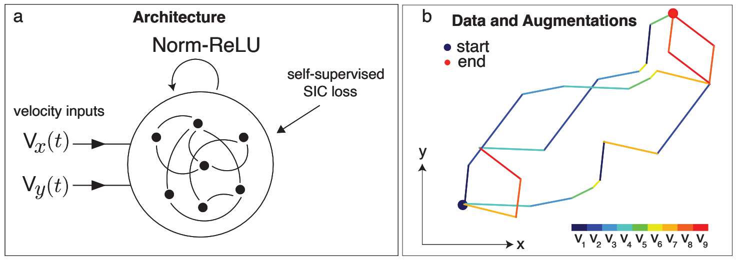

Figure 8: **A trained recurrent neural network learns multiple modules of grid cells.** (a) The architecture of the recurrent neural network used. Inputs are 2D Cartesian velocities $\boldsymbol{v}_t \in \mathbb{R}^2$ and the non-linearity is Norm-ReLU. No positional readout exists. (b) Input trajectory velocities are drawn i.i.d. from a uniform distribution, then randomly permuted to create a batch; 3 trajectories shown.

A Experimental Details

Architecture and training data + augmentations

Our code was implemented in PyTorch [39] and PyTorch Lightning [18]. Hyperparameters for our experiments are listed in Table 1. Our code will be made publicly available upon publication.

Hyperparameters	**Values**
Batch size	130
Trajectory length	60
Velocity sampling distribution	$\boldsymbol{v}_t \in \mathbb{R}^2 \sim_{i.i.d.}$ Uniform$^2(-0.15, 0.15)$ meters
RNN nonlinearity	$Norm(ReLU(\cdot))$
Number of RNN units	128
Number of MLP layers	3
Spatial length scale σ_x	0.05 meters
Neural length scale σ_g	0.4
Separation loss coefficient λ_{Sep}	1.0
Invariance loss coefficient λ_{Inv}	0.1
Capacity loss coefficient λ_{Cap}	0.5
Optimizer	AdamW [32]
Optimizer scheduler	Reduce Learning Rate on Plateau
Learning rate	2e-5
Gradient clip value	0.1
Weight decay	None
Accumulate gradient batches	2
Number of gradient descent steps	2e6

Table 1: Hyperparameters used for training the networks.

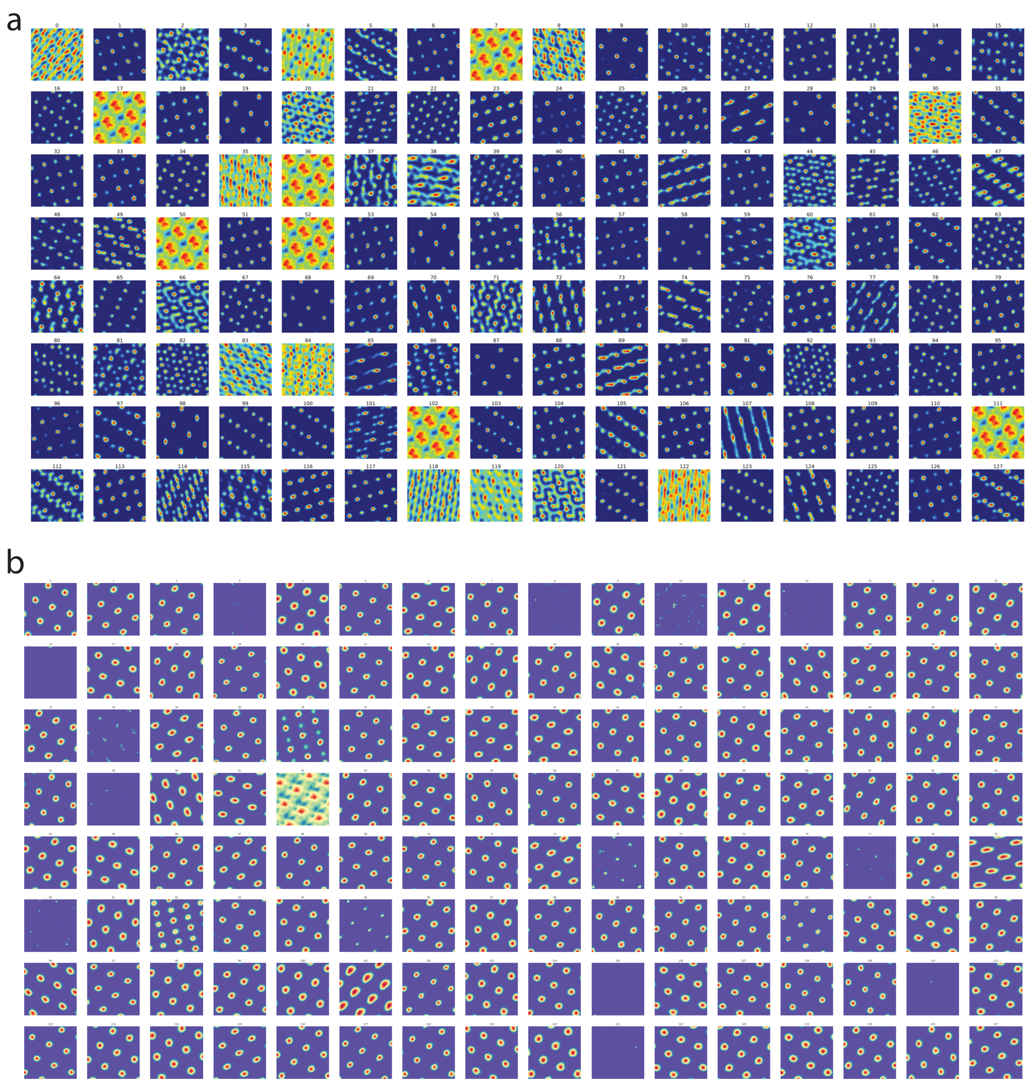

Figure 9: All 128 ratemaps evaluated on trajectories inside a 2m box. (a) Ratemaps from the corresponding to Fig. 4 (b) Ratemaps corresponding to the run in Fig.6

B All Ratemaps

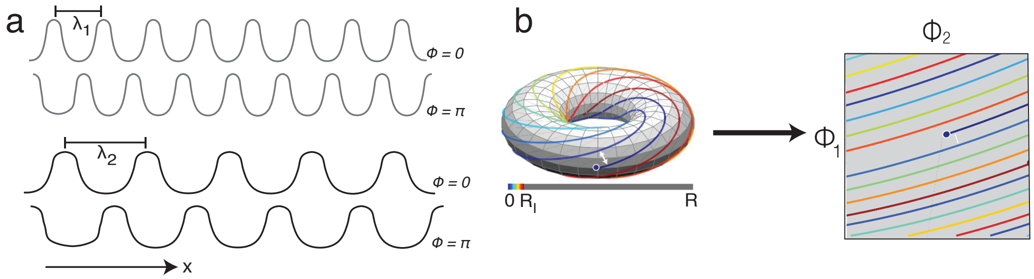

Figure 10: (a) 2 Example cells from 2 modules, with preferred phase 0 and π. (b) Visualizing the state space defined by $\{\phi^1, \phi^2\}$ as a torus (left) and on a square with periodic boundary conditions (right), which an equivalent construction of a torus.

C Construction of the grid code

To explain the structure of the grid code, we consider idealized tuning curves in 1d.

Each cell i is defined by its periodicity $\lambda^\alpha \in \mathcal{R}$ and preferred phase $\phi_i \in S^1$. All cells in the same module α have the same periodicity and uniformly tile all allowed phases. For position $x \in \mathcal{R}$,

$$r_i^\alpha(x) = R_{\text{max}}\text{ReLU}\left[\cos\left(\frac{2\pi}{\lambda_\alpha}x + \phi_i\right)\right] \tag{11}$$

The tuning curves corresponding to this module can be seen in Fig. 10a.

For this module, we can define $\phi^\alpha(x) = \dfrac{2\pi}{\lambda_\alpha}x$ modulo 2π. Here $\phi^\alpha \in S^1$.

So the firing rate can now be written as

$$r_i^\alpha(x) = R_{\text{max}}\text{ReLU}\left[\cos\left(\phi^\alpha(x) + \phi_i\right)\right] \tag{12}$$

All information about the current state of the module is encoded in the single variable ϕ^α. Thus the set of phases $\{\phi^\alpha\}_\alpha \in S^1 \times ... \times S^1$ uniquely define the coding states of the set of grid modules.

For 2 modules, defined by $\{\phi^1, \phi^2\}$, these states can be visualized as being on a torus $S^1 \times S^1$, Fig.10b.