Explaining heterogeneity in medial entorhinal cortex with task-driven neural networks

Aran Nayebi[1,*], Alexander Attinger[2], Malcolm G. Campbell[2], Kiah Hardcastle[2], Isabel I.C. Low[1,2,7], Caitlin S. Mallory[2], Gabriel C. Mel[1], Ben Sorscher[4], Alex H. Williams[6,7], Surya Ganguli[4,7,8], Lisa M. Giocomo[2,7], and Daniel L.K. Yamins[3,5,7]

[1]Neurosciences Ph.D. Program, Stanford University
[2]Department of Neurobiology, Stanford University
[3]Department of Computer Science, Stanford University
[4]Department of Applied Physics, Stanford University
[5]Department of Psychology, Stanford University
[6]Department of Statistics, Stanford University
[7]Wu Tsai Neurosciences Institute, Stanford University
[8]Facebook AI Research, Facebook, Inc.
[*]Correspondence: anayebi@stanford.edu

Abstract

Medial entorhinal cortex (MEC) supports a wide range of navigational and memory related behaviors. Well-known experimental results have revealed specialized cell types in MEC — e.g. grid, border, and head-direction cells — whose highly stereotypical response profiles are suggestive of the role they might play in supporting MEC functionality. However, the majority of MEC neurons do not exhibit stereotypical firing patterns. How should the response profiles of these more "heterogeneous" cells be described, and how do they contribute to behavior? In this work, we took a computational approach to addressing these questions. We first performed a statistical analysis that shows that heterogeneous MEC cells are just as reliable in their response patterns as the more stereotypical cell types, suggesting that they have a coherent functional role. Next, we evaluated a spectrum of candidate models in terms of their ability to describe the response profiles of both stereotypical and heterogeneous MEC cells. We found that recently developed task-optimized neural network models are substantially better than traditional grid cell-centric models at matching most MEC neuronal response profiles — including those of grid cells themselves — despite not being explicitly trained for this purpose. Specific choices of network architecture (such as gated nonlinearities and an explicit intermediate place cell representation) have an important effect on the ability of the model to generalize to novel scenarios, with the best of these models closely approaching the noise ceiling of the data itself. We then performed *in silico* experiments on this model to address questions involving the relative functional relevance of various cell types, finding that heterogeneous cells are likely to be just as involved in downstream functional outcomes (such as path integration) as grid and border cells. Finally, inspired by recent data showing that, going beyond their spatial response selectivity, MEC cells are also responsive to non-spatial rewards, we introduce a new MEC model that performs reward-modulated path integration. We find that this unified model matches neural recordings across all variable-reward conditions. Taken together, our results point toward a conceptually principled goal-driven modeling approach for moving future experimental and computational efforts beyond overly-simplistic single-cell stereotypes.

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

1 Introduction

From exploring new areas, planning shortcuts, and returning to remembered locations, the ability to self-localize within an environment subserves a range of navigational behaviors that are essential for survival. The hippocampus (HPC) and medial entorhinal cortex (MEC) are known to contain cells that encode the position of an animal by displaying firing fields with strikingly regular response patterns, influenced by self-motion [O'Keefe and Dostrovsky, 1971, Hafting et al., 2005, Kropff et al., 2015, Solstad et al., 2008, Sargolini et al., 2006]. For example, MEC grid cells possess characteristically symmetric and periodic tuning curves, yielding hexagonal arrays of neural activity over physical space. Additionally, border cells that fire maximally near environmental boundaries and head direction cells that fire only when an animal faces a particular direction are among the other MEC cell types with interpretable tuning curves.

However, a large fraction of the MEC population have unconventional and heterogeneous tuning to navigational variables and are less obviously well-described in terms of simple, stereotypical tuning patterns [Hinman et al., 2016, Hardcastle et al., 2017]. How can we characterize what these other, more heterogeneous, populations of cells do? Are they critical to the abilities of the animal in real-world memory tasks? And if so, how is their role distinct from the more stereotypical grid-like cells? Here, we take a quantitative computational modeling approach to answering these questions.

To start with, we address the question of whether there is a phenomenon to model in the first place, and how one would measure model accuracy quantitatively. To this end, we identify the *similarity transform* between neural populations in different animals — that is, a mapping which takes neuronal population vectors in a "source" animal and maps it to corresponding neuronal population vectors in a "target" animal. To the extent that robust similarity transforms can be found that match up MEC population responses across multiple animals measured in multiple experimental conditions, then a reliable pattern of neural response behaviors (across conditions) has been identified. This identification strategy is well-defined even when there is no known *a priori* taxonomy of functional response types. We find here that with the right mapping class (mostly Ridge-like linear regression), the inter-animal consistency of MEC neuronal responses is in absolute terms very high (> 0.8), and in relative terms just as high as for heterogeneous cells as for more stereotypical grid cells.

Using this same similarity transform, we then evaluate the ability of each of multiple computational models to explain response variance of MEC neurons, treating each candidate model as a potential "source animal" and measuring how well it maps to each target real animal. We look to the recent literature to identify potential candidate models. Over the past several decades, collaborations between experimental and computational neuroscientists have led to the formulation of dynamical-systems models of grid cell formation, and helped illustrate possible functional roles for grid cells in supporting path integration-based hippocampal place cells [Skaggs et al., 1992, Zhang, 1996, Fuhs and Touretzky, 2006, Burak and Fiete, 2009]. These powerful models make a number of non-obvious predictions about MEC neural properties, some of which have been confirmed in subsequent experimental work [Ocko et al., 2018, Campbell et al., 2018]. Despite their success, such models are limited in their explanatory scope, hand-designed to capture the properties of one stereotypical cell-type class (e.g. grid or border cells) at a time, or combinations of several cell types via multiple dedicated type-specific modules [Couey et al., 2013, Yoon et al., 2013].

A potential solution to this problem arises out of recent work creating learned neural networks that achieve path integration [Cueva and Wei, 2018, Banino et al., 2018]. Intriguingly, these models have been found to contain internal units that resemble grid cells, suggesting that such stereotypical cells embody a computational solution to path integration that naturally arises from satisfying an end-to-end functional constraint. Recent work has demonstrated that the underlying mathematical reason for this fact is due to pattern forming dynamics under a nonnegativity constraint [Sorscher et al., 2019, 2020]. Intriguingly, in addition to having units that resemble stereotypical grid or border cells, these learned neural networks also naturally possess a wide variety of other less easily described unit types, raising the possibility that these artificial "heterogeneous" units might be somehow resemble the actual heterogeneous cells making up the majority of real MEC populations.

Motivated by these ideas, we generate a wide variety of candidate model networks by varying architectural structure and end-to-end optimization objectives, each expressing a different hypothesis for MEC circuit structure and function. Architecturally, we formulate variants based on using different types of nonlinearities and different types of local recurrent memory circuits (e.g. RNNs [Elman, 1990], UGRNNs [Collins et al., 2017], GRUs [Cho et al., 2014], and LSTMs [Hochreiter and Schmidhuber, 1997]). From a task point of view, we test both simple dimensionality reduction [Stachenfeld et al., 2014, Dordek et al., 2016] as well as place cell mediated vector path-integration [Banino et al., 2018] and direct position estimation [Cueva and Wei, 2018].

Our core result is that there is substantial variation in the models' abilities to explain MEC responses, especially those of the heterogeneous cells, as a function of a model architecture and task. Some models, such as the classic "Grid Cell" model based on low-rank decomposition of place cell fields, do a reasonable job explaining grid cell responses but are quite poor at explaining most other neurons. Task-optimized learned models typically do better, especially those optimized for place cell-mediated vector path-integration. The best model – with memory-gated rectified nonlinearities – essentially *solves* the neurons, capturing nearly 100% of the noise ceiling of the data. This is of substantial interest, given that the nonlinear components of this model are not directly optimized to match neural data (just to solve the task), and given that a series of strong control models capture much less of the MEC neural variability. This same model also best generalizes to a variety of novel experimental conditions, and has the best match to the empirical data on a grid score distribution metric.

With this predictive model in hand, we then begin to address our second core question: what is the functional role of heterogeneous neurons? We generate several results suggesting that heterogeneous neurons are important for path integration, including cell-type specific virtual knockout experiments, in which we compare performance degradation when deleting grid and/or border cells as compared to heterogeneous units. Overall, we find that models are quite robust to knockouts, and differences between heterogeneous and stereotypical cell knockouts are very small, suggesting that stereotypical cell-types may not be especially more functionally important than units with less easily-characterized response profiles.

Building on the above results, we extend models to encompass MEC cell responses as a function of reward as well as spatial position, introducing a simple modeling paradigm that performs reward-modulated foraging in the context of the path-integration task. We find that this unified task-optimized model matches neural responses across all reward and spatial conditions, and that a reward-response mechanism at an intermediate point on the explore-exploit continuum best explains neural responses. Taken together, our results suggest how specific processes of biological performance optimization may have directly shaped the neural mechanisms in MEC as a whole, and provides a path for enlarging the study of MEC beyond overly-restrictive response stereotypes.

2 Reliability of heterogeneous cell response profiles

What firing patterns of MEC cell populations are common across multiple animals, and thus worthy of computational explanation? This question is comparatively straightforward for stereotypical MEC cells, because the very presence of these stereotypical features (e.g. hexagonal grids of a given orientation and spatial frequency) allows the definition and measurement of observables that arise reliably across trials and animals (e.g. the grid score distribution). But how can this be done generally for populations in which one does not have a prior characterization of what each cell encodes? We take inspiration from methods that have proven useful in modeling visual, auditory, and motor cortex [Yamins and DiCarlo, 2016, Kell et al., 2018, Michaels et al., 2020]. Specifically, we aim to identify the *narrowest* class of similarity transforms needed to map the firing patterns of one animal's MEC population to that of another (Fig. 1a). As with other cortical areas, this transform class likely cannot be so strict as to require fixed neuron-to-neuron mappings between MEC cells, since even within the same animal at different times, MEC and HPC populations can undergo remapping that shift cell responses across the population [Farhoodi et al., 2020, Low et al., 2020]. However, the transform class for MEC also cannot be so loose as to allow a completely unconstrained linear mapping, since the highly structured response patterns of stereotypical MEC cell types such as grid cells may not be guaranteed to be preserved under arbitrary linear transforms.

To identify this transform class, we utilize data collected from electrophysiology in 12 awake behaving mice ($n = 620$ cells) performing open foraging for randomly scattered crushed cereal in a $100cm^2$ 2D arena [Mallory et al., 2021]. We explore a variety of mapping transform classes between the population rate maps of these neurons (Fig. 1b). As a baseline we evaluate a strict one-to-one mapping transform, in which each target unit is mapped to the single most correlated unit in the source animal. We also evaluate more powerful linear transforms, including Lasso, Ridge, and ElasticNet regression,

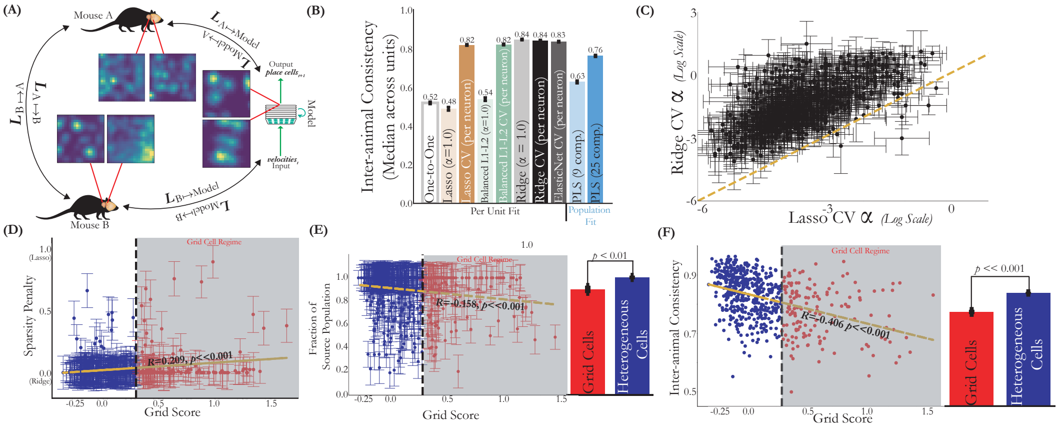

Figure 1: **Heterogeneous cells are just as reliable as grid cells across 2D environments and task conditions.** **(A)** We establish inter-animal consistency levels by mapping animals to each, using inter-animal transforms L of at-most-linear functional complexity. Computational models are then mapped to real animal's MEC data using the same transform class. **(B)** Inter-animal consistency levels as assessed with mapping transform classes of different complexity. Median and s.e.m. across 620 cells. **(C)** The alpha value per neuron as chosen by Ridge CV plotted against the alpha value chosen by Lasso CV, on a log scale. Median and s.e.m. across ten train-test splits. The unity line is in yellow. **(D)** Per unit sparsity penalty under ElasticNet CV (grey bar in (B)) as a function of grid score. Median and s.e.m. across ten train-test splits. **(E)** The fraction of source units assigned nonzero weight per target cell plotted against its grid score (median and s.e.m. across ten train-test splits). Red denotes grid cells with grid score > 0.3 and blue denotes heterogeneous cells with grid score ≤ 0.3. The bar plot on the right is the mean and s.e.m. of this median quantity for the identified grid and heterogeneous cell populations. The p-value is obtained from an independent t-test across neurons between populations. **(F)** Inter-animal consistency of each unit plotted against its grid score. The bar plot on the right is the mean and s.e.m. of this quantity for the identified grid and heterogeneous cell populations. The p-value is obtained from an independent t-test across neurons between populations.

as well as population-level Partial Least Squares (PLS) regression. For all methods with fittable parameters, mapping fit is performed on a random 20% of spatial position bins, and evaluated on the remaining 80% of position bins (see supplement for more details).

The strict one-to-one mapping yielded low inter-animal consistency among the maps considered, capturing around 50% of the target neural response variability. Linear regression with strong sparseness priors, such as Lasso (L1 penalty) and balanced Lasso-Ridge (equal L1 and L2 penalty) regression, also proved to be too strict when evaluated with fixed regularization level $\alpha = 1$, yielding hardly any improvement over the one-to-one mapping. However, pure Ridge (L2 penalty) regression at this regularization level was highly effective, recovering nearly all target variability for most neurons. Cross-validating the L1 regularization constant on a per-cell basis improved the fits, at the cost of requiring substantially looser regularization than for L2 (Fig. 1c). Under an ElasticNet mapping in which both sparsity penalty and regularization constants were chosen with cross validation, most cells were generally still best explained by an essentially Ridge-like transform with no sparsity penalty (Fig. 1d). However, different target cells required different numbers of source units to achieve effective mapping (Fig. 1e). As expected, cells with more stereotypically grid-like response patterns on average chose a higher sparsity penalty (slight positive slope in Fig. 1d) and required fewer source cells to capture (slight negative slope in Fig. 1e) as compared to heterogeneous cells, though this effect is weak. Critically, heterogeneous cells did not have lower inter-animal consistency than grid cells (Fig. 1f).

These results show that the heterogeneous non-stereotyped cell populations are reasonably similar across animals – at least up to (mostly Ridge-regularized) linear transform – establishing that there is a reliable target pattern to study in the first place.

3 Task-optimized models of MEC spatial response variation

Evaluating a spectrum of candidate models. We evaluated models of several basic types. First, we considered learnable neural networks, all of which accept a stream of two-dimensional velocity input vectors, have a single layer of hidden neurons identified as the putative MEC population, and

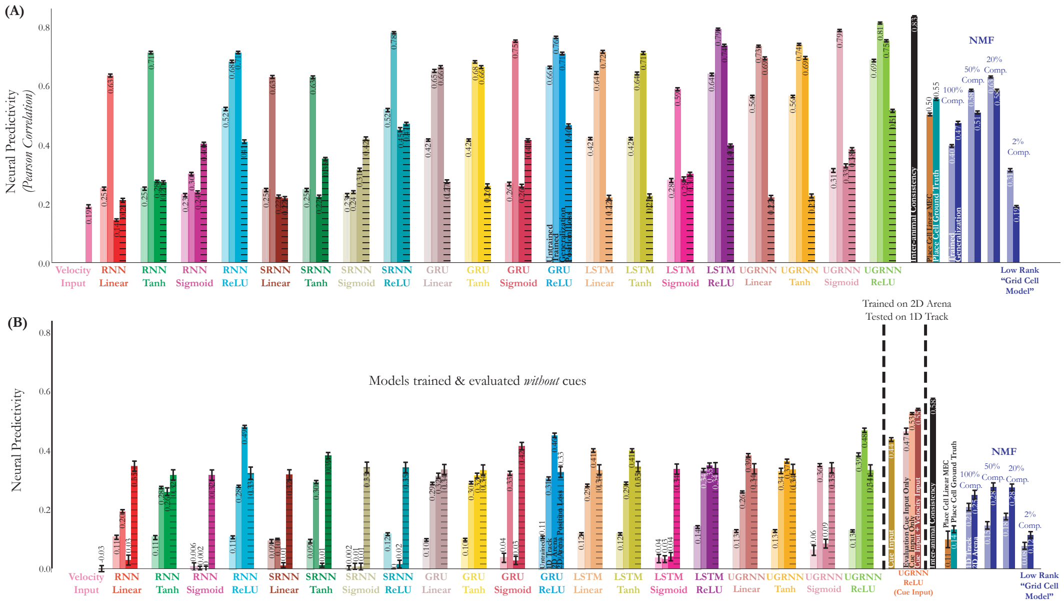

Figure 2: **Task-optimized navigational models best predict the entire MEC population.** **(A)** Neural predictivity of the model MEC units to the real MEC responses of 12 animals in a $100cm^2$ 2D open field under the ElasticNet CV regression transform (grey bar in Fig. 1b). Median and s.e.m. across 620 total units. **(B)** Same as (A), but now the models are evaluated against responses of MEC units while the animal is traversing a $400cm$ 1D track. "1D Track" refers to models trained to path integrate on this same 1D track. "2D Arena" refers to models trained to path integrate on the $2.2m^2$ arena and evaluated on the 1D track. Median and s.e.m. across 2861 units from 8 different animals.

which are optimized to perform some form of path integration readout on simulated motion paths (Erdem and Hasselmo [2012]) in a fixed-sized arena. The networks varied according to the nature of their local recurrent cell structure (ungated RNN, UGRNN, GRU, or LSTM), activation functions (Linear, Tanh, Sigmoid, or ReLU), and output objective function (explicitly constructing a set of simulated place field neurons [Banino et al., 2018] or directly performing two-dimensional position integration [Cueva and Wei, 2018]). Following Banino et al. [2018], all networks consisted of three nonlinear layers, though for the ungated RNN we included its original two layer version (following Cueva and Wei [2018]) and a three layer version ("SRNN"). We ensured that comparisons were fair by equalizing the size of the hidden layer across models, and included an architecture-only control with untrained filter weights.

We also implemented a class of models describing MEC activations as Non-negative Matrix Factorization (NMF) operating on simulated place cells. The lowest-rank version acts a "Grid Cell" model, inspired by recent work positing MEC as a low-dimensional embedding of hippocampal place fields [Stachenfeld et al., 2014, Dordek et al., 2016], and further validated by the mathematical theory developed in Sorscher et al. [2019, 2020] to explain the emergence of grid cells. Higher-rank NMF models titrate between the low-rank Grid Cell model and a full-rank control that measures how well MEC cells can be explained as a linear projection of their putative place cell outputs.

For evaluation, we map each model's proposed MEC activations to empirically measured firing patterns of the real neurons, using the same type of cross-validated ElasticNet regression transform used to measure inter-animal similarity as in Section 2. The predicted neural responses under the mapping are then compared to real target neural responses on a neuron-by-neuron basis, and the median of accuracy of these predictions taken over target neurons. In addition to evaluating model-data match when constructed on arenas of the same size in which the neural data were collected ($100cm^2$), we also evaluated models trained on larger arenas ($2.2m^2$) but tested on the $100cm^2$ arena.

Neural prediction results. The results of our model evaluation, shown in Fig. 2a, support several inferences:

- Different models are substantially different in their ability to predict neural responses. MEC electrophysiology data collected during 2D open-field foraging is thus a strong model target that effectively separates candidate models from each other.

- Task-optimized models are reliably better than their untrained controls, across all architecture types and objective functions.
- Rectification is substantially better than Linear, Tanh, or Sigmoid activation, especially for promoting generalization to new arena sizes.
- Under the rectification nonlinearity, gated circuit architectures (UGRNN, GRU, LSTM) improve model fits compared to the simple ungated alternatives (RNN and SRNN).
- The explicit place cell construction task leads to substantially better fits than the direct two-dimensional path integration task ("Position Loss"), even though both were reported in the literature to create grid cell-like units. Enforcing a place field representation at the output of the network thus appears to be an important constraint in order to recapitulate responses in MEC.
- The Grid Cell (low-rank NMF) model is a poor fit to the MEC population overall.
- Full-rank NMF and "Place Cell Linear MEC" output-based controls capture approximately half the explainable variance of MEC neurons, significantly more than the simple velocity linear input control, but substantially less than any of the input-driven ReLU networks.
- The best model is the UGRNN ReLU trained with the place cell loss ("UGRNN-ReLU-Place Cell"). This model captures nearly 100% of the explainable variance of the MEC neural population response when trained for place cell construction on the arenas of the same size as that on which neural data was collected.

Generalization to novel experimental conditions. If a model is truly correct, then once trained, it should capture neural responses in any new tested condition. We tested generalization in two ways. First, as shown in the second from the right bars of Fig. 2a for each model architecture class, we performed a 2D arena size generalization test by constructing models on one arena size, testing against neural data on another. We found this generalization test gives essentially the same rank-order comparison as in the same-arena-size test, with one striking exception: the RNN Tanh model performs well within arena size, but fails to generalize. The UGRNN-ReLU-Place Cell model again performs the best, capturing 90% explained variance of the neural responses on the novel arena size.

Second, we also evaluated the same 2D-pretrained models by running them on a 1D track, comparing models to neural data collected from mice in a 1D virtual reality setup, using the same ElasticNet mapping procedure as in the 2D comparisons (see supplement for more experimental details). We found (Fig. 2b) that 2D arena trained model results had some similar rank order as for the original 2D results (0.43 Spearman rank correlation), with the UGRNN-ReLU-Place Cell model trained in the 2D arena (and then evaluated on the 1D track) achieving the best match (82%) when trained for place-field construction. We hypothesized that the larger gap between the best model and the inter-animal noise ceiling, as compared to the 2D case, was due to the fact that during the 1D experimental data collection, mice were also presented with visual cues. MEC neurons are known to respond to visual input, but as none of our evaluated models had a visual front-end, they could not respond accordingly. To test this hypothesis we trained the UGRNN-ReLU-Place Cell model in 2D, but with phantom visual cue locations added as input while performing the path integration task. We then evaluated this model against 1D neural prediction (with the real visual cue locations as input), and found that including cues rescued model performance back to essentially the noise ceiling, as in 2D (Fig. 2b, orange bars).

Assessing grid and border score distribution match. The *grid score* is a metric of how stereotypically grid-like the response pattern of a given unit is (see supplementary material for specific definitions), with "grid cell" typically defined as having a grid score of greater than 0.3. Similarly, the border score is a measure of how responsive a given unit is in the presence of environmental boundaries, with a "border cell" typically defined as having a border score greater than 0.5 [Solstad et al., 2008]. The ground-truth distribution of grid and border scores for cells in an unbiased experimental population (gray bars in each subpanel of Fig. 3a,c) characterize the extent to which real MEC populations are non-stereotypical in their responses. To further assess model accuracy, we also compared the distribution of grid and border scores within each model to that of the real data, using the (negative) Kolmogorov-Smirnov (KS) distance of the empirical and model grid score distributions as a quantitative metric. This metric is both stronger and weaker than the mapping accuracy metric used above — stronger in the sense that, since there is no parameter fitting in the metric, to fit it well a model has to have the correct distribution in its raw feature output; and weaker in that it only assesses cells on one component of their profile (i.e. "grid-ness" or "border-ness"). We observed that the low-rank Grid Cell model has poor fit on this metric, essentially because it contains *too many* grid cells (see Fig. 3a, upper left). In contrast, the same model that achieves best fits on the neural-fit

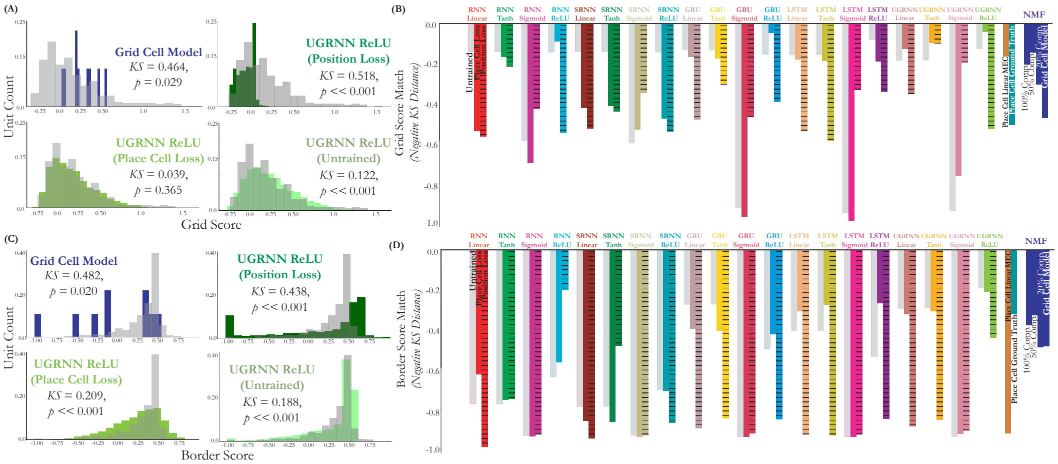

Figure 3: **Relationships to grid and border score distribution.** **(A,C)** Example distributions of grid and border scores from selected models against the ground truth distribution in the neural data (grey). Negative Kolmogorov-Smirnov distance between the distribution of grid scores of the model units to those of the 620 units in the data. The models have all been trained in a different $2.2m^2$ arena, but are evaluated on a $100cm^2$ environment that the animals are in. **(B, D)** Quantification across all models.

metric (UGRNN-ReLU-Place Cell) also achieves the best on the grid cell distribution match metric, and is the only architecture that cannot be distinguished from the ground truth at the $KS < 0.05$ significance level. Across model architectures, KS-distance was generally in line with neural regression fit metrics (see Fig. 3b,d and Fig. S2). One key difference between the metrics, however, is that for most architectures, the untrained filters provided *better* grid score distribution matches as models with task-trained parameters — except for the that best matched architecture, UGRNN ReLU, where the trained place cell construction model is better than its untrained counterpart (and similar to it for the border score) — and models trained for direct path integration were especially poor on this metric. These results suggest the two metrics in Figs. 2 and 3 are complementary which aspects of model correctness they address, and together help zero in on the most effective models overall.

4 Predicting the functional relevance of heterogeneous cells

Correlating task performance and neural predictivity. To begin to address the question of the functional relevance of heterogeneous cells, we first looked at the overall correlation between model task performance and neural fit (Fig. 4a). Though the correlation is imperfect, the most task performant models (e.g. UGRNN- and LSTM-ReLU-Place Cell) achieve the best matches to neural fit suggesting that improved task performance may be causally related to the ability capture neuron response patterns across the population. In contrast, there is a comparatively weaker relationship between grid and border score distribution match and model performance (Fig. 4b). While the most task performant models achieve the best match here, models with a gating architecture and ReLU are strong matches to these metrics with untrained filters, illustrating the importance in matching cell properties other than "grid-ness" or "border-ness" in predicting task performance.

Relative predictivity gain for heterogeneous cells. The above result is put into greater perspective by comparing neural predictivity differential between a task-trained UGRNN-ReLU-Place Cell model and the Grid Cell model, on a per-neuron basis, as a function of grid score (Fig. 4c). The task-trained model has improved neural predictivity relative to the Grid Cell model for grid cells (grid score > 0.3) and heterogeneous cells, but the improvements on the latter (as well as border cells, Fig. S3) are larger than for grid cells. This again suggests that the heterogeneous cells are playing a substantial role in allowing the trained model to achieve improved performance.

Virtual knockout experiment. To address the question most directly, we performed a cell-type-targeted virtual knockout comparison experiment. In doing this, we used the UGRNN-ReLU-Place Cell and LSTM-ReLU-Place Cell models, the two best models emerging from the previous section with essentially similar neural predictivity across multiple metrics. We identified the units in the trained model with high grid score (> 0.3) or border score (> 0.5), and gradually increased the threshold, while measuring task generalization performance (see supplement for details of this

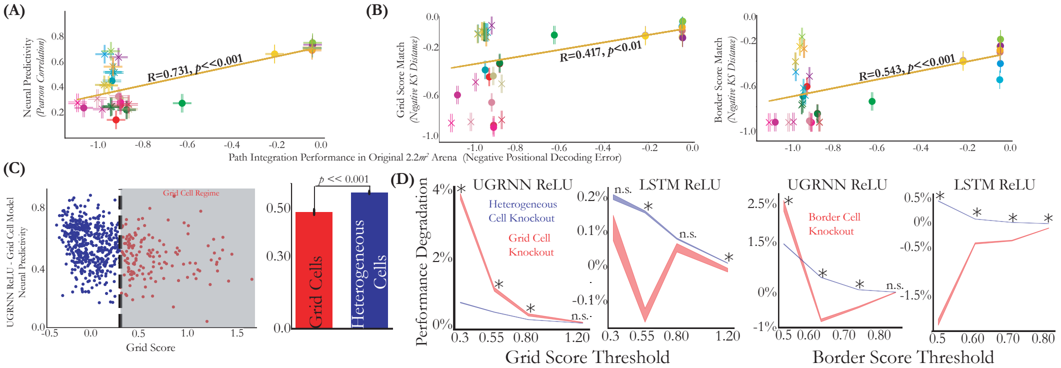

Figure 4: **Heterogeneous cells are relevant to navigation.** **(A)** Neural predictivity (median and s.e.m. across 620 units) versus path integration performance (mean and s.e.m. across 20,000 episodes), measured by negative positional decoding error, of the neural network models. The models are either untrained ("X") or trained with the place cell loss ("O") in the $2.2m^2$ arena. Positional decoding error is measured by taking the top 3 most active place cell outputs in each model, evaluated on the $100cm^2$ arena. **(B)** Same as (A), but with grid score (*Left*) and border score (*Right*) distribution instead of neural predictivity. **(C)** (*Left*) Per unit neural predictivity difference between the UGRNN-ReLU-Place Cell and Grid Cell models trained on the $2.2m^2$ arena and evaluated on the $100cm^2$ arena, plotted against that unit's grid score. (*Right*) Quantification of this difference aggregated across the grid cell and heterogeneous cell populations, respectively (mean and s.e.m). The p-value is obtained from an independent t-test across neurons between populations. **(D)** For the UGRNN-ReLU-Place Cell and LSTM-ReLU-Place Cell models, we measure the normalized performance degradation as evaluated on the $100cm^2$ arena, relative to the full model trained on the $2.2m^2$ arena with the place cell loss. We identify units in this trained model with grid and border scores of varying thresholds and knockout the same number of heterogeneous cells (randomly sampled 100 times), to yield the "heterogeneous cell knockout" (blue). The x-axis denotes the threshold used for the score. Mean and s.e.m. over evaluation episodes. * denotes a p-value < 0.01 obtained from an independent t-test between the performance degradations of the two knockouts at a given grid or border score threshold.

knockout procedure). We similarly ablated matched numbers of heterogeneous units (grid score ≤ 0.3 or border score ≤ 0.5 or both, see Fig. S4 for the latter), again measuring model performance.

The main result of this experiment (Fig. 4d) is that all networks are highly robust to knockouts, experiencing only at most 1-4% performance degradation relative to the full model even when substantial fractions of units are knocked out (corresponding to 21-24% of a single layer's units for grid score > 0.3 and 16-20% of these units for border score > 0.5). At stricter grid and border cell thresholds, the heterogeneous knockout is similarly injurious to the cell-type specific knockouts, across the two model architectures, suggesting that highly-stereotypical "classical" border and grid score cells are not more essential to the path integration behavior than heterogeneous cells. At low thresholds (when counting many relatively heterogeneous cells as "grid" or "border" cells), the two model architectures give divergent predictions, with the UGRNN model showing a small but significant effect of grid cells relative heterogeneous cells, and the LSTM model showing the opposite. It would be of substantial future interest to confirm or reject either of these models' predictions with a real targeted knockout experiment *in vivo*.

5 Modeling reward-driven modulation in MEC

Recent work in both rodents and humans has uncovered that MEC and HPC neurons represent not only literal space, but also capture spatialized layouts in a more abstract sense in modalities other than spatial position [Constantinescu et al., 2016, Aronov et al., 2017]. It has also been seen [Butler et al., 2019, Boccara et al., 2019] that non-spatial rewards can influence the shape of MEC response profiles, restructuring them to incorporate the location of the learned reward. These intriguing phenomena represent a natural direction for modeling, but are not captured by any of the neural network models described in Section 3, as they do not have reward-modulated inputs. (Note that while Banino et al. [2018] used a pretrained LSTM-Tanh-Place Cell network as a front-end on which reinforcement-based navigation tasks are evaluated downstream, reward state is not input to their network or otherwise propagated back into the MEC-like layers of their model, and thus cannot address the neural modeling question raised here.)

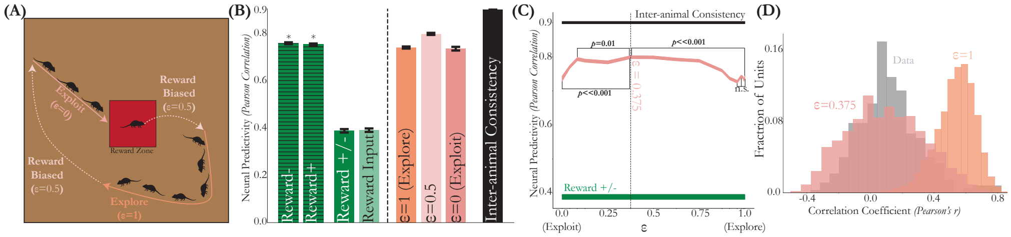

Figure 5: **Reward biased path integration captures remapping of responses in the presence of a reward.** **(A)** Schematic of the three conditions. "Explore" is the original set of random bout trajectories used to train the agent as before. "Exploit" involves the agent navigating directly to the reward zone (red box) within a fixed number of timesteps per episode (7), and spending the remaining 13 timesteps only path integrating in the reward zone. "Reward Biased" refers to training with each of the above two trajectories on $0 < \varepsilon < 1$ of the training episodes. **(B)** Neural predictivity of the UGRNN-ReLU-Place Cell model, evaluated on the $150cm^2$ arena. Median and s.e.m. of 598 cells from 7 rats. Green bars denote the model evaluated with random walk velocity inputs. $*$ denotes comparison of the model to each condition *separately*. Coral bars (to the right of the vertical line) denote the model evaluated with random walk and reward biased velocity input modulation. **(C)** Neural predictivity of the reward biased UGRNN-ReLU-Place Cell model as a function of ε. Median and s.e.m. of 598 cells from 7 rats. The p-value is obtained from an independent t-test across neurons between selected model pairs in brackets. **(D)** Histogram of the correlation coefficient of unit rate maps between the random foraging and velocity-input modulated reward condition in the $\varepsilon = 0.375$ and $\varepsilon = 1$ UGRNN-ReLU-Place Cell models. The same metric applied to the data is shown in grey.

We thus sought to build new network models that respond to the existence of an extrinsic reward in a behaviorally-meaningful fashion. One natural approach would be to build a full reinforcement-learning (RL) agent in which reward depends on environmental features (e.g.total number of food units encountered during a forage path). A model optimized end-to-end to forage under these conditions could be successful in creating a representation that matches MEC reward-response modulation. However, we took a more direct approach: simply change the behavior of the agent in response to reward in the way RL would be expected to do if successful, by modifying the foraging trajectories used during training and testing to no longer be pure random walks.

Specifically, we trained the UGRNN-ReLU-Place Cell model with a variety of foraging policies titrating between pure exploration and exploitation. Paths could either purely exploit a known reward resource by directly moving to the reward location when it is present (Fig. 5a, $\varepsilon = 0$ condition), purely explore with a random walk as in the original unmodulated model ($\varepsilon = 1$), or take an intermediate policy ($0 < \varepsilon < 1$). In creating these scenarios, we sought to roughly mimic experimental observations showing animals often take rapid and direct paths to the reward zone [Butler et al., 2019] (see supplement for details of implementation). Throughout, the actual training task of the network remained the same as above — place cell construction — but the network now would have to be tolerant to reward-modulated input velocity changes while maintaining positional knowledge.

We then compared these networks to neural data from [Butler et al., 2019] for animals in conditions both with reward (**reward+**) and without it (**reward-**), in which the animal navigates to a $20cm^2$ reward zone within a $150cm^2$ arena to receive 0.5-1 units of cereal. We first performed inter-animal consistency checks, using the same method as we did with the purely spatial-condition data in Section 2 above, finding that inter-animal consistency for neural responses is high both within reward condition (e.g. just spatial modulation in **reward+** and **reward-** separately), and across both spatial and reward-modulated conditions overall (Fig. S6).

As a baseline, we then evaluated the ability of the original (non-reward-modulated) UGRNN-ReLU-Place Cell model to match neural data. Consistent with results reported in the previous sections, this model achieved high neural predictivity in **reward+** and **reward-** separately (Fig. 5b, hatched bars). However, as expected, this non-modulated model was ineffective at predicting response patterns across both reward conditions (Fig. 5b, **reward+/-**). Moreover, simply augmenting the network input to receive an additional binary reward state (Fig. 5b,"Reward Input"), but without using that input to specifically modulate input velocities or output behavior, did not substantially improve neural predictivity, showing the need for nontrivial integration of reward-modulated state.

We then evaluated the reward-modulated networks, generating network outputs for comparison with the **reward+** and **reward-** conditions by modulating velocity inputs to match each condition during testing (see supplement for details). We found that the pure-explore ($\varepsilon = 1$) model is substantially better at matching neural responses across reward conditions than the **reward+/-** baseline. This is perhaps somewhat surprising since in this comparison the underlying neural network model is identical, just evaluated with or without reward-modulated input data during testing. This result suggests that a substantial fraction of the reported reward-modulated effect in MEC may actually simply be input-driven, lending a new interpretation to results of [Butler et al., 2019]. (It may be useful to note that this result represents a model-based control that was inaccessible to the authors of [Butler et al., 2019].) However, we did find that exposing the network to a mixture of exploration and exploitation behaviors during training does lead to networks with somewhat improved neural predictivity. Results were largely robust to the specific proportion of exploration-vs-exploitation (Fig. 5c), though the best model (at $\varepsilon = 0.375$) model was statistically-significantly better than alternatives and had a substantially closer match ($KS = 0.13$ vs. $KS = 0.81$) to the data's unit-level remapping across conditions than the original $\varepsilon = 1$ model (Fig. 5d). These results are consistent with there being some nontrivial within-MEC reward-modulated responses beyond simple input modulation alone.

6 Discussion

We have identified a goal-driven neural network model of MEC that is quantitatively accurate across a wide variety of common experimental conditions, and that can be used to generate nontrivial insights about the underlying mechanisms and functional roles of mouse MEC. More generally, our results suggest that constraint-driven neural networks may provide a fruitful approach for studying navigation and memory in the MEC, and beyond.

Our work suggests the existence not of a specialized class of heterogeneous cells that is functionally segregated from classic cell types, but rather a continuum of cells within a single unified network that naturally encompasses grid, border, and heterogeneous cells. Future research on the computational foundations of MEC/HPC may thus be well-served by putting less emphasis on identifying cleanly-stereotypical cell types (such as grid cells) or perfecting mathematically simple models of single such cell-types, and looking instead for holistic computational ideas that move beyond "easy to visualize" but perhaps overly-simplistic tuning-curve categories.

The nature of the understanding afforded by such a modeling approach comes from their ability to make inferences about what constraints (both structural and functional) are consistent with the data. The current work rules out a variety of simple network connectivity diagrams as inconsistent with MEC cell data, and narrows the space of functional goals MEC circuit weights might be optimized for over evolutionary timescales. Improving the results here will hopefully narrow these constraints yet further, with the ultimate goal of identifying constraints that yield the uniquely correct MEC circuit diagram and synaptic weights – or at least, the narrowest set of such networks consistent with the inherent variability between animals in the real population.

There are some key limitations, however, on our results. First, we attempted to identify the simplest underlying transform that would map animals within a population to each other, using this as the basis for conclusions both about unit-type reliability and model-data comparisons. While the philosophy of this approach may be sound [Cao and Yamins, 2021a], in practice it is possible that we could narrow the transform class further by (e.g.) enforcing that it be invariant to one more more properties of classical cell-types (e.g. periodicity). Finding algorithms to better identify sharp inter-animal transform classes will be an important topic for future work.

Moreover, we remain unconvinced that MEC just "is" a UGRNN-ReLU-Place Cell network, despite a network of this architecture having explained essentially all the data we had available to challenge it. It is possible that matching all our existing data is too easy a test. Would this network generalize to more complex situations, with increased variability along key axes such as spatial structure (e.g. environments with corridors and looping interconnections), nontrivial but spatially informative cues, and rewards exhibiting complex and temporally-variable patterns? We do not think it is at all obvious that it would. The proper conclusion from this work is thus not that our current best model is actually correct, but rather that the fairly simplified setup typical of experiments in MEC is not sufficiently complex to falsify it. This present situation is a concrete manifestation of the "contravariance principle" of neural modeling [Cao and Yamins, 2021b] – the idea that working in a *more* complex experimental environment might actually make it *easier* to identify an actually correct model, by virtue of reducing susceptibility to spurious models that are apparently consistent with too-simple data. Future experiments should engage mice in more complex environments and behaviors.

7 Acknowledgements

We are grateful to Alex Gonzalez, David Sussillo, and John H. Wen for helpful discussions. We thank the anonymous reviewers for their feedback on a draft of this manuscript. A.A. received support from the Swiss National Science Foundation (P2BSP3_181743 and P400PB_191076). I.L. is supported by funding from the Wu Tsai Neurosciences Institute under Stanford Interdisciplinary Graduate Fellowships and a Bertarelli fellowship. A.H.W. received funding support from the National Institutes of Health BRAIN initiative (1F32MH122998-01), and the Wu Tsai Stanford Neurosciences Institute Interdisciplinary Scholar Program. S.G. is supported by the James S. McDonnell Foundation, Simons Foundation, and National Science Foundation CAREER Award for funding while at Stanford. L.M.G. is supported by the Office of Naval Research N00141812690, Simons Foundation SCGB 542987SPI, the James S. McDonnell Foundation, and the Vallee Foundation. D.L.K.Y. is supported by the James S. McDonnell Foundation (Understanding Human Cognition Award Grant No. 220020469), the Simons Foundation (Collaboration on the Global Brain Grant No. 543061), the Sloan Foundation (Fellowship FG-2018-10963), the National Science Foundation (RI 1703161 and CAREER Award 1844724), the DARPA Machine Common Sense program, and hardware donation from the NVIDIA Corporation.

References

M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.

D. Aronov, R. Nevers, and D. W. Tank. Mapping of a non-spatial dimension by the hippocampal–entorhinal circuit. *Nature*, 543(7647):719–722, 2017.

A. Banino, C. Barry, B. Uria, C. Blundell, T. Lillicrap, P. Mirowski, A. Pritzel, M. J. Chadwick, T. Degris, J. Modayil, et al. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557 (7705):429–433, 2018.

C. N. Boccara, M. Nardin, F. Stella, J. O'Neill, and J. Csicsvari. The entorhinal cognitive map is attracted to goals. *Science*, 363(6434):1443–1447, 2019.

Y. Burak and I. R. Fiete. Accurate path integration in continuous attractor network models of grid cells. *PLoS Comput Biol*, 5(2):e1000291, 2009.

W. N. Butler, K. Hardcastle, and L. M. Giocomo. Remembered reward locations restructure entorhinal spatial maps. *Science*, 363(6434):1447–1452, 2019.

M. G. Campbell, S. A. Ocko, C. S. Mallory, I. I. Low, S. Ganguli, and L. M. Giocomo. Principles governing the integration of landmark and self-motion cues in entorhinal cortical codes for navigation. *Nature neuroscience*, 21(8):1096–1106, 2018.

R. Cao and D. Yamins. Explanatory models in neuroscience: Part 1–taking mechanistic abstraction seriously. *arXiv preprint arXiv:2104.01490*, 2021a.

R. Cao and D. Yamins. Explanatory models in neuroscience: Part 2–constraint-based intelligibility. *arXiv preprint arXiv:2104.01489*, 2021b.

K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4012. URL `https://www.aclweb.org/anthology/W14-4012`.

J. Collins, J. Sohl-Dickstein, and D. Sussillo. Capacity and trainability in recurrent neural networks. In *ICLR*, 2017.

A. O. Constantinescu, J. X. O'Reilly, and T. E. Behrens. Organizing conceptual knowledge in humans with a gridlike code. *Science*, 352(6292):1464–1468, 2016.

J. J. Couey, A. Witoelar, S.-J. Zhang, K. Zheng, J. Ye, B. Dunn, R. Czajkowski, M.-B. Moser, E. I. Moser, Y. Roudi, et al. Recurrent inhibitory circuitry as a mechanism for grid formation. *Nature neuroscience*, 16(3): 318–324, 2013.

C. J. Cueva and X.-X. Wei. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. *ICLR 2018, arXiv:1803.07770*, 2018.

Y. Dordek, D. Soudry, R. Meir, and D. Derdikman. Extracting grid cell characteristics from place cell inputs using non-negative principal component analysis. *Elife*, 5:e10094, 2016.

J. L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

U. M. Erdem and M. Hasselmo. A goal-directed spatial navigation model using forward trajectory planning based on grid cells. *European Journal of Neuroscience*, 35(6):916–931, 2012.

S. Farhoodi, M. Plitt, L. Giocomo, and U. Eden. Estimating fluctuations in neural representations of uncertain environments. *Advances in Neural Information Processing Systems*, 33, 2020.

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

T. Hafting, M. Fyhn, S. Molden, M.-B. Moser, and E. I. Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806, 2005.

K. Hardcastle, N. Maheswaranathan, S. Ganguli, and L. M. Giocomo. A multiplexed, heterogeneous, and adaptive code for navigation in medial entorhinal cortex. *Neuron*, 94(2):375–387, 2017.

J. R. Hinman, M. P. Brandon, J. R. Climer, G. W. Chapman, and M. E. Hasselmo. Multiple running speed signals in medial entorhinal cortex. *Neuron*, 91(3):666–679, 2016.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

A. J. Kell, D. L. Yamins, E. N. Shook, S. V. Norman-Haignere, and J. H. McDermott. A task-optimized neural network replicates human auditory behavior, predicts brain responses, and reveals a cortical processing hierarchy. *Neuron*, 98(3):630–644, 2018.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

E. Kropff, J. E. Carmichael, M.-B. Moser, and E. I. Moser. Speed cells in the medial entorhinal cortex. *Nature*, 523(7561):419–424, 2015.

R. F. Langston, J. A. Ainge, J. J. Couey, C. B. Canto, T. L. Bjerknes, M. P. Witter, E. I. Moser, and M.-B. Moser. Development of the spatial representation system in the rat. *Science*, 328(5985):1576–1580, 2010.

I. I. Low, A. H. Williams, M. G. Campbell, S. W. Linderman, and L. M. Giocomo. Dynamic and reversible remapping of network representations in an unchanging environment. *bioRxiv*, 2020.

C. S. Mallory, K. Hardcastle, M. G. Campbell, A. Attinger, I. I. Low, J. L. Raymond, and L. M. Giocomo. Mouse entorhinal cortex encodes a diverse repertoire of self-motion signals. *Nature communications*, 12(1):1–20, 2021.

J. A. Michaels, S. Schaffelhofer, A. Agudelo-Toro, and H. Scherberger. A goal-driven modular neural network predicts parietofrontal neural dynamics during grasping. *Proceedings of the National Academy of Sciences*, 117(50):32124–32135, 2020.

S. A. Ocko, K. Hardcastle, L. M. Giocomo, and S. Ganguli. Emergent elasticity in the neural code for space. *Proceedings of the National Academy of Sciences*, 115(50):E11798–E11806, 2018.

J. O'Keefe and J. Dostrovsky. The hippocampus as a spatial map: Preliminary evidence from unit activity in the freely-moving rat. *Brain research*, 1971.

F. Sargolini, M. Fyhn, T. Hafting, B. L. McNaughton, M. P. Witter, M.-B. Moser, and E. I. Moser. Conjunctive representation of position, direction, and velocity in entorhinal cortex. *Science*, 312(5774):758–762, 2006.

W. E. Skaggs, B. L. McNaughton, K. M. Gothard, and E. J. Markus. An information-theoretic approach to deciphering the hippocampal code. In *Proceedings of the 5th International Conference on Neural Information Processing Systems*, pages 1030–1037, 1992.

T. Solstad, C. N. Boccara, E. Kropff, M.-B. Moser, and E. I. Moser. Representation of geometric borders in the entorhinal cortex. *Science*, 322(5909):1865–1868, 2008.

B. Sorscher, G. Mel, S. Ganguli, and S. A. Ocko. A unified theory for the origin of grid cells through the lens of pattern formation. *Advances in Neural Information Processing Systems*, 32, 2019.

B. Sorscher, G. C. Mel, S. A. Ocko, L. Giocomo, and S. Ganguli. A unified theory for the computational and mechanistic origins of grid cells. *bioRxiv*, 2020.

K. L. Stachenfeld, M. Botvinick, and S. J. Gershman. Design principles of the hippocampal cognitive map. *Advances in neural information processing systems*, 27:2528–2536, 2014.

D. L. Yamins and J. J. DiCarlo. Using goal-driven deep learning models to understand sensory cortex. *Nature neuroscience*, 19(3):356–365, 2016.

K. Yoon, M. A. Buice, C. Barry, R. Hayman, N. Burgess, and I. R. Fiete. Specific evidence of low-dimensional continuous attractor dynamics in grid cells. *Nature neuroscience*, 16(8):1077–1084, 2013.

K. Zhang. Representation of spatial orientation by the intrinsic dynamics of the head-direction cell ensemble: a theory. *Journal of Neuroscience*, 16(6):2112–2126, 1996.

A Comparisons to Neural Data

A.1 Experimental Data

We analyze three neural datasets in total (two from tetrode recordings of freely moving animals in 2D arenas and one from Neuropixels recordings of head-fixed mice on a 1D virtual track). The 2D open-field foraging dataset in the $100cm^2$ arena came from Mallory et al. [2021] across 12 mice, totalling 620 MEC cells. The 2D open-field foraging (**reward-** condition) and reward dataset (**reward+** condition) in the $150cm^2$ arena came from Butler et al. [2019] across 7 rats, totalling 598 MEC cells. Note that the *same* neural population was used across these two conditions, whereby in the **reward+** condition, the animal navigates to a $20cm^2$ reward zone within this arena to receive 0.5-1 units of cereal. The reward zone location was fixed across sessions for each animal, but varied between animals. Finally, the 1D VR data was used by Mallory et al. [2021] across 8 mice, totalling 2861 MEC cells. There were five landmarks (towers) total at 0, 80, 160, 240, and $320cm$. For further experimental details, please refer to the respective cited paper referred above.

In the 1D VR data, we identified remapping events, depicted in Fig. S8, following the procedure used by Low et al. [2020], by examining the trial-by-trial similarity matrices as well as the test-set (90-10% splits) R^2 of k-means clustering (red) applied to the responses and identifying where it diverges from PCA (blue).

A.2 Rate Map Representation

We bin the positions in each environment using 5 cm bins, following prior work [Hardcastle et al., 2017, Butler et al., 2019, Low et al., 2020]. Thus, the $100cm^2$ environment used 400 (20×20) bins, the $150cm^2$ environment used 900 (30×30) bins, and the $400cm$ 1D track used 80 bins. We then calculated each animal's binned occupancy in seconds as well as how many times each MEC cell spiked in each bin. Binned firing rates for each MEC cell were calculated as the ratio of the number of spikes and the time spent in that bin in seconds. Finally, a Gaussian filter ($\sigma = 1$ bin) with dimensionality matching the environment (1D or 2D) was used to smooth the rate maps.

Since the model units do not have spikes but rates, the procedure is analogous for the agent but without Gaussian smoothing. Specifically, to generate the model rate map (from layer g for the path integrator networks, defined in (15)), we evaluate the model on 100 batches consisting of 200 evaluation trajectories (see Section C.1 for more details), with each episode being seeded to ensure the same evaluation trajectories are used across all models. Every timestep corresponded to $20ms$ increments of time.

See Fig. S5 for visualizations of the rate maps of both the neural data and the model.

A.3 Mapping Transforms

When we perform neural fits, we choose a random 20% set of these position bins to train and cross-validate the regression, and the remaining 80% to use as a test set, across ten train-test splits total. For ElasticNet-based regression (including Lasso, Balanced Lasso-Ridge, and Ridge regression as special cases), we use the `sklearn.linear_model` class. When we perform cross-validation, which we do for the $100cm^2$ arena in 2D (Fig. 1b), the $400cm$ 1D track (Fig. 2b), and the $150cm^2$ for the 2D open foraging, reward, and combined across conditions datasets (Fig. 5b), we search over $\alpha \in [10^{-9}, 10^9]$ logspaced uniformly and L1 ratio spaced uniformly $\in [0, 1]$ with two-fold cross-validation on 20% of the position bins per train-test split and neuron individually. See Fig. S1 for the inter-animal consistency (2D data) across α and L1 ratio. For the 1D data, we choose the parameters on the validation set that perform best averaged across the maps for a neuron. The implementations of all of the transforms can be found here: `https://github.com/neuroailab/mec`.

A.4 Grid Score and Border Score

We calculated each cell (and model unit) grid score by taking a circular sample of the spatial rate map autocorrelation centered on the central peak and compared it to rotated versions of the same circular sample (60° and 120° versus 30°, 90°, and 150°). The grid score [Langston et al., 2010, Butler et al., 2019] was defined as the mean correlation at 60° and 120° minus the mean correlation at 30°, 90°, and 150°.

The border score was computed following Solstad et al. [2008], by calculating $\frac{CM - DM}{CM + DM}$, where CM is the proportion of high firing rate bins along one wall, and DM is the normalized mean product of the firing rate of each bin and its distance to the nearest wall.

The grid and border scores in Fig. 3 were always computed for each model at the layer of maximum neural predictivity (median across neurons). See Fig. S5 for visualizations of grid, border, and heterogeneous cells in both the neural data and the UGRNN-ReLU-Place Cell model.

B Neural Fitting Procedure and Inter-animal Consistency Definition

Suppose we have neural responses from two animals A and B. Let t_i^p be the "true" rate map of animal $p \in \mathcal{A} = \{A, B, ...\}$ on stimulus set $i \in \{\text{train}, \text{test}\}$ given by positions in the rate map. Of course, we only receive noisy observations of t_i^p, so let $s_{j,i}^p$ be the j-th set of n trials of t_i^p. Finally, let $M(x; y)_i$ be the predictions of a mapping M (e.g. Ridge regression) when trained on input x to match output y and tested on stimulus set i. For example, $M\left(\mathrm{t}^{\mathrm{A}}_{\text{train}}; \mathrm{t}^{\mathrm{B}}_{\text{train}}\right)_{\text{test}}$ is the prediction of the mapping M on the test stimulus trained to match the true neural responses from animal B given input from the true neural responses from animal A on the train stimulus, and correspondingly, $M\left(\mathrm{s}^{\mathrm{A}}_{1,\text{train}}; \mathrm{s}^{\mathrm{B}}_{1,\text{train}}\right)_{\text{test}}$ is the prediction of the mapping M on the test stimulus trained to match the (trial-average) of noisy sample 1 on the train stimulus from animal B given inputs from the (trial-average) of noisy sample 1 on the train stimulus from animal A. Finally, r is the rate map constructed from the model units, which by construction are deterministic across trials.

With these definitions in hand, we now define the inter-animal consistency and the model neural predictivity. Namely, when we have repeated trials in the same environment (as in the case of 1D data), we compute the following quantity for all units in target animal B:

$$\text{Inter-animal Consistency}^{B}_{\text{1D}} := \left\langle \frac{\mathrm{Corr}\left(M\left(\mathrm{s}^{\mathrm{A}}_{1,\text{train}}; \mathrm{s}^{\mathrm{B}}_{1,\text{train}}\right)_{\text{test}}, \mathrm{s}^{\mathrm{B}}_{2,\text{test}}\right)}{\sqrt{\widetilde{\mathrm{Corr}}\left(M\left(\mathrm{s}^{\mathrm{A}}_{1,\text{train}}; \mathrm{s}^{\mathrm{B}}_{1,\text{train}}\right)_{\text{test}}, M\left(\mathrm{s}^{\mathrm{A}}_{2,\text{train}}; \mathrm{s}^{\mathrm{B}}_{2,\text{train}}\right)_{\text{test}}\right) \times \widetilde{\mathrm{Corr}}\left(\mathrm{s}^{\mathrm{B}}_{1,\text{test}}, \mathrm{s}^{\mathrm{B}}_{2,\text{test}}\right)}} \right\rangle_{A \in \mathcal{A} : (A,B) \in \mathcal{A} \times \mathcal{A}}, \qquad (1)$$

$$\text{Model Neural Predictivity}^{B}_{\text{1D}} := \left\langle \frac{\mathrm{Corr}\left(M\left(r_{\text{train}}; \mathrm{s}^{\mathrm{B}}_{1,\text{train}}\right)_{\text{test}}, \mathrm{s}^{\mathrm{B}}_{2,\text{test}}\right)}{\sqrt{\widetilde{\mathrm{Corr}}\left(M\left(r_{\text{train}}; \mathrm{s}^{\mathrm{B}}_{1,\text{train}}\right)_{\text{test}}, M\left(r_{\text{train}}; \mathrm{s}^{\mathrm{B}}_{2,\text{train}}\right)_{\text{test}}\right) \times \widetilde{\mathrm{Corr}}\left(\mathrm{s}^{\mathrm{B}}_{1,\text{test}}, \mathrm{s}^{\mathrm{B}}_{2,\text{test}}\right)}} \right\rangle_{A \in \mathcal{A} : (A,B) \in \mathcal{A} \times \mathcal{A}}, \qquad (2)$$

where the outermost average is across all source animals A that regress to the current target animal B, ten train-test splits, and 100 bootstrapped trials, Corr is Pearson correlation across test stimuli (in this case, held-out position bins), and $\widetilde{\mathrm{Corr}}$ is Pearson correlation with Spearman-Brown correction applied to it, namely

$$\widetilde{\mathrm{Corr}}\,(X, Y) := \frac{2\,\mathrm{Corr}\,(X, Y)}{1 + \mathrm{Corr}\,(X, Y)}.$$

In 1D, we can also have *multiple* maps within subsets of trials [Low et al., 2020], which we identify in Fig. S8. To account for this, we treat each map, B^{m_i}, which corresponds to responses of the same population in the target animal B to a subset of trials, as its own target of explanation. The average over source animals $A \in \mathcal{A} : A \neq B$ in (1) now is an average over source animals and their respective maps, A^{m_j}. Note that in the standard limit of one map per animal, this is exactly the same as the original quantity in (1).

In the absence of repeated trials in the same environment (which was the case for the 2D data) and therefore also a single map per environment per animal, the terms in the denominator of (1) are trivially 1, giving us the following quantity for all units in target animal B:

$$\text{Inter-animal Consistency}^{B}_{\text{2D}} := \left\langle \mathrm{Corr}\left(M\left(s^{\hat{A}}_{\text{train}}; s^{B}_{\text{train}}\right)_{\text{test}}, s^{B}_{\text{test}}\right)\right\rangle, \qquad (3)$$

$$\text{Model Neural Predictivity}^{B}_{\text{2D}} := \left\langle \mathrm{Corr}\left(M\left(r_{\text{train}}; s^{B}_{\text{train}}\right)_{\text{test}}, s^{B}_{\text{test}}\right)\right\rangle, \qquad (4)$$

where now this outermost average is just across the ten train-test splits, and $\hat{A}$ is the sole "pooled" source animal constructed from the pseudo-population of the remaining animals distinct from the target animal B, namely, $A \in \mathcal{A} : A \neq B$. As expanded on in Section B.4, the latter use of the pooled source animal $\hat{A}$ is to ensure that we have a relatively comparable number of units in the source animal from the 2D tetrode data as in the 1D (Neuropixels) data, since otherwise there would be a small number of units in any single tetrode session.

The inter-animal consistency and model neural predictivity then is the concatenation (denoted by $\oplus$) of all of the inter-animal consistencies of all units in each animal $B \in \mathcal{A}$, over which we take median and s.e.m.:

$$\begin{aligned}
\text{Inter-animal Consistency}_{\text{1D}} &:= \bigoplus_{B \in \mathcal{A}} \left\langle \text{Inter-animal Consistency}^{B_{m_i}}_{\text{1D}} \right\rangle_{\text{Maps } m_i \,\in\, B}, \\
\text{Model Neural Predicitivity}_{\text{1D}} &:= \bigoplus_{B \in \mathcal{A}} \left\langle \text{Model Neural Predictivity}^{B_{m_i}}_{\text{1D}} \right\rangle_{\text{Maps } m_i \,\in\, B},
\end{aligned} \qquad (5)$$

$$\begin{aligned}\text{Inter-animal Consistency}_{\text{2D}} &:= \bigoplus_{B\in\mathcal{A}} \text{Inter-animal Consistency}^{B}_{\text{2D}},\\ \text{Model Neural Predictivity}_{\text{2D}} &:= \bigoplus_{B\in\mathcal{A}} \text{Model Neural Predictivity}^{B}_{\text{2D}},\end{aligned} \tag{6}$$

Thus, Figures 1b, 2a, and 5b are the median and s.e.m. of the quantities in (6). Fig. 2b is the quantity in (5), which we pass through $\tanh$ to be $\in [-1, 1]$, the same scale for visual comparison as (6), which we then compute median and s.e.m. over. Note that we compute these quantities in the 2D and 1D data at *each* model layer, and then report the performance at the layer of maximum median neural predictivity for each model.

In the following subsections, we give background on how the full quantity (1) can be obtained from the base "true" quantity we want to estimate (7). Note that these are not formal proofs (as they rely on assumptions which do not necessarily hold in all cases), but are meant to outline the motivations behind the final quantity.

B.1 Single Animal Pair Motivation

The inter-animal consistency from one animal A to another animal B corresponds to the following "true" quantity to be estimated:

$$\text{Corr}\left(M\left(\text{t}^{\text{A}}_{\text{train}};\text{t}^{\text{B}}_{\text{train}}\right)_{\text{test}},\text{t}^{\text{B}}_{\text{test}}\right), \tag{7}$$

where Corr is the Pearson correlation across test stimuli. In what follows, we argue that this true quantity can be approximated with the following ratio of measurable quantities where we divide the noisy trial observations into two sets of equal samples:

$$\begin{aligned}&\text{Corr}\left(M\left(\text{t}^{\text{A}}_{\text{train}};\text{t}^{\text{B}}_{\text{train}}\right)_{\text{test}},\text{t}^{\text{B}}_{\text{test}}\right)\\ &\sim\frac{\text{Corr}\left(M\left(\text{s}^{\text{A}}_{1,\text{train}};\text{s}^{\text{B}}_{1,\text{train}}\right)_{\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right)}{\sqrt{\text{Corr}\left(M\left(\text{s}^{\text{A}}_{1,\text{train}};\text{s}^{\text{B}}_{1,\text{train}}\right)_{\text{test}},M\left(\text{s}^{\text{A}}_{2,\text{train}};\text{s}^{\text{B}}_{2,\text{train}}\right)_{\text{test}}\right)\times\text{Corr}\left(\text{s}^{\text{B}}_{1,\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right)}}.\end{aligned} \tag{8}$$

In words, the inter-animal consistency corresponds to the predictivity of the mapping on the test set stimuli from animal A to B on two different (averaged) halves of noisy trials, corrected by the square root of the mapping reliability on animal A's test stimuli responses on two different halves of noisy trials and the internal consistency of animal B.

We justify the approximation in (8) by gradually eliminating the true quantities by their measurable estimates, starting from the original quantity in (7). First, we make the approximation that

$$\begin{aligned}&\text{Corr}\left(M\left(\text{t}^{\text{A}}_{\text{train}};\text{t}^{\text{B}}_{\text{train}}\right)_{\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right)\\ &\sim\text{Corr}\left(M\left(\text{t}^{\text{A}}_{\text{train}};\text{t}^{\text{B}}_{\text{train}}\right)_{\text{test}},\text{t}^{\text{B}}_{\text{test}}\right)\times\text{Corr}\left(\text{t}^{\text{B}}_{\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right).\end{aligned} \tag{9}$$

by transitivity of positive correlations (which is a reasonable assumption when the number of stimuli is large). Next, by transitivity and normality assumptions in the structure of the noisy estimates and since the number of trials (n) between the two sets is the same, we have that

$$\begin{aligned}\text{Corr}\left(\text{s}^{\text{B}}_{1,\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right)&\sim\text{Corr}\left(\text{s}^{\text{B}}_{1,\text{test}},\text{t}^{\text{B}}_{\text{test}}\right)\times\text{Corr}\left(\text{t}^{\text{B}}_{\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right)\\ &\sim\text{Corr}\left(\text{t}^{\text{B}}_{\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right)^{2}.\end{aligned} \tag{10}$$

Namely, the correlation between the average of two sets of noisy observations of n trials each is approximately the square of the correlation between the true value and average of one set of n noisy trials. Therefore, from (9) and (10) it follows that

$$\text{Corr}\left(M\left(\text{t}^{\text{A}}_{\text{train}};\text{t}^{\text{B}}_{\text{train}}\right)_{\text{test}},\text{t}^{\text{B}}_{\text{test}}\right)\sim\frac{\text{Corr}\left(M\left(\text{t}^{\text{A}}_{\text{train}};\text{t}^{\text{B}}_{\text{train}}\right)_{\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right)}{\sqrt{\text{Corr}\left(\text{s}^{\text{B}}_{1,\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right)}}. \tag{11}$$

We have gotten rid of $\text{t}^{\text{B}}_{\text{test}}$, but we still need to get rid of the $M\left(\text{t}^{\text{A}}_{\text{train}};\text{t}^{\text{B}}_{\text{train}}\right)_{\text{test}}$ term. We apply the same two steps by analogy though these approximations may not always be true (though are true for Gaussian noise):

$$\begin{aligned}\text{Corr}\left(M\left(\text{s}^{\text{A}}_{1,\text{train}};\text{s}^{\text{B}}_{1,\text{train}}\right)_{\text{test}},\text{s}^{\text{B}}_{2,\text{test}}\right)\sim&\ \text{Corr}\left(\text{s}^{\text{B}}_{2,\text{test}},M\left(\text{t}^{\text{A}}_{\text{train}};\text{t}^{\text{B}}_{\text{train}}\right)_{\text{test}}\right)\\ &\times\text{Corr}\left(M\left(\text{t}^{\text{A}}_{\text{train}};\text{t}^{\text{B}}_{\text{train}}\right)_{\text{test}},M\left(\text{s}^{\text{A}}_{1,\text{train}};\text{s}^{\text{B}}_{1,\text{train}}\right)_{\text{test}}\right)\end{aligned}$$

$$
\begin{aligned}
&\mathrm{Corr}\left(M\left(\mathrm{s}^{\mathrm{A}}_{1,\mathrm{train}};\mathrm{s}^{\mathrm{B}}_{1,\mathrm{train}}\right)_{\mathrm{test}}, M\left(\mathrm{s}^{\mathrm{A}}_{2,\mathrm{train}};\mathrm{s}^{\mathrm{B}}_{2,\mathrm{train}}\right)_{\mathrm{test}}\right)\\
&\sim \mathrm{Corr}\left(M\left(\mathrm{s}^{\mathrm{A}}_{1,\mathrm{train}};\mathrm{s}^{\mathrm{B}}_{1,\mathrm{train}}\right)_{\mathrm{test}}, M\left(\mathrm{t}^{\mathrm{A}}_{\mathrm{train}};\mathrm{t}^{\mathrm{B}}_{\mathrm{train}}\right)_{\mathrm{test}}\right)^2,
\end{aligned}
$$

which taken together implies

$$
\begin{aligned}
&\mathrm{Corr}\left(M\left(\mathrm{t}^{\mathrm{A}}_{\mathrm{train}};\mathrm{t}^{\mathrm{B}}_{\mathrm{train}}\right)_{\mathrm{test}}, \mathrm{s}^{\mathrm{B}}_{2,\mathrm{test}}\right)\\
&\sim \frac{\mathrm{Corr}\left(M\left(\mathrm{s}^{\mathrm{A}}_{1,\mathrm{train}};\mathrm{s}^{\mathrm{B}}_{1,\mathrm{train}}\right)_{\mathrm{test}}, \mathrm{s}^{\mathrm{B}}_{2,\mathrm{test}}\right)}{\sqrt{\mathrm{Corr}\left(M\left(\mathrm{s}^{\mathrm{A}}_{1,\mathrm{train}};\mathrm{s}^{\mathrm{B}}_{1,\mathrm{train}}\right)_{\mathrm{test}}, M\left(\mathrm{s}^{\mathrm{A}}_{2,\mathrm{train}};\mathrm{s}^{\mathrm{B}}_{2,\mathrm{train}}\right)_{\mathrm{test}}\right)}}.
\end{aligned}
\tag{12}
$$

Equations (11) and (12) together imply the final estimated quantity given in (8).

B.2 Multiple Animals

For multiple animals, we simply consider the average of the true quantity for each target in B in (7) across source animals A in the ordered pair (A, B) of animals A and B:

$$
\begin{aligned}
&\left\langle \mathrm{Corr}\left(M\left(\mathrm{t}^{\mathrm{A}}_{\mathrm{train}};\mathrm{t}^{\mathrm{B}}_{\mathrm{train}}\right)_{\mathrm{test}}, \mathrm{t}^{\mathrm{B}}_{\mathrm{test}}\right)\right\rangle_{A\in\mathcal{A}:(A,B)\in\mathcal{A}\times\mathcal{A}}\\
&\sim \left\langle \frac{\mathrm{Corr}\left(M\left(\mathrm{s}^{\mathrm{A}}_{1,\mathrm{train}};\mathrm{s}^{\mathrm{B}}_{1,\mathrm{train}}\right)_{\mathrm{test}}, \mathrm{s}^{\mathrm{B}}_{2,\mathrm{test}}\right)}{\sqrt{\mathrm{Corr}\left(M\left(\mathrm{s}^{\mathrm{A}}_{1,\mathrm{train}};\mathrm{s}^{\mathrm{B}}_{1,\mathrm{train}}\right)_{\mathrm{test}}, M\left(\mathrm{s}^{\mathrm{A}}_{2,\mathrm{train}};\mathrm{s}^{\mathrm{B}}_{2,\mathrm{train}}\right)_{\mathrm{test}}\right)\times\mathrm{Corr}\left(\mathrm{s}^{\mathrm{B}}_{1,\mathrm{test}},\mathrm{s}^{\mathrm{B}}_{2,\mathrm{test}}\right)}}\right\rangle_{A\in\mathcal{A}:(A,B)\in\mathcal{A}\times\mathcal{A}}.
\end{aligned}
$$

Typically, we may bootstrap across split-half trials and have multiple train/test splits, in which case the average on the right hand side of the equation includes averages across these as well.

Note that each neuron in our analysis will have this single average value associated with it when *it* was a target animal (B), averaged over source animals, bootstrapped split-half trials, and train/test splits. This yields a vector of these average values, which we can take median and s.e.m. over as we do with standard explained variance metrics.

B.3 Spearman-Brown Correction

The Spearman-Brown correction is to be applied to each of the terms in the denominator individually, as they are each correlations of observations from half the trials of the *same* underlying process to itself (unlike the numerator).

B.4 Pooled Source Animal

Often times, we may not have enough neurons per animal to ensure that the estimated inter-animal consistency in our data closely matches the "true" inter-animal consistency. In order to address this issue, we holdout one animal at a time and compare it to the pseudo-population aggregated across units from the remaining animals, as opposed to computing the consistencies in a pairwise fashion. Thus, B is still the target heldout animal as in the pairwise case, but now the average over A is over the sole "pooled" source animal $\hat{A}$ constructed from the pseudo-population of the remaining animals. We found that this pooling of the source animal units helped improve the estimated inter-animal consistency, as demonstrated in Fig. S7.

C Model Training Details

All model code be found here: `https://github.com/neuroailab/mec`.

C.1 Simulated Trajectories and Place Cell Representation

Place cell receptive field centers $\vec{c}_i$, $i = 1, \ldots, n_P$, distributed uniformly randomly across each environment. This environment is the $2.2m^2$ arena for all models except for the "Trained" bars in Fig. 2a, which corresponds to training in the $100cm^2$ environment, and the "1D Track" bar in Fig. 2b which corresponds to training on the $400cm$ track. We take $n_P = 512$ place cells in all environments and models, following Banino et al. [2018].

The response of the i-th place cell is simulated using a difference of Gaussians tuning curve, $p_i(x) = e^{-\|x-c_i\|_2^2/2\sigma_1^2} - e^{-\|x-c_i\|_2^2/2\sigma_2^2}$, where x is the current location of the agent, $\sigma_1 = 0.12m$ and $\sigma_2 = 0.12\sqrt{2}m$. Agent trajectories are generated using the rat motion model of Erdem and Hasselmo [2012]. In 1D, during either training or evaluation we prevent the agent from making turns, in order to simulate the head-fixed condition that the mice experience. We collect the place cell activations at n_x locations as the animal explores its environment in a matrix $P \in \mathbb{R}^{n_x \times n_P}$.

C.2 Place Cell Input Models

While the path integrator networks are trained with the place cells as supervised *outputs*, defined in Section C.3, we have several controls based on the place cell representation. The "Place Cell Ground Truth" model directly corresponds to the matrix P.

NMF corresponds to Non-negative Matrix Factorization on the matrix P, implemented via `sklearn.decomposition.NMF`. As noted by Dordek et al. [2016], this corresponds to a 1-layer neural network with n_G hidden units via unsupervised Hebbian learning on inputs P, subject to a nonnegativity constraint. The "Grid Cell Model" corresponds specifically to NMF with $n_G = 9$ components, following Sorscher et al. [2019].

Finally, the "Place Cell Linear MEC" model is intended to be a neural data constrained linear alternative to NMF on the place cell matrix P. This is ElasticNet CV regression trained on 20% of the position bins in the current evaluation environment ($100cm^2$ arena in Fig. 2a and $400cm$ track in Fig. 2b), fitted to the neurons of animals distinct from the current target neural population (namely the units of the source animal defined in Section B).

C.3 Loss Functions

The "Place Cell Loss" corresponds to the loss function used by Banino et al. [2018], which is the softmax cross-entropy loss between the ground truth timestep t place cell targets p_i^t and model outputs $\hat{p}_i^t$, given by

$$\mathcal{L}(\hat{p}, p) := -\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{N_p} p_i^t \log \hat{p}_i^t. \tag{13}$$

The "Position Loss" [Cueva and Wei, 2018] is given by

$$\mathcal{L}(\hat{p}, p) := \frac{1}{2}\frac{1}{T}\sum_{t=1}^{T}\left(\left(p_x^t - \widehat{p_x^t}\right)^2 + \left(p_y^t - \widehat{p_y^t}\right)^2\right), \tag{14}$$

where p_x^t and p_y^t are the *Cartesian* coordinates (x, y) of the agent's ground truth position at timestep t.

Both loss functions are averaged across the batch, where the path length in each batch for both loss functions is $T = 20$ timesteps. Additionally, for either loss function, we apply an L2 penalty of 1×10^{-4} to the path integrator weights.

C.4 Network Architectures and Hyperparameters

C.4.1 Path Integrators

We use Tensorflow 2.0 for these models [Abadi et al., 2016]. The RNN path integrator network takes in 2 linear input units for x and y velocity, a set of recurrently connected input units, and linear readout units. The network update equations are as follows:

$$\begin{aligned} g^{t+1} &= f\left(Jg^t + Mv^t\right) \\ \hat{p}^{t+1} &= Wg^{t+1}, \end{aligned} \tag{15}$$

where g is the vector of model MEC activities (4096 units total), J is the matrix of recurrent weights, M is the network's velocity input weights, v is the agent's 2D velocity in the arena, f is the element-wise nonlinearity (or the identity function if it is "Linear"), $\hat{p}$ is the vector of estimated place cell activities (or outputted Cartesian coordinates if the network is being trained with the "Position Loss"), and W is the place cell (or Cartesian coordinate) readout weights.

The SRNN, GRU, LSTM, and UGRNN path integrator networks had an identical task and training protocol as (15). The model architecture was reproduced from Banino et al. [2018] except that our models had 4096 units (rather than 128) in order to match the number of units of the MEC layer (g) of the RNN path integrator above. Specifically, it consists of 2D velocity inputs to a recurrent circuit (SRNN, GRU, LSTM, or UGRNN) with 4096 units with nonlinearity either being Linear, Tanh, Sigmoid, or ReLU, followed by a nonlinear layer of 4096 units (which constitutes the model's activities g, with the nonlinearity matching that of the recurrent circuit, following Sorscher et al. [2019]), followed by a final readout to the estimated 512 place cell activities (or 2 Cartesian units of position if training with the "Position Loss"). Furthermore, the initial cell state and hidden state are initialized by computing a linear transformation of the ground truth place cells (or Cartesian positions if training with the "Position Loss") at time 0. We did not employ any dropout at the g layer of these networks during training.

All networks are trained with Adam [Kingma and Ba, 2015] with a learning rate of 1×10^{-4}, batch size of 200, and 100 training epochs consisting of 1000 batches of trajectories per epoch.

C.4.2 Cue Input

MEC neurons are known to respond to visual input, but as none of our evaluated models had a visual front-end, they could not respond accordingly when evaluated on a cue-rich environment (which was the case for the 1D data). To test this hypothesis, we trained the UGRNN-ReLU-Place Cell model in the $2.2m^2$ 2D arena, but with input visual cue locations concatenated with the 2D velocity input, corresponding to the “Cue Input + Velocity Input” model in Fig. 2b. Specifically, these visual cue locations were a vector $\ell \in \mathbb{R}^5$, corresponding to 5 cues placed in fixed, arbitrary locations in the 2D arena with widths between 0.06 to $0.3m$ on each side. Each element ℓ_i of this cue input vector corresponds to the Euclidean distance of the agent at current time t to the nearest boundary of the i-th cue (and 0 for that entry if the agent is within the boundaries of this i-th cue). We also considered a UGRNN-ReLU-Place Cell network trained in 2D without any velocity input and only the cue input, corresponding to the “Cue Input Only” bar in Fig. 2b.

Finally, we evaluated these networks with the 1D cue input which matched the widths and locations of the cues in the 1D virtual track. The neural predictivity of this 1D cue input is the “Cue Input” bar in Fig. 2b. As a control, we also included a UGRNN-ReLU-Place Cell path integrator trained with the usual 2D velocity input but with constant 0 concatenated to the velocities in place of the active cue input (thus being functionally equivalent to the path integrators in Section C.4.1). The network was then provided the 1D cue inputs during evaluation only, corresponding to the “Evaluation Cue Input Only” bar in Fig. 2b.

C.4.3 Reward Biased Path Integrator

For the reward biased path integrator (parametrized by ε in Fig. 5b), we trained the UGRNN-ReLU-Place Cell in the $2.2m^2$ arena, where on $(1-\varepsilon)$ fraction of training batches (each example in the batch corresponding to a 20 timestep path-length episode), we had the agent navigate directly within a fixed number of timesteps (7) to the center of a $20cm^2$ reward zone placed in an arbitrary, fixed location of the environment. In the remaining 13 timesteps of these episodes, the agent path integrated using the motion trajectories of [Erdem and Hasselmo, 2012] but restricted to the $20cm^2$ reward zone. In the other ε fraction of episodes, the trajectories were unchanged from before.

As a control, we consider a “Reward Input” path integrator (Fig. 5b,c), which does *not* employ the reward biased trajectories (so $\varepsilon = 1$), but instead takes an additional scalar reward signal (concatenated with the 2D velocity input) during training, indicating if it is in the reward zone or not at current timestep t.

D Performance Measure and Ablation

Positional decoding error for the place cell loss models is measured first by computing a predicted Cartesian position $\hat{p}_x^t, \hat{p}_y^t$, obtained by taking the top 3 most active place cell outputs at each timestep in each model and averaging them. Finally, for each trajectory episode (of length $T = 20$), we compute the following measure of path integration error,

$$\mathcal{E}(\hat{p}, p) := -\frac{1}{T}\sum_{t=1}^{T}\sqrt{\left((p_x^t - \widehat{p}_x^t)^2 + \left(p_y^t - \widehat{p}_y^t\right)^2\right)}, \tag{16}$$

where the error is additionally averaged over the batch dimension (20,000 examples computed from 100 batches of 200 evaluation trajectory episodes each).

The performance degradation metric that is the y-axis of Fig. 3d is given by $\left(\mathcal{E}^{curr} - \mathcal{E}^{full}\right)/\mathcal{E}^{full}$, where $\mathcal{E}^{full}$ is the performance of the trained UGRNN-ReLU-Place Cell model, and $\mathcal{E}^{curr}$ is the performance of the same model but with a subset of its population of units in the g layer set to 0. In Fig. 4d, for the “grid cell knockout”, the outputs of the set of units in layer g with grid score > 0.3 are set to 0 during evaluation, and for the “heterogeneous cell knockout”, the same number of units in layer g with grid score ≤ 0.3 are randomly set to 0 (subsampled 100 times). Analogously, for the “border cell knockout”, the border score was set to a threshold of 0.5, and the heterogeneous knockout in that case was border score ≤ 0.5.

E Supplementary Figures

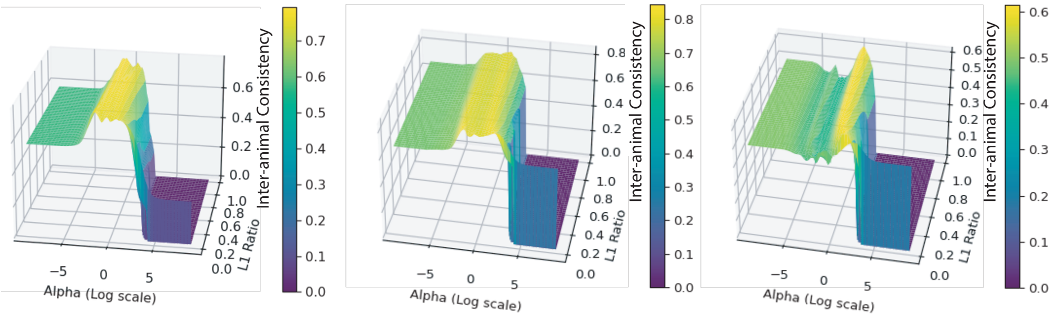

Figure S1: **Inter-animal consistency as a function of α and L1 ratio (sparsity penalty strength).** For $\alpha \in [10^{-9}, 10^{9}]$ logspaced uniformly and L1 ratio spaced uniformly $\in [0, 1]$, we plot the inter-animal consistency evaluated on 80% of position bins for a given train-test split and cell on the $100cm^2$ arena.

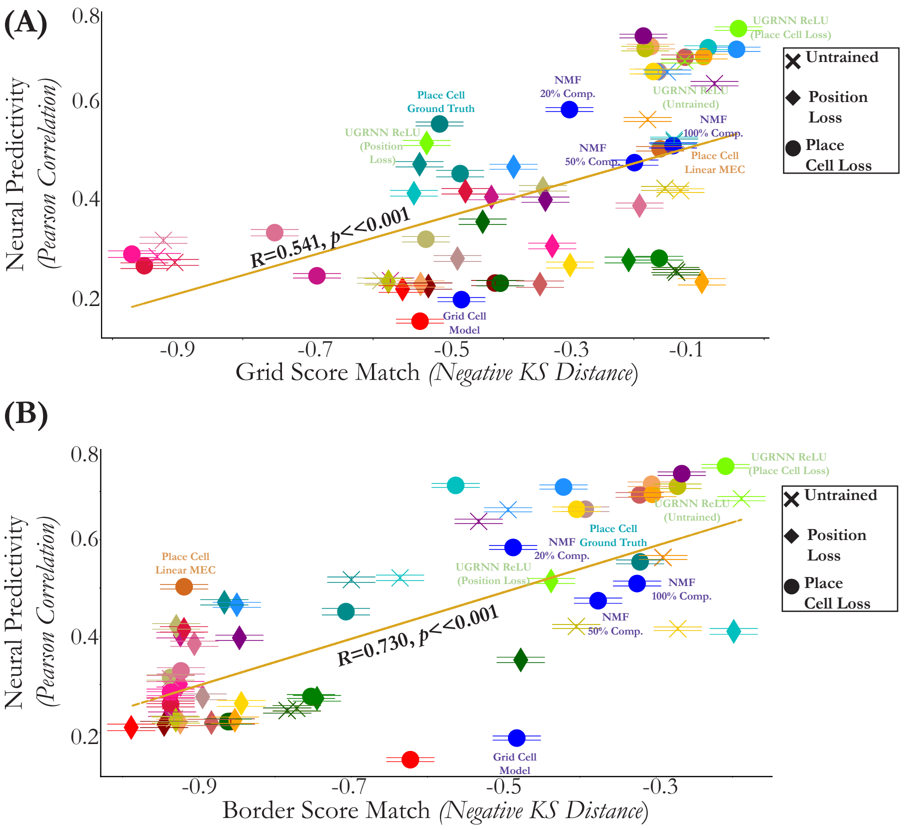

Figure S2: **Neural predictivity and grid & border score distribution match are related.** Each model's neural predictivity on the $100cm^2$ foraging data versus grid score (top) and border score (bottom) distribution match. The neural predictivity is the median and s.e.m. across 620 cells. The models are either untrained ("X"), trained with the place cell loss ("O") in the $2.2m^2$ arena, or trained with the position loss ("Diamond") in the $2.2m^2$ arena.

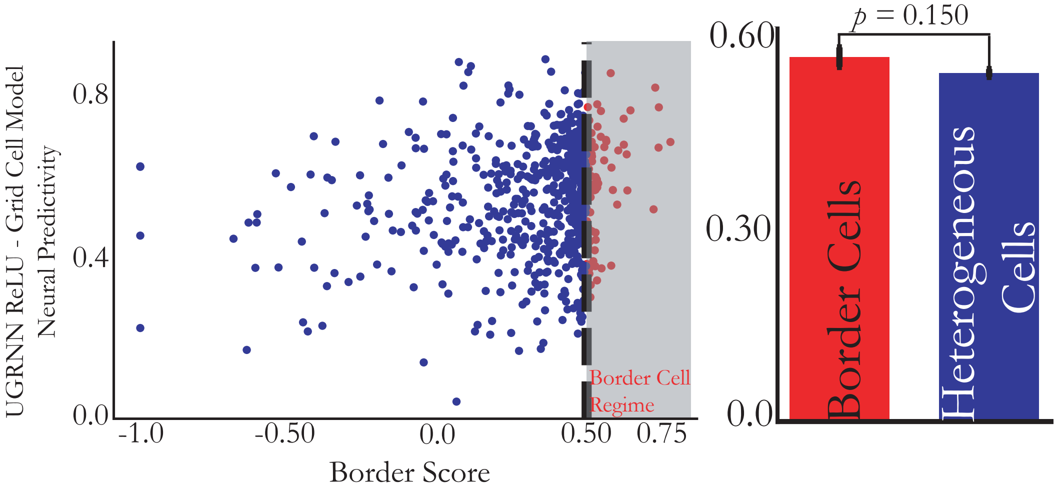

Figure S3: **Difference between UGRNN-ReLU-Place Cell and Grid Cell models (Border Score).** (*Left*) Per unit neural predictivity difference between the UGRNN-ReLU-Place Cell and Grid Cell models trained on the $2.2m^2$ arena and evaluated on the $100cm^2$ arena, plotted against that unit's border score. (*Right*) Quantification of this difference aggregated across the border cell and heterogeneous (non-border) cell populations, respectively (mean and s.e.m).

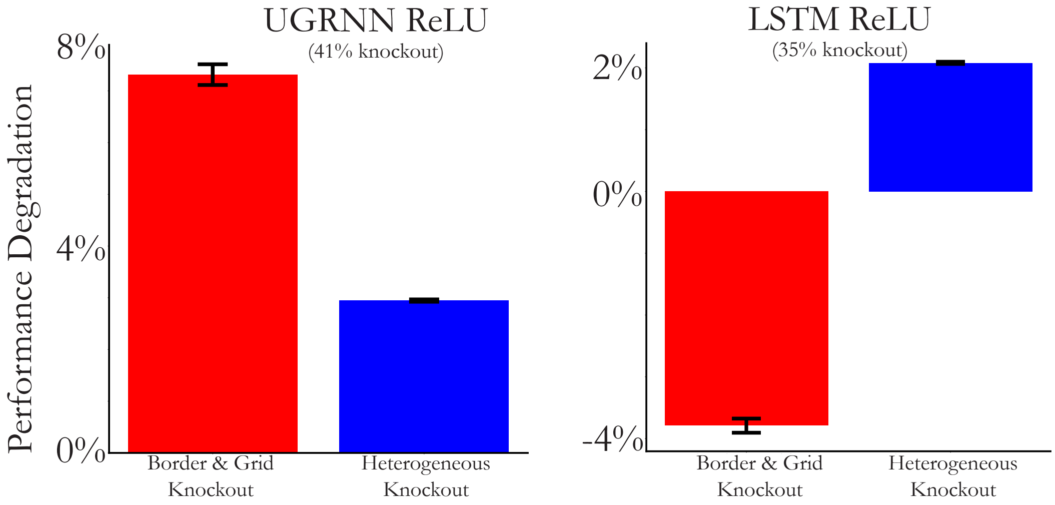

Figure S4: **Combined border and grid cell knockout.** For the UGRNN-ReLU-Place Cell and LSTM-ReLU-Place Cell models trained on the $2.2m^2$ arena, we select grid cells (grid score > 0.3) and border cells (border score > 0.5) to knockout (red), and knockout the same number of heterogeneous cells (neither border nor grid cell), randomly sampled 100 times. The percentage knockout refers to the percentage of total units knocked out in the current layer, corresponding to the intermediate layer of the three layer network. Mean and s.e.m. over evaluation episodes.

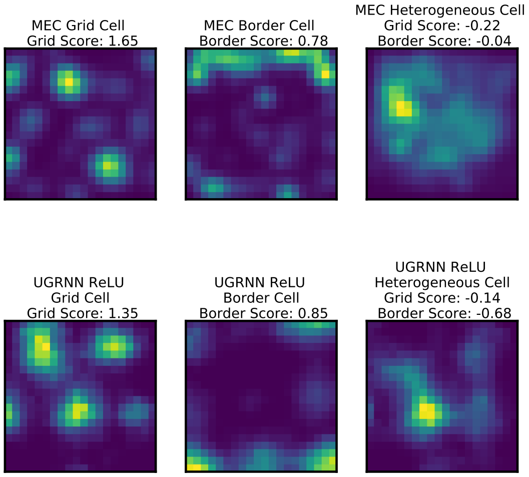

Figure S5: **Example rate maps.** Example rate maps obtained from animals foraging in the $100cm^2$ arena (top row) and rate maps from the UGRNN-ReLU-Place Cell trained in the $2.2m^2$ arena and evaluated on the $100cm^2$ arena. We also include the grid and border scores of these example units.

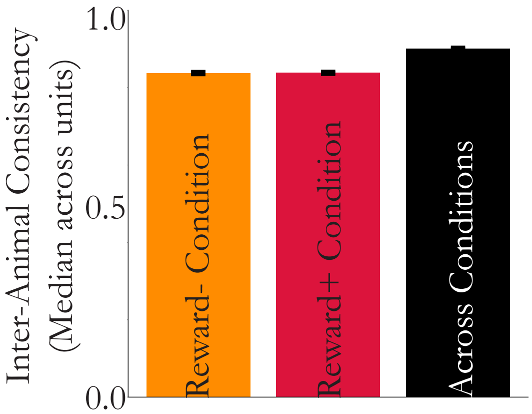

Figure S6: **Inter-animal consistency is high per condition and across conditions.** Inter-animal consistency of neural responses (under ElasticNet CV regression) is computed per **reward-** and **reward+** condition separately, and across both spatial and reward-modulated conditions overall. Median and s.e.m. across 598 MEC cells from 7 rats in the $150cm^2$ arena.

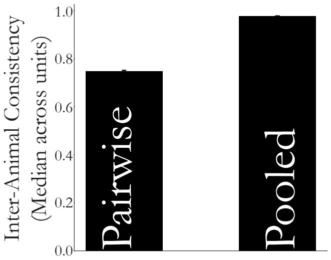

Figure S7: **Pooling across source animals.** Inter-animal consistency under Ridge regression ($\alpha = 1$) trained with 50% position bins (averaged across ten train-test splits) from 12 mice foraging in the $100cm^2$ arena. Median and s.e.m. across 620 cells. “Pooled” refers to computing the inter-animal consistency using a single “pooled” source animal from units gathered from all animals except the target animal, as described in Section B.4. “Pairwise” refers to computing this quantity mapping one source animal at a time to the target animal.

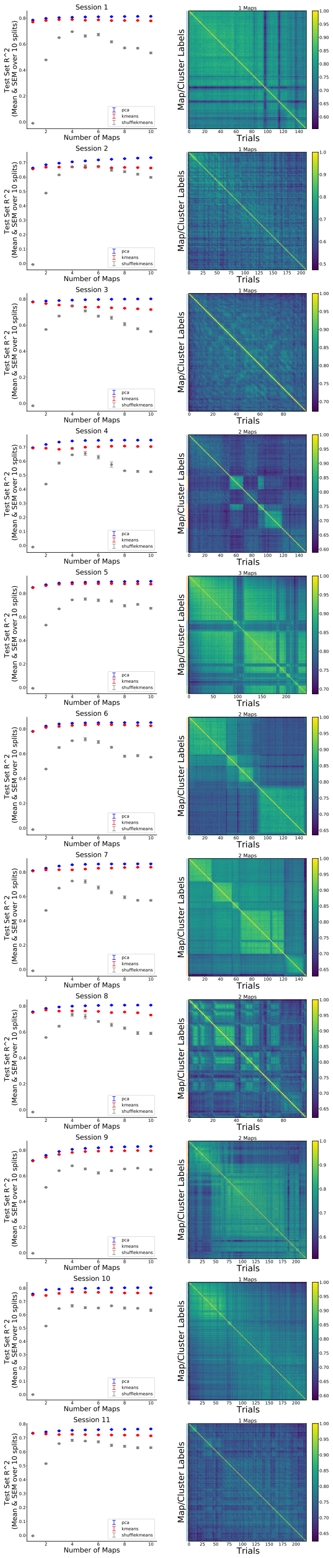

Figure S8: **Remapping analysis for 1D VR data.** For each recording session (11 total), we examine the trial-by-trial similarity matrices (right column) and the test set R^2 of k-means clustering as a function of k (left column) relative to both PCA (blue) and a shuffled control (gray). The number of identified maps per session is listed at thee top of each trial-by-trial similarity matrix, along with the colored cluster labels as to the assignments for each trial.