(\theta)\Delta r + \boldsymbol{B}(\theta)^2\Delta r^2/2 + o(\Delta r^2). \tag{13}$$

3.2 Geometric structure: rotation, periodicity, metic and error correction

If we assume $\boldsymbol{B}(\theta) = -\boldsymbol{B}(\theta)^\top$, i.e., skew-symmetric, then $\boldsymbol{I} + \boldsymbol{B}(\theta)\delta r$ in Eq. (9) is a rotation matrix operating on $\boldsymbol{v}(\boldsymbol{x})$, due to the fact that $(\boldsymbol{I} + \boldsymbol{B}(\theta)\delta r)(\boldsymbol{I} + \boldsymbol{B}(\theta)\delta r)^\top = \boldsymbol{I} + O(\delta r^2)$. For finite Δr, $\exp(\boldsymbol{B}(\theta)\Delta r)$ is also a rotation matrix, as it equals to the product of N matrices $\boldsymbol{I} + \boldsymbol{B}(\theta)(\Delta r/N)$ (Eq. (11)). The geometric interpretation is that, if the agent moves along the direction θ in the physical space, the vector $\boldsymbol{v}(\boldsymbol{x})$ is rotated by the matrix $\boldsymbol{B}(\theta)$ in the neural space, while the ℓ_2 norm $\|\boldsymbol{v}(\boldsymbol{x})\|^2$ remains fixed. We may interpret $\|\boldsymbol{v}(\boldsymbol{x})\|^2 = \sum_{i=1}^d v_i(\boldsymbol{x})^2$ as the total energy of grid cells. See Fig. 1(b).

The angle of rotation is given by $\|\boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x})\|\delta r/\|\boldsymbol{v}(\boldsymbol{x})\|$, because $\|\boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x})\|\delta r$ is the arc length and $\|\boldsymbol{v}(\boldsymbol{x})\|$ is the radius. If we further assume the isotropic scaling condition, which becomes that $\|f_\theta(\boldsymbol{v}(\boldsymbol{x}))\| = \|\boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x})\|$ is constant over θ for the linear model, then the angle of rotation can be written as $\mu\delta r$, where $\mu = \|\boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x})\|/\|\boldsymbol{v}(\boldsymbol{x})\|$ is independent of θ. Geometrically, μ tells us how fast the vector rotates in the neural space as the agent moves in the physical space. In practice, μ can be much bigger than 1 for the learned model, thus the vector can rotate back to itself in a short distance, causing the periodic patterns in the elements of $\boldsymbol{v}(\boldsymbol{x})$. μ captures the notion of metric.

For $\mu \gg 1$, the conformal embedding in Fig. 1 (b) **magnifies** the local motion in Fig. 1 (a), and this enables error correction [35]. More specifically, we have the following result, which is based on Theorem 2.

Proposition 1. *Assume the linear transformation model (Eq. (9)) and the isotropic scaling condition 2. For any fixed $\boldsymbol{x}$, let $\mu = \|\boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x})\|/\|\boldsymbol{v}(\boldsymbol{x})\|$. Suppose $\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{x}) + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \tau^2\boldsymbol{I}_d)$ and $\tau^2 = \alpha^2(\|\boldsymbol{v}(\boldsymbol{x})\|^2/d)$, so that α^2 measures the variance of noise relative to the average magnitude of $(v_i(\boldsymbol{x})^2, i = 1, ..., d)$. Suppose the agent infers its 2D position $\hat{\boldsymbol{x}}$ from $\boldsymbol{v}$ by $\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}'} \|\boldsymbol{v} - \boldsymbol{v}(\boldsymbol{x}')\|^2$. Then we have*

$$\mathbb{E}\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|^2 = 2\alpha^2/(\mu^2 d). \tag{14}$$

See Supplementary for a proof. By the above proposition, error correction of grid cells is due to two factors: (1) higher dimensionality d of $\boldsymbol{v}(\boldsymbol{x})$ for encoding 2D positions $\boldsymbol{x}$, and (2) a magnifying $\mu \gg 1$ (our analysis is local for any fixed $\boldsymbol{x}$, and μ may depend on $\boldsymbol{x}$).

3.3 Hexagon grid patterns formed by mixing Fourier waves

In this subsection, we make connection between the isotropic scaling condition 2 and a special class of hexagon grid patterns created by linearly mixing three Fourier plane waves whose directions are $2\pi/3$ apart. We show such linear mixing satisfies the linear transformation model and the isotropic scaling condition.

Theorem 4. *Let $\boldsymbol{e}(\boldsymbol{x}) = (\exp(i\langle \boldsymbol{a}_j, \boldsymbol{x}\rangle), j = 1,2,3)^\top$, where $(\boldsymbol{a}_j, j = 1,2,3)$ are three 2D vectors of equal norm, and the angle between every pair of them is $2\pi/3$. Let $\boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{U}\boldsymbol{e}(\boldsymbol{x})$, where $\boldsymbol{U}$ is an arbitrary unitary matrix. Let $\boldsymbol{B}(\theta) = \boldsymbol{U}^* \boldsymbol{D}(\theta)\boldsymbol{U}$, where $\boldsymbol{D}(\theta) = \text{diag}(i\langle \boldsymbol{a}_j, \boldsymbol{q}(\theta)\rangle, j = 1,2,3)$, with $\boldsymbol{q}(\theta) = (\cos\theta, \sin\theta)^\top$. Then $(\boldsymbol{v}(\boldsymbol{x}), \boldsymbol{B}(\theta))$ satisfies the linear transformation model (Eq. (9)) and the isotropic scaling condition 2. Moreover, $\boldsymbol{B}(\theta)$ is skew-symmetric.*

See Supplementary for a proof. We would like to emphasize that the above theorem analyzes a special case solution to our linear transformation model, but our optimization-based learning method **does not assume any superposition of Fourier basis functions** as in the theorem. Our experimental results are learned purely by optimizing a loss function based on the simple assumptions of our model with generic vectors and matrices.

We leave it to future work to theoretically prove that the isotropic scaling condition leads to hexagon grid patterns in either the general transformation model or the linear transformation model. The hexagon grid patterns are not limited to superpositions of three plane waves as in the above theorem.

3.4 Modules

Biologically, it is well established that grid cells are organized in discrete modules [4, 38] or blocks. We thus partition the vector $\boldsymbol{v}(\boldsymbol{x})$ into K blocks, $\boldsymbol{v}(\boldsymbol{x}) = (\boldsymbol{v}_k(\boldsymbol{x}), k = 1, ..., K)$. Correspondingly the generator matrices $\boldsymbol{B}(\theta) = \text{diag}(\boldsymbol{B}_k(\theta), k = 1, ..., K)$ are block diagonal, so that each sub-vector $\boldsymbol{v}_k(\boldsymbol{x})$ is rotated by a sub-matrix $\boldsymbol{B}_k(\theta)$. For the general transformation model, each sub-vector is transformed by a separate sub-network. By the same argument as in Section 3.2, let $\mu_k = \|\boldsymbol{B}_k \boldsymbol{v}_k(\boldsymbol{x})\|/\|\boldsymbol{v}_k(\boldsymbol{x})\|$, then μ_k is the metric of module k.

4 Interaction with place cells

4.1 Place cells

For each $\boldsymbol{v}(\boldsymbol{x})$, we need to uniquely decode $\boldsymbol{x}$ globally. This can be accomplished via interaction with place cells. Specifically, each place cell fires when the agent is at a specific position. Let $A(\boldsymbol{x}, \boldsymbol{x}')$ be the response map of the place cell associated with position $\boldsymbol{x}'$. It measures the adjacency between $\boldsymbol{x}$ and $\boldsymbol{x}'$. A commonly used form of $A(\boldsymbol{x}, \boldsymbol{x}')$ is the Gaussian adjacency kernel $A(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2/(2\sigma^2))$. The set of Gaussian adjacency kernels serve as inputs to our optimization-based method to learn grid cells.

4.2 Basis expansion

A popular model that connects place cells and grid cells is the following basis expansion model (or PCA-based model) [13]:

$$A(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{u}(\boldsymbol{x}')\rangle = \sum_{i=1}^{d} u_{i,\boldsymbol{x}'} v_i(\boldsymbol{x}), \qquad (15)$$

where $\boldsymbol{v}(\boldsymbol{x}) = (v_i(\boldsymbol{x}), i = 1, ..., d)^\top$, and $\boldsymbol{u}(\boldsymbol{x}') = (u_{i,\boldsymbol{x}'}, i = 1, ..., d)^\top$. Here $(v_i(\boldsymbol{x}), i = 1, ..., d)$ forms a set of d basis functions (which are functions of $\boldsymbol{x}$) for expanding $A(\boldsymbol{x}, \boldsymbol{x}')$ (which is a function of $\boldsymbol{x}$ for each place $\boldsymbol{x}'$), while $\boldsymbol{u}(\boldsymbol{x}')$ is the read-out weight vector for place cell at $\boldsymbol{x}'$ and needs to be learned. See Fig. 2 for an illustration. Experimental results on biological brains have shown that the connections from grid cells to place cells are excitatory [44, 32]. We thus assume that $u_{i,\boldsymbol{x}'} \geq 0$ for all i and $\boldsymbol{x}'$.

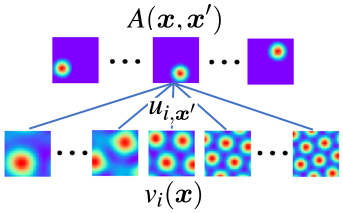


Figure 2: Illustration of basis expansion model $A(\boldsymbol{x}, \boldsymbol{x}') = \sum_{i=1}^{d} u_{i,\boldsymbol{x}'} v_i(\boldsymbol{x})$, where $v_i(\boldsymbol{x})$ is the response map of i-th grid cell, shown at the bottom, which shows 5 different i. $A(\boldsymbol{x}, \boldsymbol{x}')$ is the response map of place cell associated with $\boldsymbol{x}'$, shown at the top, which shows 3 different $\boldsymbol{x}'$. $u_{i,\boldsymbol{x}'}$ is the connection weight.

4.3 From group representation to basis functions

The vector representation $\boldsymbol{v}(\boldsymbol{x})$ generated (or constrained) by the linear transformation model (Eq. (8)) can serve as basis functions of the PCA-based basis expansion model (Eq. (15)), due to the fundamental theorems of Schur [43] and Peter-Weyl [39], which reveal the deep root of Fourier analysis and generalize it to general Lie groups. Specifically, if $\boldsymbol{M}(\Delta\boldsymbol{x})$ is an irreducible unitary representation of $\Delta\boldsymbol{x}$ that forms a compact Lie group, then the elements $\{M_{ij}(\Delta\boldsymbol{x})\}$ form a set of orthogonal basis functions of $\Delta\boldsymbol{x}$. Let $\boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{M}(\boldsymbol{x})\boldsymbol{v}(0)$ (where we choose the origin 0 as the reference point). The elements of $\boldsymbol{v}(\boldsymbol{x})$, i.e., $(v_i(\boldsymbol{x}), i = 1, ..., d)$, are linear mixings of the basis functions $\{M_{ij}(\boldsymbol{x})\}$, so that they themselves form a new set of basis functions that serve to expand $(A(\boldsymbol{x}, \boldsymbol{x}'), \forall \boldsymbol{x}')$ that parametrizes the place cells. Thus group representation in our path integration model is a perfect match to the basis expansion model, in the sense that the basis functions are results of group representation.

The basis expansion model (or PCA-based model) (Eq. (15)) assumes that the basis functions are orthogonal, whereas in our work, **we do not make the orthogonality assumption**. Interestingly, the learned transformation model generates basis functions that are close to being orthogonal automatically. See Supplementary for more detailed explanation and experimental results.

4.4 Decoding and re-encoding

For a neural response vector $\boldsymbol{v}$, such as $\boldsymbol{v}_t$ in Eq. (12), the response of the place cell associated with location $\boldsymbol{x}'$ is $\langle \boldsymbol{v}, \boldsymbol{u}(\boldsymbol{x}') \rangle$. We can decode the position $\hat{\boldsymbol{x}}$ by examining which place cell has the maximal response, i.e.,

$$\hat{\boldsymbol{x}} = \arg\max_{\boldsymbol{x}'} \langle \boldsymbol{v}, \boldsymbol{u}(\boldsymbol{x}') \rangle. \tag{16}$$

After decoding $\hat{\boldsymbol{x}}$, we can re-encode $\boldsymbol{v} \leftarrow \boldsymbol{v}(\hat{\boldsymbol{x}})$ for error correction. Decoding and re-encoding can also be done by directly projecting $\boldsymbol{v}$ onto the manifold $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$, which gives similar results. See Supplementary for more analysis and experimental results.

5 Learning

We learn the model by optimizing a loss function defined based on three model assumptions discussed above: (1) the basis expansion model (Eq. (15)), (2) the linear transformation model (Eq. (10)) and (3) the isotropic scaling condition 2. The input is the set of adjacency kernels $A(\boldsymbol{x}, \boldsymbol{x}'), \forall \boldsymbol{x}, \boldsymbol{x}'$. The unknown parameters to be learned are (1) $(\boldsymbol{v}(\boldsymbol{x}) = (\boldsymbol{v}_k(\boldsymbol{x}), k = 1, ..., K), \forall \boldsymbol{x})$, (2) $(\boldsymbol{u}(\boldsymbol{x}'), \forall \boldsymbol{x}')$ and (3) $(\boldsymbol{B}(\boldsymbol{\theta}), \forall \boldsymbol{\theta})$. We assume that there are K modules or blocks and $\boldsymbol{B}(\boldsymbol{\theta})$ is skew-symmetric, so that $\boldsymbol{B}(\boldsymbol{\theta})$ are parametrized as block-diagonal matrices $(\boldsymbol{B}_k(\boldsymbol{\theta}), k = 1, ..., K), \forall \boldsymbol{\theta})$ and only the lower triangle parts of the matrices need to be learned. The loss function is defined as a weighted sum of simple ℓ_2 loss terms constraining the three model assumptions: $L = L_0 + \lambda_1 L_1 + \lambda_2 L_2$, where

$$L_0 = \mathbb{E}_{\boldsymbol{x}, \boldsymbol{x}'}[A(\boldsymbol{x}, \boldsymbol{x}') - \langle \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{u}(\boldsymbol{x}') \rangle]^2, \text{ (basis expansion)} \tag{17}$$

$$L_1 = \sum_{k=1}^{K} \mathbb{E}_{\boldsymbol{x}, \Delta\boldsymbol{x}} \|\boldsymbol{v}_k(\boldsymbol{x} + \Delta\boldsymbol{x}) - \exp(\boldsymbol{B}_k(\boldsymbol{\theta})\Delta r)\boldsymbol{v}_k(\boldsymbol{x})\|^2, \text{ (transformation)} \tag{18}$$

$$L_2 = \sum_{k=1}^{K} \mathbb{E}_{\boldsymbol{x}, \theta, \Delta\theta}[\|\boldsymbol{B}_k(\theta + \Delta\theta)\boldsymbol{v}_k(\boldsymbol{x})\| - \|\boldsymbol{B}_k(\theta)\boldsymbol{v}_k(\boldsymbol{x})\|]^2. \text{ (isotropic scaling)} \tag{19}$$

In L_1, $\Delta\boldsymbol{x} = (\Delta r \cos\theta, \Delta r \sin\theta)$. λ_1 and λ_2 are chosen so that the three loss terms are of similar magnitudes. $A(\boldsymbol{x}, \boldsymbol{x}')$ are given as Gaussian adjacency kernels. For regularization, we add a penalty on $\|\boldsymbol{u}(\boldsymbol{x}')\|^2$, and further assume $\boldsymbol{u}(\boldsymbol{x}') \geq 0$ so that the connections from grid cells to place cells are excitatory [44, 32]. However, note that $\boldsymbol{u}(\boldsymbol{x}') \geq 0$ is not necessary for the emergence of hexagon grid patterns as shown in the ablation studies.

Expectations in L_0, L_1 and L_2 are approximated by Monte Carlo samples. L is minimized by *Adam* [26] optimizer. See Supplementary for implementation details.

It is worth noting that, consistent with the experimental observations, we assume individual place field $A(\boldsymbol{x}, \boldsymbol{x}')$ to exhibit a Gaussian shape, rather than a Mexican-hat pattern (with balanced excitatory center and inhibitory surround) as assumed in previous basis expansion models [13, 34] of grid cells.

ReLU non-linearity. We also experiment with a non-linear transformation model where a ReLU activation is added. See Supplementary for details.

6 Experiments

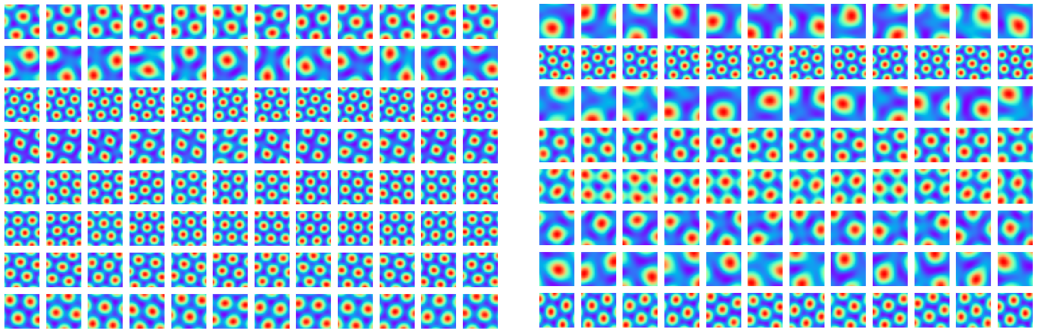


Figure 3: Hexagonal grid firing patterns emerge in the learned network. Every response map shows the firing pattern of one neuron (i.e, one element of $\boldsymbol{v}$) in the 2D environment. Every row shows the firing patterns of the neurons within the same block or module.

We conduct numerical experiments to learn the representations as described in Section 5. Specifically, we use a square environment with size 1m $\times$ 1m, which is discretized into a 40×40 lattice. For direction, we discretize the circle $[0, 2\pi]$ into 144 directions and use nearest neighbor linear interpolations for values in between. We use the second-order Taylor expansion (Eq. (13)) to approximate the exponential map $\exp(\boldsymbol{B}(\theta)\Delta r)$. The displacement Δr are sampled within a small range, i.e., Δr is smaller than 3 grids on the lattice. For $A(\boldsymbol{x}, \boldsymbol{x}')$, we use a Gaussian adjacency kernel with $\sigma = 0.07$. $\boldsymbol{v}(\boldsymbol{x})$ is of $d = 192$ dimensions, which is partitioned into $K = 16$ modules, each of which has 12 cells.

6.1 Hexagon grid patterns

Fig. 22 shows the learned firing patterns of $\boldsymbol{v}(\boldsymbol{x}) = (v_i(\boldsymbol{x}), i = 1, ..., d)$ over the 40×40 lattice of $\boldsymbol{x}$. Every row shows the learned units belonging to the same block or module. Regular hexagon grid patterns emerge. Within each block or module, the scales and orientations are roughly the same, but with different phases or spatial shifts. For the learned $\boldsymbol{B}(\theta)$, each element shows regular sine/cosine tuning over θ. See Supplementary for more learned patterns.

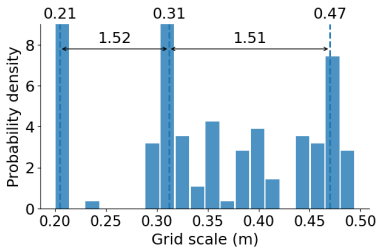


Figure 4: Multi-modal distribution of grid scales of the learned model grid cells. The scale ratios closely match the real data [38].

Table 1: Summary of gridness scores of the patterns learned from different models. To determine valid grid cells, we apply the same threshold of gridness score as in [3], i.e., gridness score > 0.37. For our model, we run 5 trials and report the average and standard deviation.

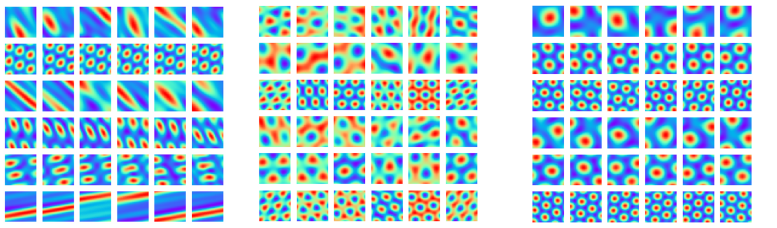
Model	Gridness score (↑)	% of grid cells
[3] (LSTM)	0.18	25.20
[34] (RNN)	0.48	56.10
Ours	**0.90** $\pm$ 0.044	**73.10** $\pm$ 1.33

We further investigate the characteristics of the learned firing patterns of $\boldsymbol{v}(\boldsymbol{x})$ using measures adopted from the literature of grid cells. Specifically, the hexagonal regularity, scale and orientation of grid-like patterns are quantified using the gridness score, grid scale and grid orientation [27, 33], which are determined by taking a circular sample of the autocorrelogram of the response map. Table 1 summarizes the results of gridness scores and comparisons with other optimization-based approaches [3, 34]. We apply the same threshold to determine whether a learned neuron can be considered a grid cell as in [3] (i.e., gridness score > 0.37). For our model, 73.10% of the learned neurons exhibit significant hexagonal periodicity in terms of the gridness score. Fig. 4 shows the

histogram of grid scales of the learned grid cell neurons (mean 0.33, range 0.21 to 0.49), which follows a multi-modal distribution. The ratio between neighboring modes are roughly 1.52 and 1.51, which closely matches the theoretical predictions [40, 37] and also the empirical results from rodent grid cells [38]. Collectively, these results reveal striking, quantitative correspondence between the properties of our model neurons and those of the grid cells in the brain.

Connection to continuous attractor neural network (CANN) defined on 2D torus. The fact that the learned response maps of each module are shifted versions of a common hexagon periodic pattern implies that the learned codebook manifold forms a 2D torus, and as the agent moves, the responses of the grid cells undergo a cyclic permutation. This is consistent with the CANN models hand-crafted on 2D torus. See Supplementary for a detailed discussion.

Ablation studies. We conduct ablation studies to examine whether certain model assumptions are empirically important for the emergence of hexagon grid patterns. The conclusions are highlighted as follows: (1) The loss term L_2 (Eq. (19)) constraining the isotropic scaling condition is necessary for learning hexagon grid patterns. (2) The constraint $\boldsymbol{u}(\boldsymbol{x}') \geq 0$ is not necessary for learning hexagon patterns, but the activations can be either excitatory or inhibitory without the constraint. (3) The skew-symmetric assumption on $\boldsymbol{B}(\boldsymbol{\theta})$ is not important for learning hexagon grid pattern. (4) Hexagon patterns always emerge regardless of the choice of block size and number of blocks. (5) Multiple blocks or modules are necessary for the emergence of hexagon grid patterns of multiple scales. See Fig. 5 for several learned patterns and Supplementary for the full studies.



(a) without L_2 (b) without $\boldsymbol{u}(\boldsymbol{x}') \geq 0$ (c) without skew-symmetry

Figure 5: Learned response maps in ablation studies where a certain model assumption is removed. (a) Remove the loss term L_2. (b) Remove the assumption $\boldsymbol{u}(\boldsymbol{x}') \geq 0$. (c) Remove the skew-symmetric assumption on $\boldsymbol{B}(\boldsymbol{\theta})$.

6.2 Path integration

We then examine the ability of the learned model on performing multi-step path integration, which can be accomplished by recurrently updating $\boldsymbol{v}_t$ (Eq. (12)) and decoding $\boldsymbol{v}_t$ to $\boldsymbol{x}_t$ for $t = 1, ..., T$ (Eq. (16)). Re-encoding $\boldsymbol{v}_t \leftarrow \boldsymbol{v}(\boldsymbol{x}_t)$ after decoding is adopted. Fig. 6(a) shows an example trajectory of accurate path integration for number of time steps $T = 30$. As shown in Fig. 6(b), with re-encoding, the path integration error remains close to zero over a duration of 500 time steps (< 0.01 cm, averaged over 1,000 episodes), even if the model is trained with the single-time-step transformation model (Eq. (18)). Without re-encoding, the error goes slight higher but still remains small (ranging from 0.0 to 4.2 cm, mean 1.9 cm in the 1m $\times$ 1m environment). Fig. 6(c) summarizes the path integration performance by fixing the number of blocks and altering the block size. The performance of path integration would be improved as the block size becomes larger, i.e., with more neurons in each module. When block size is larger than 16, path integration is very accurate for the time steps tested.

Error correction. See Supplementary for numerical experiments on error correction, which show that the learn model is still capable of path integration when we apply Gaussian white noise errors or Bernoulli drop-out errors to $\boldsymbol{v}_t$.

6.3 Additional experiments on path planning and egocentric vision

We also conduct additional experiments on path planning and egocentric vision with our model. Path planning can be accomplished by steepest ascent on the adjacency to the target position. For egocentric vision, we learn an extra generator network that generates the visual image given the position encoding formed by the grid cells. See Supplementary for details.

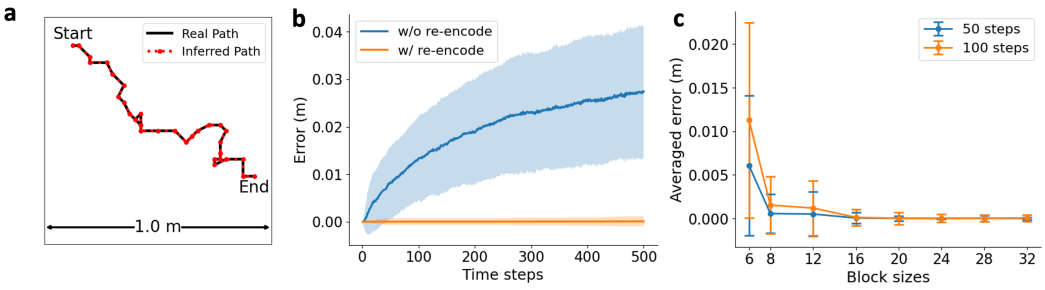


Figure 6: The learned model can perform accurate path integration. (a) Black: example trajectory. Red: inferred trajectory. (b) Path integration error over number of time steps, for procedures with re-encoding and without re-encoding. (c) Path integration error with fixed number of blocks and different block sizes, for 50 and 100 time steps. The error band in (b) and error bar in (c) are standard deviations computed over 1,000 episodes.

7 Related work

Our work is related to several lines of previous research on modeling grid cells. First, RNN models have been used to model grid cells and path integration. The traditional approach uses simulation-based models with hand-crafted connectivity, known as continuous attractor neural network (CANN) [2, 6, 7, 30, 1]. On the other hand, more recently two pioneering papers [9, 3] developed optimization-based RNN approaches to learn the path integration model and discovered that grid-like response patterns can emerge in the optimized networks. These results are further substantiated in [34, 8]. Our work analyzes the properties of the general recurrent model for path integration, and these properties seem to be satisfied by the hand-crafted CANN models. Our method belongs to the scheme of optimization-based approaches, and the learned response maps share similar properties as assumed by the CANN models.

Second, our work differs from the PCA-based basis expansion models [13, 34, 36] in that, unlike PCA, we make no assumption about the orthogonality between the basis functions, and the basis functions are generated by the transformation model. Furthermore, in previous basis expansion models [13, 34], place fields with Mexican-hat patterns (with balanced excitatory center and inhibitory surround) had to be assumed in order to obtain hexagonal grid firing patterns. However, experimentally measured place fields in biological brains were instead well characterized by Gaussian functions. Crucially, in our model, hexagonal grids emerge from learning with Gaussian place fields, and there is no need to assume any additional surround mechanisms or difference of Gaussians kernels.

In another related paper, [20] proposed matrix representation of 2D self-motion, while our work analyzes general transformations. Our investigation of the special case of linear transformation model reveals the matrix Lie group and the matrix Lie algebra of rotation group. Our work also connects the linear transformation model to the basis expansion model via unitary group representation theory.

8 Conclusion

This paper analyzes the recurrent model for path integration calculations by grid cells. We identify a group representation condition and an isotropic scaling condition that give rise to locally conformal embedding of the self-motion. We study a linear prototype model that reveals the matrix Lie group of rotation, and explore the connection between the isotropic scaling condition and hexagon grid patterns. In addition to these theoretical investigations, our numerical experiments demonstrate that our model can learn hexagon grid patterns for the response maps of grid cells, and the learned model is capable of accurate long distance path integration.

In this work, the numerical experiments are mostly limited to the linear transformation model, with the exception of an experiment with ReLU non-linearity. We will conduct experiments on the other non-linear transformation models, especially the forms assumed by the hand-crafted continuous attractor neural networks. Moreover, we assume that the agent navigates within a square open-field environment without obstacles or rewards. It is worthwhile to explore more complicated environments, including 3D environment.

Acknowledgments and Disclosure of Funding

The work was supported by NSF DMS-2015577, ONR MURI project N00014-16-1-2007, DARPA XAI project N66001-17-2-4029, and XSEDE grant ASC170063. We thank Yaxuan Zhu from UCLA Department of Statistics for his help with experiments on egocentric vision. We thank Dr. Wenhao Zhang for sharing his knowledge and insights on continuous attractor neural networks. We thank Sirui Xie for discussions. We thank the three reviewers for their constructive comments.

References

[1] Haggai Agmon and Yoram Burak. A theory of joint attractor dynamics in the hippocampus and the entorhinal cortex accounts for artificial remapping and grid cell field-to-field variability. *eLife*, 9:e56894, 2020.

[2] Daniel J Amit. *Modeling brain function: The world of attractor neural networks*. Cambridge university press, 1992.

[3] Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, Piotr Mirowski, Alexander Pritzel, Martin J Chadwick, Thomas Degris, Joseph Modayil, et al. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429, 2018.

[4] Caswell Barry, Robin Hayman, Neil Burgess, and Kathryn J Jeffery. Experience-dependent rescaling of entorhinal grids. *Nature neuroscience*, 10(6):682–684, 2007.

[5] Hugh T Blair, Adam C Welday, and Kechen Zhang. Scale-invariant memory representations emerge from moire interference between grid fields that produce theta oscillations: a computational model. *Journal of Neuroscience*, 27(12):3211–3229, 2007.

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

[7] Jonathan J Couey, Aree Witoelar, Sheng-Jia Zhang, Kang Zheng, Jing Ye, Benjamin Dunn, Rafal Czajkowski, May-Britt Moser, Edvard I Moser, Yasser Roudi, et al. Recurrent inhibitory circuitry as a mechanism for grid formation. *Nature neuroscience*, 16(3):318–324, 2013.

[8] Christopher J Cueva, Peter Y Wang, Matthew Chin, and Xue-Xin Wei. Emergence of functional and structural properties of the head direction system by optimization of recurrent neural networks. *International Conferences on Learning Representations (ICLR)*, 2020.

[9] Christopher J Cueva and Xue-Xin Wei. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. *arXiv preprint arXiv:1803.07770*, 2018.

[10] Charles Darwin. Origin of certain instincts, 1873.

[11] Licurgo de Almeida, Marco Idiart, and John E Lisman. The input–output transformation of the hippocampal granule cells: from grid cells to place fields. *Journal of Neuroscience*, 29(23):7504–7512, 2009.

[12] Christian F Doeller, Caswell Barry, and Neil Burgess. Evidence for grid cells in a human memory network. *Nature*, 463(7281):657, 2010.

[13] Yedidyah Dordek, Daniel Soudry, Ron Meir, and Dori Derdikman. Extracting grid cell characteristics from place cell inputs using non-negative principal component analysis. *Elife*, 5:e10094, 2016.

[14] William Gerard Dwyer and CW Wilkerson. The elementary geometric structure of compact lie groups. *Bulletin of the London Mathematical Society*, 30(4):337–364, 1998.

[15] Ariane S Etienne and Kathryn J Jeffery. Path integration in mammals. *Hippocampus*, 14(2):180–192, 2004.

[16] Ila R Fiete, Yoram Burak, and Ted Brookings. What grid cells convey about rat location. *Journal of Neuroscience*, 28(27):6858–6871, 2008.

[17] Mark C Fuhs and David S Touretzky. A spin glass model of path integration in rat medial entorhinal cortex. *Journal of Neuroscience*, 26(16):4266–4276, 2006.

[18] Marianne Fyhn, Torkel Hafting, Menno P Witter, Edvard I Moser, and May-Britt Moser. Grid cells in mice. *Hippocampus*, 18(12):1230–1238, 2008.

[19] Marianne Fyhn, Sturla Molden, Menno P Witter, Edvard I Moser, and May-Britt Moser. Spatial representation in the entorhinal cortex. *Science*, 305(5688):1258–1264, 2004.

[20] Ruiqi Gao, Jianwen Xie, Song-Chun Zhu, and Ying Nian Wu. Learning grid cells as vector representation of self-position coupled with matrix representation of self-motion. *arXiv preprint arXiv:1810.05597*, 2018.

[21] Mariana Gil, Mihai Ancau, Magdalene I Schlesiger, Angela Neitz, Kevin Allen, Rodrigo J De Marco, and Hannah Monyer. Impaired path integration in mice with disrupted grid cell

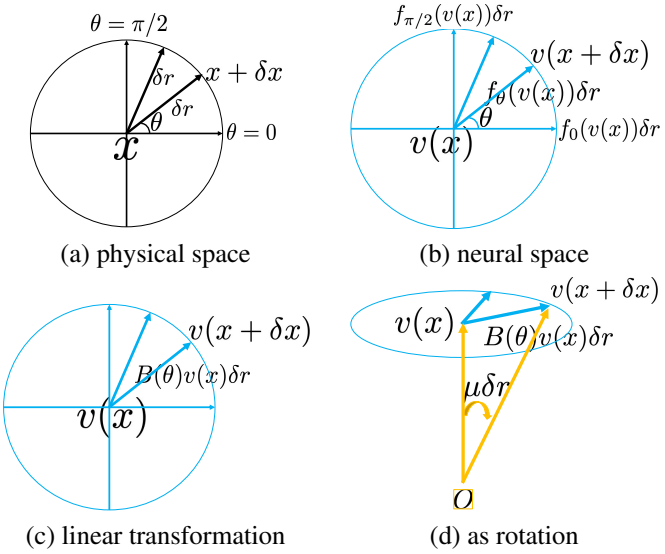
firing. *Nature neuroscience*, 21(1):81–91, 2018.
[22] Torkel Hafting, Marianne Fyhn, Sturla Molden, May-Britt Moser, and Edvard I Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801, 2005.
[23] Aidan J Horner, James A Bisby, Ewa Zotow, Daniel Bush, and Neil Burgess. Grid-like processing of imagined navigation. *Current Biology*, 26(6):842–847, 2016.
[24] Joshua Jacobs, Christoph T Weidemann, Jonathan F Miller, Alec Solway, John F Burke, Xue-Xin Wei, Nanthia Suthana, Michael R Sperling, Ashwini D Sharan, Itzhak Fried, et al. Direct recordings of grid-like neuronal activity in human spatial navigation. *Nature neuroscience*, 16(9):1188, 2013.
[25] Nathaniel J Killian, Michael J Jutras, and Elizabeth A Buffalo. A map of visual space in the primate entorhinal cortex. *Nature*, 491(7426):761, 2012.
[26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
[27] Rosamund F Langston, James A Ainge, Jonathan J Couey, Cathrin B Canto, Tale L Bjerknes, Menno P Witter, Edvard I Moser, and May-Britt Moser. Development of the spatial representation system in the rat. *Science*, 328(5985):1576–1580, 2010.
[28] Bruce L McNaughton, Francesco P Battaglia, Ole Jensen, Edvard I Moser, and May-Britt Moser. Path integration and the neural basis of the'cognitive map'. *Nature Reviews Neuroscience*, 7(8):663, 2006.
[29] John O'Keefe. A review of the hippocampal place cells. *Progress in neurobiology*, 13(4):419–439, 1979.
[30] Hugh Pastoll, Lukas Solanka, Mark CW van Rossum, and Matthew F Nolan. Feedback inhibition enables theta-nested gamma oscillations and grid firing fields. *Neuron*, 77(1):141–154, 2013.
[31] Thomas Ridler, Jonathan Witton, Keith G Phillips, Andrew D Randall, and Jonathan T Brown. Impaired speed encoding is associated with reduced grid cell periodicity in a mouse model of tauopathy. *bioRxiv*, page 595652, 2019.
[32] David C Rowland, Horst A Obenhaus, Emilie R Skytøen, Qiangwei Zhang, Cliff G Kentros, Edvard I Moser, and May-Britt Moser. Functional properties of stellate cells in medial entorhinal cortex layer ii. *Elife*, 7:e36664, 2018.
[33] Francesca Sargolini, Marianne Fyhn, Torkel Hafting, Bruce L McNaughton, Menno P Witter, May-Britt Moser, and Edvard I Moser. Conjunctive representation of position, direction, and velocity in entorhinal cortex. *Science*, 312(5774):758–762, 2006.
[34] Ben Sorscher, Gabriel Mel, Surya Ganguli, and Samuel A Ocko. A unified theory for the origin of grid cells through the lens of pattern formation. 2019.
[35] Sameet Sreenivasan and Ila Fiete. Grid cells generate an analog error-correcting code for singularly precise neural computation. *Nature neuroscience*, 14(10):1330, 2011.
[36] Kimberly L Stachenfeld, Matthew M Botvinick, and Samuel J Gershman. The hippocampus as a predictive map. *Nature neuroscience*, 20(11):1643, 2017.
[37] Martin Stemmler, Alexander Mathis, and Andreas VM Herz. Connecting multiple spatial scales to decode the population activity of grid cells. *Science Advances*, 1(11):e1500816, 2015.
[38] Hanne Stensola, Tor Stensola, Trygve Solstad, Kristian Frøland, May-Britt Moser, and Edvard I Moser. The entorhinal grid map is discretized. *Nature*, 492(7427):72, 2012.
[39] Michael Taylor. Lectures on lie groups. *Lecture Notes, available at http://www. unc. edu/math/Faculty/met/lieg. html*, 2002.
[40] Xue-Xin Wei, Jason Prentice, and Vijay Balasubramanian. A principle of economy predicts the functional architecture of grid cells. *Elife*, 4:e08362, 2015.
[41] Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9068–9079, 2018.
[42] Michael M Yartsev, Menno P Witter, and Nachum Ulanovsky. Grid cells without theta oscillations in the entorhinal cortex of bats. *Nature*, 479(7371):103, 2011.
[43] Anthony Zee. *Group theory in a nutshell for physicists*. Princeton University Press, 2016.
[44] Sheng-Jia Zhang, Jing Ye, Chenglin Miao, Albert Tsao, Ignas Cerniauskas, Debora Ledergerber, May-Britt Moser, and Edvard I Moser. Optogenetic dissection of entorhinal-hippocampal functional connectivity. *Science*, 340(6128), 2013.

Supplementary Materials

A Theoretical analysis

A.1 Graphical illustrations of key equations

Fig. 7 illustrates key equations in the main text as well as in the supplementary materials.



(a) physical space (b) neural space

(c) linear transformation (d) as rotation

Figure 7: Color-coded illustration. (a) In the 2D physical space, the agent moves from $\boldsymbol{x}$ to $\boldsymbol{x}+\delta\boldsymbol{x}$, where $\delta\boldsymbol{x} = (\delta r\cos\theta, \delta r\sin\theta)$, i.e., the agent moves by δr along the direction θ. We also show a displacement of δr in a different direction. (b) In the d-dimensional neural space, the vector $\boldsymbol{v}(\boldsymbol{x})$ is changed to $\boldsymbol{v}(\boldsymbol{x}+\delta\boldsymbol{x}) = F(\boldsymbol{v}(\boldsymbol{x}), \delta r, \theta) = \boldsymbol{v}(\boldsymbol{x}) + f_\theta(\boldsymbol{v}(\boldsymbol{x}))\delta r + o(\delta r)$, where the displacement is $f_\theta(\boldsymbol{v}(\boldsymbol{x}))\delta r = f_0(\boldsymbol{v}(\boldsymbol{x}))\delta r\cos\theta + f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))\delta r\sin\theta$. Under the isotropic condition that $\|f_\theta(\boldsymbol{v}(\boldsymbol{x}))\|$ is constant over θ, the local 2D self-motion $\delta\boldsymbol{x}$ at $\boldsymbol{x}$ in the 2D physical space is embedded conformally into the neural space as a 2D subspace around $\boldsymbol{v}(\boldsymbol{x})$. (c) Linear transformation, where $f_\theta(\boldsymbol{v}(\boldsymbol{x})) = \boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x})$. (d) 3D perspective view of linear transformation as a rotation: $\boldsymbol{v}(\boldsymbol{x}+\delta\boldsymbol{x})$ is a rotation of $\boldsymbol{v}(\boldsymbol{x})$, and the angle of rotation is $\mu\delta r$, where $\mu = \|\boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x})\|/\|\boldsymbol{v}(\boldsymbol{x}))\|$ (μ may depend on $\boldsymbol{x}$).

A.2 Proof of Theorem 1 on conformal embedding

Proof: See Fig. 7(a) and (b) for an illustration. Consider the self-motion $\delta\boldsymbol{x} = (\delta r\cos\theta, \delta r\sin\theta)$,

$$\boldsymbol{v}(\boldsymbol{x}+\delta\boldsymbol{x}) = F(\boldsymbol{v}(\boldsymbol{x}), \delta r, \theta) = \boldsymbol{v}(\boldsymbol{x}) + f_\theta(\boldsymbol{v}(\boldsymbol{x}))\delta r + o(\delta r). \tag{20}$$

We can decompose the self-motion $\delta\boldsymbol{x}$ into two steps. First move along the direction 0 by $\delta r\cos\theta$, and then move along the direction $\pi/2$ by $\delta r\sin\theta$. Then under the **group representation condition**:

$$\begin{aligned}
\boldsymbol{v}(\boldsymbol{x}+\delta\boldsymbol{x}) &= F[F(\boldsymbol{v}(\boldsymbol{x}), \delta r\cos\theta, 0), \delta r\sin\theta, \pi/2)] \\
&= F[\boldsymbol{v}(\boldsymbol{x}) + f_0(\boldsymbol{v}(\boldsymbol{x}))\delta r\cos\theta + o(\delta r), \delta r\sin\theta, \pi/2] \\
&= [\boldsymbol{v}(\boldsymbol{x}) + f_0(\boldsymbol{v}(\boldsymbol{x}))\delta r\cos\theta] + f_{\pi/2}[\boldsymbol{v}(\boldsymbol{x}) + f_0(\boldsymbol{v}(\boldsymbol{x}))\delta r\cos\theta + o(\delta r)]\delta r\sin\theta + o(\delta r) \\
&= \boldsymbol{v}(\boldsymbol{x}) + f_0(\boldsymbol{v}(\boldsymbol{x}))\delta r\cos\theta + f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))\delta r\sin\theta + o(\delta r),
\end{aligned} \tag{21}$$

The last equation holds because assuming the derivative $f'_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))$ exists, then by first-order Taylor expansion,

$$f_{\pi/2}[\boldsymbol{v}(\boldsymbol{x}) + f_0(\boldsymbol{v}(\boldsymbol{x}))\delta r\cos\theta + o(\delta r)]\delta r\sin\theta \tag{22}$$

$$= [f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x})) + f'_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))f_0(\boldsymbol{v}(\boldsymbol{x}))\delta r\cos\theta + o(\delta r)]\delta r\sin\theta \tag{23}$$

$$= f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))\delta r\sin\theta + o(\delta r). \tag{24}$$

Since $\boldsymbol{v}(\boldsymbol{x}+\delta\boldsymbol{x}) = \boldsymbol{v}(\boldsymbol{x}) + f_\theta(\boldsymbol{v}(\boldsymbol{x}))\delta r + o(\delta r)$, by Eq. (21) we have $f_\theta(\boldsymbol{v}(\boldsymbol{x})) = f_0(\boldsymbol{v}(\boldsymbol{x}))\cos\theta + f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))\sin\theta$, which is a 2D basis expansion. We are yet to prove that the two basis vectors $f_0(\boldsymbol{v}(\boldsymbol{x}))$ and $f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))$ are orthogonal with equal norm.

For notational simplicity, let $\boldsymbol{v}_1 = f_0(\boldsymbol{v}(\boldsymbol{x}))$ and $\boldsymbol{v}_2 = f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))$. Then under the **isotropic scaling condition**, $\|\boldsymbol{v}_1\| = \|\boldsymbol{v}_2\| = \|f_\theta(\boldsymbol{v}(\boldsymbol{x}))\| = s$, and $f_\theta(\boldsymbol{v}(\boldsymbol{x})) = \boldsymbol{v}_1\cos\theta + \boldsymbol{v}_2\sin\theta$ for any θ. Then we have that for any θ,

$$s^2 = \|f_\theta(\boldsymbol{v}(\boldsymbol{x}))\|^2 = \|\boldsymbol{v}_1\cos\theta + \boldsymbol{v}_2\sin\theta\|^2 = s^2 + 2\langle \boldsymbol{v}_1, \boldsymbol{v}_2\rangle \cos\theta\sin\theta. \tag{25}$$

Thus $\langle \boldsymbol{v}_1, \boldsymbol{v}_2\rangle = 0$, i.e., $f_0(\boldsymbol{v}(\boldsymbol{x})) \perp f_{\pi/2}(\boldsymbol{v}(\boldsymbol{x}))$. This leads to the conformal embedding of the local 2D polar system in the physical space as a 2D polar system in the d-dimensional neural space, with a scaling factor s (which may depend on $\boldsymbol{x}$). □

A.3 Proofs of Theorem 2 and Proposition 1 on error correction

Proof of Theorem 2: By Theorem 1, for a fixed self-position $\boldsymbol{x}$, we embed the 2D local neighborhood around $\boldsymbol{x}$ as a local 2D plane around $\boldsymbol{v}(\boldsymbol{x})$ in the d-dimensional neural space. A local perturbation in self-position, $\delta\boldsymbol{x}$, is translated into a local perturbation in $\boldsymbol{v}(\boldsymbol{v}+\delta\boldsymbol{x})$, so that

$$\|\delta\boldsymbol{v}\|^2 = \|f_\theta(\boldsymbol{v}(\boldsymbol{x}))\delta r + o(\delta r)\|^2 = s^2\|\delta\boldsymbol{x}\|^2, \tag{26}$$

where $\delta\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{x}+\delta\boldsymbol{x}) - \boldsymbol{v}(\boldsymbol{x})$.

Suppose the agent infers its 2D position $\hat{\boldsymbol{x}}$ by $\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}'} \|\boldsymbol{v} - \boldsymbol{v}(\boldsymbol{x}')\|^2$, which amounts to projecting $\boldsymbol{v}$ onto the local 2D plane around $\boldsymbol{v}(\boldsymbol{x})$. The projected vector $\boldsymbol{v}(\hat{\boldsymbol{x}})$ on the local 2D plane is $\boldsymbol{v}(\boldsymbol{x}) + \delta\boldsymbol{v}$, where $\delta\boldsymbol{v}$ is the projection of $\boldsymbol{\varepsilon}$ onto the 2D plane. More specifically, let $(\boldsymbol{v}_1, \boldsymbol{v}_2)$ be an orthonormal basis of the local 2D plane centered at $\boldsymbol{v}(\boldsymbol{x})$. Then $\delta\boldsymbol{v}$ can be written as $e_1\boldsymbol{v}_1 + e_2\boldsymbol{v}_2$, where

$$\boldsymbol{e} = (e_1, e_2)^\top = (\boldsymbol{v}_1, \boldsymbol{v}_2)^\top \boldsymbol{\varepsilon} \sim \mathcal{N}(0, \tau^2\boldsymbol{I}_2). \tag{27}$$

Let $\delta\boldsymbol{x} = \hat{\boldsymbol{x}} - \boldsymbol{x}$. Due to **isotropic scaling and conformal embedding**, the ℓ_2 squared error translate according to

$$\|\delta\boldsymbol{x}\|^2 = \|\delta\boldsymbol{v}\|^2/s^2 = (e_1^2 + e_2^2)/s^2, \tag{28}$$

whose expectation is $2\tau^2/s^2$. Thus $\mathbb{E}\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|^2 = 2\tau^2/s^2$.□

Proof of Proposition 1: It is reasonable to assume $\tau^2 = \alpha^2(\|\boldsymbol{v}(\boldsymbol{x})\|^2/d)$, where α^2 measures the variance of noise relative to $\|\boldsymbol{v}(\boldsymbol{x})\|^2/d$, which is the average of $(v_i(\boldsymbol{x})^2, i = 1, ..., d)$. In other words, α^2 measures the noise level.

In the linear case, the metric is

$$\mu = \|f_\theta(\boldsymbol{v}(\boldsymbol{x}))\|/\|\boldsymbol{v}(\boldsymbol{x})\| = \|\boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x}))\|/\|\boldsymbol{v}(\boldsymbol{x})\| = s/\|\boldsymbol{v}(\boldsymbol{x})\|, \tag{29}$$

which measures how fast $\boldsymbol{v}(\boldsymbol{x})$ rotates in the neural space as $\boldsymbol{x}$ changes. Then

$$\mathbb{E}\|\delta\boldsymbol{x}\|^2 = 2\alpha^2/(\mu^2 d). \tag{30}$$

The above scaling shows that error correction depends on two factors. One is the metric μ, and the other is the dimensionality d, i.e., the number of neurons. These correspond to two phases of error correction. One is to project the d-dimensional $\boldsymbol{\varepsilon}$ to the 2-dimensional $\delta\boldsymbol{v}$. The bigger d is, the better the error correction. The other is to translate $\|\delta\boldsymbol{v}\|^2$ to $\|\delta\boldsymbol{x}\|^2$. The bigger μ is, the better the error correction. □

A.4 Proof of Theorem 4 on hexagon grid patterns

Proof: Let $\boldsymbol{e}(\boldsymbol{x}) = (\exp(i\langle \boldsymbol{a}_j, \boldsymbol{x}\rangle), j = 1,2,3)^\top$, where $(\boldsymbol{a}_j, j = 1,2,3)$ are three 2D vectors of equal norm, and the angle between every pair of them is $2\pi/3$. Let $\boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{U}\boldsymbol{e}(\boldsymbol{x})$, where $\boldsymbol{U}$ is an arbitrary unitary matrix, i.e., $\boldsymbol{U}^*\boldsymbol{U} = \boldsymbol{I}$. Then $\|\boldsymbol{v}(\boldsymbol{x})\|^2 = \|\boldsymbol{e}(\boldsymbol{x})\|^2 = 3$, $\forall \boldsymbol{x}$, and $\boldsymbol{e}(\boldsymbol{x}) = \boldsymbol{U}^*\boldsymbol{v}(\boldsymbol{x})$. For self-motion $\delta\boldsymbol{x} = (\delta r\cos\theta, \delta r\sin\theta) = \boldsymbol{q}(\theta)\delta r$, let

$$\begin{aligned}
\Lambda(\delta\boldsymbol{x}, \theta) &= \text{diag}(\exp(\langle \boldsymbol{a}_j, \delta\boldsymbol{x}\rangle), j = 1,2,3) \\
&= \text{diag}(\exp(\langle \boldsymbol{a}_j, \boldsymbol{q}(\theta)\rangle\delta r), j = 1,2,3) \\
&= \boldsymbol{I} + \text{diag}(i\langle \boldsymbol{a}_j, \boldsymbol{q}(\theta)\rangle), j = 1,2,3)\delta r + o(\delta r) \\
&= \boldsymbol{I} + \boldsymbol{D}(\theta)\delta r + o(\delta r).
\end{aligned} \tag{31}$$

Then

$$\begin{aligned}\boldsymbol{v}(\boldsymbol{x}+\delta\boldsymbol{x}) &= \boldsymbol{U}\boldsymbol{e}(\boldsymbol{x}+\delta x)\\ &= \boldsymbol{U}\Lambda(\delta\boldsymbol{x},\theta)\boldsymbol{e}(\boldsymbol{x})\\ &= \boldsymbol{U}\Lambda(\delta\boldsymbol{x},\theta)\boldsymbol{U}^*\boldsymbol{v}(\boldsymbol{x})\\ &= (\boldsymbol{I}+\boldsymbol{U}\boldsymbol{D}(\theta)\boldsymbol{U}^*\boldsymbol{v}(\boldsymbol{x})\delta r)\boldsymbol{v}(\boldsymbol{x})+o(\delta r)\\ &= (\boldsymbol{I}+\boldsymbol{B}(\theta)\delta r)\boldsymbol{v}(\boldsymbol{x})+o(\delta r),\end{aligned} \tag{32}$$

where $\boldsymbol{B}(\theta)=\boldsymbol{U}\boldsymbol{D}(\theta)\boldsymbol{U}^*$, and $\boldsymbol{B}(\theta)=-\boldsymbol{B}(\theta)^*$. For isotropic condition,

$$\begin{aligned}\|\boldsymbol{B}(\theta)\boldsymbol{v}(\boldsymbol{x})\|^2 &= \|\boldsymbol{D}(\theta)\boldsymbol{e}(\boldsymbol{x})\|^2\\ &= \sum_{j=1}^{3}\langle \boldsymbol{a}_j, \boldsymbol{q}(\theta)\rangle^2\\ &= \text{const}\|\boldsymbol{a}_j\|^2\|\boldsymbol{q}(\theta)\|^2 = \text{const}\|\boldsymbol{a}_j\|^2,\end{aligned} \tag{33}$$

which is independent of θ, because $(\boldsymbol{a}_j, j=1,2,3)$ forms a **tight frame** in 2D.

One example of $\boldsymbol{U}$ is the following matrix:

$$\frac{1}{\sqrt{3}}\begin{pmatrix}1 & 1 & 1\\ 1 & \exp(i2\pi/3) & \exp(-i2\pi/3)\\ 1 & \exp(-i2\pi/3) & \exp(i2\pi/3)\end{pmatrix} \tag{34}$$

The resulting $(v_i(x), i=1,2,3)$ have the same orientation but different phases, i.e., they are spatially shifted versions of each other. □

The limitation of Theorem 4 is that we only show $\boldsymbol{v}(\boldsymbol{x})=\boldsymbol{U}\boldsymbol{e}(\boldsymbol{x})$ satisfies the linear model and the isotropic scaling condition, but we did not show that linear model with isotropic condition only has solutions that are hexagon grid patterns.

A.5 From group representation to orthogonal basis functions

Group representation is a central theme in modern mathematics and physics. In particular, it leads to a deep understanding and generalization of Fourier analysis or harmonic analysis.

For the set of $(\Delta\boldsymbol{x})$ that form a group, a matrix representation $\boldsymbol{M}(\Delta\boldsymbol{x})$ is equivalent to another representation $\tilde{\boldsymbol{M}}(\Delta\boldsymbol{x})$ if there exists an invertible matrix $\boldsymbol{P}$ such that $\tilde{\boldsymbol{M}}(\Delta\boldsymbol{x})=\boldsymbol{P}\boldsymbol{M}(\Delta\boldsymbol{x})\boldsymbol{P}^{-1}$ for each $\boldsymbol{x}$. A matrix representation is reducible if it is equivalent to a block diagonal matrix representation, i.e., we can find a matrix $\boldsymbol{P}$, such that $\boldsymbol{P}\boldsymbol{M}(\Delta\boldsymbol{x})\boldsymbol{P}^{-1}$ is block diagonal for every $\Delta\boldsymbol{x}$. Suppose the group is a finite group or a compact Lie group, and $\boldsymbol{M}$ is a unitary representation, i.e., $\boldsymbol{M}(\Delta\boldsymbol{x})$ is a unitary matrix. If $\boldsymbol{M}$ is block-diagonal, $\boldsymbol{M}=\text{diag}(\boldsymbol{M}_k, k=1,...,K)$, with non-equivalent blocks, and each block $\boldsymbol{M}_k$ cannot be further reduced, then the matrix elements $(M_{kij}(\Delta\boldsymbol{x}))$ are orthogonal basis functions of $\Delta\boldsymbol{x}$. Such orthogonality relations are proved by Schur [43] for finite group, and by Peter-Weyl for compact Lie group [39]. For our case, theoretically the group of displacements $\Delta\boldsymbol{x}$ in the 2D domain is $\mathbb{R}^2$, but we learn our model within a finite range, and we further discretize the range into a lattice. Thus the above orthogonal relations hold.

In our model, we also assume block diagonal $\boldsymbol{M}$, and we call each block a module. However, we do not assume each module is irreducible, i.e., each module itself may be further diagonalized into a block diagonal matrix of irreducible sub-blocks. Thus the elements within the same module $\boldsymbol{v}_k(\boldsymbol{x})$ may be linear mixings of orthogonal basis functions of the irreducible sub-blocks, and the linear mixings themselves are not necessarily orthogonal.

Fig. 8 visualizes the correlation between pairs of the learned $\boldsymbol{v}_i(\boldsymbol{x})$ and $\boldsymbol{v}_j(\boldsymbol{x})$, $i,j=1,...,d$. For different i and j, the correlations between different $\boldsymbol{v}_i(\boldsymbol{x})$ and $\boldsymbol{v}_j(\boldsymbol{x})$ are close to zero; i.e., they are nearly orthogonal to each other. The average absolute value of correlation is 0.09, and the within-block average value is about the same as the between-block average value.

Unlike previous work on learning basis expansion model (or PCA-based model [13]), we **do not constrain the basis functions $\boldsymbol{v}(\boldsymbol{x})=(v_i(\boldsymbol{x}), i=1,...,d)$ to be orthogonal to each other**. Instead, we constrain them by our path integration model via the loss term L_1. Nonetheless, the learned $v_i(\boldsymbol{x})$ are close to being orthogonal in our experiments.

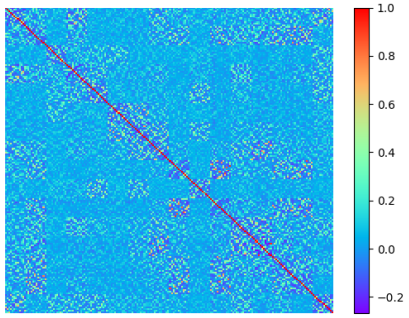


Figure 8: Correlation heatmap for each pair of the learned $v_i(\boldsymbol{x})$ and $v_j(\boldsymbol{x})$. The correlations are computed over 40×40 lattice of $\boldsymbol{x}$.

A.6 Decoding and re-encoding

In the above analysis, the projection of $\boldsymbol{v}$ onto the local 2D plane around $\boldsymbol{v}(\boldsymbol{x})$ is $\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}'} \|\boldsymbol{v} - \boldsymbol{v}(\boldsymbol{x}')\|^2$, which, for the linear model, amounts to decoding $\boldsymbol{v}$ to $\hat{\boldsymbol{x}}$ via

$$\hat{\boldsymbol{x}} = \arg\max_{x'} \langle \boldsymbol{v}, \boldsymbol{v}(\boldsymbol{x}') \rangle, \tag{35}$$

because $\|\boldsymbol{v}(\boldsymbol{x}')\|^2$ is constant. We project $\boldsymbol{v}$ to $\boldsymbol{v}(\hat{\boldsymbol{x}})$, which is an re-encoding of $\boldsymbol{v}$.

We can also perform decoding via the learned $\boldsymbol{u}(\boldsymbol{x}')$:

$$\hat{\boldsymbol{x}} = \arg\max_{x'} \langle \boldsymbol{v}, \boldsymbol{u}(\boldsymbol{x}') \rangle, \tag{36}$$

and re-encoding $\boldsymbol{v} \leftarrow \boldsymbol{v}(\hat{\boldsymbol{x}})$. For the above decoding, the heat map

$$\boldsymbol{h}(\boldsymbol{x}') = \langle \boldsymbol{v}, \boldsymbol{u}(\boldsymbol{x}') \rangle = \langle \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{u}(\boldsymbol{x}') \rangle + \langle \boldsymbol{\varepsilon}, \boldsymbol{u}(\boldsymbol{x}') \rangle = A(\boldsymbol{x}, \boldsymbol{x}') + \boldsymbol{e}(\boldsymbol{x}'), \tag{37}$$

where $\boldsymbol{e}(\boldsymbol{x}') = \langle \boldsymbol{\varepsilon}, \boldsymbol{u}(\boldsymbol{x}') \rangle \sim \mathcal{N}(0, \alpha^2 \|\boldsymbol{v}(\boldsymbol{x})\|^2 \|\boldsymbol{u}(\boldsymbol{x}')\|^2 / d)$. For $A(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2/(2\sigma^2)) = \langle \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{u}(\boldsymbol{x}') \rangle$, if σ^2 is small, $A(\boldsymbol{x}, \boldsymbol{x}')$ decreases to 0 quickly, i.e., if $\|\boldsymbol{x}' - \boldsymbol{x}\| > c$, then $A(\boldsymbol{x}, \boldsymbol{x}') < \exp(-c^2/(2\sigma^2))$, and the chance for the maximum of $\boldsymbol{h}(\boldsymbol{x}')$ to be achieved at an $\boldsymbol{x}'$ so that $\|\boldsymbol{x}' - \boldsymbol{x}\| > c$ can be very small. The above analysis also provides a justification for regularizing $\|\boldsymbol{u}(\boldsymbol{x}')\|^2$ in learning.

For error correction, we want to use small σ^2. However, for path planning, we need large σ^2 so that we can assess the adjacency as well as the change of the adjacency between the position on the path and the target position even if they are far apart.

In the experiments in the main text, we use Eq. (36) for decoding. In Fig. 9, we also show the results of path integration using Eq. (35) for decoding, whose performance is even better than Eq. (36). Especially the error would remain 0 over 300 time steps and 1,000 episodes using Eq. (35) with re-encoding. The advantage of (35) is that error correction is achieved within the grid cells system itself without interacting with the place cells.

A.7 Connection to continuous attractor neural network (CANN) defined on 2D torus

The CANN models [2, 6, 7, 30, 1] assume that the grid cells $\boldsymbol{v}(\boldsymbol{x}) = (v_i(\boldsymbol{x}), i = 1, ..., d)$ are placed on a finite 2D square lattice with periodic boundary condition, i.e., a 2D torus $\mathbb{T}$. If the lattice is $N \times N$, then $d = N^2$. Let $\boldsymbol{z} \in \mathbb{T}$ be the 2D coordinate of a pixel in $\mathbb{T}$, then each grid cell v_i is placed on a unique $\boldsymbol{z}_i \in \mathbb{T}$.

A CANN model hand-crafts the non-linear recurrent transformation $\boldsymbol{v}(\boldsymbol{x} + \Delta\boldsymbol{x}) = F(\boldsymbol{v}(\boldsymbol{x}), \Delta\boldsymbol{x})$ for some parametric form of F, and the coding manifold $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$ consists of the attracting fixed points of $F(\cdot, 0)$. In CANN, the recurrent connection weights between a pair of grid cells (v_i, v_j) only depend on the relative positions of the two cells on the 2D torus, $\boldsymbol{z}_i - \boldsymbol{z}_j$, i.e., the connection weights

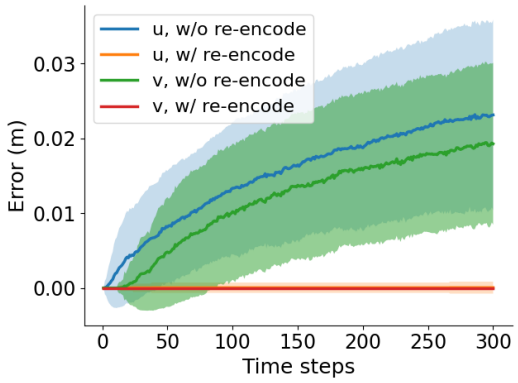


Figure 9: Path integration error over number of time steps. The mean and standard deviation band is computed over 1,000 episodes. "$\boldsymbol{v}$" means decoding by Eq. (35), and "$\boldsymbol{u}$" means decoding by Eq. (36). The squared domain is 1m $\times$ 1m.

are convolutional. Such a topographical arrangement may be physically realized on the 2D surface of the cortex in the brain, but it may also be the conceptual interpretation of the connection weights between the grid cells that are not necessarily placed on a physical 2D torus in the brain.

If we place the grid cells $\boldsymbol{v} = (v_i, i = 1, ..., d)$ on the $d = N \times N$ lattice of the 2D torus, either physically or conceptually, then their activities $\boldsymbol{v}(\boldsymbol{x}) = (v_i(\boldsymbol{x}), i = 1, ..., d)$ form an $N \times N$ "image" defined on the 2D torus. The pattern of the "image" may be a localized "bump", i.e., only a local subset of the pixels of the $N \times N$ lattice have non-zero activities. Suppose each self-position $\boldsymbol{x}$ of the agent can be mapped to a "bump" on the 2D torus centered at a corresponding $\boldsymbol{z} \in \mathbb{T}$. When the agent moves in the 2D physical space, i.e., when $\boldsymbol{x}$ changes to $\boldsymbol{x} + \Delta\boldsymbol{x}$, then the "bump" formed by $\boldsymbol{v}(\boldsymbol{x}) = (v_i(\boldsymbol{x}), i = 1, ..., d = N^2)$ moves on the 2D torus from $\boldsymbol{z}$ to $\boldsymbol{z} + \Delta\boldsymbol{z}$, while the shape of the "bump" remains the same. The connection weights of the CANN are hand-crafted so that the recurrent transformation of CANN realizes such a "mirroring" movement of the "bump".

If each displacement Δx of the agent in the 2D physical space can be mapped to a displacement $\Delta\boldsymbol{z}$ of the "bump" on the 2D torus $\mathbb{T}$, then the recurrent transformation of the CANN forms a representation of the 2D Euclidean group $\mathbb{R}^2$. If the local movement of the "bump" $\delta\boldsymbol{z}$ on the 2D torus is furthermore conformal to the local movement $\delta\boldsymbol{x}$ of the agent in the 2D physical space, then the local movement of the $d = N^2$ dimensional vector $\boldsymbol{v}(\boldsymbol{x}) = (v_i(\boldsymbol{x}), i = 1, ..., d = N^2)$ formed by the grid cells in the d-dimensional neural space, i.e., $\boldsymbol{v}(\boldsymbol{x} + \delta\boldsymbol{x}) - \boldsymbol{v}(\boldsymbol{x})$, is also conformal to the movement $\delta\boldsymbol{x}$ of the agent in the 2D physical space, and the isotropic scaling condition also holds.

In the above understanding, there are three types of movements. (1) The movement $\Delta\boldsymbol{x}$ of the agent in the 2D physical space $\mathbb{R}^2$. (2) The movement $\Delta\boldsymbol{z}$ of the "bump" on the $N \times N$ lattice of 2D torus $\mathbb{T}$. (3) The movement $\boldsymbol{v}(\boldsymbol{x} + \Delta\boldsymbol{x}) - \boldsymbol{v}(\boldsymbol{x})$ in the $d = N^2$-dimensional neural space.

Our model on either the general transformation or the linear transformation does not assume a 2D torus topography. In fact, no 2D topographical structure whatsoever is assumed in our model. The topographical arrangement is not part of our model. Instead, it may be treated as an implementation issue after the model is learned, i.e., how to arrange the grid cells physically on a 2D surface of cortex so that a pair of grid cells with strong connection weights are placed close to each other. It may also be treated as an interpretation issue after the model is learned, i.e., how to interpret the learned connection weights.

Even though our model does not make topographical assumptions, our linear transformation model appears to learn the torus topography automatically. Specifically, in our learned model, the response maps of the grid cells within each module are spatially shifted versions of the same hexagon periodic pattern. Therefore we can identify two directions in the 2D physical space that are $2\pi/3$ apart, so that $\boldsymbol{v}(\boldsymbol{x})$ rotates back to itself as $\boldsymbol{x}$ moves along these two directions for a certain distance. This implies that the codebook manifold $(\boldsymbol{v}(\boldsymbol{x}), \forall\boldsymbol{x})$ forms a 2D torus as assumed by CANN models. Moreover, the fact that the learned response maps of the grid cells within each module are spatially shifted versions of the same hexagon periodic pattern also agrees with the CANN model that moves the "bump" on the 2D torus by "mirroring" the motion in the 2D physical space. The learned hexagon

periodic patterns and the spatial shifts of the response maps may be related to the optimality of the hexagon grid in terms of sampling, interpolation and packing.

Even though the CANN model realizes the movement of the "bump" on the 2D torus by a non-linear recurrent model, such movement is a cyclic permutation of the activities of the grid cells, and the permutation can be realized by a permutation matrix, which is an orthogonal matrix. Thus the $\boldsymbol{v}(\boldsymbol{x})$ that satisfies the non-linear CANN model also satisfies our linear transformation model, where the linear rotation matrix is a cyclic permutation matrix.

The torus topology is hardly surprising, even for the general transformation model. The Lie group formed by $(F(\cdot, \Delta\boldsymbol{x}), \forall \Delta\boldsymbol{x})$ is abelian as it is a representation of the 2D additive Euclidean group $\mathbb{R}^2$. If a connected abelian Lie group is compact, then the group is automatically a torus. See [14].

Furthermore, if the scaling factor s is globally a constant for all $\boldsymbol{x}$, then the position embedding $(\boldsymbol{v}(\boldsymbol{x}), \forall \boldsymbol{x})$ is an isometric embedding up to a global scaling factor, and its intrinsic geometry remains Euclidean. It thus is a **flat torus**.

B Experiments

B.1 Implementation details

Monte Carlo samples. The expectations in loss terms are approximated by Monte Carlo samples. Here we detail the generation of Monte Carlo samples. For $(\boldsymbol{x}, \boldsymbol{x}')$ used in $L_0 = \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}'}[A(\boldsymbol{x}, \boldsymbol{x}') - \langle \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{u}(\boldsymbol{x}') \rangle]^2$, $\boldsymbol{x}$ is first sampled uniformly within the entire domain, and then the displacement $d\boldsymbol{x}$ between $\boldsymbol{x}$ and $\boldsymbol{x}'$ is sampled from a normal distribution $\mathcal{N}(0, \sigma^2 \boldsymbol{I}_2)$, where $\sigma = 0.48$. This is to ensure that nearby samples are given more emphasis. We let $\boldsymbol{x}' = \boldsymbol{x} + d\boldsymbol{x}$, and those pairs $(\boldsymbol{x}, \boldsymbol{x}')$ within the range of domain (i.e., 1m $\times$ 1m, 40×40 lattice) are kept as valid data. For $(\boldsymbol{x}, \Delta\boldsymbol{x})$ used in $L_1 = \mathbb{E}_{\boldsymbol{x},\Delta\boldsymbol{x}} |\boldsymbol{v}(\boldsymbol{x} + \Delta\boldsymbol{x}) - \exp(\boldsymbol{B}(\theta)\Delta r)\boldsymbol{v}(\boldsymbol{x})|^2$, $\Delta\boldsymbol{x}$ is sampled uniformly within a circular domain with radius equal to 3 grids and $(0, 0)$ as the center. Specifically, Δr^2, the squared length of $\Delta\boldsymbol{x}$, is sampled uniformly from $[0, 3]$ grids, and θ is sampled uniformly from $[0, 2\pi]$. We take the square root of the sampled Δr^2 as Δr and let $\Delta\boldsymbol{x} = (\Delta r \cos\theta, \Delta r \sin\theta)$. Then $\boldsymbol{x}$ is uniformly sampled from the region such that both $\boldsymbol{x}$ and $\boldsymbol{x} + \Delta\boldsymbol{x}$ are within the range of domain. For $(\theta, \Delta\theta)$ used in $L_2 = \sum_{k=1}^{K} \mathbb{E}_{\boldsymbol{x},\theta,\Delta\theta}[\|\boldsymbol{B}_k(\theta + \Delta\theta)\boldsymbol{v}_k(\boldsymbol{x})\| - \|\boldsymbol{B}_k(\theta)\boldsymbol{v}_k(\boldsymbol{x})\|]^2$, we uniformly sample θ and $\theta + \Delta\theta$ from discretized angles, i.e., 144 directions discretized for circle $[0, 2\pi]$. We will study sampling only small $\Delta\theta$ in the future.

Training details. The model is trained for 14,000 iterations. At each iteration, the samples are generated online. For the first 8,000 iterations, we update all learnable parameters, while for the following iterations, we fix the learned $\boldsymbol{v}(\boldsymbol{x})$ and update the other learnable parameters. The initial learning rate is set as 0.003 and is decreased by a factor of 0.5 every 500 iterations after 8,000 iterations. We use Adam [26] optimizer. The model is trained on a single Titan XP GPU. We apply the maximum batch size that can fit into the single GPU, which is 90,000. It takes about 3.5 hours to train the model on a single Titan XP GPU.

Baseline methods. In Table 1 of the main text, we compare the learned neurons with the ones from other two optimization-based learning methods [3, 34]. For [3], we run the code released by the authors (`https://github.com/deepmind/grid-cells`) to learn the model and compute gridness scores for the learned neurons. For [34], we use the pre-trained weights released by the authors (`https://github.com/ganguli-lab/grid-pattern-formation`) to get the learned neurons and compute the gridness scores. Both the code of [3] and pre-trained weights of [34] use Apache License V2.

Usage of data. In this paper, we mainly use simulated trajectories as training data, and thus we do not think that the data contain any personally identifiable information or offensive content. The only existing data we use is the pre-trained weights of the baseline method [34]. Under Apache License V2, we believe it is fully approved by the authors to use the pre-trained weights.

B.2 Learned patterns

Fig. 10 displays the autocorrelograms of learned patterns of $\boldsymbol{v}(\boldsymbol{x})$.

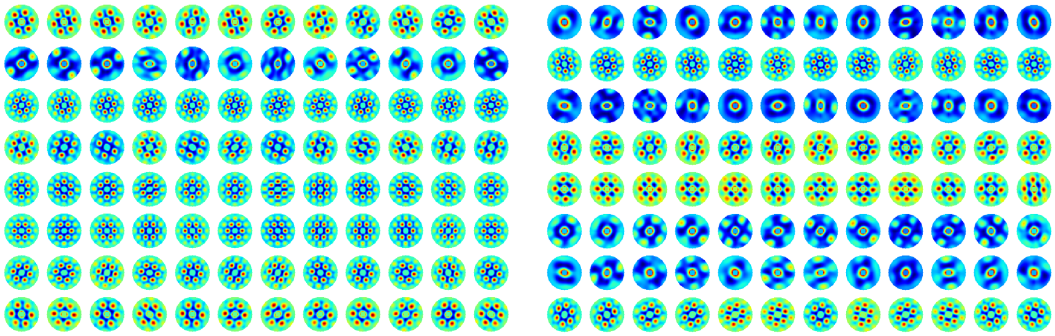


Figure 10: Autocorrelograms of the learned patterns of $\boldsymbol{v}(\boldsymbol{x})$.

Fig. 11 shows the learned patterns of $\boldsymbol{u}(\boldsymbol{x})$ with 16 blocks of 12 cells in each block. Regular hexagon patterns also emerge.

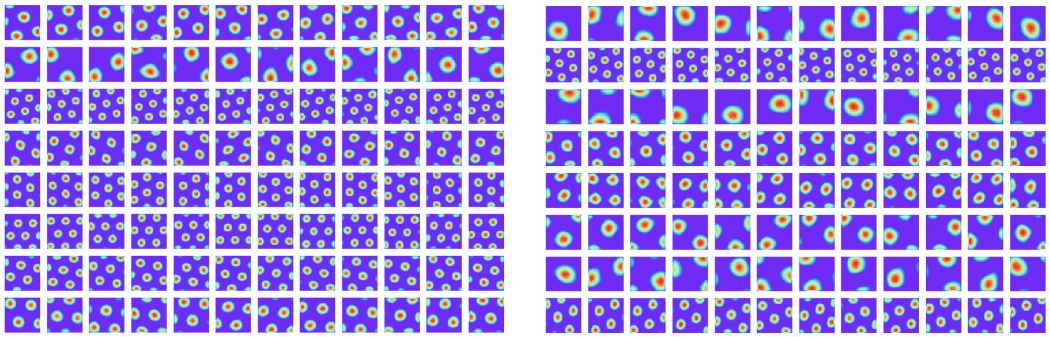


Figure 11: Learned patterns of $\boldsymbol{u}(\boldsymbol{x})$ with 16 blocks of size 12 cells in each block. Every row shows the learned patterns within the same block.

For learned firing patterns of $\boldsymbol{v}(\boldsymbol{x})$, we also display the histogram of grid orientations in Fig. 12, where we do not observe clear clusters.

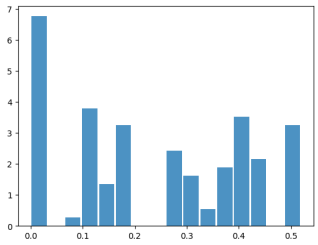


Figure 12: Histogram of grid orientations of the learned firing patterns of $\boldsymbol{v}(\boldsymbol{x})$.

In Fig. 13, we show the learned patterns of a block of $\boldsymbol{B}(\theta)$. Each element shows significant sine/cosine tuning over θ. For the other blocks, the patterns are all similar.

Gaussian kernel. Because $A(\boldsymbol{x}, \boldsymbol{x}')$ is a sharp Gaussian kernel, it contains a whole range of frequencies in the 2D Fourier domain. The learned response maps of the grid cells span a range of frequencies or scales too. Each module or block focuses on a certain frequency band, which corresponds to the metric of the module. We assume individual place field $A(\boldsymbol{x}, \boldsymbol{x}')$ to exhibit a Gaussian shape, rather than a Mexican-hat pattern (with balanced excitatory center and inhibitory

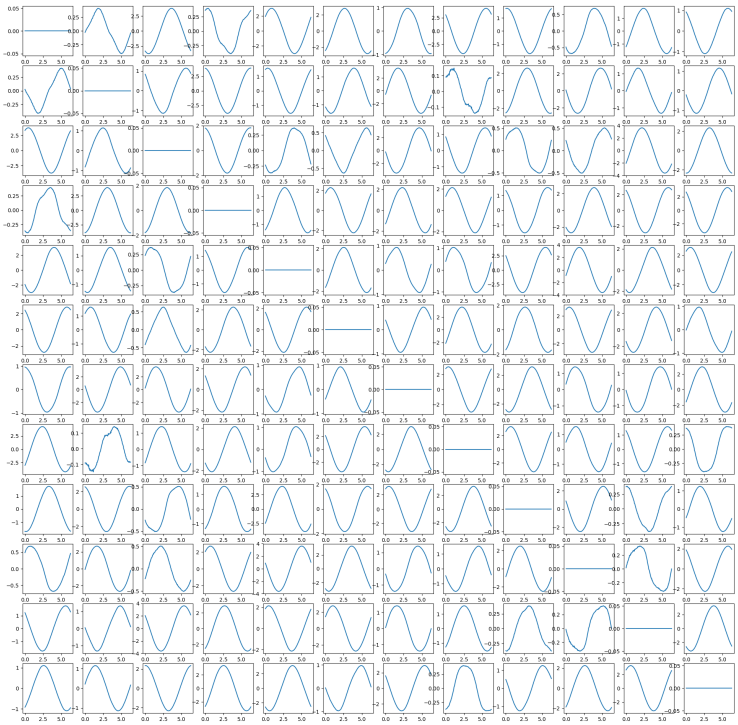


Figure 13: Learned patterns of a block of $\boldsymbol{B}(\theta)$. Each subfigure shows the value of an element in $\boldsymbol{B}(\theta)$ (vertical axis) over θ (horizontal axis).

surround) as assumed in previous basis expansion models [13, 34] of grid cells. The Mexican-hat or difference of Gaussians pattern occupies a ring in the 2D Fourier domain. It corresponds to a module in our model. But we use isotropic condition to enforce each module to be within a ring in the Fourier domain, and we use different modules to pave the whole Fourier domain.

B.3 Error correction

We begin by assessing the ability of error correction of the learned system following the setting in Proposition 1. Specifically, for a given location $\boldsymbol{x}$, suppose the neurons are perturbed by Gaussian noise: $\boldsymbol{v} = \boldsymbol{v}(x) + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \tau^2 \boldsymbol{I}_d)$ and $\tau^2 = \alpha^2(\|\boldsymbol{v}(\boldsymbol{x})\|^2/d)$, so that α^2 measures the variance of noise relative to the average magnitude of $(v_i(\boldsymbol{x})^2, i = 1, ..., d)$ and α measures the relative standard deviation. We infer the 2D position $\hat{\boldsymbol{x}}$ from $\boldsymbol{v}$ by $\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}'} \|\boldsymbol{v} - \boldsymbol{v}(\boldsymbol{x}')\|^2$. Fig. 14 displays the inference error over the relative standard deviation α of the added Gaussian noise. We also show the results using the learned $\boldsymbol{u}(\boldsymbol{x}')$ for inference (Eq. (36)). The system works remarkably well even if $\alpha = 2$.

We further assess the ability of error correction in long distance path integration. Specifically, along the way of path integration, at every time step t, two types of errors are introduced to $\boldsymbol{v}_t$: (1) Gaussian noise or (2) dropout masks, i.e., certain percentage of units are randomly set to zero. Fig. 15 summarizes the path integration performance with different levels of injected errors for $T = 100$, using $\boldsymbol{v}(\boldsymbol{x}')$ (Eq. (35)) or $\boldsymbol{u}(\boldsymbol{x}')$ (Eq. (36)) for decoding. The results show that re-encoding at each step helps error correction, especially for dropout masks. For Gaussian noise, even without decoding and re-encoding at each step, decoding at the final step alone is capable of removing much of the noise. Notably, with re-encoding, the path integration works well even if Gaussian noise with $\alpha = 1$ is added or 50% units are randomly dropped out at each step, indicating that the learned system is robust to different sources of errors.

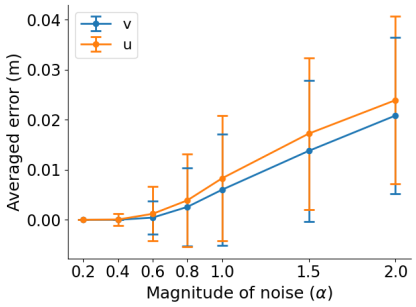


Figure 14: Error correction results following the setting in Proposition 1. The error bar stands for the standard deviation over 1,000 trials. "$\boldsymbol{v}$" means decoding by Eq. (35), and "$\boldsymbol{u}$" means decoding by Eq. (36). The squared domain is 1m $\times$ 1m.

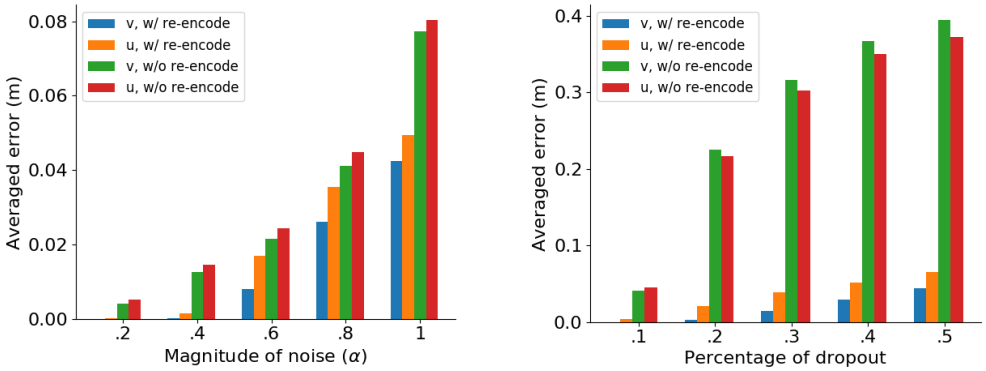


Figure 15: Path integration results with different levels of injected errors. *Left*: Gaussian noise. The magnitude of noise is measured using the average of the squared magnitudes of the units in $\boldsymbol{v}(\boldsymbol{x})$ as the reference. *Right*: dropout masks. Certain percentage of units are randomly set to zero at each step. "$\boldsymbol{v}$" means decoding by Eq. (35), and "$\boldsymbol{u}$" means decoding by Eq. (36). The squared domain is 1m $\times$ 1m.

B.4 Non-linear transformation model

We test our method with a non-linear transformation model:

$$F(\boldsymbol{v}(\boldsymbol{x}), \Delta r, \theta) = \text{ReLU}(\exp(\boldsymbol{B}(\theta)\Delta r)\boldsymbol{v}(\boldsymbol{x})), \tag{38}$$

where we insert $\text{ReLU}(a) = \max(0, a)$ into the linear transformation model.

We use numerical differentiation to define directional derivative

$$f_\theta(\boldsymbol{v}(\boldsymbol{x})) = [\boldsymbol{v}(\boldsymbol{x} + \delta\boldsymbol{x}) - \boldsymbol{v}(\boldsymbol{x})]/\delta r, \tag{39}$$

where $\delta\boldsymbol{x} = (\delta r \cos\theta, \delta r \sin\theta)$, with pre-defined δr. The reason for numerical differentiation is because the derivative of ReLU is an indicator function, which is not differentiable. $f_\theta(\boldsymbol{v}(\boldsymbol{x}))$ needs to be differentiable for minimizing the loss function (an alternative to numerical differentiation is to use sigmoid function to approximate the indicator function).

We continue to use the same loss function except with the above two changes. Interestingly, regular hexagon patterns continue to emerge (average gridness score 0.83, percentage of grid cells 70.21%). See Fig. 16 for the learned patterns of $\boldsymbol{v}(\boldsymbol{x})$.

B.5 Path planning

Our grid cells model can be applied to path planning. Specifically, according to [36], the adjacency kernel can be modeled by

$$A_\gamma(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t 1(\boldsymbol{x}_t = \boldsymbol{x}') | \boldsymbol{x}_0 = \boldsymbol{x}\right] = \langle \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{u}_\gamma(\boldsymbol{x}')\rangle, \tag{40}$$

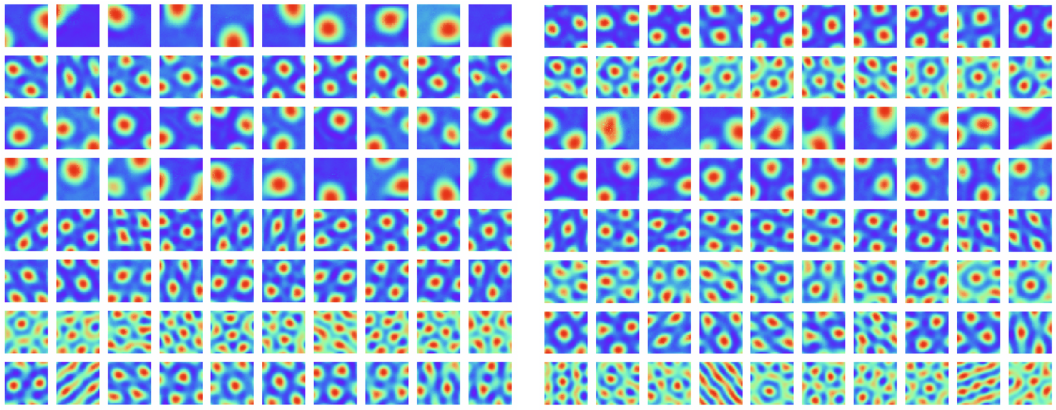


Figure 16: Learned patterns of $v(x)$ with the non-linear transformation model (Eq. (38)). Every row shows the learned patterns within the same block.

where γ is the discount factor that controls the temporal and spatial scales, $\mathbb{E}$ is with respect to a random walk exploration policy, and $1(\cdot)$ is the indicator function. For random walk in open field, $A_\gamma(x, x') \propto \exp(-\|x - x'\|^2/2\sigma_\gamma^2)$, where σ_γ^2 depends on γ.

To enable path planning, we need kernels of both big and small spatial scales to account for long and short distance planning respectively. To this end, we discretize γ into a finite list of scales, and learn a list of corresponding $u_\gamma(x')$ together with $v(x)$ and $B(\theta)$ using the loss function in Section 5 of the main text.

With the learned model, path planning can be accomplished by steepest ascent on the adjacency to the target position. Specifically, let $\hat{x}$ be the target or destination. Let $x^{(t)}$ be the current position in the path planning process, encoded by $v(x^{(t)})$. The agent plans the next displacement by steepest ascent on

$$A_\gamma(x^{(t)} + \Delta x, \hat{x}) = \langle v(x^{(t)} + \Delta x), u_\gamma(\hat{x}) \rangle = \langle M(\Delta x) v(x^{(t)}), u_\gamma(\hat{x}) \rangle, \tag{41}$$

over allowed Δx within a single step, where $M(\Delta x) = \exp(B(\theta)\Delta r)$, with $\Delta x = (\Delta r \cos\theta, \Delta r \sin\theta)$. We plan

$$\Delta x^{(t+1)} = \arg\max_{\Delta x} A_\gamma(x^{(t)} + \Delta x, \hat{x}), \tag{42}$$

and let $x^{(t+1)} = x^{(t)} + \Delta x^{(t+1)}$.

The scale γ is selected as the smallest one that satisfies $\max_{\Delta x} \langle M(\Delta x) v(x^{(t)}), u_\gamma(\hat{x}) \rangle > .2$. We can also use $\max_\gamma \max_{\Delta x} \langle M(\Delta x) v(x^{(t)}), u_\gamma(\hat{x}) \rangle$ for scale selection.

We test path planning in the open field environment. The model is first learned using a single-scale kernel function $A_\gamma(x, x') = \exp(-\|x - x'\|^2/2\sigma_\gamma^2)$ where $\sigma_\gamma = 0.07$. Then we assume a list of three scales: $\sigma_\gamma = [0.07, 0.14, 0.28]$ and learn the corresponding list of $u_\gamma(x')$. The pool of allowed displacements for a single step is defined as: dr can be 1 or 2 grids, while θ can be chosen from 200 discretized angles over $[0, 2\pi]$. Fig. 17 demonstrates several examples of path planning in the open field environment, where the agent is able to plan straight path to the target. When $x^{(t)}$ is far from the target, kernel with large σ_γ is chosen, and as $x^{(t)}$ approaches the target, the chosen kernel gradually switches to the one with small σ_γ. A planning episode is treated as a success if the distance between $x^{(t)}$ and target is smaller than 0.5 grid within 40 time steps. The agent achieves a success rate of 100% (tested for 10,000 episodes).

For a field with obstacles or rewards, we can learn the deformed $A_\gamma(x, x')$ and $(v(x), u_\gamma(x'))$ by temporal difference learning with a random walk exploration policy as suggested in [36]. After learning $A_\gamma(x, x')$ and $(v(x), u_\gamma(x'))$, we can continue to use Eq. (42) for path planning. We shall further study it in future work.

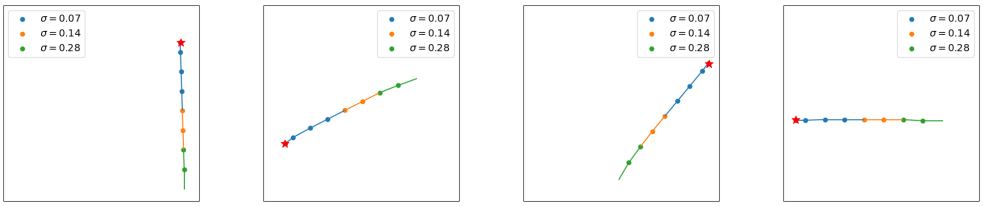


Figure 17: Examples of path planning results in an open field environment. The target is shown as a red star.

B.6 Integrating egocentric vision

When the agent moves in darkness, it can infer its self-position by integrating self-motion, as illustrated by our experiments on path integration. If there is visual input, the agent can infer its self-position (as well as head direction) from the visual image alone. We extend our grid cells model to study this problem of egocentric vision, which is important in computer vision.

Specifically, suppose the agent navigates in a 3D scene such as a room, and the height of the eye (or camera) remains fixed. Suppose at 2D self-position $\boldsymbol{x}$ and with head direction θ, the agent sees an image $\boldsymbol{I}$, which is called a posed image. We use the vector representation $\boldsymbol{v}(\boldsymbol{x})$ in our original grid cells model to represent the 2D self-position $\boldsymbol{x}$, and use another vector representation $\boldsymbol{h}(\theta)$ to represent the head direction θ. If the agent changes its head direction from θ to $\theta + \Delta\theta$, $\boldsymbol{h}(\theta)$ is transformed to

$$\boldsymbol{h}(\theta + \Delta\theta) = \exp(\boldsymbol{C}\Delta\theta)\boldsymbol{h}(\theta). \tag{43}$$

We assume that there are K modules or blocks in $\boldsymbol{h}(\theta)$ and $\boldsymbol{C}$ is skew-symmetric. This is similar to the transformation of $\boldsymbol{v}(\boldsymbol{x})$ in our grid cells model.

$(\boldsymbol{x}, \theta)$ is called the pose of the camera (or eye), and we call $(\boldsymbol{v}(\boldsymbol{x}), \boldsymbol{h}(\theta))$ the pose embedding.

To associate the pose embedding $(\boldsymbol{v}(\boldsymbol{x}), \boldsymbol{h}(\theta))$ with the posed image $\boldsymbol{I}$, we use a vector representation or scene embedding $\boldsymbol{s}$ to represent the 3D scene which is shared across different posed images of the same scene, and we learn a generator network G_β that maps the embeddings $\boldsymbol{s}$ and $(\boldsymbol{v}(\boldsymbol{x}), \boldsymbol{h}(\theta))$ to the posed image $\boldsymbol{I}$:

$$\boldsymbol{I} = G_\beta(\boldsymbol{s}, \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{h}(\theta)) + \varepsilon, \tag{44}$$

where the generator G_β is parametrized by a multi-layer deconvolutional neural network with parameters β, and ε is the residual error.

Given the above assumptions, we introduce two extra loss terms in addition to the loss function described in Section 5 of the main text.

$$L_3 = \sum_{k=1}^{K} \mathbb{E}_{\theta, \Delta\theta} \|\boldsymbol{h}_k(\theta + \Delta\theta) - \exp(\boldsymbol{C}_k \Delta\theta)\boldsymbol{h}_k(\theta)\|^2, \tag{45}$$

$$L_4 = \mathbb{E}\|\boldsymbol{I} - G_\beta(\boldsymbol{s}, \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{h}(\theta))\|^2. \tag{46}$$

L_3 is to model the head rotation, and L_4 is to model the generation of the posed image.

During training, we alternatively update $(G_\beta, \boldsymbol{s})$ and $(\boldsymbol{v}(\boldsymbol{x}), \boldsymbol{B}(\theta), \boldsymbol{u}(\boldsymbol{x}'), \boldsymbol{h}(\theta), \boldsymbol{C})$ by gradient descent on the overall loss function that is a linear combination of L_0, L_1 and L_2 in the main text, as well as L_3 and L_4 introduced above.

The learned model enables two useful applications:

(a) **Novel view synthesis**. Given an unseen pose $(\boldsymbol{x}, \theta)$, the model can predict the corresponding posed image by $G_\beta(\boldsymbol{s}, \boldsymbol{v}(\boldsymbol{x}), \boldsymbol{h}(\theta))$.

(b) **Inference of pose**, i.e., self-position $\boldsymbol{x}$ and head direction θ, from posed image $\boldsymbol{I}$ alone. Specifically, after training the model, we can learn an additional inference network F_ξ that maps an observed posed image $\boldsymbol{I}$ to its pose embedding $\boldsymbol{v}(\boldsymbol{x})$ and $\boldsymbol{h}(\theta)$. The inference network is learned by minimizing the ℓ_2 distance between the predicted and true pose embeddings: $\mathbb{E}\|(\boldsymbol{v}(\boldsymbol{x}), \boldsymbol{h}(\theta)) - F_\xi(\boldsymbol{I})\|^2$. Then

Table 2: Average error of pose inference.

	x_1	x_2	θ
Error	.0225m	.0230m	1.37°

given an unseen posed image $\boldsymbol{I}$, we can infer the pose by $\arg\min_{\boldsymbol{x},\theta} \|(\boldsymbol{v}(\boldsymbol{x}), \boldsymbol{h}(\theta)) - F_\xi(\boldsymbol{I})\|^2$. In this task, $F_\xi(\boldsymbol{I})$ is the estimate of $(\boldsymbol{v}(\boldsymbol{x}), \boldsymbol{h}(\theta))$, and it is likely that this estimate contains error. This error will translate to the error in the estimated $(\boldsymbol{x}, \theta)$. Thus our theoretical analysis of error translation in the main text is highly relevant, and the isotropic scaling condition is motivated by the analysis of error translation.

We conduct experiments on a dataset generated by the Gibson Environment [41], which provides tools for rendering images of different poses in 3D rooms. Specifically, we select 20 areas of size 2m $\times$ 2m from different rooms and render about 28k 64 $\times$ 64 RGB posed images for each area. The camera height is fixed and the camera can only rotate horizontally. The scene embedding vector $\boldsymbol{s}$ is of 512 dimensions. Both $\boldsymbol{v}(\boldsymbol{x})$ and $\boldsymbol{h}(\theta)$ are of 192 dimensions, partitioned into $K = 16$ modules.

Hexagon patterns still emerge in the learned $\boldsymbol{v}(\boldsymbol{x})$ (average gridness score 0.71). For novel view synthesis, we evaluate the performance on 374k testing posed images. The resulting peak signal-to-noise ratio (PSNR) between synthesized images and ground truth images is 25.17, indicating that the model can generate reasonable unseen posed images. Fig. 18 demonstrates several examples of the novel view synthesis results.

Figure 18: Examples of synthesizing novel views. *Left*: Ground truth unseen posed images. *Right*: synthesized unseen posed images.

For inference of pose (self-position $\boldsymbol{x} = (x_1, x_2)$ and head direction θ), we evaluate the performance on the same 374k testing posed images and report the average inference error in Table 2. The estimates are reasonably accurate.

B.7 Ablation studies

Isotropic scaling condition is necessary for hexagon grid patterns. A natural question is whether the isotropic scaling condition (condition 2) is important for learning hexagon grid patterns. To verify this, we learn the model by removing the loss term L_2 (Eq. (19) in the main text) from the loss function, which constrains the model to meet condition 2. As shown in Fig. 19, more strip-like patterns emerge without L_2, indicating that condition 2 is important for hexagon grid patterns to emerge.

Assumption of $\boldsymbol{u}(\boldsymbol{x}') \geq 0$ is not necessary for hexagon grid patterns. During training, we make an assumption of $\boldsymbol{u}(\boldsymbol{x}') \geq 0$ to make sure the connections from grid cells to place cells are excitatory [44, 32]. However, we want to emphasize this is not a key assumption in our model. Fig. 20 demonstrates the learned neurons in the network without assuming $\boldsymbol{u}(\boldsymbol{x}') \geq 0$, where hexagonal grid firing patterns also emerge. The average gridness score is 0.82 and the percentage of grid cells is 87.50%. However, the grid activations can be either positive/excitatory (in red color) or negative/inhibitory (in blue color).

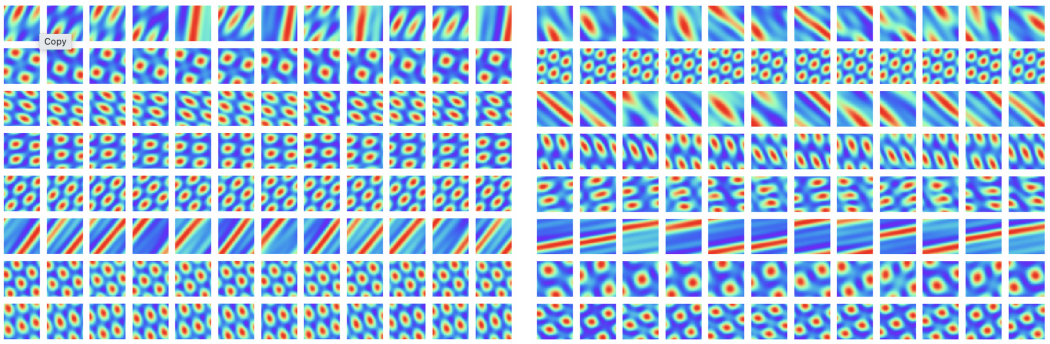


Figure 19: Learned neurons without loss term L_2, which is the constraint on isotropic scaling condition. More strip-like firing patterns emerge.

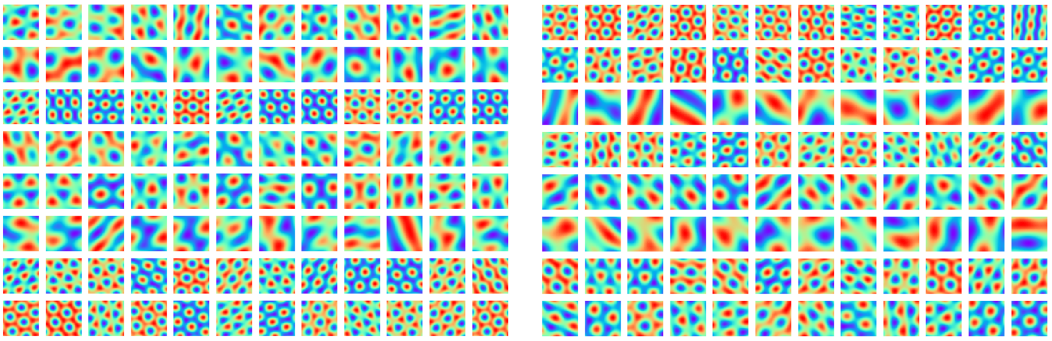


Figure 20: Learned neurons without the assumption of $\boldsymbol{u}(\boldsymbol{x}') \geq 0$. Hexagonal grid firing patterns also emerge, with the grid activations being either positive/excitatory (in red color) or negative/inhibitory (in blue color).

Skew-symmetric assumption of $\boldsymbol{B}(\theta)$ is not important for hexagon grid patterns. To make the linear transformation a rotation, we have assumed that $\boldsymbol{B}(\theta)$ is skew-symmetric, i.e., $\boldsymbol{B}(\theta) = -\boldsymbol{B}(\theta)^\top$. Nonetheless, this assumption is not important for the emergence of hexagon grid patterns. Fig. 21 demonstrates the learned neurons without assuming that $\boldsymbol{B}(\theta)$ is skew-symmetric. Hexagon grid firing patterns emerge in most of the neurons, with only one block of square grid firing patterns.

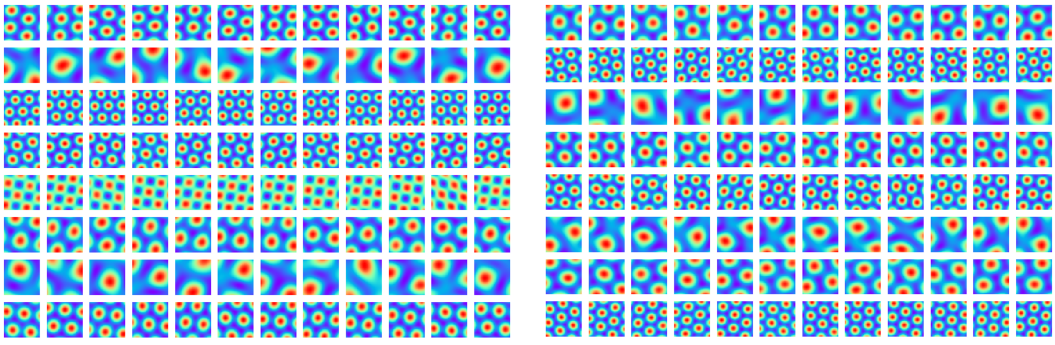
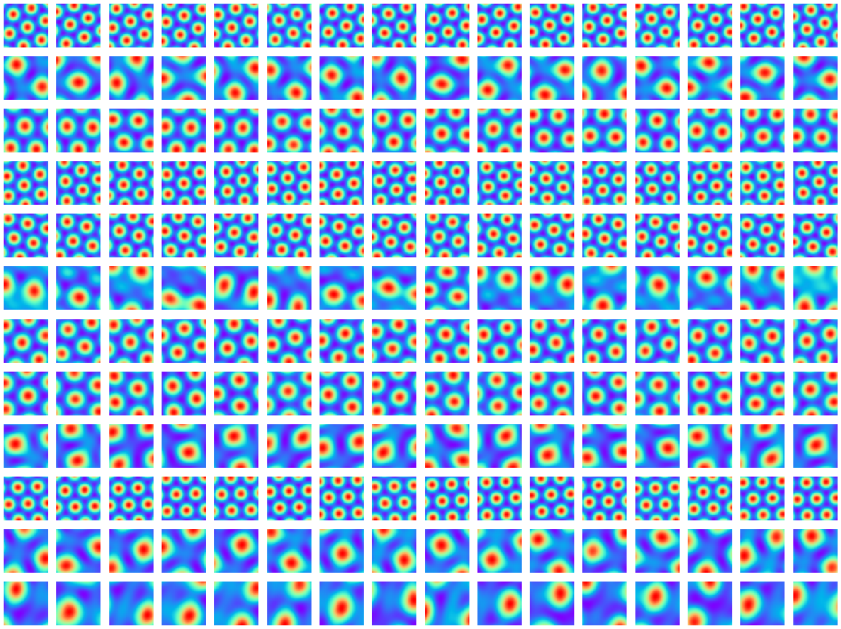


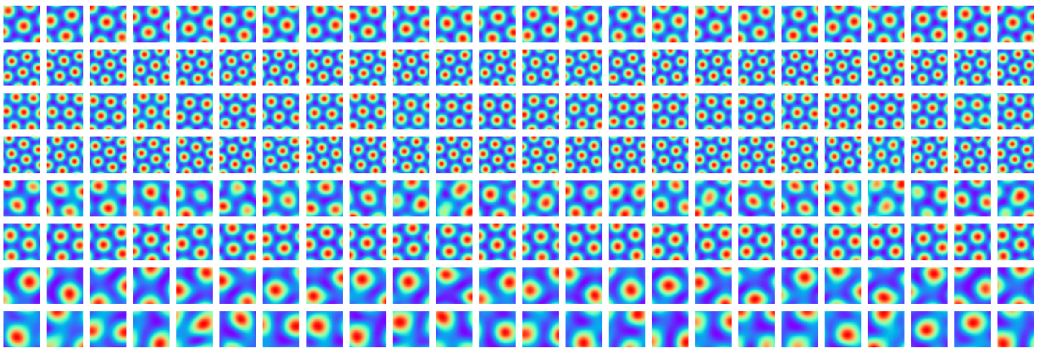
Figure 21: Learned neurons without skew-symmetric assumption of $\boldsymbol{B}(\theta)$. Hexagonal grid firing patterns emerge in most of the neurons, with a block of square grid firing patterns.

Number and sizes of blocks do not matter. It is worthwhile to mention that the emergence of hexagonal grid firing patterns in the learned neurons are not due to specific design of the block size or the number of blocks. Fig. 22 visualizes the learned neurons by fixing the total number of neurons at 192 and altering the block size and number of blocks. Hexagon patterns emerge in all the settings.

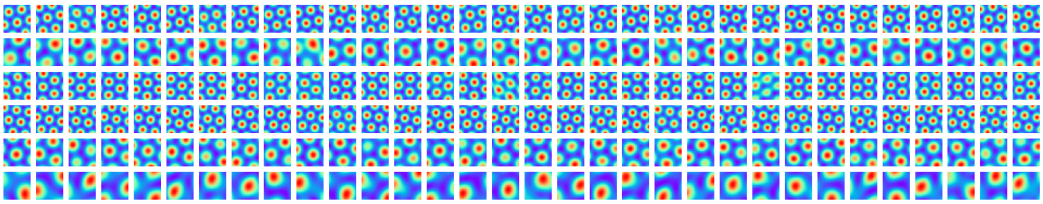
Multiple blocks or modules are necessary for learning grid patterns of multiple scales. We further try to fully remove the assumption of blocks or modules; i.e., we learn a single block of $\boldsymbol{B}(\boldsymbol{\theta})$. Fig. 23 shows the learned neurons and the corresponding autocorrelograms. All the learned neurons share similar large scales, which indicates that the high frequency part of $A(\boldsymbol{x}, \boldsymbol{x}')$ may not be fitted very well.



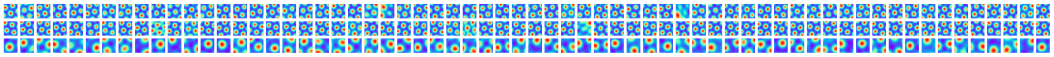
(a) Block size $= 16$



(b) Block size $= 24$



(c) Block size $= 32$



(d) Block size $= 64$

Figure 22: Learned patterns of $\boldsymbol{v}(\boldsymbol{x})$ with different block sizes. The total number of units is fixed at 192. Every row shows the learned patterns within the same block.

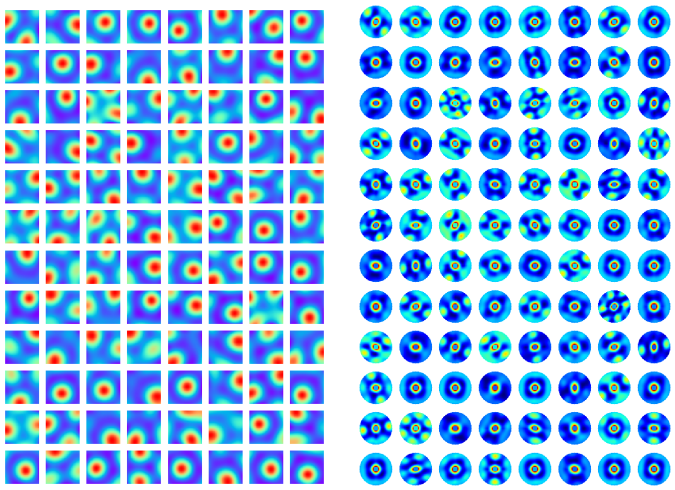


Figure 23: *Left*: learned neurons with a single block of $\boldsymbol{B}(\theta)$. The firing patterns has a single large scale, meaning that the high frequency part of $A(\boldsymbol{x}, \boldsymbol{x}')$ is not fitted very well. *Right*: autocorrelograms of the learned neurons. Some exhibit clear hexagon grid patterns, while the other do not, probably because the scale of those grid patterns are beyond the scope of the whole area.