Supporting Information

Sameet Sreenivasan and Ila Fiete

Contents

1 Iterative (recursive) estimation with accruing noise improves greatly with strong error-correcting coding **2**

2 Analytically tractable tuning curves **3**

3 Network phase and firing rate vector are homeomorphic **3**

4 Fisher information of the GPC and the CPC **4**
4.1 Fisher information of the GPC 5
4.2 Fisher information of the CPC 6
4.3 Comparision of FI for the GPC and the CPC 7
4.3.1 Equal tuning-curve widths (equal lifetime sparseness) 7
4.3.2 Tuning-curve widths matched to neuron number per network (equal population sparseness per network) 8
4.3.3 Multi-scale CPC with tuning curve diversity is qualitatively similar to a CPC 9
4.4 Summary of FI and conditional mean-squared error ratios for the GPC and CPC . . 10
4.5 Discussion: conceptual reasons for the exponential scaling of the GPC-CPC FI ratio 10

5 Optimal location readout from grid phase vector **10**

6 The model neural network's inference approximates ML decoding **11**

7 Derivation: how minimum distance (d_{min}) scales with neuron number **12**

8 Mapping the model readout to the entorhinal-hippocampal circuit **14**

9 Decoder complexity and comparison of the GPC and CPC **15**
9.1 Comparing GPC encoding against a CPC enhanced in size by GPC decoder neurons 15
9.2 The same (GPC) readout network can implement priors and improve location estimation for the CPC. 17

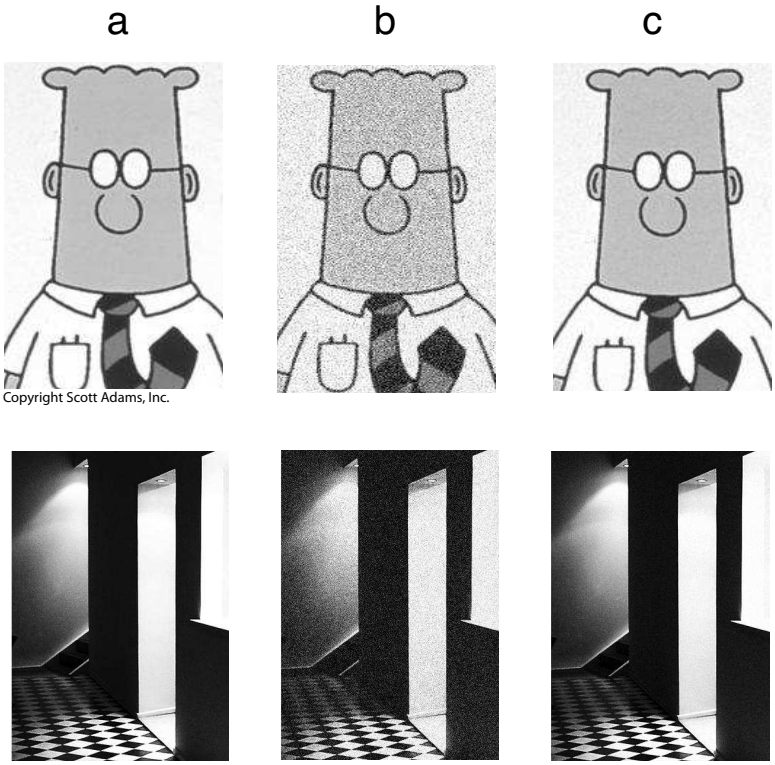
10 Performance of the error-correction loop when readout cells have multiple fields **19**

11 Partial error correction with sparsely allocated readout cells **20**

12 Robustness of error correction with imperfect grid cell-readout and readout-grid cell weights **22**

13 Results generalize to 2-dimensional space **23**

1 Iterative (recursive) estimation with accruing noise improves greatly with strong error-correcting coding



Supporting Figure 1: **A comparison of error reduction in a recursive setting with the GPC and the CPC.** (a) Two initial images. (b,c) The images are subject to 20 iterations of noise addition, with each iteration followed by decoding by the CPC (b) and the GPC (c), followed by noise addition to the decoded image, and so on. To simulate this iterative process, the pixel values for one iteration are obtained by sampling from the posterior distributions of the CPC and the GPC (using the distributions from Figure 1d of the main paper), centered at the pixel values from the end of the last iteration. After such 20 iterations, the CPC images (b) are badly degraded, while the GPC code produces an image close to the original (c) (CPC and GPC images are best viewed electronically at high magnification). This example highlights the importance of near-exact error removal for iterative or recursive noisy computation. [Numerical details: Each pixel intensity is treated as a separate, independent location variable, with a legitimate range of 0 to 255 units. The posterior distribution for the GPC is computed assuming $N = 18$ periods with $M = 50$ neurons each, with periods $10, 14, \cdots, 78$ units, and $R_l = 256$ units, with Gaussian noise in the normalized phases of standard deviation $\sigma_{GPC} = 0.1$. The posterior distribution for the CPC is computed assuming NM neurons, the same R_l, and noise standard deviation in the normalized phase of $\sigma_{GPC}/\sqrt{N}$. Thanks to Horacio Jorge di Nunzio for the second image.

2 Analytically tractable tuning curves

The Gaussian (normal) neural tuning curve of Equation (10) in the Methods, with the (non-differentiable) modulo phase variable and its special metric, Equation (11) in the Methods, can alternatively be modeled by the closely related circular normal function, which has the advantage of continuity and differentiability:

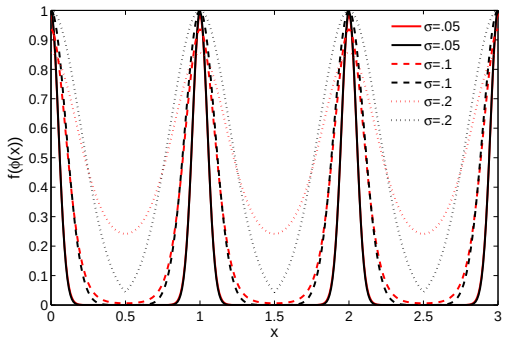
$$f_\kappa(\underline{\phi}, \phi^*) = \exp\Big(\kappa\big[\cos(2\pi||\underline{\phi} - \phi^*||) - 1\big]\Big) \tag{1}$$

In the limit $\sigma \to 0$ or $\kappa \to \infty$, the tuning curves of Equation (10) in the Methods and Equation (1) above are identical, with $\kappa = 1/(2\pi\sigma)^2$. The tuning curves are already nearly indistinguishable at $\sigma = 0.1$, Figure S2.

Next, by the periodicity and symmetry of the cosine function and by Equation (1) of the main paper and Equation (11) of the Methods, the following are equal: $\cos(2\pi\underline{\phi}), cos(2\pi||\underline{\phi}||)$, and $\cos(\frac{2\pi x}{\lambda})$. Thus,

$$f_\kappa(\underline{\phi}(x), \phi^*) = \exp\Big(\kappa\big[\cos(2\pi\Big(\frac{x}{\lambda} - \phi^*\Big)) - 1\big]\Big) \tag{2}$$

In all of the above, $\underline{\phi}$ may be substituted with $\phi(x,t)$.



Supporting Figure 2: **Gaussian tuning curves in phase space are well approximated by the circular normal distribution.** The tuning curve of Equations (10)–(11) from the Methods (in black) versus that of Equation (2) (in red). Here $\lambda = 1$, and various tuning widths are plotted. The two functions are identical as $\sigma \to 0$, and already nearly indistinguishable for $\sigma = 0.1$.

In summary, the Gaussian tuning curve with the phase metric and the circular normal distribution with argument x describe similar tuning curves whether viewed in x or ϕ. The two descriptions become identical in the limit of narrow tuning curves, and to a reasonable approximation may be used interchangeably. Our numerical results show no qualitative difference between these two representations of the tuning curves. Theoretically, we should expect no qualitative differences, because the exponentially strong error correction of the grid code comes not from the tuning curve shape around the preferred firing locations, but from the fact that the preferred locations are periodic.

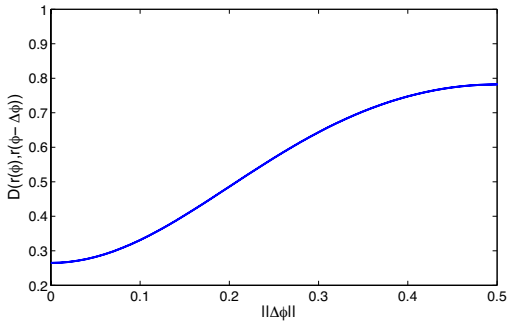
3 Network phase and firing rate vector are homeomorphic

Let the M-dimensional $\vec{r}(\phi)$ be the population-level firing rate vector of all M neurons within one network, for phase ϕ (Equations (1)-(2)):

$$r_i(\phi) = e^{-\kappa} e^{\kappa cos(2\pi(\phi - \phi_i^*))}$$

with norm $|r| = e^{-\kappa} I_0(\kappa)$, where I_0 is the modified Bessel function of order 0. The similarity of $\vec{r}(\phi)$ and $\vec{r}(\phi')$ is quantified by $d(r, r') = (1 - \vec{r} \cdot \vec{r'}/|r|^2)$, based on the dot-product:

$$\begin{aligned}
\vec{r}(\phi) \cdot \vec{r}(\phi') &= \sum_i e^{-2\kappa} e^{\kappa[\cos(2\pi(\phi - \phi_i^*)) + \cos(2\pi(\phi' - \phi_i^*))]} \\
&\approx e^{-2\kappa} \int d\theta \; e^{\kappa[\cos(2\pi(\phi-\theta)) + \cos(2\pi(\phi'-\theta))]} \\
&= e^{-2\kappa} \int d\theta \; e^{2\kappa \cos(2\pi(\frac{\Delta\phi}{2})) \cos(2\pi(\theta - \frac{(\phi+\phi')}{2}))} \\
&= e^{-2\kappa} \int du \; e^{2\kappa \cos(2\pi(\frac{\Delta\phi}{2})) \cos(u)} \\
&= e^{-2\kappa} I_0\big(2\kappa \cos(2\pi(\frac{\Delta\phi}{2}))\big) \qquad (3)
\end{aligned}$$



Supporting Figure 3: **Relationship between phase differences and firing rate vector differences in a single grid network.** The relationship between $||\Delta\phi||$ and $d\big(\vec{r}(\phi), \vec{r}(\phi - \Delta\phi)\big)$ is 1-1 onto. In this plot, the tuning width of the neural responses is $\sigma_e = 0.1$.

By the symmetry $I_0(x) = I_0(-x)$ and the symmetry of cosine, it follows that $I_0\big(\kappa \cos(2\pi(\frac{\Delta\phi}{2}))\big) = I_0\big(\kappa \cos(2\pi(\frac{||\Delta\phi||}{2}))\big)$. Thus, we have

$$d\big(\vec{r}(\phi), \vec{r}(\phi + \Delta\phi)\big) = 1 - \frac{I_0\big(\kappa \cos(2\pi(\frac{||\Delta\phi||}{2}))\big)}{I_0(\kappa)^2} \qquad (4)$$

Shown in Figure S3 is the resulting monotonic relationship of Equation (4) between the distances $d\big(\vec{r}(\phi), \vec{r}(\phi + \Delta\phi)\big)$ in population firing rates versus the distances $||\Delta\phi||$ between phases. This mapping is 1-1, onto, thus the two representations of phase and rates are topological homeomorphisms.

This homeomorphism is useful: it means the analysis on a phase variable versus on the associated mean firing rate vector will produce similar results. In the main paper, we commented that the noisy network phase is a sufficient statistic for the spatial information in the network. Here we show that any network phase and the corresponding firing rate vector for that phase are homeomorphic, or equivalent quantities with respect to the metrics in either space.

4 Fisher information of the GPC and the CPC

In the main paper, we used geometric arguments to derive how the GPC's mean-squared-error in location estimation should scale with N. We assumed that every neuron responds with independent

noise conditioned on animal location x, and used existing results from the theory of CPCs on the variance of encoding a variable (each network's phase) in a continuous attractor network of given size with independent neural noise (the choices of phase noise variance in each grid network and the CPC, respectively, were based on these results). The geometric argument was then based on the interleaving structure of the grid code, and also informed by the derivation from (1) on the exponential scaling of the coding range of the grid code with N.

Here we compute the Fisher Information (FI) of the GPC and CPC in D dimensions, using the differentiable circular normal form of the GPC tuning curves, Equation (2). The single-neuron FI in possible to derive by explicit calculation. The derivation of the total FI in the multiple-period GPC case, as in the geometric arguments of the main paper, depends on the additional knowledge obtained in (1) about the exponential scaling of R_l with the number of periods. The results of the FI calculation agree with the geometric arguments in the main text, as well as with the numerical results presented there on the mean-squared error of the GPC and its scaling with N.

4.1 Fisher information of the GPC

In the paper, we assume the phase noise is generated from the accumulation of error from multiple steps of noisy neural integration of velocity to estimate the location x.

The following computation is based on one time-step of the integration process. We assume that the response of each neuron in each network is independent when conditioned on the input location $\vec{x} = \{x^1, x^2, \cdots x^D\}$ and on the neural tuning curves. Thus, the total Fisher information is the sum over neurons in each network and a sum over networks, of the single-neuron Fisher information. We assume that the tuning curve of each neuron αi (ith neuron in αth network) is a separable function of the different dimensions:

$$f_{\alpha i}(\vec{x}) = f_{\alpha i}(x^1) f_{\alpha i}(x^2) \cdots f_{\alpha i}(x^D) \tag{5}$$

where we assume an exponential form for the tuning curve shape,

$$f(x^l) = e^{Q(x^l)}. \tag{6}$$

The neural response $r_{\alpha i}(\vec{x})$ is some probabilistic function of the tuning curve, $P(r_{\alpha i}|\vec{x}) = H(r_{\alpha i}, f_{\alpha i}(\vec{x}))$. The Fisher information matrix for neuron αi is

$$J_{\alpha i}^{mn}(\vec{x}) = -\left\langle \frac{\partial \log P(r_{\alpha i}|\vec{x})}{\partial x^m} \frac{\partial \log P(r_{\alpha i}|\vec{x})}{\partial x^n} \right\rangle_{r_{\alpha i}}. \tag{7}$$

Using the identities

$$\frac{\partial \log P(r_{\alpha i}|\vec{x})}{\partial x^m} = \frac{1}{H_{\alpha i}} \frac{\partial H_{\alpha i}}{\partial f_{\alpha i}} \frac{\partial f_{\alpha i}(\vec{x})}{\partial x_m}$$

and $\partial f(\vec{x})/\partial x^m = f(\vec{x}) \left(\partial Q(x^m)/\partial x^m\right)$ (derived from Equation (6)), we get

$$J_{\alpha i}^{mn}(\vec{x}) = -\left[\int dr_{\alpha i} \frac{1}{H_{\alpha i}^2} \left(\frac{\partial H_{\alpha i}}{\partial f_{\alpha i}}\right)^2\right] f_{\alpha i}(\vec{x})^2 \; \frac{\partial Q_{\alpha i}(x_m)}{\partial x_m} \frac{\partial Q_{\alpha i}(x_n)}{\partial x_n} \tag{8}$$

If neural responses are Poisson, then

$$H(r, f) = \frac{f^r e^{-f}}{r!}$$

and it follows that

$$\frac{\partial H}{\partial f} = \frac{r - f}{f} H(r, f) \tag{9}$$

Inserting Equation (9) into Equation(8) and computing the integral over the Poisson distribution, we get

$$\begin{aligned} J^{mn}_{\alpha i}(\vec{x}) &= -\langle (r_{\alpha i} - f_{\alpha i})^2 \rangle_{poiss} \ \frac{\partial Q_{\alpha i}(x_m)}{\partial x_m} \frac{\partial Q_{\alpha i}(x_n)}{\partial x_n} \\ &= -f_{\alpha i}(\vec{x}) \ \frac{\partial Q_{\alpha i}(x_m)}{\partial x_m} \frac{\partial Q_{\alpha i}(x_n)}{\partial x_n} \end{aligned} \tag{10}$$

Using the tuning curves $f_{\alpha i}$ defined by Equations (2) and (5), we finally have that the single-neuron FI is

$$J^{mn}_{\alpha i}(\vec{x}) = \prod_{l=1}^{D} \exp\Big(\kappa\big[\cos\big(2\pi(\frac{x^l}{\lambda_\alpha} - \phi_i^{l*})\big) - 1\big]\Big) \left(\frac{2\pi\kappa}{\lambda_\alpha}\right)^2 \ \sin\big(2\pi(\frac{x^m}{\lambda_\alpha} - \phi_i^{m*})\big) \sin(2\pi\big(\frac{x^n}{\lambda_\alpha} - \phi_i^{n*})\big) \tag{11}$$

This result agrees with the calculations of (2) for the FI of circular normal tuning curves for periodic variables, after the appropriate re-scalings of parameters between the two derivations.

The total FI of all neurons in all networks is next obtained from summing over all neurons in one network, and performing an appropriate sum over networks. Because the responses of all N networks and all M neurons per network are independent conditioned on the input, the sum is straightforward. We approximate the sum over neurons within a network by an integral, which is accurate if M, the number of neurons per network, is large (in the CPC calculation we will assume the same). Thus

$$\begin{aligned} J_{GPC}(\vec{x}) &\approx \sum_{\alpha=1}^{N} M \int J_{\alpha i} \ d\phi_i^{l^*} \\ &= M e^{-\kappa D} \sum_{\alpha=1}^{N} \left(\frac{2\pi\kappa}{\lambda_\alpha}\right)^2 \int \prod_{l=1}^{D} d\phi_i^{l^*} \exp\Big(\kappa \cos\big(2\pi(\frac{x^l}{\lambda_\alpha} - \phi_i^{l*})\big)\Big) \ \sin\big(2\pi(\frac{x^l}{\lambda_\alpha} - \phi_i^{m*})\big)^2 \\ &= M \frac{I_0(\kappa)^{D-1} e^{-\kappa D} I_1(\kappa)}{2\pi\kappa} \sum_{\alpha=1}^{N} \left(\frac{2\pi\kappa}{\lambda_\alpha}\right)^2 \end{aligned} \tag{12}$$

where $I_n(\kappa)$ are the modified Bessel functions of the first kind, of order n. Assuming all the periods are similar in size ($\lambda_1 \sim \cdots \sim \lambda_N \sim \lambda$) , we set $\sum_{\alpha=1}^{N} (\frac{1}{\lambda_\alpha})^2 \sim \frac{N}{\lambda^2}$. Therefore,

$$J_{GPC}(\vec{x}) = \frac{NM}{\sigma_e^2 \lambda^2} [e^{-\kappa D} I_0(\kappa)^{D-1} I_1(\kappa)] \tag{13}$$

where $\sigma_e^2 = 1/(2\pi\kappa)$ is the squared width of the GPC neural tuning curve in the $[0, 1)$ phase variable.

4.2 Fisher information of the CPC

The CPC single-neuron FI is derived identically as the GPC calculation, up to Equation (11), but with κ replaced by κ_p, with $\alpha = 1$ (only a single network), and with λ_α replaced by R_l, the range over which the CPC (and the GPC) must encode position $\vec{x}$. Thus

$$J_i(\vec{x}) = \prod_{l=1}^{D} \exp\Big(\kappa_p\big[\cos\big(2\pi(\frac{x^l}{R_l} - \phi_i^{l*})\big) - 1\big]\Big) \left(\frac{2\pi\kappa_p}{R_l}\right)^2 \ \sin\big(2\pi(\frac{x^m}{R_l} - \phi_i^{m*})\big) \sin(2\pi\big(\frac{x^n}{R_l} - \phi_i^{n*})\big) \tag{14}$$

The single CPC network has MN total neurons, and the total FI is the sum over all these neurons, with an integral replacing the sum as in the GPC:

$$\begin{aligned} J_{CPC}(\vec{x}) &\approx MN \int J_{1i}\, d\phi_i^{l^*} \\ &= MN\, e^{-\kappa_p D}\left(\frac{2\pi\kappa_p}{R_l}\right)^2 \int \prod_{l=1}^{D} d\phi_i^{l^*} \exp\Big(\kappa_p \cos\big(2\pi(\frac{x^l}{R_l} - \phi_i^{l*})\big)\Big)\ \sin\big(2\pi(\frac{x^m}{R_l} - \phi_i^{m*})\big)^2 \\ &= MN\, \frac{I_0(\kappa_p)^{D-1} e^{-\kappa_p D} I_1(\kappa_p)}{2\pi\kappa_p}\ \left(\frac{2\pi\kappa_p}{R_l}\right)^2 \end{aligned} \tag{15}$$

Therefore, the total FI of the CPC over range R_l is

$$J_{CPC}(\vec{x}) = \frac{NM}{\sigma_p^2 R_l^2}[e^{-\kappa_p D} I_0(\kappa_p)^{D-1} I_1(\kappa_p)] \tag{16}$$

4.3 Comparision of FI for the GPC and the CPC

Equations (12) and (16) quantify the FI of the GPC and the CPC assuming the same total number of neurons (NM), the same coding range (R_l), and the same model for stochastic neural spiking given the mean firing rate. The FI expressions differ in two ways: First, J_{GPC} scales as $1/\lambda^2$, while J_{CPC} scales as $1/R_l^2$. Second, the two expressions differ in the tuning curve widths σ_e^2 (or $1/2\pi\kappa$) versus σ_p^2 (or $1/2\pi\kappa_p$).

4.3.1 Equal tuning-curve widths (equal lifetime sparseness)

First consider the case if the tuning curve widths over the $[0, 1)$ phase variable in the CPC and the GPC networks are the same ($\kappa_p = \kappa$ or equivalently, $\sigma_p = \sigma_e$) (the same lifetime sparseness per neuron). This choice also corresponds to equal total population sparseness, if the GPC population is counted as the set of all NM GPC neurons across all of the N networks. The FI ratio of GPC and CPC, from Equations (13) and (16), is thus given by:

$$\frac{J_{GPC}(\vec{x})}{J_{CPC}(\vec{x})} = \left(\frac{R_l}{\lambda}\right)^2 \tag{17}$$

Clearly, if $R_l \gg \lambda$, then the FI of the GPC is much larger than that of the CPC. Selecting $\lambda \ll R_l \ll R$ is possible because the coding range of the GPC is R, which grows exponentially with N, the number of periods, and far exceeds any of the periods. More precisely, $R = \lambda\big(\frac{1}{\Delta\phi}\big)^{\beta N}$, where $\beta \simeq 1$, and $(1/\Delta\phi) > 1$ is a measure of the number of distinguishable states in each phase dimension (1). The phase uncertainty $\Delta\phi$ scales as $M^{-1/2D}$ (M the total number of neurons in each grid network, and $M^{1/D}$ is therefore the number of neurons per network per dimension of the encoded variable). This dependence is obtained by assuming that within each phase-coding network, the variance of the estimate of the phase is given the inverse FI, which scales with the number of neurons per dimension.

A choice of $R_l = \lambda\big(\frac{1}{\Delta\phi}\big)^{k\beta N}$ with $\frac{1}{\beta N} < k < 1$ produces $\lambda \ll R_l \ll R$ with the same number of neurons as in the CPC. Then, the FI ratio becomes

$$\frac{J_{GPC}(\vec{x})}{J_{CPC}(\vec{x})} = \left(\frac{1}{\Delta\phi}\right)^{2\rho\beta N} \tag{18}$$

So long as $k\beta N > 1$ (and this is the case for $k \gtrsim 1/N$ because $\beta \sim 1$; in Figure 1 of the main paper, $\rho \approx k \approx 1/2$; in Figure 2, $\rho = 1/2$), or in other words, whenever $R_l \gg \lambda$, the GPC carries overwhelmingly more information than the best classical population code. [If $R_l \sim \lambda$, then the GPC

turns into sets of unimodal tuning curves with slightly different scales or field widths, given by λ_α. In this case, the multi-scale unimodal representation carries a similar amount of FI as the CPC with all unimodal tuning curves sharing a single scale. Thus, the exponential advantage of the GPC holds strictly for $R_l \gg \lambda$.]

Assuming the CPC and GPC estimators saturate the Cramér-Rao bounds given by their respective FI's, the predicted ratio of conditional root-mean-squared error from the GPC and CPC codes is

$$\frac{E_{GPC}}{E_{CPC}} = \left(\frac{1}{\Delta\phi}\right)^{-k\beta N} \tag{19}$$

In actuality, the local variance of the GPC location estimate does not saturate its Cramér-Rao bound. Some of the potential for local coding precision is given away for global error-correction. There are N total periods but suppose only $N_l < N$ of them contribute to variance reduction locally over the range R_l. Then, for the purposes of computing the root-mean-squared-error, we may define some "effective" FI for the GPC, instead of Equation (13), given by

$$J_{GPC}(\vec{x}) = \frac{N_l M}{\sigma_e^2 \lambda^2}[e^{-\kappa D} I_0(\kappa)^{D-1} I_1(\kappa)] \tag{20}$$

with the only difference here being that N is replaced by $N_l < N$. (By analogy with the relationships between $\{N,\ R,\ R_l\}$, N_l should be defined by $R_l \equiv \lambda e^{\beta N_l}$, or in other words, $N_l \equiv kN$.) Using Equation (20) produces that the conditional root-mean-squared-error should be

$$\frac{E_{GPC}}{E_{CPC}} = \left(\frac{1}{\Delta\phi}\right)^{-k\beta N} \sqrt{\frac{N}{N_l}} \tag{21}$$

This scaling of conditional root-mean-squared-error with N is in agreement with the conditional root-mean squared error ratio predicted using geometric arguments in the main paper. It is also consistent with the numerical results on the conditional mean squared error obtained from sampling noisy phases and then decoding them using maximum likelihood. For equal-width phase tuning curves (equal single-neuron lifetime sparseness), the GPC-CPC error ratio is independent of the dimension D of the encoded variable.

4.3.2 Tuning-curve widths matched to neuron number per network (equal population sparseness per network)

Next, consider the case that the phase tuning curves of the CPC are narrower than the corresponding GPC tuning curves. Because there are N times more neurons encoding a single $[0, 1)$ phase variable in the CPC network than in any GPC network, we consider $\sigma_p^2 = \sigma_e^2/N$ (equal population sparseness when one grid network is compared against the full CPC network). Then, the FI ratio for the GPC and CPC is:

$$\frac{J_{GPC}(\vec{x})}{J_{CPC}(\vec{x})} = \frac{R_l^2}{\lambda^2}\frac{\sigma_p^2}{\sigma_e^2}\ \left[e^{\kappa D(N-1)}\frac{I_0(\kappa)^{D-1} I_1(\kappa)}{I_0(\kappa N)^{D-1} I_1(\kappa N)}\right] \tag{22}$$

To lowest order, the asymptotic form of $I_\nu(z)$ for large z is given by

$$I_\nu(z) \sim \frac{e^z}{\sqrt{2\pi z}} \tag{23}$$

Therefore, $I_0(\kappa N)^{D-1} I_1(\kappa N) \sim (\exp \kappa N/\sqrt{2\pi\kappa N})^D$, and thus for large κ, κ_p (small σ_e, σ_p) and large N,

$$\frac{J_{GPC}(\vec{x})}{J_{CPC}(\vec{x})} \sim \left(\frac{R_l}{\lambda}\right)^2 N^{D/2-1} \sim \left(\frac{1}{\Delta\phi}\right)^{2k\beta N} N^{D/2-1} \tag{24}$$

This result is similar to the FI ratio of Equation (18), because the exponential dependence on N renders the algebraic dependence on N mostly irrelevant. If we use the same "effective" FI approach for the GPC as in the section with equal population sparseness, we would multiply the FI ratio by N_l/N, to obtain a predicted conditional root-mean-squared error ratio of

$$\frac{E_{GPC}}{E_{CPC}} \sim \left(\frac{1}{\Delta\phi}\right)^{-k\beta N} \sqrt{\frac{1}{N_l}}\, N^{1-D/4} \tag{25}$$

This result, with narrower tuning curves in the CPC than the GPC ($\sigma_p^2 = \sigma_e^2/N$), has some algebraic dependence on D, but like Equation (19) and the main text, is dominated by the exponential dependence on N regardless of dimension.

4.3.3 Multi-scale CPC with tuning curve diversity is qualitatively similar to a CPC

Place cell field widths scale by about a decade, along the septo-temporal (long) axis of the hippocampus. This range of field widths is similar to the range of different periods across the grid cell networks. We consider the FI for the case of unimodal tuning curves, with N networks of M neurons each, having field widths $\sigma_{p1} < \cdots < \sigma_{pN}$, respectively. The single-neuron FI in the αth such network is given, as in Equation (14), by

$$J_{\alpha i}(\vec{x}) = \prod_{l=1}^{D} \exp\Big(\kappa_{p\alpha}\big[\cos\big(2\pi(\frac{x^l}{R_l} - \phi_i^{l*})\big) - 1\big]\Big)\left(\frac{2\pi\kappa_{p\alpha}}{R_l}\right)^2 \sin\big(2\pi(\frac{x^m}{R_l} - \phi_i^{m*})\big)\sin(2\pi\big(\frac{x^n}{R_l} - \phi_i^{n*})\big) \tag{26}$$

We sum over neurons M and sum over the different networks to obtain the total FI

$$\begin{aligned}
J_{CPC\ multi}(\vec{x}) &\approx \sum_{\alpha=1}^{N} M \int J_{\alpha i}\, d\phi_i^{l^*} \\
&= M\sum_{\alpha=1}^{N} e^{-\kappa_{p\alpha}D}\left(\frac{2\pi\kappa_{p\alpha}}{R_l}\right)^2 \int \prod_{l=1}^{D} d\phi_i^{l^*} \exp\Big(\kappa_{p\alpha}\cos\big(2\pi(\frac{x^l}{R_l} - \phi_i^{l*})\big)\Big)\ \sin\big(2\pi(\frac{x^l}{R_l} - \phi_i^{m*})\big)^2 \\
&= M\sum_{\alpha=1}^{N} \frac{I_0(\kappa_{p\alpha})^{D-1} e^{-\kappa_{p\alpha}D} I_1(\kappa_{p\alpha})}{2\pi\kappa_{p\alpha}}\left(\frac{2\pi\kappa_{p\alpha}}{R_l}\right)^2 \qquad (27) \\
&\sim M\frac{1}{R_l^2}\sum_{\alpha=1}^{N}(2\pi\kappa_{p\alpha})^{1-D/2} \qquad (28)
\end{aligned}$$

where for the last line, we used the asymptotic expansion for the Bessel functions given in Equation (23). The FI for this multi-scale CPC scales as

$$J_{\alpha i}(\vec{x}) = MN\frac{1}{R_l^2}(2\pi\bar{\kappa}_p)^{1-D/2} \tag{29}$$

where $\bar{\kappa}_p$ is defined through $\sum_{\alpha=1}^{N}(2\pi\kappa_{p\alpha})^{1-D/2} = N(2\pi\bar{\kappa}_p)^{1-D/2}$. This result has the same dependence on R_l as Equation (16) for the CPC, if all the field widths $\sigma_{p\alpha} \sim 1/\sqrt{\kappa_{p\alpha}} \sim \mathcal{O}(\lambda)$. As a consequence, the FI ratio with the GPC will still be dominated by the exponential dependence on N seen before. In fact for $D = 2$, the FI ratio for the GPC and the multiscale CPC are identical to Equation (17), regardless of the field widths. Thus, multi-scale unimodal representations are not in the same performance class as the GPC, and instead produce performance similar to the CPC.

4.4 Summary of FI and conditional mean-squared error ratios for the GPC and CPC

In summary, the ratio of Fisher Information of the GPC and CPC with matched neuron number, the same conditional spiking statistics, and the same coding range grows exponentially with N whenever the coding range $R_l \gg \lambda$. This is true whether the two codes have neurons with equal lifetime sparseness (equal width tuning curves), or have equal per-network population sparseness (narrow tuning curve width in the CPC) in the phase representation. The results make apparent that the gains of the GPC are not due to sparse or dense coding considerations.

The calculated Fisher Information ratio and the corresponding conditional root-mean-squared error ratios for the GPC and CPC are consistent with the conditional mean square error ratio derived geometrically in the main paper and with the numerical estimates of the mean square error ratio obtained from sampling and decoding both types of codes with maximum likelihood.

4.5 Discussion: conceptual reasons for the exponential scaling of the GPC-CPC FI ratio

The exponentially strong performance of the GPC compared to the CPC is due to its interleaving property and the combinatorially large coding range discussed in the main paper (both of which are due to the periodic and multi-period representation of location by the GPC).

Another interpretation for why the FI of the GPC is in a different class than the CPC is that as R_l grows exponentially, each GPC network packs exponentially more periodically spaced peaks into its response. In contrast, the CPC has only a single peak that grows exponentially wider. However, periodicity of response with exponentially many periods in the range is not a sufficient condition for producing the FI of the GPC because of a lack of identifiability: it is not possible to discriminate between different periods of the periodic response. The authors in (3) proposed adding a monotonic tuning curve to bring identifiability to a periodic code. However, a monotonic tuning curve would have very low resolution over the large range R_l, and the resulting conditional mean-squared error will remain large (3). In the GPC, the multiperiodic representation enables identifiability over a very large range. Thus, the existence of periodic responses with exponentially more peaks packed into an exponentially large range, together with multiple different periods – produces the exponentially large FI compared to a CPC.

5 Optimal location readout from grid phase vector

If locations are to be estimated from the noisy phases, without knowledge about possible values of $x(t)$ (save that they fall within a range R_l), the optimal readout strategy for both the GPC and the CPC is maximum likelihood (ML) estimation:

$$\hat{x}(t) = \arg \max_{x(t) \in R_l} P\big(\vec{\phi}(t) \mid x(t)\big) \tag{30}$$

Explicit ML decoding as in Equation (30) is used to decode the noisy GPC and CPC phases in Figures 1, 3 of the main paper. In Figures 4 d-e, decoding is performed by the model hippocampal layer in the neural network. This network's readout approximates ML estimation (see SI section "Neural network model readout approximates ML decoding").

6 The model neural network's inference approximates ML decoding

Readout cells performing position inference in the network model, as described in the Methods, approximate the results of ML inference. We show this in what follows.

Substituting Methods Equations (16)-(17) and (9) for the grid cell-readout weights into Methods Equation (15) for readout cell activity, we get that

$$\begin{aligned} h_i(t) &= \sum_{j,\alpha} W_{ij\alpha} r_{\alpha j}(\vec{\phi}(t)) \\ &= \sum_{j,\alpha} \Big(\sum_{x'} G_{\sigma_h}(|x' - x_i^*|) f_{\sigma_e}(\underline{\phi}_\alpha(x'), \phi_j^*) \Big) f_{\sigma_e}(\phi_\alpha(t), \phi_j^*) \\ &\approx \sum_{\alpha} \int d\phi^* \; dx' \; G_{\sigma_h}(|x' - x_i^*|) \; f_{\sigma_e}(\underline{\phi}_\alpha(x'), \phi^*) \; f_{\sigma_e}(\phi_\alpha(t), \phi^*) \end{aligned} \tag{31}$$

where we have replaced the sum over preferred phases ϕ_j^* and locations x' into an integral in the third line. Substituting Equation (2) for the neural tuning curves, we therefore have

$$\begin{aligned} h_i(t) &\approx \sum_{\alpha} \int d\phi^* \; dx' \; G_{\sigma_h}(|x' - x_i^*|) \; e^{-2\kappa} \exp\big(\kappa \cos\big(2\pi(\phi_\alpha(x') - \phi^*)\big)\big) \exp\big(\kappa \cos\big(2\pi(\phi_\alpha(t) - \phi^*)\big)\big) \\ &= \sum_{\alpha} \int d\phi^* \; dx' \; G_{\sigma_h}(|x' - x_i^*|) e^{-2\kappa} \exp\left[2\kappa \cos\left(2\pi \frac{\phi_\alpha(x') - \phi_\alpha(t)}{2}\right) \cos\left(2\pi \frac{2\phi^* - \phi_\alpha(t) - \phi_\alpha(x')}{2}\right)\right] \\ &= \sum_{\alpha} \int d\theta \; dx' \; G_{\sigma_h}(|x' - x_i^*|) e^{-2\kappa} \exp\left[2\kappa \cos\left(2\pi \frac{\phi_\alpha(x') - \phi_\alpha(t)}{2}\right) \cos(2\pi\theta)\right] \end{aligned} \tag{32}$$

where the last equality follows by simple substitution. We expand the exponential term in the argument of the integral:

$$\begin{aligned} e^{\left[\cos(2\pi u)\cos(2\pi\theta)\right] 2\kappa} &\approx \left\{1 + \cos(2\pi u)\cos(2\pi\theta) + \frac{1}{2}\cos^2(2\pi u)\cos^2(2\pi\theta) + \cdots\right\}^{2\kappa} \\ &\approx \left\{1 + 2\kappa\cos(2\pi u)\cos(2\pi\theta) + 2\kappa^2\cos^2(2\pi u)\cos^2(2\pi\theta) + \cdots\right\} \end{aligned} \tag{33}$$

The second term in Equation (33) vanishes under $\int d\theta$, and $\int d\theta \cos^2(2\pi\theta) = 1/2$, so that

$$\begin{aligned} h_i(t) &\approx \sum_{\alpha} \int dx' \; G_{\sigma_h}(|x' - x_i^*|) e^{-2\kappa} \left[1 + \kappa^2 \cos^2\left(2\pi \frac{\phi_\alpha(x') - \phi_\alpha(t)}{2}\right)\right] \\ &= \sum_{\alpha} \int dx' \; G_{\sigma_h}(|x' - x_i^*|) e^{-2\kappa} \left[1 + \frac{\kappa^2}{2}\Big(1 + \cos\big(2\pi\big(\phi_\alpha(x') - \phi_\alpha(t)\big)\big)\Big)\right] \end{aligned} \tag{34}$$

Assuming the area under the place field tuning curves, $\int dx \; G(|x - x_i^*|)$, is independent of i, it follows that

$$\begin{aligned} \arg\max_i h_i(t) &\approx \arg\max_i \sum_{\alpha} \int dx' \; G_{\sigma_h}(|x' - x_i^*|) \cos\big(2\pi\big(\phi_\alpha(x') - \phi_\alpha(t)\big)\big) \\ &\approx \arg\max_i \sum_{\alpha} \cos\big(2\pi\big(\phi_\alpha(x_i^*) - \phi_\alpha(t)\big)\big) \\ &= \arg\max_i \left\{\log\left(\prod_{\alpha} e^{\cos\left(2\pi\left(\phi_\alpha(x_i^*) - \phi_\alpha(t)\right)\right)}\right)\right\} \end{aligned} \tag{35}$$

where the second approximate equality is obtained by assuming the readout place fields are narrow. Therefore,

$$\hat{x}_{net} = \arg\max_{x_i^*}\left\{\log\left(\prod_\alpha e^{\cos\left(2\pi\left(\phi_\alpha(x_i^*)-\phi_\alpha(t)\right)\right)}\right)\right\} \tag{36}$$

We are now in a position to compare the neural network inference of location with actual ML estimation. When phase noise is Gaussian and independent per network with noise variance $\sigma_\alpha^2 \ll 1$, actual ML location inference corresponds to solving

$$\arg\max_{x\in[0,R_l)}\prod_\alpha P(\phi_\alpha(t)|\underline{\phi}_\alpha(x)) = \arg\max_{x\in[0,R_l)}\prod_\alpha e^{\frac{1}{2\pi\sigma_\alpha^2}\cos\left(2\pi\left(\underline{\phi}_\alpha(x)-\phi_\alpha(t)\right)\right)} \tag{37}$$

Comparing Equations (36) and (37) for the network and ML inferences, we see that the network essentially optimizes the log of the ML cost function. One difference between the two is that in the network, the accuracy of the inferred location is limited by half the spacing between the discrete place cell centers x_i^*. Aside from this discretization of inferred values, the neural network readout approximates ML inference.

7 Derivation: how minimum distance (d_{min}) scales with neuron number

Here, we derive how the minimum separation of codewords and the maximum correctable errors scale with N, the number of grids, and M, the number of neurons per grid, for the grid cell code. The coding space is the N-dimensional phase hypercube of unit length (with periodic boundary conditions). The coding line for the range $[0, R_l)$, which on the torus is a continuous line, is a set of parallel line segments when plotted in the hypercube. Assume all the periods are approximately the same size, $\lambda_1 \sim \lambda_2 \sim ... \sim \lambda_N$. Assume the coding lines are parallel to the vector $\vec{1}$ (good approximation if all the periods are about the same size). We are interested in computing the spacing between these lines for a total range R_l of locations, assuming these parallel lines are "well-spaced": The $N-1$ dimensional manifold orthogonal to the parallel coding lines within the phase hypercube is pierced by the coding lines. We will assume the points where the N-1 dimensional space is pierced correspond to the centers of the spheres in a good sphere-packing solution in that space.

Total number Q of coding line segments: Depending on the intercept, the coding line segment lengths vary from $\sqrt{N}$ (along the main diagonal) down to 0 (the diagonal at a vertex).

Starting at an interecept $\vec{a} = (a_1, a_2, ..., a_N)$ and assuming without loss of generality that $1 \geq a_1 \geq a_2 \geq ... \geq a_N$, the line parallel to $\vec{1}$ emerges at the opposite side of the cube at $(1, a_2 + 1 - a_1, ..., a_N + 1 - a_1)$. The length of this line is therefore $\sqrt{N}(1 - a_1)$. The average length of a line in the hypercube then is

$$\langle L\rangle = \int_0^1 L(a_1)p(a_1)da_1$$

with $p(a_1)da_1 = da_1$ (uniform distribution of intercept a_1 along the unit-length side of the hypercube) and $L(a_1) = \sqrt{N}(1 - a_1)$. The integral yields $\langle L\rangle = \sqrt{N}/2$.

To cover a disance R_l in real space with parallel lines and uniformly spaced intercepts requires Q lines in the hypercube, where

$$Q = \frac{R_l}{\lambda\langle L\rangle} = \frac{2R_l}{\lambda\sqrt{N}} \tag{38}$$

Spacing between the Q coding line segments: All Q lines pierce at right angles the $N-1$ dimensional region, defined as the hyperplane perpendicular to the main diagonal of the phase hypercube at the point $\frac{\vec{1}}{2}$, where it intersects with the phase hypercube. Call this region the largest perpendicular hyperplane (LPH). Each of the Q coding line segments pierce this volume at a single point. If the Q points are well-spaced (located at the centers of the spheres in a good sphere-packing solution) the maximum-radius balls drawn around them will occupy a large and extensive fraction of the volume of the LPH,

$$QV_{ball} = \beta V_{LPH} \tag{39}$$

where $\beta \sim 0.8$. Inverting the volume formula of an n-dimensional ball to obtain its radius (using the formula for even n; the asymptotics for large n are the same if n is odd), we find

$$R_{ball} = \left[\frac{\left(\frac{n}{2}\right)! V_{ball}}{\pi^{\frac{n}{2}}}\right]^{\frac{1}{n}} \sim \sqrt{\frac{n}{2\pi e}} V_{ball}^{\frac{1}{n}}$$

where the asymptotic approximation is based on Stirling's formula: $n! \sim n^n e^{-n}$. Using $n = N-1$ and the formulae for $Q$and V_{ball} from Equations (38) and (39) above, we have

$$R_{ball} \sim \sqrt{\frac{n}{2\pi e}} \left(\frac{\beta V_{LPH}}{Q}\right)^{\frac{1}{n}} = \sqrt{\frac{N-1}{2\pi e}} \left(\frac{\beta\lambda\sqrt{N} V_{LPH}}{2R_l}\right)^{\frac{1}{N-1}}$$

The LPH has volume $\sqrt{\pi/6}$ (independent of N; see Equation (41) and derivation below). For $N \gg 1$, the radius of the non-intersecting balls containing each coding point in the $N-1$ dimensional hyperplane region is therefore

$$R_{ball} \sim N^{\frac{1}{2}} \sqrt{\frac{1}{2\pi e}} \left(\sqrt{\frac{\pi}{6}} \frac{\beta\lambda}{2R_l}\right)^{\frac{1}{N}}.$$

In other words, the minimum distance separating the coding lines in the phase hypercube, scales with N as

$$d_{min} = 2R_{ball} \sim \sqrt{N}.$$

Volume of the largest hyperplane perpendicular to the main diagonal (LPH): The hyperplane passing through the origin and perpendicular to $\vec{1}$, the main diagonal of the hypercube, is defined by the points $\vec{x}$ satisfying

$$\vec{x} \cdot \vec{1} = 0$$

The LPH is parallel to this hyperplane but passes through the point $\left(\frac{\vec{1}}{2}\right)$, so it is defined by

$$\sum_{i=1}^{N} (x_i - \frac{1}{2}) = 0 \rightarrow \sum_{i}^{N} x_i - \frac{N}{2} = 0$$

together with the constraint that $0 \leq x_i \leq 1$ for each i. It follows for any $x_j$that

$$\sum_{i \neq j}^{N} x_i - \frac{N}{2} + x_j = 0$$

The above equation and the constraint $0 \leq x_j \leq 1$ give

$$\frac{N}{2} - 1 \leq \sum_{i=1}^{N-1} x_i \leq \frac{N}{2},$$

with the remaining constraints: $0 \leq x_j \leq 1$ for all i. The volume of this constrained region is therefore given by

$$V_{LPH} = \int_{\frac{N}{2}-1 \leq \sum_{i=1}^{N-1} x_i \leq \frac{N}{2}} d^{N-1}\vec{x}$$

Based on this formula, V_{LPH} is known to be (Thomas Lam, private communication and (4)):

$$V_{LPH} = \frac{T\big(N-1, N/2\big)}{(N-1)!\sqrt{N-1}} \tag{40}$$

where $T(n,k)$ are the Eulerian numbers. We are interested in the asymptotic behavior ($N \to \infty$) of this volume. The asymptotic form of $T(N-1, N/2)$ for large N is (5):

$$T(N-1, N/2) \approx \sqrt{\frac{6}{\pi}} \frac{1}{\sqrt{N}} (N-1)!$$

Thus, for $N \gg 1$, the volume of the LPH is

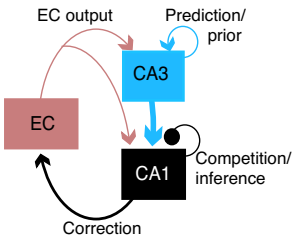
$$V_{LPH} \approx \sqrt{\frac{6}{\pi}} \frac{\sqrt{N}}{\sqrt{N-1}} \approx \sqrt{\frac{6}{\pi}} \tag{41}$$

which is independent of N.

An alternative derivation of the volume of this region comes from the following: The sought-after volume divided by $\sqrt{N}$, or $V_{LPH}/\sqrt{N}$, equals the probability that the sum of $N-1$ random variables, each distributed uniformly in $[0,1]$, falls in $\big[N/2-1, N/2\big]$. The mean of each of these variables is $1/2$, and the variance is $1/12$. By the Central Limit Theorem, when N is large the distribution of the sum approaches a Gaussian, and this probability becomes the probability that a Gaussian variable lies between $(-\sqrt{3/N}, \sqrt{3/N})$. This probability is $(1/\sqrt{2\pi})(2\sqrt{3/N})$. Multiplying by $\sqrt{N}$, we get the same result as Equation (41), that $V_{LPH} \sim \sqrt{6/\pi}$.

8 Mapping the model readout to the entorhinal-hippocampal circuit

This section contains a schematic figure of how the readout maps to the mammalian entorhinal-hippocampal network (details in the main paper's Discussion).



Supporting Figure 4: **Simple neural network model maps to the entorhinal-hippocampal circuit.** The full estimation circuit: Entorhinal cortex performs noisy path-integration with special coding. CA3 contributes contextual priors. CA1 performs inference on both its inputs through competitive dynamics, then corrects EC phase.

9 Decoder complexity and comparison of the GPC and CPC

9.1 Comparing GPC encoding against a CPC enhanced in size by GPC decoder neurons

In the main paper, we always consider CPCs that encode animal location using as many neurons as the GPC encoding. Here we consider the comparison between the GPC and CPCs, if the CPC includes as many neurons as the sum of all the GPC encoding networks as well as all the neurons used in the model readout network (decoder) for the GPC. For specificity, we call this larger CPC encoder an *enhanced-size* CPC.

The total number of neurons in the GPC is NM, where N is the number of grid networks, and M is the number of neurons per grid network. Assume the total number of readout neurons is N_r. We set $N_{CPC} = NM + N_r$, i.e., we allow the CPC the sum total of all neurons in the GPC networks and in the readout. The aim is to compute the error ratio of the GPC to the CPC codes with this adjustment.

Given a phase error of variance $\sigma_\alpha^2(\phi)$ in each grid network, the variance in the CPC network should be scaled according to its size relative to that of each grid network. Each grid network contains M neurons, but the CPC contains N_{CPC} neurons. If the tuning curve widths of the CPC (in phase), are kept the same as the tuning curve widths of the GPC, then $\sigma_{CPC}^2(\phi) = \sigma_\alpha^2(\phi)\frac{M}{N_{CPC}}$ (6; 7; 8; 9; 10; 11). But if the tuning curve widths are scaled so that they decrease as N_{CPC} grows, the optimal case, then $\sigma_{CPC}^2(\phi) = \sigma_\alpha^2(\phi)\left(\frac{M}{N_{CPC}}\right)^2$ (11). In all the calculations of the main paper, the exponential advantage of the GPC over the CPC persists for either scaling of the CPC tuning curve width. However, with the enhanced-size CPC, the GPC advantage over the CPC diminishes in the case of optimal CPC tuning curve width, i.e. when

$$\sigma_{CPC}^2(\phi) = \sigma_\alpha^2(\phi)\left(\frac{M}{N_{CPC}}\right)^2 = \sigma_\alpha^2\left(\frac{M}{NM+N_r}\right)^2. \tag{42}$$

Now we can directly write down the CPC and GPC location errors for a noise vector of total length a in the GPC. As in the main paper, the GPC phase squared error per network is

$$\sigma_{GPC}^2(\phi) = \frac{a^2}{N}. \tag{43}$$

As before, it follows that the fractional location error of the GPC (from Equation (5), main paper), is

$$\frac{\text{error}_{GPC}(x)}{R_l} \sim \frac{a}{\sqrt{N}}\,\frac{\lambda}{R_l}. \tag{44}$$

With $R_{CPC} = R_l$, the location error of the CPC is R_l times the phase error (Equation (42)). Thus the fractional location error is

$$\frac{\text{error}_{CPC}(x)}{R_l} \sim \frac{a}{\sqrt{N}}\frac{M}{NM+N_r}. \tag{45}$$

Taking the ratio of the two location errors (Equations (44), (45)), we have

$$\frac{\text{error}_{GPC}(x)}{\text{error}_{CPC}(x)} \sim \frac{\lambda}{R_l}\frac{NM+N_r}{M}. \tag{46}$$

This is the ratio of GPC and CPC fractional location errors. The same ratio results for the absolute location errors, because both codes are encoding the same range. Equation (46) is the relevant ratio to use if the aim is to compare specific numbers for a fixed R_l, N, and M.

To assign a number or a scaling to N_r, we need to assume a readout scheme. The scheme given in the main paper assumes a unary code for space in the readout, so that N_r scales linearly with the coding range. Thus, for that scheme, we assume the total number of readout neurons is $N_r = \frac{R_l}{\delta h}$ where R_l is the legitimate coding range, and δh is the spacing of centers in the readout population.

If the aim is to look for asymptotic scaling with $N \to \infty$ as ρ (the information rate) is kept constant, we must substitute $R_l \sim \lambda e^{\rho\beta N}$ based on the definition of information rate. Then, we have

$$\frac{\text{error}_{GPC}(x)}{\text{error}_{CPC}(x)} \sim e^{-\rho\beta N} \frac{NM + \frac{\lambda}{\delta h} e^{\rho\beta N}}{M}. \tag{47}$$

Taking the limit $N \to \infty$ with fixed ρ, we obtain

$$\frac{\text{error}_{GPC}(x)}{\text{error}_{CPC}(x)} \sim \frac{1}{M} \frac{\lambda}{\delta h}. \tag{48}$$

Clearly, the GPC does not now display an exponential advantage over the enhanced-size CPC. However, the GPC may still be better than the CPC. We examine what the ratio above means. M is a measure of the size of the *encoder*: it is the size of the individual grid networks. Plausible estimates of how many cells make up each grid network range from 1000-10000 [8,6]. The term δh is a measure of the size of the GPC *decoder*: the size of the decoder scales with δh: the finer the spacing between readout cell preferred locations, the larger the required readout cell population for a fixed range. If the number of neurons in each GPC network exceeds the inverse spacing of readout cell centers (in units of the typical grid period), then the GPC error is smaller than the CPC error.

To estimate the size of location errors for the GPC and enhanced-size CPC not just asymptotically but under specific and realistic parameter values, we consider some specific numbers. In our simulations of Figure 4 (main paper), for instance, $R_l = 300$ m, $\lambda_1 = 30$ cm and $\lambda_N = 74$ cm, $N = 12$ networks. All these are reasonable approximations for these quantities in actual rats. For the decoder (CA1), we used 3000 total neurons, to keep the spacing between centers at 10 cm. Finally, for M (neurons per grid cell network) we used only 50 neurons. However, this was done not because it is a good estimate of the number of neurons in the actual rat, but for computational expediency: our computers configured in parallel are still not fast enough and do not possess enough RAM to learn (or even load pre-learned values of) weight matrices for all the grid cell to all the readout cells, if the total number of grid cells (NM) and total number of readout cells exceeds 3000×3000. Nevertheless, putting these numbers with $M = 50$ into Equation (46), we get that $\text{error}_{GPC}(x)/\text{error}_{CPC}(x) \sim 1/10$. If, more correctly, we use plausible estimates for how many cells make up each grid network ($M \approx 1000 - 10000$ [8,6]), then we get that $\text{error}_{GPC}(x)/\text{error}_{CPC}(x) \sim 1/40$ or $\sim 1/50$. This is not very different from the result obtained by plugging the same numbers into Equation (7) of the main paper, in which we compared the GPC against a CPC not enhanced in size by the number of decoder neurons. From Equation (7), we get that the ratio is $\text{error}_{GPC}(x)/\text{error}_{CPC}(x) \sim 1/100$.

Thus, for finite, fixed R_l set to a plausible range of 300 m, and plausible values for other numbers, the GPC maintains a large advantage over the CPC even if the CPC is allocated the extra neurons required for reading out the GPC.

In the following, we further argue that the same readout should be used for both the GPC and the CPC, because the readout network does far more than simply decode the GPC, for both the GPC and the CPC. The readout network applied to CPC gives a bigger advantage to the CPC than when the neurons are used in the CPC itself without a readout. If that is the case, then the neuron-number cost of the readout should be considered the same for both coding schemes, and should not be charged against the GPC alone. Thus, the GPC advantage remains exponential.

9.2 The same (GPC) readout network can implement priors and improve location estimation for the CPC.

Above, we compared a GPC of NM total neurons with a separate readout of N_r neurons, against a CPC containing $NM+N_r$ neurons and no readout, which we called an enhanced-size CPC. However, we show numerically here that including a readout of N_r neurons for the CPC, to implement priors for location inference, can lead to better overall performance in the CPC than simply increasing by N_r the number of neurons in a CPC.

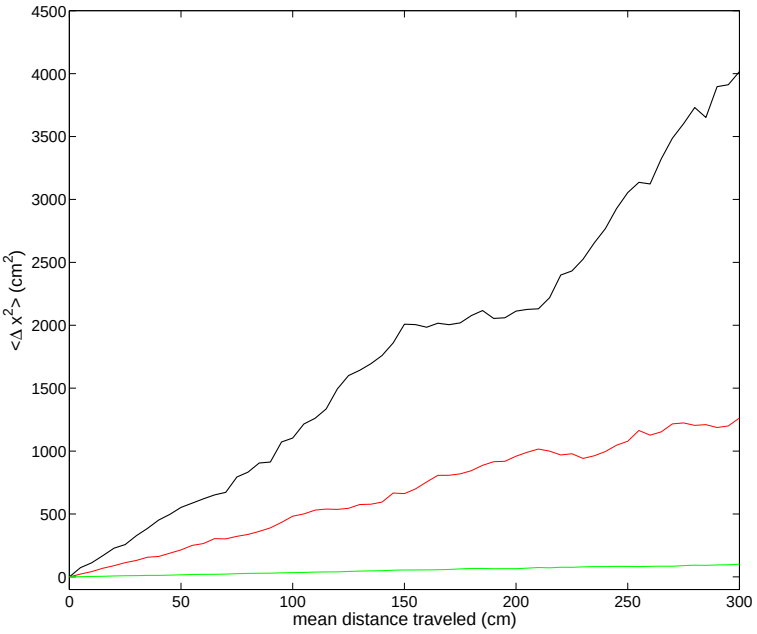
In the present example, the readout network for the GPC functions in the same way as in the main paper's Fig. 4e. Thus, the readout network applies excitatory biases to implement a prior and sums the GPC inputs, then finds the maximally active readout cell to infer location. The only difference is that this prior is more informative. The true trajectory is unidirectionally rightward, so instead of a ball prior (Methods), the readout network implements a half-ball prior: it biases inference in time-step $t + \Delta t$ toward readout cells with similar preferred locations to the inferred location from time-step t, but only in one direction, along the direction of movement. The true trajectory is advancing rightward with a maximum possible displacement of 10 cm (50 cm/s $\times$0.2 ms) per integration step, and the prior extends to place cells whose fields lie within 10 cm to the right of the previously winning place cell.

First, we show that the optimal enhanced-size CPC (with $NM + N_r$ neurons) performs less well than a CPC with NM neurons and a N_r neuron readout that's identical to the GPC readout. We assume that the enhanced-size CPC has a phase noise variance that is smaller than the phase noise variance in individual grid networks by $M/(NM + N_r)^2$, in other words, we assume that the CPC is optimal. Similarly, the NM neuron CPC is assumed to have a noise variance of size $1/N^2$ that of individual grid networks. Second, we show that this CPC with readout performs less well than the GPC with identical numbers of encoding neurons and identical readout. The results are numerical, and shown in Figure 9.2.

Why did the prior have to be implemented in a separate stage than the GPC or CPC encoding? The reason is that the CPC and GPC encoding stages are assumed to be performing the dynamics of path integration, i.e. converting velocity inputs into location estimates. Path integrators must be free of location biases to be able to faithfully update location state based on velocity input: location biases would tend to bias the integration itself adversely. Specifically, neural network integrators must have dynamics that are translationally invariant to location. Thus, any location-biasing priors must be implemented in a separate stage that weighs the unbiased (albeit noisy) location estimates with the prior to make an accurate inference about location. Thus, implementation of priors whether in the GPC or a CPC must happen in a separate, readout stage.

To summarize, we have shown that the proposed readout network has added functionality, beyond simply decoding the GPC. This added functionality includes the ability to implement priors and use these priors in combination with the noisy path integrated input from either the CPC or the GPC, to perform iterative inference. If the prior is sufficiently informative, iterative location inference with the CPC and the readout can be significantly more accurate than if the readout neurons were simply absorbed into the CPC to build a bigger optimal CPC without a readout. Thus, the readout network can be very useful for the CPC and the GPC, and should be considered as a common cost to both. In this case, the best comparison is to compare only CPC and GPC encoders, assuming both are read out by the same readout network. The CPC and GPC encodings and associated neuron numbers are exactly what are compared in the main paper, and reveal an exponential advantage of the GPC over the CPC.

Finally, although we have proposed a specific decoder in the main paper, consisting of place cell-like units with a unary place code, it is not guaranteed to be a unique or optimal decoder. Indeed, it is likely *not* the optimal decoder in terms of number of neurons used. A readout network with a sparse but combinatorial code for space, if it exists, would require far fewer neurons, and thus be far more efficient from that point of view, not requiring exponentially many neurons for decoding and



Supporting Figure 5: **Comparison of the GPC and CPC with priors, and an enhanced-size CPC without priors.** Mean squared error over simulated 300 cm trajectories (100 trials), versus displacement, similar to Fig. 4d-e of the main paper. Green: GPC ($N = 10$ networks with $M = 1000$ neurons each) with a directional and continuity (DAC) prior. Red: CPC ($NM = 1000$ neurons) with DAC prior.The GPC and CPC with prior have a prior implemented by a readout network of $N_r = 1000$ readout neurons that implement the directional prior. Black: enhanced-size CPC (i.e., CPC with as many neurons as in the GPC and its decoder, or $NM + N_r = 2000$ neurons), without prior. The coding range in all cases is R_l. The readout network's implementation of a DAC prior improves the performance of the CPC (red), compared to the case of the enhanced-size CPC (black), which has as many total neurons as the CPC encoder and readout networks: The slope of the trajectory improves from ≈ 11.67 to ≈ 4.17, resulting in half the of the cumulative squared error over time. The GPC with prior, using the same number of total (encoding and readout) neurons, has more than a 10-fold advantage over either of the CPCs (slope of squared error curve is 0.333). Parameters — GPC periods: $\lambda_1 = 10$ cm, with 4 cm increments for each of the rest. $R_l = 1000$ cm in all cases. CPC with prior: $\sigma_p = 1$ cm tuning curve widths and centers spaced uniformly over R_l. Enhanced-size CPC without prior: $\sigma_p = 0.5$ cm tuning curve widths, with spacing of 0.5 cm between preferred location centers. Trajectory: as in Fig. 4 of the main paper, the true trajectory advances with an average speed of 25cm/s, and the integration time step is $dt = 200$ms. Noise: In each network, noise is truncated Gaussian (as in Methods of Fig. 4), and is added per time-step of the simulated trajectory. Each grid network has a noise standard deviation of $\sigma_\alpha = 0.165$. The CPC with prior has σ_α/N. The CPC without prior has truncated Gaussian phase noise $\sigma_{CPC} = \sigma_\alpha/2N$. Readout cells: tuning curve width 1 cm, centers spaced evenly over R_l. $R_l = 1000$ cm.

implementing priors and inference over an exponentially large range. Indeed, some studies suggest that the number of place fields may scale with the size of the environment, hinting at a sparse but combinatorial code for large spaces (12). It largely remains to be shown whether and how such a

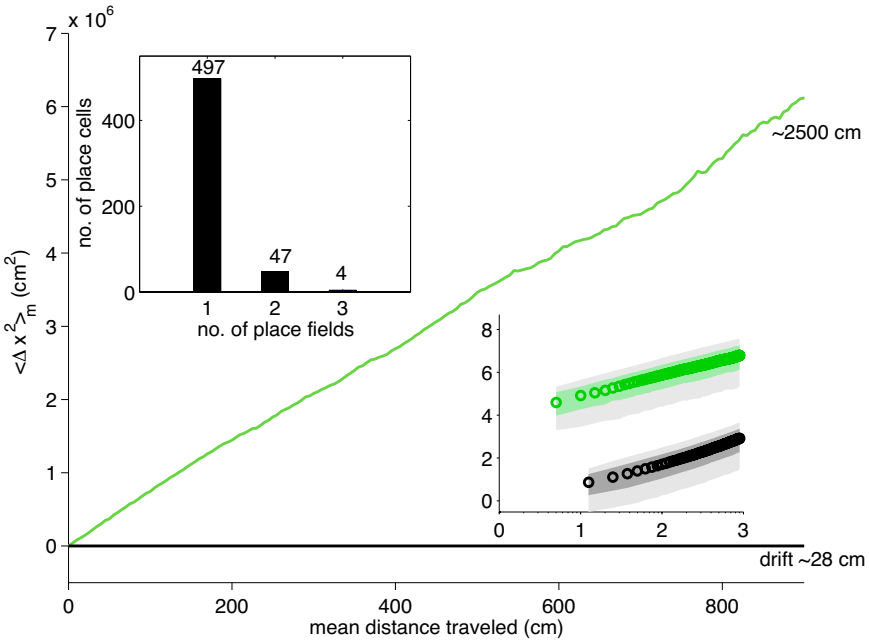
readout would work, but in the following section we show that it is possible to decode the GPC even if the decoder cells have more than one field.

10 Performance of the error-correction loop when readout cells have multiple fields

According to some experiments (12; 13; 14), individual place cells in CA1 could have multiple firing fields in a single environment. Thus, place cell encoding of locations might plausibly be an ensemble code rather than a dedicated single-location place code. To explore the effect of this possibility on error correction, we extended the network simulations of Figure 4 (main paper) to assign the readout cells a distribution of non-zero numbers of place fields. Locations, discretized into 10 cm bins over the legitimate range $R_l = 1500$ cm, were each assigned an ensemble of K place cells, drawn randomly (with replacement) from a pool of N_p cells. Thus, each location was represented by an ensemble code, and each readout cell could have multiple place fields. We chose $K = 4$ and $N_p = 3000$, which gave rise to an exponential tail in the probability that each readout cell had Z fields for $Z > 1$. This exponential distribution agrees qualitatively with the histogram shown for the number of place fields for CA1 place cells in (12) (Figure 3B, panel 2 of (12)).

In one sample of this process, the number of readout cells coding for at least one location along the range was 549. The maximum number of preferred locations in single readout cells was 3. During the training phase, we normalized the instantaneous activation of each readout cell having one or more preferred locations by the total number of its preferred locations in R_l. Post-learning, the top K cells, defined as the K cells receiving the highest input drive from the grid cells, were selected as the "winners" of a putative K-winners-take-all operation in the readout layer. The instantaneous firing rates of these winner cells were set to 1, while the rest of the readout cells' firing rates were set to zero. Given the activation of the readout layer, grid cells were corrected in exactly the same way as in the main paper, where readout cells had single preferred locations. In short: we first computed the input received by grid cells at the return projections from the readout layer, and we next assumed that the maximally activated grid cells in each network determined the corrected phases of each network, in the sense that the response of the corrected grid networks was now the regular response pattern of that network, centered on grid cells receiving maximal drive from the readout layer.

With the same number of grid networks (and the same response periods) as used in Figure 4 of the main text, the performance of the error correcting network was significantly worse; this was attributable to the decline in the uniqueness of the ensemble code for locations. To offset this decline, we increased the number of periods from $N = 12$ to $N = 15$ and increased the spacing between consecutive periods to 8 cm (as before, $\lambda_1 = 30$ cm). The time-grain of the ensemble readout code, and thus the interval between error correction steps, is 500 ms. Waiting longer between error correction steps allows the animal location to evolve to a more distant location before its position estimate is corrected again. The increase in distance between contiguous steps drives activation of more distinct populations of readout cells. Phase noise variance per unit time was consistent with Figure 4d (main paper), with $\sigma_\alpha = \sqrt{2.5} \times 0.033$. The extra factor of $\sqrt{2.5}$ in the noise standard deviation here is to compensate for the larger time-grain used here, to keep the noise variance per unit time the same as in Figure 4d, main paper. The resulting location estimates from using an ensemble place code in the readout, Figure S6, have a median error of 28 cm after 900 cm trajectories, comparable to the results with a dedicated place code. Although dedicated place coding gives the best results for a given set of grid cell networks and given noise, the results shown here indicate that error-correction is nevertheless possible even in the presence of ensemble-like features of the readout place code, albeit at the cost of requiring more grid cell networks to achieve a similar accuracy.



Supporting Figure 6: **Path integration performance over a trajectory when each readout cell can have multiple preferred locations (ensemble coding of location in the readout).** The root-median-squared error at the end of a set of 1000 trajectories of length $\sim$ 900 cm (black) is about 28 cm, similar to Figure 4d (for a dedicated place code in the readout). For reference, the median square error of CPC estimated trajectories (green) is also shown (same as in Fig 4d). The inset on bottom-right shows the same median curves on a log-log plot. The shaded areas around each median curve represent where 25-75% and 10-90% of the trials lie, respectively. The inset on top-left is a histogram of the number of readout cells with a given number of place fields (preferred locations). Overall, there were 549 cells with at least one place field. For details, see Section 10.

11 Partial error correction with sparsely allocated readout cells

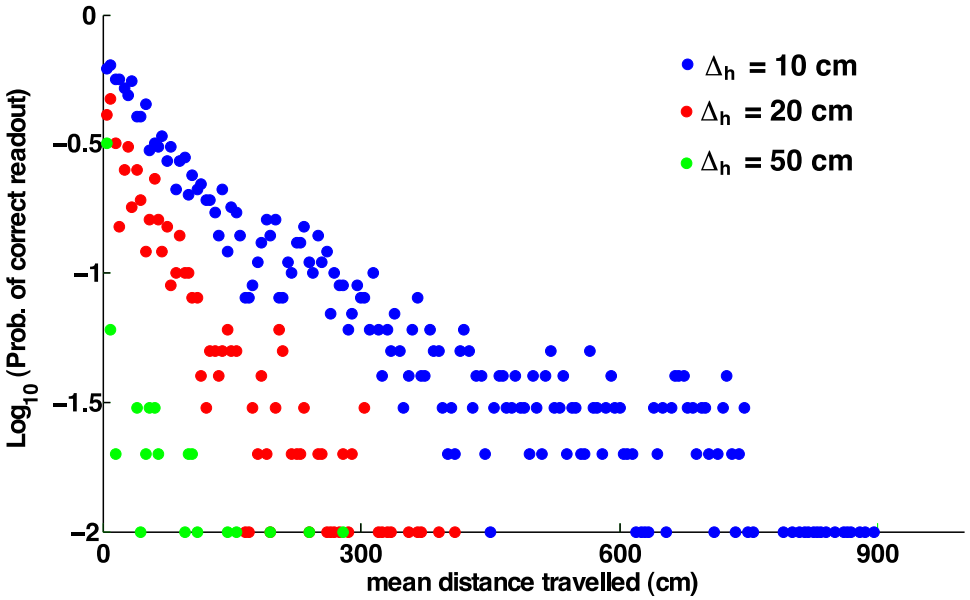
We study the effect of reducing the density of readout cell place fields within R_l. As the density of place cells decreases, many more locations are far from any readout field center. Then it is not meaningful to select the maximally activated readout cell and equate the inferred location with that readout cell's preferred location. Thus, for sparsely allocated readout cells, the readout only identifies a winning place cell if its activation exceeds a fixed threshold, κ. If the threshold is exceeded, error correction proceeds as before. If the threshold is not exceeded, there is no readout, and the grid cell activity is allowed to evolve without feedback correction from the readout layer.

For the simulations described here, $R_l = 3000$ cm, $\sigma_h = 6.0$ cm, and $\sigma_\alpha = 0.033$ for all αs. The simulation time step is $\Delta t = 200$ msec and the top speed of the animal is $v_{max} = 50$ cm/sec. Statistics are performed over 100 simulated trajectories.

Grid cell-readout weights are trained as described before. Readout cell centers are spaced at: $\Delta_h = 10, 20$, or 50 cm and the threshold $\kappa \approx 95\%$ of the peak grid cell drive to each readout cell. The value of κ was selected to minimize error in the case of sparsest readout cell allocation ($\Delta_h = 50$ cm) in the absence of grid cell noise. Lower values of κ allowed readout cell activation when the animal location was far from any the preferred location of any readout cell.

The results are shown in Figure S7. Plotted is the probability of an erroneous readout (conditioned on some readout cell being activated). Clearly, the fidelity of integration decreases as readout cell density decreases. There are fewer opportunities for error-correction when there are more gaps between readout cell centers. In the case of $\Delta_h = 50$ cm, individual grid cell phases have completely decohered (i.e., phases differ from the true phase by ≥ 0.5) by the time the simulated animal first encounters a readout cell's field, after the origin of the trajectory. Subsequent readouts are almost always wrong because the total error amplitude has exceeded $d_{min}/2$.

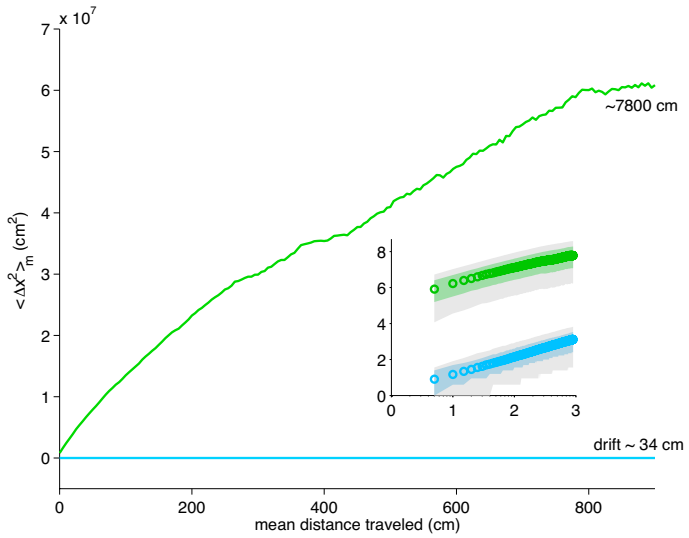
Adding a contextual ball prior of radius $B_{v_{max}}$ did not improve the performance in the cases with $\Delta_h = 20$ cm or 50 cm (not shown). In both cases, $B_{v_{max}} < \Delta_h$, and therefore the choice of readouts available to MAP inference is confined to the single readout cell which lies within the prior ball. Thus the prior, if strong enough to have an effect, forces the readout at a given time step to be identical to the one in the preceding time step, causing a rounding error. We also tested the effect of an expanding fading prior consisting of two gaussian peaks; given $\hat{x}(t)$, the prior for $t' > t$ is given by $A_0 G_{\sigma_{prior}}(\hat{x}(t) \pm (t'-t)v_{avg})$, where $G_\sigma(x-\mu)$ is a normalized Gaussian with mean μ and variance σ^2, and v_{avg} is the average speed of the animal for any run between turns. The initial amplitude A_0 of the gaussian peaks relative to the activation threshold κ of readout cells determines the outcome of the simulations. For low to intermediate initial amplitudes ($0.07\kappa - 0.7\kappa$), even with the additional drive due to the prior, the net drive to readout cells quickly falls below the threshold within a few time steps, sharply reducing the number of readouts. With large initial amplitudes ($> 0.7\kappa$), the prior is strong and it forces the subsequent (after one time step) readout almost deterministically towards a neighboring readout cell in the direction of animal motion, even if the true location is actually slightly closer to the center location. This results in the accumulation of rounding errors. In our numerical simulations we equate the inferred location with the preferred firing location of the maximally active readout cell (if its activation exceeds the threshold κ). If the readout mechanism were also to take into account the analog firing rate of the maximally active readout cell, position could be more precisely determined within its firing field, thereby reducing the rounding errors and the number of incorrect readouts.



Supporting Figure 7: **Effect of sparse readout cell allocation on the ability of the grid cell-readout network to correct errors.** See Section 11 for details.

12 Robustness of error correction with imperfect grid cell-readout and readout-grid cell weights

For error correction in the main paper, we form a weight matrix between grid cells and the readouts using Hebbian learning based on noise-free activations in both layers. We assumed that the return connections from the readouts to the grid cell layer are the transpose of the forward connections, by assuming the Hebbian rule to be symmetric. How important is it for these conditions to be exactly satisfied? Because of the high convergence of weights from grid cells into single readout cells, we expect that perturbing the weights randomly about their ideal values should not degrade performance significantly, because of the advantages of averaging. Similarly, because the grid cell network is assumed to be a continuous attractor network that, when reset by the return inputs from the readout layer, takes on an attractor state closest to the reset, we expect that any random deviations in the weights to the grid cells should not introduce biases, and the system should be robust to such deviations in weight.



Supporting Figure 8: **Trajectory estimation in the presence of perturbations and deletions of synaptic weights in the network model.** The root-median-squared integration error at the end of a set of 100 trajectories of length $\sim$ 900 cm each (cyan) is about 34 cm in the presence of an added phase noise per integration time-step, with $\sigma_\alpha = 0.165$. The setup is identical to that of Figure 4e of the main paper, except that the grid cell-readout weights in the forward and return directions are perturbed and subject to deletion. (Details in Section 12.) The comparable CPC trajectory, reproduced from Figure 4e, is shown for comparison (green). The inset shows the log-log plot of the same. The shaded areas surrounding each median curve indicate where 25-75% and 10-90% of the trials lie, respectively.

We investigated the actual robustness of the error-correction process numerically, by assessing path integration performance over a simulated trajectory. The simulations were identical to those of Figure 4e in the main paper, but 5% of the grid cell-readout weights and 5% of the readout-grid cell weights were deleted. Additionally, every non-zero (forward and return) non-zero weight was perturbed by an independent zero-mean Gaussian noise term with standard deviation set to 2% of each weight's original value. At the end of a 900 cm trajectory, with locations estimated by noisy

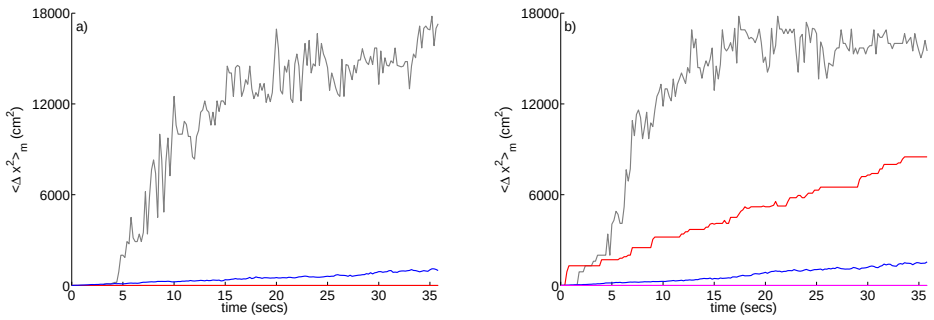
path integration and self-correction within the grid cell-readout loop starting from the beginning of the trajectory, the median error is 34 cm, Figure S7. This result is virtually identical to the error shown in Figure 4e of the main paper, indicating that the error correction process is robust to statistical perturbations of the weights away from ideal.

13 Results generalize to 2-dimensional space

The exponentially better relative performance of the GPC compared to the CPC does not depend on dimension, as we saw analytically in the SI section "Fisher information of the GPC and the CPC".

Numerical tests in dimension > 1 are difficult to perform over large coding ranges R_l because of the explosion in the size of the neural and weight vectors, so we numerically tested the performance of the grid cell-readout error-correcting network in 2-d space, using the full 2-d responses of grid lattices, over smaller regions ($\sim$ 3 m per side). Because the range R_l is substantially reduced, the relative advantage of the GPC over the CPC, though still very much present, is smaller. For these simulations, $\Delta t = 200$ msec, and $M = 32^2$ neurons in each grid network. We performed two sets of simulations, with $N = 9$ and $N = 5$ grid networks respectively. The smallest grid period is 30 cm and subsequent ones are spaced regularly at intervals of 4 cm. Neural activations are given by shifting a template 2-d grid pattern by an amount equalling the 2-d phase of the network. Each phase component is computed from the animal location and the phase noise model through Equation (2) of the main paper. The phase noise per time-step is again Gaussian, with zero mean and standard deviation $\sigma_\alpha = 0.039$ per phase component. This value corresponds to the translational drift observed in a continuous attractor network model of grid cells with 32×32 poisson neurons (CV=1) with periodic boundary conditions (15). The 2-d readout network has $M_h = 900$ place cells, and the arena size is 300 cm $\times$ 300 cm. Because of limits on computational speed and memory, this is the maximal arena size we simulate.

A single trial lasts 36 seconds, and the trajectory of the animal is given by $v = 0 \forall t$ (i.e., the animal is stationary). In all, 100 such trials are simulated. As in the 1d simulations (Figure 4), the readout of position by place cells is recorded both in the presence and absence of the error correcting loop. Shown in Figure S9 is the median squared error in the position estimated by the grid cell-readout network under these different conditions (corrected - red, uncorrected - gray). With $N = 9$ grid networks, (Figure S9 a), the error-correcting loop gives essentially perfect results i.e. in each trial, every place cell readout over the entire duration of 36 secs is correct. In contrast, without correction, the estimate of position drifts away from its true value as expected, with drift at the end of 36 secs being as high as 131 cm. The corresponding single-population response (blue) yields an error of about 31 cm over 36 secs, lower than that obtained without correction, but far worse than the performance achieved by the grid cell-readout error-correcting loop. With fewer lattices, $N = 5$, the performance of the error-correcting loop is worse, as expected (red). However, essentially perfect performance is recovered with the addition of a contextual prior (magenta).



Supporting Figure 9: **Median-squared-error in position estimation as a function of time for a stationary animal in a 2d enclosure of size** 300x300 cm^2**.** (See Section 13 for details.) (a) Using $N = 9$ lattices, the performance of the GPC (red curve) is essentially exact for the entire simulated time interval. The CPC estimate (blue) is much better than the uncorrected GPC estimate (black) but worse than the corrected GPC estimate. (b) If fewer lattices, $N = 5$, are used, the correction algorithm requires the addition of a contextual prior to curb the errors in position estimation. GPC performance with ML correction (no prior, red curve) is now worse than the CPC (the uncorrected GPC is still the worst (black)). The GPC with a contextual local ball prior (magenta) produces essentially perfect estimation.

References

[1] Fiete, I. R., Burak, Y. & Brookings, T. What grid cells convey about rat location. *J Neurosci* **28**, 6856–6871 (2008).

[2] Montemurro, M. A. & Panzeri, S. Optimal tuning widths in population coding of periodic variables. *Neural Comput* **18**, 1555–1576 (2006).

[3] Bethge, M., Rotermund, D. & Pawelzik, K. Optimal short-term population coding: when fisher information fails. *Neural Comput* **14**, 2317–2351 (2002).

[4] Sloane, N. J. A. The on-line encyclopedia of integer sequences. URL `http://www.research.att.com/ njas/sequences/`.

[5] Giladi, E. & Keller, J. Eulerian number asymptotics. *Proceedings of the Royal Society: Mathematical and Physical Sciences* **445**, 291–303 (1994).

[6] Seung, H. S. & Sompolinsky, H. Simple models for reading neuronal population codes. *Proc Natl Acad Sci U S A* **90**, 10749–10753 (1993).

[7] Abbott, L. F. & Dayan, P. The effect of correlated variability on the accuracy of a population code. *Neural Comput* **11**, 91–101 (1999).

[8] Sompolinsky, H., Yoon, H., Kang, K. & Shamir, M. Population coding in neuronal systems with correlated noise. *Phys Rev E Stat Nonlin Soft Matter Phys* **64**, 051904 (2001).

[9] Latham, P. E., Deneve, S. & Pouget, A. Optimal computation with attractor networks. *J Physiol Paris* **97**, 683–694 (2003).

[10] Brunel, N. & Nadal, J.-P. Mutual information, fisher information, and population coding. *Neural Computation* **10**, 1731–1757 (1998).

[11] Zhang, K. & Sejnowski, T. Neuronal tuning: to sharpen or broaden? *Neural Computation* **11**, 75–84 (1999).

[12] Fenton, A. A. *et al.* Unmasking the ca1 ensemble place code by exposures to small and large environments: more place cells and multiple, irregularly arranged, and expanded place fields in the larger space. *J Neurosci* **28**, 11250–11262 (2008).

[13] Barry, C. *et al.* The boundary vector cell model of place cell firing and spatial memory. *Rev Neurosci* **17**, 71–97 (2006).

[14] Ludvig, N. Place cells can flexibly terminate and develop their spatial firing. a new theory for their function. *Physiol Behav* **67**, 57–67 (1999).

[15] Burak, Y. & Fiete, I. R. Accurate path integration in continuous attractor network models of grid cells. *PLoS Comput Biol* **5**, e1000291 (2009).