Transition scale-spaces: A computational theory for the discretized entorhinal cortex

Nicolai Waniek[1]

[1]Bosch Center for Artificial Intelligence (BCAI), Robert Bosch GmbH, Robert-Bosch-Campus 1, 71272, Renningen, Germany..

Keywords: grid cells, entorhinal cortex, spatial navigation, transition systems, scale-space, multi-resolution

Abstract

Goal-directed navigation is crucial for mobile animals and attributed to place and grid cells of the Hippocampus. However, the computational purpose of grid cells remains disputed. Recently, grid cells were proposed to represent spatial transitions in spatio-temporal sequences and to convey this knowledge downstream to a symbolic space, represented by place cells.

Here, reward-free transition encoding is shown, in a theoretical model, to

be compatible with the discretization along the dorso-ventral axis of medial Entorhinal Cortex (MEC). First, a single scale of transitions is shown to perform insufficiently for retrievals of long sequences. Then, equivalence between retrievals and interval searches on linked lists is noted. Consequently, the performance is addressed exclusively on algorithmic levels of the computation using well-studied data structures. The resulting Transition Scale-Space (TSS) forms a hierarchy of encoders that simultaneously operate on the input and symbolic space. On discrete data, existing optimality results mandate a scale increment of 2 between consecutive scales. The optimal spatial scale increment is then derived to be $\sqrt{2}$ for data that follows neurally plausible tuning curves. Subsequently, temporal integration windows are shown to increase likewise, which requires a temporal buffer mechanism for online learning of a TSS. Then, two traversal modes for sequence retrieval, namely top-down and bottom-up, are presented. These modes are evaluated in symbolic simulations, i.e., without biologically plausible spiking neurons. Finally, a TSS is used for short-cut discovery in a simulated Morris water maze.

The presented algorithms are related in particular to multi-resolution path planning, a high-performance technique widely deployed in mobile robotics. The model also explicitly connects to scale-space theory, widely used in computer vision and normative models of visual receptive fields. Because the theoretical model generalizes beyond navigation, TSS are suggested to be a general-purpose cortical data structure for sequence retrievals.

1 Introduction and Motivation

Spatial navigation and localization depend on a number of cells which represent spatial modalities (Rowland et al., 2016). Place cells, for instance, selectively respond to the location of an animal by firing only in distinct localized regions of an environment, called place fields (O'Keefe, 1979), and were found to be of critical importance for localization and path planning (de Lavilléon et al., 2015; Morris et al., 1982). The most perplexing cells are, however, grid cells of the Entorhinal Cortex (EC) (Hafting et al., 2005). Their eponymous periodic spatial firing patterns, called grid fields, regularly tessellate space. Embedded in an inhibitory local recurrent circuit (Couey et al., 2013), they interact with several other spatially modulated cells of the EC, for instance head direction (Ranck, 1984; Sargolini et al., 2006), border (Lever et al., 2009; Savelli et al., 2008), or speed cells (Hinman et al., 2016; Kropff et al., 2015). Moreover, they are influenced by the geometry of an animal's surrounding (Krupic et al., 2014; Wernle et al., 2017), and project to place cells (Fyhn et al., 2008; E. I. Moser and M.-B. Moser, 2008). In turn, place cells recurrently project to the EC (Bonnevie et al., 2013).

Theoretical studies suggested early on that place fields can be generated by converging grid activity of different sizes (Franzius et al., 2007; Fuhs and Touretzky, 2006; Jeffery, 2007; McNaughton, Battaglia, et al., 2006; Solstad et al., 2006). The idea is that, while a single periodic grid response leads to ambiguous localization results, superimposed grid responses of several sizes cancel each other out and lead to unambiguous localized activity that is similar to place fields even in large environments. Afterwards it turned out that grid cells organize, in

fact, in distinct modules along the dorso-ventral axis of the EC (Stensola et al., 2012): Cells within one grid module share the orientation of the hexagonal pattern, show a relative phase shift to each other, and express the same size with respect to their grid fields. Surprisingly though, the sizes of the grid fields increase from one module to the next in discrete steps with a scale factor that is suspiciously close to $\sqrt{2}$.

Grid cells are considered a vehicle to understand higher cortical representations and functionality (E. I. Moser, Roudi, et al., 2014). This is not only due to their remarkable properties, but also because they are situated at the apex of the cortical processing hierarchy (Fellemann and Van Essen, 1991). They are commonly believed to perform either of two functions. On the one hand, hexagonal grid fields were shown to be suitable for path integration (Burak and Fiete, 2009; Fuhs and Touretzky, 2006; McNaughton, Battaglia, et al., 2006). On the other hand, they theoretically outperform place cells in localization tasks (Mathis et al., 2012; Stemmler et al., 2015), and were recently shown to emerge in recurrent models for vector-based navigation (Banino et al., 2018; Cueva and X.-X. Wei, 2018). However, the difference between most existing models lies primarily in the way that either of the two is achieved (see reviews in Giocomo et al., 2011; Shipston-Sharman et al., 2016; Zilli, 2012).

There are concerns about both perspectives. First, many path integration models quickly accumulate noise and require additional mechanisms to prevent drift (Burak, 2006; Burak and Fiete, 2009), especially when deployed in real-world scenarios (Mulas et al., 2016). It was also observed that place cell activity

of preweaning rats depends on traveled distance, even though the grid cell system is still unstable during this stage of development (Bjerknes et al., 2018). Second, there is accumulating evidence that is in conflict with the idea that the grid code is a primary source for the formation of place cell activity and, thus, localization. For instance, grid cells emerge only after place cells during postnatal development (Langston et al., 2010; Wills et al., 2010). In addition, removal of projections from grid to place cells in lesion studies demonstrated that the place code is only partially influenced by the elimination of grid inputs (Chen et al., 2019; Hales et al., 2014). It also remains unclear how downstream neurons should read out the ambiguous hexagonal representation (Bush et al., 2014). Third, vector-based navigation models that use multiple scales of grid cells to represent a target vector typically generalize only up to an area that depends on the number of scales. That is, ambiguous target vectors will appear inevitably as soon as the surrounding environment with which the model has to cope is too large. Several grid coding strategies were proposed that are based on or related to residue number systems to address the issue of ambiguous responses (Fiete et al., 2008; Gorchetchnikov and Grossberg, 2007; Masson and Girard, 2010; Mathis et al., 2012; Sreenivasan and Fiete, 2011; X.-x. Wei et al., 2015). Although they significantly improve the issue, they still have a maximal distance after which the grid patterns repeat and are thus also subject to the third concern. Conclusively, the computational purpose of grid cells remains opaque and explanations of their peculiarities, namely their hexagonal grid fields as well as discrete scale increments, unsatisfying.

1.1 Contribution and organization

It was proposed previously that the functional purpose of grid cells is to represent transitions in spatio-temporal sequences (Waniek, 2018). This was the result of a top-down approach that is in contrast to many other models of grid cells. That is, first a behaviorally relevant computation was stated, namely learning and representation of sequences for goal-directed navigation. Only then were the algorithms and representations of data which are necessary to perform the computation derived under consideration of optimality. It was proved analytically that, given an Euclidean input space, a hexagonal distribution of transition encoders is optimal and minimizes the number of required neurons.

The main contribution of this paper is to demonstrate that reward-free transition representations are also consistent with the observed discretization of EC. The specific computational problem that is considered is retrieval of goal-directed sequences from memory with behaviorally relevant performance. It will be argued that a data structure, called Transition Scale-Space (TSS), with properties similar to responses in EC, namely a discretized multi-scale representation, is required to achieve optimality for this computation. Moreover, the scale-increment of spatial receptive fields between consecutive scales of this representation will be shown to be $\sqrt{2}$ when data is represented with neurally plausible receptive fields. In addition, it will be presented that also the temporal integration window needs to increase by the same factor. Following this, it will be argued that temporal buffering of inputs is required. This observation leads to a relation between the theoretical model and Theta Phase Precession

(TPP). The capability to retrieve sequences from a TSS will be demonstrated in several simulations. Specifically, simulated navigation experiments show how sequences can be retrieved from a TSS using two different modes. In addition, a simulated Morris water maze experiment will present computation of approximate solutions using higher scales of the data structure. Finally, the abstract data structure will be put into context of observations from rodent experiments. An overview of the TSS model for grid cells is depicted in Figure 13.

The paper is organized as follows. First, relevant nomenclature and concepts from Multi-Transition System (MTS) will be introduced in Subsection 1.2. For convenience, the section also contains a summary in tabular form. Then, Section 2 will present related work and highlight similarities with and differences to the ideas of this paper. Following this, Section 3 and Section 4 will provide the main contribution of the paper. The first contains the motivation and theoretical derivation of the proposed data structure, while the latter examines the data structure in simulations of navigational tasks. Subsequently, Section 5 will provide a detailed discussion about both the theoretical and experimental results. In addition, some of the related work of Section 2 will be examined in further depth. Section 6 will close the paper with a conclusion and outlook to future work.

This paper remains on an abstract algorithmic level, i.e., the second level of Marr's suggested analysis of neural computations (Marr, 1982). It does not provide simulations of spiking neural networks that implement the algorithms and data structure. In addition, it omits input-output relations from and to

other cortical areas when possible. Yet, this abstract perspective allows to derive several testable predictions for neural realizations that will be discussed in depth.

1.2 Nomenclature and concepts of Multi-Transition Systems

This paper builds on terminology, concepts, and mathematical symbols that were introduced in Waniek, 2018. For convenience, they are repeated here and the most important aspects briefly discussed. Table 1 summarizes the nomenclature and mathematical symbols. The table also provides an overview how the concepts of Multi-Transition Systems (MTS) are believed to relate to cells or networks of the Hippocampus. Further details about MTS, including formal proofs and an exhaustive discussion, can be found in Waniek, 2018.

A transition bundle $\boldsymbol{\pi} = \{\tau_1, \ldots, \tau_N\}$ is a set of N transitions τ_k. Each transition is a tuple $\tau_k = (\sigma_m, \sigma_n)$ between symbols $\sigma_m, \sigma_n \in \Sigma$ of an alphabet Σ. The *domain* of a transition contains the symbol from which a transition starts (also called source symbol), and the *image* of a transition contains the symbols to which it leads (also called target symbols). These terms are transitive to bundles, i.e., the domain (image) of a bundle is the set of symbols in which transitions of the bundle start (end). Note that, for instance in Reinforcement Learning (RL), transitions τ_k are usually viewed as functions that map one symbol to another (Sutton and Barto, 1998).

Transition bundling is restricted by certain constraints. In particular, only those transitions with mutually exclusive source and target symbols can be bundled. In spaces where arbitrary transitions between any two symbols are

possible, for instance in the case of temporal transitions between arbitrary symbols, a bundle contains only transitions that start at one source symbol. In complete metric spaces, bundling can be performed for all those transitions for which the intersection of domains and images is empty. For the two-dimensional case, Waniek, 2018 showed that bundling is maximized by a periodic hexagonal grid. In this case, transition bundles can be understood to form a dense packing of on-center and off-surround receptive fields, and the symmetry of this packing depends on the structure of the input space.

These concepts are believed to correspond to place cells as follows. Place cells represent symbols $\sigma_i \in \Sigma$ in a neural memory M_Σ and receive spatial, contextual, and possibly other cortical afferents. Of these, the spatial afferents are believed to be due to a suitable sensory representation, such as head and boundary vector cell activity, and denoted as Δ. Temporal transitions between symbols of Σ are bundled in a neural memory M_Π and allow reproduction of exact temporal sequences. As discussed in Waniek, 2018, in the absence of other afferents, any neuron in M_Π is expected to co-activate with the place cell of M_Σ for which it defines feasible temporal transitions, thereby effectively inheriting the place field.

Grid cells of a neural memory M_Γ are thought to represent spatial transitions in the following way. In contrast to temporal transition bundles, they learn spatial transition knowledge not in the symbolic space given by Σ, but directly on Δ. To achieve optimal transition bundling, each grid cell performs dense packing of on-center and off-surround receptive fields on Δ as part of its dendritic

computation, depicted in Figure 1A. As a consequence of this process, place and grid cells were predicted to have independent but related recruitment mechanisms. Each on-center is denoted as a spatial symbol $\delta_j \in \Delta$, and dense packing of spatial symbols necessarily discretizes the input space due to a finite number of neurons and dendrites. Moreover, a network of such cells should express strong Winner-Take-All (WTA) dynamics to reduce overlap of symbols and, thereby, ambiguities in the representation. Finally, each spatial transition bundle is believed to associate to symbolic representations in M_Σ via co-activation learning. That is, a grid cell learns to convey feasible spatial transitions from place cells that are active at a certain spatial location to all place cells in the surrounding area, illustrated in Figure 1B. Summarized, the hippocampal-entorhinal loop is believed to form an MTS, illustrated in Figure 1C. In an MTS, neurons of M_Σ can be understood to form nodes of a topological map, while transitions that are represented by neurons of M_Γ and M_Π are edges between these nodes.

Explicitly decoupling symbols (place cells) from spatial transition knowledge (grid cells) has several benefits. Most important, it makes place cells independent of the structure that is underlying sensory inputs. Consider an animal that wants to plan a path from its current location A to a remote target B, which means it needs to search a viable trajectory from A to B. This operation is called *expansion* of the path, denoted $A \rightsquigarrow B$, into a sequence $A, \sigma_1, \sigma_2, \ldots, B$ of $N \geq 0$ intermediate locations σ_i. Because of the decoupling, an expansion needs to iterate only through memories M_Σ, M_Π and M_Γ of Figure 1, without

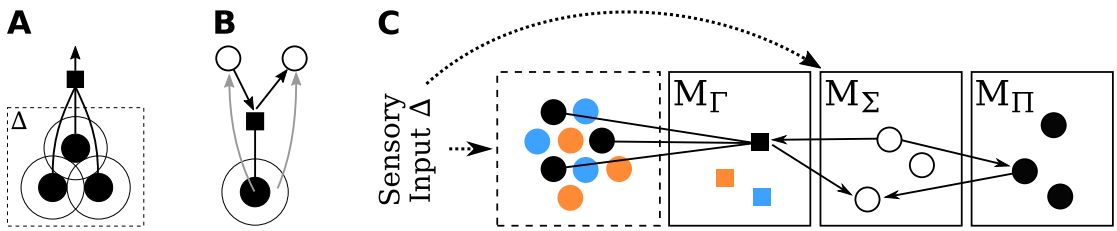

Figure 1: Conceptual overview of the hippocampal-entorhinal Multi-Transition System proposed in (Waniek, 2018). (A) A spatial transition bundle (black square) learns as many transitions as possible to maximize storage capacity of its dendritic tree (dense packing of on-center and off-surround circles). The input space on which this is performed is denoted Δ, and on-center areas are called spatial symbols $\delta_i \in \Delta$. (B) A spatial transition bundle γ learns feasible transitions in a suitable input space and conveys this information to place cells δ_i, δ_j (white circles). (C) Learning a dense coverage of Δ requires multiple spatial transition bundles (grid cells; colored squares) in memory M_Γ. These bundles convey spatial transition information to symbols σ_i (place cells; white circles) of memory M_Σ. Another memory, M_Π, records temporal transitions between symbols σ_i, σ_j, and is independent of spatially modulated input. Neurons of M_Π (black circles) are expected to show place cell characteristics. Note that not all connections from transitions to bundles are depicted to improve clarity of the illustration.

any further access to sensory representations such as Δ. Also, a currently active representation of place cells may change abruptly, for instance due to a contextual change that induces a remapping event (Muller and Kubie, 1987). Metric knowledge about feasible movements remains intact within grid cells during such an event, because grid cells are anchored to the sensory space independently from place cells as part of their dendritic computation.

Couey et al., 2013 found that interactions between neurons in MEC are predominantly inhibitory, which is indicative of a strong WTA dynamics. The prediction of a strong dependence of Cornu Ammonis 1 (CA1) on Cornu Ammonis 3 (CA3), illustrated in Figure 1C, is supported by latest evidence from rodent experiments (Davoudi and Foster, 2019). Other recent evidence underpins the prediction that grid and place cells are subject to independent recruiting processes (Chen et al., 2019). Finally, also others recently noted the functional benefits of separating (i) transitions that are recruited from sensory inputs from (ii) spatial representations that live independently of the structure of the sensory input, and provided an elegant probabilistic account of their model (Whittington et al., 2018).

2 Related work

The periodic firing fields of grid cells introduce ambiguities when used for localization. A straightforward and widely adopted solution is to use independent grid cells with multiple sizes of grid fields (Fuhs and Touretzky, 2006; Jeffery,

Symbol	Term	Correspondence or interpretation
σ_i	symbol	A place cell of CA1
Σ	symbolic space	All representations of place cells in CA1
Π	temporal transition space	All representations of place cells in CA3
$\boldsymbol{\pi}_j$	temporal transition bundle	A place cell of CA3
$\pi_{j,m}$	temporal transition	Single response m of a cell m in CA3
Γ	spatial transition space	Representations formed by all grid cells
$\boldsymbol{\gamma}_k$	spatial transition bundle	Grid cell
$\gamma_{k,l}$	spatial transition	Single grid response l of a grid cell k
δ_n	spatial symbol	Distributed representation of sensory modalities that uniquely identifies a location, e.g., via boundary vector cell activity
Δ	spatial input space	Entirety of spatial symbols
M_Σ	Memory that implements Σ	Place cells of CA1
M_Π	Memory that implements Π	Place cells of CA3
M_Γ	Memory that implements Γ	Grid cells of EC

Table 1: Nomenclature and neural interpretations. Note that indices will be dropped if clear from context.

2007; McNaughton, Battaglia, et al., 2006; Solstad et al., 2006). Converging the responses onto a single sheet of neurons, for instance using multiplication or summation, yields unambiguous singly peaked activity in environments that are significantly larger than the biggest grid period (Mathis et al., 2012). A theoretical study used constrained optimization to minimize the number of required neurons for this purpose and showed that this technique is optimal for localization when grid fields increase in discrete steps by a factor of $\sqrt{e}$, or about $1.4 - 1.7$ when neurons fire stochastically (X.-x. Wei et al., 2015). Others proposed that multiple scales of grid cells form a population vector for localization that can be read out using Bayesian inference (Stemmler et al., 2015), in which case the scale ratio was found to be ideally 3/2, or to improve representational capabilities by residue number systems (Fiete et al., 2008; Gorchetchnikov and Grossberg, 2007; Masson and Girard, 2010; Sreenivasan and Fiete, 2011). Although these values are in the range of measured scale increments, there exist doubts about the contribution of grid cells to localization (Bush et al., 2014). For instance, Hales et al., 2014 showed that the place code is only partially influenced when grid inputs were severed in lesion studies. More recently, Chen et al., 2019 presented evidence that self-motion and environmental cues have differential affects on place and grid cells, which contradicts the assumption that place cells are formed based on converging grid cell inputs.

Two other theoretical models showed that responses similar to those of grid cells can be produced by extracting temporally stable inputs (Franzius et al., 2007; Wyss et al., 2006). In these models, grid cells receive spatially modulated inputs

and self-organize their responses to learn only slowly varying features. Although this leads to multiple firing fields, and also multi-scale representations (Franzius et al., 2007), the distribution of these responses is not hexagonal. In addition, these models use feed-forward systems that are driven by grid cell activity to produce place cells. Hence, they are also contested by recent observations (Chen et al., 2019; Hales et al., 2014).

The TSS model presented in this paper is conceptually related to vector-based navigation models, and in particular to the hierarchical linear look-ahead model presented in Erdem and Hasselmo, 2014 and Erdem, Milford, et al., 2015. It also shares several ideas with the navigation model by Redish and Touretzky, 1998, which predates the discovery of grid cells, and the linear look-ahead model by Kubie and Fenton, 2012. Brief summaries of these models, as well as detailed discussions of similarities and differences between these and the TSS model can be found in Subsection 5.3. Along the same lines of these models, Bush et al., 2014 argue that grid cells could potentially be used to compute linear target vectors between locations. They base their arguments on the observation that the entirety of grid fields of a single grid cell, itself discussed to be the result of path integration, might form a constant spatial metric. In turn, this could be used to compute translational vectors. However, Bush et al., 2014 do not present a mathematical or algorithmic model for their argumentation.

The paper assumes that knowledge in the Hippocampus is represented in the form of a topological map, which was previously suggested in Dabaghian, Memoli, et al., 2012 and Dabaghian, Brandt, et al., 2014. It is also based on prior

work that derived that a hexagonal distribution of fields is optimal to minimize the number of required neurons for transition coding (Waniek, 2018).

Transition encoding in the Hippocampus was previously suggested and explored in a biologically plausible model (Cuperlier, Laroque, et al., 2004; Cuperlier, Quoy, et al., 2006). The model uses visual processing to extract landmarks, while transition cells support learning and planning of action sequences given visual stimuli. Moreover, it contains detailed biologically plausible hippocampal-prefrontal interactions. This model was evaluated on a robotics platform, and later extended via RL for the selection of an ideal trajectory to a target location (Hirel et al., 2010). Albeit influential for this paper, the authors neither addressed grid cells nor multiple scales of representation.

This work is also related to the idea that grid cells perform Principal Component Analysis (Dordek et al., 2016), or that they compute a Successor Representation (Dayan, 1993; Momennejad et al., 2016; Stachenfeld et al., 2016). The relationship and differences between these and the TSS model will be discussed in further detail in Section 5.

Gustafson and Daw, 2011 proposed that grid and place cells are modulated by geodesic distance. Their work was motivated by the idea that similarity between places should be defined locally along a path and along obstacles, and not globally due to coordinates. Moreover, the authors noted a similarity between spatial basis functions and the multi-scale representation of grid cells. Using a virtual RL agent, they showed that Euclidean distance measures can have detrimental effects on learning a value function. Specifically, the model

learns approximate values of locations by linearly combining a set of basis functions. In their model, these basis functions were pre-determined according to an oscillatory interference model of grid cells, i.e. as the sum of three cosine waves. The authors conclude that spatial representations in the brain should follow geodesic distances instead of Euclidean. In addition, they modelled how value functions should deform under geodesic considerations and made several testable predictions about the deformations of the grid code. Recently, experiments by two different labs showed that the grid response indeed deforms due to environmental influences (Boccara et al., 2019; Butler et al., 2019). MTS were motivated by the same fundamental idea that navigation needs to learn similarity between locally adjacent places and not across global distances. In contrast to the work by Gustafson and Daw, 2011, triangulation of the input space (i.e. hexagonal placement of grid cells) was not predefined but derived as an optimality result in MTS. Moreover, Waniek, 2018 hypothesized that this should emerge due to local interactions of receptive fields, and showed this using simulations of a particle system with local push-pull dynamics. It can be shown that an MTS that learns grid fields in this manner can compute the geodesic between points (see Appendix C). Finally, to create basis functions with multiple scales for their RL agent, Gustafson and Daw, 2011 used arbitrarily chosen predefined grid spacings, and required a reward structure in the environment. In contrast, MTS, as well as the extension to multiple scales presented in this paper, is reward-free. Moreover, this paper will derive the optimal scale-increments analytically for a TSS.

Recently, McNamee et al., 2016 argued from the point of information theory that neural networks should *in general* express some form of discretized scales if they have to perform optimal planning operations. Specifically, they proved that the discretization minimizes the description length of a planning task and thereby achieves efficient encodings and optimal retrieval. Unfortunately, their theoretical investigation did not link to grid cells. Nevertheless, the study points to general principles of neural computation that involve multiple scales, which is also of concern in this paper.

One contribution of this paper is to explicitly connect spatial navigation in the rodent brain to algorithms and data structures from robotics and computer science. Multi-resolution path planning has been an active field of research in robotics for several decades, and thus it is difficult to give a comprehensive but compact survey of existing techniques. However, many algorithms are based on a fundamental paper by Kambhampati and Davis, 1986, who demonstrated how to use a quad-tree to accelerate path planning. Extensions of their idea include trajectory planning in rough terrain (Pai and Reissell, 1995), representations with variable resolutions (Behnke, 2004), dynamic path planning (Lu et al., 2011), compact mesh representations (Hwang et al., 2003), or beamlets (Lu et al., 2011). These algorithms usually represent an environment topologically or with an imposed grid and use some variant of Dijkstra's algorithm (W. Dijkstra, 1959) or A* (Hart et al., 1968) to determine an optimal path from start to target. This paper falls into the same category, but extends previous work by addressing biologically plausible representations.

The work presented here is intimately related to scale-space theory, a formal mathematical framework that is widely used in the computer vision and signal processing communities (Lindeberg, 1994; Lindeberg, 2010; Witkin, 1983). One of the primary results of scale-space theory is the Gaussian pyramid, a data structure that represents an input signal in multiple scales. Each scale is computed by smoothing (or blurring) an input signal with a Gaussian kernel. It can be proved analytically not only that the Gaussian function and its derivatives are the only smoothing kernels that do not create spurious extrema, but also that the optimal scale increment for the kernels is $\sqrt{2}$ (Lindeberg, 1994). Among applications in computer vision and signal processing, Gaussian pyramids were also used to describe biologically plausible retinal and visuocortical receptive fields (Behnke and Rojas, 1998; Georgeson et al., 2007; Young, 1987). Here, an alternative ansatz for the proof of a $\sqrt{2}$ increment is presented that is rooted in building an optimal data structure for multi-resolution path planning, and thus explicitly links scale-space theory with spatial navigation in the rodent brain.

3 Scale-spaces for optimal transition look-ahead

This section examines the computational complexity of transition encoding during retrieval, and how this relates to behavioral necessities. Then, optimal acceleration of such retrievals using a scale-space structure is presented. Although provided in the context of spatial navigation, the results generalize to other transition systems. For instance, temporal transition systems for episodic memories

are expected to form scale-space structures along appropriate dimensions.

Retrieval of sequences from a memory M_Σ requires activating a specific source symbol, and monitoring when a target symbol activates. This form of in- and output is expected to be the result of prefrontal-hippocampal interactions, similar to the prefrontral cortex columns and their simplifications to reward cells in the work by Erdem and Hasselmo, 2012 and, respectively, Erdem and Hasselmo, 2014. However, details about such interactions are unimportant for the scale-space structure and its operations that are presented in this section, and are therefore omitted.

3.1 Computational complexity of retrievals and pre-play

Consider an MTS $\mathcal{M}$ that is used for navigational purposes in an idealized two-dimensional input space Δ. $\mathcal{M}$ consists of a finite number of symbols in a memory M_Σ, and finite numbers of spatial and temporal transitions in memories M_Γ and M_Π, respectively. The modular organization of $\mathcal{M}$ is depicted in Figure 1C, and Figure 2B emphasizes the recurrent connectivity between these memories exemplary for M_Σ and M_Γ. Moreover, $\mathcal{M}$ is subject to the following simplifications: (i) any uncertainty in the representation of the input space is ignored and (ii) spatial transitions organize according to a WTA mechanism. Then, spatial transitions perform a Voronoi tessellation that leads to hexagonal fields, illustrated in Figure 2A. These simplifications will be relaxed in Subsection 3.3. Finally, assume that the environment was explored sufficiently long and symbols and transitions were learned and stored in $\mathcal{M}$ accordingly.

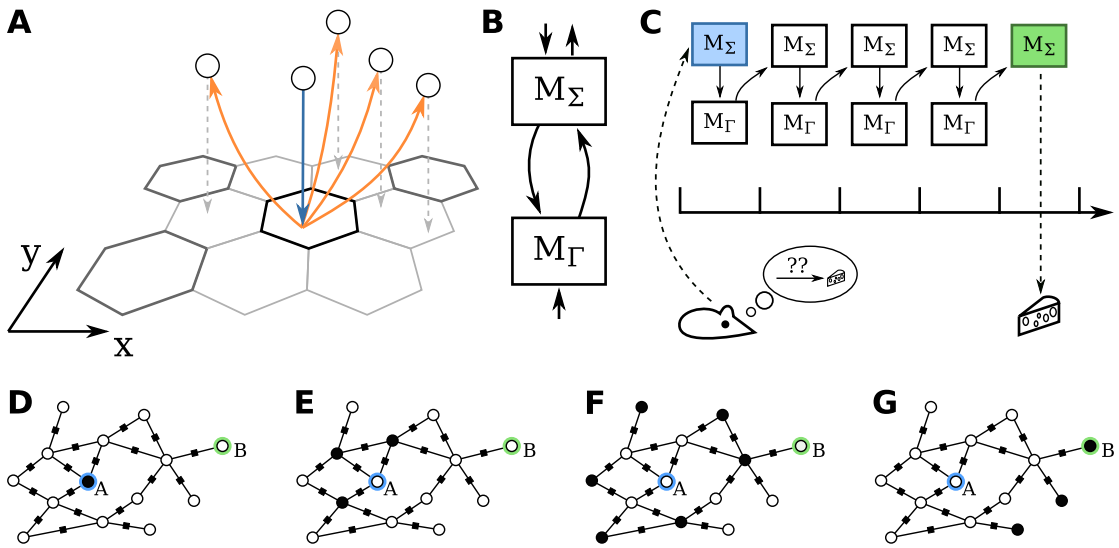

Figure 2: (A) Discrete Voronoi clustering on a suitable input space (indicated by the x-y axes) with absolute WTA dynamics yields hexagonal fields. A spatial transition (black hexagon) associates with all symbols that are co-active with its *domain* (circle with blue arrow) during exploration, and learns transitions (orange arrows) to symbols in its image (circles with gray dashed arrows). Dark gray, periodically repeating hexagons indicate the bundle to which the spatial transition belongs. (B) Memories of a MTS are recursively connected, shown only for M_Σ and M_Γ. (C) Given activation of a starting symbol (indicated by blue color of M_Γ) on a linear track that was discretized into bins (black arrow with vertical bars), mentally traveling to a target involves recursively activating M_Σ and the transition memories M_Π and M_Γ (only M_Γ shown here) until the target symbol is active in M_Σ (indicated by a green box). (D-G) Propagating wave of activity (black circles) in a graph when searching if a connected path exists between symbols A and B. Each edge involves a spatial transition, indicated by a small black box.

Determining the existence of a path between two symbols depends linearly on the number of recursive computations. In terms of $\mathcal{M}$, the path $A \rightsquigarrow B$ needs to be expanded into a sequence $A, \ldots, \sigma_i, \ldots, B$ of intermediary symbols σ_i, which can be achieved as follows. First, the start symbol is activated in M_Σ. Then, (indirect) recursive propagation of activity via memories M_Γ and M_Π will eventually activate B, if an expanded path from A to B exists. This simple algorithm can be depicted as a propagation algorithm on a graph because the data stored in an MTS forms a topological space (see Figure 2D-G). Moreover, the activity propagation is a flooding algorithm, which can be easily implemented in distributed processing systems (Raynal, 2013). The worst case scenario for this algorithm is when A and B are on the two ends of a linear track, in which case recursive retrievals via M_Π and M_Γ coincide. The runtime complexity resides in $O(N)$, where N is the number of symbols on the shortest path between A and B.

The linear dependency on N is problematic especially when temporal dynamics of real neural networks are respected. Consider an implementation of $\mathcal{M}$ that consists of neural associative memories for M_Σ, M_Γ, and M_Π, and with temporal dynamics that consume about 10 ms per memory for spike generation, propagation, and axonal delays. Furthermore, let A and B be on a linear track with distance $D = 200\,\text{m}$ and grid field size $d_g = 20\,\text{cm}$. As above, recursive expansions through M_Γ and M_Π coincide due to linearity of the track. The expanded path from A to B consists of 1000 intermediate invocations of the $M_\Sigma - M_\Gamma$ loop. In total, this consumes 20 s for mentally traveling from A to B via recursive expansion. Figure 2C shows an illustration of the recursive retrieval

using memories M_Σ and M_Γ. The runtime will increase further when there is a set of candidate sequences from which the animal has to chose, for instance with some reward propagation mechanism or graph search, and especially with stochastic neural responses. Moreover, the expansion must be repeated for each novel target. This behaviorally questionable performance for model-based algorithms was previously noted in the context of grid cells by Dordek et al., 2016 and Stachenfeld et al., 2016. Summarizing, recursive retrieval in a single-scale transition system has a time complexity that is impractical from a behavioral perspective.

Another problem is the combinatorial complexity. Recall that transition systems induce a graph, in which vertices correspond to states or symbols, and edges to transitions between symbols. The number of vertices and edges in such a graph grows exponentially with the number of dimensions of the input space. Moreover, MTS assume constant cost to move from one symbol to another, which means that the weights of all edges are constant. Thereby, the number of feasible paths between two vertices, but also the actual runtime for shortest-path computations, grows similarly. In principle and given the assumption that each vertex of the graph is an independent computational unit, such shortest path computations can be parallelized easily using flooding algorithms (Lynch, 1996; Raynal, 2013). Then, shortest-path computation corresponds to a traveling wave of messages from source to target. Although this reduces searching to the linear case that was described above, it is still affected by the aforementioned problematic runtime complexity.

3.2 Optimal searches in discrete transition systems

This section presents how to optimally accelerate retrievals of sequences from a discrete MTS, denoted $\mathcal{M}_\mathrm{d}$. Assuming discrete representations, optimal acceleration can be achieved with existing and widely adopted data structures and algorithms. This section therefore briefly reviews those data structures that are of particular interest in the context of MTS and explains how they apply to $\mathcal{M}_\mathrm{d}$.

Skip lists are a data structure that can be used to accelerate searching an element in linked lists of data. The acceleration is achieved by a hierarchy of *fast lanes* that jump over elements of the original list (Pugh, 1990). Searching an element in a skip list starts on the highest (or coarsest) level of the hierarchy and proceeds until an element is found that is larger than the target. Subsequently, the search drops down to the next finer level of the hierarchy until the element is recovered. Given uniformly sampled data, skip lists are dual to binary search (Dean and Jones, 2007) and thus become asymptotically optimal (Knuth, 1998). Specifically, the hierarchy of fast-lanes improves operations on equidistant data exponentially and retrievals, insertions, and deletions can be performed in $O(\log N)$ time with high probability. Due to their favorable properties, they found widespread application in various applications and are still actively researched (see for instance Bender et al., 2017 and references therein). Moreover, skip lists generalize to higher dimensions (Eppstein et al., 2005). Searching in a skip list is illustrated in Figure 3A.

Skip lists apply to MTS as follows. Observe that the information that is stored

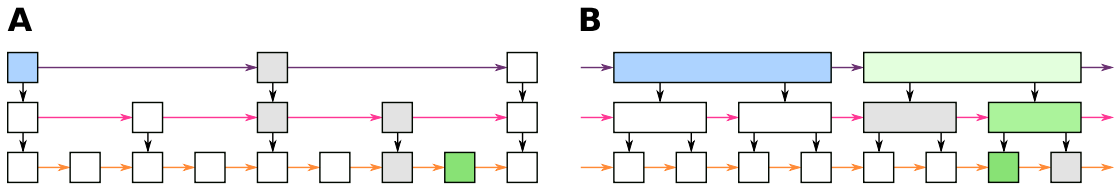

Figure 3: (A) A skip list is a multi-layer linked list data structure (Pugh, 1990). Elements of the list must be ordered. Higher layers have fast lanes to skip over elements. Given a start (blue), searching an element (green) in a skip list starts in the highest layer. Skip links are followed until an element on the corresponding layer is found which is larger than the target. Then, the search is continued on a lower level (see gray boxes). (B) In contrast to skip lists, interval skip lists (Hanson, 1991) store intervals in higher layers. Searching an element is similar to skip lists, but tests if the target element is in an interval (indicated by larger boxes), and subsequently drops down in the hierarchy. Interval skip lists are related to segment trees (see de Berg et al., 1997).

in an MTS forms linked lists of temporally and spatially ordered elements (see Figure 2D-G). Hence, searching specific elements in an MTS can be accelerated with a skip list. Moreover, transitions of an MTS densely and uniformly cover the input space, which is required for asymptotic optimality. However, only finding a certain element in accumulated knowledge is insufficient for most meaningful behavior, in particular during spatial navigation. In such a case, a path $A \rightsquigarrow B$ needs to be expanded into all intermediary symbols.

Retrieval of all intermediary items between two elements can be computed with *interval queries*. Given a suitable data structure $\mathcal{D}$, for instance an array of elements, an interval query $\text{iq}(\mathcal{D}, q_s, q_e)$ returns all intervals in $\mathcal{D}$ that are

between query start q_s and query end q_e. Like binary search, interval queries can be performed in logarithmic time with suitable data structures, such as segment trees (de Berg et al., 1997), or interval skip lists (Hanson, 1991). Figure 3B depicts searching an element using an interval skip list. Answering $\mathrm{iq}(\mathcal{D}, q_s, q_e)$ is performed by first locating the interval that contains q_s. Then, intermediate intervals are retrieved until the interval that contains q_e is found. This is typically performed on the highest available level for performance reasons. Intermediate low-level values can be extracted by traversing the lower levels of the data structure on the chain of higher-level intervals. Thus, reporting all intermediate K values between q_s and q_e can be performed in $O(logN + K)$ time. Moreover, segment trees or interval skip lists can be built bottom up in $O(N)$ time (de Berg et al., 1997). Each additional layer in the tree or list contains nodes with intervals that enclose the nodes of the lower level, illustrated in Figure 3B. Likewise binary trees, segment trees and interval skip lists generalize to higher dimensions.

The construction procedure for interval skip lists directly translates to an optimal discrete Transition Scale-Space $\mathcal{M}_{\mathrm{d}}^{\mathsf{S}}$, in the following presented in terms of spatial transitions. Let scale 0 be the lowest level of $\mathcal{M}_{\mathrm{d}}^{\mathsf{S}}$. On scale 0, the domain of a spatial transition γ_i^0 corresponds to a spatial symbol δ_i. Its image is given by symbols surrounding δ_i, for instance δ_{i-1} and δ_{i+1} in a one-dimensional setting. The domain of a spatial transition $\gamma_i^{(n)}$ on scale $n > 0$ corresponds to the integrated domains of 2 adjacent transitions of the previous scale along each dimension, where 2 follows from constructing an optimal search data structure (de Berg et al., 1997; Knuth, 1998; Pugh, 1990). Likewise, the image of

$\gamma_i^{(n)}$ grows by a factor of 2 along each dimension with respect to scale $n-1$ and encloses all symbols that surround the domain of $\gamma_i^{(n)}$.

The scale-space construction is illustrated in Figure 4. Panel A recalls the on-center and off-surround properties of transitions. Panel B illustrates the transitions as elements of two conjoint interval skip lists. This not only emphasizes the relationship to the data structure, but also reduces visual clutter of the illustration. Panel C depicts two scales of a discrete TSS $\mathcal{M}_{\mathrm{d}}^{\mathsf{S}}$. Note that the construction process is, in principle, independent for each scale of the hierarchy. That is, each scale can be constructed without interactions from other scales when random access to afferents from Δ and Σ is provided. Finally, Panel D illustrates the on-center and off-surround properties of a transition likewise panel A, but for the case of scale $s > 0$ in two-dimensions. Importantly, a transition on this scale leads to all symbols in its image simultaneously.

A illustration that further simplifies the depiction of transitions is presented in Panel A of Figure 6. The panel plots only on-centers of receptive fields of transitions γ, and omits all arrows from transitions to symbol spaces.

3.3 Optimal searches in transition systems with biologically plausible representations

The previous section reviewed algorithms for optimal acceleration of sequence retrievals for discrete data, and introduced how these algorithms relate to MTS. Specifically, the set of spatial transitions in a discrete TSS can be seen as two conjoint interval-skip lists, one list operating on elements of Δ, the other on

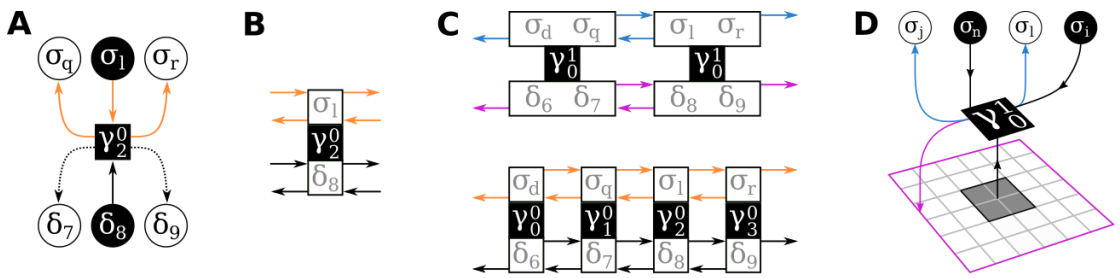

Figure 4: Scale-space construction and co-activation learning in discrete spaces. (A) Recall that a spatial transition represents transitions in Δ and, simultaneously, in Σ (cf. Figure 1). While spatial symbols $\delta_i \in \Delta$ need to be spatially neighboring (consecutive indices), symbols $\sigma_j \in \Sigma$ need to be temporally adjacent. In a neural realization, solid arrows are hypothesized to be axonal connections, dashed arrows could be the result of on-center and on-surround dynamics. (B) Each transition $\gamma_k \in \Gamma$ can be depicted as an element of two conjoint interval skip lists (cf. Figure 3). (C) The domains of transitions on scale 0 of a discrete Transition Scale-Space $\mathcal{M}_{\mathrm{d}}^{\mathrm{S}}$ contain singular elements of the corresponding input spaces (bottom row). For optimality, transitions on a scale $s > 0$ merge the domains of two transitions of scale $s - 1$ (top row, shown for scale $s = 1$). (D) Two dimensional example that emphasizes the on-center (entire gray shaded area, bottom row) and off-surround (entire area enclosed by magenta border, excluding gray shaded region, bottom row) receptive field of a transition (black box) and co-activation learning symbols σ (top row: co-active symbols shaded black, transition leads other symbols via blue arrows). The gray grid is depicted for illustrative purposes only and to highlight the increase of the size of domain and image (cf. interval skip list, Figure 3B).

Σ (Figure 4C). In the discrete case, classical optimality results require that the intervals in such data structures double in size from one level of the hierarchy to the next. However, responses of neurons typically follow bell-shaped tuning curves (Butts and Goldman, 2006; Jazayeri and Movshon, 2006). Hence, this section will address biologically plausible representations.

Recall that an MTS bundles transitions to minimize the number of required neurons. To this end, spatial transition bundles densely pack on-center and off-surround receptive fields (see Subsection 1.2 or Waniek, 2018), where each center-surround field corresponds to a feasible spatial transition from one location to its surrounding region. Especially important here is that each on-center and off-surround receptive field can be treated as an individual component that is part of a bundle. Consequently, the following will examine a single center-surround field in isolation. To distinguish receptive fields (individual spatial transitions) from spatial symbols and bundles, they will be denoted as a function parameterized on the spatial symbol, i.e., $f(x; \delta_j)$. This also emphasizes the preferred stimulus δ_j of a receptive field.

Classically, on-center and off-surround receptive fields were modelled in the context of retinal ganglion cells (Koenderink and van Doorn, 1990; Rodieck, 1965). There, the components of the receptive fields were combined either linearly (Marr and Hildreth, 1980; Rodieck, 1965), for instance with a Difference-of-Gaussians (DoG) approach, or with other mechanisms (Furman, 1965; Grossberg, 1970). These approaches can be combined into one formal framework (Neumann et al., 1999). Most important, the center and surround portions of the receptive field

can be modelled independently of each other.

Using the results of modeling center-surround fields in other cortical areas, the following will treat the center and surround in isolation. Specifically, the on-center and off-surround properties of a spatial transition are modelled as a DoG (Figure 5A-B). That is, the receptive field of a spatial transition is given by a function

$$f_{\delta_j}(x) = f(x;\delta_j) = f^{(c)}(x;\delta_j) - f^{(s)}(x;\delta_j) \quad , \tag{1}$$

where $f^{(c)}$ and $f^{(s)}$ are the center and surround, respectively, and δ_j is the preferred stimulus of the receptive field.

To simplify the analysis, the receptive field components are assumed to be isotropic Gaussians. That means that they are symmetric, and that the Gaussians are linearly separable in higher dimensions. Thereby, the results that are derived below for one dimension generalize to higher dimensions. Finally, these properties are shared amongst all receptive fields (individual spatial transitions) of all spatial transition bundles.

The on-center $f^{(c)}$ of a spatial transition on the smallest scale $s = 0$ of transitions determines a highest possible resolution with which elements $\delta \in \Delta$ can be distinguished. More precisely, all transitions necessarily adhere to the Cramér-Rao bound, which presents a lower bound with which a decoding scheme can extract information from pre-synaptic inputs (Cover and Thomas, 2006; Pilarski and Pokora, 2015). This bounded resolution depends on the finite number of presynaptic neurons as well as the local (dendritic) decoding scheme. Conclusively, it is sufficient to consider the size of $f^{(c)}$ on scale 0 to derive

receptive field sizes on higher scales. Here, the size (or width) of $f^{(c)}$ on scale 0 is parameterized by a constant scalar $\mathfrak{s}_0$ in the one-dimensional case, and a constant matrix $\mathfrak{S}_0$ in higher dimensions. Likewise other properties, this bound is assumed equal across all transition bundles to simplify analytical treatment.

Put together, the on-center component of a receptive field with preferred stimulus $X := \delta_j$ on the smallest scale is given by a Gaussian function, i.e.,

$$f_X^{(c)}(x) = f^{(c)}(x; \delta_j) = \mathcal{N}(x; \mu_X, \mathfrak{s}_X^2) \tag{2}$$

$$= \frac{1}{\sqrt{2\pi}\mathfrak{s}_X} \exp\big(-\frac{(x-\mu_X)^2}{2\mathfrak{s}_X^2}\big) \tag{3}$$

where $\mathfrak{s}_X^2 = \mathfrak{s}_0^2$ denotes the variance, and the mean $\mu_X = \mu(\delta_j)$ depends on the preferred stimulus δ_j.

Recall the constructive process to build an additional scale in an interval segment tree, described in Subsection 3.2. Assuming uniformly distributed data, the interval on a scale $s > 0$ has to integrate both intervals of the scale $s - 1$ to achieve optimality. Moreover, this new interval is centered between the two smaller intervals *by construction*. In terms of receptive fields, the on-center region on a scale $s > 0$ thus needs to merge the representations of the on-center regions of two smaller receptive fields. Intuitively, this can be understood as follows. The receptive field on the higher scale needs to determine if arriving input falls into either of the smaller intervals. Clearly, this form of mixing two functions is a convolution (Seeger and Volle, 1995). Another way to see this is to realize that the on-center region $f_X^{(c)}$ describes a random variable X. Combining two receptive fields thus corresponds to adding two random variables X and Y, which requires the convolution of their probability densities. That is, the probability density of

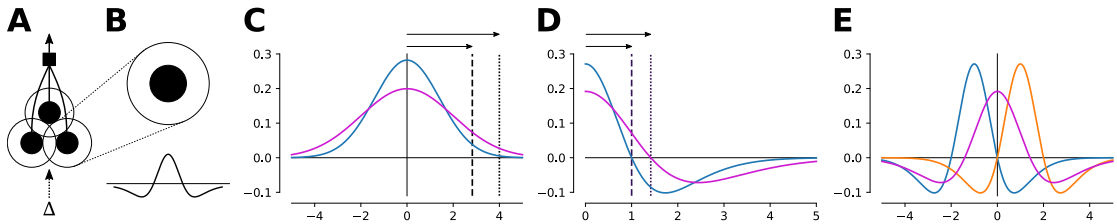

Figure 5: (A) Each spatial transition bundle receives afferents from a spatially modulated input space Δ and densely packs on-center and off-surround receptive fields. (B) Each receptive field can be treated independently. Here, they are modelled as Difference of Gaussians (DoG). (C) Optimal construction of the scale-space structure doubles the variance $\mathfrak{s}_s^2$ of the on-center Gaussian. This scales the width by $\sqrt{2}$, marked here for the $2\mathfrak{s}_s$ boundary (dashed and dotted lines). (D) Scaling the standard-deviation of a DoG by $\sqrt{2}$ also scales the roots by $\sqrt{2}$, shown here for the positive domain of a DoG of a DoG with $\mathfrak{s}_0 = {}^1/_{2e}$ (blue) and $\mathfrak{s}_1 = \sqrt{2}\mathfrak{s}_0$ (magenta). (E) Convolution of two consecutive DoGs X and Y (blue, orange), parameterized by $\mathfrak{s}_0$ and $\mu_X = -\mu_Y$, leads to a DoG that is parameterized by $\mu_Z = 0$ and $\mathfrak{s}_1 = \sqrt{2}\mathfrak{s}_0$ (magenta).

$X + Y$ follows from $f_X^{(c)} * f_Y^{(c)}$, where $*$ denotes the convolution operator.

This insight is now used to derive the on-center function $f_Z^{(c)}(x)$ for a spatial transition on scale $s = 1$. Let $\mathcal{F}$ be the Fourier transform, and $\mathcal{F}^{-1}$ its inverse, and let $f_X^{(c)}$ and $f_Y^{(c)}$ be the on-center components of two consecutive receptive fields on scale 0. That is, they are given by

$$f_X^{(c)}(x) = \mathcal{N}(x; \mu_X, \mathfrak{s}_X^2) \quad , \qquad f_Y^{(c)}(x) = \mathcal{N}(x; \mu_Y, \mathfrak{s}_Y^2) \quad , \tag{4}$$

where $\mathfrak{s}_X = \mathfrak{s}_Y = \mathfrak{s}_0$. In addition, note that

$$\mathcal{F}\left\{f_X^{(c)}\right\} = \mathcal{F}_X(\omega) = \exp\left(-i\omega\mu_X\right)\exp\left(-\frac{\mathfrak{s}^2\omega^2}{2}\right) \quad . \tag{5}$$

Then, using the convolution theorem, it follows that

$$\begin{aligned}
f_Z^{(c)}(x) &= (f_X^{(c)} * f_Y^{(c)})(x) && (6)\\
&= \mathcal{F}^{-1}\left\{\mathcal{F}\left\{f_X^{(c)}\right\}\cdot\mathcal{F}\left\{f_Y^{(c)}\right\}\right\}(x) && (7)\\
&= \mathcal{F}^{-1}\left\{\exp\left(-i\omega\mu_X\right)\exp\left(-\frac{\mathfrak{s}_X^2\omega^2}{2}\right)\exp\left(-i\omega\mu_Y\right)\exp\left(-\frac{\mathfrak{s}_Y^2\omega^2}{2}\right)\right\}(x) && (8)\\
&= \mathcal{F}^{-1}\left\{\exp\left(-i\omega(\mu_X+\mu_Y)\right)\exp\left(-\frac{(\mathfrak{s}_X^2+\mathfrak{s}_Y^2)\omega^2}{2}\right)\right\}(x) && (9)\\
&= \frac{1}{\sqrt{2\pi}\mathfrak{s}_Z}\exp\left(-\frac{(x-\mu_Z)^2}{2\mathfrak{s}_Z^2}\right) && (10)\\
&= \mathcal{N}(x;\mu_Z,\mathfrak{s}_Z^2) && (11)
\end{aligned}$$

where

$$\mathfrak{s}_Z = \sqrt{\mathfrak{s}_X^2+\mathfrak{s}_Y^2} = \sqrt{2\mathfrak{s}_0^2} = \mathfrak{s}_0\sqrt{2} \quad . \tag{12}$$

Furthermore, $\mu_Z = \mu_X + \mu_Y$. By construction, $\mu_X = -\mu_Y \Rightarrow \mu_Z = 0$.

The above results yield the following characteristic properties of receptive fields on higher scales, and of transition bundles to which they belong.

First, assume that the response is determined only by the on-center region of receptive fields that are modelled as DoG. As above, consider a single receptive field in isolation. The response is characterized by a Probability Density Function (pdf) in the form of a Gaussian function. On the smallest scale $s = 0$ of the data structure, this pdf is parameterized by variance $\mathfrak{s}_0^2$ and the mean follows from the preferred stimulus. According to Equation 12, the variance on scale $s = 1$ is

$\mathfrak{s}_1^2 = (\mathfrak{s}_0\sqrt{2})^2$. Recursively applied for higher scales, this yields

$$\mathfrak{s}_s = \mathfrak{s}_{s-1}\sqrt{2} = \mathfrak{s}_0(\sqrt{2})^s \quad . \tag{13}$$

Because the Gaussians were assumed to be isotropic, this result generalizes to higher dimensions. That is, the covariance matrix $\mathfrak{S}_0$ of the multi-variate Gaussian is a diagonal matrix of the form $\mathfrak{S}_0 = \mathrm{diag}(\mathfrak{s}_0^2)$ and higher scales follow according to Equation 13. Intuitively, the result means that the integration area of the receptive field, determined by the variance of the representation, doubles. Finally, assume that there is only a response if the input stimulus falls within $l\mathfrak{s}_s, l > 0, l \in \mathbb{R}$, for instance for $l = 2 \rightarrow 2\mathfrak{s}_s$. By Equation 13, the width (or radius) of the receptive field also increases by $\sqrt{2}$ from one scale to the next. This property is illustrated in Figure 5B.

Now, assume that the response on scale s is determined jointly by on-center and off-surround dynamics of a DoG. Then, the field is defined by $f_{\delta_j}(x) = f(x;\delta_j) = f^{(c)}(x;\delta_j,\mathfrak{s}_s^{(c)}) - f^{(s)}(x;\delta_j,\mathfrak{s}_s^{(s)})$. Because both $f^{(c)}$ and $f^{(s)}$ are Gaussians, they relate to each other by $\mathfrak{s}_s^{(s)} = K\mathfrak{s}_s^{(c)}$ with $K > 1$. To improve terseness, the super-script $^{(c)}$ will be dropped in the following, i.e., $\mathfrak{s}_s := \mathfrak{s}_s^{(s)}$. It is straightforward to analytically derive that a (one-dimensional) DoG has roots at

$$z_{0,1} = \pm\frac{2K^2\mathfrak{s}_s^2}{(1-K^2)}\sqrt{-4\log(K)\frac{(1-K^2)}{2K^2\mathfrak{s}_s^2}} \quad . \tag{14}$$

From this, it follows immediately that $r\mathfrak{s}_s \Rightarrow rz_{0,1}$. That is, scaling the standard deviation $\mathfrak{s}_s$ by a factor $r \in \mathbb{R}_+$ scales the roots also by r, irrespective of the value of K. An example of this is depicted in Figure 5D. Note that this result also generalizes to higher dimensions, again because of the assumed isotropy of the

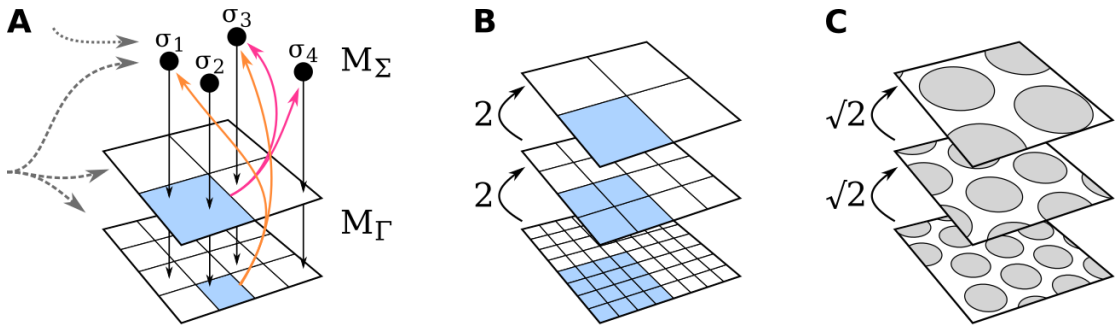

Figure 6: (A) Simplified illustration of learning transitions in a scale-space. Inputs to the Transition Scale-Space $\mathcal{M}_{\mathrm{d}}^{\mathsf{S}}$ are indicated as gray dashed arrows. Transitions of a memory M_Γ learn their domains on the input space according to their scale (blue boxes), and relay this structural information to symbols $\sigma \in \Sigma$ of memory M_Σ (black dots). In this example, a transition that is defined for σ_2 leads to σ_1 and σ_3 on the lowest scale (orange arrows), whereas it leads to σ_3 and σ_4 on scale 1 (magenta arrows). (B) In the discrete case, the optimal scale increment of the domain (receptive field) of transitions is 2 along each dimension. In two-dimensions, illustrated here, this leads to the well known quad-tree data structure. (C) In stochastic input spaces that can be described with normally distributed pdfs, the optimal scale increment from one scale to the next is $\sqrt{2}$.

Gaussians. To conclude, the width (or radius) of the response increases according to the scaling factor r. Following the results from above, optimally constructing the data structure by convolution yields $r = \sqrt{2}$, and thus the on-center region of receptive fields increase by $\sqrt{2}$, illustrated in Figure 5E.

Figure 6B and C depict simplified illustrations of discrete and a probabilistic scale-spaces.

Finally, these properties are expected to also increase the grid period by $\sqrt{2}$.

One reason is that in an optimal interval skip list, a transition leads to neighboring intervals that have the same size (cf. Figure 3B). Another reason is that in an MTS, each (spatial) bundle densely packs transitions to maximize dendritic capacity (cf. Subsection 1.2). Given the off-surround area that is required to avoid violation of logical considerations of transition bundling, dense coverage of an entire input space requires multiple bundles. Following arguments in Waniek, 2018, an entire network of bundles is expected to reduce overlap of on-center regions by competition to avoid ambiguities. Hence, these competitive interactions of transition bundles are conjectured to generate a push-pull mechanism that leads to grid periods that depend on $\mathfrak{s}_s^{(c)}$ and increases the grid period consecutively by $\sqrt{2}$. Intuitively, while a single transition bundle tries to densely pack receptive fields, competing transition bundles push them apart to insert their own receptive fields. Conclusively, this generates a dense coverage of the spatial input space Δ with on-center receptive fields of the same size, and a grid period that also follows from the size of on-center areas.

3.4 Temporal integration, buffering, and bottom-up learning

The previous sections introduced the Transition Scale-Space (TSS) and derived the increment of the spatial receptive field analytically. Given random access to symbols $\sigma_i \in \Sigma$, the data structure can be computed offline. It is, however, unlikely that an animal has this random access and must learn the data structure bottom up. Hence, this section examines how the temporal integration window changes between scales, and how on-line learning can be implemented.

Any $\gamma \in \Gamma$ represents transitions in the spatial input space Δ and the symbolic space Σ simultaneously. Figure 4 illustrates this as two conjoint interval skip lists. That means that each $\gamma \in \Gamma$ has a spatial domain dom_Δ and image im_Δ in Δ, but also a domain dom_Σ and image im_Σ in Σ. The previous sections showed that, to learn a transition $\gamma^{(s)}$ on a scale $s > 0$, $\gamma^{(s)}$ needs to ingrate multiple spatial symbols of Δ. Certainly, this also applies to afferents from Σ.

Learning a transition $\gamma^{(s)}$ on scale $s > 0$ requires to integrate multiple symbols from Σ. However, recall that symbols $\sigma_i \in \Sigma$ were hypothesized to be recruited independently from the same spatial afferents as spatial symbols $\delta_k \in \Delta$. In turn, this means that they might not be temporally co-active with $\gamma^{(s)}$. Therefore, the temporal integration window for dom_Σ of $\gamma^{(s)}$ needs to increase. Moreover, the symbols need to be presented in temporal order to preserve their spatio-temporal relationship. That is, suppose there exists a path of N symbols $\sigma_1, \ldots, \sigma_N$. This means that symbols σ_i and σ_{i+1} are not only spatially related, but also temporally adjacent.

Activating symbols $\sigma_i \in \Sigma$ of a path in their temporal order induces a linked list in time, depicted in Figure 7A. Then, increasing the temporal integration window from one scale to the next follows immediately according to Subsection 3.2 and Subsection 3.3.

Specifically, for the discrete case, the receptive field for the temporal integration can be modelled as a Haar wavelet, illustrated in the top row of Figure 7B. The positive part of the wavelet corresponds to the start of the transition between symbols of Σ, and the negative part its end. To remain consistent with the size

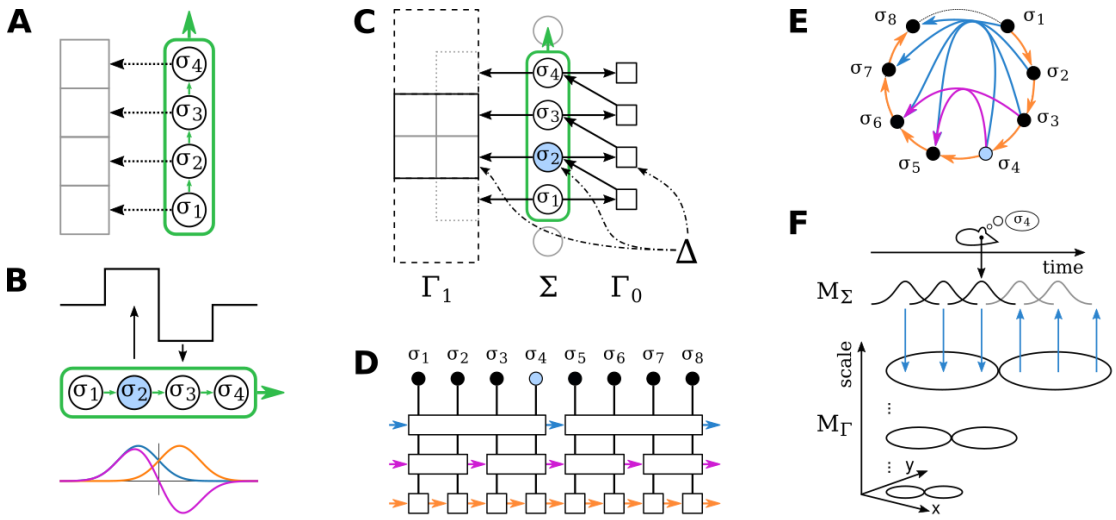

Figure 7: On-line learning of the Transition Scale-Space with temporal buffering. (A) Consecutive activation of symbols $\sigma_i \in \Sigma$ induces a linked list in time (green box). This preserves temporal (small green arrows) and spatial structure (indicated as arrows to discrete spatial areas). (B) Receptive fields of transitions for temporal integration as Haar wavelet (discrete case, top row) and Difference of mean-shifted Gaussians (biologically plausible representations, bottom row) on scale 0. Temporal transition start and end indicated as black arrows. (C) A transition on scale $s > 1$ (left) needs to associate its on-center (black solid box) with multiple symbols of Σ (middle). To keep their spatio-temporal structure, they are activated in sequence (green box) by iterating scale $s = 0$ of the scale-space (right). Black dashed box indicates spatial off-center (partial area only for visual purposes). (D) Simplified depiction of (C) for three scales. (E) The temporal buffer can be implemented as circular buffer. (F) Multi-scale illustration for biologically plausible receptive afferents. For further explanations for each panel, see Subsection 3.4.

increment of the spatial receptive field from one scale to the next, the temporal integration window thus has to increase by a factor 2.

Now consider biologically plausible representations. Likewise spatial afferents in Subsection 3.3, the activity of a symbol $\sigma \in \Sigma$ is assumed to be Gaussian. Then, the temporal receptive field of a transition $\gamma \in \Gamma$ can also be modelled as Gaussians, for instance as the first derivative of a Gaussian, or as a difference of mean-shifted Gaussians (see bottom row of Figure 7B for an illustration of the latter). The scaling of the temporal receptive field can thus be derived analytically according to Equation 6 – Equation 11. Conclusively, the variance of the temporal integration window doubles, which means, following the same arguments as in Subsection 3.3, that its width increases according to $\sqrt{2}$ from one scale to the next.

It remains to understand how symbols $\sigma_i \in \Sigma$ can be presented to transitions $\gamma^{(s)} \in \Gamma$ on scale $s > 0$ in temporal order and in a suitable temporal window. It is unlikely that spatial inputs activate symbols σ_i in the correct manner. The reason is that symbols $\sigma_i \in \Sigma$ were hypothesized to be recruited on spatial afferents Δ and that only one (or a few) symbols $\sigma_i \in \Sigma$ are active at any single place (Waniek, 2018). That is, spatial input will activate only singular symbols. However, observe that scale $s = 0$ of the TSS learns transitions on singular elements of both Δ and Σ. Then, iterating the smallest scale of the TSS will induce a suitable temporal buffer.

More precisely, suppose that spatial inputs from Δ arrive at a given frequency at Σ and Γ. This frequency determines a time window T_Δ between consecutive

inputs from Δ. Further assume that T_Δ is longer than the time it takes to iterate the transition system on the smallest scale. Then, it is possible to nest multiple such iterations in T_Δ. This creates a spatio-temporally ordered list of symbols $\sigma_i \in \Sigma$, or in other words, a temporal buffer. In turn, the temporal buffer can be used to associate higher scales. This process is depicted in Figure 7C with interactions between Σ and the two spatial transition scales Γ_0 and Γ_1. Figure 7D shows a simplified illustration for three scales. Changing the illustration such that the ends of the temporal buffer are connected reveals that this form of temporal buffering induces an oscillation around spatial input from Δ, depicted in Figure 7E. In other words, it can be implemented as a ring buffer that gets updated by novel inputs arriving from Δ. Figure 7F illustrates the interaction between temporally adjacent symbols and the spatial scale-space.

3.5 Sequence planning and shortcut discovery with Transition Scale-Spaces

This section answers how an animal can produce sequences between two arbitrary symbols σ_s and σ_t using a scale-space representation $\mathcal{M}^{\mathsf{S}}$. The presented multi-resolution approaches are similar to a method that was previously described by Behnke, 2004 in the domain of robotic navigation, and related to the hierarchical look-ahead model presented in Erdem and Hasselmo, 2014; Erdem, Milford, et al., 2015. Moreover, the described methods are graph search algorithms and based on well-known techniques such as Dijkstra's algorithm (W. Dijkstra, 1959) or flooding algorithms (Fishkin and Barsky, 1985; Lynch, 1996; Silvela and

Portillo, 2001).

As in the previous sections, transitions are treated irrespective of the transition bundle to which they belong. This simplifies the description without violation of the logical considerations of MTS and, thus, improves clarity of the presentation. To further reduce complexity, the temporal transition memory M_Π will be omitted. Finally, assume that the memories already formed, i.e., the scale-space was already built and corresponding symbols assigned to transition images and domains on each scale.

Consider an $\mathcal{M}^{\mathsf{S}}$ with only a single scale. Finding a sequence from a symbol σ_s to σ_t requires only few operations. First, σ_s must be activated in M_Σ. Then, a corresponding transition γ_i that is defined for σ_s will be activated in M_Γ. In turn, γ_i activates a set $\{\sigma_j\}$ of subsequent symbols σ_j to which γ_i leads to. The combination of these two steps is called *expansion*. If there exists at least one path from σ_s to σ_t, recursively repeating this process will eventually activate σ_t in M_Σ. Obviously, this is a flooding algorithm applied on a graph (Lynch, 1996; Silvela and Portillo, 2001), where nodes correspond to symbols σ_i, and transitions are made explicit by transition encoders γ_j

During the expansion step, any expanded symbol needs to remember its parent symbols. For instance, if symbol σ_6 activated transition encoder γ_0, which in turn activated σ_7, then σ_7 needs to remember σ_6 as its parent symbol. Then, after successful activation of the target symbol σ_t, a viable sequence from σ_s to σ_t can be found by backtracking from σ_t via any of the parent symbols. This is the backtracking step of Dijkstra's algorithm, modified to select an arbitrary parent

from the set of feasible parents. Note that by following the symbolic sequence $\sigma_s \rightsquigarrow \sigma_t$, the spatial sequence can be inferred because transition encoders γ_i learned transitions in the symbolic space Σ and in the spatial input space Δ jointly.

The animal can use multiple scales in several ways. One possibility is to apply the expansion and backtracking algorithms only on a behaviorally necessary level. For instance, if the animal needs to find an exact sequence, then it should use the lowest level of M_Γ because it contains the highest resolution. If, however, the animal needs only an approximate direction towards a goal or when computational time is limited, it can operate on the highest available level. Intuitively speaking, the highest level represents approximate knowledge and, thus, can be used to find coarse estimates. Another possibility is to use all scales during sequence planning, which can be implemented in different ways.

One variant is *descending mode*, in which the sequence is first approximated on the highest available scale. After finding the target symbol σ_t on the highest scale, backtracking to the starting symbol σ_s yields sub-goals. These sub-goals are defined as all those symbols that are reachable via only one transition on this scale. Then, the sub-goals are used on the next smaller scale to refine the trajectory, illustrated in Figure 8. In both panels of the figure, the start symbol is σ_3 (blue color) and the target is σ_6 (green color). Panel A shows two scales in a one dimensional example and includes all details about the transition encoders $\gamma_i^{(n)}$, where n is the scale. That is, a transition encoder $\gamma_i^{(n)}$ (black box) learns transitions simultaneously in the symbolic space Σ as well as

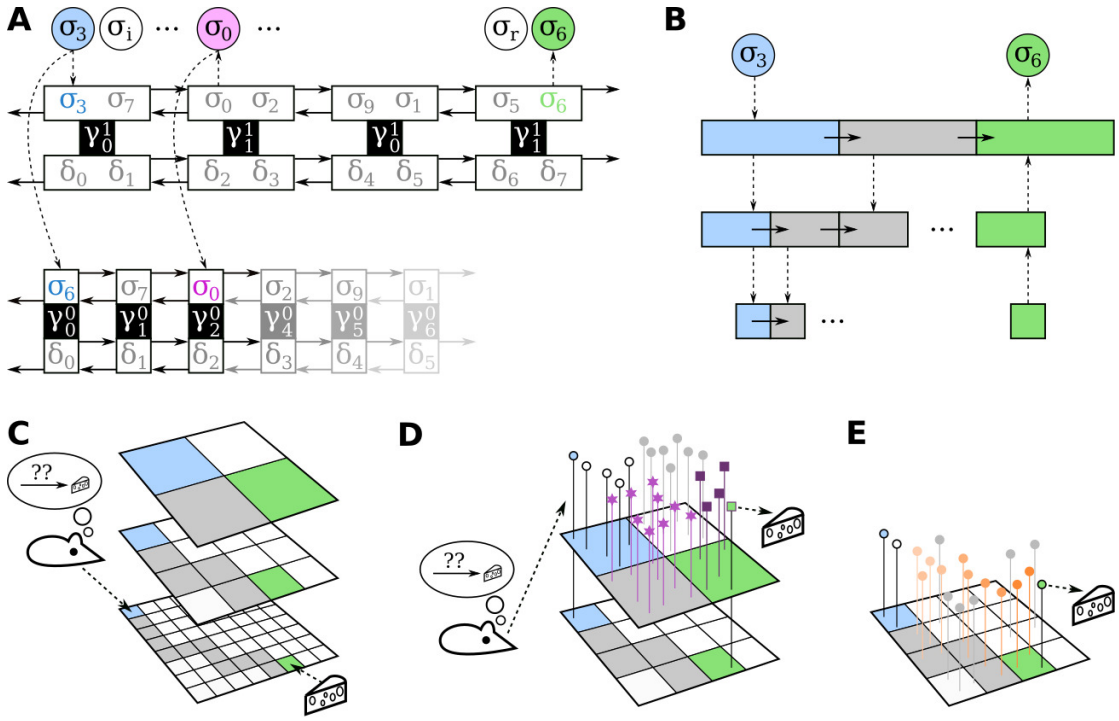

Figure 8: Retrieval of sequences with a scale-space in *descending mode*. (A) $\gamma_i^{(n)}$ learn transitions in two spaces simultaneously (cf. Figure 4). During retrieval, one space is sufficient (dashed lines) and expansion starts on the highest level from start (blue circle) to goal (green circle). After an approximate solution was found on a level, sub-goals are generated (magenta circle). Then, the next lower scale computes expansions to these sub-goals. (B) Simplified depiction of (A) that omits details about the input spaces of transitions. (C) An animal wants to compute a path from its current location to a target using a Transition Scale-Space. (D) It can first compute an estimate on the highest scale, prune unnecessary symbols (gray circles), and generate sub-goals (magenta stars). (E) The trajectory is subsequently refined by planning on a finer scale to these sub-goals. Note that, in the discrete case, diagonal movements violate constant cost assumptions of MTS (see Waniek, 2018). In case of multiple viable trajectories, the algorithm chooses one arbitrarily.

the spatial input space Γ, indicated by white boxes containing corresponding symbols. The expansion and backtracking steps on scale 1 yield intermediate symbols, shown demonstratively only for σ_0 (pink color). Then, the sequence is refined via scale 0 that computes only the trajectory from σ_3 to the sub-goal σ_0. Note that during retrieval, transition encoders $\gamma_i^{(n)}$ don't need access to spatial symbols δ_j. However, these spatial symbol can be inferred due to activation of the transition encoders. Panel B introduces a reduced graphical depiction of this process, in which details about transition encoders are omitted. The panel also illustrates that the integration areas of higher scales don't need to perfectly align at the borders with smaller scales. Figure 8C-E present an illustration of this process for a two-dimensional example in a grid-world. An animal that wants to compute a trajectory from its current location to a target (cheese) first retrieves a coarse estimate that activates multiple symbols in a coarse target region (panel D, purple small squares) and intermediate sub-goals (pink stars). Backtracking effectively reduces the search space for the refinement process. That is, symbols that are not on the coarse path (panel D, gray circles) don't need to be expanded on the next finer scale (panel E).

Another variant is *ascending mode*, in which the search for σ_t starts on the lowest scale. After a number of failed attempts to discover σ_t on this level, the search continues on a larger scale. This is repeated until the target is found, which may eventually, but not necessarily, involve the highest scale. In other words, this automatically adapts the scale to the real distance to the target. In contrast to *descending mode*, this variant requires the backtracking operation only

once the target is found.

In both modes, the animal has access to the first steps of a detailed sequence after only a brief computational time, and can refine this sequence while already moving. Note that there might be multiple feasible ways to expand a path on each scale in either mode. In this case, the algorithms used in Section 4 and presented in Appendix A randomly select one possible solution during backtracking. Finally, both modes have the same amortized runtime complexity. Specifically, assume a planning operation with a TSS to a remote location, and that this location is significantly further away than size of domains of the largest scale. Then, the runtime will be dominated by the number of retrievals on the highest level of the TSS in either retrieval mode, and the cost of expansions on lower scales are negligible. Pseudo-code and further details of the algorithms and sub-routines can be found in Appendix A.

Conclusively, the runtime issue that was motivated in Subsection 3.1 can be solved by an appropriate number of levels in a TSS.

4 Experiments

The scale-space data structure and algorithms were examined using simulations of a virtual agent on a plane surface. In particular, the $\mathcal{M}^{\mathsf{S}}$ that was simulated consists of a memory M_Σ that stores symbols σ_k, $k \in N_\Sigma$ and a memory M_Γ of spatial transitions. The temporal memory M_Π was omitted.

If not stated otherwise, symbols and domains and images of transitions were

computed in an offline manner. That is, transition centers were distributed hexagonally at the beginning of a simulation because of results presented in Waniek, 2018. Then, symbols were distributed randomly in the environment. Subsequently, images and domains of transitions were computed from the relative locations between symbols and transitions. Specifically, each transition pooled all symbols according to a nearest neighbor search, and lead to all symbols in its image simultaneously (see Figure 2 for reference). Thereby, transitions discretize the input space and perform Voronoi clustering.

The source code to reproduce all data for the figures and results presented in this section are available online under an Open Source License[1].

4.1 Worst case timings on a linear track

Segment trees or interval skip lists are widely used and their characteristics well known. The objective of this experiment was thus to simply show the acceleration capabilities of the data structure to find an element in the worst-case scenario.

A simulated agent was put on a linear track of length 10 m to compute the number of recursive retrievals that are required to find a specific element. Spatial symbols were distributed on the track quasi-randomly using the Hammersley point set, a low-discrepancy sequence. Then, a Transition Scale-Space (TSS) consisting of 7 scales with periods $0.2, \sqrt{2} \cdot 0.2, 0.4, \sqrt{2} \cdot 0.4, 0.8, \sqrt{2} \cdot 0.8$ and 1.6 m between transition encoders was computed offline by associating symbols to

[1] https://github.com/rochus/transitionscalespace

Grid Period	0.2	$\sqrt{2} \cdot 0.2$	0.4	$\sqrt{2} \cdot 0.4$	0.8	$\sqrt{2} \cdot 0.8$	1.6
Recursions	50	35	25	18	12	9	6

Table 2: Number of required recursive expansions until a target symbol σ_t is found on a linear track.

transition encoders using a nearest-neighbor search. The starting symbol closest to the start of the track was designated as σ_s, and the symbol closest to the end as σ_t. The numbers of necessary recursive retrievals from σ_s until σ_t was found are presented for each scale numerically in Table 2.

4.2 Wave propagation dynamics in two dimensions

In two dimensional settings, the expansion procedure is expected to resemble a propagating wave of activity when no additional restrictions on the expansion are imposed. To examine and illustrate this behavior, expansion was computed according to Algorithm 2 in a square environment of size 1×1 m. A total of 400 spatial symbols were distributed either randomly with a minimal distance of 0.02 m or according to the Hammersley point set. The objective of the latter was to observe any difference in behavior with more regularly placed symbols. Then, transitions were computed for a period of 0.2 m between transition encoders offline. The expansion was started at symbol σ_s, which was defined to be the closest symbol to position $(0.2, 0.2)$, and stopped at the target symbol σ_t, closest to position $(0.8, 0.8)$. After expanding and localizing σ_t, 300 Monte Carlo samples of symbolic sequences were computed using backtracking and randomly selecting

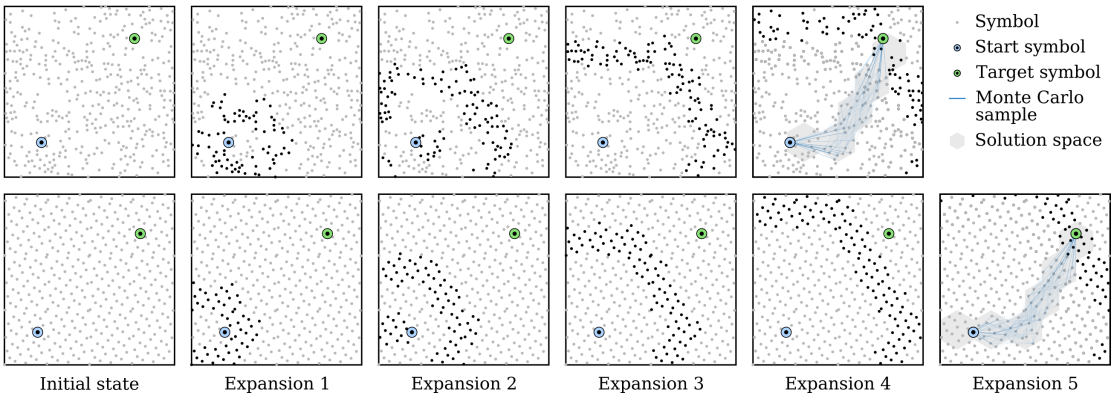

Figure 9: Propagating wave during expansion from start to target in a square environment. Top row shows results for randomly distributed symbols (gray dots), bottom row for symbol coordinates according to the Hammersley point set. Transition encoders were placed regularly on a hexagonal lattice (not shown) due to results presented in Waniek, 2018. Final panels include the solution space and 300 Monte Carlo samples (gray area). The solution space are all those images of transition encoders which contain symbols that are on a shortest path from goal to target. See Appendix C for results in a non-Euclidean space.

a parent during each backtracking step.

Figure 9 shows the results of the expansion and the emerging propagating wave from σ_s to σ_t. Black dots indicate symbols which are currently active during the expansion. The final expansion panel of each row shows a solution space as gray hexagons and Monte Carlo samples as blue lines. The solution space is given by the sequence of spatial domains of subsequently activated transitions on a solution sequence and forms a channel from start to target.

The only distinctive qualitative difference between randomly and more regularly placed symbols is the form of the propagating wave. When the

symbols are regularly placed, then the wavefront is hexadirectional during later expansion steps due to the perfectly regular placement of transition centers. For randomly placed symbols, the wavefront is more tattered, but still resembles the hexadirectional characteristic of the regular case. The difference in the solution space and the number of expansions is due to the selection of the start and target symbols and, thus, negligible.

Note that more than just one solution space may exist due to the underlying assumption of MTS. Specifically, MTS assume a constant-cost operation to move from the domain of a transition to its image. However, only one solution space is illustrated for each example in Figure 9. Also note the active symbols close to the start symbol σ_s during the second expansion. This behavior is because these symbols were not co-activated during start even though they are in close vicinity to σ_s, and because the implemented transition system does not include any knowledge about spatial directivity of the expansion.

4.3 Scale-space planning with sub-goal generation in descending mode

The descending mode of trajectory refinement was analyzed using a rectangular area of $2 \times 5\,\mathrm{m}^2$. In this area, 250 symbols were distributed randomly, keeping a minimal distance of 0.05 m between each pair of symbols A TSS consisting of 5 scales with periods 0.2, $\sqrt{2} \cdot 0.2, 0.4$, $\sqrt{2} \cdot 0.4$ and 0.8 m between transition encoders was then computed offline.

The symbol closest to coordinate $(0.4, 0.1)$ was selected as start symbol σ_s,

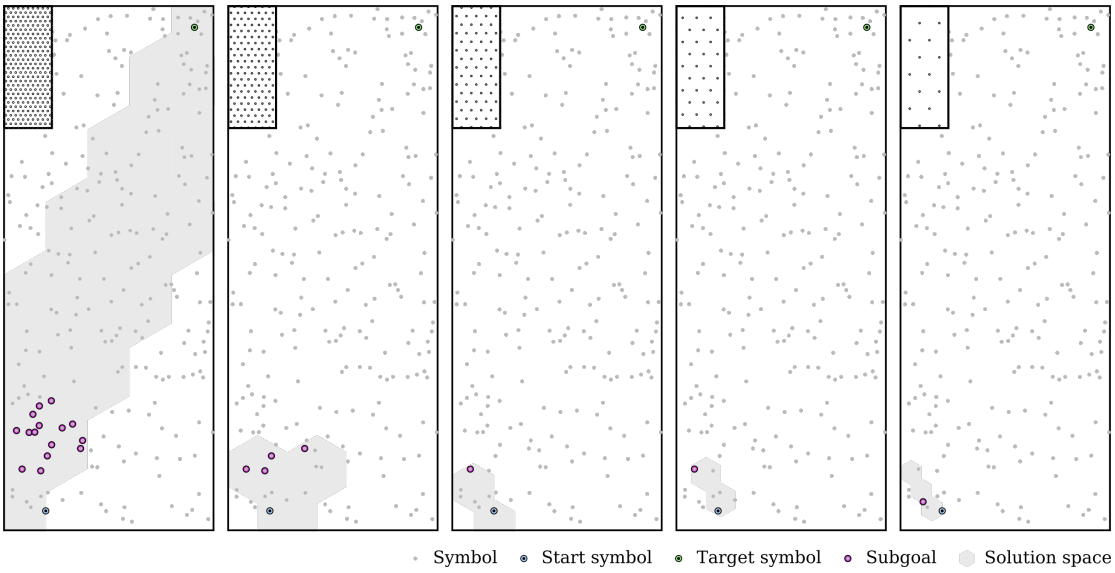

Figure 10: Searching a viable sequence from start (lower left) to target (upper right) using descending mode in a space with randomly sampled symbols (gray dots). The search from start (blue symbol) to target (green symbol) starts on the highest available scale (left panel) and generates sub-goals (magenta symbols). Gray area displays the solution space, which consists of all images of spatial symbols that were expanded and contain trajectories with equal distance. The solution is then refined using finer scales, which compute only partial solutions to the sub-goals from the next higher scale (remaining panels). Distributions of transition encoders are shown in the small inlays in each panel, and were precomputed based on results presented in Waniek, 2018.

and the symbol closest to coordinate $(1.8, 4.8)$ as target symbol σ_t. Qualitative results are illustrated in Figure 10. The search starts on the highest available scale, which computes an entire approximate solution from start to target. The solution space of this scale is indicated by gray hexagons in the leftmost panel of the figure. Also, this generates sub-goals, indicated by magenta circles.

After defining sub-goals, the procedure drops down to the next scale and computes trajectories only to the previously determined sub-goals. This limits the search space on smaller scales. The process repeats until the smallest scale is found, which is illustrated in the other panels of the figure.

4.4 Scale-space planning in ascending mode

The qualitative behavior of the ascending mode was examined likewise the descending variant. That is, 250 symbols were placed randomly in an environment of $2 \times 5\,\mathrm{m}^2$, with a minimal distance of $0.05\,\mathrm{m}$ between symbols. Then, a TSS consisting of 5 scales with periods 0.2, $\sqrt{2} \cdot 0.2$, 0.4, $\sqrt{2} \cdot 0.4$ and 0.8 m was computed offline.

The initial symbol was determined by finding the closest symbol to coordinate $0.4, 0.1$, and the target by the symbol closest to coordinate $1.8, 4.8$. During search, 3 expansions were computed on each scale before ascending the search to the next scale.

An example of the ascending mode is depicted in Figure 11. Each column shows the expansion on one scale. The first scale searches locally in the area that is indicated by the gray hexagons. Subsequent scales expand the respective

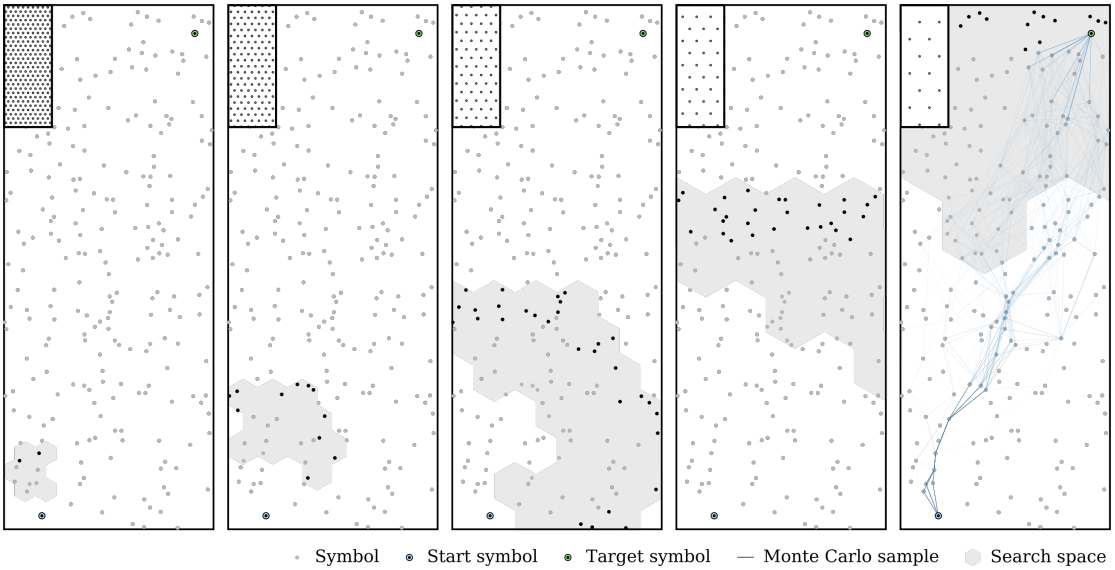

Figure 11: Searching a viable sequence from start (lower left) to target (upper right) using ascending mode. Each scale is expanded at most 3 times before ascending to the next scale (panels from left to right), until the highest scale is reached. Note that, in contrast to Figure 10, here the gray area depicts all images of spatial transitions that were expanded during search, and not the solution space. Black dots indicate the last symbols that were active on a scale before ascending to the next scale or when the target was encountered. The rightmost panel includes Monte Carlo samples (blue lines), generated using the backtracking algorithm. Distribution of transition encoders is shown in the small inlays in each panel, and was precomputed according to prior theoretical results presented in Waniek, 2018. Symbols (gray dots) were distributed randomly on the input space.

search radius, and propagate the search outward until the target is found on the highest scale. For each scale, black dots indicate the symbols that were active at the end of the search on this scale. After finding the target symbol, 100 Monte Carlo trajectory samples were drawn by backtracking from the target symbol to the start symbol, and selecting a random parent during each backtracking step. Note that this results in small trajectory segments and low variability close to the start of the trajectory, but larger segments and higher variability close to the target. In the figure, these samples are illustrated as blue lines on top of the search of the largest scale.

4.5 Shortcut discovery in the Morris water maze experiment

A simulated version of the Morris water maze experiment was used to examine the capability to find shortcuts in spatial information using a TSS. Qualitative results are presented in Figure 12 for three independent simulations. In each simulation, an agent started from the initial location (blue square) and traveled the environment with a radius of 2 m until the target platform (green circle) was found using movement statistics similar to real rodents. Symbols (or place cells; gray dots) were distributed along the recorded trajectory (black line) as follows, inspired by Growing Neural Gas (GNG)(Fritzke, 1995). As soon as the agent was further away than $m_{\text{dist}} = 0.05\,\text{m}$ from the closest symbol, a new symbol was recruited. To introduce stochastic symbol placement, this novel symbol was associated with a coordinate that was drawn from a unit circle centered at the current position of the agent and normally distributed with a standard deviation

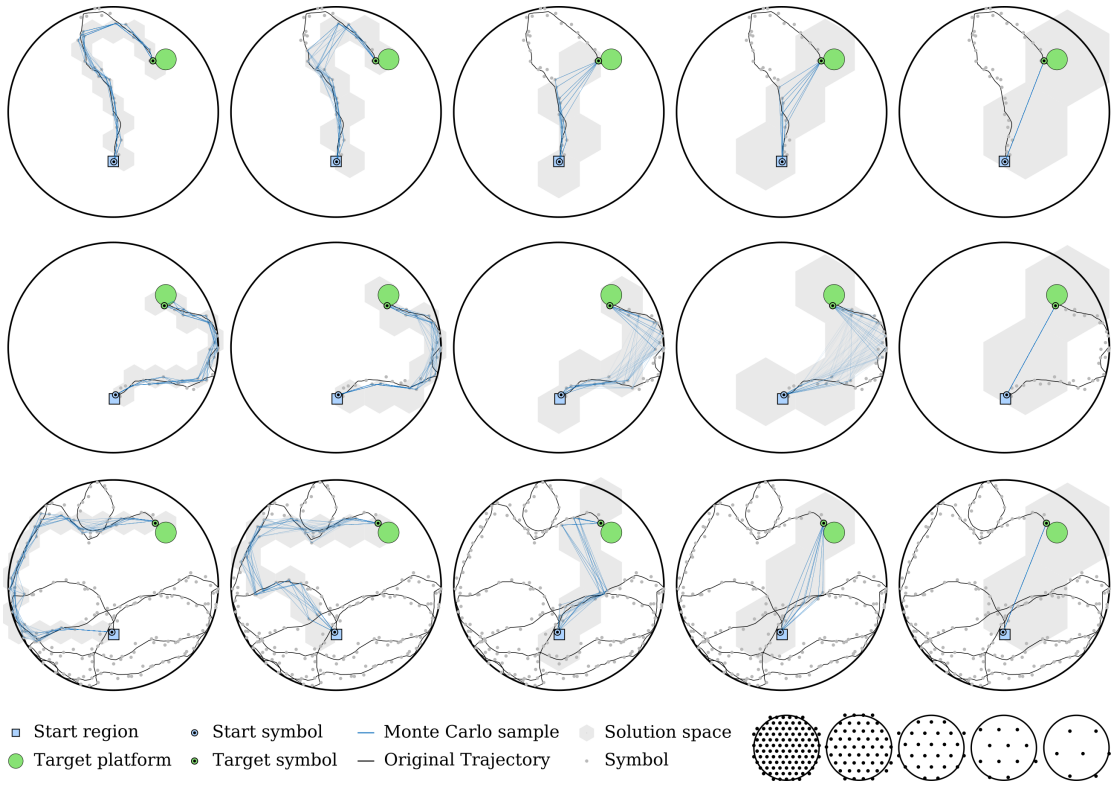

Figure 12: Results for three simulated Morris water maze experiment. Initially, the agent performed a random walk from Start (blue square), until it found the target platform (green circle). Meanwhile, novel symbols $\sigma_i \in \Sigma$ (gray small points) were spawned based on distance to previous symbols. The original random trajectory that the agent followed is shown in all plots as black line. Subsequently, possible solutions were computed for each scale individually (one scale per column). Each row represents one experiment, each column using one specific scale to compute the solution space (gray hexagons) and 100 candidate solutions (blue lines). The legend shows the distributions of transition encoders for each scale, which were precomputed according to prior results and assumed constant over the course of the simulation. See Subsection 4.5 for further details.

of $\frac{m_{\text{dist}}}{20.0}$, where the denominator was chosen arbitrarily. Subsequently, a TSS with 5 scales was computed offline for grid periods of 0.2, $\sqrt{2} \cdot 0.2$, 0.4, $\sqrt{2} \cdot 0.4$ and 0.8 m (distribution of centers shown in the legend of the figure). Specifically, each symbol was associated with the domain of the closest transition encoder, and to the image of the surrounding encoders.

For each scale individually, the solution space was computed according to Algorithm 2. That is, starting from the initial symbol, a sequence of viable transitions and symbols was computed until activation of the target symbol, which was defined as the symbol closest to the target platform. Figure 12 shows the solution space as gray hexagons. For each scale, 100 Monte Carlo samples (blue lines) were computed by backtracking from the target symbol to the initial symbol and selecting a parent symbol randomly during each backtracking step.

The figure shows that, on the smallest scale, reproduced sequences and the solution space remain close to a shortest path on the original trajectory. On higher scales, sample sequences and the solution space exhibit shortcut characteristic, essentially crossing parts of the environment that were not explored previously by the agent.

5 Discussion

Motivated by the runtime complexity of sequence retrievals in transition systems, this paper introduced the Transition Scale-Space (TSS). The TSS is an abstract data structure that contains knowledge about transitions between symbols on multiple

scales, shares several properties with the rodent Hippocampal formation, and in particular MEC. Most important, Subsection 3.3 and Subsection 3.4 showed that, given the assumption that they follow a Gaussian tuning curve, the optimal scale increment for both spatial and temporal receptive fields between consecutive scales of a TSS is $\sqrt{2}$. Consequently, it is hypothesized that the entorhinal-hippocampal loop forms a TSS, and that individual grid fields are spatio-temporal kernels that learn transitions both in spatial inputs, as well as a symbolic space. While the first is presumably given by spatially modulated input such as boundary-vector-cells, head direction cells, or speed cells, the latter is represented by place cells. Moreover, bottom-up learning of a TSS needs a mechanism to temporally buffer the symbolic input space. In total, this yields the TSS model for grid cells, depicted in Figure 13.

Although the results were presented in the context of spatial navigation due to the involvement of grid and place cells in navigational tasks, the results apply to arbitrary transition systems that operate on topological data. Now, these results will be discussed concerning their plausibility, predictions, and relation to other work.

This section is organized as follows. First, the data structure, algorithms, and results of the previous sections will be critically discussed with respect to biological plausibility, and testable predictions derived from TSS. Subsequently, temporal buffering and its potential relationship to Theta and TPP will be examined. This is followed by a detailed discussions about similarities and differences to other models for grid cells. Finally, TSS will be related to other

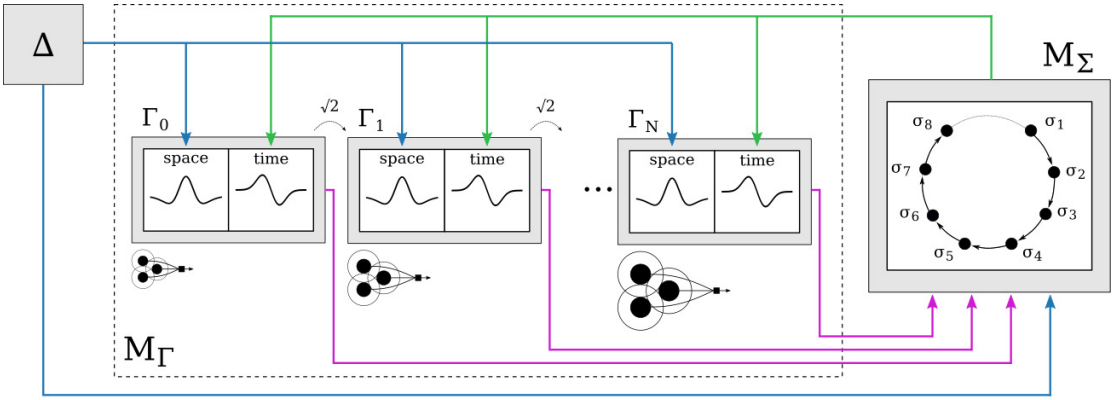

Figure 13: Transition scale-space model for grid cells with memories M_Γ (dashed box, middle), M_Σ (gray box, right), and spatial afferents Δ (gray box, left). It is hypothesized that each grid field of a grid cell is a space-time-kernel (gray small boxes within M_Γ) that represents spatial transitions. The spatial part of the receptive field learns transitions on input space Δ (blue arrows), which is presumably given by boundary-vector-, head direction, speed cells and other spatially modulated inputs. The temporal part integrates temporally ordered activity in a symbolic space Σ, itself hypothesized to be represented by place cell activity. For accelerated searches and retrievals of sequences, M_Γ consists of N different scales Γ_i (multiple scales depicted as horizontally arranged small gray boxes). For biologically plausible representations, the space-time kernels increase optimally by $\sqrt{2}$ from one scale to the next (dashed arrow between small gray boxes). Learning multiple scales requires temporally buffered access to symbols $\sigma_i \in \Sigma$ (ring buffer within M_Σ), which allows to relate the model to Theta phase precession (see Subsection 5.2).

areas of research, in particular to results from signal processing, mobile robotics, and computer science.

Note that, throughout the section, the terms symbol and neuron may be used interchangeably. Moreover, and if not stated otherwise, the term TSS will be used inclusively to mean both Multi-Transition System (MTS) and the extension to multiple scales presented in this paper.

5.1 Scale-space computations and biological plausibility

The properties and dynamics of TSS are compatible with observations from the Hippocampal formation. For instance, the activity of symbols in a TSS forms a traveling wave during expansion from start to target, depicted in Figure 9. Also, the Hippocampal formation shows traveling waves of activity (Lubenov and Siapas, 2009; Patel et al., 2012). However, TSS was not presented as a spiking neural network but in mathematical and algorithmic form. Thus, aspects of TSS will now be examined with respect to their biological plausibility. This will also be used to derive testable predictions from the mathematical model.

Figure 9 shows that, during expansion, TSS activates all intermediate symbols in form of a traveling wave until the target is reached. This is unlikely to occur in a biological neural network, as this would suggest a saturation of activity in the network. Rather, it is expected that in a biological network only a sparse activation of neurons in the domain of a transition is sufficient to trigger the transition and to keep the overall activity in the network at an minimal necessary energetic level. Also, it is expected that only a sparse subset of neurons in

the image of a transition will become active. Moreover, real neurons are noisy and, thus, might fail to spike. This poses a specific challenge for transition systems. Namely, if a transition misses to activate symbol neurons close to a goal, the retrieval fails. The problem can be solved easily by over-representing goal locations with a significantly larger number of symbolic neurons than usual. That is, if a single neuron's task is to indicate an important event, but has a certain small probability to fail spiking, then using two or more neurons will lower the chance to miss the event significantly. Hence, this predicts that the number of cells that are used for over-representation should be inversely proportional to the stochasticity of a typical neuron and the importance of the goal. Note that this idea is consistent with evidence that, in rodents, place cells accumulate near goal locations (Boccara et al., 2019; Dupret et al., 2010; Hollup et al., 2001; Lee et al., 2006; Mamad et al., 2017). Also, similar ideas, for instance *bagging* or *boosting* which both use multiple classifiers to improve inference tasks, are widely and successfully used in machine learning.

Table 2 shows the decrease in computational time to find a certain symbol. Moreover, Figure 10 and Figure 11 show that a TSS can be used to quickly compute approximate sequences. Importantly, the first steps of the retrieved sequences are *valid* according to the definition given in Waniek, 2018, which means that they are consecutive and connected and thus allow deterministic movement. Valid sequences might be of importance during tasks that require precise movement along previously acquired places. On the other hand, approximate solutions could also be used directly. This appears to be of utmost interest in other

behavioral tasks. For instance, consider the Morris water maze experiment, in which a rodent has to establish a route to reach the target platform. Waiting for an accurate computation to finish is not an option for a rodent, given the perilous situation in which it is during this experiment. Rather, the animal requires an algorithm that quickly yields an estimate of the solution so that it can start moving, and refine the trajectory online. The qualitative results shown in Figure 12 demonstrate that a TSS provides such an algorithm. Using a large scale quickly narrows the search space, or is even able to present a direction directly towards the goal. Note that a large scale comes at the cost of an increased variance of the solution, indicated by the large gray areas in the figures. Conclusively, it is expected that rodents operate not only on the highest available scale, but either in ascending mode to retrieve sequences that are initially valid, or on all scales in parallel.

The TSS and the simulation results were not presented in form of spiking neural networks. However, a neural network implementation of a TSS is conjectured to require only a comparatively simple neural micro-circuit. Recall Subsection 3.2 and Subsection 3.3 which explain that each spatial transition of a TSS represents transitions as intervals. Moreover, each level of the hierarchy is conceptually identical to other levels and differs only by the size of the interval. Then, learning that a symbol falls into the domain of a spatial transition on a higher scale follows exactly the same rules as lower scales and, thus, generalizes

well [2]. Moreover, representing associativity between transitions and symbols requires only simple binary logic elements, as discussed previously in Waniek, 2018. Thus, a neural network implementation of a TSS needs to only differ in the size of the spatial and temporal sampling process on each scale. Finally, the expansion algorithms that were used during the simulations can be parallelized for a distributed message passed system, and thus appear to be suited for spiking neural networks.

Both expansion methods, descending and ascending, as described in Subsection 3.5, lead to two different waves of activity. The first, already discussed above and presented in Figure 9, is a wave of activity from start to target. The second is a wave of activity that propagates along the hierarchy of TSS. Figure 10 shows that, during descending mode, the second wave starts on the highest (or coarsest) level of TSS and, after determining sub-goals, activity moves successively to lower scales. In contrast, the second wave starts on the lowest level and moves to higher scales, as depicted in Figure 11. In particular the latter is compatible with evidence from the Hippocampal formation. Not only increase grid fields systematically along the dorsal to ventral axis (Brun et al., 2008; Stensola et al., 2012), also activity propagates in the form of a traveling wave along the axis (Lubenov and Siapas, 2009; Patel et al., 2012). This is also compatible with the observation that increased temporal integration windows of transitions on higher scales necessarily lead to temporal lag, as will be discussed further below.

[2]This is well reflected in the implementation that was used to simulate the TSS. It uses a single method, parameterized only by scale, for data construction. Source code available online at https://github.com/rochus/transitionscalespace.

Both modes were used to quickly retrieve approximate solutions with detail only for the next steps. The modes correspond to usage of multi-resolution data structures that found widespread adoption in other areas of computer science (Behnke, 2004; Finkel and Bentley, 1974; Kambhampati and Davis, 1986; Lu et al., 2011; Orenstein, 1982). In particular the computer graphics community characterized the performance benefit that comes with multi-resolution representations and relies heavily on acceleration data structures of this form (MacDonald and Booth, 1990; Wald et al., 2009). Intuitively, large scales of such representations can be used to quickly prune data and select only a subset of the original data. Successive use of finer scales then refines the data to yield only singular elements of interest. This also applies to TSS in the sense that higher scales help to quickly retrieve feasible trajectories, and leads to the following prediction. Disruption of all larger scales of a TSS will lead to decreased performance during retrieval. Specifically, if there is just a single scale, then the TSS will suffer from the computational times that motivated the development of TSS, presented in Subsection 3.1. This could be shown in an experiment in which animals have to navigate to very remote targets. Targeted lesion or inhibition of grid cells with large grid fields should lead to decreased performance, measurable as an increase of the temporal delay between onset of pre-play and movement. Moreover, the number of failed navigation attempts should increase. The latter is because the activity of the traveling wave from start to target might diffuse due to noise of neural activity. In addition, increased navigation failure is also likely due to reward diffusion during sequence selection, an issue that was addressed

previously in the model by Erdem and Hasselmo, 2014.

Sequence selection was computed using classical backtracking in the results presented in Section 4. In this form, the presented algorithm is biologically implausible as it would require reverse information propagation. However, there are likely candidates for alternative implementations. First, recall that the data within a TSS represents a connected graph with constant edge weights, and searching a trajectory corresponds to finding a shortest path in a weighted graph. In turn, this edge weight can be interpreted as a temporal delay for spike propagation between vertices of the graph. Then, computing a winning trajectory could be achieved by reducing the temporal delay between vertices that activate the goal location. Iterating this is expected to successively compress expansion of a path *in time*. The outlined process describes the computation of an eligibility trace, known from RL, and is also related to reward diffusion that was used in Erdem and Hasselmo, 2012 and Erdem and Hasselmo, 2014. Also, this mechanism could explain why rats appear to pre-play multiple similar trajectories during homing tasks, observed by Pfeiffer and Foster, 2013. More specifically, repeated pre-play of similar sequences is expected to reduce the interspike interval between consecutive neurons that are on a sequence from goal to target. Analyzing temporal compression during pre-play will require significant amounts of data, and in particular simultaneous recordings of a large number of place and grid cells from a single animal.

5.2 Spatial pooling, temporal buffering, and Theta Phase Precession

Subsection 3.4 deduced that learning a TSS in an online fashion requires temporal buffering of symbols. Note that the necessity for temporal buffering in scale-spaces was previously noted already in the context of modeling visual receptive fields (Lindeberg, 2013). The difference is that here it is suggested that this temporal buffering can be achieved bottom-up using iterations of the lowest scale of a TSS that are nested within a slower input cycle. This makes the prediction that targeted lesion of the lowest grid scale should lead to disruption of higher grid scales. Another prediction is that – in the absence of other mechanisms that provide the data in a suitable fashion – there are no grid scales without temporal buffering.

Note that nested iterations of lower scales expand a path around a certain symbol, as depicted in Figure 7E. That means that starting nested iterations at a given symbol, for instance due to activation from Δ, will predict immediate next symbols. On the other hand, converging nested iterations from past symbols towards the given symbol repeats existing knowledge. To preserve linearity of spatio-temporal sequences, nested iterations should have directional tuning and ideally only expand (or retract) symbols that are in the heading direction of the animal. Linearity of activation is well supported by data from rodent experiments (Jeewajee et al., 2013; O'Keefe and Recce, 1993). More precisely, experiments showed that, when an animal is running, place and grid cells that are along the animal's heading direction perform Theta Phase Precession

(TPP), which means that their firing activity progresses relative to Theta. It was previously noted that the phase reference that is provided by TPP is required to encode temporal order (Lisman, 2005). In addition, Theta modulation was previously used to recall episodic sequences of states in models of regions CA1 and CA3 (Hasselmo, Bodelón, et al., 2002; Hasselmo and Eichenbaum, 2005). However, these models predate the discovery of grid cells and, thus, do not consider them. Note that conjunctive representations that have directional tuning, and which are supposed to be required for heading dependent linear temporal buffering, are not yet included in the TSS model. In addition, it is currently unclear how the inputs from the spatial input space Δ are optimally aligned with a temporal buffer.

TPP suggests that the alignment is ideally centered around the symbol that corresponds to the animal's current location during constant running speed. Note that during TPP, cells in the heading direction of an animal spike in order of their temporal succession (Jeewajee et al., 2013; Schmidt et al., 2009). Setting this observation in context of TSS, it is hypothesized that Theta and TPP in particular are the observable effects of temporal buffering to learn and retrieve data from multiple scales. Moreover, it is conjectured that, assuming constant running speed, the optimal alignment is such that afferents from Δ arrive precisely in the middle of the buffer, indicated in all panels of Figure 7 as blue circle. Then, there is $^1/_2 T_\Delta$ time for consolidation, and $^1/_2 T_\Delta$ time for prediction of immediate next place cells. Note that two distinct phases of retrieval and consolidation relative to Theta were previously described in the model by Hasselmo, Bodelón, et al., 2002,

and was also suggested by Dragoi and Buzsáki, 2006 and Lubenov and Siapas, 2009. In particular, Hasselmo, Bodelón, et al., 2002 found that memory retrieval and consolidation was best when entorhinal inputs to CA1 were in phase with Theta.

The rate of TPP of place cells depends on the width of their place fields (Dragoi and Buzsáki, 2006; Skaggs et al., 1996). That is, place cells with larger place fields spike later relative to Theta than cells with smaller place fields. A recent study modelled this effect mathematically and found that consistency with respect to distance between place fields is only preserved when TPP depends on the firing field width (Leibold and Monsalve-Mercado, 2017). The increase of the temporal integration window of spatial transitions, presented in Subsection 3.4, is complementary to these findings and model. Specifically, the size of the temporal integration window increases from one scale to the next. Essentially, this implies that the activation of a transition on a larger scale is slower than the activation on a smaller scale. This not only predicts that phase precession of grid cell will be later along the dorso-ventral axis due to the increase of grid field sizes along this axis, but also that precession will match the increase of $\sqrt{2}$ of the temporal integration window from one scale to the next.

In addition to a temporal (ring) buffer, the brain might rely on other mechanisms to support the formation of large scale transition knowledge, for instance Sharp Waves and Ripples (SPW-R) (see O'Keefe and Nadel, 1978 and review of rhythms related to Theta in Colgin, 2016). During SPW-R, actually traveled trajectories are replayed in a temporally compressed manner, which was found

to support the consolidation of spatial memories (Jadhav et al., 2012). It is exactly this form of temporal compression that is necessary to form large scale transition knowledge that would otherwise not fit into the temporal integration window of Spike-Timing Dependent Plasticity (STDP).

As of now, this interpretation of Theta and TPP leaves several questions with respect to TSS unanswered. For instance, it is unclear how the discontinuity between symbols in the temporal ring buffer is achieved. Possible candidates are distinct input streams arriving at opposing ends of Theta (Brankačk et al., 1993; Buzsáki et al., 1983). In addition, it was discovered that grid cells are differentially phase locked (Newman and Hasselmo, 2014). That is, there are distinct functional groups of grid cells which spike predominantly either at the trough or the peak of Theta. Conclusively, future work will address and incorporate these findings into the temporal buffer mechanism of TSS.

5.3 Relationship to other vector-based navigation models and successor representations

The Transition Scale-Space (TSS) is related to several other models of grid cells, and in particular to models of vector-based navigation in the rodent brain. In these models, multiple scales of grid cells are commonly used to encode a single spatial location, or a vector to a location.

For instance, the hierarchical goal directed navigation model presented in Erdem and Hasselmo, 2014 and also used in Erdem, Milford, et al., 2015 plans trajectories by first computing approximate locations using the highest

scale of its representation, and narrows down on the exact position using finer scales. Their model, itself an extension of their previous work (Erdem and Hasselmo, 2012), consists of head direction, persistent spiking, grid and place cells, as well as theoretical reward cells. Because, in their model, grid cells receive head direction modulated inputs from persistent spiking cells, the model falls into the category of oscillatory interference models for grid cell formation. Moreover, it uses grid cells for localization because the activity of any place cell is defined by integrating the activity of multiple grid cells. In turn, each place cell is bi-directionally connected to a reward cell. The functional role of these theoretical cells is to provide in- and output connectivity, and represent rewards. They are a simplification of prefrontal cortical columns that were used in their previous work Erdem and Hasselmo, 2012. Their model operates in two different phases. During exploration, the first phase, an animat randomly explores an environment and recruits place cells. The recruitment is either triggered when there is currently no active place cell, or by a Poisson process that is parameterized on the inter-recruitment-interval. During the second phase, the animat first activates the reward cell that is associated with its goal location. Then, it sends linear look-ahead probes with different heading directions that start at the animat's current location through the network of head direction, persistent spiking, grid, and place cells. When one such probe activates the representation of the goal location, the animat can navigate along the linear path towards the goal. Using just a single scale, this model has several issues. Most important, linear look-ahead probes are prone to miss the target especially when the distance

between current location and target is large. The reason is that, although the angle between probes remains constant, the absolute distance between probes increases with distance from the animal. Thus, chance for missing the goal location increases. Another is that, given a neural implementation, it is to be expected that noise accumulates over time and reduces the accuracy of the probes. Finally, the authors note that reward activity might diffuse over time. Their extension of this model to multi-scale representations addresses these issues as follows (Erdem and Hasselmo, 2014). Instead of having a single scale of place cells, the recruitment process creates place cells of several scales. The scale increment is motivated to be inversely proportional to the firing field radius of place cells. Then, during the second phase, linear look-ahead probes are more likely to activate a goal place cell due to the increased firing field radius of the cell. Upon activation of a goal reward cell, the reward cell then also activates all reward cells on levels with smaller firing field radii. Intuitively, the model thus first coarsely localizes the target, and then narrows down on the exact goal.

MTS, introduced in Waniek, 2018, and TSS, have several similarities to the model presented by Erdem and Hasselmo, 2014. For instance, TSS also operate in two phases: exploration during which symbols $\sigma_i \in M_\Sigma$ (place cells) are recruited and associated with spatial transitions $\sigma_j \in M_\Sigma$ (grid cells) and temporal transitions $\pi_k \in M_\Pi$, and retrieval, during which the system is iterated recursively and computing potential trajectories that are outbound from an agent's location. Likewise Erdem and Hasselmo, 2014, TSS recruit symbols either randomly or by distance to previous symbols. Note that in Section 4, the

exploration phase was mostly replaced by random sampling from the input space. The descending mode of trajectory computation, presented in Subsection 4.3, is particularly similar to the approach described in Erdem and Hasselmo, 2014. Both models first locate a target location in the coarsest representation. However, whereas the model by Erdem and Hasselmo, 2014 then narrows down at the target location, the TSS model generates sub-goals in the vicinity of the agent and re-plans to these sub-goals. This is due to the basic idea of TSS to expand a path from start to target into a coherent and connected sequence of intermediate symbols (Waniek, 2018). This is fundamentally different from computing a linear target vector and, thus, from Erdem and Hasselmo, 2014 or other navigation models that are based on global vectors. Specifically, and in contrast to Erdem and Hasselmo, 2014, the information that is stored within a TSS is not a sparse representation of locations, but a connected graph in which symbols (i.e. vertices of the graph) are connected explicitly via spatial and temporal transitions (edges of the graph). In addition, TSS only require iteration through grid and place cells during retrieval. Thereby, TSS are independent of the heading direction during exploration. This is exemplified in Appendix C, which demonstrates that TSS compute the geodesic between points and not a linear vector if the input space is a non-Euclidean manifold. Note that path expansion via traversal of a connected graph also means that an TSS can compute trajectories that would require the agent to first walk into an opposing direction. For example, assume an environment with a large U-shaped barrier and in which the goal is to navigate from the center of the environment to a location behind the

barrier. Then, linear look-ahead probes fail to accomplish the task, whereas TSS successfully compute trajectories. One drawback of this is that the necessary connectivity of the represented graph means that TSS cannot compute trajectories to locations that were not previously explored. However, this is in line with evidence from rodent experiments in which rats were unable to take novel shortcuts (Grieves and Dudchenko, 2013). There are several other significant differences, as for instance that grid cells are not assumed to form on the basis of oscillatory interference but because of dense packing of receptive fields that represent transitional information (Waniek, 2018). In addition, place cells were hypothesized to be recruited from spatial afferents independently from grid cells. Recent evidence from rodent experiments supports this idea (Chen et al., 2019). Note also that, although this paper uses multi-scale grid cells, it does not recruit multi-scale place cells. Another difference is the form of activity during retrieval. Whereas linear look ahead probes lead to targeted and directional activity, the search strategy of TSS resembles a propagating wave of activity from the goal to the target (see Figure 9). The latter is consistent with evidence that shows that activity in the Hippocampal formation propagates in waves (Lubenov and Siapas, 2009; Patel et al., 2012). Unlike the model by Erdem and Hasselmo, 2014, multiple scales of representation were motivated in Subsection 3.1 not by linear probes missing the goal location or by reward diffusion, but by examining the abstract runtime complexity of sequence retrievals in transition systems and, thus, wave propagation. Finally, Erdem and Hasselmo, 2014 did not provide a mathematical justification for the scale increment in their model. In contrast, the

scale increment between consecutive levels of the TSS was derived analytically for biologically plausible receptive fields and with the goal to optimally construct a data structure that stores look-ahead transitions in Subsection 3.3.

Another linear look-ahead model was presented by Kubie and Fenton, 2012. First, the authors highlight the difference between navigation with a place cell map, in which a path can be computed as a trajectory that minimizes the place fields that are crossed between goal and target, and vector navigation using grid cells. The authors note that the regularity of the grid pattern should contain sufficient information to compute linear vectors from start to a goal location even across previously uncharted territory. Based on evidence from rodent experiments, the authors then introduce the concept of a rigid module. Such a module contains only grid cells with the same scale and orientation. Moreover, the authors define a tile: a region of space with only one grid bump or, in other words, the induced Voronoi cluster of each bump of a grid cell. In total, tiles of a rigid module densely cover the input space. Then, the authors proceed to define a shortest inter-bump vector, which is the shortest vector or shortest relative shift between any two bumps that could appear within one tile. The authors continue by asking if it is possible to learn predictions about subsequent bump locations within a tile during navigation. In simulations of a virtual agent, they find that grid cells in a single rigid module cannot learn predictive codes for subsequent grid bumps using Hebbian plasticity. However, they then show that rigid modules of conjunctive cells, i.e. grid cells that are modulated by head direction, in fact express strong connectivity between cells

with different bump locations within one tile and thus represent the inter-bump vector. Based on these results, the authors finally speculate that learning such predictions could be used for linear vector based navigation. The model by Kubie and Fenton, 2012 is similar to TSS only to a limited extent. For instance, both models assume that grid cells compute or represent a local relationship structure. However, Kubie and Fenton, 2012 take the grid pattern as given, whereas the hexagonal arrangement of grid fields was derived mathematically as the result of optimality considerations in Waniek, 2018. Moreover, TSS do not use grid cells for localization. Although an inherent part of detecting the start of a transition is to determine a specific input space configuration, this information is not used for downstream localization in the form of place cells, but to learn a neighborhood relationship and, thus, the adjacency between place cells and not between grid cells. Most important, however, is that the model by Kubie and Fenton, 2012 learns to predict succeeding grid representations, whereas TSS learn transitions to (or predictions of) subsequent place cells. Finally, and similar to the model by Erdem and Hasselmo, 2014, the authors note that their model has limited capabilities in complex navigation tasks. To stress this point, linear vector based navigation models cannot compute non-linear solutions such as required in the U-barrier environment that was mentioned above, or in environments or input spaces where the geodesic is the desired solution.

Prior to the discovery of grid cells, a vector based navigation model of the Hippocampus was presented by Redish, Touretzky, et al., 1996. In their model, vectors are computed due to visual landmarks and the relative positions of a start

and target with respect to these. Subsequent work of the authors then introduced the idea that the Hippocampus works in different two different modes (Redish and Touretzky, 1998). During exploration, the animal acquaints itself with the environment and learns spatial representations in place cells of CA3. Then, when the animal re-enters an environment, the authors suggest that it learns routes along the previously acquired representations. That is, when an animal moves from a location a to a location b, their model increases the strength of the recurrent collaterals between these two representations given the direction $a \rightarrow b$, but not the strength of $b \rightarrow a$. Although these asymmetric connections are conceptually identical to transitions, TSS explicitly represent temporal transitions in a memory M_{Π}, as briefly reviewed in Subsection 1.2. The benefit of separation is that a currently active representation in place cells is not prone to be immediately overwritten or changed due to asymmetric recurrent connectivity. That is, place cells can be stored in an auto-associative memory M_{Σ} in which place cell activity is a local attractor and, thus, performs pattern completion, and in contrast, transitions can be stored in a hetero-associative memory to toggle switching of local attractors in the auto-associative memory (Palm, 1980; Wennekers and Palm, 2009; Willshaw et al., 1969). Pattern completion appears to be an import concept for place recognition, and it is thus unsurprising that, based on evidence for local connectivity, associative memories were previously used to model the Hippocampal formation (Graham et al., 2010; Marr, 1971; McNaughton and Morris, 1987; Treves and Rolls, 1994). However, TSS predicted a specific strong recurrent connectivity between memories M_{Σ} and M_{Π}, which was recently

supported by evidence from rodent experiments (Davoudi and Foster, 2019). Another difference between the work presented in this paper and the work by Redish and Touretzky, 1998 is that TSS store spatial transitions within grid cells of memory M_Γ, and this paper extends the spatial transition memory to multiple scales. Multiple scales of representation were not considered in Redish and Touretzky, 1998. The authors note, however, that SPW-R should consolidate knowledge about sequences of places. They then suggest that phase precession is a result of these asymmetric connectivities during retrieval of such sequences. Also the work presented in this paper presents phase precession as a result of asymmetric information which re- or pre-plays previously acquired sequences. In contrast to Redish and Touretzky, 1998, phase precession is speculated to be a necessary mechanism that allows the formation of multi-scale grid cell representations. Additionally, SPW-Rs are believed to allow learning very large scales that would otherwise not fit into the temporal integration window of STDP learning rules. That is, because SPW-Rs contain temporally compressed sequences, a neuron is hypothesized to be able to associate with a larger number of temporally ordered elements than would normally fit into the temporal STDP integration window during uncompressed sequences. Due to the numerous similarities between and complementary elements of the TSS presented in this paper, and the model presented in Redish and Touretzky, 1998, it appears highly likely that both models can be combined.

Cueva and X.-X. Wei, 2018 and, independently, Banino et al., 2018 observed that training deep neural networks on spatial navigation tasks leads to the

formation of place and grid representations. The latter in particular argue that their system learns vector-based navigation and spatial codes. However, a closer examination of their model architecture reveals that the cells which exhibit grid codes learn predictive codes for downstream spatial representations. More precisely, the recurrent network in which the grid representations emerged receives spatially modulated input and forwards information to a planning module. Hence, grid representations appear to be, in fact, an optimality result when learning from navigational tasks in recurrent networks. Moreover, their model learns to navigate to locations at which the agent was presented with a reward. Thus, the computation that is performed by their model closely resembles prior work on Successor Representation (Momennejad et al., 2016; Stachenfeld et al., 2016).

The TSS model of grid cells is also related to other transition models of grid cells, and in particular models that assume grid cells perform Principal Component Analysis (PCA) or compute a Successor Representation (SR) (Dordek et al., 2016; Momennejad et al., 2016; Stachenfeld et al., 2016). These models compute grid like responses based on place cell inputs and represent correlate information between grid cell and place cell activity in multiple scales. For instance, higher scales of SR models represent transitions to remote locations and learn this information exclusively on place cell afferents. In contrast, grid cells of the scale-space TSS model learn their representation on a suitable input space and link to place cells only due to co-activation learning. Thereby, they relay structural information from the input space to higher cortical representations,

which can then be used independently of the input space during retrieval (see Waniek, 2018 for a discussion thereof). In addition, SR models propose that grid cells learn to represent the weighted sum of rewards of distant future states. That is, these models require a certain policy by which the animal navigates the environment and also that an environment and its intermediate states are sampled sufficiently often for the representation to converge. Moreover, the animal needs to learn a distinct SR representations for each specific goal. Given recent evidence that place cells are insensitive to reward information (Duvelle et al., 2019), it remains to determine if grid cells support reward storage, or in which way reward information could be available to support learning of an SR. Other recent experiments showed that the hexagonal grid representation is, in fact, locally distorted upon changes to the reward structure of an environment Boccara et al., 2019; Butler et al., 2019. In these experiments, grid fields moved closer to a certain location if this location corresponded to a singly peaked reward. On a first glance, these findings appear to be in favor of SR models. Yet, it is unclear in which way exactly this fits into these models, and, more importantly, what exactly drives the grid distortion. In contrast to PCA and SR models, TSS are independent of any policy or reward and can be recomputed easily for arbitrary targets. Thus, TSS can be used more flexibly in different environments and in particular when the observed reward modulation is not intrinsic to grid cells, but, for instance, attached to the spatial input space. Also, TSS require only limited information about traversed states and do not need dense or repeated coverage of the input, as demonstrated in the Morris water maze experiment presented

in Section 4. Nevertheless, SR models have the benefit that they can compute trajectories to target locations without any additional reward mechanism, which is currently lacking from TSS. Possible future work therefore includes to merge these models.

5.4 Relationship to scale-spaces, data structures, and graph algorithms

The constructive procedure presented in Section 3 generates a data structure that contains transitions in multiple resolutions. Specifically, it forms a scale-space in the form of a Gaussian or a Laplacian-of-Gaussian pyramid. This type of data structure is well known in the computer vision and signal processing communities (Lindeberg, 1994; Lindeberg, 2010; Witkin, 1983). Among other application areas, Gaussian pyramids were used to describe biologically plausible retinal and visuocortical receptive fields (Behnke and Rojas, 1998; Georgeson et al., 2007; Young, 1987). In addition, it was shown previously that a scale increment of $\sqrt{2}$ is optimal for normally distributed inputs in vision tasks (Lindeberg, 2010). In image processing, application of a Gaussian pyramid corresponds to consecutively smoothing an image, i.e. an input gets low-pass filtered by which fine-scale information is removed. Consequently, this allows to detect features in a scale-invariant way, which is commonly used in classical computer vision algorithms (Lindeberg, 2015; Lowe, 2004).

The same intuitive idea is certainly true for spatial navigation. To plan sequences to remote locations, it appears behaviorally unnecessary to factor fine

details close to the target into the solution. Rather, it seems more important to quickly establish an approximate solution, which can be refined during execution. In other cases, fine details may not be relevant at all, and approximate solutions entirely sufficient. Gaussian pyramids provide a principled solution to compute these approximations, as demonstrated in the examples presented in this paper. More specifically, Figure 10 and Figure 11 show how a TSS can compute trajectories with details near the agent and approximate steps at the target, and Figure 12 show the level of approximation that each scale provides.

The computational benefits of spatial approximations with and clustering into multiple scales of representation is also well known in other areas. For instance, the computer graphics community has widely adopted data structures that are based on or extensions of quad-trees (Finkel and Bentley, 1974; Samet, 1984), or higher dimensional variants such as octrees (Orenstein, 1982), to achieve real-time graphics. Also the robotics and path planning communities have adopted multi-scale data structures (Behnke, 2004; Hwang et al., 2003; Kambhampati and Davis, 1986), which are often combined with (extensions to) classical algorithms such as Dijkstra's algorithm (W. Dijkstra, 1959).

The method presented in this paper is related to these classical graph algorithms, and in particular A* (Hart et al., 1968). A* generates optimal solutions during graph searches, but its runtime crucially depends on a suitable heuristic. The basic idea of A* is the same as in Dijkstra's algorithm, meaning that the algorithm sequentially expands nodes and tests if they are on a path to the target. In contrast to Dijkstra's algorithm, A* uses a heuristic that guides the

expansion of nodes and thereby limits the search space significantly. Still, A* requires that the heuristic is admissible, which means that the heuristic always has to underestimate the real distance to the target. The scale-space approach presented in this paper adheres to this condition, and is thus a suitable heuristic for A*.

6 Conclusion and outlook

In this paper, transition encoding was examined from the perspective of behavioral requirements for path planning. An example showed that a single scale of transition encoders is insufficient to provide run-times that lie in a meaningful range when neuronal axonal delays are respected. However, transition encoding allows interpretation in form of linked lists of data, for which asymptotically optimal search strategies are well known from computer science. In particular, interval skip lists introduce hierarchies to accelerate searches exponentially.

It was then derived that the optimal scale increment for a data structure with biologically plausible spatial receptive fields is $\sqrt{2}$. Following this, it was argued that also the temporal integration window needs to increase accordingly, and that temporal buffering of data is mandatory.j The resulting hierarchical representation, called TSS, can accelerate path planning by transition look-ahead forms a scale-space that provides approximate path planning solutions. This was shown in simulated navigation experiments using two different modes for sequence retrieval. In addition, a simulated Morris water maze experiment

highlighted computation of approximate solutions on higher scales.

Given that scale-spaces are well known from other cortical regions, primarily the visual cortex, this allowed to discuss the results in terms of a general principle that underlies cortical processing. Furthermore, the TSS was related to techniques for multi-resolution path planning that are well-established in the robotics community. Conclusively, the discretized EC could be explained as an effective, yet general purpose, data structure that achieves optimal run-time behavior during goal-directed sequence planning.

Future work will include a biologically plausible model for the self-organization of the dendritic tree of grid cells and, thus, the formation of hexagonal grid fields in a dynamical model. Moreover, the TSS will be extended with reward propagation mechanisms and a probabilistic model of computation.

Acknowledgments

The author would like to thank Christoph Richter and Jörg Conradt for discussions and feedback on the original idea of multi-transition encoding. Also, thanks to Christoph Schied for invaluable discussions on acceleration data structures and algorithms. Finally, thanks to Maike Krause for significant contributions to the structure of the manuscript.

The work presented in this paper was conceived primarily while the author was with the Neuroscientific System Theory Group (NST), Technische Universität München, Arcisstraße 21, 80333 München, Germany, and was partially funded by EU FET project GRIDMAP 600725. The author declares no competing financial

interest.

A Algorithms

The following pseudo-code uses member-notation similar to common programming languages in the form σ.parents, where `parents` is a member of symbol σ. Also, multiple assignment instructions may be stated in a single line (e.g., Algorithm 4, line 2). An implementation in python of the algorithms and additional code for visualization can be found at https://github.com/rochus/transitionscalespace.

Algorithm 1 computes the expansion of a set of symbols. That is, given a set of transitions Γ and currently active symbols S_t at time t it computes all subsequent symbols of Σ that can be reached and have not been expanded previously. The algorithm also updates the parents of all such newly activated symbols. A transition γ has members `image` and `domain`, which are sets containing the symbols in the image and domain of the transition.

Given a set of transitions Γ, a set of symbols Σ, a set of starting symbols Σ_{start}, and a set of target symbols Σ_{target}, Algorithm 2 computes a sequence from any of the start symbols to any of the target symbols on one scale of transitions. The algorithm does so by calling Algorithm 1 repeatedly until the target symbol is in the set of currently active symbols.

Backtracking a solution sequence from target to start is computed as specified in Algorithm 3. That is, given a set of symbols Σ, a set of starting symbols Σ_{start}

Algorithm 1 Expanding a graph to all subsequent symbols

```
procedure EXPAND(Γ, S_t, t)
    E ← ∅
    for all γ ∈ Γ do
        D_active ← S_t ∩ γ.domain
        if D_active ≠ ∅ then
            E ← E ∪ γ.image
            for all σ ∈ γ.image do
                if σ.parents = ∅ or σ.t_parents = t then
                    σ.parents ← σ.parents ∪ D_active
                    σ.t_parents ← t
                end if
            end for
        end if
    end for
    return E
end procedure
```

Algorithm 2 Find a sequence on one scale of transitions with flood fill

```
procedure FindSequenceOnScale(Γ, Σ, Σ_start, Σ_target)
    t ← 0
    S_t, T ← Σ_start, Σ_target
    while S_t ∩ T = ∅ do
        for all σ ∈ S_active do
            σ.t_retrieval ← t
        end for
        N ← Expand(M.Γ[s], S_t, t)
        S_{t+1} ← {σ | σ ∈ N, σ ∉ S_t, σ.t_retrieval < 0}
        t ← t + 1
    end while
    return S_t ∩ T
end procedure
```

and a set of target symbols Σ_{target}, a path is computed by repeatedly randomly selecting a parent node until one starting symbol is found. Note that this requires that Algorithm 1 was applied to the symbols, whereby the parent field of each symbol is set accordingly.

Algorithm 3 Backtrack from target to start

```
procedure BACKTRACK(Σ, Σ_start, Σ_target)
    S ← Σ_target
    P ← {σ_p | σ_p ∈ σ_s.parents, σ_s ∈ S}
    while Σ_start ∩ P = ∅ do
        S, P ← P, {σ_p | σ_p ∈ σ_s.parents, σ_s ∈ S}
    end while
    return S, P
end procedure
```

Algorithm 4 takes a scale-space MTS $\mathcal{M}$, a set of symbols Σ, a set of start symbols Σ_{start}, a set of target symbols Σ_{target}, and a number of maximal expansions per scale. It then computes at most $i_{\max}$ expansion according to Algorithm 1 per scale. If this maximum number is reached, the algorithm escalates to the next scale. $\mathcal{M}.\Gamma[s]$ denotes transition scale s of $\mathcal{M}$. Afterwards, Algorithm 3 can be used to compute a Monte Carlo solution sample.

Algorithm 5 takes the same arguments as Algorithm 4 except the maximal number of scales. The algorithm starts on the highest available transition scale and finds a sequence with it. After finding a solution, it backtracks to the starting symbols and generates sub-goals. Subsequently, it drops down one scale to

Algorithm 4 Find a sequence of symbols by ascending the scale-space

```
procedure FindSequenceAscending(M, Σ, Σ_start, Σ_target, i_max)
    i, t, s ← 0, 0, 0
    S_t, T ← σ_start, σ_target
    while S_t ∩ T = ∅ do
        for all σ ∈ S_active do
            σ.t_retrieval ← t
        end for
        N ← Expand(M.Γ[s], S_t, t)
        S_{t+1} ← {σ | σ ∈ N, σ ∉ S_t, σ.t_retrieval < 0}
        if i ≥ i_max then
            s, i ← min{s + 1, N_scales}, 0
        end if
        t, i ← t + 1, i + 1
    end while
end procedure
```

refine the trajectory to the sub-goals. In its current form, this requires to reset the parent information in each symbol (line 6).

Algorithm 5 Find a sequence of symbols by descending the scale-space

```
1:  procedure FindSequenceDescending(M, Σ, Σ_start, Σ_target)
2:      s ← N_scales − 1
3:      T ← Σ_target
4:      while s ≥ 0 do
5:          for all σ ∈ Σ do
6:              σ.t_retrieval, σ.t_parents, σ.parents ← −1, −1, ∅
7:          end for
8:          H ← FindSequenceOnScale(M.Γ[s], Σ, Σ_start, T)
9:          T, P ← Backtrack(Σ, Σ_start, H)
10:         s ← s − 1
11:     end while
12: end procedure
```

B Relationship to category theory

Multi-Transition System (MTS) as proposed in Waniek, 2018 and extended to a scale-space representation in this paper have a relationship to category theory (see Adamek et al., 2009 for an introductory text). Changing the terms that were used throughout this paper makes this connection more pronounced. Let any transition between two symbols of the same alphabet be a *morphism* that operates

on some objects. For instance, spatial transitions are morphisms $g : x \to y$ that operate on objects x, y of the category I of spatial symbols. Learning spatial transitions, part of the proposed dendritic computation of grid cells, is thus learning of morphisms in the category I. In MTS however, grid cells simultaneously also learn transitions in the representation that is created by place cells, i.e., morphisms $f : p \to q$ that operate on the category of place symbols P. Observe that, according to scale-space MTS – but also in the context of Successor Representations – grid cells learn to preserve the structure of the input space and convey this knowledge to place cell representations. Consequently, grid cells can be interpreted as *functors* F that transport objects $x \in I$ to objects $Fx \in P$ and morphisms from the category of spatial symbols I to morphisms in the category of place cells P. Namely, they transport transitions between symbols from one category to the other, that is $F\,g : F\,x \to F\,y$, while preserving structure. Also, for every functor F, there exists an inverse functor F^{-1}, i.e. $F^{-1}\,f : F^{-1}\,p \to F^{-1}\,q$. Note that these observations apply across different scales of transition representations, regardless of the size of the grid field, and thus expose a fundamental cortical operation. Figure 14 illustrates the relationship between MTS and category theory. During spatial navigation, the existence of the inverse functor allows to perform inference on the sensory inputs $\delta_i \in \Delta$ given a sequence of symbols $\sigma_k \in \Sigma$. This is demonstrated by computing the solution space in the examples of Section 4. Certainly, also other cortical areas have to learn relationships between objects of different categories while preserving structure, for instance to link the visual representation of an "apple" with the

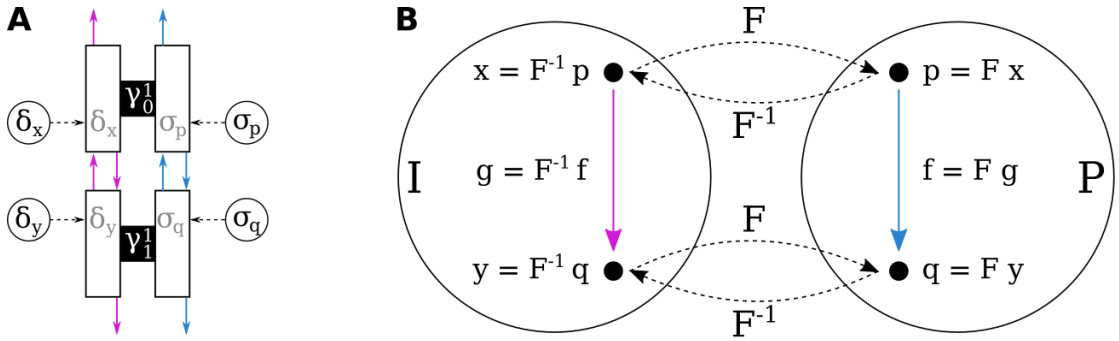

Figure 14: Transitions, grid cells, and category theory. (A) Visualization of grid cells according to scale-space MTS that learn transitions from symbols δ_x to δ_y, while also learning transitions from σ_p to σ_q. (B) Commutative diagram of category theory, in which a morphism g maps objects x to y in a category I, a morphism f maps objects p to q in a category P. The morphisms and objects are linked by a functor F that preserves structure of the categories.

word "apple". Conclusively, it appears reasonable to investigate if cortical computations can be understood and expressed compactly in terms of category theory.

C Relationship to differential geometry

Multi-Transition Systems (MTS) hypothesize that grid cells learn spatial transitions. Specifically, grid cells are believed to densely pack on-center and off-surround receptive fields, their grid fields, on a suitable input space. For analytical treatment, the input space was assumed to be the Euclidean in Waniek, 2018.

Suppose now that the input space is not Euclidean, but a Riemannian

manifold. Still, assume that grid cells densely pack on-center and off-surround receptive fields on this manifold. Due to their limited extent, receptive fields and, consequently the cells to which they belong, interact only locally. Then, it appears that receptive fields capture a locally Euclidean region of the manifold, and that the crossover from one receptive field to the next is smooth. In other words, the on-center component of a single receptive field forms a chart (or coordinate system) on the manifold, and the entirety of on-centers of all receptive fields form an atlas of the manifold.

Having an atlas of a manifold allows to compute geodesics between arbitrary points. That is, it is possible to compute the shortest path between two points on a curved surface. Intuitively, this can be related to grid cells as follows. Densely packing grid fields onto the manifold leads to a surface triangulation of the manifold. Computing shortest paths on such a mesh is a well studied problem. It can be solved approximately for instance with Dijkstra's algorithm, the Fast Marching Method (Sethian, 1996), or the Heat Method (Crane et al., 2013). The latter two techniques have in common that they directly solve a wave propagation problem, and also Dijkstra's algorithm propagates a wave from source to target (see experiments in Section 4).

The following simulation of a virtual agent and an MTS was used to explicitly demonstrate this relationship.

The agent moved at constant speed, and its heading direction was drawn from a Laplacian distribution at each time-step of the simulation. The input space, i.e. the space on which the animal moved, was the outside of an upper

hemisphere. Note that, due to symmetry, the presented results are identical to an agent walking on the inside of a lower hemisphere.

First, receptive fields and their interactions were simulated as a particle system with push-pull dynamics similar to the method described in Waniek, 2018. That is, only particles close to the agent were updated. Specifically, the closest particle to the agent was pulled closer to the agent. All other particles in the surrounding of the agent exerted relative forces on each other. The update rate with which these forces acted slowly decayed over time. The agent and particle system was simulated for an arbitrary time until the particles converged to an approximately stable solution (see 3D view in Figure 15). The figure shows that the placement of receptive fields forms a blue noise distribution on the input manifold, or a low-discrepancy sequence. Holes in the arrangement are probably due to non-uniform sampling of the input space, and because of the simplicity of the particle system. However, visual inspection (without computing gridness scores) shows that fields form a dense packing. The particle locations defined the centers of spatial transitions.

Then, 5000 points on the hemisphere were randomly sampled to be used as symbols. Subsequently, each symbol was associated with the closest spatial transition, which in turn was associated to point to all surrounding symbols. This forms the MTS.

Finally, the shortest path between two arbitrarily chosen points was computed by expansion of the MTS and backtracking 50 Monte Carlo samples (see Section 4 or Appendix A for details). The results are presented in Figure 15. Although the

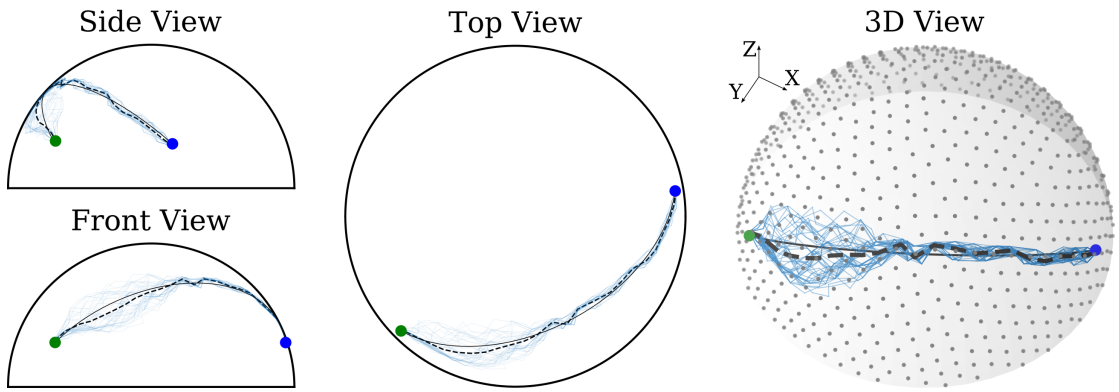

Figure 15: MTS on hemisphere. A virtual agent equipped with a push-pull particle system was simulated for an arbitrary time until the particle system converged. The particle locations were then used as transition centers (gray hexagonally arranged dots in 3D view). Subsequently 5000 symbols were randomly placed on the hemisphere (data not shown), and transitions and symbols associated. This formed the data contained in the MTS Then, the MTS was queried to expand a path from start (green dot) to target (blue dot). Finally, 50 Monte Carlo samples were drawn from the solution space (thin blue lines). The figures display the ground truth geodesic between start and target (thin black line), as well as the average of all Monte Carlo samples (dashed thick line). See Waniek, 2018 for description of the particle system, and Section 4 and Appendix A for details about the algorithms.

simulation did not yield the perfect solution, the samples and, in particular, the sample mean are close to the geodesic between the two points. The deviation from the geodesic is primarily due to the placement of the particles and distribution of symbols, and the expansion from the start symbol towards the goal symbol. That is, the MTS is not aware of real physical distance between locations, and treats all transitions with equal cost. Moreover, the search from start towards the to goal propagates in discrete steps in this simulation and thus has multiple solutions with equal cost, especially in the beginning.

Note that the presented simulation results do not imply that grid fields necessarily arrange in the depicted way on a hemisphere. Rather, the purpose of the simulation is to demonstrate that MTS can compute (approximately) shortest paths even when the input space is non-Euclidean. This appears to be particularly important given recent evidence that shows grid field deformations due to changes in the environmental reward structure (Boccara et al., 2019; Butler et al., 2019).

Source code for all simulations presented in this paper can be found at https://github.com/rochus/transitionscalespace.

References

Adamek, J., Herrlich, H., & Strecker, G. E. (2009). *Abstract and concrete categories: The joy of cats (dover books on mathematics)*. Dover Publications. Retrieved from https://www.amazon.com/Abstract-Concrete-Categories-Dover-

Mathematics/dp/0486469344?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0486469344

Banino, A., Barry, C., Uria, B., Blundell, C., Lillicrap, T., Mirowski, P., ... Kumaran, D. (2018). Vector-based navigation using grid-like representations in artificial agents. *Nature*, *557*(7705), 429–433. doi:10.1038/s41586-018-0102-6

Behnke, S. (2004). Local multiresolution path planning. In D. Polani, B. Browning, A. Bonarini, & K. Yoshida (Eds.), *Robocup 2003: Robot soccer world cup vii* (pp. 332–343). Berlin, Heidelberg: Springer Berlin Heidelberg.

Behnke, S., & Rojas, R. (1998). Neural abstraction pyramid: A hierarchical image understanding architecture. In *1998 ieee international joint conference on neural networks proceedings. ieee world congress on computational intelligence (cat. no.98ch36227)* (Vol. 2, 820–825 vol.2). doi:10.1109/IJCNN.1998.685873

Bender, M. A., Farach-Colton, M., Johnson, R., Mauras, S., Mayer, T., Phillips, C. A., & Xu, H. (2017). Write-optimized skip lists. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI symposium on principles of database systems - PODS '17*. doi:10.1145/3034786.3056117

Bjerknes, T. L., Dagslott, N. C., Moser, E. I., & Moser, M.-B. (2018). Path integration in place cells of developing rats. *Proceedings of the National Academy of Sciences*, *115*(7), E1637–E1646. doi:10.1073/pnas.1719054115

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

doi:10.1126/science.aav4837. eprint: http://science.sciencemag.org/content/363/6434/1443.full.pdf

Bonnevie, T., Dunn, B., Fyhn, M., Hafting, T., Derdikman, D., Kubie, J. L., ... Moser, M.-B. (2013). Grid cells require excitatory drive from the hippocampus. *Nat Neurosci*, *16*(3), 309–317. doi:10.1038/nn.3311

Brankačk, J., Steward, M., & Fox, S. E. (1993). Current source density analysis of the hippocampal theta rhythm: Associated sustained potentials and candidate synaptic generators. *Brain Research*, *615*(2), 310–327. doi:10.1016/0006-8993(93)90043-m

Brun, V. H., Solstad, T., Kjelstrup, K. B., Fyhn, M., Witter, M. P., Moser, E. I., & Moser, M.-B. (2008). Progressive increase in grid scale from dorsal to ventral medial entorhinal cortex. *Hippocampus*, *18*(12), 1200–1212. doi:10.1002/hipo.20504

Burak, Y. (2006). Do we understand the emergent dynamics of grid cell activity? *Journal of Neuroscience*, *26*(37), 9352–9354. doi:10.1523/jneurosci.2857-06.2006

Burak, Y., & Fiete, I. (2009). Accurate path integration in continuous attractor network models of grid cells. *PLOS Computational Biology*, *5*(2), 1–16. doi:10.1371/journal.pcbi.1000291

Bush, D., Barry, C., & Burgess, N. (2014). What do grid cells contribute to place cell firing? *Trends Neurosci. 37*(3), 136–145.

Butler, W. N., Hardcastle, K., & Giocomo, L. M. (2019). Remembered reward locations restructure entorhinal spatial maps. *Science*, *363*(6434), 1447–1452. doi:10.1126/science.aav5297

Butts, D. A., & Goldman, M. S. (2006). Tuning curves, neuronal variability, and sensory coding. *PLoS Biol. 4*(4), e92.

Buzsáki, G., Lai-Wo S., L., & Vanderwolf, C. H. (1983). Cellular bases of hippocampal EEG in the behaving rat. *Brain Research Reviews*, *6*(2), 139–171. doi:10.1016/0165-0173(83)90037-1

Chen, G., Lu, Y., King, J. A., Cacucci, F., & Burgess, N. (2019). Differential influences of environment and self-motion on place and grid cell firing. *Nature Communications*, *10*(1). doi:10.1038/s41467-019-08550-1

Colgin, L. L. (2016). Rhythms of the hippocampal network. *Nature Reviews Neuroscience*, *17*(4), 239–249. doi:10.1038/nrn.2016.21

Couey, J. J., Witoelar, A., Zhang, S.-J., Zheng, K., Ye, J., Dunn, B., . . . Witter, M. P. (2013). Recurrent inhibitory circuitry as a mechanism for grid formation. *Nat. Neurosci. 16*(3), 318–324.

Cover, T. M., & Thomas, J. A. (2006). *Elements of information theory (wiley series in telecommunications and signal processing)*. Wiley-Interscience.

Crane, K., Weischedel, C., & Wardetzky, M. (2013). Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Trans. Graph. 32*(5), 152:1–152:11. doi:10.1145/2516971.2516977

Cueva, C. J., & Wei, X.-X. (2018). Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. In

International conference on learning representations. Retrieved from https://openreview.net/forum?id=B17JTOe0-

Cuperlier, N., Laroque, P., Gaussier, P., & Quoy, M. (2004). Planning and navigation strategies using transition cells and neural fields. In *Proc. of asc/iasted 2004 (artificial intelligence and soft computing / international association of science and technology for development)*, Marbella. Retrieved from http://publi-etis.ensea.fr/2004/CLGQ04

Cuperlier, N., Quoy, M., Giovannangeli, C., Gaussier, P., & Laroque, P. (2006). Transition cells for navigation and planning in an unknown environment. In S. Nolfi, G. Baldassarre, R. Calabretta, J. C. T. Hallam, D. Marocco, J.-A. Meyer, . . . D. Parisi (Eds.), *From animals to animats 9: 9th international conference on simulation of adaptive behavior, sab 2006, rome, italy, september 25-29, 2006. proceedings* (pp. 286–297). doi:10.1007/11840541_24

Dabaghian, Y., Brandt, V. L., & Frank, L. M. (2014). Reconceiving the hippocampal map as a topological template. *eLife*, *3*, e03476. doi:10.7554/eLife.03476

Dabaghian, Y., Memoli, F., Frank, L. M., & Carlsson, G. (2012). A topological paradigm for hippocampal spatial map formation using persistent homology. *PLoS Comput. Biol. 8*(8), e1002581.

Davoudi, H., & Foster, D. J. (2019). Acute silencing of hippocampal CA3 reveals a dominant role in place field responses. *Nature Neuroscience*. doi:10.1038/s41593-018-0321-z

Dayan, P. (1993). Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, *5*(4), 613–624. doi:10.1162/neco.1993.5.4.613

de Berg, M., van Kreveld, M., Overmars, M., & Schwarzkopf, O. (1997). *Computational geometry*. doi:10.1007/978-3-662-03427-9

de Lavilléon, G., Masako Lacroix, M., Rondi-Reig, L., & Benchenane, K. (2015). Explicit memory creation during sleep demonstrates a causal role of place cells in navigation. *Nature Neuroscience*, *18*(4), 493–495. doi:10.1038/nn.3970

Dean, B. C., & Jones, Z. H. (2007). Exploring the duality between skip lists and binary search trees. In *Proceedings of the 45th annual southeast regional conference* (pp. 395–399). ACM-SE 45. doi:10.1145/1233341.1233413

Dordek, Y., Soudry, D., Meir, R., & Derdikman, D. (2016). Extracting grid cell characteristics from place cell inputs using non-negative principal component analysis. *eLife*, *5*. doi:10.7554/elife.10094

Dragoi, G., & Buzsáki, G. (2006). Temporal encoding of place sequences by hippocampal cell assemblies. *Neuron*, *50*(1), 145–157.

Dupret, D., O'Neill, J., Pleydell-Bouverie, B., & Csicsvari, J. (2010). The reorganization and reactivation of hippocampal maps predict spatial memory performance. *Nature Neuroscience*, *13*(8), 995–1002. doi:10.1038/nn.2599

Duvelle, É., Grieves, R. M., Hok, V., Poucet, B., Arelo, A., Jeffery, K., & Save, E. (2019). Insensitivity of place cells to the value of spatial goals in a two-choice flexible navigation task. *The Journal of Neuroscience*, 1578–18. doi:10.1523/jneurosci.1578-18.2018

Eppstein, D., Goodrich, M. T., & Sun, J. Z. (2005). The skip quadtree. In *Proceedings of the twenty-first annual symposium on computational geometry - SCG '05*. doi:10.1145/1064092.1064138

Erdem, U. M., & Hasselmo, M. E. (2012). A goal-directed spatial navigation model using forward trajectory planning based on grid cells. *European Journal of Neuroscience*, *35*(6), 916–931. doi:10.1111/j.1460-9568.2012.08015.x

Erdem, U. M., & Hasselmo, M. E. (2014). A biologically inspired hierarchical goal directed navigation model. *J. Physiol. Paris*, *108*(1), 28–37.

Erdem, U. M., Milford, M. J., & Hasselmo, M. E. (2015). A hierarchical model of goal directed navigation selects trajectories in a visual environment. *Neurobiology of Learning and Memory*, *117*, 109–121. Memory and decision making. doi:http://dx.doi.org/10.1016/j.nlm.2014.07.003

Fellemann, D. J., & Van Essen, D. C. (1991). Distributed hierarchical processing in the primate cerebral cortex. *Cereb. Cortex*, *1*(1), 1–47.

Fiete, I., Burak, Y., & Brookings, T. (2008). What grid cells convey about rat location. *Journal of Neuroscience*, *28*(27), 6858–6871. doi:10.1523/jneurosci.5684-07.2008

Finkel, R. A., & Bentley, J. L. (1974). Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, *4*(1), 1–9. doi:10.1007/BF00288933

Fishkin, K. P., & Barsky, B. A. (1985). An analysis and algorithm for filling propagation. In *Computer-generated images* (pp. 56–76). doi:10.1007/978-4-431-68033-8_6

Franzius, M., Vollgraf, R., & Wiskott, L. (2007). From grids to places. *Journal of Computational Neuroscience*, *22*(3), 297–299. doi:10.1007/s10827-006-0013-7

Fritzke, B. (1995). A growing neural gas network learns topologies. In *Advances in neural information processing systems 7* (pp. 625–632). MIT Press.

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

Furman, G. (1965). Comparison of models for subtractive and shunting lateral-inhibition in receptor-neuron fields. *Kybernetik*, *2*(6), 257–274. doi:10.1007/bf00274089

Fyhn, M., Solstad, T., & Hafting, T. (2008). Entorhinal grid cells and the neural basis of navigation. In *Hippocampal place fields* (pp. 237–252). doi:10.1093/acprof:oso/9780195323245.003.0018

Georgeson, M. A., May, K. A., Freeman, T. C. A., & Hesse, G. S. (2007). From filters to features: Scale–space analysis of edge and blur coding in human vision. *Journal of Vision*, *7*(13), 7. doi:10.1167/7.13.7

Giocomo, L. M., Moser, M.-B., & Moser, E. I. (2011). Computational models of grid cells. *Neuron*, *71*(4), 589–603. doi:http://dx.doi.org/10.1016/j.neuron.2011.07.023

Gorchetchnikov, A., & Grossberg, S. (2007). Space, time and learning in the hippocampus: How fine spatial and temporal scales are expanded into population codes for behavioral control. *Neural Networks*, *20*(2), 182–193. doi:10.1016/j.neunet.2006.11.007

Graham, B. P., Cutsuridis, V., & Hunter, R. (2010). Associative memory models of hippocampal areas CA1 and CA3. In *Hippocampal microcircuits* (pp. 459–494). doi:10.1007/978-1-4419-0996-1_16

Grieves, R. M., & Dudchenko, P. A. (2013). Cognitive maps and spatial inference in animals: Rats fail to take a novel shortcut, but can take a previously experienced one. *Learning and Motivation*, *44*(2), 81–92. doi:10.1016/j.lmot.2012.08.001

Grossberg, S. (1970). Neural pattern discrimination. *Journal of Theoretical Biology*, *27*(2), 291–337. doi:10.1016/0022-5193(70)90143-8

Gustafson, N. J., & Daw, N. D. (2011). Grid cells, place cells, and geodesic generalization for spatial reinforcement learning. *PLoS Computational Biology*, *7*(10), e1002235. doi:10.1371/journal.pcbi.1002235

Hafting, T., Fyhn, M., Molden, S., Moser, M.-B., & Moser, E. I. (2005). Microstructure of a spatial map in the entorhinal cortex. *Nature*, *436*(7052), 801–806. doi:10.1038/nature03721

Hales, J. B., Schlesiger, M. I., Leutgeb, J. K., Squire, L. R., Leutgeb, S., & Clark, R. E. (2014). Medial entorhinal cortex lesions only partially disrupt hippocampal place cells and hippocampus-dependent place memory. *Cell Reports*, *9*(3), 893–901. doi:10.1016/j.celrep.2014.10.009

Hanson, E. N. (1991). The interval skip list: A data structure for finding all intervals that overlap a point. In F. Dehne, J. Sack, & N. Santoro (Eds.), *Algorithms and data structures* (pp. 153–164). doi:10.1007/bfb0028258

Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems, Science, and Cybernetics*, *SSC-4*(2), 100–107.

Hasselmo, M. E., Bodelón, C., & Wyble, B. P. (2002). A proposed function for hippocampal theta rhythm: Separate phases of encoding and retrieval enhance reversal of prior learning. *Neural Computation*, *14*(4), 793–817. doi:10.1162/089976602317318965

Hasselmo, M. E., & Eichenbaum, H. (2005). Hippocampal mechanisms for the context-dependent retrieval of episodes. *Neural Networks*, *18*(9), 1172–1190. doi:10.1016/j.neunet.2005.08.007

Hinman, J. R., Brandon, M. P., Climer, J. R., Chapman, G. W., & Hasselmo, M. E. (2016). Multiple running speed signals in medial entorhinal cortex. *Neuron*, *91*(3), 666–679. doi:10.1016/j.neuron.2016.06.027

Hirel, J., Gaussier, P., Quoy, M., & Banquet, J.-P. (2010). Why and how hippocampal transition cells can be used in reinforcement learning. In S. Doncieux, B. Girard, A. Guillot, J. Hallam, J.-A. Meyer, & J.-B. Mouret (Eds.), *From animals to animats 11: 11th international conference on simulation of adaptive behavior, sab 2010, paris - clos lucé, france, august 25-28, 2010. proceedings* (pp. 359–369). doi:10.1007/978-3-642-15193-4_34

Hollup, S. A., Molde, S., Donnett, J. G., Moser, M.-B., & Moser, E. I. (2001). Accumulation of hippocampal place fields at the goal location in an annular watermaze task. *The Journal of Neuroscience*, *21*(5), 1635–1644. doi:10.1523/jneurosci.21-05-01635.2001

Hwang, J. Y., Kim, J. S., Lim, S. S., & Park, K. H. (2003). A fast path planning by path graph optimization. *IEEE Trans. Systems, Man, and Cybernetics, Part A*, *33*(1), 121–129. doi:10.1109/TSMCA.2003.812599

Jadhav, S. P., Kemere, C., German, P. W., & Frank, L. M. (2012). Awake hippocampal sharp-wave ripples support spatial memory. *Science*, *336*(6087), 1454–1458. doi:10.1126/science.1217230

Jazayeri, M., & Movshon, J. A. (2006). Optimal representation of sensory information by neural populations. *Nat Neurosci*, *9*(5), 690–696. doi:10.1038/nn1691

Jeewajee, A., Barry, C., Douchamps, V., Manson, D., Lever, C., & Burgess, N. (2013). Theta phase precession of grid and place cell firing in open environments. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, *369*(1635). doi:10.1098/rstb.2012.0532

Jeffery, K. J. (2007). Self-localization and the entorhinal–hippocampal system. *Current Opinion in Neurobiology*, *17*(6), 684–691. doi:10.1016/j.conb.2007.11.008

Kambhampati, S., & Davis, L. S. (1986). Multiresolution path planning for mobile robots. *IEEE J. Robotics and Automation*, *2*(3), 135–145. doi:10.1109/JRA.1986.1087051

Knuth, D. E. (1998). *The art of computer programming, volume 3: (2nd ed.) sorting and searching*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc.

Koenderink, J. J., & van Doorn, A. J. (1990). Receptive field families. *Biological Cybernetics*, *63*(4), 291–297. doi:10.1007/bf00203452

Kropff, E., Carmichael, J. E., Moser, M.-B., & Moser, E. I. (2015). Speed cells in the medial entorhinal cortex. *Nature*, *523*(7561), 419–424. Article. Retrieved from http://dx.doi.org/10.1038/nature14622

Krupic, J., Bauza, M., Burton, S., Lever, C., & O'Keefe, J. (2014). How environment geometry affects grid cell symmetry and what we can learn from it. *Philos. Trans. R. Soc. Lond., B, Biol. Sci. 369*(1635), 20130188.

Kubie, J. L., & Fenton, A. A. (2012). Linear look-ahead in conjunctive cells: an entorhinal mechanism for vector-based navigation. *Front Neural Circuits*, *6*, 20.

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

Lee, I., Griffin, A. L., Zilli, E. A., Eichenbaum, H., & Hasselmo, M. E. (2006). Gradual translocation of spatial correlates of neuronal firing in the hippocampus toward prospective reward locations. *Neuron*, *51*(5), 639–650. doi:10.1016/j.neuron.2006.06.033

Leibold, C., & Monsalve-Mercado, M. M. (2017). Traveling theta waves and the hippocampal phase code. *Scientific Reports*, *7*(1). doi:10.1038/s41598-017-08053-3

Lever, C., Burton, S., Jeewajee, A., O'Keefe, J., & Burgess, N. (2009). Boundary vector cells in the subiculum of the hippocampal formation. *Journal of Neuroscience*, *29*(31), 9771–9777. doi:10.1523/jneurosci.1319-09.2009

Lindeberg, T. (1994). Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of Applied Statistics*, *21*, 225–270.

Lindeberg, T. (2010). *Scale-space theory in computer vision (the springer international series in engineering and computer science)*. Springer.

Lindeberg, T. (2013). A computational theory of visual receptive fields. *Biol Cybern*, *107*(6), 589–635.

Lindeberg, T. (2015). Image matching using generalized scale-space interest points. *Journal of Mathematical Imaging and Vision*, *52*(1), 3–36. doi:10.1007/s10851-014-0541-0

Lisman, J. (2005). The theta/gamma discrete phase code occuring during the hippocampal phase precession may be a more general brain coding scheme. *Hippocampus*, *15*(7), 913–922. doi:10.1002/hipo.20121

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, *60*(2), 91–110. doi:10.1023/B:VISI.0000029664.99615.94

Lu, Y., Huo, X., Arslan, O., & Tsiotras, P. (2011). Incremental multi-scale search algorithm for dynamic path planning with low worst-case complexity. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, *41*(6), 1556–1570. doi:10.1109/TSMCB.2011.2157493

Lubenov, E. V., & Siapas, A. G. (2009). Hippocampal theta oscillations are travelling waves. *Nature*, *459*(7246), 534–539. doi:10.1038/nature08010

Lynch, N. A. (1996). *Distributed algorithms*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

MacDonald, J. D., & Booth, K. S. (1990). Heuristics for ray tracing using space subdivision. *The Visual Computer*, *6*(3), 153–166. doi:10.1007/bf01911006

Mamad, O., Stumpp, L., McNamara, H. M., Ramakrishnan, C., Deisseroth, K., Reilly, R. B., & Tsanov, M. (2017). Place field assembly distribution encodes preferred locations. *PLOS Biology*, *15*(9), e2002365. doi:10.1371/journal.pbio.2002365

Marr, D. (1971). Simple memory: A theory for archicortex. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *262*(841), 23–81. doi:10.1098/rstb.1971.0078

Marr, D. (1982). *Vision: A computational investigation into the human representation and processing of visual information*. New York, NY, USA: Henry Holt and Co., Inc.

Marr, D., & Hildreth, E. (1980). Theory of edge detection. *Proc. R. Soc. Lond., B, Biol. Sci. 207*(1167), 187–217.

Masson, C., & Girard, B. (2010). Decoding the grid cells for metric navigation using the residue numeral system. In *Advances in cognitive neurodynamics (II)* (pp. 459–464). doi:10.1007/978-90-481-9695-1_73

Mathis, A., Herz, A. V. M., & Stemmler, M. (2012). Optimal population codes for space: Grid cells outperform place cells. *Neural Computation*, *24*(9), 2280–2317. doi:10.1162/NECO_a_00319

McNamee, D., Wolpert, D. M., & Lengyel, M. (2016). Efficient state-space modularization for planning: Theory, behavioral and neural signatures. In *Advances in neural information processing systems* (pp. 4511–4519).

McNaughton, B. L., Battaglia, F. P., Jensen, O., Moser, E. I., & Moser, M.-B. (2006). Path integration and the neural basis of the 'cognitive map'. *Nat Rev Neurosci*, *7*(8), 663–678. doi:10.1038/nrn1932

McNaughton, B. L., & Morris, R. (1987). Hippocampal synaptic enhancement and information storage within a distributed memory system. *Trends in Neurosciences*, *10*(10), 408–415. doi:10.1016/0166-2236(87)90011-7

Momennejad, I., Russek, E. M., Cheong, J. H., Botvinick, M. M., Daw, N., & Gershman, S. J. (2016). The successor representation in human reinforcement learning. doi:10.1101/083824

Morris, R., Garrud, P., Rawlins, J. N., & O'Keefe, J. (1982). Place navigation impaired in rats with hippocampal lesions. *Nature*, *297*(5868), 681–683.

Moser, E. I., & Moser, M.-B. (2008). A metric for space. *Hippocampus*, *18*(12), 1142–1156.

Moser, E. I., Roudi, Y., Witter, M. P., Kentros, C., Bonhoeffer, T., & Moser, M.-B. (2014). Grid cells and cortical representation. *Nat Rev Neurosci*, *15*(7), 466–481. Review. Retrieved from http://dx.doi.org/10.1038/nrn3766

Mulas, M., Waniek, N., & Conradt, J. (2016). Hebbian plasticity realigns grid cell activity with external sensory cues in continuous attractor models. *Front Comput Neurosci*, *10*, 13. 26924979[pmid]. doi:10.3389/fncom.2016.00013

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

Neumann, H., Pessoa, L., & Hanse, T. (1999). Interaction of on and off pathways for visual contrast measurement. *Biological Cybernetics*, *81*(5), 515–532. doi:10.1007/s004220050580

Newman, E. L., & Hasselmo, M. E. (2014). Grid cell firing properties vary as a function of theta phase locking preferences in the rat medial entorhinal cortex. *Frontiers in Systems Neuroscience*, *8*, 193. doi:10.3389/fnsys.2014.00193

O'Keefe, J. (1979). A review of the hippocampal place cells. *Prog. Neurobiol. 13*(4), 419–439.

O'Keefe, J., & Nadel, L. (1978). *The hippocampus as a cognitive map*. Clarendon Press.

O'Keefe, J., & Recce, M. L. (1993). Phase relationship between hippocampal place units and the EEG theta rhythm. *Hippocampus*, *3*(3), 317–330. doi:10.1002/hipo.450030307

Orenstein, J. A. (1982). Multidimensional tries used for associative searching. *Information Processing Letters*, *14*(4), 150–157. doi:http://dx.doi.org/10.1016/0020-0190(82)90027-8

Pai, D. K., & Reissell, L. (1995). Multiresolution rough terrain motion planning. In *Proceedings of IEEE/RSJ international conference on intelligent robots and systems, IROS 1995, august 5 - 9, 1995, pittsburgh, pa, USA* (pp. 39–44). doi:10.1109/IROS.1995.526136

Palm, G. (1980). On associative memory. *Biological Cybernetics*, *36*(1), 19–31. doi:10.1007/BF00337019

Patel, J., Fujisawa, S., Berényi, A., Royer, S., & Buzsáki, G. (2012). Traveling theta waves along the entire septotemporal axis of the hippocampus. *Neuron*, *75*(3), 410–417. doi:10.1016/j.neuron.2012.07.015

Pfeiffer, B. E., & Foster, D. J. (2013). Hippocampal place-cell sequences depict future paths to remembered goals. *Nature*, *497*(7447), 74–79.

Pilarski, S., & Pokora, O. (2015). On the cramér–rao bound applicability and the role of fisher information in computational neuroscience. *Biosystems*, *136*, 11–22. Selected papers presented at the Eleventh International Workshop on Neural Coding, Versailles, France, 2014. doi:https://doi.org/10.1016/j.biosystems.2015.07.009

Pugh, W. (1990). Skip lists: A probabilistic alternative to balanced trees. *Commun. ACM*, *33*(6), 668–676. doi:10.1145/78973.78977

Ranck, J. B. (1984). Head-direction cells in the deep cell layers of dorsal presubiculum in freely moving rats. *Society for Neuroscience Abstracts*, *10*.

Raynal, M. (2013). *Distributed algorithms for message-passing systems*. Berlin New York: Springer.

Redish, A. D., & Touretzky, D. S. (1998). The role of the hippocampus in solving the morris water maze. *Neural Computation*, *10*(1), 73–111. doi:10.1162/089976698300017908

Redish, A. D., Touretzky, D. S., Ikeuchi, K., & Veloso, M. (1996). Navigating with landmarks: Computing goal locations from place codes.

Rodieck, R. W. (1965). Quantitative analysis of cat retinal ganglion cell response to visual stimuli. *Vision Research*, *5*(12), 583–601. doi:10.1016/0042-6989(65)90033-7

Rowland, D. C., Roudi, Y., Moser, M.-B., & Moser, E. I. (2016). Ten years of grid cells. *Annual Review of Neuroscience*, *39*(1), 19–40. doi:10.1146/annurev-neuro-070815-013824

Samet, H. (1984). The quadtree and related hierarchical data structures. *ACM Comput. Surv. 16*(2), 187–260. doi:10.1145/356924.356930

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), 758–762.

Savelli, F., Yoganarasimha, D., & Knierim, J. J. (2008). Influence of boundary removal on the spatial representations of the medial entorhinal cortex. *Hippocampus*, *18*(12), 1270–1282.

Schmidt, R., Diba, K., Leibold, C., Schmitz, D., Buzsáki, G., & Kempter, R. (2009). Single-trial phase precession in the hippocampus. *Journal of Neuroscience*, *29*(42), 13232–13241. doi:10.1523/jneurosci.2270-09.2009

Seeger, A., & Volle, M. M. (1995). On a convolution operation obtained by adding level sets : Classical and new results. *RAIRO - Operations Research - Recherche Opérationnelle*, *29*(2), 131–154. Retrieved from http://eudml.org/doc/105101

Sethian, J. A. (1996). A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, *93*(4), 1591–1595. doi:10.1073/pnas.93.4.1591

Shipston-Sharman, O., Solanka, L., & Nolan, M. F. (2016). Continuous attractor network models of grid cell firing based on excitatory-inhibitory interactions. *J. Physiol. (Lond.) 594*(22), 6547–6557.

Silvela, J., & Portillo, J. (2001). Breadth-first search and its application to image processing problems. *IEEE Transactions on Image Processing*, *10*(8), 1194–1199. doi:10.1109/83.935035

Skaggs, W. E., McNaughton, B. L., Wilson, M. A., & Barnes, C. A. (1996). Theta phase precession in hippocampal neuronal populations and the compression of temporal sequences. *Hippocampus*, *6*(2), 149–172. doi:10.1002/(sici)1098-1063(1996)6:2<149::aid-hipo6>3.0.co;2-k

Solstad, T., Moser, E. I., & Einevoll, G. T. (2006). From grid cells to place cells: A mathematical model. *Hippocampus*, *16*(12), 1026–1031.

Sreenivasan, S., & Fiete, I. (2011). Grid cells generate an analog error-correcting code for singularly precise neural computation. *Nat Neurosci*, *14*(10), 1330–1337. Article. Retrieved from http://dx.doi.org/10.1038/nn.2901

Stachenfeld, K. L., Botvinick, M. M., & Gershman, S. J. (2016). The hippocampus as a predictive map. doi:10.1101/097170

Stemmler, M., Mathis, A., & Herz, A. V. M. (2015). Connecting multiple spatial scales to decode the population activity of grid cells. *Science Advances*, *1*(11). doi:10.1126/science.1500816

Stensola, H., Stensola, T., Solstad, T., Froland, K., Moser, M.-B., & Moser, E. I. (2012). The entorhinal grid map is discretized. *Nature*, *492*(7427), 72–78. doi:10.1038/nature11649

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction (adaptive computation and machine learning series)*. A Bradford Book.

Treves, A., & Rolls, E. T. (1994). Computational analysis of the role of the hippocampus in memory. *Hippocampus*, *4*(3), 374–391.

W. Dijkstra, E. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*(1), 269–271. doi:10.1007/bf01386390

Wald, I., Mark, W. R., Günther, J., Boulos, S., Ize, T., Hunt, W., . . . Shirley, S. (2009). State of the art in ray tracing animated scenes. *Computer Graphics Forum*, *28*(6), 1691–1722. doi:10.1111/j.1467-8659.2008.01313.x

Waniek, N. (2018). Hexagonal grid fields optimally encode transitions in spatiotemporal sequences. *Neural Computation*, *30*(10), 2691–2725. doi:10.1162/neco_a_01122

Wei, X.-x., Prentice, J., & Balasubramanian, V. (2015). A principle of economy predicts the functional architecture of grid cells. *eLife*. doi:10.7554/eLife.08362

Wennekers, T., & Palm, G. (2009). Syntactic sequencing in hebbian cell assemblies. *Cogn Neurodyn*, *3*(4), 429–441. 9095[PII]. doi:10.1007/s11571-009-9095-z

Wernle, T., Waaga, T., Mørreaunet, M., Treves, A., Moser, M.-B., & Moser, E. I. (2017). Integration of grid maps in merged environments. *Nature Neuroscience*. doi:10.1038/s41593-017-0036-6

Whittington, J., Muller, T., Mark, S., Barry, C., & Behrens, T. (2018). Generalisation of structural knowledge in the hippocampal-entorhinal system. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in neural information processing systems 31* (pp. 8493–8504). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/8068-generalisation-of-structural-knowledge-in-the-hippocampal-entorhinal-system.pdf

Wills, T. J., Cacucci, F., Burgess, N., & O'Keefe, J. (2010). Development of the hippocampal cognitive map in preweanling rats. *Science*, *328*(5985), 1573–1576.

Willshaw, D. J., Buneman, O. P., & Longuet-Higgins, H. C. (1969). Non-holographic associative memory. *Nature*, *222*(5197), 960–962. doi:10.1038/222960a0

Witkin, A. P. (1983). Scale-space filtering. In *Proceedings of the eighth international joint conference on artificial intelligence - volume 2* (pp. 1019–1022). IJCAI'83. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. Retrieved from http://dl.acm.org/citation.cfm?id=1623516.1623607

Wyss, R., König, P., & Verschure, P. F. M. J. (2006). A model of the ventral visual system based on temporal stability and local memory. *PLoS Biology*, *4*(5), e120. doi:10.1371/journal.pbio.0040120

Young, R. A. (1987). The Gaussian derivative model for spatial vision: I. Retinal mechanisms. *Spat Vis*, *2*(4), 273–293.

Zilli, E. (2012). Models of grid cell spatial firing published 2005-2011. *Frontiers in Neural Circuits*, *6*, 16. doi:10.3389/fncir.2012.00016