Recurrent Spiking Networks Solve Planning Tasks

Elmar Rueckert[1,*], David Kappel[2], Daniel Tanneberg[1], Dejan Pecevski[2], and Jan Peters[1,3]

[1]Intelligent Autonomous Systems Lab, Technische Universität Darmstadt, 64289, Germany
[2]Institute for Theoretical Computer Science, Technische Universität Graz, 8020, Austria
[3]Robot Learning Group, Max-Planck Institute for Intelligent Systems, Tuebingen, 72076, Germany
[*]rueckert@ias.tu-darmstadt.de

ABSTRACT

A recurrent spiking neural network is proposed that implements planning as probabilistic inference for finite and infinite horizon tasks. The architecture splits this problem into two parts: The stochastic transient firing of the network embodies the dynamics of the planning task. With appropriate injected input this dynamics is shaped to generate high-reward state trajectories. A general class of reward-modulated plasticity rules for these afferent synapses is presented. The updates optimize the likelihood of getting a reward through a variant of an Expectation Maximization algorithm and learning is guaranteed to convergence to a local maximum. We find that the network dynamics are qualitatively similar to transient firing patterns during planning and foraging in the hippocampus of awake behaving rats. The model extends classical attractor models and provides a testable prediction on identifying modulating contextual information. In a real robot arm reaching and obstacle avoidance task the ability to represent multiple task solutions is investigated. The neural planning method with its local update rules provides the basis for future neuromorphic hardware implementations with promising potentials like large data processing abilities and early initiation of strategies to avoid dangerous situations in robot co-worker scenarios.

Introduction

Probabilistic inference has emerged as a promising framework for solving Markov decision problems (*probabilistic planning*).[1,2] In parallel to this development, the probabilistic inference perspective has also been successfully used in cognitive science and neuroscience for modeling how biological organisms solve planning problems.[3–6] However, it was not clear how probabilistic planning can be implemented in neural substrates with biologically realistic learning rules. Here, we show that probabilistic planning can be implemented and learned by recurrent networks of spiking neurons such that the generated spike sequences realize mental plans.

Recently, it was shown that recurrent spiking networks can implement *Bayesian filtering* and are able to learn a generative model of temporal sequences from presented examples drawn from a target distribution (i.e., the distribution that has generated the samples).[?,7,8] Training was realized through synaptic plasticity rules without supervision. After learning, when running freely, the neural networks effectively perform forward sampling in the dynamic Bayesian network representing the target distribution. For solving a *planning* problem, however, it is necessary to sample from the posterior distribution conditioned on receiving a reward. In other words, forward sampling needs to integrate future rewards propagating backward in time in the dynamic Bayesian network.

We consider a recurrent network of stochastic spiking neurons, which without any input, samples sequences through forward sampling from the corresponding dynamic Bayesian network. We show that by injecting appropriate input in the network from a layer of task-related *context neurons*, it can generate samples from the posterior distribution conditioned on getting a reward. Optimal reward-modulated Hebbian learning rules are derived that implement planning as probabilistic inference in

the spiking network through iterative local updates. The recurrent dynamics of the neural network can be reused for solving different planning problems through activating different sets of context neurons which encode different goals and constraints (e.g., reaching for different goals or avoiding obstacles).

The presented theory is compatible with recent results on transient firing observed in behaving animals in phases of rest.[9–12] In particular, the model reproduces characteristic spatiotemporal spiking activity that preplays movement paths to remembered home locations[9] and provides an alternative to attractor networks,[13] which are typically used to explain movement planning in maze navigation tasks.[14–17] In attractor networks only the end point can be adapted. For non-straight line paths, for example to escape a labyrinth, complex attractor sequence models were proposed.[18] The presented neural model allows to learn such non-straight line paths from single scalar rewards that encode in addition to desired goals and obstacles knowledge about rewarding routes. The ability to modulate the shape of a movement path is a testable prediction for future neuroscience studies.

Another contribution of the presented theory is that it provides the basis for neuromorphic hardware implementations of control and planning strategies in mobile robots. Similar architectures building also on winner-take-all circuits and spike-timing dependent plasticity (STDP) rules were already proposed as computational models for such brain-like chips.[?,?,?] For planning, neuromorphic hardware implementations promise to process large input streams from visual and tactile sensors through parallel computing,[?] go round strategies may be initiate in real-time to avoid dangerous situations in robot co-worker scenarios and event based neural network implementations are energy efficient alternatives to classical von Neumann architectures.[?] In first experiments, we demonstrate that spiking neural networks can be trained to simultaneously represent multiple obstacle avoiding strategies in a real robot arm reaching task.

Methods

We first formulate finite horizon probabilistic planning tasks with terminal rewards only. Later we will generalize to infinite horizons where rewards can be received at any point in time. Let $\boldsymbol{x}_t \in \mathscr{R}^d$ denote the d dimensional continuous state of a behaving agent at time t. The goal of the agent is to find the sequence of states $\underline{\boldsymbol{x}} = (\boldsymbol{x}_1 \ldots \boldsymbol{x}_T)$ that maximizes the total received reward (i.e., the return) $\hat{r}$ at the end of the trial.

Such planning tasks can be modeled as inference problems[1,2] where the joint distribution over state sequences and returns is given by $p(\underline{\boldsymbol{x}}, r) = p(\boldsymbol{x}_0)\, p(r \mid \underline{\boldsymbol{x}}) \prod_{t=1}^{T} \mathscr{T}(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})$. The distribution $p(\boldsymbol{x}_0)$ encodes an initial prior over states, $\mathscr{T}$ corresponds to the state transition model, and the distribution $p(r \mid \underline{\boldsymbol{x}})$ determines the probability of receiving the return r given the trajectory of states $\underline{\boldsymbol{x}}$. As in related work[1,2,19] we assume that $p(r \mid \underline{\boldsymbol{x}})$ can be factorized as $p(r \mid \underline{\boldsymbol{x}}) = 1/Z_r \prod_{t=1}^{T} \psi_t(r|\boldsymbol{x}_t)$. In this formulation, r denotes a binary random variable, where without loss of generality, the probability of observing such a binary return event is a modulo rescaling of the original reward maximization problem.[1]

An agent can use such an internal model of the environment to plan a sequence of movements by solving the inference

problem

$$p\left(\underline{\boldsymbol{x}} \mid r=1\right)=\frac{1}{\mathscr{Z}} p\left(r \mid \underline{\boldsymbol{x}}\right) p\left(\boldsymbol{x}_0\right) \prod_{t=1}^{T} \mathscr{T}\left(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}\right) \quad , \tag{1}$$

where $\mathscr{Z}=\sum_{\underline{\boldsymbol{x}}} p\left(\underline{\boldsymbol{x}}, r\right)$ is a term that guarantees that equation (1) is normalized. In our formulation of the planning problem, the actions in the probabilistic model are integrated out. It is assumed that the actions can be subsequently inferred from the posterior over state sequences.

The unconstrained process for planning models a freely moving agent by

$$p\left(\underline{\boldsymbol{x}}\right)=p\left(\boldsymbol{x}_0\right) \prod_{t=1}^{T} \mathscr{T}\left(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}\right) \quad . \tag{2}$$

Sampling from this probability distribution can be implemented by a recurrent network of spiking neurons (e.g., using ideas from[?,7,8]). However, it is not straightforward for a recurrent network to solve the inference problem in equation (1), which requires to integrate future returns backward in time. Only local temporal information is available when sampling from the network. Such temporal models are different to model-based Markov decision process methods encoding global value or Q functions.[20]

We propose here a solution to this problem that relies on replacing $p\left(\underline{\boldsymbol{x}}\right)$ with a model distribution $q\left(\underline{\boldsymbol{v}}; \boldsymbol{\theta}\right)$, where sampling from $q\left(\underline{\boldsymbol{v}}; \boldsymbol{\theta}\right)$ is implemented with an extended neural network architecture and $\underline{\boldsymbol{v}}$ is the neural approximation of $\underline{\boldsymbol{x}}$. The parameters $\boldsymbol{\theta}$ are learned such that the Kullback-Leibler divergence between the true posterior for planning in equation (1) and a model distribution $q\left(\underline{\boldsymbol{v}}; \boldsymbol{\theta}\right)$ converges to zero.

Planning with recurrent neural networks

We propose a recurrent spiking neural network to implement planning. Our network consists of two populations of neurons, which we denote by Y and V (see Fig. 1A). V is a layer of K *state neurons* that control the state (e.g., the agent's spatial location) of a freely moving agent. These neurons receive lateral connections from neighboring state neurons and from all N neurons in a population of *context neurons* Y with weights w_{ki} and θ_{kj}. The context neurons produce spatiotemporal spike patterns that represent high-level goals and context information (e.g., the target state that should be reached after T time steps). We show that probabilistic planning problems defined in equation (1) can be implemented in the network by training the synapses θ_{kj}.

We denote the activity of the state neurons at time t by $\boldsymbol{\nu}_t=\left(\nu_{t,1}, \ldots, \nu_{t,K}\right)$, where $\nu_{t,k}=1$ if neuron k spiked at time t and $\nu_{t,k}=0$ else. Discrete random variables $\boldsymbol{x}_t$ can be encoded as a multinomial distribution, where one neuron maps to one state instance. For continuous variables a simple encoding scheme is used, i.e., $\boldsymbol{x}_t=1 / \left|\boldsymbol{\nu}_t\right| \sum_{k=1}^{K}\left(\nu_{t,k} \boldsymbol{p}_k\right)$, where $\left|\boldsymbol{\nu}_t\right|=\sum_{k=1}^{K} \nu_{t,k}$ and $\boldsymbol{p}_k$ is the preferred position of state neuron k.

Analogously, we define the spiking activity of the context neurons at time t by a binary vector $\boldsymbol{y}_t=\left(y_{t,1}, \ldots, y_{t,N}\right)$. Using

these definitions of $\boldsymbol{\nu}_t$ and $\boldsymbol{y}_t$, we define the membrane potential $u_{t,k}$ and firing probability $\rho_{t,k}$ of state neuron k at time t by

$$u_{t,k} \;=\; \sum_{i=1}^{K} w_{ki}\,\nu_{t-1,i} + \sum_{j=1}^{N} \theta_{kj}\,y_{t-1,j} \quad \text{and} \quad \rho_{t,k} \;=\; p\left(\nu_{t,k}=1\right) \;=\; f(u_{t,k})\,. \tag{3}$$

The last term $f(u_{t,k})$ denotes the activation function, where we only require that it is differentiable. The probability that the network generates a spike sequences $\underline{\boldsymbol{\nu}} = (\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_T)$ of length T starting from a given initial state $\boldsymbol{\nu}_0$ is thus

$$q\left(\underline{\boldsymbol{\nu}};\boldsymbol{\theta}\right) = p(\boldsymbol{\nu}_0) \prod_{t=1}^{T}\prod_{k=1}^{K} {\rho_{t,k}}^{\nu_{t,k}} \left(1-\rho_{t,k}\right)^{1-\nu_{t,k}}\;. \tag{4}$$

We assume that the transition model (encoded in the synaptic weights w_{ki}) is known or was acquired in a pre-learning phase, e.g., using contrastive divergence learning.[21] Using this assumption, we define the goal of probabilistic planning as minimizing the Kullback-Leibler divergence between the true posterior for planning in equation (1) and the model distribution

$$D_{KL}(p(\underline{\boldsymbol{\nu}}|r=1)\,||\,q\left(\underline{\boldsymbol{\nu}};\boldsymbol{\theta}\right)) = \sum\nolimits_{\underline{\boldsymbol{\nu}}} p(\underline{\boldsymbol{\nu}}|r=1) \log \frac{p(\underline{\boldsymbol{\nu}}|r=1)}{q\left(\underline{\boldsymbol{\nu}};\boldsymbol{\theta}\right)} \;= -H_{p(\underline{\boldsymbol{\nu}}|r=1)} - \left\langle \log q\left(\underline{\boldsymbol{\nu}};\boldsymbol{\theta}\right)\right\rangle_{p(\underline{\boldsymbol{\nu}}|r=1)}\,, \tag{5}$$

where $H_{p(\underline{\boldsymbol{\nu}}|r=1)}$ denotes the entropy of the true data distribution. Thus, solving the inference problem in equation (1) is equal to minimizing the Kullback-Leibler divergence in equation (5).

Typically, $p(\underline{\boldsymbol{\nu}}|r=1)$ is unknown and we cannot draw samples from it. However, we can draw samples from the model distribution $q\left(\underline{\boldsymbol{\nu}};\boldsymbol{\theta}\right)$ and update the parameters such that the probability of receiving a reward event is maximized

$$\Delta\theta_{kj} \;\;=\;\; \eta \left\langle r\,\frac{\partial}{\partial\theta_{kj}} \log q\left(\underline{\boldsymbol{\nu}};\boldsymbol{\theta}\right) \right\rangle_{p(r\,|\,\underline{\boldsymbol{\nu}})\,q(\underline{\boldsymbol{\nu}};\boldsymbol{\theta})}\,, \tag{6}$$

where η denotes a small learning rate. Note that this general update rule is the result of a standard maximum likelihood formulation where we exploited that $p(\underline{\boldsymbol{\nu}}|r=1) \propto p\left(r\,|\,\underline{\boldsymbol{\nu}}\right) p\left(\underline{\boldsymbol{\nu}}\right)$ using equation (1) and equation (2). The update is an instance of the Expectation-Maximization (EM) algorithm,[22] where evaluating the expectation with respect to $p\left(r\,|\,\underline{\boldsymbol{\nu}}\right) q\left(\underline{\boldsymbol{\nu}};\boldsymbol{\theta}\right)$ corresponds to the E-step and the parameter update realizes the M-step. The update is also related to policy gradient methods[?,?,?] with the difference that we interpret the parameters $\boldsymbol{\theta}$ as having the role of linear controls.[23]

To derive the update rule for the proposed neural network architecture, the network dynamics in equation (4) is used in equation (6). For a detailed derivation we refer to the supplement. The spiking network update rule reads

$$\Delta\theta_{kj} \;=\; \eta \left\langle r \sum_{t=1}^{T} \left(\nu_{t,k}-\rho_{t,k}\right) \frac{\partial}{\partial\theta_{kj}} \operatorname{logit}(\rho_{t,k}) \right\rangle_{p(r\,|\,\underline{\boldsymbol{\nu}})\,q(\underline{\boldsymbol{\nu}};\boldsymbol{\theta})}\,, \tag{7}$$

where $\operatorname{logit}(\rho_{t,k}) = \log\left(\rho_{t,k}/(1-\rho_{t,k})\right)$ are the log-odds of neuron k firing at time t. Equation (7) is the general learning rule for arbitrary differentiable activation functions $f(u_{t,k}) \;=\; \rho_{t,k}$. It adapts the weights θ_{kj} to maximize the return. For many relevant activation functions (e.g., exponential or sigmoid functions), equation (7) turns into a simple reward-modulated Hebbian-type

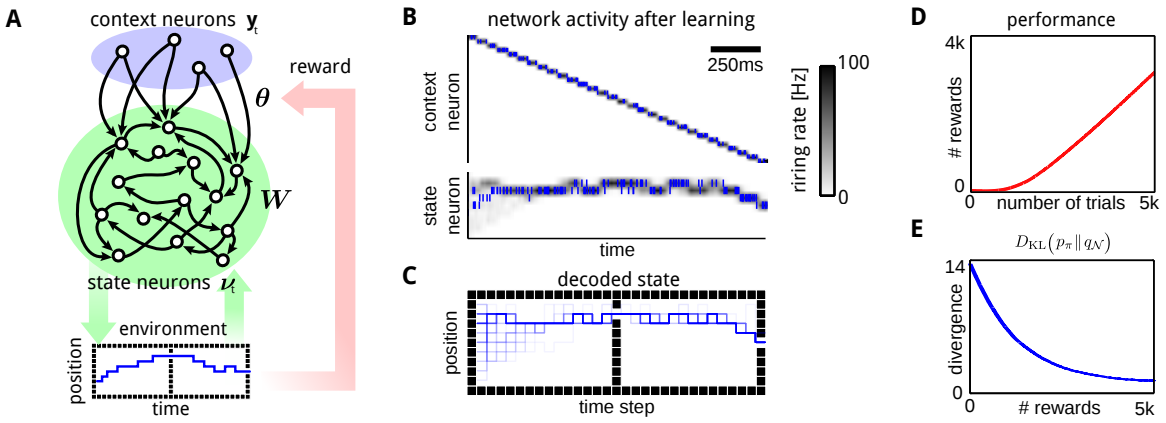

Figure 1. Illustration of the model for finite horizon planning. **A:** The neural network architecture considered here for solving the probabilistic planning problem. A recurrent layer of state neurons (green) that control the behavior of the agent receive feedforward input from context neurons (blue), the activity of which determine the desired goal. **B,C:** A simple planning problem that requires to pass through a passage at two specific points in time. Superposition of network activity averaged over 100 trial runs (B) and decoded network states (C) are shown. Blue dots in (B) show 1 example spike train. **D:** The accumulated number of rewards for the spiking network. **E:** The Kullback-Leibler divergence between the learned distribution and the true posterior. (C) and (D) show averages over 100 trial runs.

update rule.

We will compare *online* and *offline* updates of equation (7). In its stochastic *online* variant, the E-step is approximated by sampling a finite set of L samples to estimate the expectation,[24] or in the simplest case after a single sample ($L = 1$) as done in our experiments. We refer to this as the online approximation of equation (7). With *offline* updates, implemented as batch learning, the KL divergence $D_{KL}(p(\underline{\boldsymbol{v}}|r=1)||q(\underline{\boldsymbol{v}};\boldsymbol{\theta}))$ between the true posterior for planning in equation (1) and the model distribution converges to zero for $L \to \infty$ (assuming an exact encoding of the state and the transition model). This KL divergence establishes the relation between the inference problem for planning in equation (1) and the introduced problem of finding the network parameters that maximize the expected return.

A finite horizon planning task

To evaluate the spiking neural network model we consider a simple one dimensional planning problem, where the agent moves on a linear track and the activity of the state neurons population V directly determines its position. $K = 9$ state neurons encode nine discrete locations. A final reward is only received if the agent passes through two obstacles, one at time $T/2$ and one at time T (see Fig. 1C). Furthermore the agent is constrained not to jump to distant states within one time step. We model this constraint by the state transition model, i.e., $\mathscr{T}(\boldsymbol{v}_t = k \mid \boldsymbol{v}_{t-1} = i) = 1/3$, if $k-1 \leq i \leq k+1$ and (close to) zero otherwise (see the supplement for further details).

Due to the limitation on the state transitions, this problem requires planning ahead in order to avoid the obstacles successfully, i.e., to start moving to the passage before the obstacle actually appears. We show that the optimal planning policy can be learned using the reward modulated update rule in equation (7) in a network where the state neurons follow (soft) winner-take-all (WTA) dynamics. The probability $\rho_{t,k}$ of neuron k to spike at time t is given by $\rho_{t,k} = \exp(u_{t,k}) \,/\, \sum_{l=1}^{K} \exp(u_{t,l})$. Thus, in each

time step exactly one state neuron is active and encodes the current position of the agent.

The precise timing required to solve this task can be learned if the context neurons provide sufficient temporal structure. We study here the case of only one context neuron being active for one time-step, i.e., $y_{t,j} = 1$ for $j = t$ and $y_{t,j} = 0$ else. The weights θ_{kj} were adapted according to the online approximation of equation (7). Prior to learning, the agent performs a random walk according to the state transition model encoded in the weights w_{ki}, performing successful trials only occasionally. As learning proceeds, the activity of the context neurons shapes the behavior of the agent leading to nearly optimal performance. Figure 1D shows the accumulated reward throughout learning. After 5000 training iterations, the network generates rewarded trajectories in $97.80 \pm 4.64\%$ of the trials. We also evaluated a more detailed spiking version of the network model which produced similar results (success rate: $87.40 \pm 15.08\%$, see Fig. 1B,C in the supplement).

In addition to the online learning rule, we also evaluated the offline update rule. The network draws samples from a fixed distribution $q(\underline{\boldsymbol{\nu}}; \boldsymbol{\theta}_0 = 0)$ simulating random walks without any input (the initial state distribution $p(\boldsymbol{\nu}_0)$ was uniform). Offline updates are applied to the parameters $\boldsymbol{\theta}$ and Kullback-Leibler divergence converges towards zero with an increasing number of updates as shown in Fig. 1E.

Extension to the infinite horizon problem

Previously, we demonstrated how our network can model finite horizon planning tasks with terminal rewards (i.e., returns). Here we generalize to infinite horizon planning problems where rewards can be received at any point in time. The goal of the planning problem is to optimize the parameters $\boldsymbol{\theta}$ in the neural network so that it generates infinite trajectories that maximize the expected total discounted reward $\langle \sum_{t=0}^{\infty} \gamma^t \hat{r}_t \rangle$, where γ is the discount factor. We can reformulate this planning problem as probabilistic inference in an infinite mixture of Markov chains of finite lengths T.[1] The corresponding mixture distribution of trajectories is given by

$$q(r, \underline{\boldsymbol{\nu}}; \boldsymbol{\theta}) = \sum_{T=1}^{\infty} p(T)\, q(\underline{\boldsymbol{\nu}}_T; \boldsymbol{\theta})\, p(r|\underline{\boldsymbol{\nu}}_T) \tag{8}$$

where $p(T) = (1-\gamma)\gamma^T$ is the prior distribution over trajectory lengths, and $q(\underline{\boldsymbol{\nu}}_T; \boldsymbol{\theta})$ is the distribution over spike trains of length T according to the network dynamics in equation (4). The probability of getting a reward r at the end of the trajectory is given by $p(r = 1|\underline{\boldsymbol{\nu}}_T)$. Intuitively, in infinite horizon planning tasks the agent seeks a solution that balances getting to the goal as fast as possible (imposed by the prior) against the cost of large state jumps (imposed by the state transition model).

For the infinite horizon model we consider network dynamics where each state neuron has a sigmoid activation function, i.e., $\rho_{t,k} = \sigma(u_{t,k})$ with $\sigma(u) = 1/(1+e^{-u})$. Using this activation function we find that for learning the infinite planning task the parameters θ_{kj} should undergo a change in each time step t where the reward is present, according to

$$\Delta\theta_{t,kj} \;=\; \eta\, \hat{r}_t \sum_{\tau=1}^{t} \sum_{T=t-\tau}^{\infty} p(T)\, y_{\tau,j} \left(\nu_{\tau,k} - \rho_{\tau,k}\right) \;=\; \eta\, \hat{r}_t \sum_{\tau=1}^{t} \gamma^{t-\tau}\, y_{\tau,j} \left(\nu_{\tau,k} - \rho_{\tau,k}\right) . \tag{9}$$

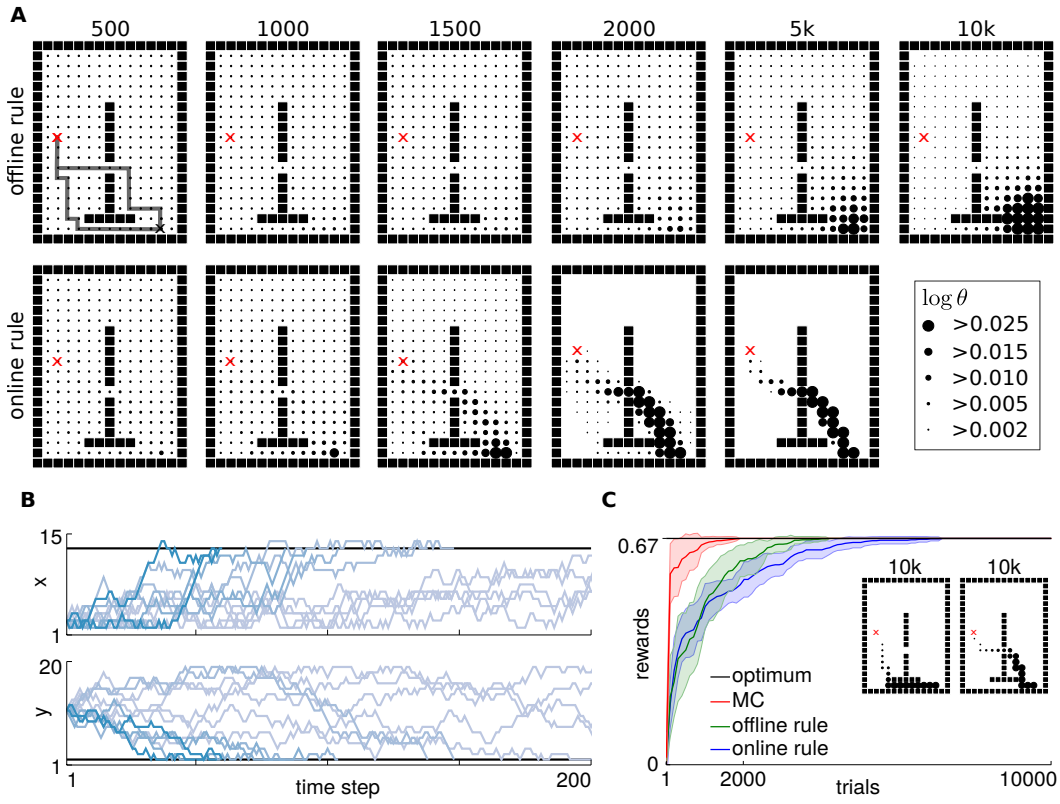

Figure 2. Illustration of the model for infinite horizon planning. **A:** The agent has to move from the red cross the black cross. The radii of the dots are proportional to the log of the $\boldsymbol{\theta}$ parameters. The results for the offline and the online learning rules are shown in the two rows, respectively. **B:** Illustration of 12 sampled trajectories after 10000 trials of offline learning. **C:** The mean of the received rewards over 20 experiments. We compare to Monte-Carlo policy evaluation (MC).

This synaptic weight update can be realized using an eligibility trace[?] $e_{t,kj}$ associated to each synapse with dynamics

$$e_{t,kj} = \gamma e_{t-1,kj} + y_{t,j}\left(v_{t,kj} - \rho_{t,kj}\right) \quad \text{and} \quad \theta_{t,kj} = \theta_{t-1,kj} + \eta\, r_t\, e_{t,kj}\,. \tag{10}$$

The eligibility trace is updated in each time step, whereas the weight updates are only applied for $\hat{r}_t > 0$. More details on the learning rule can be found in the supplement.

Note that the precise timing of attracting or repelling states cannot be modeled through explicit context neurons per time step as in the finite horizon model (since $t \to \infty$). Therefore, we consider stationary activity patterns of context neurons. This assumption implies that after convergence of the parameter updates an attractor cannot be visited twice.

An infinite horizon planning task

To test the infinite horizon model we consider a planning task, where the goal of the agent is to navigate from a given initial state to a target state in a grid maze with obstacles (dimensions $[15 \times 20]$). The network has 300 state neurons, one for each grid cell. The agent can perform only one-step moves, to the left, to the right, up or down, with equally probable transitions in each direction, and receives a reward $\hat{r}_t = 1$ only at the target state. The sampling process is either terminated if the target state

is reached, or if the time step exceeds the maximum number of allowed steps ($T = 300$). The discount factor was $\gamma = 0.98$.

With the offline learning rule the learned parameters $\boldsymbol{\theta}$ setup a gradient towards the target state, which covers *multiple* solution trajectories that lead to high total received rewards ($\boldsymbol{\theta}_0$ is chosen such that the agent starts at the initial state). This gradient is indicated by the radii of the dots in the first row in Fig. 2A. Figure 2B illustrates 12 example trajectories with weights obtained after 10000 trials of learning. Out of the shown 12 trajectories 9 reached the target state that is denoted by the black horizontal lines.

With the online learning rule, the learned parameters $\boldsymbol{\theta}$ specialize on one locally optimal path through the maze, which is illustrated in the second row of Fig. 2A. In the evaluated example, there are two locally optimal trajectories that are also the global optima. They are shown in the inset in Fig. 2C. For both, the offline and the online updates the average received reward converges to the maximum value (see Fig. 2C), where we compare to Monte-Carlo policy evaluation (MC).[20]

Results

A computational model for hippocampal sweeps

We show that the neural network reproduces the transient firing recorded in place cells in rats during planning phases. We compare our model predictions to recent results in,[9] where the authors analyzed the neural activity of 250 simultaneously recorded hippocampal neurons during phases of mental planning (using a 40-tetrode microdrive). In the experiments the animals received a reward, in an alternating manner, either at a known home location or at some (unknown) random location (one out of 36 locations arranged in a grid in a 2×2m maze). Here, we model only events that were observed while the animal was *resting* at some known current location and plans a route to the memorized home location (mental planning). An example event of a rat is illustrated in Fig. 3C, which illustrates the transient in the reconstructed position posterior probabilities. The resulting movement plan, i.e., the decoded and summed place cells' activity across time is shown in the lower panel in Fig. 3C.

In our network, the current location and the target location of the rat are modeled by $N = 20$ context neurons. The activity of which is denoted by $\mathbf{y}(t)$ and shown in Fig. 3A. The first ten context neurons encode the current location through a transient pattern. The remaining ten context neurons represent the desired target location. These neurons fire with a stationary Poisson process throughout the experiment. In this experiment we did not learn the weights of the context neurons. The weights were set proportional to the Euclidean distance of the preferred position of place cells to the initial state or the home location.

The network activity of 100 place cells (the preferred positions of which are aligned with a 10-by-10 grid) is solely driven by the context neurons and the recurrent weights. The recurrent synapses implement a Gaussian state transition model which prevents that the network directly draws samples close to the target state. In contrast to the ideal model used in the previous experiments multiple place cells can be active simultaneously, i.e., the recurrent weights encode a soft WTA circuit as in.[25]

Encoding continuous state variables

A simple encoding scheme is used to reconstruct the two dimensional state of the simulated rat $\boldsymbol{x}(t)$ from the place cells' activity, i.e., $\boldsymbol{x}(t) = 1 / |\boldsymbol{\nu}(t)| \sum_{l=1}^{L} (\nu_l(t)\, \boldsymbol{p}_l)$, where $|\boldsymbol{\nu}(t)| = \sum_{l=1}^{L} \nu_l(t)$ and $\boldsymbol{p}_l$ is the preferred position of place cell l.

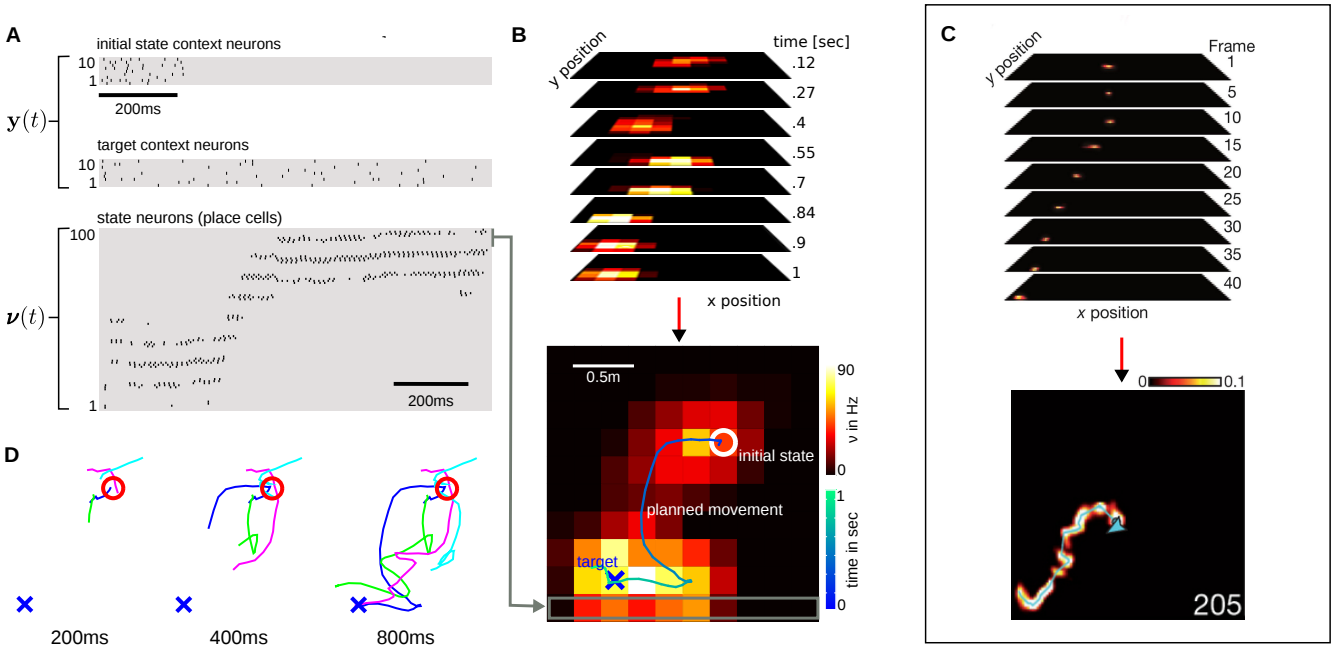

Figure 3. Planning and modeling transient firing in place cells with simulation results in (A-B,D). **C:** Biological data showing the position posterior probabilities in selected frames (the event duration was 205ms). Reprinted by permission from Macmillan Publishers Ltd: Nature 497, 74–79, © 2013. **A:** The first ten context neurons are only activated for the first 200ms to initialize the network at the desired initial state. The ten context neurons in the 2nd row in (A) implement a gradient towards the desired target state. The transient neural activity of 100 place cells is shown in the 3rd row in (A). The reconstructed movement trajectory is denoted by the line in (B). **D:** The result of four planning processes for an increasing planning horizon.

The generated transient activity of the place cells realize a path between the initial and the home location as shown in the 3rd row in Fig. 3A. The reconstructed movement plan is denoted by the line in Fig. 3B, where the integrated activity over the whole movement duration of each place cell is encoded by the color of the corresponding grid position. The simulated sequential firing shows a coherence to the transient firing in place cells in rats that is illustrated in Fig. 3C (see also the online material in a work of B. Pfeiffer and D. Foster[9]). A notable difference from the biological data is the resolution of the simulated activity in Fig. 3B. To visualize the corresponding spike events only 100 place cells were modeled. For a higher density of 900 uniformly distributed place cells we refer to additional results provided in the supplement.

Task adaptation through context neurons in a real robot

Typical robot planning tasks have to consider a large number of constraints that dynamically change and planning algorithms have to adapt to new solutions online. Here, we show that by activating context neurons multiple constraints can be modeled and used for task adaptation. As test platform we used a KUKA lightweight arm controlled in a two dimensional Cartesian space. The network generated movement plans in $2D$ and the complexity of the control problem itself is absorbed by a built-in Cartesian tracking controller. The two coordinates modeled, x and y span the transverse plane. The trajectory was executed using inverse kinematics to obtain a reference joint trajectory and inverse dynamics control to execute it.

We used $K = 225$ state neurons that receive excitatory input from context neurons modeling the initial state and the target state (as in the previous experiment), see Fig. 4A. Strong inhibitory input is used to model obstacles (the gray areas in Fig. 4B).

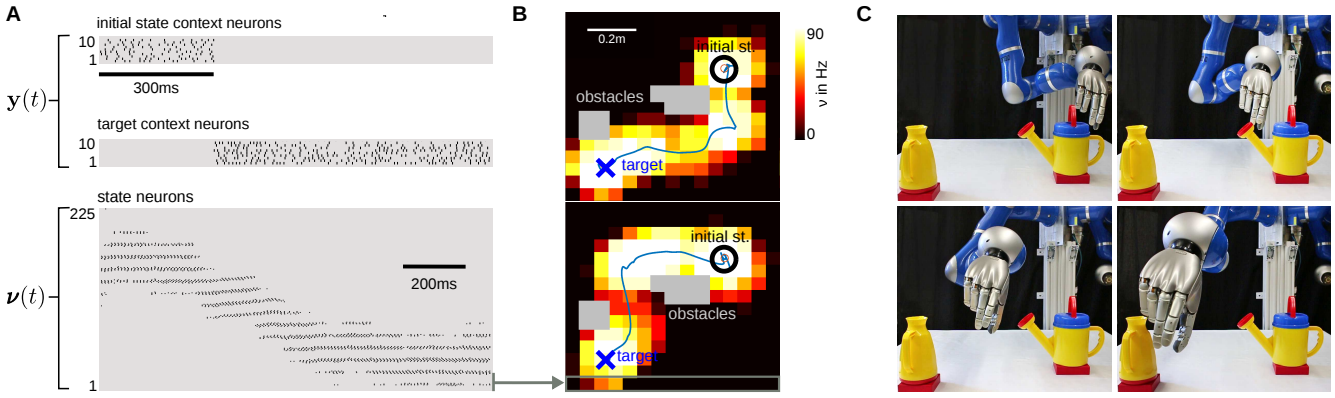

Figure 4. Planning with multiple constraints on a real robot. **A:** Generated spike train (top: context neurons, bottom: state neurons) after contrastive divergence learning of the transition model. **B:** Two sampled movement plans solving the obstacle avoidance task. **C:** Snapshots of the executed movement on the real robot.

The transition model was learned from demonstrated state transitions (through kinesthetic teaching) in contrastive divergence learning[21] (see the supplement for details).

After learning, the network generates goal-directed movement plans that can be used for obstacle avoidance. Sampling a movement plan of a duration of 1.3 seconds took on average 635 ± 8 milliseconds on a standard computer (the symbol $\pm$ denotes the standard deviation computed from 1000 trajectories). The minimum distance to the target was 2.71 ± 2.24 cm in an operation area of 70×70 cm.

An interesting property of the spiking model is that multiple solutions can be encoded in the same model. For the considered obstacle avoidance task, two solutions are shown in Fig. 4B and snapshots of the second solution are depicted in Fig. 4C.

Discussion

The brain efficiently processes and predicts sequential activity patterns in the context of planning tasks.[9–12] Understanding these processes and how information is possibly encoded in stochastic neurons are major goals of theoretical neuroscience. In this paper, we demonstrated how recurrent spiking neural networks can solve planning problems and provide a solid theory based on the framework of probabilistic inference. The model reproduces observed neural dynamics, predicts that contextual information is one of the key modulating factors and is a promising low-energy control strategy for mobile robot applications.

Theoretical contributions

Spiking neural networks are a reasonable neuroscience model as verified by data.[26–29] Their capabilities in solving planning tasks however have been underexplored to date. It was shown that spiking networks can encode and draw samples from arbitrary complex distributions,[30,31] models of temporal sequences can be learned[7,8] and Bayesian filtering was studied.[29,32–34]

Our model builds on these probabilistic sampling results and a solution to planning problems is suggested that approximates a stochastic process for planning through forward sampling from a parameterized model distribution. The parameters of the model distribution denote the synaptic weights of a population of afferent neurons for which local Hebbian update rules were

derived from the principle of probabilistic inference. The derivations include arbitrary differentiable activation functions and postsynaptic potential shapes for the neuron model (details are provided in the supplement).

Links to expectation maximization[22,24] and policy gradient methods[?,?,?] were established, where the resulting offline learning rules are similar to Monte Carlo policy evaluation[20] and the network parameters subject to these updates converge to the globally optimal solution for WTA network dynamics. The online learning rules resemble the online Monte Carlo policy iteration algorithm and converge to local lower bounds of the optimum.

The correctness of the neural planning method was validated in two toy tasks, a finite horizon planning task with a known optimal solution and an infinite horizon task, where the neural network achieved the same performance level as Monte-Carlo policy evaluation.[20]

Implications for neuroscience

Previous neural models that implement path planning have focused on attractor networks or potential fields[14,35–38] and their activity was related to hippocampal firing.[16,17] A deficit of these models is however that the path which was taken to reach a desired state cannot be modeled with attractors. To overcome this limitation a sequence of successive metastable states in attractor networks was proposed[18] but it is left unclear how these networks can be trained from rewards. We followed a different approach where attractors emerge through reinforcement of rewarding trajectories. As a result, different input neurons with its synaptic weights can model different routes to multiple attractors. Thus, the proposed model extends the modulation abilities of attractor networks and can be validated, e.g., in a study on contrasting planning of safe versus straight-line paths.

We demonstrated in simulation results that the proposed recurrent neural network can reproduce the dynamically changing firing rates observed during hippocampal sweeps.[9] The input modulated activity in our network hypothesizes that a cognitive map representation (the recurrently connected state neurons) receives contextual input from other brain regions. Potential sites for these contextual inputs are projections from the entorhinal and the prefrontal cortex.[39,40] It is worth mentioning that the network is not limited to model hippocampal sweeps. It may be used to model frequently observed dynamically changing firing rates from other brain regions.[41–44]

Embedded in the framework of probabilistic inference, the proposed network can be naturally extended in multiple ways, e.g., the place cells encoding the state transition model might be learned,[16] actions might be encoded additionally,[25] forward and backward replays[10,11] can be simulated, or multiple cognitive maps can be installed.[17] Furthermore, Poisson neurons were chosen for simplicity and the model generalizes to noisy integrate and fire neurons.[45]

Implications for robotics

State-of-the-art planning algorithms in robotics generate movement plans within seconds and scale to many degrees of freedom.[?,?,?] Spiking neural networks could not compete with these methods due to the encoding of continuous variables, e.g., in our robot experiment, we used population codes[?,?] to encode a two-dimensional continuous state variable and more than a few hundred neurons could not be simulated without risking to run out of memory in a standard computer. A potential

solution that is currently under investigation are factorized population codes which scale but can not capture correlations (which are needed to avoid obstacles). Another promising alternative are neuromorphic chips which were already used to learn 62-dimensional joint angle sequences of human jump motions in recurrent networks.[?] In addition, related spiking network models were proposed which also build on winner-take-all circuits and local plasticity rules.[?,?,?] Therefore, it is reasonable to assume that the presented theory provides the basis for future neural controller implementations in neuromorphic hardware for robot control.

In contrast to previous work on spiking neurons in a reinforcement learning framework,[25] we followed here a model-based approach where the recurrent dynamics of the network can be reused to learn multiple related tasks with different sets of weights from the context neurons (e.g., representing different goal positions or obstacles). The state transition model does not need to be re-learned when switching between environments.

Furthermore, our model has the advantage that multi-modal solutions to planning tasks can be learned. This feature was exploited in an obstacle avoidance task in a real robot, where the network randomly sampled one out of two paths. This ability to encode non-linear mappings is in particular beneficial for learning forward and inverse kinematic models in robotics.

References

1. Toussaint, M. & Storkey, A. Probabilistic inference for solving discrete and continuous state markov decision processes. In *proceedings of the ICML*, 945–952 (ACM, 2006).

2. Kappen, H. J., Gómez, V. & Opper, M. Optimal control as a graphical model inference problem. *Machine Learning* **87**, 159–182 (2012).

3. Botvinick, M. & Toussaint, M. Planning as inference. *Trends in Cognitive Sciences* **16**, 485 – 488 (2012).

4. Solway, A. & Botvinick, M. M. Goal-directed decision making as probabilistic inference: A computational framework and potential neural correlates. *Psychological Review* **119**, 120–154 (2012).

5. Penny, W. D., Zeidman, P. & Burgess, N. Forward and backward inference in spatial cognition. *PLoS Comput Biol* **9** (2013).

6. Pezzulo, G., van der Meer, M. A., Lansink, C. S. & Pennartz, C. M. Internally generated sequences in learning and executing goal-directed behavior. *Trends in cognitive sciences* **18**, 647–657 (2014).

7. Brea, J., Senn, W. & Pfister, J.-P. Sequence learning with hidden units in spiking neural networks. In *Advances in Neural Information Processing Systems*, 1422–1430 (2011).

8. Kappel, D., Nessler, B. & Maass, W. STDP installs in winner-take-all circuits an online approximation to hidden Markov model learning. *PLoS Comp. Biol.* **10**, e1003511 (2014).

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

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

11. Johnson, A. & Redish, A. D. Neural ensembles in ca3 transiently encode paths forward of the animal at a decision point. *The Journal of neuroscience* **27**, 12176–12189 (2007).

12. Carr, M. F., Jadhav, S. P. & Frank, L. M. Hippocampal replay in the awake state: a potential substrate for memory consolidation and retrieval. *Nature Neuroscience* **14**, 147–153 (2011).

13. Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences* **79**, 2554–2558 (1982).

14. Samsonovich, A. & McNaughton, B. L. Path integration and cognitive mapping in a continuous attractor neural network model. *The Journal of neuroscience* **17**, 5900–5920 (1997).

15. McNaughton, B. L., Battaglia, F. P., Jensen, O., Moser, E. I. & Moser, M.-B. Path integration and the neural basis of the'cognitive map'. *Nature Reviews Neuroscience* **7**, 663–678 (2006).

16. Erdem, U. M. & Hasselmo, M. A goal-directed spatial navigation model using forward trajectory planning based on grid cells. *European Journal of Neuroscience* **35**, 916–931 (2012).

17. Azizi, A. H., Wiskott, L. & Cheng, S. A computational model for preplay in the hippocampus. *Frontiers in computational neuroscience* **7** (2013).

18. Rabinovich, M., Huerta, R. & Laurent, G. Transient dynamics for neural processing. *Science* **321**, 48–50 (2008).

19. Rawlik, K., Toussaint, M. & Vijayakumar, S. On stochastic optimal control and reinforcement learning by approximate inference (extended abstract). In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI'13, 3052–3056 (AAAI Press, 2013).

20. Sutton, R. S. & Barto, A. G. *Reinforcement learning: An introduction*, vol. 28 (MIT press, 1998).

21. Hinton, G. Training products of experts by minimizing contrastive divergence. *Neural computation* **14**, 1771–1800 (2002).

22. Dempster, A. P., Laird, N. M. & Rubin, D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* **39**, 1–38 (1977).

23. Todorov, E. Linearly-solvable markov decision problems. In *NIPS*, 1369–1376 (2006).

24. Wei, G. C. G. & Tanner, M. A. A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association* **85**, 699–704 (1990). http://www.tandfonline.com/doi/pdf/10.1080/01621459.1990.10474930.

25. Frémaux, N., Sprekeler, H. & Gerstner, W. Reinforcement learning using a continuous time actor-critic framework with spiking neurons. *PLoS Comp. Biol.* **9**, e1003024 (2013).

26. Gerstner, W. & Kistler, W. M. *Spiking neuron models: Single neurons, populations, plasticity* (Cambridge university press, 2002).

27. Izhikevich, E. M. *et al.* Simple model of spiking neurons. *IEEE Transactions on neural networks* **14**, 1569–1572 (2003).

28. Izhikevich, E. M. Which model to use for cortical spiking neurons? *IEEE transactions on neural networks* **15**, 1063–1070 (2004).

29. Deneve, S. Bayesian spiking neurons i: inference. *Neural computation* **20**, 91–117 (2008).

30. Buesing, L., Bill, J., Nessler, B. & Maass, W. Neural dynamics as sampling: A model for stochastic computation in recurrent networks of spiking neurons. *PLoS Comput Biol* **7**, e1002211 (2011).

31. Berkes, P., Orbán, G., Lengyel, M. & Fiser, J. Spontaneous cortical activity reveals hallmarks of an optimal internal model of the environment. *Science* **331**, 83–87 (2011).

32. Bobrowski, O., Meir, R. & Eldar, Y. C. Bayesian filtering in spiking neural networks: Noise, adaptation, and multisensory integration. *Neural computation* **21**, 1277–1320 (2009).

33. Boerlin, M. & Denève, S. Spike-based population coding and working memory. *PLoS computational biology* **7**, e1001080 (2011).

34. Legenstein, R. & Maass, W. Ensembles of spiking neurons with noise support optimal probabilistic inference in a dynamically changing environment. *PLoS computational biology* **10**, e1003859 (2014).

35. Glasius, R., Komoda, A. & Gielen, S. C. Neural network dynamics for path planning and obstacle avoidance. *Neural Networks* **8**, 125–133 (1995).

36. Miller, W. T., Werbos, P. J. & Sutton, R. S. *Neural networks for control* (MIT press, 1995).

37. Stringer, S., Rolls, E., Trappenberg, T. & De Araujo, I. Self-organizing continuous attractor networks and path integration: two-dimensional models of place cells. *Network: Computation in Neural Systems* **13**, 429–446 (2002).

38. Lebedev, D. V., Steil, J. J. & Ritter, H. J. The dynamic wave expansion neural network model for robot motion planning in time-varying environments. *Neural Networks* **18**, 267–285 (2005).

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

40. Redish, A. D. *Beyond the cognitive map: from place cells to episodic memory* (MIT Press Cambridge, MA, 1999).

41. Abeles, M. *et al.* Cortical activity flips among quasi-stationary states. *Proceedings of the National Academy of Sciences* **92**, 8616–8620 (1995).

42. Jones, L. M., Fontanini, A., Sadacca, B. F., Miller, P. & Katz, D. B. Natural stimuli evoke dynamic sequences of states in sensory cortical ensembles. *Proceedings of the National Academy of Sciences* **104**, 18772–18777 (2007).

43. Luczak, A., Barthó, P., Marguet, S. L., Buzsáki, G. & Harris, K. D. Sequential structure of neocortical spontaneous activity in vivo. *Proceedings of the National Academy of Sciences* **104**, 347–352 (2007).

44. Zhang, Q.-f. *et al.* Priming with real motion biases visual cortical response to bistable apparent motion. *Proceedings of the National Academy of Sciences* **109**, 20691–20696 (2012).

45. Rao, R. P. Hierarchical bayesian inference in networks of spiking neurons. In Saul, L., Weiss, Y. & Bottou, L. (eds.) *Advances in Neural Information Processing Systems 17*, 1113–1120 (MIT Press, 2005).

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements No. 248311 (AMARSI) and No. 600716 (CoDyCo). The authors would like to thank Wolfgang Maass, Matthew Botvinick and Tucker Hermans for comments that greatly improved the manuscript. We would also like to thank Brad Pfeiffer and David Foster for the permission to print parts of their inspiring results.[9]

Author contributions

E.R., D.K. and D.P. conceived the story, designed the model and derived the update rules. E.R. and D.K. performed the model simulations. D.T. implemented contrastive divergence learning. D.T. and E.R. performed the real robot experiment. E.R., D.K., D.P., and J.P. wrote the manuscript.

Additional information

Competing financial interests: The authors declare no competing financial interests.