A model of path integration that connects neural and symbolic representation

Nicole Sandra-Yaffa Dumont (ns2dumon@uwaterloo.ca)

Jeff Orchard (jorchard@uwaterloo.ca)
The Neurocognitive Computing Lab, University of Waterloo, Computer Science, Waterloo, ON, Canada

Chris Eliasmith (eliasmith@uwaterloo.ca)
Applied Brain Research Inc., Waterloo, ON, Canada

Abstract

Path integration, the ability to maintain an estimate of one's location by continuously integrating self-motion cues, is a vital component of the brain's navigation system. We present a spiking neural network model of path integration derived from a starting assumption that the brain represents continuous variables, such as spatial coordinates, using Spatial Semantic Pointers (SSPs). SSPs are a new kind of representation for encoding continuous variables as high-dimensional vectors, and can also be used to create structured, hierarchical representations for neural cognitive modelling. Path integration can be performed by a recurrently-connected neural network using SSP representations. Unlike past work, we show that our model can be used to continuously update variables of any dimensionality. Finally, we demonstrate that symbol-like object representations can be bound to continuous SSP representations. Specifically, we incorporate a simple model of working memory to remember environment maps with such symbol-like representations situated in 2D space.

Keywords: path integration; spiking neural network; recurrent neural networks; vector symbolic architecture; grid cells; neural engineering framework; cognitive maps

Introduction

Path integration is the process of integrating idiothetic cues, obtained from a variety of sensory systems, to maintain an estimate of one's position in space (relative to some starting position). Animals can perform this computation while navigating environments in the absence of allocentric spatial cues (Mittelstaedt & Mittelstaedt, 1982). Path integration is an essential component of biological spatial navigation systems and is the foundation of both vector-based and map-based navigational strategies. Maintaining homing vectors to important locations (e.g., shelter or food) while moving through an environment requires updating those vectors using self-motion information. Building a cognitive map of an environment also requires tracking one's position relative to landmarks.

Beyond navigation, path integration (PI) is an important component of larger systems capable of advanced reasoning, planning, and cognition. The results of PI are known to be used in downstream tasks, like creating maps, associating memories with locations, and planning trajectories for exploration (Savelli & Knierim, 2019; Buzsáki, 2005). Furthermore, there is evidence that the same neural systems involved in PI and spatial representation may be involved in integration and mapping of non-spatial continuous features (Garvert et al., 2017). Even episodic memory can be thought of as mapping events in space and time (Schiller et al., 2015).

Significant advances have been made in understanding the neural substrates of spatial representation. The hippocampal formation is generally believed to be the site of PI in mammals due to the presence of spatially-sensitive neurons, namely place and grid cells. A place cell will fire at a consistent location in an environment, even without external sensory information (Quirk et al., 1990). Grid cells, which fire at hexagonally tiled locations in space, appear to represent a spatial coordinate system and are considered instrumental in PI. In the entorhinal cortex, they live alongside neurons sensitive to head-direction and speed (Sargolini et al., 2006), both required for PI. It has been suggested that the function of grid cells may be to encode a representation of space robust to noise and to help error-correct integration (Sreenivasan & Fiete, 2011). Grid cells have also been linked to navigation of non-spatial variables, including visual and auditory features (Aronov et al., 2017; Constantinescu et al., 2016). This suggests that the neural mechanisms behind creating and navigating spatial cognitive maps may be used more generally for internal maps of continuous variables – even ones that may be conceptual and high-dimensional.

While much work has been done on computational modelling of PI, there is still a gap in understanding how the output of a neural PI model can be used for cognitive mapping, involving symbol-like representations of objects, landmarks, and other discrete features. We approach the problem of PI using a framework for symbol-like representation in the brain, the Semantic Pointer Architecture (SPA; Eliasmith (2013)). The SPA is a Vector-Symbolic Architecture (VSA) that can represent information in a vector space, and uses vector operations to perform symbolic computations. In particular, it offers methods for implementing such representation and operations using spiking neural networks.

Recently, these methods have been extended to include Spatial Semantic Pointers (SSPs; Komer et al. (2019)), an encoding for continuous variables, such as spatial coordinates (of any dimension), time, size, value, etc. SSPs naturally produce oscillators, and can be encoded by the collective activity of grid cells with patterns of varying orientation and scale (Dumont & Eliasmith, 2020). In this work, we derive and model the dynamics of SSPs and discover configurations that result in an oscillator-interference model of PI. Furthermore,

In J. Culbertson, A. Perfors, H. Rabagliati & V. Ramenzoni (Eds.), *Proceedings of the 44th Annual Conference of the Cognitive Science Society.*

we demonstrate how SSPs can be coupled with more symbol-like representations via the methods of SPA to build spatial cognitive maps.

Background

Modelling of path integration

Many computational models of grid cells and their role in path integration have been proposed since their discovery. Models generally to fall into one of two categories: oscillator-interference (OI) models (O'Keefe & Burgess, 2005) and continuous attractor neural network (CANN) models (Samsonovich & McNaughton, 1997). In CANN models, path integration is performed by a recurrently connected neural sheet whose dynamics sustain a single Gaussian-like activity bump that represents self-position. Particular connectivity is used to achieve this – excitation from nearby neurons to maintain the bump and inhibition from faraway neurons to prevent multiple bumps forming. The activity bump moves across the sheet in response to velocity signals. Grid cell patterns emerge when the sheet is given the boundary conditions of a twisted torus (Conklin & Eliasmith, 2005).

In OI models, grid cell firing patterns are the result of interference between velocity-controlled oscillators (VCOs), an oscillator whose frequency is modulated by a velocity signal. In this setup, a position estimate is encoded in the phase differences between oscillators. The biological basis of such oscillators can differ between OI models, from intrinsic oscillation of membrane potentials (Burgess et al., 2007), to the more widely accepted recurrently-connected oscillator networks (Blair et al., 2008; Orchard et al., 2013). It should be noted that VCOs can also create other spatial patterns, such as bands and pseudo-periodic patterns (Krupic et al., 2012). Fig. 1 illustrates a VCO, and its corresponding spatial map. Attractor dynamics can also be added to OI models to limit phase drift and error accumulation (Bush & Burgess, 2014).

However, all path integration models suffer from error accumulation. Without external cues or corrections, position estimates can only worsen over time. Animals have access to a plethora of external sensory information, such as visual landmarks and odour trails, which can be used to correct the errors that would collect when using PI alone. Such multi-sensory information can be used in conjugation with PI to create cognitive maps of environments. Simultaneous localization and mapping (SLAM), a well-studied problem in robotics, is exactly the problem of path integrating while simultaneously constructing internal spatial maps. A biological inspired SLAM mode, RatSLAM, uses an CANN consisting of "pose cells" (combining place & head direction-like cells) to maintain an estimate of self-position and orientation (Milford et al., 2004).

Semantic Pointer Architecture

The Semantic Pointer Architecture (SPA) posits that high dimensional vectors (called semantic pointers) are the building blocks of cognition (Eliasmith, 2013). The SPA includes a form of Vector Symbolic Architecture (VSA). A VSA represents symbols and structured compositions of symbols as high-dimensional vectors, and operations on those vectors perform symbolic operations, such as bundling, binding, comparison, and inversion. The specification of these operations differentiates particular VSAs. The SPA uses Holographic Reduced Representations (HHRs; Plate (1995)), implemented in spiking neural networks.

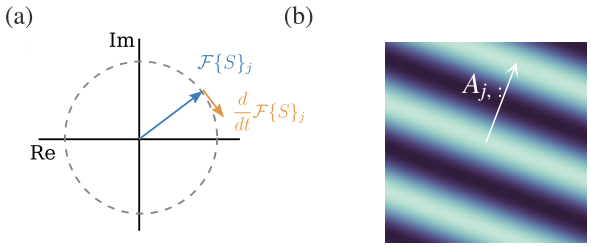

Figure 1: **(a)** A single component of the Fourier transform of an SSP, $\mathcal{F}\{S\}_j$, is a unit-modulus complex number. As the animal moves, the phase changes, making it a VCO. **(b)** The phase of $\mathcal{F}\{S\}_j$ over space is plotted via a heat map over space. The inner-product between the animal's velocity and the preferred heading direction $A_{j,:}$ (see Eq. (2)) determines how quickly the phase changes.

A *similarity measure* between two vectors indicates the semantic similarity of the symbols they represent. For example, the vector representation of a 'rose' might have a higher similarity measure with a 'lily' vector compared to a 'toad' vector. In HHRs, this is given by the cosine similarity.

A *bundling* operation maps a set of vectors to a single vector, such that the resulting vector is similar to all from the set. Vector addition is the bundling operation in HRRs. A *binding* operation maps a pair of vectors to a single vector that is dissimilar to both. For example, 'colour' and 'red' could be bound together to represent that something is red in colour. Using an HRR, binding can be done by circular convolution,

$$A \circledast B = \mathcal{F}^{-1}\{\mathcal{F}\{A\} \odot \mathcal{F}\{B\}\}, \tag{1}$$

where $\mathcal{F}$ is the Fourier transform, and $\odot$ is the Hadamard product. The *inverse* operation takes a single input vector and produces a single output vector that approximately reverses the effect of binding with the input vector, $(A \circledast B) \circledast B^{-1} \approx A$, where $B^{-1} = [B_1, B_d, B_{d-1}, \ldots, B_2]$.

To see how these operations can be used to encode and manipulate symbolic representations, consider the example of an ordered list. Vector representations of each object (fillers) can be bound with vector representations of indices (slots) and the results summed to represent their collection in a single ordered list. The final representation can be queried to, for example, retrieve the position of a given object in the list, by binding the total vector with the inverse of that object's vector. The result of such a binding will be approximately equal to the vector that represents that object's index.

While VSAs prescribe these operations for manipulating

symbols in some vector space, they do not prescribe how the mapping from symbols to vectors is done. For some tasks, the corpus of symbols are distinct and so it is sufficient that all vector representations are dissimilar from one another. In this case, random vectors with high enough dimensionality will do. For other tasks, vectors can be chosen to have desired similarity relations, or machine learning techniques can be used to obtain vector representations (Mitrokhin et al., 2020).

Spatial Semantic Pointers (SSPs) extend VSAs to support representation of continuous features (Komer et al., 2019). An d-dimensional SSP representing an n-dimensional variable $\mathbf{x}$ is given by

$$S(\mathbf{x}) = \mathcal{F}^{-1}\left\{e^{iA\mathbf{x}}\right\} \tag{2}$$

where $A \in \mathbb{R}^{d\times n}$ is the encoding matrix of the representation. A useful property of SSPs is that binding in the SSP space is equivalent to addition in the variable space, $S(\mathbf{x}) \circledast S(\mathbf{x}') = S(\mathbf{x}+\mathbf{x}')$. SSPs can be used in tandem with symbolic-like representations from standard VSAs. For example, an object represented by vector B, located at a position encoded by an SSP S, can be represented by $B \circledast S$. Similarly, a set of objects at different locations can be represented by $\sum_i B_i \circledast S(\mathbf{x}_i)$. This vector is a compressed representation associating features and locations – a spatial map. The vector could be stored in memory and later accessed and queried for object locations using the inverse operation. By 'unbinding' the object vector B_j from the map, an approximation of its location as an SSP can be recovered, $M \circledast B_j^{-1} \approx S(\mathbf{x}_j)$.

Neural Engineering Framework To create biologically realistic neural networks that make use of SSPs, we require methods to represent vectors by the activity of spiking neurons, and to be able to perform computation on said vectors via projections between neural populations.

The principle of *representation* explains how the collective neural activity of a population can encode a vector, $S \in \mathbb{R}^d$. A single neuron's activity is given by,

$$a_i(t) = \mathcal{G}_i\left[\alpha_i \mathbf{e}_i \cdot S + \beta_i\right], \tag{3}$$

where $a_i(t)$ is the activity of neuron i (a spike train), $\alpha_i > 0$ is its gain, β_i is its bias, $\mathcal{G}_i$ is a nonlinear function (in this work, the leaky-integrate-and-fire function), and $\mathbf{e}_i$ is the encoder of the neuron. Gains and biases are parameters that are randomly selected for different neurons and relate to their maximum firing rates. Encoders define what sort of input a particular neuron is sensitive to, hence capturing the 'receptive field' of a neuron. In the case where neurons are a part of a population representing SSPs, it would be natural to set encoders to be SSPs that represent random points across space. This would result in a population whose neurons are sensitive to particular spatial locations – e.g., place cells. However, encoders can also be set to obtain grid cells (Dumont & Eliasmith, 2020).

The representation principle also explains how to decode the vector represented by the activity of a population of N neurons:

$$\hat{S} = \sum_{i=1}^{N} a_i(t) * h(t)\mathbf{d}_i, \tag{4}$$

where $*$ is convolution and $\mathbf{d}_i \in \mathbb{R}^d$ are the decoders of the population. The decoders are solved for via least-squares optimization. The function $h(t)$ is a post-synaptic filter and is parameterized by τ, the post-synaptic time constant.

The *transformation* principle of the NEF provides the method for setting weights between two neural populations to compute a desired function. Assume a population of N neurons representing a vector, S, is fully connected to a different population of N' neurons. We would like the second population to represent some function of the vector, $f(S)$. This function can be decoded out of the first population's activity,

$$\widehat{f(S)} = \sum_{i=1}^{N} a_i(t) * h(t)\mathbf{d}_i^{(f)}. \tag{5}$$

These function-specified decoders can be solved for using least-squares optimization and samples of the desired function output. The second population's neurons will receive this as input. Decoding the output of the first population and encoding it in the activity of the second population is equivalent to multiplying the filtered activities of the first population with a weight matrix and feeding that current into the second population, which will have activities given by

$$b_j(t) = \mathcal{G}_i\left[\sum_{i=1}^{N} w_{ij}a_i(t) + \beta_j\right], \quad w_{ij} = \alpha_j \mathbf{e}_j \times \mathbf{d}_i^{(f)}. \tag{6}$$

This is a standard neural network, with populations connected via weighted synapses – only here, the weight matrices are set as the outer product between the decoders of the first population (which are optimally solved for) and the encoders of the second (which are pre-set, randomly or to match biological tuning curves). The last principle of the NEF is *dynamics*. Dynamical systems can be encoded in a recurrently connected population of spiking neurons.

Methods

Dynamics of Spatial Semantic Pointers

Despite the fact that SSPs are defined in terms of a high-dimensional vector space, we can show that they naturally implement velocity controlled oscillators (VCOs). Consider how $S(\mathbf{x})$ changes if $\mathbf{x}$ is a function of time. We can relate the rate of change of S to $\dot{\mathbf{x}}(t)$, the rate of change of $\mathbf{x}(t)$ (Voelker et al., 2021). In the case of an SSP representing an animal's position, $\dot{\mathbf{x}}(t)$ is the animal's velocity.

The derivative of an SSP representing the coordinates $\mathbf{x}(t) \in \mathbb{R}^n$ is

$$\dot{S}(\mathbf{x}(t)) = \mathcal{F}^{-1}\{e^{iA\mathbf{x}(t)} \odot iA\dot{\mathbf{x}}(t)\}, \tag{7}$$

$$= S(\mathbf{x}(t)) \circledast \ln S(\dot{\mathbf{x}}(t)), \tag{8}$$

where $\odot$ is element-wise multiplication, and the logarithm of an SSP is defined as the element-wise logarithm in the Fourier

domain, $\ln S \equiv \mathcal{F}^{-1}\{\ln \mathcal{F}\{S\}\}$. Now let us consider the dynamics of an SSP in the Fourier domain. Taking the Fourier transform of (7), we get,

$$\mathcal{F}\{\dot{S}(\mathbf{x}(t))\} = (iA\dot{\mathbf{x}}) \odot \mathcal{F}\{S(\mathbf{x}(t))\}. \tag{9}$$

Note that the dynamics of the Fourier components of an SSP are independent of each other. The dynamics of the j^{th} Fourier component of the SSP can be written as

$$\frac{d}{dt}\begin{bmatrix}\mathrm{Re}\mathcal{F}\{S\}_j \\ \mathrm{Im}\mathcal{F}\{S\}_j\end{bmatrix} = \begin{bmatrix}0 & -\omega_j \\ \omega_j & 0\end{bmatrix}\begin{bmatrix}\mathrm{Re}\mathcal{F}\{S\}_j \\ \mathrm{Im}\mathcal{F}\{S\}_j\end{bmatrix}, \tag{10}$$

$$\text{where } \omega_j \equiv A_{j,:}\cdot\dot{\mathbf{x}}(t) = -i\ln \mathcal{F}\{S(\mathbf{x}(\dot{t}))\}_j. \tag{11}$$

Each Fourier component of the SSP is thus the state of a simple harmonic oscillator whose frequency is given by ω_j. The oscillators' frequencies are modulated by the velocity $\dot{\mathbf{x}}$; in other words, they are velocity controlled oscillators (VCOs).

Attractor dynamics

Path integration can be accomplished by integrating velocity. To do so with SSPs, we may construct a neural network that integrates (8) or (10). The advantage of integrating (10) is that each Fourier component is an independent VCO. This means that many sets of small neural populations, each recurrently connected only to itself, can realize these dynamics.

However, a well-known issue with the VCO method is that noise causes drift in the oscillators and their accuracy quickly deteriorates (Zilli et al., 2009). This noise drift becomes even more acute when using spiking neurons. However, the VCO approach can be modified to improve the stability of the oscillators. VCOs can either be coupled together (Zilli & Hasselmo, 2010; Burgess & Burgess, 2014) or attractor dynamics can be used to make the dynamics more robust (Bush & Burgess, 2014). We use the latter approach here. Cyclic attractors have been previously modelled with the NEF to describe the head-direction system (Eliasmith, 2005). Here, instead of using the dynamics of a simple harmonic oscillator, we use a nonlinear oscillator with a stable limit cycle,

$$\frac{d}{dt}\begin{bmatrix}\mathrm{Re}\mathcal{F}\{S\}_j \\ \mathrm{Im}\mathcal{F}\{S\}_j\end{bmatrix} = \begin{bmatrix}-\omega_j \mathrm{Im}\mathcal{F}\{S\}_j + \frac{1-r^2}{r}\mathrm{Re}\mathcal{F}\{S\}_j \\ \omega_j \mathrm{Re}\mathcal{F}\{S\}_j + \frac{1-r^2}{r}\mathrm{Im}\mathcal{F}\{S\}_j\end{bmatrix}, \tag{12}$$

$$\text{where } r \equiv |\mathcal{F}\{S\}_j|.$$

This ensures that the SSP remains unitary as it evolves. Considering the set of VCOs as a whole, the system has a toroidal attractor (Komer, 2020), making this a hybrid of oscillator-interference and continuous attractor models of PI.

Path integration model

To represent a spatial location, $\mathbf{x}(t)$, we define an SSP in a higher-dimensional space, $S(\mathbf{x}(t)) \in \mathbb{R}^d$. To implement this representation in spiking neurons using the NEF, we use $\lfloor \frac{d}{2} \rfloor$ VCOs, each of which consists of N neurons (see Fig. 2). The j^{th} neural population represents the real and imaginary parts of the Fourier components of the SSP. Only $\lfloor \frac{d}{2} \rfloor$ oscillator

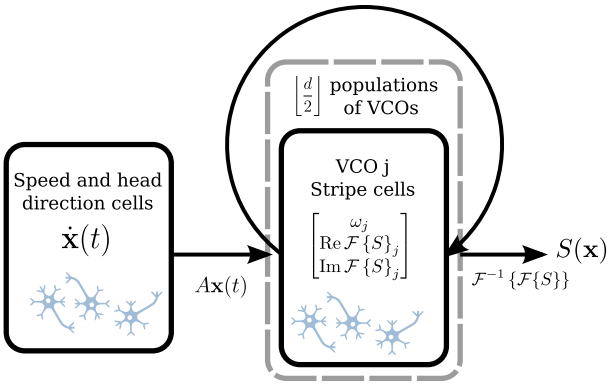

Figure 2: A diagram of the path integration model.

populations are needed since the Fourier transform of the SSP will have conjugate symmetry (half of its Fourier components can be computed from the other half).

To compute the nonlinear dynamics in (12), we need to represent both real and imaginary parts of $\mathcal{F}\{S\}_j$, as well as ω_j, all in the same population of neurons (as is standard in the NEF). The vector being represented by the collective activity of the j^{th} VCO population is, thus,

$$\begin{bmatrix}\omega_j & \mathrm{Re}\mathcal{F}\{S\}_j & \mathrm{Im}\mathcal{F}\{S\}_j\end{bmatrix}^T. \tag{13}$$

The VCO frequency, ω_j, is computed by a set of connections from a population that encodes velocity. This population is connected to the j^{th} VCO by connection weights $A_{j,:}$, and hence computes $A_{j,:}\cdot\dot{\mathbf{x}}(t)$. Within each VCO, the neuron population is recurrently connected using weights optimized by least squares to implement the dynamics of (12). This model performs PI of continuous variables of any dimensionality, as long as they are represented by SSPs of the same dimension; the only difference for higher-dimensional vectors is the calculation of the frequency input.

We have not yet addressed how this model can be used to generate grid cells. Each oscillator is a population representing a frequency (related to speed and head direction) and a single Fourier component of the SSP. This results in neurons with conjunctive sensitivity to head direction, speed, and spatial position (in a periodic fashion, resembling a plane wave). Their firing patterns are velocity dependent bands or stripes and not hexagonal patterns. Grid cells do not intrinsically emerge from PI in this model, although particular choices of axis vectors do realize grid cells. Furthermore, such a choice provides an ideal basis for subsequent generation of place cells (Dumont & Eliasmith, 2020).

Cognitive mapping model

A core value of this model compared to pre-existing PI models is its position within the broader SPA framework. The output of this PI model can be used in a variety of ways within the SPA. It can, for example, be bound with other SSPs to compute vectors between self-position and other locations. It can also be bound with semantic pointers representing some

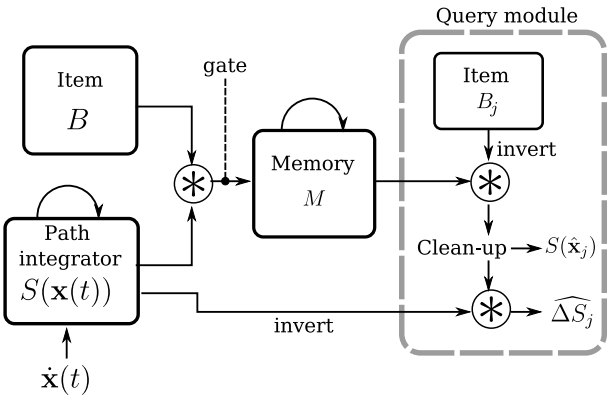

Figure 3: A diagram of the cognitive mapping model.

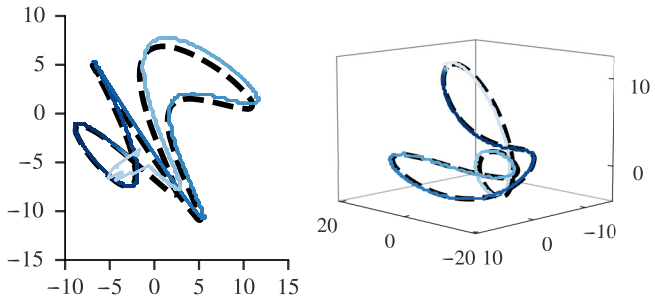

Figure 4: Path integration results on one-minute-long paths. In these plots, the true path is plotted as a dashed black line and a moving average of the path estimate produced by the PI model is plotted as a solid blue line. The colour gradient indicates similarity of the estimate to the true SSP.

sort of feature or landmark, and used to create an environment map. Here, we will demonstrate that the PI model can also be integrated with a working memory model for cognitive mapping.

This mapping model, illustrated in Fig. 3, consists of a PI module that represents an SSP estimate of current position, a neural population that represents nearby items as symbol-like semantic pointers, a gated working memory module that represents the cognitive map, and a query module to probe the accuracy of the map. The output of the PI model, $S(\mathbf{x}(t))$, is bound with the output of the item network, B, by a neural network that performs circular convolution. The result, $B \circledast S(\mathbf{x}(t))$, is the input of the memory module.

The memory module is a gated neural integrator (a recurrently connected population). The activity of the population stores a spatial map. A gating signal determines whether the input to the module is allowed through. When no object is nearby, the input is gated and the integrator maintains its encoded values. When objects are observed, the input is added to the value represented by the integrator. After seeing m items, the memory would ideally be $M = \sum_{i=1}^{m} B_i \circledast S(\mathbf{x}_i)$, where B_i are the items observed and $\mathbf{x}_i$ are their locations.

The output of the memory module is given to the query module, where it is bound with the inverse of each item's vector representation. This produces, $M \circledast B_j^{-1} = \widehat{S(\mathbf{x}_j)}$, a noisy recall of the location at which the item was observed. The result is 'cleaned-up' to be a proper SSP, $S(\hat{\mathbf{x}}_j)$; it is compared to an array of SSPs (representing a grid of locations) to find the SSP with which it has the highest similarity. This similarity, along with the distance between the resulting SSP and the item's true location, indicates the accuracy of the map. The output of the clean-up step is also bound with the inverse of the vector from the PI model to give an SSP representation of the displacement between the agent's current location and the location of the item, as remembered by the cognitive map, $S(\widehat{\mathbf{x}}_j) \circledast S(\mathbf{x}(t))^{-1} = S(\hat{\mathbf{x}}_j - \mathbf{x}(t))$ (labelled $\widehat{\Delta S_j}$ in the diagram).

Results

Path integration results

The PI model was tested on minute-long paths randomly generated from frequency-bounded white noise signals. The model was initialized with the SSP representation of the starting point of the path. As input, it received the velocity along the path (computed using finite differences) over the simulation run time. The model used 151-dimensional SSPs, 75,000 spiking neurons in total for the VCO populations, and 1,000 neurons for the population representing the velocity input.

To determine the accuracy of the model, the raw spiking data was interpreted as a position estimate. The vector represented by the VCO populations was decoded from neural activities using a grid of SSPs covering the space. The similarities between these SSPs and the vector decoded from the PI model were computed at every time step. We use the position represented by the SSP with which the vector was most similar as the position estimate of the PI model. This process of using a large set of SSPs to decipher the raw output of a neural network is a type of clean-up method that has been previously used with SSPs (Voelker et al., 2021). Fig. 4 shows examples (in both 2D and 3D) of the path estimate of the model compared to the exact path: the model accurately follows the true path for the entire trajectory.

Firing patterns

The path integration models were also tested on a one-minute-long wiggly, spiral path to record the spike trains of neurons in the models. This path was used because it covers space well, allowing for firing patterns to be easily discerned. Examples of neurons in the model populations are shown in Fig. 5. The VCO neurons have striped or band-like patterns over space, but some only exhibit these patterns in certain areas because they are also sensitive to head direction. An example grid cell from the population representing the output of the PI model is also shown.

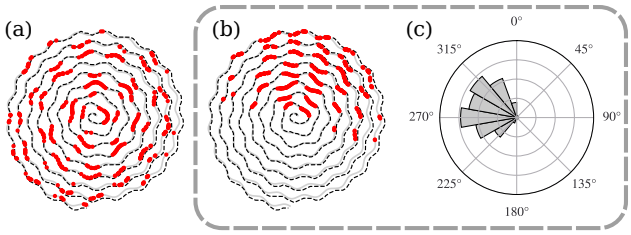

Figure 5: Sample neurons from the PI model. In **(a)** and **(b)**, the solid grey line is the true path the agent follows. The dotted black line shows the model's estimate of the agent's location. Red dots are plotted along this line at locations where a particular neuron fired during the path integration task. **(a)** An example grid cell from a population representing the VCO output. **(b)** An example neuron from a VCO population, which has a banded firing pattern. **(c)** Number of spikes binned by heading direction for the VCO neuron in **(b)**.

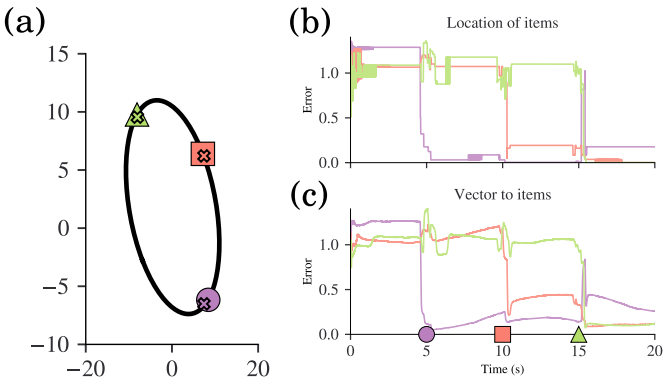

Figure 6: The cognitive mapping results. **(a)** The path and items along it (marked by coloured shapes). The 'x' markers indicate recalled item locations at the end of the simulation. **(b)** The error in recollection of the item locations as SSPs over time, where the error is equal to the absolute value of the difference between one and the similarity of the recalled vector to the true vector. The markers on the time axis show when the items appeared along the path. **(c)** Error in the model's estimate of the vector between items and self-position.

Mapping results

The working memory model of cognitive mapping was tested on a 20-second-long elliptical path. Three items appeared along the path at five-second time intervals. The values of $S(\hat{\mathbf{x}}_j)$ (the item location recalled by the model, post clean-up operation) and $\widehat{\Delta S_j}$ (the displacement vector between the current self-position estimate and recalled item locations, see Fig. 3), were recorded over the course of the simulation for each of the items. The errors over the course of the simulation are plotted in Fig. 6, where the error is a measure of dissimilarity between the recalled SSPs and the true SSP vectors. The error for each recalled location was near one (i.e., the similarity measure was near zero) until item j was first observed. The error for the recalled circle location went to zero once it was observed at the five-second mark, and increased slightly over time as more item-location pairs were added to the working memory. By the end of the simulation, reasonable position estimates are obtained. Fig. 6(a) shows these final estimates compared to their actual locations. Additionally, Fig. 6(c) plots the error in the network's estimate of the vector displacement between self-position at time t and each item's exact location. The error is low for each item once observed. The result of this neural calculation can be used in down-stream tasks like vector-based navigation.

Conclusion

We have proposed a novel model of path integration that both captures neural cell types observed in the brain and allows for symbol-like representations to be incorporated into cognitive maps. We believe that coupling low-level neural models with cognitive architectures in this manner is critical for building sophisticated models of biological cognition. The activity of hippocampual neurons has provided valuable insight into how the brain represents space but does not reveal its underlying algorithms, such as PI. By constructing a path integrator out of spiking neurons, we have linked the activity of spatial sensitive neurons to a symbolic description of PI using SSPs. Furthermore by incorporating this PI model into a model that uses working memory, we showed how such a system can learn a simple map of objects in a continuous space.

However, still much work remains to cement the value of this approach. Working memory is only accurate over short time periods and there are limits to the amount of information that can be stored. The model presented here could be improved by replacing this component with a network that learns a mapping between object symbols and SSPs. This long-term memory would be realized with synaptic weight changes rather than sustained neural activity. Finally, combining our PI model with this more robust memory would constitute a SLAM model. The memory would act as an environment map that's corrects the PI network, minimizing the drift from error accumulation.

While the model we propose here is simple, it provides a new, robust, and scalable means of integrating discrete symbol-like and continuous representations in a neural substrate. Future work will focus on increasing the sophistication of the maps that are learned, the number of maps that are learned, and the complexity of spatial reasoning tasks that are performed using these representations.

References

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

Blair, H. T., Gupta, K., & Zhang, K. (2008). Conversion of a phase-to a rate-coded position signal by a three-stage model of theta cells, grid cells, and place cells. *Hippocampus*, *18*(12).

Burgess, C. P., & Burgess, N. (2014). Controlling phase noise in oscillatory interference models of grid cell firing. *J. Neurosci.*, *34*(18).

Bush, D., & Burgess, N. (2014). A hybrid oscillatory interference/continuous attractor network model of grid cell firing. *J. Neurosci.*, *34*(14).

Buzsáki, G. (2005). Theta rhythm of navigation: link between path integration and landmark navigation, episodic and semantic memory. *Hippocampus*, *15*(7).

Conklin, J., & Eliasmith, C. (2005). A controlled attractor network model of path integration in the rat. *J. Comput. Neurosci.*, *18*(2).

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

Dumont, N. S.-Y., & Eliasmith, C. (2020). Accurate representation for spatial cognition using grid cells..

Eliasmith, C. (2005). A unified approach to building and controlling spiking attractor networks. *Neural Comput.*, *17*(6).

Eliasmith, C. (2013). *How to build a brain: A neural architecture for biological cognition*. Oxford University Press.

Garvert, M. M., Dolan, R. J., & Behrens, T. E. (2017). A map of abstract relational knowledge in the human hippocampal–entorhinal cortex. *Elife*, *6*.

Komer, B. (2020). *Biologically inspired spatial representation*. Phd thesis, University of Waterloo.

Komer, B., Stewart, T. C., Voelker, A. R., & Eliasmith, C. (2019). A neural representation of continuous space using fractional binding. In *41st annual meeting of the cognitive science society*. Cognitive Science Society.

Krupic, J., Burgess, N., & O'Keefe, J. (2012). Neural representations of location composed of spatially periodic bands. *Science*, *337*(6096).

Milford, M. J., Wyeth, G. F., & Prasser, D. (2004). Ratslam: a hippocampal model for simultaneous localization and mapping. In *Ieee international conference on robotics and automation, 2004. proceedings. icra'04. 2004* (Vol. 1, pp. 403–408).

Mitrokhin, A., Sutor, P., Summers-Stay, D., Fermüller, C., & Aloimonos, Y. (2020). Symbolic representation and learning with hyperdimensional computing. *Front Robot AI*, *7*.

Mittelstaedt, H., & Mittelstaedt, M.-L. (1982). Homing by path integration. In *Avian navigation*. Springer.

O'Keefe, J., & Burgess, N. (2005). Dual phase and rate coding in hippocampal place cells: theoretical significance and relationship to entorhinal grid cells. *Hippocampus*, *15*(7).

Orchard, J., Yang, H., & Ji, X. (2013). Does the entorhinal cortex use the fourier transform? *Front. Comput. Neurosci.*, *7*.

Plate, T. A. (1995). Holographic reduced representations. *IEEE Trans Neural Netw Learn Syst*, *6*(3).

Quirk, G. J., Muller, R. U., & Kubie, J. L. (1990). The firing of hippocampal place cells in the dark depends on the rat's recent experience. *J. Neurosci.*, *10*(6).

Samsonovich, A., & McNaughton, B. L. (1997). Path integration and cognitive mapping in a continuous attractor neural network model. *J. Neurosci.*, *17*(15).

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

Savelli, F., & Knierim, J. J. (2019). Origin and role of path integration in the cognitive representations of the hippocampus: computational insights into open questions. *J. Exp. Biol.*, *222*.

Schiller, D., Eichenbaum, H., Buffalo, E. A., Davachi, L., Foster, D. J., Leutgeb, S., & Ranganath, C. (2015). Memory and space: towards an understanding of the cognitive map. *J. Neurosci.*, *35*(41).

Sreenivasan, S., & Fiete, I. (2011). Grid cells generate an analog error-correcting code for singularly precise neural computation. *Nat. Neurosci.*, *14*(10).

Voelker, A. R., Blouw, P., Choo, X., Dumont, N. S.-Y., Stewart, T. C., & Eliasmith, C. (2021). Simulating and predicting dynamical systems with spatial semantic pointers. *Neural Comput.*, *33*(8).

Zilli, E. A., & Hasselmo, M. E. (2010). Coupled noisy spiking neurons as velocity-controlled oscillators in a model of grid cell spatial firing. *J. Neurosci.*, *30*(41).

Zilli, E. A., Yoshida, M., Tahvildari, B., Giocomo, L. M., & Hasselmo, M. E. (2009). Evaluation of the oscillatory interference model of grid cell firing through analysis and measured period variance of some biological oscillators. *PLoS Comput. Biol.*, *5*(11).