The Spatial Memory Pipeline: a model of egocentric to allocentric understanding in mammalian brains

Benigno Uria*[,1], Borja Ibarz*[,1], Andrea Banino[1], Vinicius Zambaldi[1], Dharshan Kumaran[1], Demis Hassabis[1], Caswell Barry[2], Charles Blundell[1]
[1]DeepMind
[2]University College London
*Equal contribution

In the mammalian brain, allocentric representations support efficient self-location and flexible navigation. A number of distinct populations of these spatial responses have been identified but no unified function has been shown to account for their emergence. Here we developed a network, trained with a simple predictive objective, that was capable of mapping egocentric information into an allocentric spatial reference frame. The prediction of visual inputs was sufficient to drive the appearance of spatial representations resembling those observed in rodents: head direction, boundary vector, and place cells, along with the recently discovered egocentric boundary cells, suggesting predictive coding as a principle for their emergence in animals. The network learned a solution for head direction tracking convergent with known biological connectivity, while suggesting a possible mechanism of boundary cell remapping. Moreover, like mammalian representations, responses were robust to environmental manipulations, including exposure to novel settings, and could be replayed in the absence of perceptual input, providing the means for offline learning. In contrast to existing reinforcement learning approaches, agents equipped with this network were able to flexibly reuse learnt behaviours - adapting rapidly to unfamiliar environments. Thus, our results indicate that these representations, derived from a simple egocentric predictive framework, form an efficient basis-set for cognitive mapping.

Introduction Animals navigate easily and efficiently through the world[1]. Yet artificial agents struggle with even simple spatial tasks requiring self-localisation and goal directed behaviours. In mammals, the neural circuits supporting spatial memory have been studied extensively. Empirical work has generated a detailed knowledge about populations of spatially modulated neurons – place[2], grid[3], and head direction[4] cells represent body position and direction of facing, while border responsive neurons encode immediate environmental topography[5,6]. Current thinking sees these networks as components of a cognitive map thought to provide an efficient spatial basis, allowing the structure of novel environments to be learnt rapidly and subsequently supporting flexible navigation, such as short-cuts, detours, and novel routes to remembered goals[2,7].

Surprisingly, there are significant gaps in our understanding. In particular, it is unclear how these populations interact to support spatial behaviour and how they are themselves derived and updated by incoming sensory information. Fundamentally, we lack a unifying computational account for the emergence of allocentric (world centred) representations from egocentric (self centred) sensory experience[8,9]. This poses a problem for neuroscientists, who seek simplifying normative accounts of the brain, and for AI practitioners who aim to build systems with navigational abilities that match those of animals.

Historically, theoretical approaches to this problem largely presented hand-coded designs fo-

cused on a single layer of the biological network[10,11]. A smaller number of models have presented a more extensive description of hippocampal networks[12], including the process by which representations might be learnt[13]. Notably a potential architecture for ego–allocentric transformation[14] successfully anticipated the existence of egocentric border responsive neurons[15–17]. Still, in almost all cases existing models are the product of extensive human oversight.

Here we show that a deep learning model trained to predict visual experience from self-motion is sufficient to explain the hierarchy of egocentric to allocentric representations found in mammals, bridging the gap from visual experience to place cells. In particular, we identify head-direction cells, border cells with both ego and allocentric responses, and place cells. Critically, we do so without the need for any allocentric inputs. Like their biological counterparts, the firing fields of these units are shaped by the surrounding cues - particularly environmental boundaries - but their characteristics are retained across different environments.

Thus, when embodied in artificial agents, this redeployment of existing responses supports rapid adaptation to novel environments, enabling effective navigation to remembered goals with limited experience of the test enclosure –demonstrating the flexible transfer of skills at which animals excel but artificial systems have struggled. Finally, artificial agents with these representations are able to coherently replay hypothetical rollouts of the future, providing a potential basis for model-based planning and offline learning[18].

Spatial Memory Pipeline The hippocampal formation has been characterised as a predictor-comparator, comparing incoming perceptual stimuli with predictions derived from memory[19–21]. With this in mind we trained a deep neural network to predict the forthcoming visual experience of an agent exploring virtual environments. Specifically, we developed a model composed of a modular recurrent neural network (RNN) with an external memory– the 'Spatial Memory Pipeline' (see supplementary text, Extended Data Fig 1). At each time-step visual inputs entering the pipeline were compressed using a convolutional neural network and compared to previous embeddings stored in the slots of a memory store, the best matching being most strongly reactivated. The remainder of the pipeline was trained to predict which visual memory slot would be reactivated next (Fig 1A). Predictions were generated solely on the basis of self-motion information, provided differentially to RNN modules as angular and linear velocity, in addition to visual corrections from previous steps. There was no direct pathway from detailed visual features to latter parts of the pipeline – visual information being communicated only by the activation of slots in the memory stores (Fig 1B). Thus the Spatial Memory Pipeline forms a predictive code[22,23], anticipating which future visual slot will be activated, based solely on self-motion.

Emergence of allocentric representations In our first unsupervised-learning experiment, the Spatial Memory Pipeline was trained in a simulated square environment (2.2 by 2.2 meters) resembling those used in rodent studies – plain white walls with visible distal cues (Fig 1C, D), using a rat-like motion model[24]. After training, the network was able to accurately predict the reactivation of visual memories (80% loss reduction with respect to uniform distribution, Fig 1E), effectively integrating self-motion information to anticipate the visual scene. To understand how the network performed this task we inspected the activity profile of units in the modular RNNs. These units exhibited a range of spatially stable allocentric responses strongly resembling those found in the mammalian hippocampal spatial memory system, including head-direction (HD) cells, boundary vector cells (BVCs), and place cells, as well as egocentric boundary vector cells

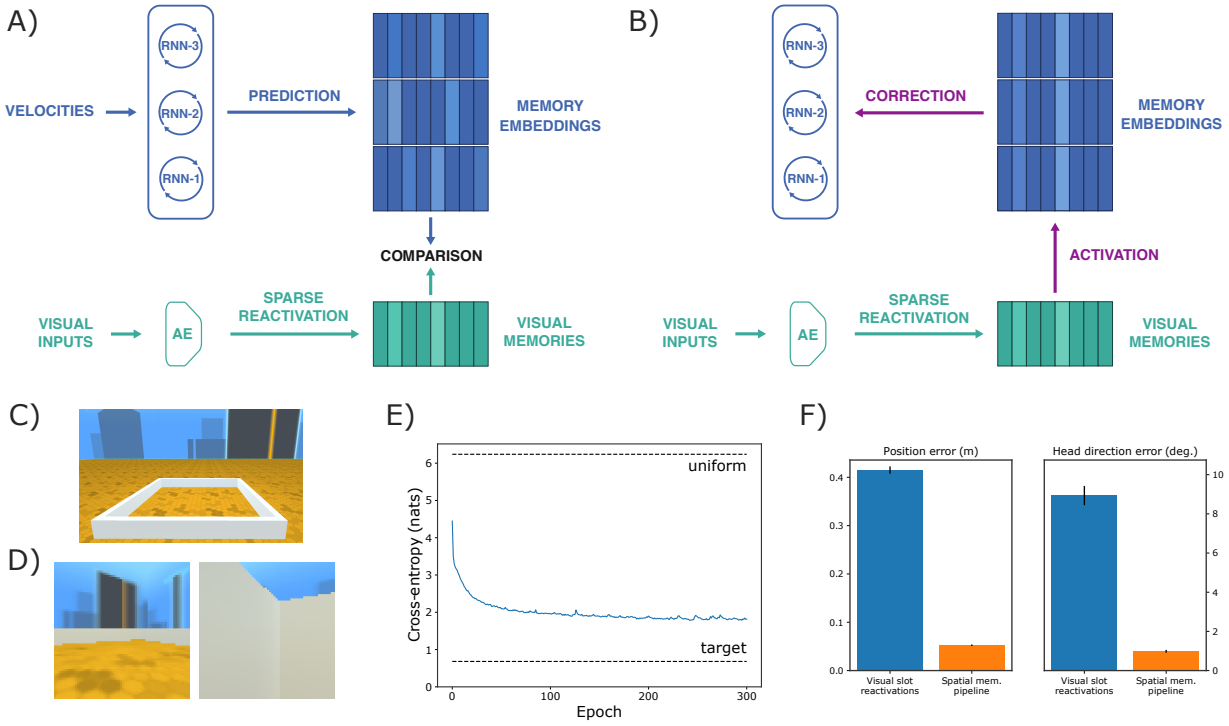


Figure 1: **A, B)** Computational diagrams for the Spatial Memory Pipeline. A) In the prediction step, a set of RNNs anticipate the reactivation of past visual memories by integrating egocentric velocities –AE denotes autoencoder. B) In the correction step, the reactivated visual memories correct the RNNs' state to reduce accumulated errors. **C)** External view of white square enclosure with distal cues. **D)** Agent's view from the middle of the enclosure (left) and close to a wall (right). **E)** Prediction loss along training. The model reduces the uncertainty in visual reactivations by 80% with respect to a uniform distribution. **F)** Average decoding error of position and head direction from visual slot activations and spatial memory pipeline RNNs, black bars show one standard error of the mean.

(egoBVCs) (Fig 2, see Methods). Indeed, in the trained network, the agent's position and direction of facing could be accurately decoded from the RNN activations (Fig 1F, see Supplementary Methods). Most strikingly, these representations were complementary to one another – each RNN module developed distinct response patterns, dependent on the form of self-motion input it received (Fig 2A-D). Similar results were observed in environments with different geometries (see supplementary text, Extended Data Fig 2).

The majority of units in the first RNN module (RNN-1), which received only angular velocities and corrections from the visual memories, exhibited activity that was modulated by the agent's direction of facing (Fig 2A). These responses were strikingly similar to those of head-direction cells[4,25] – individual units having a single preferred firing direction invariant across spatial locations (Fig 2B). In total 91% (29/32) of units were classified as HD cells (resultant vector length $>$0.58, 99th percentile of shuffled data, see Methods) with unimodal responses (average tuning width 76.2^o) distributed uniformly in the unit circle (Fig 2H, uniform distribution was selected under Bayes Information Criteria (BIC) over mixtures of Von Mises, see Supplementary Methods).

The second module (RNN-2) received angular velocity and speed. Here units exhibited more complex patterns of activity (mean resultant vector = 0.02, 0/128 units classified as HD cells), encoding distances to boundaries at a particular heading (Fig 2C). Thus, they appeared to be similar to egocentric BVCs[15–17] which are theorised to play a central role in transforming between ego and allocentric reference frames[14,26]. To quantify this impression we adopted an egoBVC metric applied previously[17] (see Methods). In this way 58 of 128 units (45.3%) were identified as egoBVCs (ego-boundary score $>$0.07, 99th percentile of bin-shuffled distribution, Fig 2I), with the remaining cells displaying mixed patterns of head-direction and egocentric distance to a wall. Rodent egoBVCs exhibit a uniform distribution of preferred firing directions - albeit with a slight tendency to cluster to the animal's left and right side - while preferred distances have been reported up to 50cm, with shorter range responses being more numerous[17]. Analysis of the model egoBVCs (Extended Data Fig 3A-C) revealed a similar pattern of distances, but a cluster of cells responded to a wall directly in front - plausibly reflecting the monocular input and narrower field of view available to the model (60^o vs $>180^o$ in rodents) [27] .

In contrast to the first two RNNs, the third (RNN-3) received no self-motion inputs, thus being dependent upon temporal coherence[13], and corrections from mispredictions as its sole input and learning signal. Units in this layer were not modulated by heading direction (mean resultant vector = 0.11, 0/128 units classified as HD cells), but were characterised by spatially stable responses (Fig 2F) often extending parallel to particular walls of the environment (Fig 2D). As such, the activity profile of this module was reminiscent of boundary vector cells (BVCs), which form an allocentric representation of space defined by environmental boundaries[5,6,28,29]. To assess the units' activity we applied a similar approach to that used for RNN-2 – calculating the resultant vector of the units' activity projected into allocentric boundary space (see Methods). Applying this measure to RNN-3 confirmed that 80.5% (103/128) of units were classified as BVCs (BVC score $>$0.11, 99th percentile of shuffled distribution) – 64% (82/128) met the criteria for egoBVCs, but 70 of those 82 had a higher BVC score than egoBVC score (Fig 2I). The same approach applied to RNN-2 identified 0 of 128 units as BVCs. Analysis of the preferred firing directions and distances of the model BVCs revealed a uniform distribution of directions (Extended Data Fig 3E).

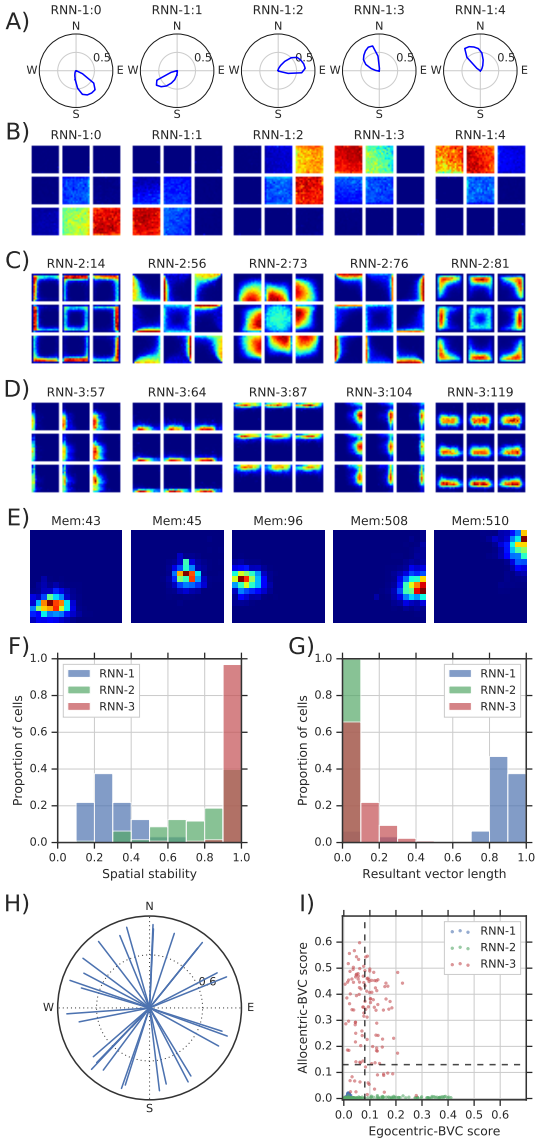


Figure 2: Representations in a spatial memory pipeline trained in the square enclosure shown in Fig 1C. **A)** Polar plot of average activity by heading direction of five units classified as HD cells. **B)** Spatial ratemaps of the units shown in A. Central plot shows average activation for each location across all head directions. The eight plots around each central plot show location-specific activity restricted to times when the heading direction of the agent was in the the corresponding 45° range (e.g. plot located above the central plot shows average activity when the agent was facing in the north direction). **C)** Spatial ratemaps of five units classified as egoBVCs. **D)** Spatial ratemaps of five units classified as BVCs. **E)** Spatial ratemaps of five memory slots reactivated by RNN-3 resembling hippocampal place-cell activity. **F)** Spatial stability of units in each RNN. **G)** Resultant vector length of units in each RNN. **H)** Resultant vector of each unit in RNN-1. **I)** Comparison of egocentric versus allocentric scores for units in each RNN. Dashed lines indicate cell-type classification thresholds.

In the brain, BVCs are hypothesised to be a core contributor to the spatial activity of place cells[28]. Consistent with this, BVC-like responses are found in mEC, afferent to hippocampus, as well as within the hippocampus proper[5,6,29]. In our model, the allocentric representation of RNN-3 were memorised as a second set of targets - being reactivated at each time step by comparison to the current state of the RNN-3. The activity of these memory slots strongly resembled place cells with sparse, spatially localised responses (average spatial field size of 0.21 m^2 SD:0.26 to a total environment area of 4.84 m^2) that were stable across trajectories, and independent of head direction (see Fig 2E and Extended Data Fig 4).

Responses to geometric transformations Simple manipulations of the test environment, such as rotating distal cues or stretching the enclosure, produced commensurate changes in the receptive fields of spatially modulated cells without retraining the model (Extended Data Fig 5, Extended Data Fig 6, and Supplemental Results). Providing a direct parallel with the responses of rodent spatial neurons after environmental transformations[30,31]. Similarly, an extra barrier inserted into a familiar environment becomes a target for BVC and egoBVC responses (Extended Data Fig 7, and Supplementary Results), causing some place fields to duplicate and others to be suppressed[5,6,29].

Learned attractor dynamics in the head-direction system In the mammalian brain, head-direction tuning is widely believed to originate from a neural ring attractor which constrains activity to lie on a 1D manifold - when appropriately connected with neurons responsive to angular velocity, this provides a mechanism to integrate head turns[32]. In our model, when RNN-1 was instantiated separately and provided only with angular velocities (see Methods), it displayed a single activity bump that closely tracked the apparent head direction for several hundred steps, effectively integrating angular velocity over periods that greatly exceeded the duration of training trajectories (Fig 3A). Projecting the observed activity vectors onto their first two principal components revealed that they resided close to a 1-D circular manifold (Fig 3B), a characteristic associated with linear continuous attractor systems. Indeed, with zero angular velocity inputs, random initialisations of the network state quickly converged to a set of point attractors (Fig 3C-D, see Methods).

In contrast, positive or negative angular velocities drove the state to periodic orbits along 1D cyclic attractors (Fig 3E-F). In the mammalian head direction system these dynamics[4] are hypothesised to be supported by a double ring network with each ring having counter rotated tuning - a similar solution has been observed in the fly[33]. A strikingly similar connectivity can be learned by simple RNN formulations[34] (Extended Data Fig 8, Supplementary Results).

Angular velocity integration alone is not sufficient to maintain a stable representation of head direction – visual cues are required to reset the activity of units and correct for drift. To investigate how our model incorporates visual information in its representation of heading, we simulated the input of visual corrections (512 images from the training environment) and zero angular velocity (see Methods). The majority of images (352/512) resulted in network dynamics with a single attractor point per image, regardless of the initial network state (Fig 3G). These images typically display unambiguous distal cues (Fig 3H). The remaining images (160/512) did not result in a single attractor point, possibly reflecting the geometrical symmetries of the environment. Interestingly, upon visual examination, these images usually lack distal cues and correspond to views of walls and corners (Fig 3I).

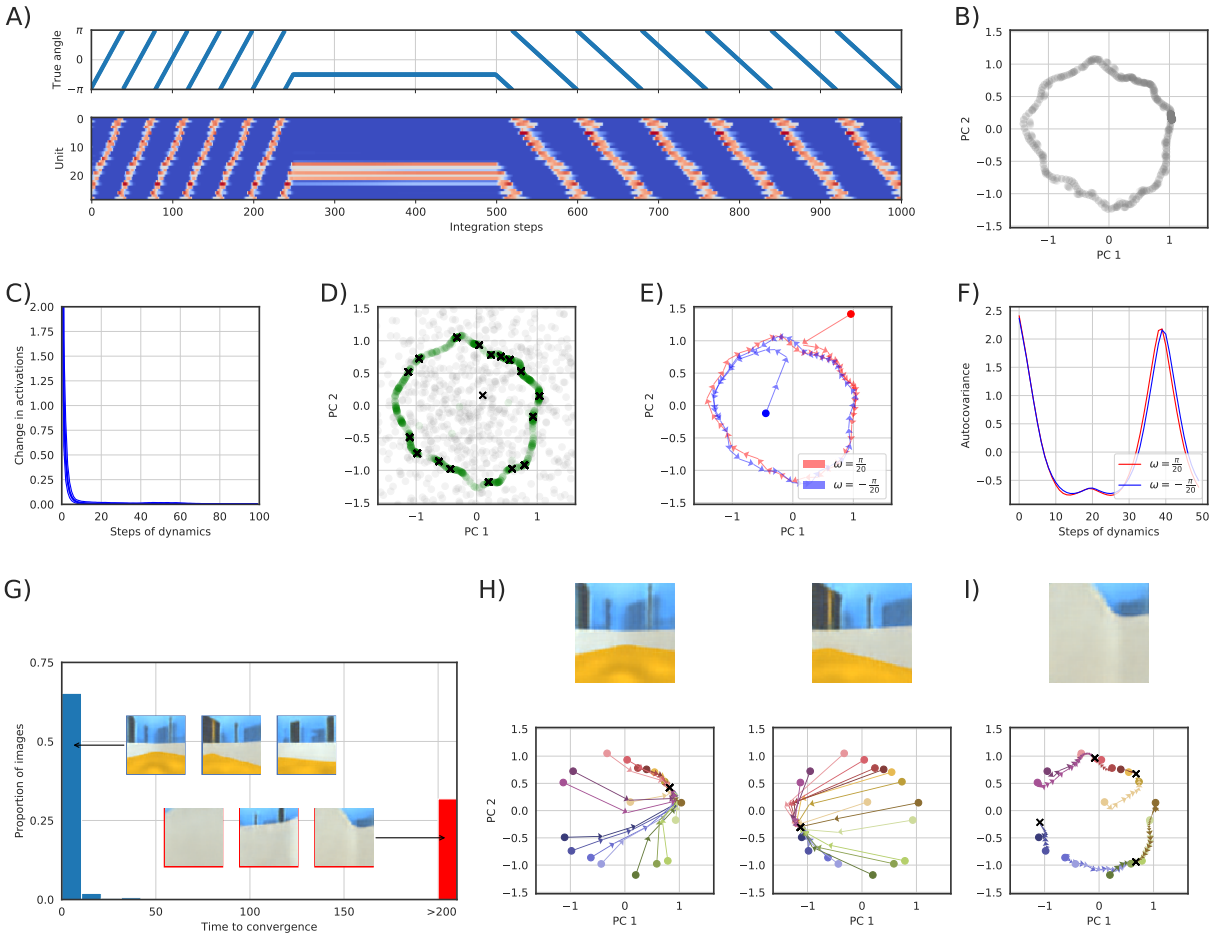


Figure 3: Head-direction cell dynamics. **A)** Activation of each unit over 1000 steps of blind integration. Top: True integrated angle. Bottom: Activation of HD-units in RNN-1 through time (units ordered by the phase of their resultant vectors). For the first 250 steps the angular velocity was set to $\pi/20$ rad/step, the following 250 steps 0 rad/step, the remaining 500 steps $-\pi/40$ rad/step. **B)** Projection of the 1000 activity vectors shown in A onto their first two principal components. **C)** Euclidean distance between successive states in 1000 trajectories with zero angular velocity and random initialisation (see Methods). Thick blue shows the average over one-thousand different initialisations, the thin lines show the 5-th and 95-th percentiles. **D)** Grey shows the projection of the random initial states used in the simulations of panel C onto the space of panel B. Green shows the state after 10 steps of dynamics with zero angular velocity. Black crosses show the states after 1000 steps. All the trajectories converged to a discrete set of attractor states. **E)** Projection on the space of panel B of two 50-step trajectories with constant angular velocity input, one positive and one negative, each starting from a random initialisation. **F)** Autocovariance of states as a function of time lag computed over 1000 randomly initialised 200-step trajectories with positive and negative velocities. **G)** Histogram of steps to convergence to a single attractor state for 512 correction images. Insets show examples with quick convergence (blue) and some that do not converge to a single state (red). **H)** Several example state trajectories, initialised from the attractor states in panel D (different colours), providing visual inputs with unambiguous cues (top) for 200 steps. Black crosses show the final states. **I)** Same as H with ambiguous visual input of a wall.

Spatial characteristics are preserved across environments Next, we sought to establish how stable the different representations in our model were between environments that changed both in terms of their geometry and visual composition. To this end the network was trained concurrently in three environments inspired by empirical studies – a square, a circle, and a trapezoid, all with distinct distal cues, floor and wall textures (Fig 4A, Extended Data Fig 9). The same RNNs were used throughout but different external memory stores were provided for each environment. Despite the different training environments we again found similar proportions of head direction, egoBVC, BVC, and place cells in the network (see Supplementary Results). Indeed, although the three environments were visually and geometrically distinct, the classification of units did not materially change between them: 84% (27/32) RNN-1 units were classified as HD cells, 78% (100/128) of RNN-2 units as egoBVCs, and 61% (78/128) of RNN-3 units as BVCs, in all environments simultaneously.

Predicted mechanism of BVC activity A full understanding of the spatial circuits in the mammalian brain will require resolving the functional dependence between different representations. In our model the relationship between the HD and BVC systems is determined by their joint association to visual memories (Extended Data Fig 10) - thus they rotate coherently to follow changes in the angular location of a familiar distal cue[35] (Extended Data Fig 5). Conversely, in different enclosures with non-overlapping sets of visual memories, groups of BVCs may rotate and flip relative to the HD-system while retaining their internal coherence[5]. The fast association to memories allows rapid redeployment of the representations in novel settings (see next section). This mechanism of HD and BVC dependence contrasts with the hypothesis that postulates BVC activity as the result of direct conjunction of HD and egoBVC activations[28]. Hence, new experimental studies on animals jointly recording the HD and BVC systems across enclosures that elicit global remapping will be required to ascertain the mechanisms that give rise to BVCs in several neocortical areas.

Representation re-use allows transfer of behaviour In animals, allocentric representations such as head-direction cells and BVCs are rapidly redeployed in novel settings - likely as a result of being constrained to low-dimensional manifolds which decouple the integration of self-motion from high-dimensional perceptual experience[5,36]. We hypothesised that this self-consistency is an adaptive characteristic allowing spatial behaviour learned in one environment to be quickly transferred to novel environments.

To test this proposal we incorporated the Spatial Memory Pipeline into a deep reinforcement learning agent[37] trained to find an invisible goal in a 4-room enclosure (Fig 4B), a task inspired by the classic Morris water maze[38]. This task captured two forms of localisation: locally within a room (where in the room?), and globally among all rooms (which room?). When the agent reached the goal, it received a reward and was teleported to a random location in the enclosure - to maximise reward it had to reach the goal as many times as possible within each episode. Three visually distinct enclosures were used simultaneously for training (Fig 4C). Separate memory stores were used for each enclosure but RNNs were shared.

Once trained, the agent reached a high degree of proficiency in the task, routinely following the shortest path to the goal (better than human performance). Inspection of the memory pipeline confirmed that the RNNs contained head-direction cells, egoBVCs, and BVCs with response characteristics that were consistent across the three visually distinct environments (Extended

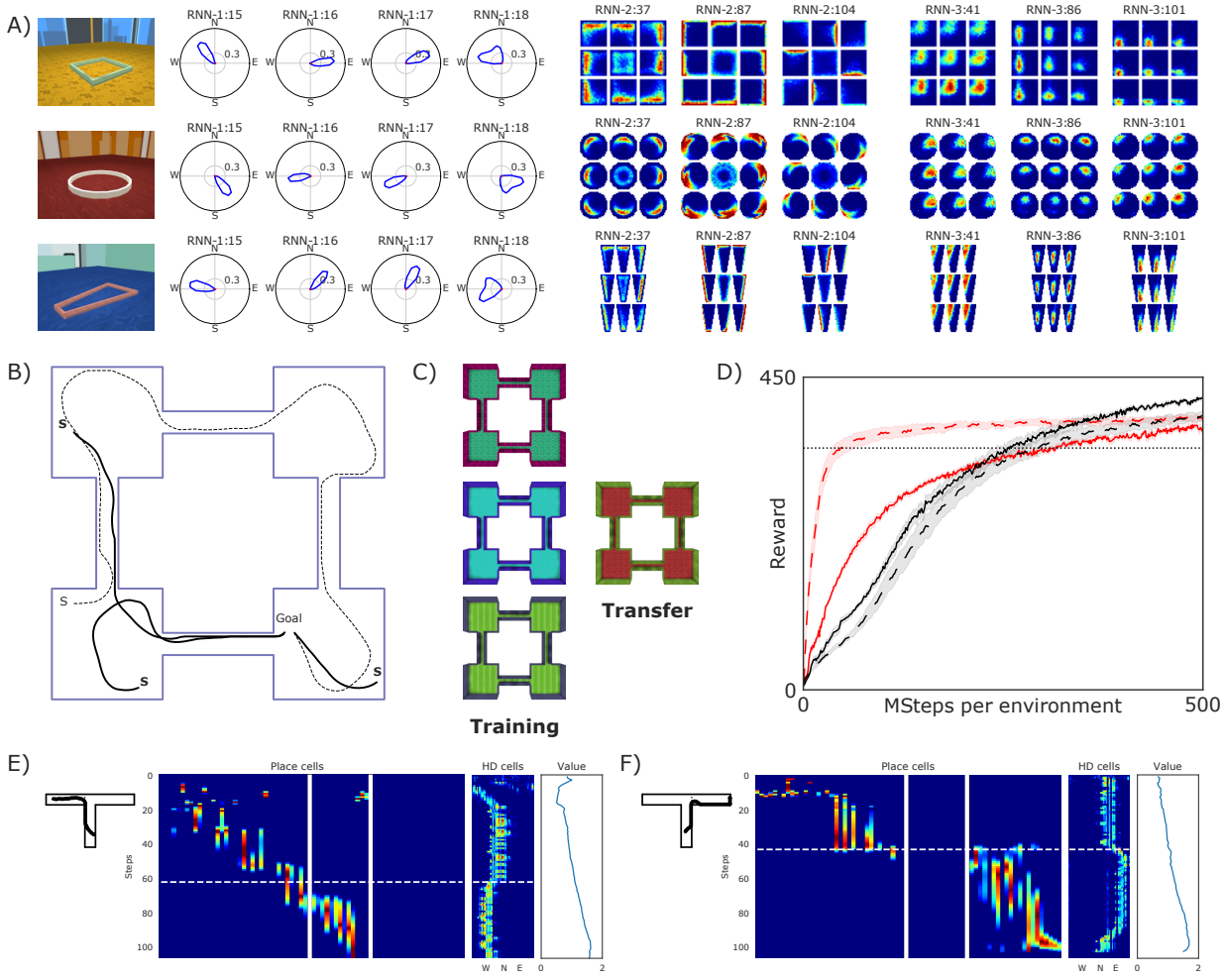


Figure 4: **A)** Stability of spatial representations across three environments, one per row. Left to right: view of the enclosure; allocentric orientation tuning of four units classified as HD cells; spatial ratemap of three units classified as egoBVCs; and spatial ratemap of three units classified as BVCs. **B)** Reinforcement learning task in a 4-room enclosure. Spawning points marked with an S. The initial trajectory to the reward is dotted, later trajectories solid. **C)** Floor and wall textures of the three training enclosures and one transfer enclosure. **D)** Comparison of learning curves of spatial memory pipeline (red) and RL baseline agents (black). Solid, mean reward of 25 agents training in 3 environments. Dashed, mean reward of one transfer attempt for each agent to 5 transfer environments. Dotted line indicates human performance. Surrounding light area covers one SE of the mean across the 25 original training agents (see Methods). **E)** Artificial pre-play in a T-maze. From left to right: schematic of artificial trajectory; place-cell activations; HD-cell activations; and value prediction along the trajectory. Dashed line marks the time when the trajectory crosses from the trunk to the left arm of the T. Place cells are sorted along the horizontal axis in three groups according to the position of their centroid of activity: along the stem, the left arm and the right arm of the T (see Methods). HD-cells are sorted according to preferred angle. Visual inputs were provided only for the first 20 steps of the trajectory. **F)** Same as E with trajectory turning right instead of left.

Data Fig 12). To test our claim that these allocentric representations provide efficient bases for transfer, the weights of the RNNs were frozen and the trained agent was placed into a fourth, visually distinct 4-room enclosure (Fig 4C). The agent was able to quickly transfer the learned behaviour (Fig 4D). In contrast, if generic recurrent networks were used instead of the Spatial Memory Pipeline, the agent was still able to learn the initial task, but failed to transfer to visually novel environments (Fig 4D, Extended Data Fig 11). Thus, our RNNs spatial responses support rapid generalisation to novel settings, an ability that is commonly observed in mammals but has been an elusive target for artificial agents.

Artificial replay During rest and pauses in behaviour, spatially modulated neurons – including place cells and head-direction cells – can decouple their activity from an animal's self-location and recapitulate trajectories through an environment[39]. Replay is transient but rapid and has been proposed as a mechanism relevant to navigational planning, imagination, and system-level consolidation[39–41]. To understand if our model was able to generate replay-like sequences we trained the same agent in a virtual T-maze. After training, starting from the base of the T, we let the agent run without visual reactivations (save for the first few steps of each trajectory, necessary for initial localisation). Despite the absence of perceptual input, we found that sequences of activity in the RNNs and memory slots were coherently reactivated, strongly resembling spatial trajectories along the track. Distinct sets of place cells were active for right and left turns (Fig 4E-F), and distinct sequences of visual memory slots were predicted. These sequences can be utilised to *imagine*[42] the future course of action –effectively generating predictions regarding future experiences (see Methods, Movie S1 and S2).

Discussion The Spatial Memory Pipeline described by our model develops an array of sensory-derived allocentric spatial representations strongly resembling those found in the mammalian brain. These representations are learnt solely using a predictive coding principle from egocentric visual perception and self-motion inputs - the network does not have access to allocentric information. Thus the model differs markedly from prior work in which brain-inspired agents have been limited by the need for allocentric input during training[43–46]. Equally, being trained 'end to end' it contrasts with simultaneous localisation and mapping approaches which use handcrafted algorithms to extract pose estimates from select visual key-frames[47,48]. The forms of egocentric representation used here are, however, known to contribute to the activity of spatially modulated neurons in the hippocampal region[49,50] while the predictive framework agrees with empirical and theoretical work linking the hippocampus with anticipation of the future[7,51,52]. Our core contribution then is the provision of a normative model linking most of the known mammalian spatial representations to a simple objective function based on perceptual experience. This stands in marked contrast to recent studies that have focused on the derivation of place cells from idealised grid cells and vice versa[43,44,53,54] - to this existing work we add an understanding of how place cells result from interactions with the sensory world.

The spatial cell-types that we identified were segregated between different modules, being defined by the forms of information available to each of them: angular velocity and visual corrections for head-direction cells, angular velocity, speed, and visual corrections for egoBVCs, and only visual corrections for BVCs. Reactivation of memories by BVCs resembled place cell activity. This framework implicitly captures several properties of the mammalian spatial system - distinct functional populations combining to form a sparse, spatially precise representation of self-location similar to that seen in the hippocampus. Like their biological counterparts, spatial

responses in the RNNs were robust to environmental manipulations, responding predictably to the changes made to local cues and retaining their fundamental firing correlates between entirely different enclosures. In contrast, activity in the memory stores was, by design, entirely distinct between environments, emulating hippocampal remapping, a phenomenon that occurs in response to significant manipulations of environmental cues[55,56].

In the absence of visual input, activity in the RNNs and the place cell-like memory slots were reactivated solely on the basis of the network's internal dynamics, resembling the sweeps of spatial activity that occur during replay. These representations constitute a flexible and efficient spatial basis, available to be quickly redeployed in a novel setting - sufficient to support rapid transfer learning in a reinforcement learning agent.

In the case of head-direction cells in the first RNN, the activity of units resulted from learnt attractor dynamics – both in the integration of velocities and in the anchoring to visual cues. Low dimensional manifolds of this form are widely accepted to provide the preeminent account of the biological head-direction system[36,57], even being instantiated topographically in flies[58,59].

In conclusion, we show that an artificial network replicates both the form and function of a biological network central to self-localisation and navigation. A simple training objective is sufficient, in this case, to approximate the development of neural circuits that have been shaped by both selective pressure applied over evolutionary time as well as by direct experience during an animal's life.

1. Tolman, E. C. Cognitive maps in rats and men. *Psychological review* **55**, 189 (1948).

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

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

4. Taube, J. S., Muller, R. U. & Ranck, J. B. Head-direction cells recorded from the postsubiculum in freely moving rats. i. description and quantitative analysis. *Journal of Neuroscience* **10**, 420–435 (1990).

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

6. Lever, C., Burton, S., Jeewajee, A., O'Keefe, J. & Burgess, N. Boundary vector cells in the subiculum of the hippocampal formation. *Journal of Neuroscience* **29**, 9771–9777 (2009).

7. Stachenfeld, K. L., Botvinick, M. M. & Gershman, S. J. The hippocampus as a predictive map. *Nature neuroscience* **20**, 1643 (2017).

8. Schiller, D. *et al.* Memory and space: towards an understanding of the cognitive map. *Journal of Neuroscience* **35**, 13904–13911 (2015).

9. Moser, M.-B., Rowland, D. C. & Moser, E. I. Place cells, grid cells, and memory. *Cold Spring Harbor perspectives in biology* **7**, a021808 (2015).

10. Zipser, D. A computational model of hippocampal place fields. *Behavioral neuroscience* **99**, 1006 (1985).

11. Sharp, P. E. Computer simulation of hippocampal place cells. *Psychobiology* **19**, 103–115 (1991).

12. Fuhs, M. C. & Touretzky, D. S. Synaptic learning models of map separation in the hippocampus. *Neurocomputing* **32**, 379–384 (2000).

13. Franzius, M., Sprekeler, H. & Wiskott, L. Slowness and sparseness lead to place, head-direction, and spatial-view cells. *PLoS computational biology* **3**, e166 (2007).

14. Byrne, P., Becker, S. & Burgess, N. Remembering the past and imagining the future: a neural model of spatial memory and imagery. *Psychological review* **114**, 340 (2007).

15. Wang, C. *et al.* Egocentric coding of external items in the lateral entorhinal cortex. *Science* **362**, 945–949 (2018).

16. Hinman, J. R., Chapman, G. W. & Hasselmo, M. E. Neuronal representation of environmental boundaries in egocentric coordinates. *Nature Communications* **10**, 2772 (2019).

17. Alexander, A. S. *et al.* Egocentric boundary vector tuning of the retrosplenial cortex. *Science Advances* **6**, eaaz2322 (2020).

18. Sutton, R. S. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, 216–224 (Elsevier, 1990).

19. Eichenbaum, H. Hippocampus: cognitive processes and neural representations that underlie declarative memory. *Neuron* **44**, 109–120 (2004).

20. Kumaran, D. & Maguire, E. A. An unexpected sequence of events: mismatch detection in the human hippocampus. *PLoS biology* **4**, e424 (2006).

21. Lisman, J. & Redish, A. D. Prediction, sequences and the hippocampus. *Philosophical Transactions of the Royal Society B: Biological Sciences* **364**, 1193–1201 (2009).

22. Rao, R. P. & Ballard, D. H. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience* **2**, 79 (1999).

23. Friston, K. & Kiebel, S. Predictive coding under the free-energy principle. *Philosophical Transactions of the Royal Society B: Biological Sciences* **364**, 1211–1221 (2009).

24. Raudies, F. & Hasselmo, M. E. Modeling boundary vector cell firing given optic flow as a cue. *PLoS computational biology* **8**, e1002553 (2012).

25. Ranck, J. B. Head direction cells in the deep cell layer of dorsolateral pre-subiculum in freely moving rats. *Electrical activity of the archicortex* (1985).

26. Bicanski, A. & Burgess, N. A neural-level model of spatial memory and imagery. *ELife* **7**, e33752 (2018).

27. Wallace, D. J. *et al.* Rats maintain an overhead binocular field at the expense of constant fusion. *Nature* **498**, 65–69 (2013).

28. Hartley, T., Burgess, N., Lever, C., Cacucci, F. & O'Keefe, J. Modeling place fields in terms of the cortical inputs to the hippocampus. *Hippocampus* **10**, 369–379 (2000).

29. Barry, C. *et al.* The boundary vector cell model of place cell firing and spatial memory. *Reviews in the Neurosciences* **17**, 71–98 (2006).

30. O'Keefe, J. & Burgess, N. Geometric determinants of the place fields of hippocampal neurons. *Nature* **381**, 425 (1996).

31. Barry, C., Hayman, R., Burgess, N. & Jeffery, K. J. Experience-dependent rescaling of entorhinal grids. *Nature neuroscience* **10**, 682 (2007).

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

33. Seelig, J. D. & Jayaraman, V. Neural dynamics for landmark orientation and angular path integration. *Nature* **521**, 186–191 (2015).

34. Cueva, C. J., Wang, P. Y., Chin, M. & Wei, X.-X. Emergence of functional and structural properties of the head direction system by optimization of recurrent neural networks. *arXiv preprint arXiv:1912.10189* (2019).

35. Poulter, S., Lee, S. A., Dachtler, J., Wills, T. J. & Lever, C. Vector trace cells in the subiculum of the hippocampal formation. *bioRxiv* 805242 (2019).

36. Redish, A. D., Elga, A. N. & Touretzky, D. S. A coupled attractor model of the rodent head direction system. *Network: Computation in Neural Systems* **7**, 671–685 (1996).

37. Espeholt, L. *et al.* Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures (2018). 1802.01561.

38. Morris, R. G. Spatial localization does not require the presence of local cues. *Learning and motivation* **12**, 239–260 (1981).

39. Foster, D. J. & Wilson, M. A. Reverse replay of behavioural sequences in hippocampal place cells during the awake state. *Nature* **440**, 680–683 (2006).

40. Lee, A. K. & Wilson, M. A. Memory of sequential experience in the hippocampus during slow wave sleep. *Neuron* **36**, 1183–1194 (2002).

41. Pfeiffer, B. E. & Foster, D. J. Hippocampal place-cell sequences depict future paths to remembered goals. *Nature* **497**, 74 (2013).

42. O'Craven, K. M. & Kanwisher, N. Mental imagery of faces and places activates corresponding stimulus-specific brain regions. *Journal of cognitive neuroscience* **12**, 1013–1023 (2000).

43. Banino, A. *et al.* Vector-based navigation using grid-like representations in artificial agents. *Nature* **557**, 429 (2018).

44. Cueva, C. J. & Wei, X.-X. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. *arXiv preprint arXiv:1803.07770* (2018).

45. Bicanski, A. & Burgess, N. A neural-level model of spatial memory and imagery. *ELife* **7**, e33752 (2018).

46. Whittington, J., Muller, T., Mark, S., Barry, C. & Behrens, T. Generalisation of structural knowledge in the hippocampal-entorhinal system. In *Advances in neural information processing systems*, 8484–8495 (2018).

47. Klein, G. & Murray, D. Parallel tracking and mapping for small ar workspaces. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, 1–10 (IEEE Computer Society, 2007).

48. Engel, J., Schöps, T. & Cremers, D. LSD-SLAM: Large-scale direct monocular SLAM. In *European conference on computer vision*, 834–849 (Springer, 2014).

49. Muller, R. U. & Kubie, J. L. The effects of changes in the environment on the spatial firing of hippocampal complex-spike cells. *Journal of Neuroscience* **7**, 1951–1968 (1987).

50. Gothard, K. M., Skaggs, W. E., Moore, K. M. & McNaughton, B. L. Binding of hippocampal ca1 neural activity to multiple reference frames in a landmark-based navigation task. *Journal of Neuroscience* **16**, 823–835 (1996).

51. Blum, K. I. & Abbott, L. A model of spatial map formation in the hippocampus of the rat. *Neural computation* **8**, 85–93 (1996).

52. Hassabis, D. & Maguire, E. A. The construction system of the brain. *Philosophical Transactions of the Royal Society B: Biological Sciences* **364**, 1263–1271 (2009).

53. Solstad, T., Moser, E. I. & Einevoll, G. T. From grid cells to place cells: a mathematical model. *Hippocampus* **16**, 1026–1031 (2006).

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

55. Hayman, R. M., Chakraborty, S., Anderson, M. I. & Jeffery, K. J. Context-specific acquisition of location discrimination by hippocampal place cells. *European Journal of Neuroscience* **18**, 2825–2834 (2003).

56. Leutgeb, S. *et al.* Independent codes for spatial and episodic memory in hippocampal neuronal ensembles. *Science* **309**, 619–623 (2005).

57. Sharp, P. E., Blair, H. T. & Cho, J. The anatomical and computational basis of the rat head-direction cell signal. *Trends in neurosciences* **24**, 289–294 (2001).

58. Kim, S. S., Rouault, H., Druckmann, S. & Jayaraman, V. Ring attractor dynamics in the drosophila central brain. *Science* **356**, 849–853 (2017).

59. Fisher, Y. E., Lu, J., DAlessandro, I. & Wilson, R. I. Sensorimotor experience remaps visual input to a heading-direction network. *Nature* **576**, 121–125 (2019).

Methods

The Spatial Memory Pipeline The *Spatial Memory Pipeline* consists of two submodules where learning occurs independently. The first level is a visual feature extraction network that reduces the dimensionality of visual inputs. Following visual feature extraction, there is a level of integration through time that encodes estimates of the agent's location by integrating egocentric velocities. We refer to this integration level as *Memory-Map*. It is composed of a fast-binding slot-based associative memory that plays the role of an idealised hippocampus, and a set of recurrent neural networks that receive as inputs egocentric velocity signals and play the role of an artificial neocortex.

Visual-feature extraction The first level in the *Spatial Memory Pipeline* hierarchy employs a convolutional autoencoder[60] to reduce the dimensionality of the raw visual input (Extended Data Fig 1A). The autoencoder is composed of an encoder network that transforms the raw visual input, $\mathbf{y}^{(raw)}$ into a low-dimensional visual encoding vector, $\mathbf{y}^{(enc)}$, that summarises the structure of the input. To learn such compressed code, $\mathbf{y}^{(enc)}$ is passed trough a decoder that produces a reconstruction of the original input, $\hat{\mathbf{y}}^{(raw)}$. All of the encoder and decoder parameters are optimised to minimise the reconstruction error $|\mathbf{y}^{(raw)} - \hat{\mathbf{y}}^{(raw)}|^2$. Table 3 summarises the autoencoder configuration, see Supplementary Methods for more details.

For most of our experiments only the encoder network was used to transform raw images into visual encodings serving as inputs to a Memory-map downstream. However, in the replay experiments the decoder network was also used to recreate images from memorised encodings.

Memory-maps A Memory-map module consists of two components: a set of R recurrent neural networks, $\{F_r\}_{r\in 1\ldots R}$, that integrate egocentric velocities, and a slot-based associative memory, $\mathcal{M}$, that binds upstream inputs, $\mathbf{y}$, to RNN state values:

$$\mathcal{M} \equiv \left\{ (\mathbf{m}_s^{(\mathbf{y})}, \mathbf{m}_{1,s}^{(\mathbf{x})} \ldots \mathbf{m}_{R,s}^{(\mathbf{x})}) \right\}_{s\in 1\ldots S} , \tag{1}$$

where the variable s indexes the different memory slots, while the super-index denotes the type of information ($\mathbf{y}$ for upstream inputs, and $\mathbf{x}$ for the state of RNNs).

At each time step, t, the current upstream input, $\mathbf{y}_t$, reactivates the memory slots with the most similar contents (Extended Data Fig 1B). We formalise this concept using a categorical distribution over memory slots:

$$P_{react}(s \mid \mathbf{y}_t, \mathcal{M}) \propto \mathrm{e}^{\beta \mathbf{y}_t^\top \mathbf{m}_s^{(\mathbf{y})}} , \tag{2}$$

where β is a positive scalar that is automatically adjusted to match an entropy target, H_{react} –enforcing sparse activity (see Supplementary Methods).

At every time step, the states corresponding to each of the predictive RNNs, $\{\mathbf{x}_r\}_{r\in 1\ldots R}$ where $\mathbf{x}_r \in \mathbb{R}^{N_r}$, are updated using the current egocentric velocities, $\mathbf{v}_{r,t}$:

$$\hat{\mathbf{x}}_{r,t} = F_r\left(\mathbf{x}_{r,t-1}, \mathbf{v}_{r,t}\right) . \tag{3}$$

The type of velocity inputs used for each RNN is reported in Tables 4 and 5. For instance, in most of our experiments the first RNN takes as input the $\sin$ and $\cos$ of the angular velocity, ω,

whereas the second RNN takes the same inputs as the first RNN and also the linear speed, s. Finally, the third RNN receives no velocity inputs and relies only of temporal correlations.

These RNN states also induce a predictive categorical distribution over memory slots (Extended Data Fig 1B):

$$P_{pred}(s \mid \mathcal{M}, \hat{\mathbf{x}}_{1,t} \dots \hat{\mathbf{x}}_{R,t}) \propto \prod_{r=1}^{R} \mathrm{e}^{\pi_r \hat{\mathbf{x}}_{r,t}^{\top} \mathbf{m}_{r,s}^{(\mathbf{x})}} , \tag{4}$$

where π_r are positive scalar parameters that determine the relative importance of each RNN in the predictive distribution and its entropy.

Model parameters are optimised to minimise the cross-entropy from this predictive distribution to the distribution of visually driven reactivations:

$$L = \sum_{s=1}^{S} P_{react}(s \mid \mathbf{y}_t, \mathcal{M}) \log P_{pred}(s \mid \mathcal{M}, \hat{\mathbf{x}}_{1,t} \dots \hat{\mathbf{x}}_{R,t}) . \tag{5}$$

Thus, the model is trained so P_{pred} anticipates the distribution of memory reactivations P_{react}.

Note that at time t, P_{pred} has not yet received any information from the current upstream visual input, $\mathbf{y}_t$, forcing the RNNs to use previous egocentric velocity inputs to produce good predictions.

However, in order for the RNN representations to be allocentrically grounded, and to correct for the accumulation of integration errors, the RNNs must incorporate positional and directional information from upstream visual inputs as well. This correction step should not be performed at every time step, or the integration of velocities would be unnecessary; in our experiments, it was performed at random timesteps with probability $P_{correction} = 0.1$. The incorporation of visual information is implemented by calculating a correction code for each RNN:

$$\tilde{\mathbf{x}}_{r,t} = \sum_{s=1}^{S} w_{s,t} \mathbf{m}_{r,s}^{(\mathbf{x})} \text{ , where } \quad w_{s,t} = \frac{\mathrm{e}^{\gamma \mathbf{y}_t^{\top} \mathbf{m}_s^{(\mathbf{y})}}}{\sum_{s'=1}^{S} \mathrm{e}^{\gamma \mathbf{y}_t^{\top} \mathbf{m}_{s'}^{(\mathbf{y})}}} , \tag{6}$$

where γ is a positive scalar parameter that determines the entropy of the distribution of weights (Extended Data Fig 1C). Each $\tilde{\mathbf{x}}_{r,t}$ can be thought of as the result of a weighted reactivation of the RNN memory embeddings by the current visual input, $\mathbf{y}_t$.

In steps when corrections are provided, these correction codes are input to correction RNN cells, G_r, that combine the predictive state with the correction code (Extended Data Fig 1C):

$$\mathbf{x}_{r,t} = G_r \left(\hat{\mathbf{x}}_{r,t}, \tilde{\mathbf{x}}_{r,t} \right) . \tag{7}$$

While in steps when no correction is available, the input to the next time step is simply the predicted state (Extended Data Fig 1D):

$$\mathbf{x}_{r,t} = \hat{\mathbf{x}}_{r,t} . \tag{8}$$

Model parameters $\mathbf{\Theta} = \left\{ \gamma, \left\{ F_r, G_r, \mathbf{m}_{r,\cdot}^{(\mathbf{x})}, \pi_r \right\}_{r \in 1 \dots R} \right\}$ are trained by gradient descent on the prediction loss of equation (5).

Note that the contents of the memory store corresponding to the upstream inputs, $\mathbf{m}_{\cdot}^{(\mathbf{y})}$, are not part of the optimised parameters, $\mathbf{\Theta}$, as they are used to calculate the target distribution. Therefore, in order to fill the slots of $\mathcal{M}$ with memories of upstream inputs, at every timestep with small probability, $P_{storage}$, a slot s is chosen at random and assigned $(\mathbf{m}_s^{(\mathbf{y})}, \mathbf{m}_{1,s}^{(\mathbf{x})} \dots \mathbf{m}_{R,s}^{(\mathbf{x})}) := (\mathbf{y}_t, \mathbf{x}_{1,t} \dots \mathbf{x}_{R,t})$.

Due to the sparse activation of memory slots there is low interference between memories, i.e. modifying a $\mathbf{m}_{r,\cdot}^{(\mathbf{x})}$ only has a local effect. For this reason these associative memory embeddings can be optimised at a higher learning rate (see Table 2), resulting in fast binding of new memories once the RNN dynamics have been learned.

Further integration levels Extra downstream levels of temporal integration are possible. For the reinforcement learning experiments we added a second Memory-Map whose memories are reactivated by an RNN in the first level. As we saw in the main text, the activity of these memories resembles hippocampal place cells. Therefore, we expect the representations learned by RNN in this second Memory-Map to be useful for flexible navigation strategies[43].

Architecture used in our experiments The experimental results in this paper used the architecture described in Tables 3-5.

Visual autoencoders The visual-encoding vectors $\mathbf{y}^{(enc)}$ were obtained by inputting RGB images, $\mathbf{y}^{(raw)}$, of the environment to an encoder network with 4 convolutional layers with bias and ReLU output nonlinearities, chained to a flat output layer (Extended Data Fig 1A). The architecture parameters are summarised in Table 3. The encoders used for the reinforcement learning experiments had greater capacity than the encoders used for the unsupervised-learning experiments, since they were trained to encode a variety of environments, as explained below.

In our experiments, the visual autoencoders were not affected by the prediction loss in equation (5). Therefore, the autoencoders were trained beforehand, separately from the rest of the model. During the autoencoder training, images were passed through the convolutional layers and flat output layer to produce the vector of visual encodings, $\mathbf{y}^{(enc)}$. This vector was then passed through a decoder network of de-convolutional layers (with transposed architecture from the encoder but independent parameters) to produce a reconstruction, $\hat{\mathbf{y}}^{(raw)}$, of the input image (Extended Data Fig 1A). All parameters in the autoencoder were optimised to minimise the mean square distance between the input and reconstructed images.

For all experiments, the autoencoders were trained by minibatch gradient descent using an Adam optimiser with learning rate 10^{-4}.

For unsupervised experiments, training minibatches consisted of 50 images taken at random from the same trajectories used to train the spatial memory pipeline. Training was complete after 200,000 minibatches. We trained a separate autoencoder for each environment (square, circular, and trapezoid cages). Note that the manipulated environments in Extended Data Fig 5, Extended Data Fig 7, and Extended Data Fig 6, used the same autoencoder as their original, non-manipulated square environment.

For reinforcement learning experiments the training minibatches consisted of 216 images

mixed at random from trajectories generated in the training and testing environments using a rat-like motion model (see below). Training was complete after 200,000 minibatches.

The replay experiments used the same autoencoder architecture and training procedure as the reinforcement learning experiments, except minibatches had only 64 images and training took 125,000 minibatches.

Rat-like motion model The rat-like motion model used to generate trajectories for unsupervised experiments (and for training the visual autoencoders for reinforcement learning and replay experiments, see above) was based on published work[24], with parameters specified in Table 1. At each time step a linear velocity was generated from a Rayleigh distribution and a rotational velocity from a Gaussian distribution. However, if the trajectory was closer than 3 cm to a wall at an angle narrower than 90^o, a deterministic rotation was added that turned it to continue to move parallel to the wall, and the linear velocity was reduced by a factor of 4. Additionally, the trajectory switched between periods of movement (linear velocity Rayleigh parameter 0.13) and periods of pure rotation (linear velocity Rayleigh parameter 0.0). The move and stop periods lasted for an exponentially distributed number of steps with mean 50. The trajectories used to train the spatial memory pipeline were sampled every 7 simulation steps. The angular and linear velocity signals fed to the RNNs in unsupervised experiments were computed dividing the total spatial displacement and angular rotation of the agent over each 7-simulation-step interval by the simulation time of those 7 steps. The corresponding visual embedding came from the image of the environment at the given position and orientation, with the camera 20 cm above and parallel to the ground. Each trajectory consisted of 500 samples, corresponding to 3500 simulation steps.

Environment dimensions For the single-environment unsupervised-learning experiments a square enclosure 2.2 m in side was used. The multi-environment experiments, additionally, included a circular enclosure of radius 1.5 m, and an isosceles trapezoidal enclosure with altitude 4.4 m and parallel sides 2.2 m and 0.75 m long. Single-environment experiments were also carried out in the circular enclosure of 1.5 m radius (see Supplementary Results). The height of the walls in all environments was 0.25 m.

Sigmoid-LSTM and Sigmoid-Vanilla As our aim was to compare the activation of artificial RNNs to firing rate data from biological neurons, we limited the activations of all RNNs to positive values. In order to do this, we modified the commonly used *LSTM* and *Vanilla-RNN* cells by substituting their *tanh* output non-linearity for a *sigmoid*. We called these cells *Sigmoid-LSTM* and *Sigmoid-Vanilla-RNN* respectively. As it learns faster, we used *Sigmoid-LSTM* in all our experiments; with the exception of the supplementary HD double ring analysis (see Methods), where we used a *Sigmoid-Vanilla-RNN* for ease of analysis. In a Vanilla-RNN the mapping from the current state to the next is extremely simple. Namely, the current vector of cell activations, $\mathbf{h}_t$, depends on the previous activations, $\mathbf{h}_{t-1}$, and inputs, $\mathbf{x}_t$, through equation:

$$\mathbf{h}_t = \sigma\left(\mathbf{W}\mathbf{h}_{t-1} + \mathbf{V}\mathbf{x}_t + \mathbf{b}\right) , \tag{9}$$

where σ is the element-wise sigmoid function, and we refer to $\mathbf{W}$ as the weight matrix of dynamics and $\mathbf{V}$ as the weight matrix of inputs.

Sparse reactivation The inverse-temperature parameter in the reactivation of memory slots, β in equation (2), is constantly regulated so the entropy of P_{react} matches the hyperparameter H_{react} (0.5 nats in unsupervised-learning experiments, 1.0 nats in reinforcement learning and replay). In order to do this, β was parameterised as $\beta = \mathrm{e}^{\beta_{logit}}$ and after every trajectory, β_{logit} was increased/decreased by 0.001 when the average entropy of P_{react} was lower/higher than H_{react}.

Training details The training parameters for the unsupervised experiments are summarised in Table 2. The training was implemented with the TensorFlow platform. We used back-propagation through time (BPTT) with an unroll length of 50 trajectory steps; since trajectories consisted of 500 steps, each trajectory produced 10 BPTT unrolls. Batch size was 32, each batch consisting of unrolls from different trajectories.

The single-environment experiments used the Adam optimisation algorithm. Different learning rates were used for the memory slot contents and the rest of the network parameters: since the target distribution over slots was very sparse, we could apply a larger learning rate for the memory slot contents.

For reasons of ease of implementation, the multi-environment experiments used the RMSProp optimisation algorithm. Each of the three environments trained in a separate process, updating the shared parameters synchronously with Distributed TensorFlow. The shared parameters were the weights and biases of the RNNs (both the F_r prediction RNNs and the G_r correction RNNs). The memory contents $\mathcal{M}$ and the entropy-scaling variables γ and π_r were separate for each environment.

Similarly to previous work[43], dropout was used in the RNN outputs when predicting the memory reactivations.

Assessment of attractor dynamics in the head-direction system For the evaluation of integration dynamics in Fig 3A the state of RNN-1 was initialised to $\mathbf{m}_{1,1}^{(\mathbf{x})}$, and angular integration (equation 3) iterated for a thousand steps using the angular velocity $\frac{\pi}{20}\ \mathrm{rad/step}$ for 250 steps, $0\ \mathrm{rad/step}$ for the following 250 steps, and $-\frac{\pi}{40}\ \mathrm{rad/step}$ for the final 500 steps. Visual input was not given to the model. Fig 3A shows the evolution of the RNN state, $\mathbf{x}_{1,t}$, ordering its units by the phase of its resultant vector.

The principal-component space displayed in Fig 3B was calculated by taking the two principal eigenvectors of the covariance matrix of unit activity in Fig 3A. Repeated activity vectors were discarded from the calculation of the mean and covariance matrix.

For the assessment of fixed attractor points in Fig 3C-D, the state of each unit in RNN-1 was independently initialised by drawing a sample from a Gaussian with four times the standard deviation of its activations in Fig 3A. A thousand different random initialisations were used and each run for 1000 steps of angular integration (equation 3) inputting $0\ \mathrm{rad/step}$ angular velocity.

For the assessment of periodic orbits in Fig 3E-F, the same one thousand initialisations were used, but each was run for 200 steps of angular integration (equation 3) using either a clockwise $-\frac{\pi}{20}\ \mathrm{rad/step}$ or counter-clockwise $\frac{\pi}{20}\ \mathrm{rad/step}$ angular velocity. The 50 first steps of two arbitrary trajectories with opposing velocities were shown in Fig 3E. Each of the two autocovariance

curves in Fig 3F was calculated using all 1000 trajectories.

For the assessment of point-attractors under the influence of visual inputs in Fig 3G-I, we took 512 random images from the training dataset, and run the model for 200 steps of prediction and correction. Each of these 200 steps consisted of a step of visual correction, equations 7 and 8, followed by zero angular-velocity integration, equation 3. For every image we run 18 trajectories initialised from each the fixed points found in Fig 3D. We considered that a single fixed-point had been reached at a particular time step if the maximum distance among the 18 trajectories was less than 0.1 in the PCA space. The bottom plots in Fig 3H-I show the trajectory of each of these 18 initialisations for thre examples of visual inputs.

Unsupervised learning in multiple environments We trained the Spatial Memory Pipeline simultaneously in three environments, sharing the RNN parameters across the environments but keeping a separate memory store for each. The separation of memories simulates hippocampal remapping, since there is no correspondence between the slots for the different environments. The RNN units, on the other hand, behaved coherently across the environments, giving further credence to their role as analogues of HD, egoBVC and BVC cells. Extended Data Fig 9 summarises the stability of these representations across the three environments (1, square; 2, circle; 3, trapezoid). The relative preferred angles between HD units in RNN-1 were preserved with great accuracy between environments (Fig 9A), i.e., the head direction system rotated as a whole without change in functionality. In contrast, the relative directional tuning of BVC units in RNN-3, although not entirely random (Kuiper test rejected the uniform distribution for the angle differences, p-value< 0.01 for all environment pairs), displayed high variability across units (Fig 9C). This means that BVC units in our model, under remapping, are not locked to the head direction system, unlike the case without remapping (see rotation manipulations, Extended Data Fig 5). The preferred angles of EgoBVC units in RNN-2 stayed the same in different environments (inter-environment angular differences clustered tightly around zero, Fig 9B), as expected for a system of egocentric coordinates. Distance tuning of egoBVCs was very stable across environments (Fig 9D), while BVC distance tuning, although significantly preserved (mean across units of the absolute difference between environments significantly smaller than the mean across shuffled pairs, p-value$< 10^{-4}$), again showed more variability (Fig 9E).

The experiment described in Fig 4A and Extended Data Fig 9 was typical, but not all experiments resulted in the same distribution of representations across RNNs and environments. Specifically, across 15 multi-environment experiments (using 5 different random seeds and 3 different RNN learning rates, 10^{-3}, $3 \cdot 10^{-4}$, and 10^{-4}), 8 had HD cells purely in RNN-1, as in the experiment shown, while the other 7 had a significant number of them in RNN-2 instead.

Environment for reinforcement learning experiments We assessed the performance of reinforcement learning agents in the DeepMind Laboratory platform [61]. The layout of the 4-room enclosures (Fig3C) consisted of four 5 x 5-tile rooms (1.25 x 1.25 m assuming an agent speed of 15 cm/s) linked by four 5 x 1 corridors. The enclosure was surrounded by a sky-box at infinity - so as to provide directional but not distance information - textured with buildings, clouds and trees, which provided distal cues. There were no special markings on the walls or floors of the enclosure, making all rooms and corridors identical except for their relative orientation with respect to the distal cues. At the start of each episode the agent (described below) was placed in a random location and was required to explore in order to find an unmarked goal, paralleling the

task of rodents in the classic Morris water maze. When the goal was reached, the agent received a reward of 10 points and was teleported to a random location in the enclosure. The goal remained fixed for the length of the episode, so a well-trained agent would first explore the enclosure in order to discover the goal location and then, after each teleportation, orient itself to return to the same goal as quickly as possible. The length of the episodes was 3000 agent steps (12000 environment frames, see below).

The agent received observations in the form of camera input (96 x 72 pixels), rotation velocity, and egocentric translation velocity (components parallel and perpendicular to the agent's orientation).

The agent could start at any position in the enclosure. The action space was discrete (six actions) but afforded fine-grained motor control (that is, the agent could rotate in small increments, accelerate forwards or backwards, or effect rotational acceleration while moving forwards). Agent actions were repeated for 4 consecutive environment frames[37]; one actor step consisted therefore of 4 environment steps.

The visual appearance of the environment was determined by the floor, wall and distal cues. We used 8 different sets of textures, three of them for training and the other five to evaluate transfer.

Reinforcement learning agent architecture We used importance-weighted actor-learner agent (IMPALA) [37] to learn the task. IMPALA is an actor-critic setup to learn a policy distribution π over the available actions and a baseline function V^{π}. Our IMPALA setup consisted of 96 actors, repeatedly generating trajectories of experience, and one learner that used the trajectories sent by actors to learn the policy. At the beginning of each episode actors updated their local policy-network, value-network and Spatial Memory Pipeline parameters to the latest learner parameters and, at the end of each episode, sent the trajectory of observations, rewards, states, and policy distributions to the learner. The learner continuously updated its policy, value and Spatial Memory Pipeline parameters via back-propagation through time (BPTT) on batches of 64 100-step chunks of trajectories collected from different actors. Both the actor-critic parameters and the Spatial Memory Pipeline parameters were optimised simultaneously, but with respect to separate losses. The Spatial Memory Pipeline loss is just as described previously in the unsupervised experiments, and determines the gradients for the Spatial Memory Pipeline parameters. The actor-critic loss is composed of three terms: a policy gradient loss to maximise expected advantage, a baseline value loss to predict the episode returns, and an entropy bonus to prevent premature convergence. The possible policy lag between the actors and the learner was corrected in the policy gradient and baseline losses with V-trace[37]. We used the Adam[62], a stochastic gradient descent optimiser, for both the Spatial Memory Pipeline and the actor-critic loss. A schematic of the agent network is displayed in Extended Data Fig 11A.

The inputs to the actor-critic network were the previous time-step reward and one-hot encoded action, the current state of all the Spatial Memory Pipeline's RNNs (current allocentric code), and the the state of all the Spatial Memory Pipeline's RNNs observed last time the goal was reached (goal allocentric code, which was set to zero as long as the goal had not been reached in the episode). These inputs were fed to a 256-unit LSTM whose output in turn went through a policy MLP (one 256-unit hidden layer with ReLU activation) to produce the policy, and a value

MLP (one 256-unit hidden layer with ReLU activation) to predict the value. Note that the actor-critic network did not receive direct visual input to learn the task - only representations from the Spatial Memory Pipeline.

The inputs to the Spatial Memory Pipeline, similarly to the unsupervised learning experiments, consisted of egocentric velocities and vision. Because the DeepMind Laboratory agent has inertia, it could shift sideways while moving, even though the action set did not allow strafing. Therefore we provided as inputs not only the translation velocity in the heading direction of the agent, but also the component perpendicular to it. These, together with the sine and cosine of the rotational velocity of the agent, formed the 4-component velocity input vector (as opposed to the 3-component vector used in unsupervised experiments, that lacked a sideways translation component). The visual input was the embedding vector (length 128) of a convolutional autoencoder trained offline to reconstruct images of all the training and transfer environments, as explained previously. Compared to the unsupervised experiments, the Spatial Memory Pipeline for the reinforcement learning experiments used bigger RNN sizes, more memory slots and one extra level of temporal integration, where memories were reactivated by RNN-3. The parameters of the architecture are described in Tables 3-5.

In the training phase, each actor was assigned one of the 3 visually different training environments. The Spatial Memory Pipeline had correspondingly three sets of memory stores, and experiences coming from each environment were channelled automatically to the corresponding bank. The RNN parameters, in contrast, were shared among all the environments. The agent trained for a combined 2 billion actor steps, then it was tested on the transfer environments.

We evaluated the 5 transfer environments separately. In each transfer experiment all the IMPALA actors experienced only the transfer environment, and memory stores were initialised blank. The memory store contents were overwritten at a quicker pace (probability 0.01 per step) at the beginning of the transfer experiments, until full, then continued to be overwritten at the normal rate (probability 0.0001 per step). The rest of the agent parameters started from their values in the trained agent. During transfer the RNN parameters of the Spatial Memory Pipeline were frozen; only the memory bank contents and the policy network parameters were trained.

Reinforcement learning baselines To support the hypothesis that it is the Spatial Memory Pipeline representations that allow the agent to transfer the behaviour to a novel environment with little training, we repeated the transfer experiments replacing the Spatial Memory Pipeline with networks of comparable capacity: 1) a Spatial Memory Pipeline identical to the main experiment pipeline, except the visual correction was performed at every time step ($P_{correction} = 1.0$); this Spatial Memory Pipeline did not need to integrate velocities over multiple agent steps in order to predict the slot activations, and consequently did not develop any useful spatial representations. 2) A generic recurrent network (LSTM) (Extended Data Fig 11B) that integrated visual and velocity inputs, trained with the agent policy loss. And 3) a two-layer network (Extended Data Fig 11C) where the first layer consisted of three LSTMs, each integrating the visual and velocity inputs from one of the three training environments, and the second layer consisted of a single LSTM integrating the batched outputs of the first-layer LSTMs; this network paralleled the Spatial Memory Pipeline architecture, where each training environment had its own separate memory stores.

The three replacement networks, while able to learn the Morris water-maze task in the training environments, failed to transfer their learned behaviour to visually novel environments (Extended Data Fig 11D). After 50 million steps of transfer training in a novel environment, all the Spatial Memory Pipeline agents had reached human performance. In contrast, the agents with generic recurrent networks needed a similar number of training steps to transfer the learned policy as it took to learn it in the first place (>250 million steps). The Spatial Memory Pipeline that was purposefully trained with continuous vision ($P_{correction} = 1.0$) to disrupt the development of spatial representations took much longer, demonstrating that it is not an architectural bias, but the spatial representations learned that assist in the transfer of behaviour.

Replay experiments For artificial replay experiments we trained the same water maze task as in the reinforcement learning experiments (see above) in a T-shaped enclosure in the DeepMind Laboratory platform. The stem of the T was 3 x 12 tiles, the top line (consisting of the left and right arms) 25 x 3 tiles. The enclosure was surrounded by a sky-box at infinity textured with buildings. There was a marking at the end of each arm, different on the right and on the left. During training, the goal position and the agent spawning positions could be anywhere in the enclosure, so that the Spatial Memory Pipeline could learn to integrate trajectories across the whole area. The length of the episodes was 7200 environment steps (1800 agent steps). The agent architecture and hyperparameters were the same as in the reinforcement learning experiments. Visual reactivations were provided on average every 10 steps (uniform probability of 0.1 in every step), except at the beginning of each episode or after teleporting, when they were provided at every step for 10 steps. The agent was trained until performance plateaued (mean episode reward ~ 250, about 600 millions of steps of training).

The trained agent was tested on episodes where the agent's initial and teleporting positions were always at the bottom of the T-stem (agent orientation random), and the goals always at the end of either of the two arms. In the testing phase, visual reactivations were provided at every step for the first 20 steps of a trajectory, to allow the agent to localise. After 20 steps no visual reactivations were provided, and the agent followed the learned policy. The mean episode reward obtained by the agent in the test episodes was ~ 200.

For Fig 4E and F we chose one test episode with goal on the left arm and one with goal on the right arm. We display trajectories after the agent had reached the goal for the first time in the episode. For the the place-cell activity *vs.* time plots, place cell activity fields were computed from the training experiments as described elsewhere in Methods. Place cells were split into three blocks: first one corresponding to the vertical stem of the T-maze, the second to the left stem, and third to the right stem. Place cells in the first block were ordered by increasing y-coordinate (bottom to top) of their activity field centroids. The cells in the second block were ordered by decreasing (right to left) x-coordinate of their activity field centroids, and the third block by increasing x-coordinate (left to right) of their activity field centroids. During the testing episodes, without vision, place cell activation was computed from the predictions of the second integration level. All the place cells that were activated in either (left turn or right turn) trajectory were shown in both figures. In the head-direction cell plots, RNN-1 cells categorised as head-direction cells were sorted by their preferred direction of activation (see categorisation methods above), and their activation colour-mapped. Finally, the value plot represents the value output of the actor-critic agent.

The visual reconstructions on the left-hand side of Movie S1 and S2 are shown only for comparison: they were calculated simply as the reconstructions, $\hat{\mathbf{y}}^{(raw)}$, obtained by feeding ground-truth visual inputs, $\mathbf{y}^{(raw)}$, along the imagined trajectory into the autoencoder. The right-hand side reconstructions, on the other hand, were obtained by feeding into the decoder the content $\mathbf{m}_{s^*}^{(\mathbf{y})}$ of the visual memory slot corresponding to the highest predicted probability, where:

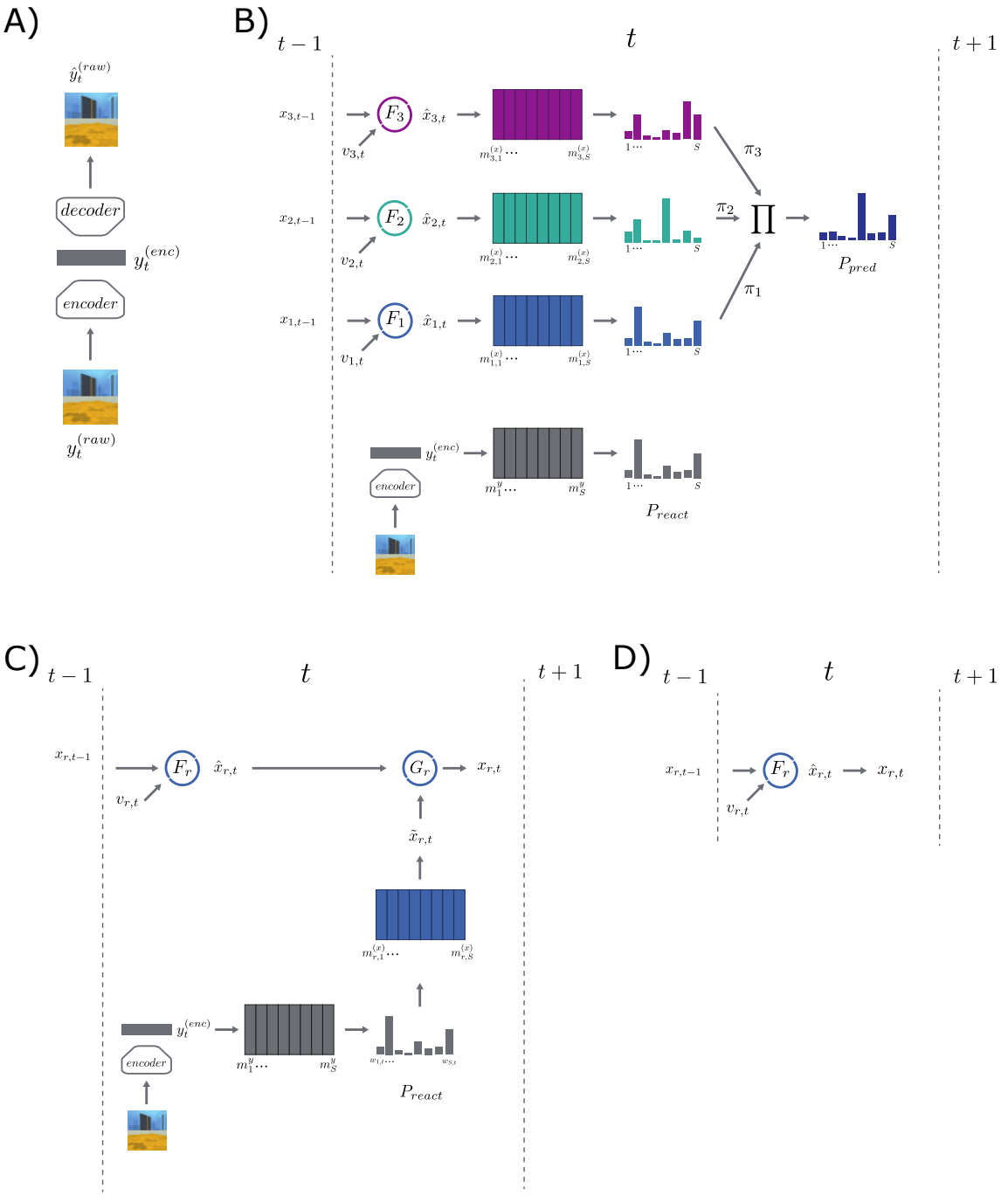
$$s^* = \arg\max_s P_{pred}(s \mid \mathcal{M}, \hat{\mathbf{x}}_{1,t} \ldots \hat{\mathbf{x}}_{R,t}), \tag{10}$$

and show the visual input predicted by the model.

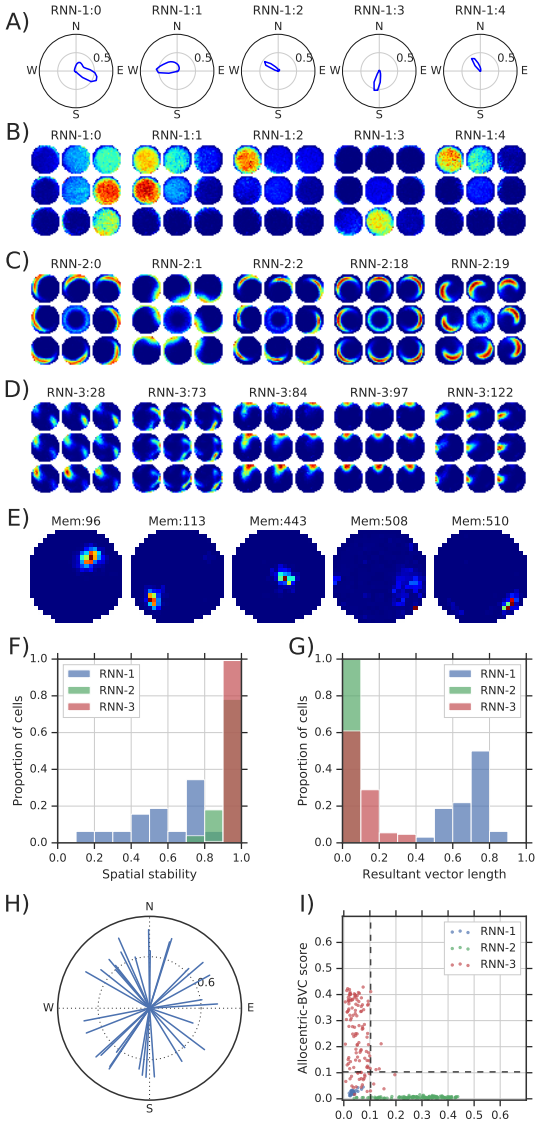
60. Kramer, M. A. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal* **37**, 233–243 (1991).

61. Beattie, C. *et al.* Deepmind lab (2016). 1612.03801.

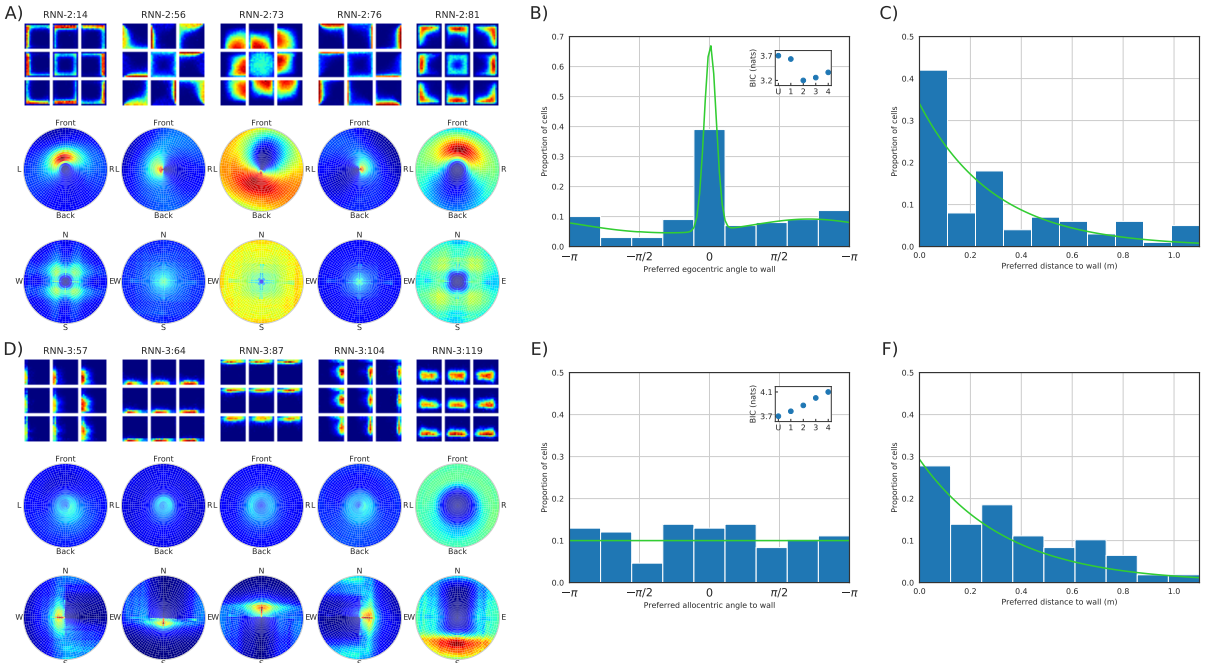
62. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).



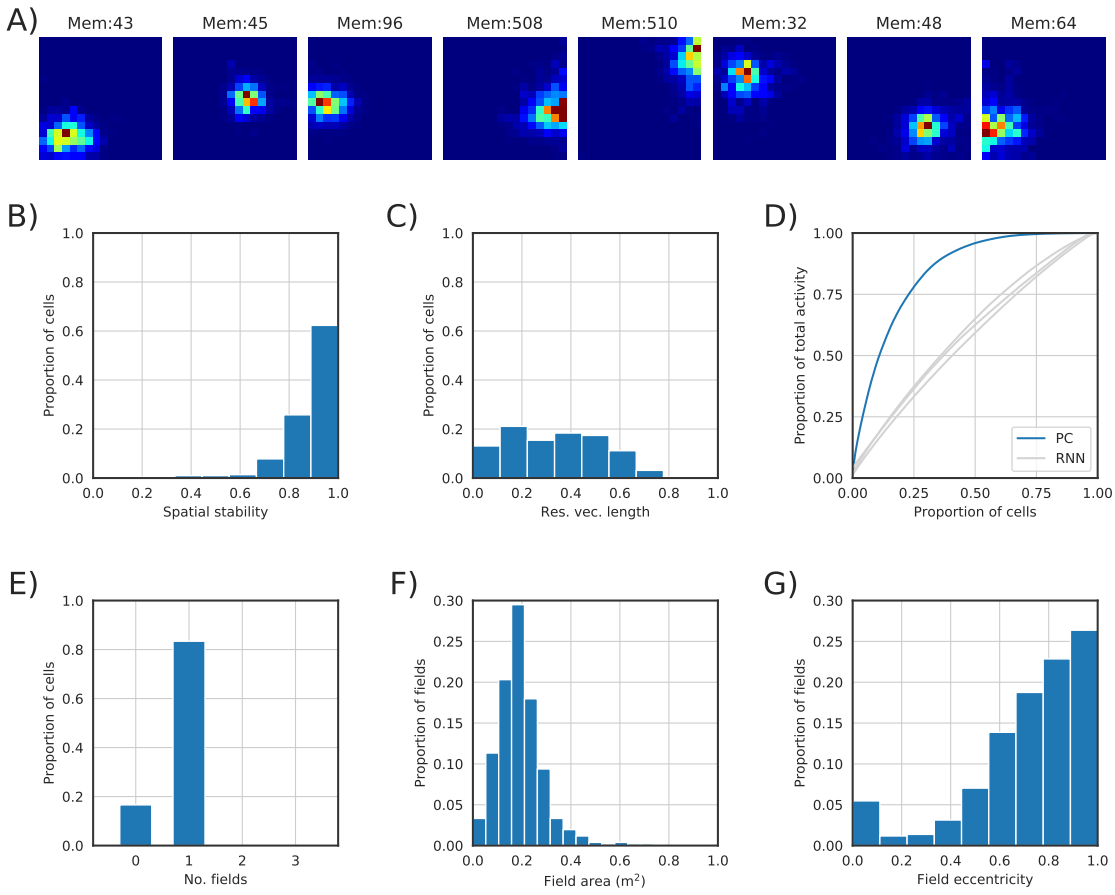
Extended Data Figure 1: Diagram of the spatial memory pipeline. **A)** Computational diagram showing the encoder and decoder networks of the convolutional autencoder used during its training. **B)** Computational diagram showing how the loss is calculated at every time step for an example model with three predictive RNNs. P_{react} is the target distribution and P_{pred} the predicted distribution over memories. **C)** Computational diagram of the model dynamics for a time step with visual correction (10% of steps). A correction code $\tilde{\mathbf{x}}$ is calculated as a visually-dependent weighted average of visual memory embeddings $\mathbf{m}^{(\mathbf{x})}_{r,\cdot}$. **D)** Computational diagram of the model dynamics for a time step without visual correction (90% of steps).



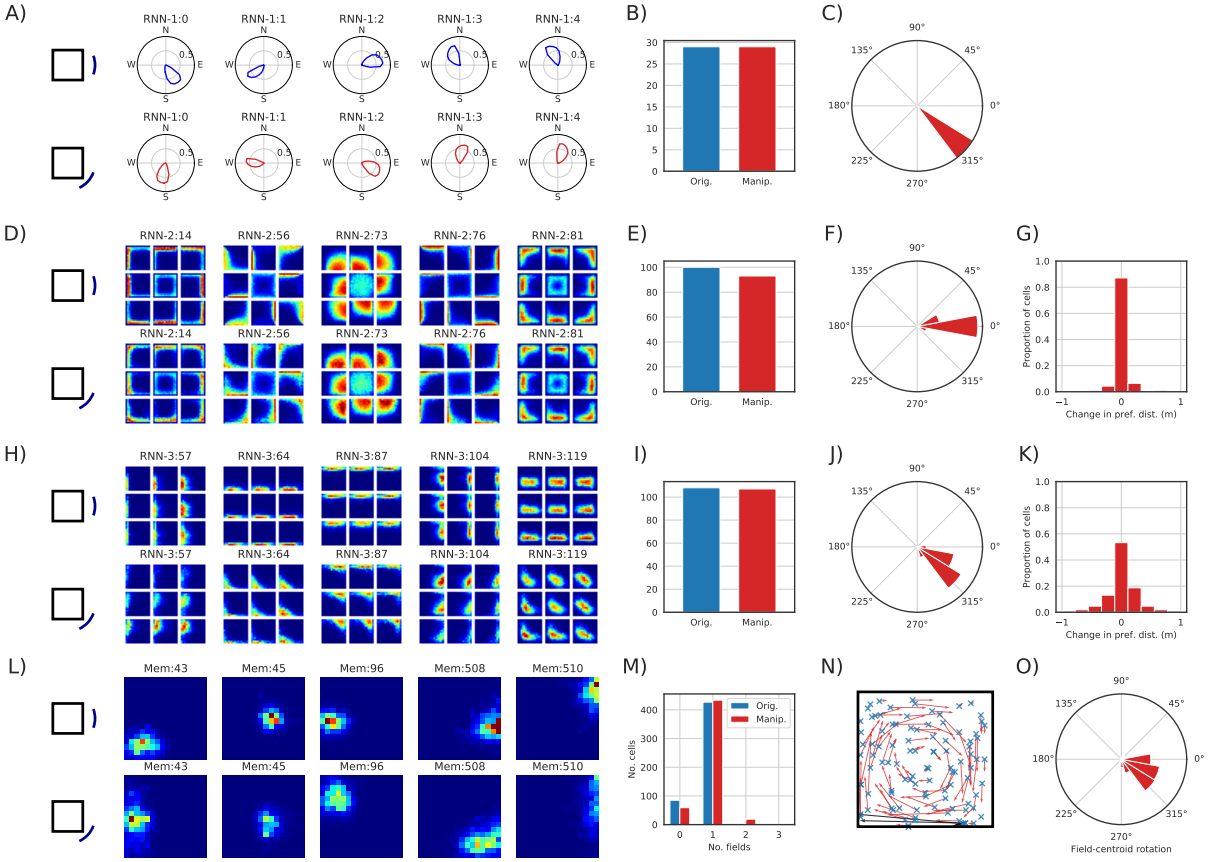
Extended Data Figure 2: Spatial representations in the spatial memory pipeline trained in a circular open field environment with white walls. **A)** Polar plot of activity by heading direction of five cells classified as HD cells. **B)** Spatial ratemaps of five cells shown in A. Central plot shows average activation for each location across all head directions. The eight plots around each central plot show location-specific activity restricted to times when the heading direction of the agent was in the corresponding 45° range (e.g. plot located above the central plot shows average activity when the agent is facing in the north direction). **C)** Spatial ratemaps of five cells classified as egocentric-boundary cells (egoBVCs). **D)** Spatial ratemaps of five cells classified as boundary-vector cells (BVCs). **E)** Spatial ratemaps of five memory slots reactivated by RNN-3. **F)** Spatial stability of cells in each RNN (see Supplementary Methods). **G)** Resultant vector length of cells in each RNN. **H)** Resultant vector of each cell in RNN-1. **I)** Comparison of cells responses to egocentric versus allocentric boundaries in each RNN.



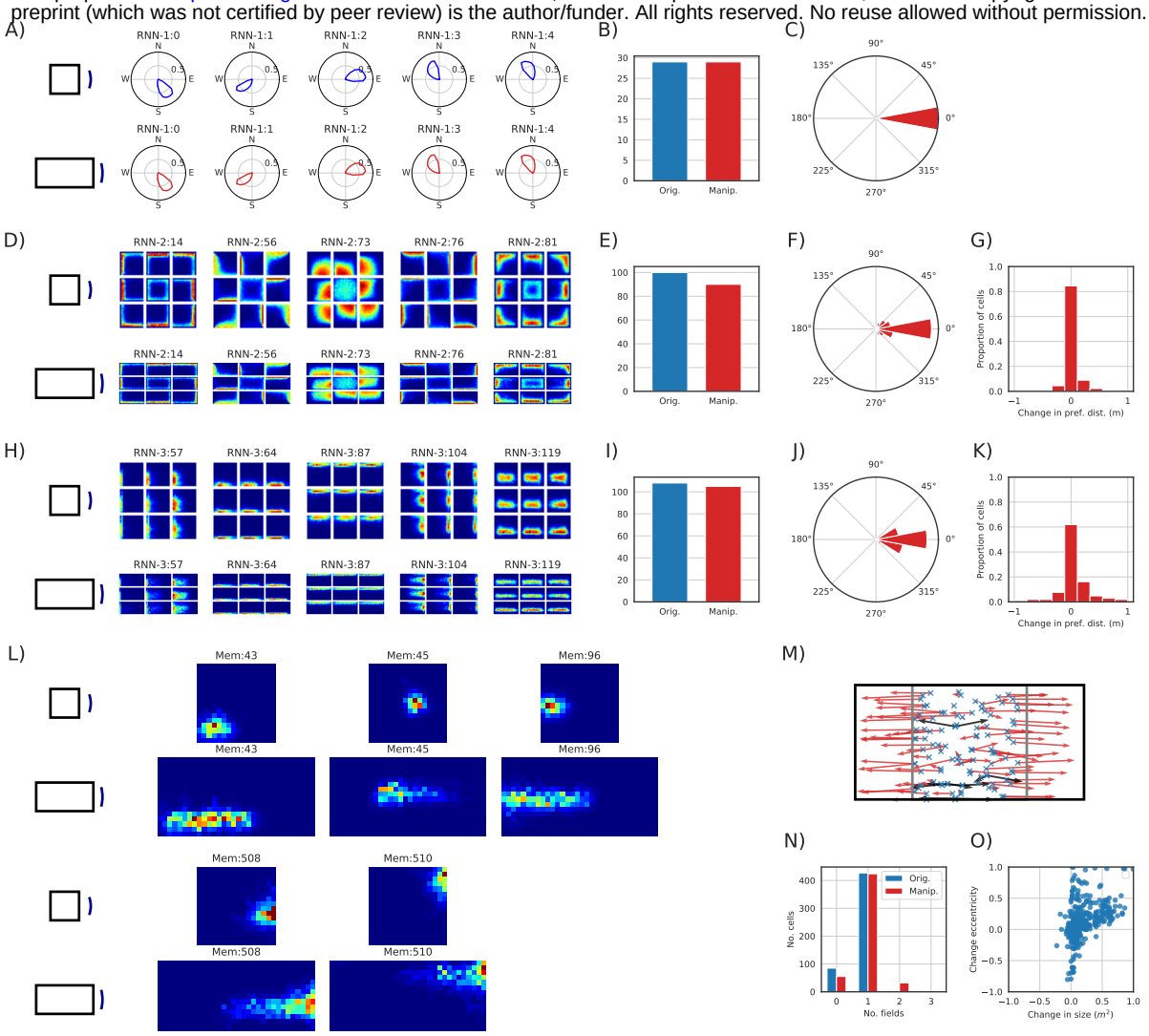
Extended Data Figure 3: Detail of boundary cells in Fig 2. **A)** First row, ratemap of five cells classified as egoBVCs. Second row, egocentric-boundary ratemap (see Supplementary Methods). Third row, allocentric-boundary ratemap (see Supplementary Methods). **B)** Histogram of preferred egocentric direction to boundary of all units classified as egoBVCs. Inset, Bayes information criterion numbers for a uniform distribution over angles (U) and mixtures of Von Mises distributions with different numbers of components. A mixture of two Von Mises distributions (shown in green), with a mode for boundaries in front agent, was selected (lowest BIC) (see Supplementary Methods). **C)** Histogram of preferred distance to boundary of all units classified as egoBVCs. The distribution over distances was well fit by an exponential distribution (shown in green) with mean distance of 35 cm. **D)** First row, ratemap of five cells classified as (allocentric) BVCs. Second row, egocentric-boundary ratemap. Third row, allocentric-boundary ratemap. **E)** Histogram of preferred allocentric direction to boundary of all units classified as BVCs. Inset shows Bayes information criterion numbers for a uniform distribution over angles (U) and mixtures of Von Mises distributions with different numbers of components. A uniform distribution (shown in green) was selected (lowest BIC). **F)** Histogram of preferred distance to boundary of all units classified as BVCs. The distribution over distances was well fit by an exponential distribution (shown in green) with mean distance of 29 cm.



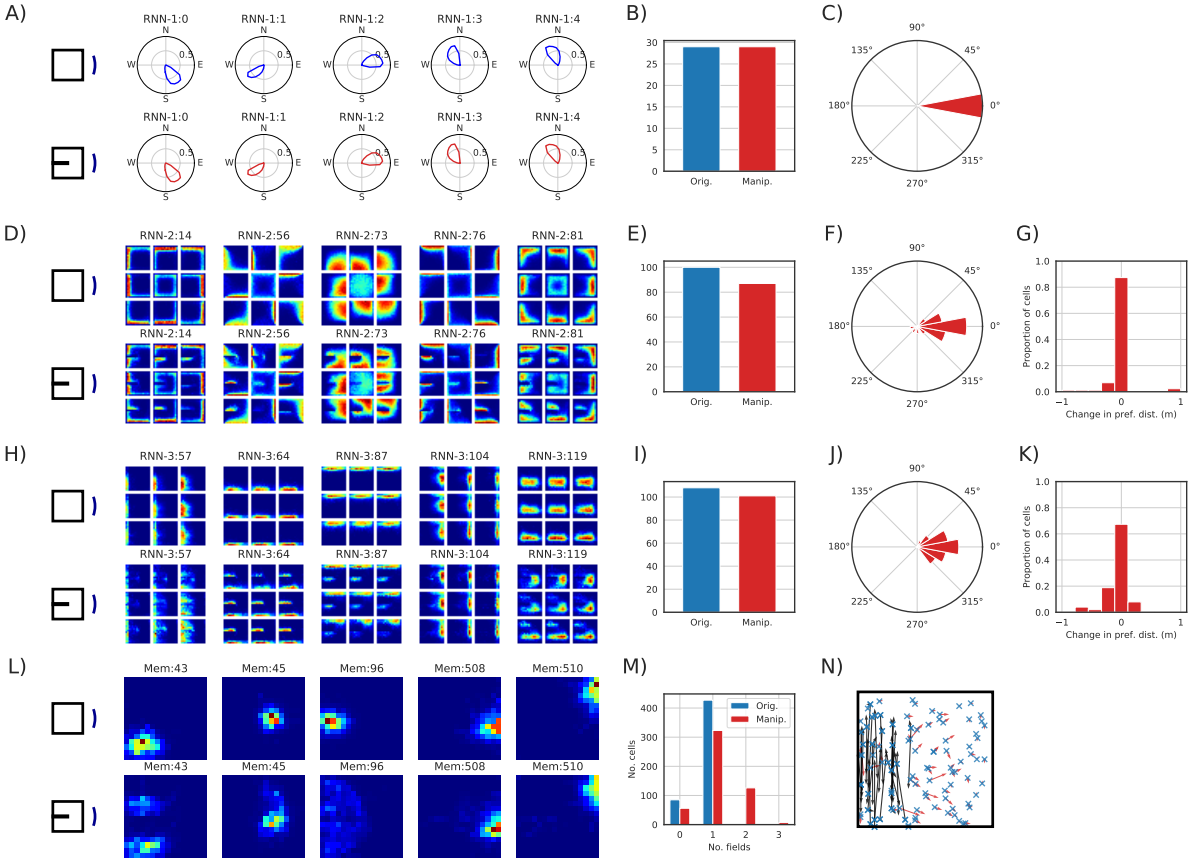
Extended Data Figure 4: Reactivation of memory slots (containing past RNN-3 states) resembles place-cell responses. **A)** Spatial ratemap of the reactivation of eight memory slots in the experiment described in Fig 2. Activity resembles that of biological place cells in the hippocampus. **B)** Spatial stability of memory slot reactivations (see Supplementary Methods). **C)** Resultant vector of memory slot reactivations. **D)** Cumulative total activity of place cells (memory slot reactivations) ordered by their total activity (blue). Place cells showed a high sparsity: 25% of cells account for 75% of activity. Activity in the three RNNs shows almost no sparsity (cells activated equally on average). **E)** Number of fields per place cell (see Supplementary Methods). **F)** Area of each activity field in square meters (total environment area is 4.8 m^2). **G)** Eccentricity of fields. A score of 1.0 indicates a perfectly round field; lower scores indicate elongation.



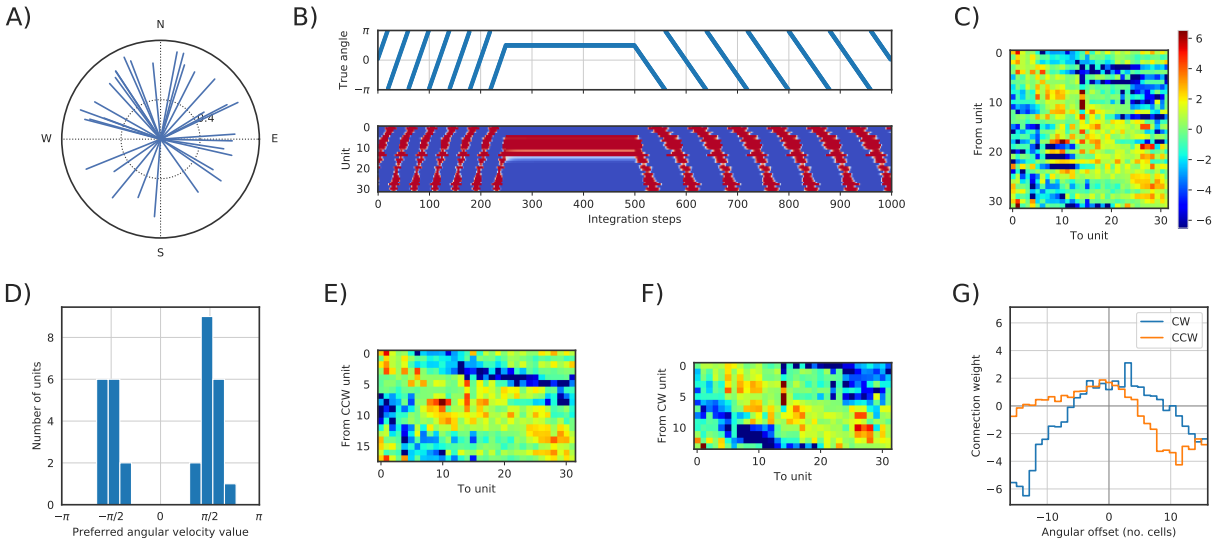
Extended Data Figure 5: Effects of distal cue rotation (45^o) on the representations of the model shown in Fig 2. There was no training after the manipulation. **A)** Polar plots of five cells identified as HD cells. Top row, original environment. Second row, after manipulation. HD activity followed the rotation of distal cues. **B)** Number of RNN units identified as HD cells before and after the environment manipulation. **C)** Histogram of change in phase of the resultant vectors of all HD cells. **D)** Ratemap of five cells identified as egoBVCs. Top row, original environment. Second row, after manipulation. Rotation of distal cues did not affect egoBVCs' activity. **E)** Number of RNN units identified as egoBVCs before and after the environment manipulation. **F)** Histogram of change in preferred egocentric angle to boundary of egoBVCs. **G)** Histogram of change in preferred distance to boundary of egoBVCs. **H)** Ratemap of five cells identified as (allocentric) BVCs. Top row, original environment. Second row, after manipulation. BVCs followed the rotation of distal cues. **I)** Number of RNN units identified as BVCs before and after the environment manipulation. **J)** Histogram of change in preferred allocentric angle to boundary of BVCs. **K)** Histogram of change in preferred distance to boundary of BVCs. **L)** Ratemap of five place-cell-like units. Top row, original environment. Second row, after manipulation. **M)** Number of fields for each place cell before and after the environment manipulation. **N)** Displacement of place-cell field centroid after manipulation. Blue cross indicates the original centroid, red arrows show their displacement. Black arrows show the displacement when a single field in the original environment split into several fields after manipulation. **O)** Histogram of change in phase of the place-cell centroid of activity when calculated in polar coordinates taking as origin the centre of the enclosure.



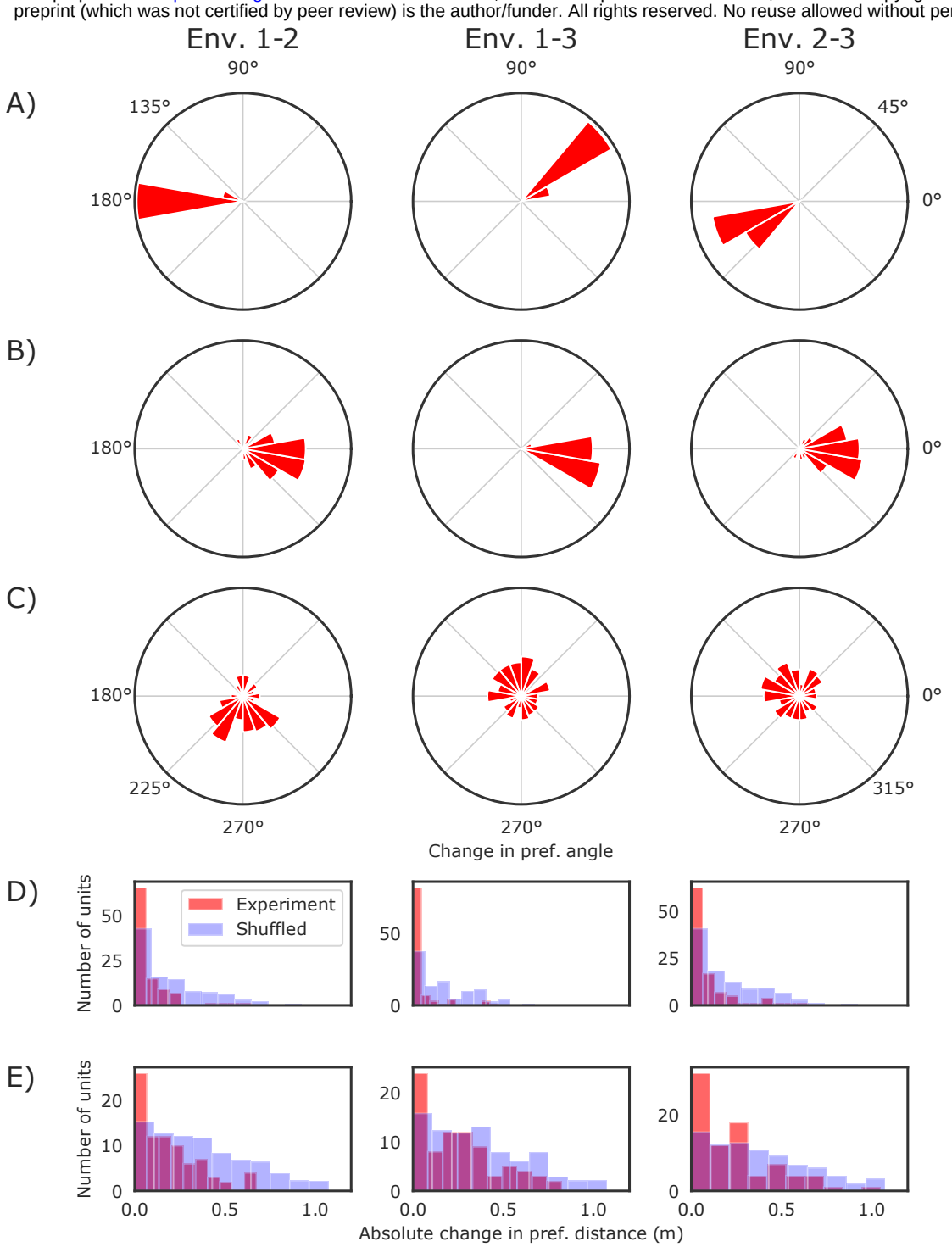
Extended Data Figure 6: Effects of enclosure resizing (width was doubled) on the representations of the model shown in Fig 2. There was no training after the manipulation. **A)** Polar plots of five cells identified as head-direction cells. Top row, original environment. Second row, after manipulation. The manipulation did not affect HD-cells. **B)** Number of RNN units identified as head-direction cells before and after the environment manipulation. **C)** Histogram of change in phase of the resultant vectors of all HD cells. **D)** Ratemap of five cells identified as egoBVCs. Top row, original environment. Second row, after manipulation. Cells maintained their egocentric response after the stretch. **E)** Number of RNN units identified as egoBVCs before and after the environment manipulation. **F)** Histogram of change in preferred egocentric angle to boundary of egoBVCs. **G)** Histogram of change in preferred distance to boundary of egoBVCs. **H)** Ratemap of five cells identified as (allocentric) BVCs. Top row, original environment. Second row, after manipulation. Cells maintained their allocentric response after the stretch. **I)** Number of RNN units identified as BVCs before and after the environment manipulation. **J)** Histogram of change in preferred allocentric angle to boundary of BVCs. **K)** Histogram of change in preferred distance to boundary of BVCs. **L)** Ratemap of five place-cell-like units. Top, original environment; below, after manipulation. The manipulation caused the activity fields of most cells to stretch with the stretched environment axis. However some cells' activity field split into two fields. **M)** Displacement of place-cell field centroid after manipulation. Blue cross indicates the original centroid, red arrows show their displacement. Black arrows show the displacement when a single field in the original environment split into several fields after manipulation. **N)** Number of fields for each place cell before and after the environment manipulation. **O)** Change in size and eccentricity of each place cell's activity field.



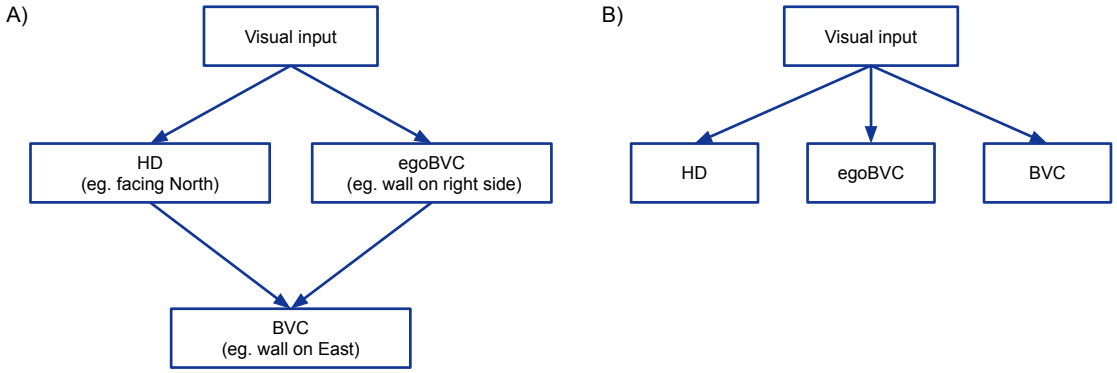
Extended Data Figure 7: Effects of extra barrier insertion on the representations of the model shown in Fig 2. There was no training after the manipulation. **A)** Polar plots of five cells identified as HD cells. Top row, original environment. Second row, after manipulation. The manipulation did not affect HD cells. **B)** Number of RNN units identified as HD cells before and after the environment manipulation. **C)** Histogram of change in phase of the resultant vectors of all HD cells. **D)** Ratemap of five cells identified as egoBVCs. Top row, original environment. Second row, after manipulation. Cells maintained their egocentric response to the new barrier. **E)** Number of RNN units identified as egoBVCs before and after the environment manipulation. **F)** Histogram of change in preferred egocentric angle to boundary of egoBVCs. **G)** Histogram of change in preferred distance to boundary of egoBVCs. **H)** Ratemap of five cells identified as (allocentric) BVCs. Top row, original environment. Second row, after manipulation. Cells maintained their allocentric response to the new barrier. **I)** Number of RNN units identified as BVCs before and after the environment manipulation. **J)** Histogram of change in preferred allocentric angle to boundary of BVCs. **K)** Histogram of change in preferred distance to boundary of BVCs. **L)** Ratemap of five place-cell-like units. Top row, original environment. Second row, after manipulation. The extra barrier caused cells in the region to duplicate their fields. Cells with distant activity fields were not affected. **M)** Number of fields for each place cell before and after the environment manipulation. **N)** Displacement of place-cell field centroid after manipulation. Blue cross indicates the original centroid, red arrows show their displacement. Black arrows show the displacement when a single field in the original environment split into several fields after manipulation.



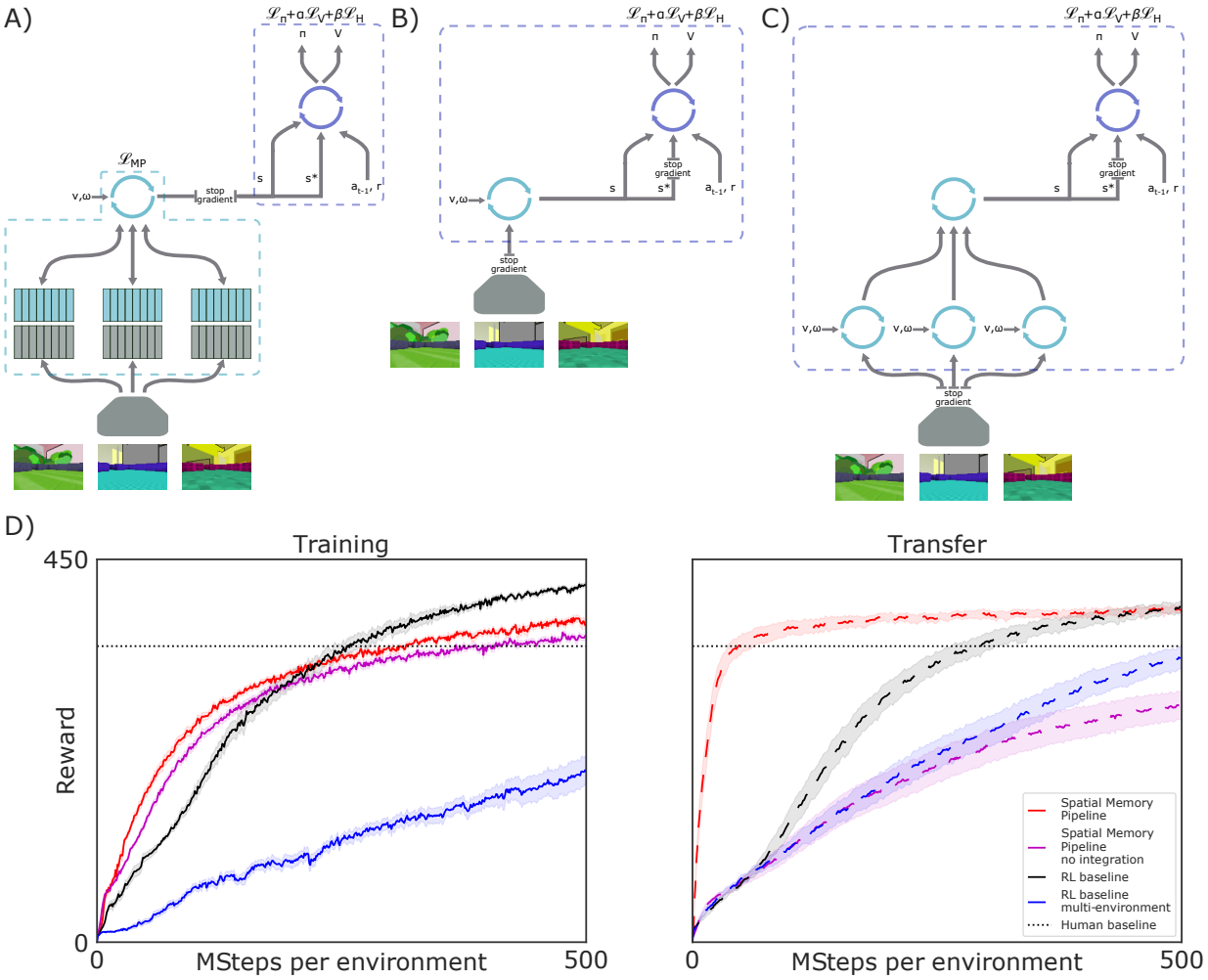
Extended Data Figure 8: Head direction cell connectivity learnt by a Vanilla-RNN network. **A)** Polar plot of the resultant vectors of cells in RNN-1. **B)** Activation of each unit over 1000 steps of blind integration. Top: True integrated angle. Bottom: Activation of HD-units in RNN-1 through time (units ordered by the phase of their resultant vectors). For the first 250 steps the angular velocity was set to $\pi/20$ rad/step, the following 250 steps 0 rad/step, the remaining 500 steps $-\pi/40$ rad/step. **C)** Weights in the RNN dynamics matrix, $\mathbf{W}$. Columns and rows ordered by the phase of their resultant vector. **D)** Histogram of preferred angular velocities for each cell. **E)** Matrix of weights from CCW cells to all other cells. **F)** Matrix of weights from CW cells to all other cells. **G)** Average weights connecting cells in each of the two rings to all other cells ordered by their angular offset.



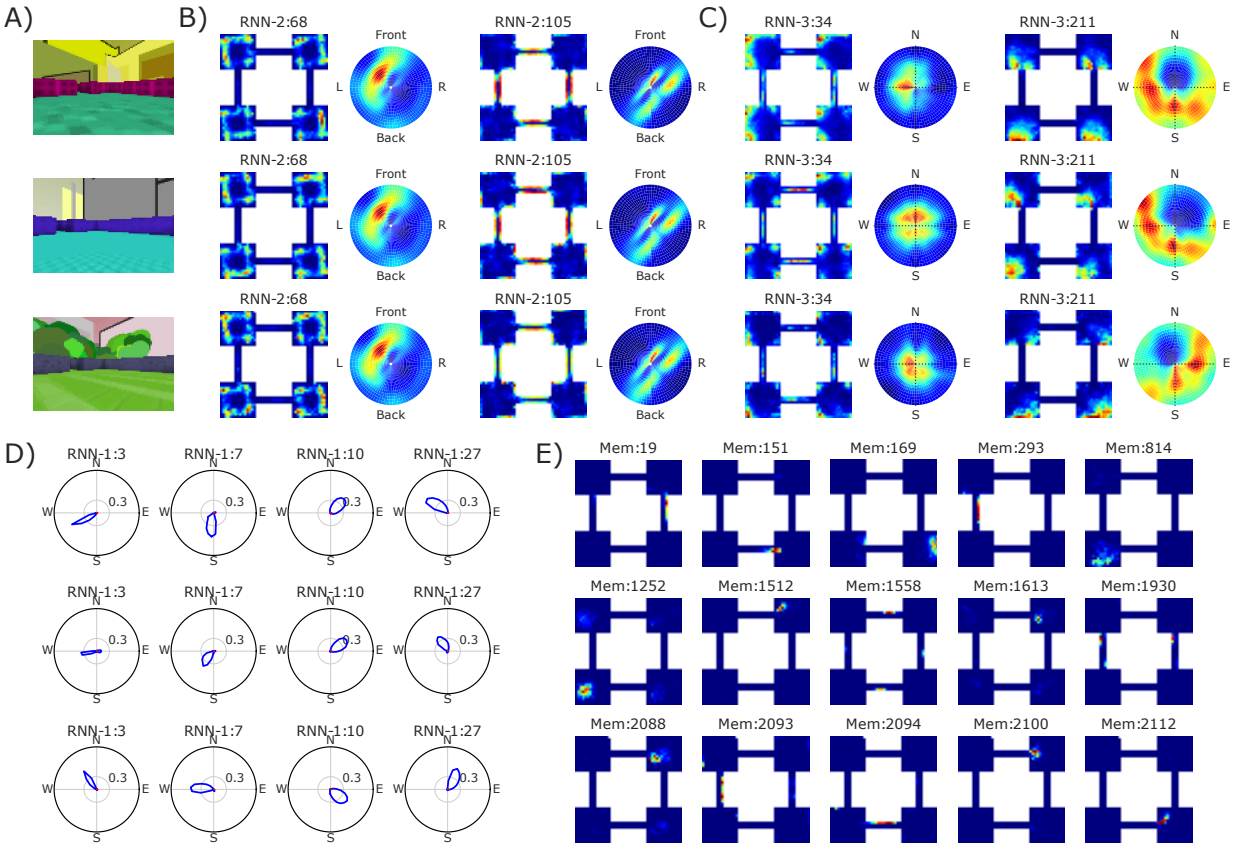
Extended Data Figure 9: Stability of representations learnt across three environments: square (1), circular (2) and trapezoidal (3). First column compares environments 1 and 2, second column environments 1 and 3, third column environments 2 and 3. **A)** Histogram of HD cell preferred direction differences between environment pairs. HD cells rotate coherently between environments. **B)** Histogram of egoBVC preferred egocentric direction differences between environments. The egoBVCs preserve their angular tuning between environments. **C)** Histogram of BVC preferred allocentric direction differences between environments. The BVCs do not rotate coherently between environments. **D)** Histogram of egoBVC preferred distance differences between environments (red) and normalised histogram of differences between randomly shuffled units (blue). The egoBVCs preserve their distance tuning across environments. **E)** Same as D) for BVCs. The distance tuning is significantly preserved across environments, although not as tightly as in the case of egoBVCs.



Extended Data Figure 10: Hypothesised BVC mechanisms. **A)** Traditional BVC mechanism: BVCs result from the conjunction of HD and egoBVC signals. **B)** Proposed BVC mechanism: BVCs are driven by the reactivation of visual memories and temporal coherence.



Extended Data Figure 11: Reinforcement learning architectures and baselines. **A)** Agent training with the Spatial Memory Pipeline. The visual inputs from three visually different training environments were stored in separate memory slot banks. The Spatial Memory Pipeline (light blue frame) was trained as in the unsupervised experiments to minimise the loss in prediction of memory reactivations. The policy (purple frame) was trained to minimise a combination of policy gradient, value and entropy losses. **B)** Agent training baseline with an LSTM replacing the Spatial Memory Pipeline. **C)** Agent training baseline with one LSTM per training environment and a common LSTM replacing the Spatial Memory Pipeline. **D)** Left, training learning curves, and right, transfer learning curves, for the architectures in A (two curves), B and C. Error regions represent the standard deviation of the average episode returns across 3 training environments and 25 training seeds (training curves) or 5 transfer environments for each of the 25 trained agents (transfer curves).



Extended Data Figure 12: Representations across three reinforcement learning training enclosures, one enclosure in each row. **A)** Agent view of the enclosures. **B)** Ratemaps of two egoBVCs in RNN-2, with their egocentric-boundary ratemap (see Supplementary Methods). **C)** Ratemaps of two BVCs in RNN-3, with their allocentric-boundary ratemap (see Supplementary Methods). **D)** Polar plots of four HD cells in RNN-1. **E)** Ratemaps of five place cells in each environment. Note that, unlike the egoBVC, BVC or HD units, there is no relationship between the place cell units across environments, since they correspond to separate memory banks.

Movie S1: `https://youtu.be/tkpX7Javyh0` Reconstructions of visual input along the replay trajectory in Fig 4E (goal on the left arm of the T-maze). Left, ground-truth vision, reconstructed from the actual visual encoding vectors along the agent's trajectory. Right, the agent's reconstruction of the visual input from the sequence of visual embeddings in the memory slot with highest prediction probability according to the RNN states at each time step. Middle plot, agent trajectory in the enclosure. The arrow shows the agent's position and heading direction. The arrow is red while the agent is receiving visual input, at the beginning of the trajectory; for the rest of the trajectory the agent is blind (arrow is black).

Movie S2: `https://youtu.be/_kTZ9x2ZPfo` Reconstructions of visual input along the replay trajectory in Fig 4F (goal on the right arm of the T-maze).

Configuration	Value	Meaning
b_m	0.13	Linear velocity Rayleigh scale while moving (m/s)
b_s	0.0	Linear velocity Rayleigh scale while stopped (m/s)
$\mu^{(\phi)}$	0	Rotation velocity Gaussian distribution mean (deg/s)
$\sigma^{(\phi)}$	330	Rotation velocity Gaussian distribution standard deviation (deg/s)
d	0.03	Distance to wall to activate wall avoidance (m)
a	90	Angle limit to activate wall avoidance (deg)
slowdown	0.25	Velocity reduction factor when avoidance activated
dt	0.02	Simulation step time increment (s)
$\mu_{(stop)}$	50	Mean period for stop/move switching (simulation steps)
n	7	Number of simulation steps per trajectory sample
N	500	Number of samples per trajectory

Extended Data Table 1: Parameters of rat-like motion model[24] to generate trajectories for unsupervised experiments.

Parameter	Single-environment	Multi-environment
BPTT unroll length	50	50
Batch size	32	32
Training batches	$200,000$	$200,000$
Optimiser	Adam	RMSProp
Learning rate (RNNs)	$3 \cdot 10^{-5}$	$3 \cdot 10^{-4}$
Learning rate (memory embeddings)	$1 \cdot 10^{-2}$	$3 \cdot 10^{-2}$
Other optimiser params	$\beta_1 = 0.9$, $\beta_2 = 0.999$	decay=0.9, momentum=0
Dropout in RNN states predicting memories	0.5	0.5

Extended Data Table 2: Parameters for training in unsupervised experiments.

Configuration	Value (unsupervised experiments)	Value (RL experiments)
Image shape	64 x 64 x 3	96 x 72 x 3
Number of conv. layers	4	4
Conv. kernel sizes	5, 5, 3, 3	5, 5, 3, 3
Conv. strides	2, 2, 1, 1	2, 2, 1, 1
Conv. output channels	16, 16, 32, 32	32, 32, 64, 64
Conv. padding	Same	Same
Output vector ($\mathbf{y}_t^{(enc)}$) size	64	128

Extended Data Table 3: Configuration of the visual encoding level in *Spatial Memory Pipeline*.

Configuration	Value (unsupervised experiments)	Value (RL experiments)
S	512	1024
R	3	3
$\mathbf{y}_t$	$\mathbf{y}_t^{(enc)}$	$\mathbf{y}_t^{(enc)}$
F_1, F_2, F_3	Sigmoid-LSTM	Sigmoid-LSTM
G_1, G_2, G_3	Sigmoid-LSTM	Sigmoid-LSTM
N_1	32	64
N_2	128	256
N_3	128	256
$\mathbf{v}_{1,t}$	$10 \cdot (\cos(\omega_t), \sin(\omega_t))$	$10 \cdot (\cos(\omega_t), \sin(\omega_t))$
$\mathbf{v}_{2,t}$	$10 \cdot (\cos(\omega_t), \sin(\omega_t), s_t)$	$10 \cdot (\cos(\omega_t), \sin(\omega_t), s_t, s_{\perp,t})$
$\mathbf{v}_{3,t}$	()	()
H_{react}	0.5 nats	1.0 nats
$P_{correction}$	0.1	0.1
$P_{storage}$	0.0000625	0.0001

Extended Data Table 4: Configuration of the first integration level in the *Spatial Memory Pipeline*. Where $\mathbf{y}_t^{(enc)}$ is the output of the visual encoder, ω_t is the angular velocity, s_t speed parallel to the direction of heading, and (only for RL experiments) $s_{\perp,t}$ the speed perpendicular to the direction of heading.

Configuration	Value (RL experiments)
S	512
R	1
$\mathbf{y}_t$	$\mathbf{x}_{3,t}$
F_1	Sigmoid-LSTM
G_1	Sigmoid-LSTM
N_1	256
$\mathbf{v}_{1,t}$	$(\mathbf{x}_{1,t}, s_t, s_{\perp,t})$
H_{react}	1.0 nats
$P_{correction}$	0.1
$P_{storage}$	0.0001

Extended Data Table 5: Configuration of the second integration level in reinforcement learning experiments. $\mathbf{x}_{1,t}$ is the state in RNN-1 of level 1, where head-direction cells develop; and $\mathbf{x}_{3,t}$ is the state in RNN-3 of level 1, where allocentric BVCs appear.

There is Supplemental Information that contains additional results, discussion and methods.

Acknowledgements We thank Matt Botvinick, Vivek Jayaraman, Tim Behrens, Daan Wierstra, Sergei Lebedev, Christopher Summerfield, Kimberly Stachenfeld, and Charlie Beattie for their valuable advice.

Competing Interests The authors declare that they have no competing financial interests.

Correspondence Correspondence and requests for materials should be addressed to Benigno Uria and Charles Blundell (email: buria@google.com, cblundell@google.com).

Author Contributions
Conceived project: B.U, A.B, B.I, C.Ba, C.Bl, D.K, D.H.
Contributed ideas to experiments: B.U, C.Ba, B.I, C.Bl, A.B, V.Z, D.K
Performed experiments and analysis: B.U, B.I, V.Z, A.B
Development of testing platform and environments: B.I, V.Z, B.U, A.B
Wrote paper: C.Ba, C.Bl, B.U, B.I, A.B, D.K, D.H
Managed project:C.Bl, C.Ba, D.H

Supplemental Information for *The Spatial Memory Pipeline: a model of egocentric to allocentric understanding in mammalian brains*

Benigno Uria[*,1], Borja Ibarz[*,1], Andrea Banino[1], Vinicius Zambaldi[1], Dharshan Kumaran[1], Demis Hassabis[1], Caswell Barry[2], Charles Blundell[1]
[1]DeepMind
[2]University College London
[*]Equal contribution

This section contains:

Supplementary Results for *The Spatial Memory Pipeline: a model of egocentric to allocentric understanding in mammalian brains*

Emergent representations in a circular environment The appearance of spatial representations shown in Fig 2 does not depend on the square geometry of the environment. The emergence of such representations is robust to training in enclosures with different environment shapes. To demonstrate this, the model was trained from scratch in a simulated circular environment of radius 1.5 meters, white walls, no internal cues, and distal cues consisting of a city-like scene. The model developed similar representations (Extended Data Fig 2) to those appearing in the square enclosure.

The majority of units in the first RNN module (RNN-1), which received only angular velocities and corrections from the visual memories, exhibited activity modulated by the agent's direction of facing (Extended Data Fig 2A-B). In total 94% (30/32) of units were classified as head direction cells (resultant vector length $>$0.48) with unimodal responses distributed uniformly in the unit circle (Extended Data Fig 2H, uniform distribution was selected under BIC over mixtures of Von Mises).

The second module (RNN-2) received angular velocity and speed, together with corrections from the visual memories. Units in this RNN encoded distances relative to boundaries at a particular heading (Extended Data Fig 2C). As in the square arena, they resembled egoBVC responses, with 94% (120/128) of units identified as egoBVCs (ego-boundary score $>$0.1, 99th percentile of bin-shuffled distribution, Extended Data Fig 2I). None of the 128 units were identified as BVCs. Here cells did not exhibit HD cell-like patterns of activity (mean resultant vector 0.01, 0/128 units classified as HD cells).

The third (RNN-3) received no self-motion inputs, thus was dependent upon corrections from visual reactivations as its sole input and learning signal. Units in this RNN were characterised by spatially stable responses (Extended Data Fig 2F). As in the square enclosure, the activity profile of this module was reminiscent of BVCs (Extended Data Fig 2D). 83% (106/128) of units were classified as BVCs (BVC score$>$0.11, 99th percentile of shuffled distribution) – 16% (20/128) met the criteria for egoBVCs, but 13 of those 20 had a higher BVC score than egoBVC score. Units in this layer were not modulated by heading direction (mean resultant vector 0.11, 0/128 units classified as HD cells).

Effects of distal cue rotation on the model's representations First, we rotated the visible distal cues by 45^o clockwise (Extended Data Fig 5). The preferred orientation of HD units in RNN-1 rotated en masse, tracking the cue rotation the same way that rodent head direction cells are controlled by visual cues (resvec phase rotated by 43^o on average, 1^o SD) – tuning width was unchanged (average 76.5^o, 17.8^o SD). The manipulation did not significantly affect egoBVC activity (preferred egocentric directions to wall rotated by 3^o on average, 6^oSD). The firing correlates of BVCs rotated with the head direction system (35^o, 9^oSD). The activity field of the place-cell-like memory slots rotated around the centre of the environment (27^o on average, 20^o SD).

Effects of enclosure stretch on the model's representations We transformed the training environment stretching it by 100% along the horizontal axis to form a 4.4 m by 2.2 m rectangle (Extended Data Fig 6). Again, the response characteristics of HD cells as well as egoBVCs

and BVCs were unchanged, and simply extended along the elongated walls like their biological counterparts[5,6,16,17]. Place cell responses closely mirrored those observed in rodents[30], tending to be stretched along the axis of elongation and in some cases becoming bimodal (Extended Data Fig 6L-O).

Effects of barrier insertion on the model's representations In a similar fashion, we introduced a barrier into the virtual environment (Extended Data Fig 7). The responses of HD cells were largely unaffected, maintaining the same direction of tuning (0^o average change in phase, 0.2^o SD). Similarly, egoBVCs and BVCs retained their basic firing correlates, responding to the new barrier as they had to the existing walls, resulting in the inception of additional firing fields (Extended Data Fig 7D,H). Biological BVCs and egoBVCs are known to respond similarly; indeed, the predictable response of these cells to extra walls is considered to be a diagnostic feature[5,6]. Place cell responses were more complex, some – typically those further from the barrier – were unaffected (60%), whereas 22% formed a duplicate field on either side of the barrier (Extended Data Fig 7L-N). Notably, similar outcomes have been reported in empirical studies[6329].

Preservation of spatial characteristics across environments To study the properties of the Spatial Memory Pipeline representations across environments we trained concurrently in three different enclosures (Fig 4A). In RNN-1, although the preferred firing directions of HD cells changed between environments, the responses of all units were rotated by a similar amount, maintaining the relative angular tuning between cell pairs (resultant vector phase rotation between environments 1 and 2 was 179^o on average, 0.5^o SD; between environments 1 and 3, -41^o on average, 0.6^o SD; Fig 4A, Extended Data Fig 9A). The preservation of HD-cell characteristics across environments is consistent with the presence of ring-attractor dynamics, as analysed in single-environment experiments (Fig 3).

The ensemble response properties of egoBVCs in RNN-2 were similarly preserved. Between the three different environments individual egoBVCs maintained their distance and directional tuning, without rotation (mean resultant vector phase rotation 2^o on average, 25^o SD, between environments 1 and 2; 0^o on average, 6^o SD between 1 and 3, Extended Data Fig 9B; mean distance tuning difference 0.07 m between 1 and 2, 0.04 m between 1 and 3, $p < 10^{-4}$ compared to differences with shuffled units, Extended Data Fig 9D). These representations do not depend on arbitrary distal cues, and are therefore preserved.

The BVCs in RNN-3 tended also to preserve distance tuning (0.19 m mean difference in distance tuning between environments 1 and 2, 0.26 m between 1 and 3, SD 0.22, $p < 10^{-4}$ compared to differences with shuffled units, Extended Data Fig 9E), but not directional tuning across environments (phase rotation had SD 69^o between environments 1 and 2, and 89^o between 1 and 3; Extended Data Fig 9C).

RNN-1 ring attractor analysis To understand the mechanism by which our model tracks head direction, we repeated the experiment shown in Fig 3 substituting the efficient but complex LSTM integrators with Vanilla-RNNs. In Vanilla-RNNs each activity state is a simple function of the previous state and inputs, thus is amenable to a mechanistic analysis[34] (see Supplementary Methods). As expected, this new experiment also developed head-direction cells in RNN-1, all 32 units being classified as head direction cells (resultant vector length $>$0.4, 99-th percentile of shuffled data, Extended Data Fig 8A). Like the LSTM-based model, the Vanilla-RNN effectively

integrated angular velocity over many time steps (Extended Data Fig 8B).

The simplicity of Vanilla-RNNs allowed us to examine the connectivity that supported angular velocity integration, revealing a striking similarity with that found in the fly[33] and hypothesised for mammals[32]. The weight matrix of dynamics, $\mathbf{W}$, when displayed ordered according to each units' preferred firing direction (Extended Data Fig 8C), resembled a circulant matrix, with a diagonal band of excitatory connections surrounded by diagonal bands of inhibition. That is, the connectivity between units forms a ring with local excitation and long-range inhibition. The angular-velocity tuning of the cells, calculated as the arc-tangent of the 2-dimensional vector of control weights, $\mathbf{V}$, that receive as input the cosine and sine of the angular velocity, showed a clear split of the units in two groups, 18/32 units being activated by positive angular velocities (ccw-cells) and 14/32 activated by negative angular velocities (cw-cells) (Extended Data Fig 8D). Plotting the weight matrix of dynamics separately for each group revealed the mechanism by which angular velocity was integrated (Extended Data Fig 8E-F). Each cell preferentially excited other cells offset around the ring in the same direction as their angular-velocity tuning, an asymmetry that became more obvious when the average weights to units with different relative firing directions were calculated (Extended Data Fig 8G).

Supplementary Methods for *The Spatial Memory Pipeline: a model of egocentric to allocentric understanding in mammalian brains*

Decoding of position and heading angle For decoding of position and head direction (Fig 1D) we trained a single-layer MLP (256 hidden units with tanh non-linearity) to predict either the true (x, y) position of the agent in the environment, or the cosine and sine of its true head direction, from either the vector of log probabilities of activation of the visual slots or the concatenation of all the RNN outputs. We used RMSProp as optimiser (learning rate 10^{-4}, decay 0.9, no momentum) with square loss. The decoder was trained on 10 million batches of 25 frames sampled at random out of 300,000 frames from the same trajectories used for unsupervised training of the Spatial Memory Pipeline. The decoding error shown in Fig 1D for position is the mean Euclidean distance over the whole training set between the (x, y) output of the decoder and the true position. For head direction, it is the mean absolute difference between the true head direction and the arc-tangent of the cosine and sine prediction of the decoder. Error bars show the standard error of the mean calculated as the standard deviation of 100 bootstrapped means computed from 5000 decoded samples.

Quantitative categorisation of spatial representations Where possible we have used the same functional characterisations of cells used for rodent data. Resultant vector lengths of allocentric direction-of-facing were used to characterise head-direction cells[64]. Resultant vector lengths of egocentric-boundary ratemaps were used to characterise egocentric boundary cells (egoBVCs) [65]. We used the same procedure with allocentric-boundary ratemaps to detect allocentric boundary cells (BVCs).

Our simulation datasets correspond to much longer duration than is common in rodent studies. For this reason we avoided using thresholds based on random shifts of the data, as this results in very low thresholds due to a null hypothesis that removes most of the dependency of activations from any environmental features. We used a bin-shuffling procedure that results in much more stringent, and qualitatively pleasing, thresholds. Thresholds were calculated as the 99-th percentile of resultant vector lengths for 1000 bin shuffles for each unit. All units from all RNNs were used to calculate the threshold of each cell category.

Spatial ratemaps We measured and plotted the dependence of a cell's activity on the agent's location by using spatial ratemaps. We partitioned the environment into a grid of 14cm by 14cms square bins aligned with the cardinal directions. The spatial ratemap for a cell is the matrix of its average activity when the agent is located inside each of these bins.

When we aimed to show the dependence of cell's activity on both spatial location and head-direction, the per-octant spatial ratemaps were calculated and plotted surrounding the spatial-ratemap. These are eight spatial ratemaps where the data is restricted to times when the agent's head direction is contained in each of the eight intervals of 45^o centered at the cardinal and inter-cardinal directions.

Resultant vector of binned data Resultant vectors were calculated as a single complex number from mean activities binned by angle:

$$R_u = \sum_b \frac{F_{b,u}}{\sum_{b'} F_{b',u}} e^{ib\frac{2\pi}{B}} \tag{11}$$

where $F_{b,u}$ is the average activity of unit u for the b-th bin out of a total of B bins.

The length of the resultant vector corresponds to the modulus of R_u and serves as a statistical measure of circular concentration (inverse variance). While the argument of R_u is a statistical measure of preferred direction of activation.

Head-direction cells Head direction cells characteristically fire in a small range of allocentric direction. This preference for a single direction can be measured by the length of the resultant vector of activations.

In order to calculate the resultant vector of each unit, we partitioned allocentric directions into 20 bins. The angular ratemap, $F_{b,u}$, of the u-th unit, with instantaneous activity $a_{u,t}$ at time t, was calculated as:

$$F_{b,u} = \mathop{\mathbb{E}}_{t|\alpha_t \in b-\text{th bin}} a_{u,t} \tag{12}$$

where α_t the agent's direction of facing at time t, and $\alpha_t \in b$-th bin if $(b-0.5)\frac{2\pi}{20} < \alpha_t \leq (b+0.5)\frac{2\pi}{20}$.

The tuning-curve width for units classified as head-direction cells was calculated as $\frac{2\pi}{20}$, the width of a single bin, times the number of bins where the mean activity was greater than 0.5 of the greatest mean-bin activity for that unit.

Egocentric boundary score Egocentric boundary cells were characterised by the resultant vector of a the egocentric-boundary ratemap (EBR) [65]. The EBR uses the agent's position and direction of facing as frame of reference and computes the average instantaneous activity when a boundary is present at a particular distance and egocentric direction. In our unsupervised experiments the EBR was calculated for bins of 4° by 2.5cm. The maximum distance considered was half of the maximum distance between opposing walls. Examples of EBRs can be seen in the middle row of Extended Data Fig 3A and D.

The length of resultant vector of the EBR (averaged over distance) was used to detect egocentric boundary cells. For a cell classified as an egoBVC, the angle of the resultant vector was taken as its preferred direction and the distance of the maximum activity along this direction in the EBR as its preferred distance.

Allocentric boundary score We characterised allocentric boundary cells using an allocentric-boundary ratemap (ABR). In the ABR the agent's position is used as frame of reference but not its direction of facing, the relative angle to boundaries is locked to the allocentric north direction. The ABR computes the average instantaneous activity where a boundary is present at a particular distance and allocentric direction. In our unsupervised experiments the ABR was calculated for

bins of 4° by 2.5cm. The maximum distance considered was half of the maximum distance between opposing walls. Examples of ABRs can be seen in the bottom row of Extended Data Fig 3A and D

The length of resultant vector of the ABR (averaged over distance) was used to detect allocentric boundary cells (BVC). For a cell classified as a BVC, the angle of the resultant vector was taken as its preferred direction and the distance of the maximum activity along this direction in the ABR as its preferred distance.

Spatial stability score Where reported, the spatial stability of a unit was measured as the Pearson correlation of its spatial ratemap for two halves of the data. In our experiments each half of the data was comprised of 200,000 data points, which under the motion model used would amount to 3000 seconds for each half.

Place cell field characterisation The activity field of a cell was calculated using the following procedure on its spatial ratemap: (1) a Gaussian filter of radius 1 bin was applied, (2) bins with value higher than half the maximum value were considered part of the activity fields, (3) fields were segmented using the `skimage.measure.label` function, (4) fields smaller than 3 adjacent bins were discarded, and (5) the attributes of each field (position, size, and eccentricity) were calculated using the `skimage.measure.regionprops` function with the original ratemap as `intensityimage` parameter. We used version 0.14.0 of `skimage` throughout.

Model selection criterion Where a probability distribution is reported as fitting data (e.g. the distribution of HD angles, or the angular preferences of egoBVCs) we used the Bayesian information criterion to compare several alternative distributions. This criterion penalises the log-likelihood of the model, L, by a term that depends on the number of parameters, k, and number of data points, n:

$$BIC = k \ln n - 2L \tag{13}$$

A lower BIC signals a better model fit.

Reinforcement learning baselines Extended Data Fig 11A depicts the architecture of our agent as described in the previous paragraph. The baselines we used to compare performance in the water maze transfer task are the following:

- An agent with the same architecture, receiving visual corrections at every time step (instead of every 10 steps on average). This is the baseline labelled *Memory pipeline no integration* in Extended Data Fig 11D.
- An agent where the Spatial Memory Pipeline was replaced by a generic recurrent network (LSTM) (Extended Data Fig 11B) integrating visual and velocity inputs. The size of the LSTM output was 832 units, to make it the same as the concatenation of all the RNNs in the Spatial Memory Pipeline. Unlike the memory pipeline, the LSTM did not have a separate unsupervised loss for training, it was trained directly from the gradients of the policy loss.
- An agent where the Spatial Memory Pipeline was replaced by a two-layer network of LSTMs (Extended Data Fig 11C), where the first layer consisted of three LSTMs (output

size 512 units), each integrating the visual and velocity inputs from one of the three training environments, and the second layer consisted of a single LSTM (output size 832 units) integrating the batched outputs of the first-layer LSTMs. This network paralleled the Spatial Memory Pipeline architecture, with the first-layer LSTMs playing the role of the per-environment memory banks and the second-layer LSTM the role of the memory pipeline RNNs. For transfer, the first layer was replaced by a single 512-unit LSTM, and the rest of parameters were kept from the trained agent. As with the single-layer LSTM baseline, the LSTMs did not have a separate unsupervised loss for training, and were trained directly from the gradients of the policy loss.

Videos of replay in T-maze The Spatial Memory Pipeline can run on velocity inputs without visual reactivations. In our artificial replay experiments, an agent trained on the water maze task in a T-shaped enclosure was subsequently tested without providing visual inputs, except for a number of steps at the beginning of each trajectory. As shown in Fig 4E-F, predicted place cell activations and head-direction cell activity, as well as value predictions, correspond closely to what would be expected in the presence of visual input. Since the RNNs have been trained to predict the activations of visual memory slots, it is also possible to reconstruct the visual scene from the low-dimensional embeddings contained in the slots. We demonstrate this in Movie S1 and S2. The left-hand side images in the videos, shown only for comparison, are reconstructed from the ground-truth visual inputs along the agent's imagined trajectory (which are not provided to the agent, save for the first few steps; see Methods). The right-hand side images are reconstructed from the visual embedding in the memory slot with the highest prediction probability according to the RNN states, equation (4). Note that the reconstruction from memory is limited to the available embeddings in the memory (1024, see Methods). As with place cell, HD-cell and value predictions, the reconstruction from memory agrees well with the ground-truth visual inputs along the trajectory.

63. Rivard, B., Li, Y., Lenck-Santini, P.-P., Poucet, B. & Muller, R. U. Representation of objects in space by two classes of hippocampal pyramidal cells. *The Journal of general physiology* **124**, 9–25 (2004).

64. Knight, R. *et al.* Weighted cue integration in the rodent head direction system. *Philosophical Transactions of the Royal Society B: Biological Sciences* **369**, 20120512 (2014).

65. Alexander, A. *et al.* Egocentric boundary vector tuning of the retrosplenial cortex. *bioRxiv* 702712 (2019).