Modelling the Grid-like Encoding of Visual Space in Primates

Jochen Kerdels and Gabriele Peters
University of Hagen, Universitätsstrasse 1, D-58097 Hagen, Germany

Keywords: Recursive Growing Neural Gas, Entorhinal Cortex, Primates, Grid-like Encoding, Grid Cell Model.

Abstract: Several regions of the mammalian brain contain neurons that exhibit grid-like firing patterns. The most prominent example of such neurons are *grid cells* in the entorhinal cortex (EC) whose activity correlates with the animal's location. Correspondingly, contemporary models of grid cells interpret this firing behavior as a specialized, functional part within a system for orientation and navigation. However, Killian et al. report on neurons in the primate EC that show similar, grid-like firing patterns but encode gaze-positions in the field of view instead of locations in the environment. We hypothesized that the phenomenon of grid-like firing patterns may not be restricted to navigational tasks and may be related to a more general, underlying information processing scheme. To explore this idea, we developed a grid cell model based on the recursive growing neural gas (RGNG) algorithm that expresses this notion. Here we show that our grid cell model can – in contrast to established grid cell models – also describe the observations of Killian et al. and we outline the general conditions under which we would expect neurons to exhibit grid-like activity patterns in response to input signals independent of a presumed, functional task of the neurons.

1 INTRODUCTION

Within the last decade neurons with grid-like firing patterns were identified in several regions of the mammalian brain. Among these the so-called *grid cells* are most prominent. Their firing pattern correlates with the animal's location resulting in a periodic, triangular lattice of firing fields that cover the animal's environment. Grid cells were first discovered in the entorhinal cortex of rats (Fyhn et al., 2004; Hafting et al., 2005), and could later be shown to exist in mice (Domnisoru et al., 2013), bats (Yartsev et al., 2011), and humans (Jacobs et al., 2013) as well. Besides the entorhinal cortex (rats, mice, bats, humans), grid cells were also found in the pre- and parasubiculum (rats) (Boccara et al., 2010), and the hippocampus, parahippocampal gyrus, amygdala, cingulate cortex, and frontal cortex (humans) (Jacobs et al., 2013), albeit in low numbers. In all reported cases the observed firing patterns correlated with the animal's location[1]. However, Killian et al. (Killian et al., 2012) observed neurons with grid-like firing patterns in the entorhinal cortex of primates that show a different behavior. Instead of encoding positions in physical space the observed neurons encoded the visual space of the animal with a grid-like pattern, i.e., the activity of the neurons correlated with gaze-positions in the animal's field of view.

These observations suggests that grid-like neuronal activity may be a phenomenon that is more widespread and may represent a more general form of information processing as assumed so far. Existing computational models of grid cells (Moser and Moser, 2008; Welinder et al., 2008; Giocomo et al., 2011; Barry and Burgess, 2014; Burak, 2014; Moser et al., 2014) are based on priors that restrict their applicability to the domain of path integration and navigation. Thus, to explore the hypothesis stated above we developed a novel computational model based on the *recursive growing neural gas* (RGNG) algorithm that describes the behavior of neurons with grid-like activities without relying on priors that would restrict the model to a single domain (Kerdels and Peters, 2013; Kerdels and Peters, 2015b; Kerdels, 2016). Here we show that our model is able to describe not only typical *grid cells* but also cells with similar grid-like firing patterns that operate in different domains like the neurons observed by Killian et al. (Killian et al., 2012).

The next section summarizes our RGNG-based model and outlines the general conditions under which the model's neurons exhibit grid-like firing patterns. Based on these conditions, a possible input sig-

[1] In some cases the subjects navigated a virtual environment (Domnisoru et al., 2013; Jacobs et al., 2013).

Kerdels, J. and Peters, G.
Modelling the Grid-like Encoding of Visual Space in Primates.
DOI: 10.5220/0006045500420049
In *Proceedings of the 8th International Joint Conference on Computational Intelligence (IJCCI 2016) - Volume 3: NCTA*, pages 42-49
ISBN: 978-989-758-201-1

nal that may underlie the phenomena reported by Killian et al. is described in section 3. Simulation results demonstrating that such a signal can indeed cause the observed grid-like firing patterns are presented in section 4. Finally, conclusions reflecting on the results are given in section 5.

2 GRID CELL MODEL

The majority of conventional grid cell models rely on mechanisms that directly integrate information on the velocity and direction of an animal into a periodic representation of the animal's location (Kerdels, 2016). As a consequence, the particular models do not generalize well, i.e., they can not be used to describe or investigate the behavior of neurons that receive other kinds of input signals but may also exhibit grid-like firing patterns. In contrast, the RGNG-based model does not rely on specific types of information as input. It describes the general behavior of a group of neurons in response to inputs from arbitrary input spaces.

The main hypothesis of our RGNG-based model is that the behavior observed in grid cells is just one instance of a more general information processing scheme. In this scheme each cell in a group of neurons tries to learn the structure of its entire input space while being in competition with its peers. Learning the input space structure is modelled on a per-cell level with a growing neural gas – an unsupervised learning algorithm that approximates the input space structure with a network of units (Martinetz and Schulten, 1994; Fritzke, 1995). Each unit in this network is associated with a *reference vector* or *prototype* that represents a specific location in input space. The network structure reflects the input space topology, i.e., neighboring units in the network correspond to neighboring regions of input space.

Interestingly, the competition between neurons can be described by the same GNG dynamics as the per-cell learning process resulting in a joint, recursive model that describes both the learning processes within each cell as well as the competition within a group of neurons. At the model's core lies the recursive growing neural gas (RGNG) algorithm, which is described formally in the next section. It represents a generalization of the original GNG algorithm that allows the unit's prototypes to be entire GNGs themselves.

2.1 Recursive Growing Neural Gas

The recursive growing neural gas (RGNG) has essentially the same structure as a regular GNG. Like a GNG an RGNG g can be described by a tuple[2]:

$$g := (U, C, \theta) \in G,$$

with a set U of units, a set C of edges, and a set θ of parameters. Each unit u is described by a tuple:

$$u := (w, e) \in U, \quad w \in W := \mathbb{R}^n \cup G, \ e \in \mathbb{R},$$

with the *prototype* w, and the *accumulated error* e. Note that in contrast to the regular GNG the prototype w of an RGNG unit can either be a n-dimensional vector or another RGNG. Each edge c is described by a tuple:

$$c := (V, t) \in C, \quad V \subseteq U \wedge |V| = 2, \ t \in \mathbb{N},$$

with the units $v \in V$ connected by the edge and the *age* t of the edge. The *direct neighborhood* E_u of a unit $u \in U$ is defined as:

$$E_u := \{k | \exists (V, t) \in C, \ V = \{u, k\}, \ t \in \mathbb{N}\}.$$

The set θ of parameters consists of:

$$\theta := \{\varepsilon_b, \varepsilon_n, \varepsilon_r, \lambda, \tau, \alpha, \beta, M\}.$$

Compared to the regular GNG the set of parameters has grown by $\theta.\varepsilon_r$ and $\theta.M$. The former parameter is a third learning rate used in the adaptation function A (see below). The latter parameter is the maximum number of units in an RGNG. This number refers only to the number of "direct" units in a particular RGNG and does not include potential units present in RGNGs that are prototypes of these direct units.

Like its structure the behavior of the RGNG is basically identical to that of a regular GNG. However, since the prototypes of the units can either be vectors or RGNGs themselves, the behavior is now defined by four functions. The distance function

$$D(x, y) : W \times W \to \mathbb{R}$$

determines the distance either between two vectors, two RGNGs, or a vector and an RGNG. The interpolation function

$$I(x, y) : (\mathbb{R}^n \times \mathbb{R}^n) \cup (G \times G) \to W$$

generates a new vector or new RGNG by interpolating between two vectors or two RGNGs, respectively. The adaptation function

$$A(x, \xi, r) : W \times \mathbb{R}^n \times \mathbb{R} \to W$$

adapts either a vector or RGNG towards the input vector ξ by a given fraction r. Finally, the input function

$$F(g, \xi) : G \times \mathbb{R}^n \to G \times \mathbb{R}$$

feeds an input vector ξ into the RGNG g and returns the modified RGNG as well as the distance between ξ and the best matching unit (BMU, see below) of g. The input function F contains the core of the RGNG's behavior and utilizes the other three functions, but is also used, in turn, by those functions introducing several recursive paths to the program flow.

[2]The notation $g.\alpha$ is used to reference the element α within the tuple.

$F(g,\xi)$: The input function F is a generalized version of the original GNG algorithm that facilitates the use of prototypes other than vectors. In particular, it allows to use RGNGs themselves as prototypes resulting in a recursive structure. An input $\xi \in \mathbb{R}^n$ to the RGNG g is processed by the input function F as follows:

- Find the two units s_1 and s_2 with the smallest distance to the input ξ according to the distance function D:

$$\begin{aligned} s_1 &:= \arg\min_{u \in g.U} D(u.w, \xi), \\ s_2 &:= \arg\min_{u \in g.U \setminus \{s_1\}} D(u.w, \xi). \end{aligned}$$

- Increment the age of all edges connected to s_1:

$$\Delta c.t = 1, \quad c \in g.C \wedge s_1 \in c.V .$$

- If no edge between s_1 and s_2 exists, create one:

$$g.C \Leftarrow g.C \cup \{(\{s_1, s_2\}, 0)\}.$$

- Reset the age of the edge between s_1 and s_2 to zero:

$$c.t \Leftarrow 0, \quad c \in g.C \wedge s_1, s_2 \in c.V .$$

- Add the squared distance between ξ and the prototype of s_1 to the accumulated error of s_1:

$$\Delta s_1.e = D(s_1.w, \xi)^2 .$$

- Adapt the prototype of s_1 and all prototypes of its direct neighbors:

$$\begin{aligned} s_1.w &\Leftarrow A(s_1.w, \xi, g.\theta.\varepsilon_b), \\ s_n.w &\Leftarrow A(s_n.w, \xi, g.\theta.\varepsilon_n), \forall s_n \in E_{s_1}. \end{aligned}$$

- Remove all edges with an age above a given threshold τ and remove all units that no longer have any edges connected to them:

$$\begin{aligned} g.C &\Leftarrow g.C \setminus \{c | c \in g.C \wedge c.t > g.\theta.\tau\}, \\ g.U &\Leftarrow g.U \setminus \{u | u \in g.U \wedge E_u = \emptyset\}. \end{aligned}$$

- If an integer-multiple of $g.\theta.\lambda$ inputs was presented to the RGNG g and $|g.U| < g.\theta.M$, add a new unit u. The new unit is inserted "between" the unit j with the largest accumulated error and the unit k with the largest accumulated error among the direct neighbors of j. Thus, the prototype $u.w$ of the new unit is initialized as:

$$\begin{aligned} u.w := I(j.w, k.w), \quad & j = \arg\max_{l \in g.U} (l.e), \\ & k = \arg\max_{l \in E_j} (l.e). \end{aligned}$$

  The existing edge between units j and k is removed and edges between units j and u as well as units u and k are added:

$$\begin{aligned} g.C &\Leftarrow g.C \setminus \{c | c \in g.C \wedge j,k \in c.V\}, \\ g.C &\Leftarrow g.C \cup \{(\{j,u\}, 0), (\{u,k\}, 0)\}. \end{aligned}$$

  The accumulated errors of units j and k are decreased and the accumulated error $u.e$ of the new unit is set to the decreased accumulated error of unit j:

$$\begin{aligned} \Delta j.e &= -g.\theta.\alpha \cdot j.e, \quad \Delta k.e = -g.\theta.\alpha \cdot k.e, \\ u.e &:= j.e . \end{aligned}$$

- Finally, decrease the accumulated error of all units:

$$\Delta u.e = -g.\theta.\beta \cdot u.e, \quad \forall u \in g.U .$$

The function F returns the tuple $(g, d_{\min})$ containing the now updated RGNG g and the distance $d_{\min} := D(s_1.w, \xi)$ between the prototype of unit s_1 and input ξ. Note that in contrast to the regular GNG there is no stopping criterion any more, i.e., the RGNG operates explicitly in an online fashion by continuously integrating new inputs. To prevent unbounded growth of the RGNG the maximum number of units $\theta.M$ was introduced to the set of parameters.

$D(x,y)$: The distance function D determines the distance between two prototypes x and y. The calculation of the actual distance depends on whether x and y are both vectors, a combination of vector and RGNG, or both RGNGs:

$$D(x,y) := \begin{cases} D_{RR}(x,y) & \text{if } x,y \in \mathbb{R}^n, \\ D_{GR}(x,y) & \text{if } x \in G \wedge y \in \mathbb{R}^n, \\ D_{RG}(x,y) & \text{if } x \in \mathbb{R}^n \wedge y \in G, \\ D_{GG}(x,y) & \text{if } x,y \in G. \end{cases}$$

In case the arguments of D are both vectors, the Minkowski distance is used:

$$\begin{aligned} D_{RR}(x,y) := \left(\sum_{i=1}^n |x_i - y_i|^p\right)^{\frac{1}{p}}, \quad & x = (x_1, \ldots, x_n), \\ & y = (y_1, \ldots, y_n), \\ & p \in \mathbb{N}. \end{aligned}$$

Using the Minkowski distance instead of the Euclidean distance allows to adjust the distance measure with respect to certain types of inputs via the parameter p. For example, setting p to higher values results in an emphasis of large changes in individual dimensions of the input vector versus changes that are distributed over many dimensions (Kerdels and Peters, 2015a). However, in the case of modeling the behavior of grid cells the parameter is set to a fixed value of 2 which makes the Minkowski distance equivalent to the Euclidean distance. The latter is required in this context as only the Euclidean distance allows the GNG to form an induced Delaunay triangulation of its input space.

In case the arguments of D are a combination of vector and RGNG, the vector is fed into the RGNG using function F and the returned minimum distance is taken as distance value:

$$\begin{aligned} D_{GR}(x,y) &:= F(x,y).d_{\min}, \\ D_{RG}(x,y) &:= D_{GR}(y,x). \end{aligned}$$

In case the arguments of D are both RGNGs, the distance is defined to be the pairwise minimum distance between the prototypes of the RGNGs' units, i.e., *single linkage* distance between the sets of units is used:

$$D_{GG}(x,y) := \min_{u \in x.U, k \in y.U} D(u.w, k.w).$$

The latter case is used by the interpolation function if the recursive depth of an RGNG is at least 2. As the RGNG-based grid cell model has only a recursive depth of 1 (see next section), the case is considered for reasons of completeness rather than necessity. Alternative measures to consider could be, e.g., *average* or *complete* linkage.

$I(x,y)$**:** The interpolation function I returns a new prototype as a result from interpolating between the prototypes x and y. The type of interpolation depends on whether the arguments are both vectors or both RGNGs:

$$I(x,y) := \begin{cases} I_{RR}(x,y) & \text{if } x,y \in \mathbb{R}^n, \\ I_{GG}(x,y) & \text{if } x,y \in G. \end{cases}$$

In case the arguments of I are both vectors, the resulting prototype is the arithmetic mean of the arguments:

$$I_{RR}(x,y) := \frac{x+y}{2}.$$

In case the arguments of I are both RGNGs, the resulting prototype is a new RGNG a. Assuming w.l.o.g. that $|x.U| \geq |y.U|$ the components of the interpolated RGNG a are defined as follows:

$$\begin{aligned} a &:= I(x,y), \\ a.U &:= \left\{ (w,0) \;\middle|\; \begin{array}{l} w = I(u.w, k.w), \\ \forall u \in x.U, \\ k = \arg\min\limits_{l \in y.U} D(u.w, l.w) \end{array} \right\}, \\ a.C &:= \left\{ (\{l,m\},0) \;\middle|\; \begin{array}{ll} \exists c \in x.C & \\ \wedge & u,k \in c.V \\ \wedge & l.w = I(u.w,\cdot) \\ \wedge & m.w = I(k.w,\cdot) \end{array} \right\}, \\ a.\theta &:= x.\theta. \end{aligned}$$

The resulting RGNG a has the same number of units as RGNG x. Each unit of a has a prototype that was interpolated between the prototype of the corresponding unit in x and the nearest prototype found in the units of y. The edges and parameters of a correspond to the edges and parameters of x.

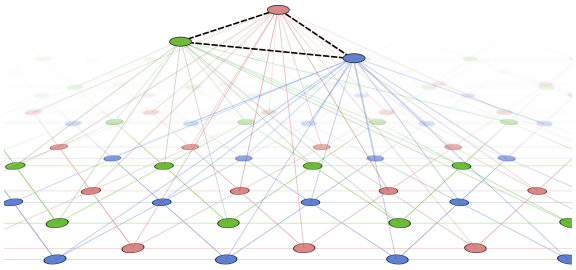

Figure 1: Illustration of the RGNG-based neuron model. The top layer is represented by three units (red, green, blue) connected by dashed edges. The prototypes of the top layer units are themselves RGNGs. The units of these RGNGs are illustrated in the second layer by corresponding colors.

$A(x,\xi,r)$**:** The adaptation function A adapts a prototype x towards a vector ξ by a given fraction r. The type of adaptation depends on whether the given prototype is a vector or an RGNG:

$$A(x,\xi,r) := \begin{cases} A_R(x,\xi,r) & \text{if } x \in \mathbb{R}^n, \\ A_G(x,\xi,r) & \text{if } x \in G. \end{cases}$$

In case prototype x is a vector, the adaptation is performed as linear interpolation:

$$A_R(x,\xi,r) := (1-r)x + r\xi.$$

In case prototype x is an RGNG, the adaptation is performed by feeding ξ into the RGNG. Importantly, the parameters ε_b and ε_n of the RGNG are temporarily changed to take the fraction r into account:

$$\begin{aligned} \theta^* &:= (\, r,\; r \cdot x.\theta.\varepsilon_r,\; x.\theta.\varepsilon_r,\; x.\theta.\lambda,\; x.\theta.\tau, \\ &\qquad x.\theta.\alpha,\; x.\theta.\beta,\; x.\theta.M), \\ x^* &:= (x.U, x.C, \theta^*), \\ A_G(x,\xi,r) &:= F(x^*,\xi).x. \end{aligned}$$

Note that in this case the new parameter $\theta.\varepsilon_r$ is used to derive a temporary ε_n from the fraction r.

This concludes the formal definition of the RGNG algorithm.

2.2 RGNG-based Neuron Model

The RGNG-based neuron model uses a single RGNG to describe a group of neurons that compete against each other. The RGNG has a recursive depth of one resulting in a two-layered structure (Fig. 1). The units

Table 1: Parameters of the RGNG-based model.

θ_1			θ_2		
ε_b	=	0.04	ε_b	=	0.01
ε_n	=	0.04	ε_n	=	0.0001
ε_r	=	0.01	ε_r	=	0.01
λ	=	1000	λ	=	1000
τ	=	300	τ	=	300
α	=	0.5	α	=	0.5
β	=	0.0005	β	=	0.0005
M	=	100	M	=	$\{20, 80\}$

in the top layer (TL) correspond to the individual neurons of the group. The prototypes of these TL units are RGNGs themselves and can be interpreted as the dendritic trees of the corresponding neurons. They constitute the bottom layer (BL) of the model. The prototypes of the BL units are regular vectors representing specific locations in input space.

2.2.1 Parameterization

Top and bottom layer of the model have their own set of parameters θ_1 and θ_2, respectively. Parameter set θ_1 controls the main TL RGNG while parameter set θ_2 controls all BL RGNGs, i.e., the prototypes of the TL units. Table 1 summarizes typical parameter values for an RGNG-based neuron model. For a detailed characterization of these parameters we refer to (Kerdels, 2016). Parameter $\theta_1.M$ sets the maximum number of neurons in the model and Parameter $\theta_2.M$ controls the maximum number of specific input space locations that a single neuron can represent within its dendritic tree. The learning rates $\theta_2.\varepsilon_b$ and $\theta_2.\varepsilon_n$ determine to which degree each neuron performs a form of sub-threshold, competition independent learning. In contrast, learning rate $\theta_1.\varepsilon_b$ controls how strongly the single best[3] neuron in the group adapts towards a particular input. Finally, learning rate $\theta_1.\varepsilon_n$ can be interpreted as being inversely proportional to the lateral inhibition that the most active neuron exerts on its peers.

2.2.2 Learning

The RGNG algorithm does not have an explicit training phase. It learns continuously and updates its input space approximation with every input. In case of the RGNG-based neuron model the learning process can be understood as a mixture of two processes. For each input the TL RGNG has to determine which TL unit lies closest to the input based on the distance function D. The distance function will in turn feed

[3]The neuron that is most active for a given input.

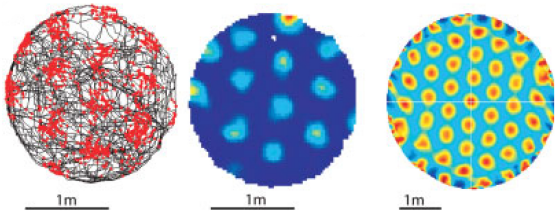

Figure 2: Typical visualization of a grid cell's firing pattern as introduced by Hafting et al. (Hafting et al., 2005). **Left**: trajectory (black lines) of a rat in a circular environment with marked locations (red dots) where the observed grid cell fired. **Middle**: color-coded firing rate map of the observed grid cell ranging from dark blue (no activity) to red (maximum activity). **Right**: color-coded spatial autocorrelation of the firing rate map ranging from blue (negative correlation, -1) to red (positive correlation, +1) highlighting the hexagonal structure of the firing pattern. Figure from Moser et al. (Moser and Moser, 2008).

the input to each BL RGNG allowing the BL RGNGs to learn the input space structure independently from each other. Once the closest TL unit is identified this single unit and its direct neighbors are allowed to preferentially adapt towards the respective input. This second learning step aligns the individual input space representations of the BL RGNGs in such a way that they evenly interleave and collectively cover the input space as well as possible (Kerdels, 2016).

2.2.3 Activity Approximation

Observations of biological neurons typically use the momentary firing rate of a neuron as indicator of the neuron's activity. This activity can then be correlated with other observed variables such as the animal's location. The result of such a correlation can be visualized as, e.g., *rate map* (Fig. 2), which is then used as the basis for further analysis. Accordingly, the RGNG-based neuron model has to estimate the activity of each TL unit in response to a given input ξ as well. To this end, the "activity" a_u of a TL unit u is defined as:

$$a_u := e^{-\frac{(1-r)^2}{2\sigma^2}},$$

with $\sigma = 0.2$ and ratio r:

$$r := \frac{D(s_2.w, \xi) - D(s_1.w, \xi)}{D(s_1.w, s_2.w)}, \quad s_1, s_2 \in u.w.U,$$

with BL units s_1 and s_2 being the BMU and second BMU in $u.w.U$ with respect to input ξ. Based on this measure of activity it is possible to correlate the responses of individual, simulated neurons (TL units) to the given inputs and compare the resulting artificial rate maps with observations reported in the literature.

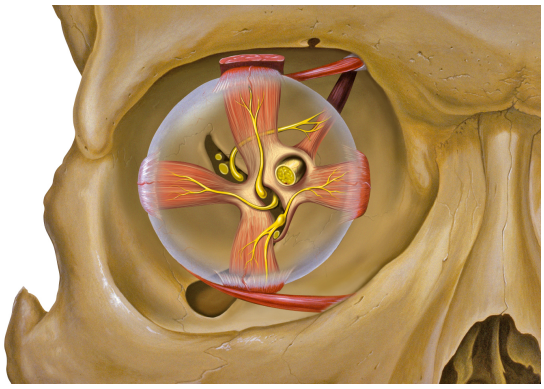

Figure 3: Eye and orbit anatomy with motor nerves by Patrick J. Lynch, medical illustrator; C. Carl Jaffe, MD, cardiologist (CC BY 2.5).

2.3 Input Space Properties

The RGNG-based neuron model can approximate arbitrary input spaces as long as the set of inputs constitute a sufficiently dense sampling of the underlying space. However, not every input space will result in a grid-like firing pattern. To observe such a pattern, the input space has to exhibit certain characteristics. Specifically, the input samples have to originate from a two-dimensional, uniformly distributed manifold. Since the BL RGNGs will approximate such a manifold with an induced Delaunay triangulation the BL prototypes will be spread evenly across the manifold in a periodic, triangular pattern.

This predictability of the RGNG-based model allows to form testable hypothesis about potential input signals with respect to the activity observed in biological neurons. An example for such a hypothesis and the resulting simulation outcomes are presented in the next sections.

3 GAZE-POSITION INPUT

Killian et al. (Killian et al., 2012) report on neurons in the primate medial entorhinal cortex (MEC) that show grid-like firing patterns similar to those of typical grid cells present in the MEC of rats (Fyhn et al., 2004; Hafting et al., 2005). Their finding is especially remarkable since it is the first observation of a grid-like firing pattern that is not correlated with the animal's location. Instead, the observed activity is correlated with gaze-positions in the animal's field of view.

Given a suitable input space the RGNG-based neuron model can replicate this firing behavior. The gaze-position in primates is essentially determined by the four main muscles attached to the eye (Fig. 3). Thus, a possible input signal to the RGNG-based model could originate[4] from the population of motor neurons that control these muscles. In such a signal the number of neurons that are active for a particular muscle determines how strongly this muscle contracts. A corresponding input signal $\xi := (v^{x_0}, v^{x_1}, v^{y_0}, v^{y_1})$ for a given normalized gaze position (x, y) can be implemented as four concatenated d-dimensional vectors v^{x_0}, v^{x_1}, v^{y_0} and v^{y_1}:

$$
\begin{aligned}
v_i^{x_0} &:= \max\left[\min\left[1-\delta\left(\tfrac{i+1}{d}-x\right),\,1\right],\,0\right],\\
v_i^{x_1} &:= \max\left[\min\left[1-\delta\left(\tfrac{i+1}{d}-(1-x)\right),\,1\right],\,0\right],\\
v_i^{y_0} &:= \max\left[\min\left[1-\delta\left(\tfrac{i+1}{d}-y\right),\,1\right],\,0\right],\\
v_i^{y_1} &:= \max\left[\min\left[1-\delta\left(\tfrac{i+1}{d}-(1-y)\right),\,1\right],\,0\right],\\
&\forall i \in \{0\ldots d-1\},
\end{aligned}
$$

with $\delta = 4$ defining the "steepness" of the population signal and the size d of each motor neuron population.

4 SIMULATION RESULTS

To test the input space described in the previous section we conducted a set of simulation runs with varying sizes d of the presumed motor neuron populations. All simulations used the fixed set of parameters given in table 1 (with $\theta_2.M = 20$) and processed random gaze-positions. The resulting artificial rate maps that correlate each TL unit's activity with corresponding gaze-positions were used to calculate *gridness* score distributions for each run. The *gridness* score was introduced by Sargolini et al. as a measure of how grid-like an observed firing pattern is (Sargolini et al., 2006). Gridness scores range from -2 to 2 with scores greater than zero indicating a grid-like firing pattern, albeit more conservative thresholds between 0.3 and 0.4 are chosen in recent publications.

Figure 4 summarizes the results of the simulation runs. The shown rate maps as well as the gridness score distributions show that the neurons described by the RGNG-based model form grid-like firing patterns in response to the population signal defined above. Even small population sizes yield a significant proportion of simulated neurons with gridness scores above a 0.4 threshold.

The firing patterns shown in figure 4 cover the entire two-dimensional manifold of the population signal, i.e., the entire field of view. At its borders a strong

[4] As a so-called *efference copy*

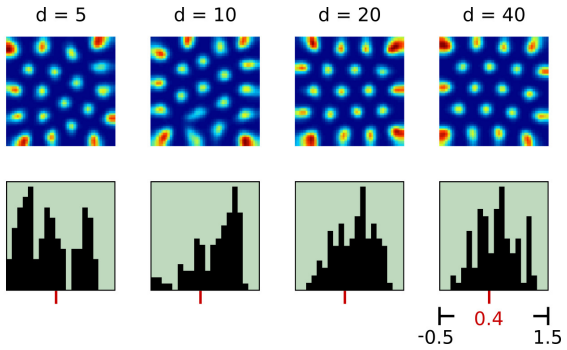

Figure 4: Artificial rate maps and gridness distributions of simulation runs that processed input from a varying number d of presumed motor neurons per muscle (columns) that control the gaze-position. All simulation runs used a fixed set of parameters as given in table 1 ($\theta_2.M = 20$) and processed random gaze locations. Each artificial rate map was chosen randomly from the particular set of rate maps and displays all of the respective TL unit's firing fields. The gridness distributions show the gridness values of all TL units. Gridness threshold of 0.4 indicated by red marks.

alignment of the firing fields can be observed which may influence the resulting gridness scores. Since it is unlikely that experimental observations of natural neurons will cover the entire extent of an underlying input space, we investigated how the partial observation of firing fields may influence gridness score distributions. Figure 5 shows the results of a second series of simulation runs. Again, all simulation runs used the set of parameters given in table 1 but with $\theta_2.M = 80$ prototypes per neuron instead of 20. In addition to artificial rate maps that contain all firing fields of the respective neurons, we also generated rate maps containing only one-quarter or one-sixteenth of the particular firing fields. The resulting gridness score distributions show that rate maps based on local subsets of firing fields tend to have higher gridness scores than the rate maps containing all firing fields. This indicates that local distortions of the overall grid pattern remain local with respect to their influence on the gridness of other regions. As a consequence, grid-like firing patterns may be observed in natural neurons that receive signals from input spaces that are only partially two-dimensional and evenly distributed. As long as the experimental conditions restrict the input signals to these regions the resulting firing rate maps will exhibit grid-like patterns. If the input signals shift to other regions of input space, the grid-like firing patterns may then get distorted or fully disappear.

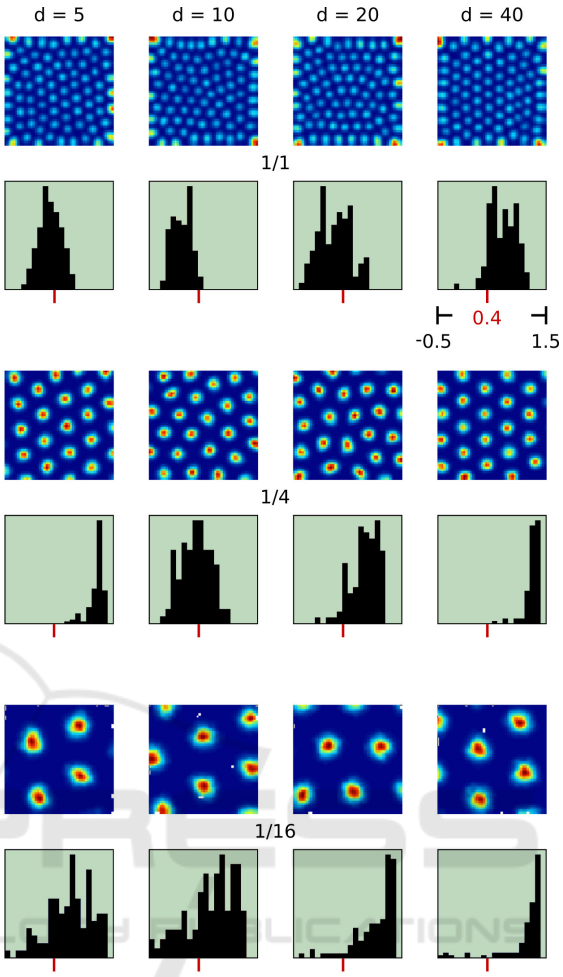

Figure 5: Artificial rate maps and gridness distributions of simulation runs presented like in figure 4, but with parameter $\theta_2.M = 80$ and containing either all, one-quarter, or one-sixteenth (rows) of the respective TL unit's firing fields.

5 CONCLUSIONS

The contemporary view on grid cells interprets their behavior as a specialized part within a system for orientation and navigation that contributes to the integration of speed and direction. However, recent experimental findings indicate that grid-like firing patterns are more prevalent and functionally more diverse than previously assumed. Based on these observations we hypothesized that a more general information processing scheme may underlie the firing behavior of grid cells. We developed a RGNG-based neuron model to test this hypothesis. In this paper we demonstrate that the RGNG-based model can – in contrast to established grid cell models – describe the grid-like activity of neurons other than typical grid

cells, i.e., neurons that encode gaze-positions in the animal's field of view rather than the animal's location in its environment.

In addition, we outlined the general conditions under which we would expect grid-like firing patterns to occur in neurons that utilize the general information processing scheme expressed by the RGNG algorithm. As these conditions depend solely on characteristics of the input signal, i.e., on the *data* that are processed by the respective neurons, the RGNG-based model allows to form testable predictions on the input-output relations of biological, neuronal circuits.

Shifting interpretations of neurobiological circuits from models based on application-specific priors to models based primarily on general, computational principles may prove to be beneficial for the wider understanding of high-level cortical circuits. They may allow to relate experimental observations made in very different contexts on an abstract, computational level and thus promote a deeper understanding of common neuronal principles and structures.

REFERENCES

Barry, C. and Burgess, N. (2014). Neural mechanisms of self-location. *Current Biology*, 24(8):R330 – R339.

Boccara, C. N., Sargolini, F., Thoresen, V. H., Solstad, T., Witter, M. P., Moser, E. I., and Moser, M.-B. (2010). Grid cells in pre- and parasubiculum. *Nat Neurosci*, 13(8):987–994.

Burak, Y. (2014). Spatial coding and attractor dynamics of grid cells in the entorhinal cortex. *Current Opinion in Neurobiology*, 25(0):169 – 175. Theoretical and computational neuroscience.

Domnisoru, C., Kinkhabwala, A. A., and Tank, D. W. (2013). Membrane potential dynamics of grid cells. *Nature*, 495(7440):199–204.

Fritzke, B. (1995). A growing neural gas network learns topologies. In *Advances in Neural Information Processing Systems 7*, pages 625–632. MIT Press.

Fyhn, M., Molden, S., Witter, M. P., Moser, E. I., and Moser, M.-B. (2004). Spatial representation in the entorhinal cortex. *Science*, 305(5688):1258–1264.

Giocomo, L., Moser, M.-B., and Moser, E. (2011). Computational models of grid cells. *Neuron*, 71(4):589 – 603.

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

Jacobs, J., Weidemann, C. T., Miller, J. F., Solway, A., Burke, J. F., Wei, X.-X., Suthana, N., Sperling, M. R., Sharan, A. D., Fried, I., and Kahana, M. J. (2013). Direct recordings of grid-like neuronal activity in human spatial navigation. *Nat Neurosci*, 16(9):1188–1190.

Kerdels, J. (2016). *A Computational Model of Grid Cells based on a Recursive Growing Neural Gas*. PhD thesis, Hagen.

Kerdels, J. and Peters, G. (2013). A computational model of grid cells based on dendritic self-organized learning. In *Proceedings of the International Conference on Neural Computation Theory and Applications*.

Kerdels, J. and Peters, G. (2015a). Analysis of high-dimensional data using local input space histograms. *Neurocomputing*, 169:272 – 280.

Kerdels, J. and Peters, G. (2015b). A new view on grid cells beyond the cognitive map hypothesis. In *8th Conference on Artificial General Intelligence (AGI 2015)*.

Killian, N. J., Jutras, M. J., and Buffalo, E. A. (2012). A map of visual space in the primate entorhinal cortex. *Nature*, 491(7426):761–764.

Martinetz, T. M. and Schulten, K. (1994). Topology representing networks. *Neural Networks*, 7:507–522.

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

Moser, E. I., Moser, M.-B., and Roudi, Y. (2014). Network mechanisms of grid cells. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1635).

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

Welinder, P. E., Burak, Y., and Fiete, I. R. (2008). Grid cells: The position code, neural network models of activity, and the problem of learning. *Hippocampus*, 18(12):1283–1300.

Yartsev, M. M., Witter, M. P., and Ulanovsky, N. (2011). Grid cells without theta oscillations in the entorhinal cortex of bats. *Nature*, 479(7371):103–107.