Supplementary Materials for

Connecting multiple spatial scales to decode the population activity of grid cells

Martin Stemmler, Alexander Mathis, Andreas V. M. Herz

Published 18 December 2015, Sci. Adv. 1, e1500816 (2015)
DOI: 10.1126/science.1500816

The PDF file includes:

Text
Fig. S1. Hierarchical self-similar scales enable error correction.
Fig. S2. Noninteger scale ratios imply that the decoding algorithm must be able to rotate population vectors.
Fig. S3. The lengths of the population vectors $\hat{K}_i$ along the hexagonal grid’s axes are correlated.
Fig. S4. Possible distributed representations of the position vector estimate $\hat{\mu}$ of eq. S6 across a population of readout neurons.
Fig. S5. Comparison of different decoding schemes for a single module in two dimensions.
Fig. S6. Comparison of different decoding schemes for multiple modules in two dimensions.
Fig. S7. If the grid scales are not organized into discrete modules, population vector decoding is no longer straightforward.
Fig. S8. Continuum decoding requires multiple population vectors.
References (46–50)
1 Supplementary Text

1.1 Decoding from a single module – one spatial scale

Consider a single module with $M$ neurons. For large $M$, either positioning the $c_j$’s equidistantly or distributing them uniformly ensures that the sum $\sum_j \Omega(\bar{x} - \bar{c}_j)$ is nearly independent of the position $\bar{x}$. Therefore, the multiplicative factor $\exp \left( \sum_j \Omega(\bar{x} - \bar{c}_j) \right)$ in Eq. (1) of the main text can be approximated by a constant. The conditional probability for the position of the animal within the unit cell, given the neuronal spike counts $n_i$, becomes

$$P(\bar{x}|n_1, \ldots, n_M) = C \exp \left[ \frac{\kappa}{3} \sum_{j=1}^{M} \sum_{l=1}^{3} n_j \cos \left( \omega \vec{k}_l \cdot (\bar{x} - \bar{c}_j) \right) \right], \quad (S1)$$

where $C$ is a normalization constant. This expression can be written more simply as

$$P(\bar{x}|n_1, \ldots, n_M) = C' \exp \left[ \sum_{l=1}^{3} \hat{\kappa}_l \cos \left( \omega \vec{k}_l \cdot (\bar{x} - \bar{\mu}_l) \right) \right] \quad (S2)$$

with

$$\mu_l = \vec{k}_l \cdot \bar{\mu}_l = \omega^{-1} \arg \left( \sum_{j=1}^{M} n_j \exp(i \omega \vec{k}_l \cdot \bar{c}_j) \right) \quad (S3)$$

and

$$\hat{\kappa}_l = \kappa/3 \sum_{j=1}^{M} n_j \cos \left( \omega \vec{k}_l \cdot (\bar{\mu}_l - \bar{c}_j) \right). \quad (S4)$$

The value of $\mu_l$ represents the normalized population vector projected onto $\vec{k}_l$, whereas $\hat{\kappa}_l$ is the length of that population vector. The Fisher information along $\vec{k}_l$, which measures the local resolution, is directly proportional to $\hat{\kappa}_l$ (10).

To determine the most likely solution, we need to find the maximum of Eq. (S2), which is equivalent to maximizing $\sum_{l=1}^{3} \hat{\kappa}_l \cos \left( \omega \vec{k}_l \cdot (\bar{x} - \bar{\mu}_l) \right)$. Suppose that the maximum occurs
at $\bar{x} = \bar{\mu}$ and rewrite $\bar{\mu}_l = \bar{\mu} + \bar{\delta}_l$. Now assume that $||\omega \bar{k}_l \cdot \bar{\delta}_l|| \ll 1$, which will hold when either $M$ or $n_{max}$ are large. We can then solve the maximization problem explicitly. The solution is

$$\bar{\mu} = \frac{1}{2C''} \left( \sum_{m,l=1}^{3} \hat{\kappa}_m \hat{\kappa}_l \mu_l \cos(\phi_l) \right) - \frac{1}{2C''} \left( \sum_{m,l=1}^{3} \hat{\kappa}_m \hat{\kappa}_l \mu_l \sin(\phi_l) \right)$$

(S5)

with $C'' = \sum_{3 \ge l > m \ge 1} \hat{\kappa}_l \hat{\kappa}_m \sin(\phi_l - \phi_m)^2$. Now, the $\hat{\kappa}_l$ are random variables, which typically have high variance; unlike the spike counts $n_j$, though, the $\hat{\kappa}_l$ are not independent, but are correlated (see Fig. S3). These correlations lead us to consider the case in which all $\hat{\kappa}_l$ are equal. In this case, the expression in Eq. (S5) simplifies to

$$\bar{\mu} = \frac{2}{3} \sum_{l=1}^{3} \mu_l \bar{k}_l,$$

(S6)

which corresponds geometrically to drawing three lines perpendicular to $\bar{k}_l$ at the points $\mu_l \bar{k}_l$, computing the intersections between pairs of these, and averaging. Indeed, if one parametrizes these lines as $\mu_l \bar{k}_l + t \bar{p}_l$, with $\bar{p}_l \perp \bar{k}_l$ and $t$ an independent variable, the three pairwise intersections occur at

$$x^{(l,n)}_{\text{cross}} = \frac{-\sin(\phi_n)\mu_l + \sin(\phi_l)\mu_n}{\sin(\phi_l - \phi_n)},$$

(S7)

$$y^{(l,n)}_{\text{cross}} = \frac{\cos(\phi_n)\mu_l - \cos(\phi_l)\mu_n}{\sin(\phi_l - \phi_n)}.$$  

(S8)

Averaging Eqs. (S7)–(S8) across the three intersections yields Eq. (S6). In the regular hexagonal map, the three crossing points form an equilateral triangle, the squared edge of which measures

$$\frac{4}{3}(\mu_2 - \mu_1 - \mu_3)^2.$$

As the encoding is periodic, we can always add or subtract from any of the $\mu_l$ the short edge length of the rectangle used for computing the population vector. The correct choice entails
making the equilateral triangle formed by the three intersections as small as possible. Positions and distances between points in space are always taken modulo the unit cell $L$.

A quantitative comparison of the decoding precision using Eq. S5 or the simplified approach based on Eq. S6 is shown in Figs. S5 and S6. As illustrated in these figures, the population-vector based methods fare well compared to the computationally expensive approach of explicitly representing the posterior probability (Eq. (7) in the main text) and numerically maximizing it.

Equations (S3) and (S6) underlie the biologically plausible decoder we propose. The first formula is identical to the population vector described by Georgopolous et al. for decoding arm movements from motor cortex (11); only in our case, the encoded variable $\mathbf{z}$ is not intrinsically periodic. The individual components $\mu_l$ along the directions $\mathbf{k}_l$ can be obtained by weighting the neuronal activity for each neuron $n_j$ by $w_2 := \cos(\omega \mathbf{k}_l \cdot \mathbf{c}_j)$ and $w_{2j+1} := \sin(\omega \mathbf{k}_l \cdot \mathbf{c}_j)$. Since $\exp(i\phi) = \cos(\phi) + i\sin(\phi)$, this would allow a read-out mechanism – be it at the cellular, sub-cellular or network-level – to calculate $\mu_l$ as described in Eq. (S3). Figure 4A of the main text shows how decoding the position in 2D corresponds to finding a trilateral intersection, but trigonometry allows one to reduce the problem to the extremely simple algebraic formula of Eq. (S6): the three components $\mu_1$, $\mu_2$, and $\mu_3$ are summed linearly to yield the most likely position within the unit cell.

Just as one can ‘turn the dial’ of a clock to measure time in relation to a different starting point or ‘time zero’, a modified read-out can decode the vector relative to a different starting or goal position. In other words, the animal can use its estimated position in allocentric coordinates to calculate a homing vector in egocentric coordinates, as would be required for planning its movements. Indeed, multiple read-outs could simultaneously calculate vectors to different objects of interest from the same neuronal activity vector $\mathbf{n}$ (see Fig. 4B). Moving the origin of the coordinate system adds a phase offset $\mathbf{k}_l \cdot \mathbf{z}$; weighting the spike count $n_j$
by the angle $\vec{k}_l \cdot (\vec{c}_j - \vec{z})$ makes the population vector point to the new origin $\vec{z}$. Because $\exp(i(\varphi + \psi)) = \exp(i\varphi) \exp(i\psi)$, a phase shift is equivalent to a rotation of the population vector.

### 1.2 Decoding from a single module – one spatial scale with elliptic grid patterns

We now show that ellipticity, as observed by (13), can be readily incorporated into the readout we present in this paper. Suppose that the field centers lie on an ellipse with a semi-major axis $a$ and semi-minor axis $b$, with $a > 1 > b > 0$, such that the major axis of the ellipse coincides with a lattice vector. To model ellipticity, the wave vectors in Eq. (6) (main text) are rescaled to $\vec{k}_l = \{\cos(\phi_l)/a, \sin(\phi_l)/b\}$; here, $\phi_l$ is still a multiple of $\pi/3$ and the major axis has been set to the horizontal x-axis (but this is arbitrary). Ellipticity breaks the rotational symmetry, but maintains a (local) six-fold axial symmetry of the pattern, which implies that the construction shown in Fig. 4B for decoding $\vec{\mu}_t$ from a set of three population vectors remains feasible. Extending the calculation of Eq. (S5) to the elliptic case, Eq. (S6) becomes

$$\hat{\mu}_x = \frac{2}{3} \sum_l a \cos(\phi_l) \mu_l \quad \text{and} \quad \hat{\mu}_y = \frac{2}{3} \sum_l b \sin(\phi_l) \mu_l,$$

where the population vectors $\mu_l$ in this case are normalized to $[0, 1]$, which makes the scaling with $a$ and $b$ explicit in the equation.

Thus, even for ‘elliptic grid cells’, like the ones described by (5, 13), the proposed readout can calculate the position. Given the elliptical eccentricity $\epsilon$ and grid spacing $\lambda$, which determine the semi-major and semi-minor axes $a$ and $b$ (13), the population vectors in Eq. (S9) yield reliable position estimates.
1.3 Distributed representations of the homing vector

Eq. (S6) describes a two-dimensional quantity, namely the vector between the current position and the origin of the coordinate system, the latter of which we interpret as a goal location. Many measures of neural activity, such as the firing rate of a neuron, are intrinsically one-dimensional, however. So how would read-out neurons represent the computation of Eq. (S6)?

One possible answer is that the representation of \( \vec{\mu} \) from Eq. (S6) is distributed across different neurons. In general, the response of a read-out neuron might represent a linear combination \( \sum_l w_l \mu_l \) of the population vectors \( \mu_l \). One particular linear representation stands out, however, as it consists of the coordinates of \( \vec{\mu} \) in some basis \( \vec{e}_r \). In fact, the basis set \( \vec{e}_r \), which we assume to be linearly independent, might well be overcomplete, i.e., contain many more vectors than just the minimum two required to represent Euclidean space. Imagine we have a set of \( R \) read-out neurons, each associated with a vector \( \vec{e}_r \). The response \( r \) of a neuron within this set will then be proportional to

\[
 r \propto \frac{1}{3\pi} \sum_l \arg \left( \sum_{j=1}^{M} n_j \exp(i \omega \vec{k}_l \cdot \vec{c}_j) \right) \vec{k}_l \cdot \vec{e}_r.
\]

If \( r \) were a firing rate, which cannot be negative, one would set \( r \) to zero whenever the right-hand side of the equation above becomes negative; otherwise, \( r \) is linearly proportional to the right-hand side. Such responses are often called threshold-linear. Each of these read-out neurons will then represent the (threshold-linear) projection of Eq. (S6) onto its “preferred direction” vector \( \vec{e}_r \).

The argument function has a branch cut, at which the value of \( \arg(x) \) jumps by \( 2\pi \). If one places the branch cut at the angular coordinate \( \pm \pi \), as is standard convention, then a threshold-linear response \( r \) increases with distance from the goal. Alternatively, one could envision a case in which the highest response \( r \) is observed at the origin, with the response decreasing the farther away the animal is from the goal. Such behavior is observed when one rotates all...
population vectors by $\exp(i\pi)$ before taking the argument.

The resulting spatial activity maps consist of wedge-shaped ramps of activity, which either rise or fall towards the goal location (Fig. S4).

1.4 Decoding from nested modules – multiple scales

We now extend population vector decoding to multiple modules with different grid sizes (spatial scales) but co-aligned grid orientations. Each module is assumed to be independent. Define the ratio of scales as $s_m = \lambda_0/\lambda_m$ and let the $m$-th module have $M_m$ neurons. The key assumption of the theory is that the modules are self-similar: as the grid scale decreases, the firing fields also shrink commensurately. Self-similarity makes error correction possible (see main text and Fig. S1). Ignoring possible translations and rotations of the lattices, self-similarity implies

$$\Omega^{(m)}(\vec{x}) = \Omega^{(0)}(s_m \vec{x}).$$

Provided that the tuning curves cover the unit cell of the spatial lattice densely and (at least approximately) uniformly within each module, then the posterior (Eq. (7) in the main text) can be written as

$$P(\vec{x}|n) = C' \exp \left[ \frac{\kappa}{3} \sum_{m} \sum_{j=1}^{M_m} \sum_{l=1}^{3} n_j \cos \left( \omega s_m \vec{k}_l \cdot (\vec{x} - \vec{c}_j) \right) \right]$$ (S10)

for some constant $C'$.

Do self-similar tuning curves imply that the posterior $P(\vec{x}|n)$ is also composed of self-similar elements? In the strict sense, this is only true for the average logarithm of the posterior probability:

$$\langle \ln P(\vec{x}|n) \rangle = \sum_{m} \sum_{j=1}^{M_m} \Omega^{(m)}_j(\vec{x}) \left\{ \ln \left[ \Omega^{(m)}_j(\vec{x}) \right] - 1 \right\} + C$$ (S11)

where $C$ is a constant and the average is taken over all realizations of the spike-count vector $n$ elicited at a particular position $\vec{x}$, which we set to $\vec{x} = 0$ without loss of generality. The function $g^{(m)}(\vec{x})$ has the same scaling property as $\Omega^{(m)}(\vec{x})$, i.e., $g^{(m)}(\vec{x}) = g^{(0)}(s_m \vec{x})$. Within
each module, the population vectors of neuronal activity can be used to find the approximate maximum of $P(\vec{x}|n)$ within the fundamental domain at each length scale, as outlined above. Call these maxima $\vec{\delta}_m$; if the true position is $\vec{x} = 0$, then $\vec{\delta}_m$ is also the error at each individual scale.

Approximate the log-probability to second order as

$$\ln P(\vec{x}|n) \approx \sum_m M_m \frac{1}{2} \left( \vec{x} - \vec{\delta}_m \right)^T D^2 g^{(m)}(\vec{x}) \Big|_{\vec{x}=0} \left( \vec{x} - \vec{\delta}_m \right) + \tilde{C},$$

where $\tilde{C}$ is some constant. The Hessian matrices $D^2 g^{(m)}(\vec{x})$ control the curvature around each maximum $\vec{\delta}_m$. Note that the equation above makes several assumptions: we replace the true curvature with the curvature averaged over different realizations of $n$, as given by Eq. (S11); moreover, the peak of each $g^{(m)}(\vec{x})$ occurs at $\vec{x} = 0$, so we evaluate the Hessian at that point, and not at $\vec{x} = \vec{\delta}_m$. The resulting expression shown above is simple, which will allow us to find the maximum likelihood solution for the combined ensemble of all modules. We can use the scaling property to write

$$\ln P(\vec{x}|n) = \sum_m \frac{M_m}{2} s_m^2 \left( \vec{y} - \vec{\chi}_m \right)^T D^2 g^{(0)}(s_m \vec{x}) \Big|_{\vec{x}=0} \left( \vec{y} - \vec{\chi}_m \right) + \tilde{C},$$

$D^2 g^{(0)}(\vec{x})\Big|_{\vec{x}=0}$ is a symmetric matrix. Assume this matrix is not degenerate, so that it admits a decomposition

$$D^2 g^{(0)} = P^T \Lambda P,$$

where $\Lambda$ is diagonal, $P$ contains the normalized eigenvectors of $D^2 g^{(0)}$, and $P^T P = I$. Call $\vec{y} = P\vec{x}_{ML}$ and $\vec{\chi}_m = P\vec{\delta}_m$. Now find the $\vec{y}$ that maximizes

$$\sum_m \frac{M_m s_m^2}{2} (\vec{y} - \vec{\chi}_m)^T \Lambda (\vec{y} - \vec{\chi}_m).$$
The solution for $L$ modules is $\vec{y} = \sum_{m=0}^{L-1} \frac{M_m s_m^2 \vec{x}_m}{\sum_{m=0}^{L-1} M_m s_m^2}$. Using $\vec{x}_{ML} = P^T \vec{y}$ and $\vec{\delta} = P^T \vec{x}_m$, this is equivalent to

$$\vec{x}_{ML} = \frac{\sum_{m=0}^{L-1} M_m s_m^2 \vec{\delta}_m}{\sum_{m=0}^{L-1} M_m s_m^2}. \tag{S13}$$

Eq. (S13) is the basis for the recursive algorithm to estimate the animal’s position. Suppose the ML estimate after combining $L$ modules is $\vec{x}_L$ and now we wish to compute the ML estimate $\vec{x}_{L+1}$ for $L + 1$ modules. We have

$$\vec{x}_{L+1} = \frac{\sum_{k=0}^{L} M_k s_k^2 \vec{\delta}_k}{\sum_{k=0}^{L} M_k s_k^2} = \frac{\sum_{k=0}^{L-1} M_k s_k^2 \vec{\delta}_k}{\sum_{k=0}^{L-1} M_k s_k^2} \left( \sum_{k=0}^{L-1} M_k s_k^2 \vec{\delta}_k \right) + \frac{M_L s_L^2}{\sum_{k=0}^{L} M_k s_k^2} \vec{\delta}_L.$$

The optimal decoding strategy relies not on estimating $\vec{\delta}_L$ in an arbitrary fixed reference frame, but in the reference frame centered on the current estimate after combining $L$ modules, namely on $\vec{x}_L$ (cf. Eq. 3 in the main text; Fig. 6 for illustration). This strategy yields $\Delta \vec{\delta}_L = \vec{\delta}_L - \vec{x}_L$ and, therefore,

$$\vec{x}_{L+1} = \vec{x}_L + \frac{M_L s_L^2}{\sum_{k=0}^{L} M_k s_k^2} \left( \vec{x}_L + \Delta \vec{\delta}_L \right),$$

which simplifies to

$$\vec{x}_{L+1} = \vec{x}_L + \frac{M_L s_L^2}{\sum_{m=0}^{L} M_m s_m^2} \Delta \vec{\delta}_L.$$

Using $s_m = \lambda_0/\lambda_m$, this can also be written as

$$\vec{x}_{L+1} = \vec{x}_L + \frac{M_L \lambda_L^{-2}}{\sum_{m=0}^{L} M_m \lambda_m^{-2}} \Delta \vec{\delta}_L.$$
The factor $M_L\lambda_L^{-2}/\sum_{m=0}^{L} M_m\lambda_m^{-2}$ thus describes the weight with which the estimate from the $(L+1)^{st}$-module has to be integrated by the read-out. Additionally, $\Delta\vec{\delta}_L$ has to be calculated in the reference frame centered on the current estimate after combining $L$ modules. Fig. S2 illustrates why centering the reference frame is necessary for accurate decoding.

This algorithm is described in Eqs. (2) and (3) of the main text for the special case of 2 modules in 1D. In Eq. (3), modulating the second module’s population vector by the factor $\exp(-i2\pi/\lambda_1x_0)$ centers the reference frame. Biologically, this multiplicative interaction could be implemented through the product of basis functions, as has been proposed for the gain fields of receptive fields in parietal cortex (40, 41). These receptive fields are fixed with respect to the retina, but are modulated by eye position (38). Our theory specifically predicts that the read-out neurons of a multi-scale grid code should show gain-field-like responses. This prediction could be tested by recording from putative read-out neurons during navigation. For instance, electrically or optogenetically stimulating grid cells with the largest grid sizes should affect the read-out neuron’s synaptic responses to grid cells at other scales.

A biological read-out network would need to perform the following steps iteratively:

1. calculate the population vector estimate $\vec{x}_0$ from the neuronal activity at the coarsest spatial scale $\lambda_0$.

2. use the estimate $\vec{x}_0$ to bias the next population vector estimate at the succeeding spatial scale. This bias is mathematically equivalent to a multiplication, akin to a ‘gain field’ modulating tuning curves in parietal cortex (40, 41).

3. sum over population vectors.

As described earlier for a single module network, the neuronal activity in a multi-scale grid code could also be used to simultaneously calculate vectors to different objects or locations.
Figure 5B compactly describes the algorithm. Within each separate module (with length scale $\lambda_m$), the population vectors map onto a vector on a standardized torus, described by two angular coordinates $(\Delta \phi_m, \Delta \theta_m) = \Delta \vec{\varphi}_m$. Each angular coordinate lies within an interval of length $2\pi$, so these coordinates need to be scaled to reflect the true dimensions of the grid. With $\Delta \vec{\delta}_m = \frac{\lambda_m}{2\pi} \Delta \vec{\varphi}_m$, we obtain

$$\vec{x}_1 = \frac{\lambda_0}{2\pi} \Delta \vec{\varphi}_0$$

and

$$\vec{x}_{L+1} = \vec{x}_L + \frac{1}{2\pi} \frac{M_L \lambda_L^{-1}}{\sum_{m=0}^{L} M_m \lambda_m^{-2}} \Delta \vec{\varphi}_L.$$  \hfill (S14)

If the number of neurons per module is the same across all modules and the scale ratio $s = \lambda_{k+1}/\lambda_k$ is constant, then $\vec{x}_{L+1} = \vec{x}_L + \lambda_0/\lambda_0 (s_0/\sum_{l=0}^{L} s_l^2) \Delta \vec{\varphi}_L$.

For completeness, we state the iterative, multi-scale algorithm that incorporates information from the length of the population vectors. Such an algorithm extends Eq. (S5) from a single module to multiple modules. Denote the population vector lengths as $\hat{\kappa}_{l,m}$, where $l$ is the index of the corresponding lattice axis, and $m$ indexes the module. Define $\mu_{l,m} = \vec{k}_l \cdot \Delta \vec{\varphi}_m$. The terms $\hat{\kappa}_{l,m}$ are proportional to $M_m$, so the dependence on $M_m$ will be implicit in the following formula. For hexagonal lattices, we obtain

$$\vec{x}_{L+1} = \vec{x}_L + \frac{\lambda_0}{2\pi} \frac{1}{N_L} \left( \sum_{l,p=1}^{3} \nu_{l,L} s_L \hat{\kappa}_{p,L} \mu_{p,L} \sin(\phi_l) \sin(\phi_l - \phi_p) \right) \sin(\phi_l) \sin(\phi_l - \phi_p) \right) \left( \sum_{l,p=1}^{3} \nu_{l,L} s_L \hat{\kappa}_{p,L} \mu_{p,L} \cos(\phi_l) \sin(\phi_p - \phi_l) \right)$$  \hfill (S15)

with $\nu_{l,L} = \sum_{m=1}^{L} s_m^2 \hat{\kappa}_{l,m}$ and $N_L = \sum_{p>q} \nu_{p,L} \nu_{q,L}$.  

Figure S6 compares the result of decoding using either Eq. (S14) or Eq. (S15) to numerically maximizing Eq. (1) for the posterior probability $P(\vec{x} | \vec{n})$. Both methods approximate the numerical ML method when the tuning curves densely cover the unit cell ($M \geq 64$).
1.5 Universality and the metric nature of the read-out

Equations (S5) and (S14) convert the spike count \( n \) across the neuronal population into a vector in the two-dimensional plane of \( \mathbb{R}^2 \). Computing the length of this vector corresponds to the standard Euclidean metric for \( \mathbb{R}^2 \). Yet as \( n \) is a random variable, the decoded vector is a noisy version of the true vector in space. Is the neural representation then truly a metric?

A metric (defined in the mathematical sense as a distance measure \( d(p, q) \)) must satisfy three conditions (46):

1. \( d(\vec{x}, \vec{y}) \geq 0 \) with equality if and only if \( \vec{x} = \vec{y} \)
2. \( d(\vec{x}, \vec{y}) = d(\vec{y}, \vec{x}) \) symmetry
3. \( d(\vec{x}, \vec{y}) + d(\vec{z}, \vec{y}) \geq d(\vec{x}, \vec{z}) \) triangle inequality

In the neural representation of distances, we have two components: the true vector \( \vec{x} \), which gives rise to the population spike count \( n \), and possible origins of the coordinate system relative to which the estimate of \( \vec{x} \) is measured. Choose two different origins and call them \( \vec{y} \) and \( \vec{z} \).

Suppose that both \( \vec{y} \) and \( \vec{z} \) are known with perfect precision. Then

\[
d(\vec{x}_{ML}(n), \vec{y}) + d(\vec{y}, \vec{z}) \geq d(\vec{x}_{ML}(n), \vec{z}),
\]

as the triangle inequality of the Euclidean metric holds for any realization of \( \vec{x}_{ML}(n) \). Interchanging the roles of \( \vec{x} \) and the origin \( \vec{y} \) will lead to \( d(\vec{x}_{ML}(n(\vec{x})), \vec{y}) = d(\vec{y}_{ML}(n(\vec{y})), \vec{x}) \) on average, provided the neurons’ tuning curves uniformly tile the hexagonal unit cells of the grid lattices across all modules. The distance \( d((\vec{x}_{ML}(n(\vec{x})), \vec{x}) > 0 \), but a properly designed grid code will ensure that \( d((\vec{x}_{ML}(n(\vec{x})), \vec{x}) < \delta \ll 1 \) with high probability.

This demonstrates that the nested grid code observed in the entorhinal cortex (5), when endowed with the read-out presented in this study can be used by the animal to make metric comparisons and to calculate vectors to multiple points of interests. While there are certainly
other neuronal representations of space in the brain (like place cells, border cells, head direction cells, or turn-preferring cells \((35, 47, 48)\)) that could be used to learn a metric representation, the fixed relationship of grid cells across environments \((1, 49, 50)\) allows a read-out that has been learned once to be used across many contexts.

1.6 Quantitative analysis of scaling ratio \(s\)

Consider a nested grid code with fixed \(s = \lambda_{k+1}/\lambda_k\), starting with a largest spatial period \(\lambda_0\). We are interested in finding the scaling ratio that maximizes the spatial information. The performance of maximum likelihood decoding is measured by the probability distribution of typical deviations from the true position: \(\vec{\epsilon} = \vec{x}_{\text{ML}} - \vec{x}\). The information is \(\langle \log_2 (P(\vec{\epsilon})) \rangle\) up to a constant, where \(\langle \cdot \cdot \cdot \rangle\) indicates an average over \(P(\vec{\epsilon})\). To normalize the information, we assume that, in the absence of a response, any \(\vec{\epsilon}\) is equally likely and hence has probability \(1/\det(\mathcal{L})\), where \(\det(\mathcal{L})\) is the area of the lattice \(\mathcal{L}\)'s unit cell.

The root-mean-square (RMS) error is \(\sqrt{\langle |\vec{\epsilon}|^2 \rangle}\). For the optimization results shown in Fig. 7E, we replace \(P(\vec{\epsilon})\) by

\[
P(\vec{\epsilon}) \rightarrow \exp \left\{ \langle \ln P(\vec{x}|n) \rangle_n \right\},
\]

for which the average is taken over all realizations of the spike count \(n\). The right-hand side of the equation above can be written in closed form. This yields a fairly tight lower bound for the second moment of \(P(\vec{\epsilon})\) and, hence, a lower bound on the RMS error.

Robust encoding requires that all moments of \(P(\vec{\epsilon})\) decrease with the addition of modules at finer scales. In particular, the worst possible decoding error \(|\vec{\epsilon}| \sim \lambda_0/2\) for a nested grid code with largest spatial scale \(\lambda_0\) should not become any more probable. For \(\text{von Mises}\) spatial firing rate maps, the posterior probability \(P(\vec{x}|n)\) is a superposition of cosine functions, and these interfere. A second module contributes a term proportional to \(\cos(s\pi)\) at the edge of the range, and this term is zero for \(s = 3/2\). There is a small effect due to the overall normalization of the
probability. For moderate to strong spatial modulation of the tuning ($\kappa > 1$ in the von Mises function), the normalization $\int P(\vec{x}|\mathbf{n})d\vec{x} = 1$ ensures that the probability of the worst possible error, instead of remaining unchanged for $s = 3/2$, in fact decreases slightly. For weak spatial modulation of the firing rate ($\kappa < 1$), the normalization of the probability has the opposite effect, as it slightly increases the relative probability of the worst possible error. We, therefore, examine the limit $\kappa \to 0$. In this limit, if the transcendental equation

$$\cos(\pi s) + \frac{\sin(\pi s)}{\pi s} = 0$$

is satisfied, then the probability of the worst error is not increased. The solution to this equation is $s \approx 1.4303$. As long as $s < 1.43$, decoding is robust, regardless of the degree of spatial modulation. These results support the observation that for a broad range of tuning widths $\kappa$ the optimal scaling ratio for small numbers of spikes lies around $3/2$.

Figure 7A shows why a scale ratio $s = 3/2$ is appropriate in one dimension. In two dimensions, if the lattice orientations are aligned across modules, then the secondary peaks in the posterior distribution occur along the axes of the lattice. The axes $\vec{a}_l$ and the wave vectors $\vec{k}_l$ differ by a rotation of $\pi/6$. If we set $\vec{x} - \vec{\mu}_l = z\vec{a}_q$, then the expected posterior distribution obeys

$$\langle \ln(P(z|\mathbf{n}) \rangle = \sum_m \hat{\kappa}/3 \left[ 1 + 2 \cos \left( 2\pi \frac{s^m}{\lambda_0} z \right) \right],$$

where, for the expected value of the log-probability, we approximate the $\hat{\kappa}_l$ for the three population vectors by one constant $\hat{\kappa}$. Aside from an additional constant, the expression above replicates the one-dimensional case. If $s = 2$, the worst case error becomes more likely, whereas $s = 3/2$ is a conservative choice.

1.7 The requirement of discrete grid scales

If the grid scale and grid orientation is the same for each neuron within a module, then the expected probability density $\langle P(\vec{x}|\mathbf{n}) \rangle$ is periodic. The reason for this lies in the fact that
\[ \ln P(\bar{x} | \mathbf{n}) \sim \sum_{j=1}^{M} \sum_{l=1}^{3} n_j \cos \left( \omega \vec{k}_l \cdot (\bar{x} - \vec{c}_j) \right) \] and that the sum of cosines with different spatial phases, but the same spatial frequency, is another cosine. In two dimensions (or higher dimensions), we have the additional requirement that the grid lattice vectors \( \vec{k}_l \) be (approximately) the same for \( \langle P(\bar{x} | \mathbf{n}) \rangle \) to be periodic (as shown in Fig. 3A of the main text).

We can consider the case in which the grid scales are not discrete, but arranged along a continuum. For such a continuum, the log-posterior is the superposition of sinusoids of different spatial frequencies; hence, the posterior is no longer periodic over short distances. With respect to the Fisher information, which measures the asymptotic efficiency of maximum likelihood (ML) decoding, grid codes with discretized scales are not intrinsically superior to grid codes with a continuum of scales (3). While ML decoding is certainly feasible for a continuum of grid scales, we now investigate whether population vector decoding remains possible. For simplicity, we focus on the one-dimensional case. We define a population vector estimate \( \hat{x} \) as

\[ \hat{x} = \frac{\lambda}{2\pi} \arg \left( \sum_j n_j \exp \left( i \tilde{\phi}_j \right) \right). \]

The key feature of a population vector is that it results from a weighted linear sum of the spike counts; the weights are complex, but are not a function of \( x \). In a module with a single, well-defined grid scale, we would take \( \tilde{\phi}_j = c_j \), where \( c_j \) is the spatial phase of the \( j \)-th neuron’s tuning curve (the point at which it reaches its peak). For a continuum of grid scales, we will not restrict ourselves to \( \tilde{\phi}_j = c_j \).

Approximate a continuum of grid scales by giving each neuron a tuning curve \( \Omega_j(x) \) with its own spatial scale:

\[ \Omega_j(x) = n_{\text{max}} \exp \{ \kappa [\cos(s_j x - c_j) - 1] \} \quad \text{with} \quad s_j = s_{\text{min}} + \frac{j - 1}{M - 1} (s_{\text{max}} - s_{\text{min}}), \]

\[ \text{(S16)} \]

such that there are \( M \) distinct grid scales \( \lambda_j = 1/s_j \). This particular model has more scales closer to the smallest scale \( 1/s_{\text{max}} \) than to the longest scale \( 1/s_{\text{min}} \).
To illustrate, we take $M = 512$ neurons with random phases $c_j$ and $n_{\text{max}} = \kappa = 2$. If one plots $\Omega(x)$ against phase $c_j$ for $x = 0$, the ensemble follows a von Mises function (Fig. S7A-B). However, for $x \neq 0$, the terms $s_j x$ in Eq. (S16) disrupt the orderly arrangement of firing rate versus spatial phase (Fig. S7C). Yet for a particular $x = x_0$, we can define a new set of phases

$$\tilde{\phi}_j(s_j, x_0) = \left( c_j - (s_j - s_{\text{min}}) * \frac{2\pi}{\lambda_0 x_0} \right) \mod 2\pi. \quad (S17)$$

Figures S7C and S7D show this transformation. We can then form the population vector $n_j \exp \left( i\tilde{\phi}_j \right)$, as in the main text, to decode $x$.

This would presuppose prior knowledge of $x_0$, however. The solution is to compute not just one population vector, but a multitude of population vectors for different trial $x_0$’s and then to choose the longest one. In contrast, a modular arrangement of grid scales in 1D requires the computation of a single population vector for each module; no comparisons across different population vectors need be made. Population vector decoding using Eq. (S17) is shown in Fig. S8. Because the length of the population vector is another random variable, choosing the longest such vector adds uncertainty to the position estimate $\hat{x}$. Taking the wrong $x_0$ to center the phases will mean that $\hat{x}$ will be biased (Fig. S8). In the continuum case, population vector decoding also sacrifices resolution: even though the ensemble’s expected firing rate against $\tilde{\phi}_j$ can be mapped onto a von Mises function of any desired spatial frequency (for instance, by multiplying the right-hand side of Eq. (S17) by $s_{\text{max}}$ to define a new set of $\tilde{\phi}_j$’s), no resolution is gained by moving to higher spatial frequencies, because the number of neurons that contribute to the population vector decreases.

### 1.8 Modular arithmetic for variable numbers of neurons and modules

The optimal scale ratio $s = 3/2$ is unchanged if one allows for different numbers of neurons $M_m$ per module, for which $\Delta \ln P \propto \frac{2\pi^2 M_0 M_1}{M_0 M_1 + M_1 q^2}$; the value $s = p/q = 3/2$ still maximizes $\Delta P$. For arbitrary $L > 1$, with a geometric progression of scales with $s = p/q$ and a constant number
of neurons per module, the difference between the primary peak and the largest secondary peaks is

\[ \Delta \ln P \propto \frac{\sum_{i=0}^{L-2} p^{2i} q^{2(L-2-i)}}{\sum_{i=0}^{L-1} p^{2i} q^{2(L-1-i)}} \]

The value \( s = p/q = 3/2 \) maximizes \( \Delta P \) for arbitrary \( L \), as long as we require \( q > 1 \). As \( L \) becomes large, the difference \( \Delta \ln P \) scales as \( 2\pi^2 / \text{max}(p, q)^2 \). By comparison, for \( L \) modules with spatial periods \( \lambda_0, \ldots, \lambda_{L-1} \) that are incommensurate, but similar in size, the smallest \( \Delta \ln P \) will be proportional to \( 2\pi^2 / \lambda_0^2 \).
2 Supplementary Figures

Figure S1: Hierarchical self-similar scales enable error correction. (A) Two independent modules at length scales $\lambda_0$ and $\lambda_1$ yield a posterior probability $P(\vec{x}|n) = P_0(\vec{x}|n_0) \cdot P_1(\vec{x}|n_1)$. As a consequence, the log-posterior is the sum of the modules’ log-posterior probabilities. The maximum of each log-posterior is generically quadratic (non-quadratic maxima occur only in degenerate cases, when the second-order Taylor series coefficient is constrained to be zero). Under self-similar scaling, all functions are scaled $f(x) \rightarrow f(sx)$ with $s = \lambda_0/\lambda_1$. (B) The maximum of the combined log-posterior probability occurs when the slope becomes zero. The case illustrated shows an error of $\delta$ made on the coarse scale $\lambda_0$; the error made on the fine scale $\lambda_1$ is zero (i.e., no error). The slope averaged over the two modules is shown in green. If both modules contain the same number of neurons, adding the second module reduces the error from $\delta$ to $\delta/(1 + s^2)$. 

$\epsilon = \frac{1}{1 + s^2}$
Figure S2: Non-integer scale ratios imply that the decoding algorithm must be able to rotate population vectors. (A) Let three discrete locations be resolved at the coarsest scale, at the 12 o’clock, 4 o’clock and 8 o’clock positions. If the next scale is 2/3 of the previous scale, then the estimate can be refined within the range of the blue arc. (B) A shift from the 12, 4, or 8 o’clock positions is signaled by the population vector at the finer scale. For the 4 o’clock position, the population vector must be rotated by 180°, lest ‘10 minutes after the hour’ be wrongly interpreted as ‘20 minutes before the hour’.
Figure S3: The lengths of the population vectors $\hat{\kappa}_l$ along the hexagonal grid’s axes are correlated. The parameters of the tuning curves $\Omega_j(x)$ are $\kappa = n_{\text{max}} = 2$ and $M = 64$ with equally spaced spatial phases $\vec{c}_j$. Note the high variance of $\hat{\kappa}_l$ as measured for $2^{10}$ realizations of the spike count $n$. The correlation coefficient between $\hat{\kappa}_l$’s is $\rho = 0.49$. 
Figure S4: Possible distributed representations of the position vector estimate $\vec{\mu}$ of Eq. (S6) across a population of read-out neurons. The origin of the coordinate system (goal location) is at the center of the hexagon. (A) By projecting Eq. (S6) onto a “preferred direction” $\vec{e}_r$ and thresholding, the spatial firing map $r(x, y)$ of such a read-out cell exhibits wedge-like ramps that increase as the animal moves away from the goal location. $M = 36$ grid cells were decoded that covered a fundamental domain of size $1 \text{ m}^2$, denoted by the hexagon. The units of the read-out neuron’s response $r(x, y)$ are arbitrary, as the scaling is not specified. (B) If we multiply the population vectors by $\exp(i\pi)$ before taking the argument in Eq. (S6), followed by projection and thresholding, then the ramps in $r(x, y)$ increase towards the goal location.
Figure S5: Comparison of different decoding schemes for a single module in two dimensions. (A) The x-coordinate $\hat{\mu}_x$ of the position estimate for $2^{10}$ realizations of the spike count in response to random $\vec{x}$, plotting the result of Eq. (S6) (population vector) or Eq. (S5) ($\kappa$-weighted population vector) against the result of numerically maximizing the posterior $P(\vec{x}|n)$. The parameters are $M = 64$, $\kappa = n_{\text{max}} = 2$. The y-coordinate $\hat{\mu}_y$ behaves similarly. (B) The x-coordinate of the estimate $\hat{\mu}_x$ (at larger magnification) against the true x-coordinate $x_0$ for the three decoding schemes. (C) For these parameters, only a fraction of the $M = 64$ neurons participate in encoding the position.
The winner-take-all scheme estimates $\vec{\mu}$ to be the spatial phase $\vec{c}_j$ of the neuron that fired the greatest number of spikes; spike count ties are broken at random. Decoding based on Eq. (S14) (population vector) or Eq. (S15) ($\kappa$-weighted population vector) is comparable in performance to numerical maximum likelihood. The winner-take-all decoder performs poorly. (A) The root-mean-square error of decoding for $2^{14}$ realizations of the spike count, as a function of module size $M$ and the number of modules. (B) The distribution of radial errors $|\vec{\mu} - \vec{x}|$ of the simple population vector decoder in comparison to the maximum likelihood decoder.
Figure S7: If the grid scales are not organized into discrete modules, population vector decoding is no longer straightforward. We take $M = 512$ tuning curves in one dimension, each with a random phase $\phi_j$ and a unique $s_j$ between $s_{\text{min}} = 1$ and $s_{\text{max}} = 2$. This $s_j$ describes the inverse spatial period of the tuning curve. (A) For $x = x_0 = 0$, the expected firing rate plotted against $c_j$ in the ensemble follow a von Mises function. (B) If we change $x$, then this orderly arrangement is disrupted (red points). However, we can define a new set of phases $\tilde{\phi}_j$ change (right panels) that would maintain the orderly arrangement of the expected firing rates (green points). (C) We sort the neuron index according to the phase $c_j$. For $x = 0$, $\tilde{\phi}_j = c_j$ is the identity map, and thus an increasing function of the neuron index. (D) If we now change to $x = -\lambda_0/2$ and use Eq. (S17), the phases $\tilde{\phi}_j$ change.
Figure S8: Continuum decoding requires multiple population vectors. In one dimension (corresponding to a linear track), \( M = 16 \) tuning curves were given distinct scales, so that the spatial frequency ranged between \( s_{\text{min}} = 1 \) and \( s_{\text{max}} = 2 \). The fixed spatial phases for each tuning curve (\( c_j \) in Eq. (S16)) were optimized to yield the same sum of spike counts over the interval \([-1/2, 1/2]\), regardless of the position \( x \). For decoding, the population spike count was simulated 2\(^{17}\) times given a true position of \( x = 0.15 \). Different \( x_0 \) are chosen to center a new set of phases \( \tilde{\phi}_j \) according to Eq. (S17), yielding a set of population vectors. (A) The circular average position estimate \( \langle \hat{x} \rangle \) for each population vector, as a function of \( x_0 \). For \( x_0 \neq x \), the estimate is systematically biased. (B) For \( x_0 \) close to the true position \( x \), the population vector reaches maximum length. (C) The standard deviation of \( \hat{x} \) is smallest for \( x_0 \) close to the true \( x \).