University of Reading

Lessons from reinforcement learning for biological representations of space

Article

Accepted Version

Creative Commons: Attribution-Noncommercial-No Derivative Works 4.0

Muryy, A., Narayanaswamy, S., Nardelli, N., Glennerster, A. ORCID: https://orcid.org/0000-0002-8674-2763 and Torr, P. H. S. (2020) Lessons from reinforcement learning for biological representations of space. Vision Research, 174. pp. 79-93. ISSN 0042-6989 doi: 10.1016/j.visres.2020.05.009 Available at https://centaur.reading.ac.uk/90944/

It is advisable to refer to the publisher's version if you intend to cite from the work. See Guidance on citing.

To link to this article DOI: http://dx.doi.org/10.1016/j.visres.2020.05.009

Publisher: Elsevier

All outputs in CentAUR are protected by Intellectual Property Rights law, including copyright law. Copyright and IPR is retained by the creators or other copyright holders. Terms and conditions for use of this material are defined in the End User Agreement.

www.reading.ac.uk/centaur

CentAUR

University of Reading

Central Archive at the University of Reading

Reading's research outputs online

Lessons from reinforcement learning for biological representations of space

Alex Muryy[a], Siddharth Narayanaswamy[b], Nantas Nardelli[b], Andrew Glennerster[a,*], Philip H. S. Torr[b]

[a] *School of Psychology and Clinical Language Sciences, University of Reading, UK*
[b] *Department of Engineering, University of Oxford, UK*

Abstract

Neuroscientists postulate 3D representations in the brain in a variety of different coordinate frames (e.g. 'head-centred', 'hand-centred' and 'world-based'). Recent advances in reinforcement learning demonstrate a quite different approach that may provide a more promising model for biological representations underlying spatial perception and navigation. In this paper, we focus on reinforcement learning methods that reward an agent for arriving at a target image without any attempt to build up a 3D 'map'. We test the ability of this type of representation to support geometrically consistent spatial tasks such as interpolating between learned locations using decoding of feature vectors. We introduce a hand-crafted representation that has, by design, a high degree of geometric consistency and demonstrate that, in this case, information about the persistence of features as the camera translates (e.g. distant features persist) can improve performance on the geometric tasks. These examples avoid Cartesian (in this case, 2D) representations of space. Non-Cartesian, learned representations provide an important stimulus in neuroscience to the search for alternatives to a 'cognitive map'.

Keywords: deep neural networks, 3D spatial representation, moving observer, navigation, view-based, parallax

*Corresponding author
Email address: a.glennerster@reading.ac.uk (Andrew Glennerster)

1. Introduction

The discovery of place cells, grid cells, heading direction cells, boundary vector cells and similar neurons in the mammalian hippocampus and surrounding cortex has been interpreted as evidence that the brain builds up an allocentric, world-based representation or 'map' of the environment and indicates the animals movement within it [1–4]. However, this interpretation is increasingly questioned and alternative models are proposed that do not involve a 'cognitive map' [5–7]. Computer vision and robotics provide a useful source of inspiration for models of spatial representation and navigation in animals because their performance can be tested. Until recently, the predominant computer vision model for 3D navigation has been simultaneous localisation and mapping (SLAM) where a 3D reconstruction of the scene and the agent's location within it are continually updated as new sensory information is received [8, 9] (and non-visual precursors of SLAM such as Chatila and Laumond [10]). Although there are many variations on this theme, the essence of SLAM is that a set of corresponding features in images taken from different vantage points are used to recover (i) the 3D structure of those points in the scene and (ii) the rotation and translation of the camera per frame, where scene structure and camera pose are all described in the same 3D coordinate frame.

However, since the advent of deep neural networks, there has been a move to try out a quite different approach to navigation, in which the agent is tasked with matching the input resulting from a particular camera pose (i.e. an image, not a 3D location) and rewarding, however sparsely, the actions that lead it on a path to that goal. Eventually, after many trials, the agent learns to take a sequence of actions ('turn left', 'turn right', 'go forward') that take it from the current image to the goal although it never builds a 'map' in the sense of a representation of the scene layout with an origin and coordinate frame. These networks are different from earlier attempts to model mammalian navigation that used information about the location of the agent gained from model place cells [11] or using idiothetic information from proprioceptive and vestibu-

lar inputs [12]. Instead, they rely on visual information alone to build up a representation of space and are, in that sense, directly comparable with SLAM models. The recent RL models also differ from early attempts to represent space using very simple visual inputs such as Franz et al. [13] where the input was a 1-D omnidirectional measurements of luminance values and the robot laid down a new 'snapshot' whenever the view differed significantly from its current stored snapshots, generating a topological graph of space as it went [10, 14]. For one thing, the rules for storing the feature vectors were quite different in these approaches, although in some ways they were forerunners of the modern RL approach. Also radically different are the inverse RL approaches that have been used to predict human movement in relation to obstacles and goals [15]. These fit human navigation data in a low dimensional space of control parameters that, while successful in explaining obstacle avoidance, do not relate to an allocentric space representation.

A classic reinforcement learning approach to navigation: Zhu et al

In 2017, Zhu et al. [16] showed that reinforcement learning could be applied successfully to a navigation task in which the agent was rewarded for arriving at a particular image (i.e. a given location and pose of the camera, although these 3D variables were not explicitly encoded in the input the agent received, only the current image and the goal image). It is one of the key papers in this emerging field of reinforcement learning (RL)-based *perceptual-goal-driven* navigation [17–23]. Zhu et al. [16] in particular was one of the first to show it is possible to construct an end-to-end architecture for visual-goal-driven navigation using a modern deep learning stack trained with RL. This was in contrast to more typical RL work that treats the task of navigation to particular positions of the world just as part of general, global, state-based reward function (e.g. all the work on taxi-world [24] and most other tasks based on minigrids, or even the more recent BabyAI [25]). We illustrate what the system has learned by relating the stored vectors in the representation to the agent's location and orientation in space. We show, in particular, that the contexts that the repre-

sentation recognises are heavily dependent on the agent's current goal. The fact that the agent's task is integrated into the representation of current and stored states is reminiscent of many results in biological representation of shape and space [6, 7, 26–28]. Since Zhu et al. [16], there have been a number of important developments in this type of approach. Mirowski et al. [29] adapted the method to cover much larger spatial regions using images from *Google StreetView*; Eslami et al. [30] have shown that behaviour one might have thought would require a 3D model (e.g. predicting a novel view from a novel location in a novel scene) can be learned by carrying out the same task in many similar scenes; and others have included an explicit coordinate frame in the stored memory Mirowski et al. [29], Chen et al. [31], Gupta et al. [32], Mirowski et al. [33], Kumar et al. [34] while Kanitscheider and Fiete [35] have built on the biologically-inspired (but allocentric, coordinate-based) RatSLAM model of Milford and Wyeth [36]. In contrast to these coordinate-based advances, progress since Zhu et al. [16] on pure image-based approaches to large-scale spatial representation for navigation has slowed, as the community has been primarily focused on improving the visual navigation testbeds [37]. Another paper that incorporates an explicit biological perspective in relation to navigation is Wayne et al. [38] who have shown the importance of storing 'predictions that are consistent with the probabilities of observed sensory sequences from the environment'. They use a Memory Based Predictor to do this and draw attention to the similarities between the MBP and some of the proposed functions of the hippocampus.

In this paper, we examine the feature vectors in the stored representation after learning in the Zhu et al. [16] study to explore the extent to which they reflect the spatial layout of the scene. We show that, although spatial information *is* present in the representation, sufficient to be decoded, the organisation of the feature vectors is dominated by other factors such as the goal and the orientation of the camera (as one might expect, given the inputs to the network) and that it is possible to use these feature vectors to carry out simple spatial tasks such as interpolating between two learned locations.

A hand-crafted alternative representation using relative visual directions

We compare the performance of this RL network to a representation of location that (i) avoids any explicit 3D coordinate frame (like the RL approach), (ii) represents the current sensory state as a high dimensional vector (like the RL approach) but (iii) unlike the RL approach, is built on information that is known to be important in biological vision. The visual system is much more sensitive to the spatial separation (relative visual direction) of points than it is to their absolute visual direction [39–44] and it has been suggested on the basis of psychophysical evidence [45] that the visual system uses a reference frame for egocentric visual direction that is built from the relative visual direction of points and hence has no single 2D coordinate frame encompassing the sphere of visual directions [44–48]. This representation is very similar to a list of the saccades (magnitude and direction) that would take the eye from one point to another in the scene. Information about the 3D structure of the scene can be added to this representation by incorporating information about the *change* in the relative visual direction of points when the camera translates (motion parallax or binocular disparity). Glennerster et al. [44] showed how the pattern of eye movements that animals generally adopt, which is to fixate on a point as they move, is a distinct advantage for interpreting retinal flow if one assumes that the goal of the visual system is to update a representation of this sort. If animals fixate a point as they move, retinal motion provides information straightforwardly about changes in the relative visual direction of points with respect to the fixation point and the information can be used to build up a representation like the one we describe below. We call this a 'relative visual direction' representation (RVD) [7, 44, 47, 48]. In the simplistic implementation we describe here, the input is 1-dimensional and spans the entire 360^o field of view, whereas in practice the input would be 2-dimensional and the field of view would be limited so information would have to be gathered over successive fixations. The skeletal version used here nevertheless illustrates some key points about the information that is available in a representation that stores information in a relatively raw form, without building a 3D coordinate

frame. In particular, we show how motion parallax can be useful in separating out information in the representation that is likely to persist as the observer translates while other information is likely to go rapidly 'out of date'.

Comparison of feature vector models

We report on the performance of both types of model when faced with tasks that require basic spatial knowledge. The tasks we chose were interpolating between two locations or interpolating between two visual directions because these test whether the network contains information about novel locations or directions that it has not learned about during training. Bisection tasks have been carried out in humans [49–51] and are simpler to imitate than other tests of 'map-like' properties of spatial representation in humans such as a triangle completion task [52, 53].

The input to the two algorithms is utterly different (2D images of a naturalistic scene or a 1D image of synthetic points and the fields of view are quite different) and so it is not possible to make a fair comparison of their performance in these tasks. Instead, our aim is to show how, in principle, a representation that does not include a 3D coordinate frame (which is true of both models) could, nevertheless, contain useful information relating to the distance of features, rather like Marr's idea of a $2\frac{1}{2}$-D sketch [54] and to demonstrate how this information could be useful in the tasks we examine. The way forward for these non-3D representations is clearly to build on the success of RL demonstrations such as Zhu et al. [16], not simplistic handcrafted models, but, we argue, this development may be helped by considering ways to incorporate motion parallax information.

2. Methods

Our goal is to compare performance of two algorithms, one based on a learned representation, developed by Zhu et al. [16], and one based on a hand-crafted representation. To analyse these methods, we use two different tasks: the first is

to find the mid-point in space between two locations that have been learned (or are 'known') already; the second is to do the same in the orientation domain, i.e. to find the mid-bearing between two 'known' bearings. These tasks test for geometric consistency within a representation i.e., in this case, whether there is any implicit knowledge in the representation about locations or orientations other than the ones that have been learned about during training. We also probe the representations more directly, looking for systematic spatial organisation in the arrangement of the learned feature vectors when related to corresponding locations in space.

We begin with an account of the contribution Zhu et al. [16] make in the context of reinforcement learning and describe how decoding can be used to query the information stored in the network. We then describe our hand-crafted representation which records information about the angles between pairs of visible points and about the extent to which these change as the optic centre translates. It is hardly a surprise that this representation performs well on geometric tasks, and we are not making a claim that this representation is in any sense 'better' than the learnt one - the representations are, after all, utterly different. Nevertheless, it is informative to compare the performance of the representations side-by-side in order to inform the debate about improving learned representations in future in a way that incorporates information that is particularly important to animals.

2.1. Reinforcement Learning for Visual Navigation

Reinforcement learning (RL) [55] is a framework for optimizing and reasoning about sequential decision-making. Tasks are modelled as Markov Decision Processes (MDPs), $\langle S, A, T, R, \gamma \rangle$ tuples where S represents the state space, A the set of actions, $T : S \times A \times S \to [0, 1]$, R the reward function $R : S \times A \times S \to \mathbb{R}$, and $\gamma \in [0, 1)$ a discount factor. Solving an MDP is defined as finding a policy $\pi(a|s) = p(A = a|S = s)$ that maximizes the expected discounted cumulative return $\sum_{k=0}^{\infty} \gamma^k r_{k+1}$.

Deep reinforcement learning (DRL) is an extension of standard RL in which

the policy is approximated by a Deep Neural Network, and where RL algorithms are combined with stochastic gradient descent to optimise the parameters of the policy. Popular instances of DRL methods include: Deep Q-Network (DQN) [56] and its variants [57], which regress a state-action value function; policy gradient methods, which directly approximate the policy [58], and actor-critic methods [24, 59], which combine value-based methods with policy gradient algorithms to stabilize the training of these policies. DRL methods have been successful in solving complex tasks such as Go and other popular board games [60, 61], and have proved to be necessary to tackle decision-making tasks with high-dimensional or visual state representation [56, 62]. These breakthroughs in visual learning and control have also created a surge in work on *active vision* [63], and several *visual-based navigation* [37] frameworks have recently been proposed to formalize and tackle many 3D navigation tasks.

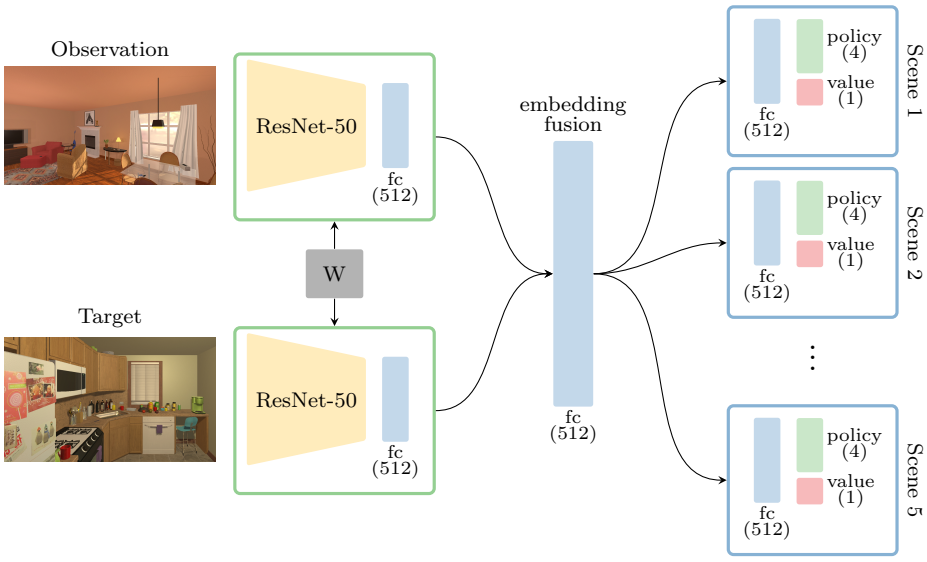


Figure 1: ***Architecture of the Zhu et al. [16] siamese network.*** *See text for details.*

We focus here on the task of goal-driven visual navigation, where the agent is asked to navigate to an entity in a high-fidelity 3D environment, given either an

image of the entity, a natural language description, some coordinates, or other relevant information. As we set out in the Introduction, the case we have chosen to analyse is the one proposed by Zhu et al. [16], which aims to solve the problem of learning a policy conditioned on both the target image and the current observation. The architecture is composed as follows: the observation and target images are generated using an agent in a virtual environment, AI2-THOR [64]. First, these images are passed separately through a set of siamese layers (which means that the parameters in the twinned networks are identical, despite the input to the two networks being different) [65]. These are based on a pretrained ResNet-50 network and have a feedforward layer, embedding these images into the same embedding space. These embeddings are then concatenated and further passed through a fusion layer, which outputs a joint representation of the state. The joint representation is finally sent to *scene-specific* feedforward layers, which produce a policy output and a value as required by a standard actor-critic model (see Fig. 1). This split architecture allows for the embedding layers to focus on providing a consistent representation of the MDP instance based on the goal and the agent's observation, while providing capacity to the network to create separate feature filters that can condition on specific scene features such as map layouts, object arrangement, lighting, and visual textures, thus obtaining the capability to arbitrarily generalize across many different scenes.

2.2. Knowledge Decoder

It is not possible to tell from the architecture described in the previous section whether any of the environment properties that are available in a 'cognitive map' (e.g., location / orientation of the target, agent position, angles to the target) are present in the transformations encoded in the network's weights. To test whether information about location and orientation is encoded, we trained a decoder which takes the agent's internal representation as input and outputs one of the desired properties, such as (x, y) coordinates of a chosen observation. More specifically, to build the dataset we used Zhu et al. [16]'s architecture as described above. This generates an embedding up to the final feedforward layer

(before it gets sent into the policy and value heads) for each target-observation pair of the training set, while also recording the agent's (x, y) coordinates and angle θ. We primarily employ multi-layer perceptrons (MLPs) to perform this decoding. MLPs characterise flexible non-linear functions, and are constructed by interleaving linear transformations with non-linear activations/transformations (e.g. ReLU, TanH, . . .). The decoder is a 2-layer MLP in the case of a single value regressor (i.e., the angle), or a 3-layer MLP with multiple "heads"—additional MLPs to split common computation—when regressing to (x, y) coordinates or to the orientation, θ, of the agent. We use an MSE loss trained with Adam [66], together with dropout (see Table A1 for hyperparameters).

2.3. Relative Visual Direction (RVD) representation

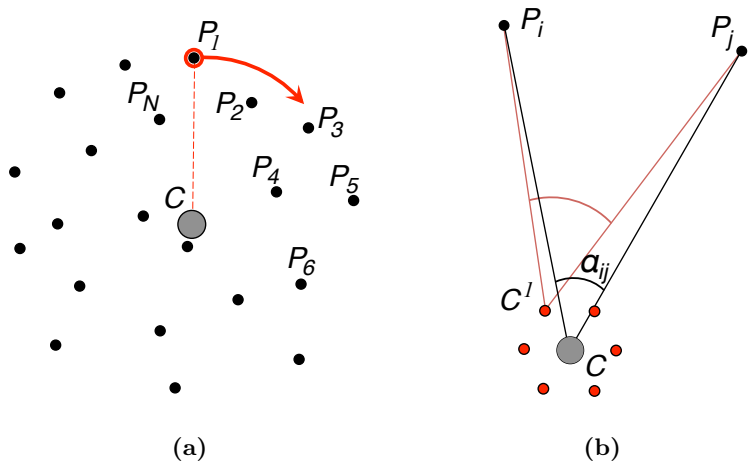


Figure 2: *. A) 2D scene containing N random points and camera C in the centre. The points are ordered clockwise in angular sense with respect to the reference point P_1, which is marked red. B) Angular and parallax features. P_i and P_j are scene points, C is camera location, $C^1 - C^6$ are sub-cameras.*

This section describes a simple representation of the angles between pairs of points around the observer. It is not learned, like the Zhu et al. [16] representation; it is hand-crafted and it contains all the information that would be required to reconstruct the 3D structure of the scene. However, it does not do

that. Instead, it keeps the information in a relatively raw state so that current and stored states can be compared in a high dimensional space, just as they are in the Zhu et al. [16] representation. As discussed in the Introduction, information about relative visual directions and changes in relative visual direction are important in biological vision and are key to this representation. Figure 2a shows a 2D scene containing an optic centre C, surrounded by N random points $P_1, \ldots, P_N$. The points in the scene are numbered and ordered clockwise with respect to the first point P_1, marked in red (this is relevant for the mid-bearing task described later). The angle subtended by a pair of points (P_i, P_j) at the optic center C, indicating the relative visual direction, is denoted $\alpha_{ij} = \angle P_i C P_j$ (such that $\alpha_{ji} = \angle P_j C P_i = 2\pi - \alpha_{ij}$). The vector of all such angles, between every possible pair of points P_i and P_j viewed from the optic center C is denoted $\boldsymbol{\varepsilon}$ (Fig. 2b). We assume an omnidirectional view with no occlusions. The dimensionality of $\boldsymbol{\varepsilon}$ is thus $M = N^2 - N$, since we exclude angles between a point and itself. The elements of $\boldsymbol{\varepsilon}$ are ordered in a particular way, following

$$\boldsymbol{\varepsilon} = \{\alpha_{ij} : i = 1, \ldots, N, j = (i+1), \ldots, N, 1, \ldots, (i-1)\}. \tag{1}$$

The reason $\boldsymbol{\varepsilon}$ is ordered in such a manner is to assist in extracting subsets of elements when the task relates to visual direction (Section 2.5). However, elements of $\boldsymbol{\varepsilon}$ can be indexed in other ways, as the next section shows.

2.4. Mid-point for translation of the camera

Although $\boldsymbol{\varepsilon}$ contains all possible angular features, for certain tasks such as interpolating between locations some angular features are more informative than others. In particular, angular features that arise from pairs of distant points are more stable (i.e. vary less) during translation of the optic centre and thus are more useful for the interpolation task than are the angles between nearby points since these vary rapidly with optic centre translation. First, we extract a subset of the elements of $\boldsymbol{\varepsilon}$ using a criterion based on parallax information. We define a measure of parallax that assumes we have access to more views of the scene, as if the camera has moved by a small amount as shown in Fig. 2b.

For such individual 'sub-cameras' C^k, $k = 1, \ldots, n_C$, where n_C is the number of sub-cameras, we can construct angular feature vectors $\boldsymbol{\varepsilon}_{C^k}$ similar to that constructed at the optic centre, $\boldsymbol{\varepsilon}$, with exactly the same ordering of elements. A 'mean parallax vector', $\boldsymbol{\psi}$, can then be computed from the difference between these sub-camera views, C^k, and the original view at C.

$$\boldsymbol{\psi} = \{\psi_n\}_{n=1}^{N} = \frac{1}{n_C} \sum_{k=1}^{n_C} \frac{\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}_{C^k}}{\boldsymbol{\varepsilon}} \tag{2}$$

Since $\boldsymbol{\psi}$ has the same ordering of elements as $\boldsymbol{\varepsilon}$, each element of $\boldsymbol{\psi}$ contains a parallax-related measure referring to that particular pair of points.

It will prove useful to identify the pairs of points that are more distant, using the observation that the parallax values recorded in $\boldsymbol{\psi}$ are small in these cases. For a particular threshold value T_ψ on parallax, we define $\boldsymbol{\rho}$ as the *mask* on $\boldsymbol{\psi}$, such that $\rho_i = 1, \text{if}\, \psi_i \leq T_\psi$, to identify the subset of $\boldsymbol{\varepsilon}$ with relatively small parallax values as $\boldsymbol{\varepsilon} \odot \boldsymbol{\rho}$. These elements of $\boldsymbol{\varepsilon}$ are, by design, those that are likely to change relatively slowly as the camera moves over larger distances.

2.5. Mid-bearing for rotation of the camera

We now consider a task of interpolating between camera *bearing* (viewing direction), rather than location. The goal is to estimate a bearing that is half way between two given views of the camera. A view, $\boldsymbol{\vartheta}^{\theta,\omega}$, in this context, involves both a bearing, θ, and an angular range, ω, specifying the field of view for that camera (here, taken to be a fixed value of 90^o) and is defined as a list of all the elements of $\boldsymbol{\varepsilon}$ that appear within that field of view. Note that the goal here is closely related, but not identical, to the task in the previous section of picking out an entire view that is half-way between two given views captured from different locations. The way $\boldsymbol{\varepsilon}$ is organised, such that elements are ordered by reference point (see Eqn. 1), means that there is a consistent (albeit approximate) relationship between the index of the element and the bearing of the reference point for that element.

To consider all the elements of $\boldsymbol{\varepsilon}$ that appear in a given view we construct a mask, $\boldsymbol{\kappa}$, similar to $\boldsymbol{\rho}$ above, but now the mask is based on whether both scene

points P_i and P_k that define an element in $\boldsymbol{\varepsilon}$ are visible in a particular view: $\boldsymbol{\kappa}^{\theta,\omega} = \{\kappa_j\}_{j=1}^{N}$, where $\kappa_j = 1, \text{if } P_i, P_k \in \boldsymbol{\vartheta}^{\theta,\omega}, \text{where } \boldsymbol{\varepsilon}^j = \alpha_{ik} = \angle P_i C P_k$. The relevant elements are denoted $\boldsymbol{\varepsilon} \odot \boldsymbol{\kappa}^{\theta,\omega}$. Given two such views $\boldsymbol{\vartheta}_i^{\theta_i,\omega}$ and $\boldsymbol{\vartheta}_j^{\theta_j,\omega}$, we can use the indices of the elements in each view to estimate the indices of the view that is mid-way between the two (Section 3.3).

3. Results

Figures 3 to 5 show the results for three comparisons between the models. Fig. 3 relates physical distance between locations to the separation of corresponding feature vectors in the representation. Fig. 4 illustrates the ability of both models to interpolate correctly between the representation of two learned/known locations while Fig. 4 does the same for interpolation between two learned/known visual directions.

3.1. Correlation between physical separation and feature separation in the representation.

Figure 3 compares the representation of a scene in the two models we have discussed, based on Zhu et al. [16] (left hand column) or relative visual direction (RVD, right hand column). Figure 3a shows a plan view of the scenes used by Zhu et al. [16] (where filled and closed symbols show the camera locations at test and training) and Fig. 3b shows the 2D layout of scene points (black dots) and camera locations (coloured points) in a synthetic 2D scene that was used as input for the RVD method. In Zhu et al. [16], the environment was a highly realistic 3D scene in which the agent was allowed to make 0.5m steps and turn by 0, +90 or -90 degrees (figures are from the Bathroom scene, see Appendix for others). Target views are marked by blue stars and arrows. For the RVD method, we generated a random 2D scene with 100 points. Cameras were placed in the middle of the scene as a regular 50×50 grid, which occupied 1/5 of the scene (Fig. 3b). The colour indicates the distance of a camera from the central reference camera.

For each learned context in Zhu et al. [16] (where a learned context is defined by an observation location, a camera orientation and a target), there is a corresponding feature vector (i.e. 20 feature vectors per location). These observation locations are the 'trained' locations illustrated by open circles in Fig. 3. Figure 3c shows the Euclidean distance between pairs of feature vectors ($\mathbb{R}^{512}$) from the test set, for all possible pairings, and plots this distance against the distance between the corresponding observation locations ($\mathbb{R}^2$). Figure 3c shows that there is only a weak correlation between distance in the embedding space and physical distance between observation locations for this scene in the Zhu et al. [16] paper (Pearson correlation coefficient, R, is 0.09, see Fig. A3 for other scenes) whereas Fig. 3d shows that, for the RVD method, there is a clear positive correlation ($R = 0.99$). Zhu et al. [16] quoted a correlation of 0.62 between feature vector separation and separation in room space, but we are only able to reproduce a similarly high correlation by considering the distance between pairs of feature vectors when the agent had the *same* goal and the *same* viewing direction ($R = 0.67$ for all such pairings in the Bathroom scene). By contrast, Fig. 3c refers to all possible pairings in the test phase.

The right hand column of Fig. 3 shows results of the 'relative visual direction' (RVD) model. At each camera camera location ($N = 2500$), we generated a truncated angular feature vector $\boldsymbol{\varepsilon} \odot \boldsymbol{\rho}$ (see Section 2.4) as a representation of the scene as viewed from that location. We used the 30^{th} percentile of the parallax values as a threshold for inclusion of elements (T_ψ), i.e. the truncated feature vectors contained only the elements of $\boldsymbol{\varepsilon}$ that corresponded to pairs of points with the smallest parallax values, where 'small' in this case means the bottom 30% when ordered by parallax magnitude. The exact choice of threshold is not important; in the Appendix, Fig. A1, we show the same result for different values of this threshold. Using the *top* 30% of $\boldsymbol{\varepsilon}$ when ordered by parallax, or using the entire $\boldsymbol{\varepsilon}$ vector, gives rise to worse performance on the interpolation task. Note that we have used the same ordering of elements in $\boldsymbol{\varepsilon} \odot \boldsymbol{\rho}$ for all cameras. Specifically, the ordering of $\boldsymbol{\varepsilon}$ and $\boldsymbol{\rho}$ were established for the central reference camera and applied to all other cameras (see Eqn. 1).

Figures 3e and 3f visualise the embedding space for the Zhu et al. [16] and RVD representations respectively using a t-SNE projection [67]. This projection attempts to preserve ordinal information about the Euclidean distance between high dimensional vectors when they are projected into 2-D. In Zhu et al. [16] (Fig. 3e), feature vectors are clumped together in the t-SNE plot according to the agent's target image. Targets 4 and 5 were very similar images, so it is understandable that the feature vectors for locations with these targets are mixed (yellow and orange points). Although target is the dominant determinant of feature vector clustering, information about camera orientation and camera location is still evident in the t-SNE plot. The top-right sub-panel colour-codes the same T4/T5 cluster but now according to the orientation of the camera: this shows that orientation also separates out very clearly. Finally, there is also information in the t-SNE plot about camera location. Colours in the bottom right subplot indicate distance of the camera location from a reference point, (0,0); there is a gradation of colours along strips of a common camera orientation and this systematic pattern helps to explain why camera location can be decoded (see Section 3.2). For the RVD method, the configuration of feature vectors preserves the structural regularity of the camera positions, as can be seen from the t-SNE projection in Fig. 3f. We now explore how these differences affect the ability of each representation to support interpolation between learned/stored locations.

3.2. Interpolation between stored locations in the representation.

Figure 4 shows the results of the location interpolation task which was to estimate the mid-point between two locations (e.g. in Fig. 4a O_{mid} is halfway between O_1 and O_2) based on the midpoint between two feature vectors. For the Zhu et al. [16] model, this requires a decoder for 2-D position learned from the stored feature vectors (see Section 2.2). The results are shown in Fig. 4c using a normalized scale to illustrate the errors relative to the two input locations.

For the RVD model, decoding is much more direct, as one would expect from the t-SNE plot (Fig. 3f). The details are as follows. Figure 4b shows a

random 2D scene with a 6×6 grid of cameras in the middle. For each camera C_j, $j = 1, \ldots, 36$, we calculated a feature vector $\boldsymbol{\varepsilon}_{C_j}$ and a parallax mask $\boldsymbol{\rho}_{C_j}$ as described in Section 2.4. The feature vector for the mid-point between two cameras C_i and C_j was computed as $\boldsymbol{\varepsilon}_{C_i,C_j} = \frac{1}{2}\Big((\boldsymbol{\varepsilon}_{C_i} \odot \boldsymbol{\rho}_{C_i}) + (\boldsymbol{\varepsilon}_{C_j} \odot \boldsymbol{\rho}_{C_j})\Big)$. Then, to find the midpoint, we searched over a fine regular grid (step = 1) of camera locations to find the camera $C_k^\star$ that was best matched with the estimated feature $\boldsymbol{\varepsilon}_{C_i,C_j}$, that is,

$$C_k^\star = \arg\min_{c_k} \|\boldsymbol{\varepsilon}_{C_i,C_j} - \boldsymbol{\varepsilon}_{C_k}\| \tag{3}$$

This is equivalent to, but simpler than, the decoding stage using a MLPs for the Zhu et al. [16] model. Figure 4d shows estimated mid-points calculated this way for all possible pairs of the 36 cameras (n=630). For the Zhu et al. [16] method, Fig. 4e shows the absolute errors relative to the true mid-point between O_1 and O_2 as a function of the separation between O_1 and O_2. Figure 4f shows the same for the RVD method. As discussed in the Introduction, it is not fair to make a direct comparison between the magnitude of the errors for the two models given how different their inputs are but one can compare the way that the errors change with separation between O_1 and O_2. This shows a monotonic rise for the RVD model, as one would expect from a geometric representation, whereas this is not true for the Zhu et al. [16] method (Fig. 4e).

3.3. Interpolation between stored viewing orientations in the representation.

Figure 5a shows the scene layout from Zhu et al. [16] and two views from a single location. The goal in this case is to find an intermediate *bearing* (as shown by the black arrow) half way between the bearing of the two reference images (orange and purple arrows). Figure 5c shows the error in the decoded mid-bearing when the input images are taken from views that are 0, 90 or 180^o apart. Note that the two input images need not necessarily be taken from the same location in the room (either in training the decoder or in recovering a mid-bearing). Figures 5c and 5e show that there is no systematic bias to the mid-bearing errors but the spread of errors is large compared to that for the

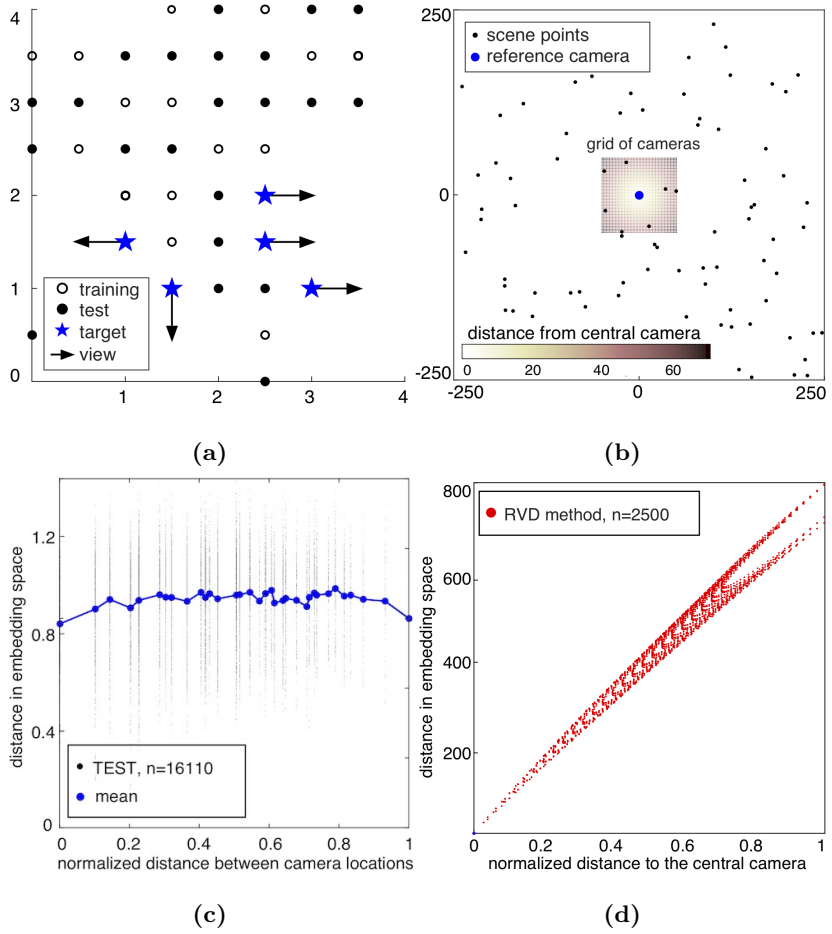
'relative visual direction' (RVD) method (Fig. 5f). The RVD method uses a very simple algorithm to estimate the mean bearing. It assumes that the ordering of elements in $\boldsymbol{\varepsilon}$ has a linear relationship to the bearing of a view, i.e. that as the bearing changes (going from orange view to purple view in Fig. 5d) the index of the corresponding elements in $\boldsymbol{\varepsilon}$ will change systematically and hence the mean index of the elements within a view is useful in determining the bearing of that view. This is not strictly true, but the fact that it is a useful approximation is because of the way that the vector, $\boldsymbol{\varepsilon}$, was set up in the first place (Eqn. 1). In more detail, Figs. 5b and 5d shows how the bearing of a mid-view (θ_{mid}) is estimated using the over-simplified assumption that the bearing of the reference point in a pair of views varies linearly with index in $\boldsymbol{\varepsilon}$. In fact, of course, the relationship between bearing and element index depends on the layout of the scene. The mean index of a view, $\boldsymbol{\vartheta}^{\theta,\omega}$, is computed from its corresponding mask, $\boldsymbol{\kappa}^{\theta,\omega}$, as the middle index, μ, of all 'on' mask elements, $\kappa_j = 1$, for that view. Given two views $\boldsymbol{\vartheta}_i^{\theta_i,\omega}$ and $\boldsymbol{\vartheta}_j^{\theta_j,\omega}$, we estimate a nominal bearing of the mid-view image, μ_{mid}, from the average of their mean indices:

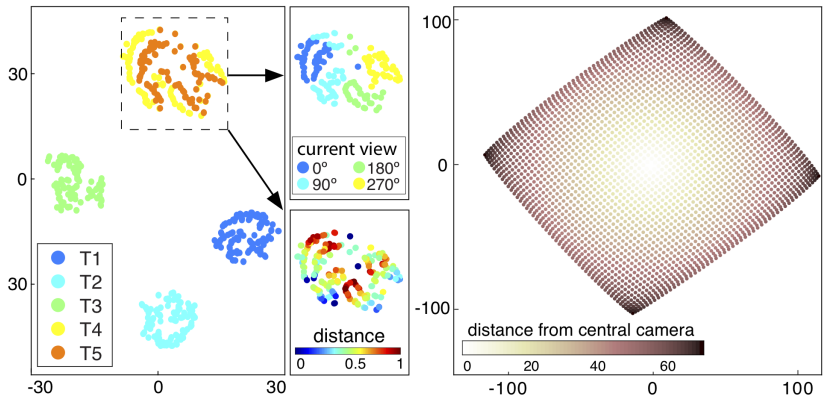
$$\mu_{mid} = (\mu_i + \mu_j)/2. \tag{4}$$

and $\theta_{mid} \propto \mu_{mid}$.

This heuristic is illustrated in Fig. 5d. For the purposes of illustration only, this shows the i^{th} element in the orange image (pair of dots outlined in orange) and the i^{th} pair in the purple image (outlined in purple). Considering the indices of these two elements in $\boldsymbol{\varepsilon}$, the rounded mean of these two indices gives an index to an element of $\boldsymbol{\varepsilon}$, i.e. it corresponds to a pair of points. For the purposes of illustration, these are shown by the black squares in Fig. 5d which, in this case, happen to lie close to the mid-bearing direction. However, the heuristic simply reports the estimated orientation of the mid-view as described above (Equation (4)). The bias and variability of the estimates of the mid-view in both models are shown in Fig. 5e and Fig. 5f respectively. Again, given the very different nature of the inputs to the two models, it is not fair to comment

on the relative magnitude of errors in the two models. Neither model shows the Weber's law increase in errors with angular separation between μ_1 and μ_2 that we saw in Fig. 4f.





(e) **(f)**

Figure 3: ***Relationship between scene location and feature vectors for Zhu et al. [16] and the relative visual direction (RVD) method.*** ***a)*** *shows a plan view of the Bathroom scene in Zhu et al. [16]. Open circles show the camera locations for images used in the training set, closed circles show the locations used in the test set. Blue stars and black arrows show the location and viewing direction of the camera for the target images.* ***b)*** *An example of a random 2D scene with N=100 points used in the RVD model. Cameras are placed in the middle of the scene as a 50×50 grid, which is 1/5 of the scene. The colour of each camera location indicates the distance of the camera from the central camera, C. For each of the 2500 camera locations we calculated a vector, $\boldsymbol{\varepsilon}$, describing the angle between pairs of scene points as viewed from that camera (see Methods).* ***c)*** *For the Zhu et al. [16] method, the Euclidean distance in $\mathbb{R}^{512}$ between pairs of embedded feature vectors is plotted against the separation between the corresponding pairs of camera locations in the scene.* ***d)*** *For the 'relative visual direction' (RVD) method, the Euclidean distance between the feature vectors for each camera and the feature vector for the central camera, C, is plotted against the separation between the corresponding pairs of camera locations in the scene.* ***e)*** *A t-SNE plot that projects the stored feature vectors in the Zhu et al. [16] network into 2D (see text for details).* ***f)*** *Same as e) but now for the RVD model.*

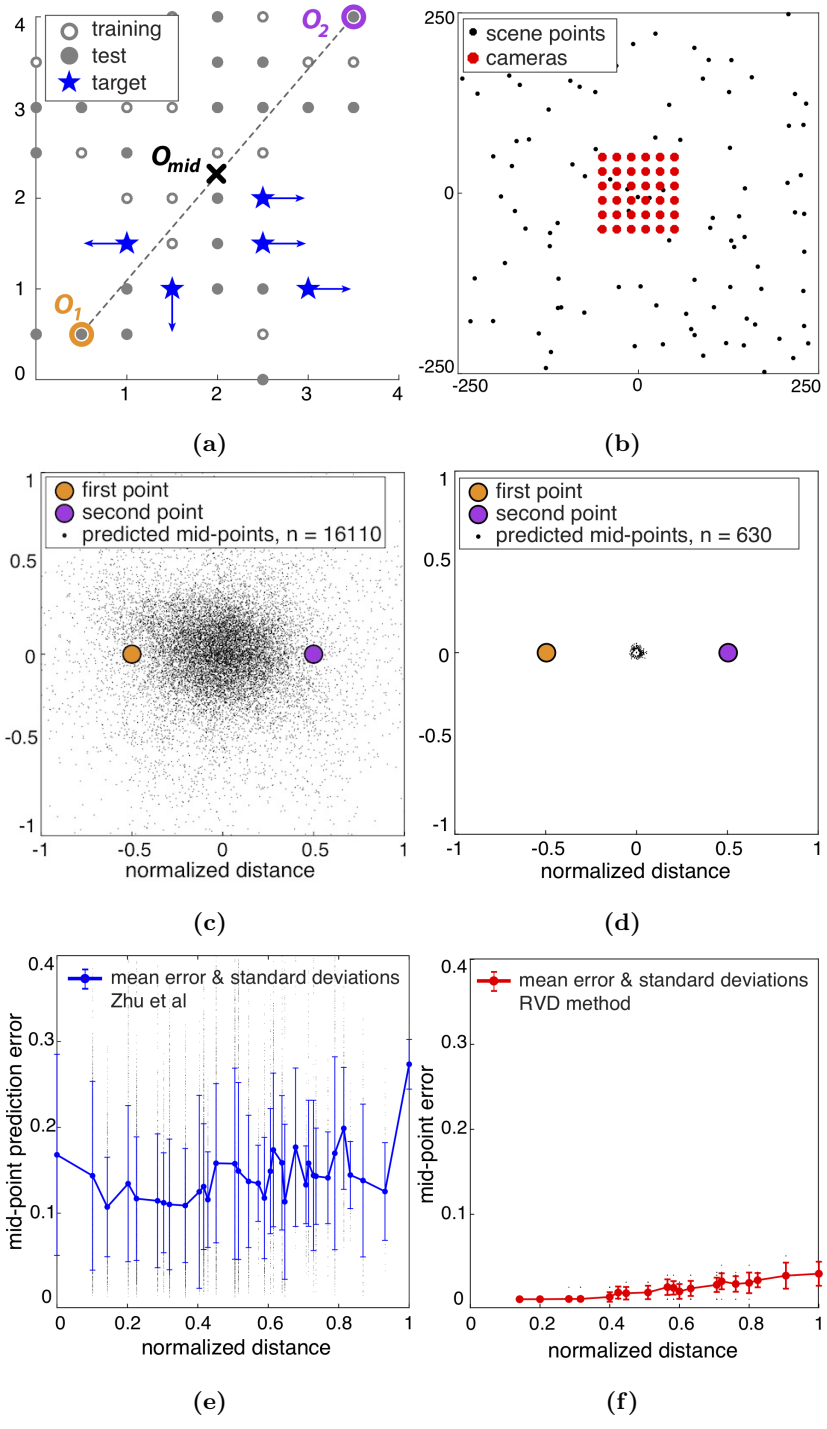

training
test
target
O_1
O_2
O_{mid}
scene points
cameras
first point
second point
predicted mid-points, n = 16110
normalized distance
(a)
(b)
first point
second point
predicted mid-points, n = 630
normalized distance
(c)
(d)
mean error & standard deviations
Zhu et al
mid-point prediction error
normalized distance
mean error & standard deviations
RVD method
mid-point error
normalized distance
(e)
(f)

Figure 4: ***Estimate of midpoints between pairs of observation locations. a)*** *shows the Bathroom scene with two observation locations, O_1 and O_2, and a midpoint, O_{mid}.* ***b)*** *shows a random 2D scene with a 6×6 grid of cameras in the middle. For each camera, we calculated a feature vector $\boldsymbol{\varepsilon} \odot \boldsymbol{\rho}$ (see Section 2.4).* ***c)*** *shows the estimated midpoints for all possible pairs of observations (where an observation is defined as a location, orientation and target), using Zhu et al. [16] feature vectors and decoding (see Section 2.2). Orange and purple circles show the normalised location of the two observation locations and the black dots show, in this normalised coordinate frame, the location of the estimated midpoints.* ***d)*** *shows the same as c) but for the feature vectors in the RVD model. The black dots show midpoints for all possible pairs of camera locations.* ***e)*** *shows the midpoint prediction error from c) (absolute errors) plotted against the separation of the observation locations (O_1 and O_2). The separation between observation locations is normalised by the maximum possible location of two observation locations in the room. Error bars show one standard deviation.* ***f)*** *shows the same for the RVD method. We considered all possible pairs of cameras (n=630).*

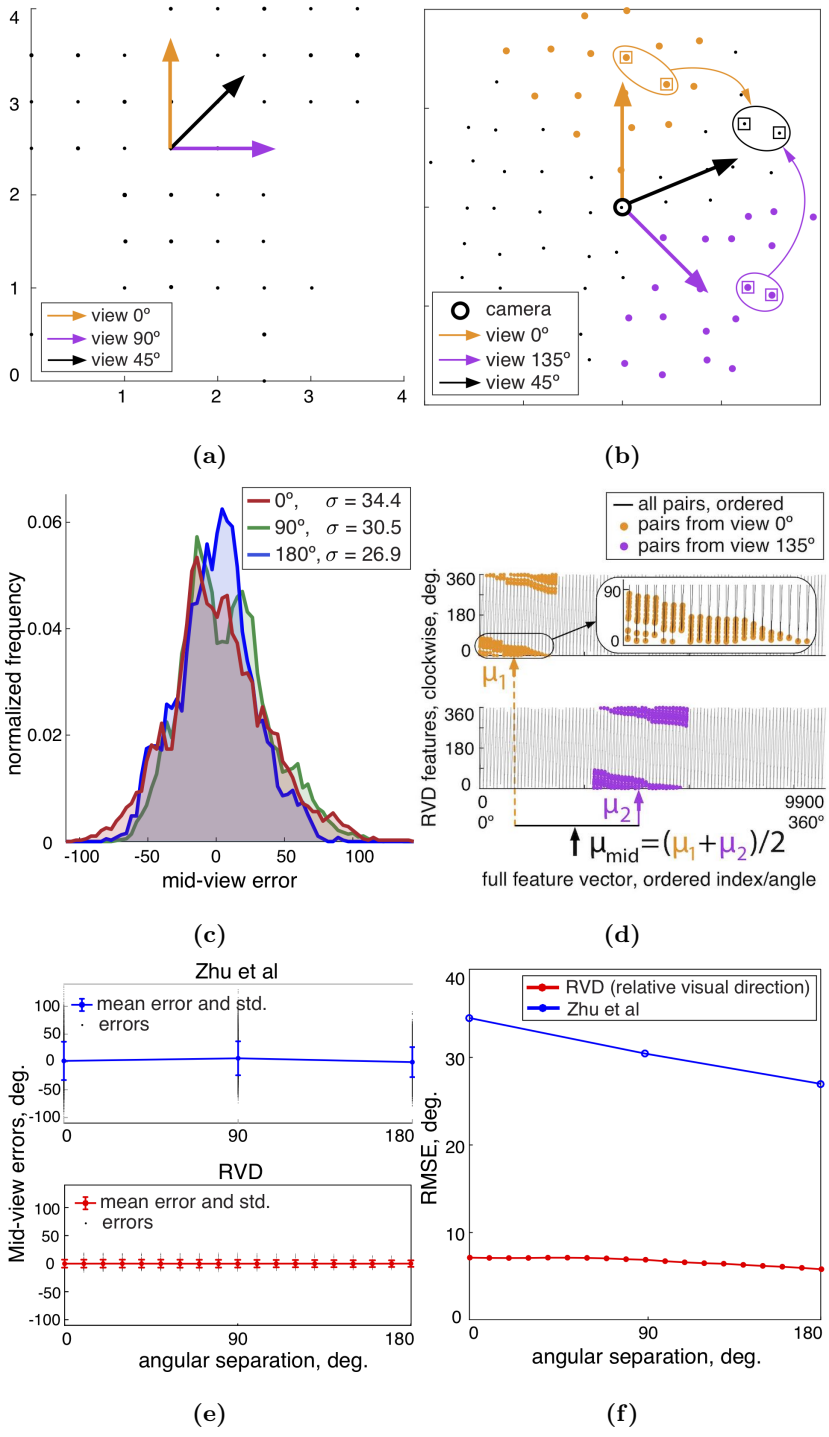

view 0°
view 90°
view 45°
camera
view 0°
view 135°
view 45°
(a)
(b)
0°, σ = 34.4
90°, σ = 30.5
180°, σ = 26.9
normalized frequency
mid-view error
(c)
all pairs, ordered
pairs from view 0°
pairs from view 135°
RVD features, clockwise, deg.
μ1
μ2
μmid=(μ1+μ2)/2
full feature vector, ordered index/angle
(d)
Zhu et al
mean error and std.
errors
RVD
Mid-view errors, deg.
angular separation, deg.
(e)
RVD (relative visual direction)
Zhu et al
RMSE, deg.
angular separation, deg.
(f)

Figure 5: ***Estimate of new views at an orientation half way between learned views.*** ***a)*** *shows a plan view of a bathroom scene in Zhu et al. [16] and the 45 locations the camera could occupy. Orange and purple arrows indicate two camera orientations and the black arrow indicates an orientation halfway between these (not used in Zhu et al. [16]).* ***b)*** *Similar to a) but for the RVD method. Points visible in views* 0° *(north) and* 135° *(south-east) are marked as orange and purple circles, where the field of view* (ω) *is limited to* 90°*. The ground-truth mid-view is indicated by the black arrow (see text).* ***c)*** *Distribution of errors in computing the mid-view orientation from a decoding of orientation in the Zhu et al. [16] trained network. Red, green and blue distributions are for camera orientations separated by 0, 90 and 180 degrees respectively.* ***d)*** *Full vector of angular features,* $\boldsymbol{\varepsilon}$*, (black saw-tooth plot). The y-axis shows the magnitude the elements in* $\boldsymbol{\varepsilon}$*, i.e. the angle between pairs of points. The x-axis represents indices of the vector's elements (9900 in this case) see Eqn. 1. The x-axis also provides an approximate indication of visual direction, from* 0° *to* 360°*, see text. The elements that correspond to pairs of points visible in the north and south-east views are marked with orange and purple circles respectively (see inset). Mid-indices* μ_1 *and* μ_2 *are marked as orange and purple arrows, while the index of the predicted mid-view* μ_{mid} *is marked as a black arrow.* ***e)*** *All the mid-view errors for the Zhu et al. [16] method for camera orientations separated by 0, 90 and 180 degrees. Mean and standard error shown in blue. Plot below shows the same for the RVD method. Mean and standard error shown in red.* ***f)*** *Shows the RMSE error of predicted mid-view with respect to the ground truth as a function of angular separation between the views. For the RVD method we considered views separated by many different angles (in increments of* 10°*), while for Zhu et al. [16] the data limited analysis to only three separations.*

4. Discussion

There has been a long-standing assumption that the brain generates spatial representations from visual input and does so in a variety of 3D coordinate frames including eye-centred (V1), ego-centred (parietal cortex) or world-centred (hippocampus and parahippocampal gyrus). Computer vision and robotics research has also concentrated on algorithms that generate representations in 3D frame (a world-based one). Biological models have not tried to recapitulate the complexities of photogrammetry (computing 3D structure from images) but instead have generally assumed that the generation of a 'cognitve map' relies on other inputs such as proprioceptive signals or pre-existing place cell or grid cell input, to provide spatial structure to the representation [11, 68–70].

We have chosen to examine in detail the RL method described by Zhu et al. [16] for learning to navigate to an image using visual inputs alone, because this has now become a general method on which several more recent and complex algorithms have been based [29, 31–34, 71]. We have compared the Zhu et al. [16] representation to a hand-crafted representation (based on relative visual directions and using highly simplistic input) in order to illustrate two points. First, in Zhu et al. [16], the relationship between stored feature vectors and the locations of the camera in the scene (Fig. 3a) is quite a complex one, while for the RVD model the relationship is simple and transparent. In the case of Zhu et al. [16], it is possible to build a decoder to describe the mapping between feature vectors and location (as illustrated by the systematic distance information visible in Fig. 3e) but this is quite different from the smooth, one-to-one relationship between stored feature vectors and space illustrated in Fig. 3f, at least over the range of camera locations illustrated here (Fig. 3b). The decoding required to extract location from the Zhu et al. [16] representation is reminiscent of the decoding that has been described as a way to use the aliased grid cell activity as a signal for location in rats [68], i.e. substantially more complex than the than interpolation of the feature vectors of the RVD model which generates a sensible result directly (e.g. Fig. 4d). Like the decoding of

location in the Zhu et al. [16] model, interpreting the output of grid cells would need a sophisticated decoding mechanism if they were to be used on their own for navigation [68] and neural network implementations have been proposed to solve this problem. For example, it is possible to decode the distance and direction of a goal given high dimensional vectors ($\mathbb{R}^{512}$) of grid cell activity at the current and goal locations[69] but grid cell firing rates are not the only high dimensional vectors encoding spatial location that could be used. The vector $\boldsymbol{\varepsilon}$ that we have described in this paper would be likely to do equally well and potentially even better since the aliased nature of grid cell firing is a disadvantage rather than an advantage in this context.

Figure 6: ***Visual servo-ing to maintain postural stability.*** *Looking straight out on the mountains, almost all motion parallax is removed because the scene is distant and so cannot drive postural reflexes. In a normal scene, there are objects visible at a range of distances, giving rise to both large and small magnitudes of motion parallax. Removal of close objects in this scene has the same effect as setting T_ψ to mask out all but the lowest parallax elements of $\boldsymbol{\varepsilon}$ in the RVD representation. This is one example, in addition to the two examined in Figs. 4 and 5, where indexing different elements of $\boldsymbol{\varepsilon}$ and monitoring changes in those elements is helpful for accomplishing a task. License to use Creative Commons Zero - CC0.*

Answering the question 'where am I?' does not necessarily imply a coordinate frame [6, 7, 47, 72, 73]. Instead, one can offer a restricted set of alternative

hypotheses. These potential answers to the question may correspond to widely separated locations in space, in which case the catchment area of each hypothesis is large, but the answer can be refined by adding more alternatives (i.e. more specific hypotheses about where the agent is). This makes the representation of space hierarchical [74–76] and compositional in the following sense. Consider the RVD representation of a scene that includes distant objects such as the stars or the mountains in Fig. 6. The angles between these (which are elements of ε) do not change, however much the observer moves. If the objects are stars, then the catchment area of the hypothesis covers the whole Earth. Adding in objects that are nearer than the mountains refines the catchment area and this can be done progressively, providing a more and more accurate estimate of the location of the observer (hence, the representation is compositional) as elements with higher parallax are added to ε. This provides a hierarchy of hypotheses about location, from coarse to fine, without generating a 3D coordinate frame.

Conclusion

Biological models of spatial representation have often assumed that the brain builds a of the world using allocentric (world-based) or ego-centric 3-dimensional coordinate frame. The representations we have examined here are different in that they store high dimensional vectors describing the sensory information (and, in the case of Zhu et al. [16], also the agent's goal) at each location. Given that this type of representation is being used increasingly in deep reinforcement learning implementations of agents that are capable of predicting novel views of scene, route-following and taking short-cuts [30, 33, 69], this type of model is an important existence proof that there are alternatives to 3-dimensional coordinate frame hypotheses of spatial representation. We have shown here how, in developing high dimensional features to represent images, it can be advantageous to introduce information about the distance of features and, especially, to identify elements of the input that are likely to be long- or short-lived in the scene as the camera translates.

Acknowledgements

We are grateful to Abhinav Gupta for providing code and advice and to Aidas Kilda and Andrew Gambardella for their help. This research was supported by EPSRC/Dstl grant EP/N019423/1 (AG). PHST, NS, & NN were supported by EPSRC/MURI grant EP/N019474/1. PHST was additionally supported by ERC grant ERC-2012-AdG 321162-HELIOS and EPSRC grant Seebibyte EP/M013774/1, and would also like to acknowledge the Royal Academy of Engineering and FiveAI.

[1] J. O'Keefe, J. Dostrovsky, The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat, Brain Research 34 (1971) 171–175.

[2] S. Taube, U. Muller, B. Ranck, Head-direction cells recorded from the postsubiculum in freely moving rats. I. Description and quantitative analysis., The Journal of Neuroscience 10 (2) (1990) 420–435.

[3] Microstructure of a spatial map in the entorhinal cortex., Nature 436 (7052) (2005) 801–6, URL http://www.ncbi.nlm.nih.gov/pubmed/15965463.

[4] C. Lever, S. Burton, A. Jeewajee, J. O'Keefe, N. Burgess, Boundary vector cells in the subiculum of the hippocampal formation, Journal of Neuroscience 29 (31) (2009) 9771–9777.

[5] L. Acharya, Z. M. Aghajan, C. Vuong, J. J. Moore, M. R. Mehta, Causal influence of visual cues on hippocampal directional selectivity, Cell 164 (1-2) (2016) 197–207.

[6] W. H. Warren, Non-Euclidean navigation, Journal of Experimental Biology 222 (Suppl 1) (2019) jeb187971.

[7] A. Glennerster, A moving observer in a three-dimensional world, Phil. Trans. R. Soc. B 371 (1697) (2016) 20150265.

[8] A. J. Davison, Real-time simultaneous localisation and mapping with a single camera, in: ICCV, 1403–1410, 2003.

[9] J. Fuentes-Pacheco, J. Ruiz-Ascencio, J. M. Rendón-Mancha, Visual simultaneous localization and mapping: a survey, Artificial Intelligence Review 43 (1) (2015) 55–81.

[10] R. Chatila, J.-P. Laumond, Position referencing and consistent world modeling for mobile robots, in: Proceedings. 1985 IEEE International Conference on Robotics and Automation, vol. 2, IEEE, 138–145, 1985.

[11] D. Foster, R. Morris, P. Dayan, A model of hippocampally dependent navigation, using the temporal difference learning rule, Hippocampus 10 (1) (2000) 1–16.

[12] A. Arleo, W. Gerstner, Spatial cognition and neuro-mimetic navigation: a model of hippocampal place cell activity, Biological cybernetics 83 (3) (2000) 287–299.

[13] M. O. Franz, B. Schölkopf, H. A. Mallot, H. H. Bülthoff, Learning View Graphs for Robot Navigation, Auton. Robot 5 (1998) 111–125.

[14] A. Barrera, A. Weitzenfeld, Biologically-inspired robot spatial cognition based on rat neurophysiological studies, Autonomous Robots 25 (1-2) (2008) 147–169.

[15] C. A. Rothkopf, D. H. Ballard, Modular inverse reinforcement learning for visuomotor behavior, Biological cybernetics 107 (4) (2013) 477–490.

[16] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, A. Farhadi, Target-driven visual navigation in indoor scenes using deep reinforcement learning, in: Robotics and Automation (ICRA), 2017 IEEE International Conference on, IEEE, 3357–3364, 2017.

[17] P. Sermanet, K. Xu, S. Levine, Unsupervised Perceptual Rewards for Imitation Learning, CoRR abs/1612.06699, URL http://arxiv.org/abs/1612.06699.

[18] A. Singh, L. Yang, K. Hartikainen, C. Finn, S. Levine, End-to-End Robotic Reinforcement Learning without Reward Engineering, CoRR abs/1904.07854, URL http://arxiv.org/abs/1904.07854.

[19] A. D. Edwards, Perceptual Goal Specifications for Reinforcement Learning, Ph.D. thesis, PhD thesis, Georgia Institute of Technology, 2017.

[20] W. Yang, X. Wang, A. Farhadi, A. Gupta, R. Mottaghi, Visual Semantic Navigation using Scene Priors, CoRR abs/1810.06543, URL http://arxiv.org/abs/1810.06543.

[21] Y. Zhu, D. Gordon, E. Kolve, D. Fox, L. Fei-Fei, A. Gupta, R. Mottaghi, A. Farhadi, Visual semantic planning using deep successor representations, in: Proceedings of the IEEE International Conference on Computer Vision, 483–492, 2017.

[22] V. Dhiman, S. Banerjee, B. Griffin, J. M. Siskind, J. J. Corso, A Critical Investigation of Deep Reinforcement Learning for Navigation, CoRR abs/1802.02274, URL http://arxiv.org/abs/1802.02274.

[23] P. Anderson, A. X. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, A. R. Zamir, On Evaluation of Embodied Navigation Agents, CoRR abs/1807.06757, URL http://arxiv.org/abs/1807.06757.

[24] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, in: International conference on machine learning, 1928–1937, 2016.

[25] M. Chevalier-Boisvert, D. Bahdanau, S. Lahlou, L. Willems, C. Saharia, T. H. Nguyen, Y. Bengio, BabyAI: First steps towards grounded language learning with a human in the loop, arXiv preprint arXiv:1810.08272 .

[26] A. Glennerster, B. J. Rogers, M. F. Bradshaw, Stereoscopic depth constancy depends on the subject's task 36 (1996) 3441–3456.

[27] M. F. Bradshaw, A. D. Parton, A. Glennerster, The task-dependent use of binocular disparity and motion parallax information. 40 (2000) 3725–3734.

[28] J. B. Smeets, E. Brenner, Grasping Weber's law, Current Biology 18 (23) (2008) R1089–R1090.

[29] P. Mirowski, M. Grimes, M. Malinowski, K. M. Hermann, K. Anderson, D. Teplyashin, K. Simonyan, A. Zisserman, R. Hadsell, et al., Learning to navigate in cities without a map, in: Advances in Neural Information Processing Systems, 2419–2430, 2018.

[30] S. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, et al., Neural scene representation and rendering, Science 360 (6394) (2018) 1204–1210.

[31] T. Chen, S. Gupta, A. Gupta, Learning exploration policies for navigation, arXiv preprint arXiv:1903.01959 .

[32] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, J. Malik, Cognitive mapping and planning for visual navigation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2616–2625, 2017.

[33] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, et al., Learning to navigate in complex environments, arXiv preprint arXiv:1611.03673 .

[34] A. Kumar, S. Gupta, J. Malik, Learning Navigation Subroutines by Watching Videos, arXiv preprint arXiv:1905.12612 .

[35] I. Kanitscheider, I. Fiete, Training recurrent networks to generate hypotheses about how the brain solves hard navigation problems, in: Advances in Neural Information Processing Systems, 4529–4538, 2017.

[36] M. Milford, G. Wyeth, Persistent navigation and mapping using a biologically inspired SLAM system, The International Journal of Robotics Research 29 (9) (2010) 1131–1153.

[37] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, et al., Habitat: A platform for embodied ai research, arXiv preprint arXiv:1904.01201 .

[38] G. Wayne, C.-C. Hung, D. Amos, M. Mirza, A. Ahuja, A. Grabska-Barwinska, J. Rae, P. Mirowski, J. Z. Leibo, A. Santoro, et al., Unsupervised Predictive Memory in a Goal-Directed Agent, arXiv preprint arXiv:1803.10760 .

[39] R. Kinchla, Visual movement perception: A comparison of absolute and relative movement discrimination, Perception & Psychophysics 9 (2) (1971) 165–171.

[40] G. Westheimer, Cooperative Neural Processes Involved in Stereoscopic Acuity 36 (1979) 585–597.

[41] C. J. Erkelens, H. Collewijn, Motion perception during dichoptic viewing of moving random-dot stereograms 25 (1985) 583–588.

[42] O. M. Thomas, B. G. Cumming, A. J. Parker, A specialization for relative disparity in V2, Nature neuroscience 5 (5) (2002) 472–478.

[43] D. Regan, C. J. Erkelens, H. Collewijn, Necessary conditions for the perception of motion in depth., Investigative Ophthalmology & Visual Science 27 (4) (1986) 584–597.

[44] A. Glennerster, M. E. Hansard, A. W. Fitzgibbon, Fixation could simplify, not complicate, the interpretation of retinal flow 41 (2001) 815–834.

[45] R. J. Watt, Scanning from coarse to fine spatial scales in the human visual system after the onset of a stimulus, Journal of the Optical Society of America A 4 (1987) 2006–2021.

[46] R. J. Watt, Visual processing: computational, psychophysical and cognitive research, Lawrence Erlbaum Associates, Hove, 1988.

[47] A. Glennerster, M. E. Hansard, A. W. Fitzgibbon, View-based approaches to spatial representation in human vision, Lecture Notes in Computer Science 5064 (2009) 193–208.

[48] A. Glennerster, J. C. Read, A single coordinate framework for optic flow and binocular disparity, arXiv preprint arXiv:1808.03875 .

[49] J. Purdy, E. J. Gibson, Distance judgment by the method of fractionation., Journal of Experimental Psychology 50 (6) (1955) 374.

[50] J. J. Rieser, D. H. Ashmead, C. R. Talor, G. A. Youngquist, Visual perception and the guidance of locomotion without vision to previously seen targets, Perception 19 (5) (1990) 675–689.

[51] B. Bodenheimer, J. Meng, H. Wu, G. Narasimham, B. Rump, T. P. McNamara, T. H. Carr, J. J. Rieser, Distance estimation in virtual and real environments using bisection, in: Proceedings of the 4th Symposium on Applied Perception in Graphics and Visualization, 35–40, 2007.

[52] R. L. Klatzky, A. C. Beall, J. M. Loomis, R. G. Golledge, J. W. Philbeck, Human navigation ability: Tests of the encoding-error model of path integration, Spatial Cognition and Computation 1 (1) (1999) 31–65.

[53] P. Foo, W. H. Warren, A. Duchon, M. J. Tarr, Do humans integrate routes into a cognitive map? Map-versus landmark-based navigation of novel shortcuts., Journal of Experimental Psychology: Learning, Memory, and Cognition 31 (2) (2005) 195.

[54] D. Marr, Vision: A computational investigation into the human representation and processing of visual information., W.H. Freemann and Company, New Y, ISBN 0262514621, 1982.

[55] R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, MIT press, 2018.

[56] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529.

[57] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, D. Silver, Rainbow: Combining improvements in deep reinforcement learning, in: Thirty-Second AAAI Conference on Artificial Intelligence, 3215–3222, 2018.

[58] R. S. Sutton, D. A. McAllester, S. P. Singh, Y. Mansour, Policy gradient methods for reinforcement learning with function approximation, in: Advances in neural information processing systems, 1057–1063, 2000.

[59] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller, Deterministic Policy Gradient Algorithms, in: International Conference on Machine Learning, 387–395, 2014.

[60] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al., Mastering the game of go without human knowledge, Nature 550 (7676) (2017) 354.

[61] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al., A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play, Science 362 (6419) (2018) 1140–1144.

[62] S. Levine, C. Finn, T. Darrell, P. Abbeel, End-to-end training of deep visuomotor policies, The Journal of Machine Learning Research 17 (1) (2016) 1334–1373.

[63] J. Ruiz-del Solar, P. Loncomilla, N. Soto, A survey on deep learning methods for robot vision, arXiv preprint arXiv:1803.10862 .

[64] E. Kolve, R. Mottaghi, D. Gordon, Y. Zhu, A. Gupta, A. Farhadi, Ai2-thor: An interactive 3d environment for visual ai, arXiv preprint arXiv:1712.05474 .

[65] S. Chopra, R. Hadsell, Y. LeCun, et al., Learning a similarity metric discriminatively, with application to face verification, in: CVPR (1), 539–546, 2005.

[66] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 .

[67] L. v. d. Maaten, G. Hinton, Visualizing data using t-SNE, Journal of machine learning research 9 (Nov) (2008) 2579–2605.

[68] D. Bush, C. Barry, D. Manson, N. Burgess, Using grid cells for navigation, Neuron 87 (3) (2015) 507–520.

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

[70] T. E. Behrens, T. H. Muller, J. C. Whittington, S. Mark, A. B. Baram, K. L. Stachenfeld, Z. Kurth-Nelson, What is a cognitive map? Organizing knowledge for flexible behavior, Neuron 100 (2) (2018) 490–509.

[71] A. Chen, G. C. DeAngelis, D. E. Angelaki, Convergence of vestibular and visual self-motion signals in an area of the posterior sylvian fissure, Journal of Neuroscience 31 (32) (2011) 11617–11627.

[72] S. Gillner, H. A. Mallot, Navigation and acquisition of spatial knowledge in a virtual maze., Journal of Cognitive Neuroscience 10 (4) (1998) 445–463.

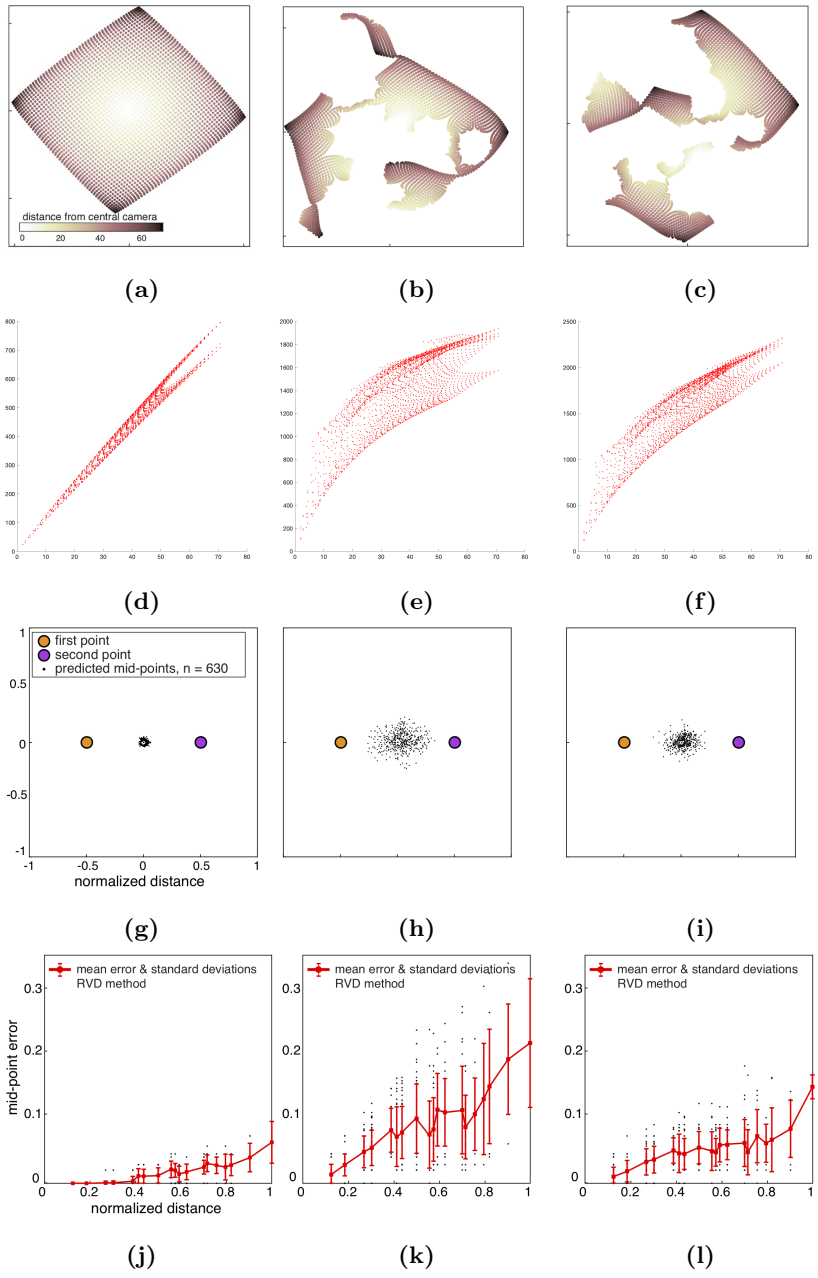
[73] D. Rosenbaum, F. Besse, F. Viola, D. J. Rezende, S. Eslami, Learning models for visual 3D localization with implicit mapping, arXiv preprint arXiv:1807.03149 .

[74] S. C. Hirtle, J. Jonides, Evidence of hierarchies in cognitive maps, Memory & Cognition 13 (3) (1985) 208–217.

[75] J. M. Wiener, H. A. Mallot, 'Fine-to-coarse'route planning and navigation in regionalized environments, Spatial Cognition and Computation 3 (4) (2003) 331–358.

[76] M. Milford, G. Wyeth, Persistent navigation and mapping using a biologically inspired SLAM system, The International Journal of Robotics Research 29 (9) (2010) 1131–1153.

Appendix



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

Figure A1: ***Consequences of using large-parallax elements in the RVD model.*** *a) re-plots the t-SNE projection of the RVD feature vectors from Fig 3f. b) shows the disruption in the representation caused by using a different subspace of $\boldsymbol{\varepsilon}$, namely picking out 30% of the elements of $\boldsymbol{\varepsilon}$ that have the* greatest *magnitude of motion parallax (Eqn. 2) rather than the smallest parallax, as we have used in all the previous figures. c) shows the effect of using* all *of $\boldsymbol{\varepsilon}$ rather than a subspace. d), e) and f) show the distance between feature vectors plotted against distance to the central camera (see Fig. 3d) using the feature vectors illustrated in a), b) and c) respectively. g), h) and i) show the consequence of using the vectors illustrated in a), b) and c) for the mid-point task (so i) is a repeat of Fig. 4d). j), k), l) show the magnitude of the midpoint errors, following the format of Fig. 4f.*

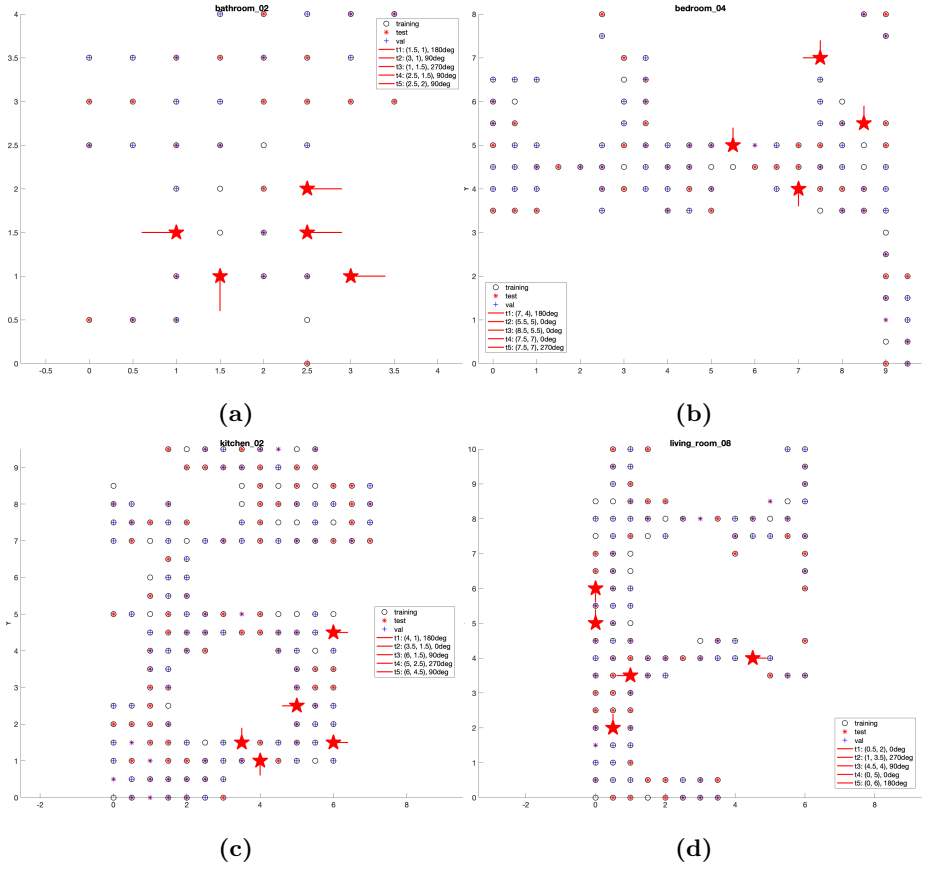


Figure A2: *Plan views of all 4 scenes used by Zhu et al. [16]. a) bathroom, b) bedroom, c) kitchen, d) living room. Symbols are as for Fig. 3a.*

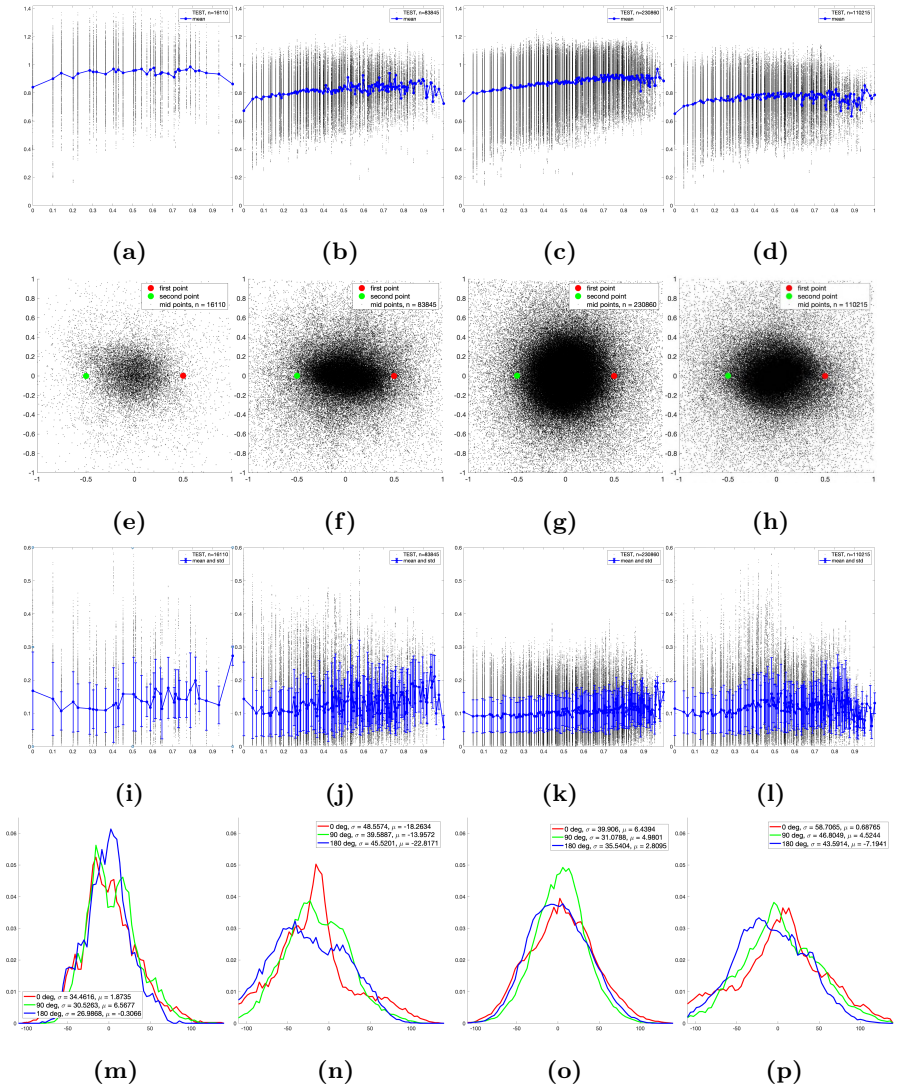


Figure A3: *Results for the bathroom scene were shown in Figs. 3 to 5 and are re-plotted here (left hand column). Results for the bedroom, kitchen and living room are shown in columns 2 to 4 respectively. In the top row, a-d), the correlations, R, are 0.088, 0.22, 0.24 and 0.14 respectively. For details of what is plotted in e-h) see Fig. 4c, for i-l) see Fig. 4e, and for m-p) see Fig. 5c.*

Default parameters for Adam	
β_1	0.9
β_2	0.999
ϵ	10^{-8}
use-locking	False
Position decoder	
learning rate	0.00001
λ_{L2}	0
Viewing angle decoder	
learning rate	0.0005
λ_{L2}	0.04

Table A1: ***Hyperparameters for the original trained network and the two decoder networks***. *The original trained network from Zhu et al. [16] was used throughout the paper, eg the t-SNE plot in Fig. 3e. The position decoder was used for the results shown in Fig. 4. The angle decoder was used for Fig. 5.*