ARTICLE **Communicated by Ron Meir**

A Spiking Neural Network Model of an Actor-Critic Learning Agent

Wiebke Potjans
wiebke_potjans@brain.riken.jp
Abigail Morrison
abigail@brain.riken.jp
Computational Neuroscience Group, RIKEN Brain Science Institute, Wako City, Saitama 351-0198, Japan

Markus Diesmann
diesmann@brain.riken.jp
Computational Neuroscience Group, RIKEN Brain Science Institute, Wako City, Saitama 351-0198, Japan, and Bernstein Center for Computational Neuroscience, Albert Ludwigs University, 79104 Freiburg, Germany

The ability to adapt behavior to maximize reward as a result of interactions with the environment is crucial for the survival of any higher organism. In the framework of reinforcement learning, temporal-difference learning algorithms provide an effective strategy for such goal-directed adaptation, but it is unclear to what extent these algorithms are compatible with neural computation. In this article, we present a spiking neural network model that implements actor-critic temporal-difference learning by combining local plasticity rules with a global reward signal. The network is capable of solving a nontrivial gridworld task with sparse rewards. We derive a quantitative mapping of plasticity parameters and synaptic weights to the corresponding variables in the standard algorithmic formulation and demonstrate that the network learns with a similar speed to its discrete time counterpart and attains the same equilibrium performance.

1 Introduction

A general problem in neuroscience is how system-level learning is realized on the cellular level. It is a basic assumption that synaptic plasticity underlies learning, but in higher organisms, the gap between the description level of synaptic efficacy and that of animal behavior remains largely unbridged (but see Antonov, Antonova, Kandel, & Hawkins, 2003). One model of learning that attracts considerable interest is reinforcement learning (RL). Unlike supervised learning, which requires an explicit teacher signal, RL requires only an evaluative feedback, which can be noisy or sparse. An RL

agent attempts to maximize the amount of reward it receives from the environment. This is more in keeping with the majority of learning situations an animal encounters in the natural world, where direct supervision is largely absent but rewards and penalties such as hunger, satiety, pain, and pleasure abound. There is considerable evidence from behavioral and neurophysiological studies that animals perform some kind of reinforcement learning. In particular, many results can be accounted for by a variant of RL known as temporal-difference (TD) learning, in which reward estimates at successive times are compared. By comparing reward estimates rather than waiting for a reward from the environment, a TD learning system is effective at solving tasks where the reward is sparse. TD learning has been used to interpret the activity of dopamine neurons in reward learning in nonhuman primates (Schultz, Dayan, & Montague, 1997; Schultz, 2002); in humans, dopamine-dependent prediction error has been shown to guide decisions (Pessiglione, Seymour, Flandin, Dolan, & Frith, 2006). Moreover, in human functional neuroimaging, a high correlation was found between the blood-oxygenation-level-dependent (BOLD) activity and the error signal required by a TD learning algorithm (O'Doherty, Dayan, Friston, Critchley, & Dolan, 2003; Seymour et al., 2004). It has also been used to model bee foraging in uncertain environments (Montague, Dayan, Person, & Sejnowski, 1995; Niv, Joel, Meilijson, & Ruppin, 2002) and human decision making (Montague, Dayan, & Sejnowski, 1996). In the field of machine learning, TD learning has been successfully applied to a variety of applications such as backgammon (Tesauro, 1994), balancing a pole on a cart (Barto, Sutton, & Anderson, 1983), and robotic control (Morimoto & Doya, 2001). For a comprehensive introduction to TD learning and reinforcement learning in general, see Sutton and Barto, 1998, and Bertsekas and Tsitsiklis, 1996.

However, it remains unclear how TD learning could be implemented in the brain. The classical formulation of TD learning is defined in discrete steps. In each step, the agent chooses an action, which is communicated to the environment. The environment responds with a signal informing the agent of its new state. It is therefore a simple operation to compare the reward estimate for one state to the reward estimate for the state occupied in the previous step. Neurons interact by means of pulses in continuous times, so it is not immediately clear how to define the concept of "previous step," or which observable quantity should be compared. As a result of this incompatibility, previous modeling attempts have generally focused on either non-TD reinforcement learning strategies (Seung, 2003; Xie & Seung, 2004; Florian, 2007; Baras & Meir, 2007) or nonspiking models (Suri & Schultz, 1999, 2001; Foster, Morris, & Dayan, 2000). The former category has demonstrated that spiking neuronal networks can solve simple tasks such as the XOR problem, where a reward or penalty is awarded after every decision of the network. The ability of such networks to solve more difficult tasks with sparse rewards has not been shown. In the latter category, Foster et al. (2000) have shown that a nonspiking neuronal implementation of TD

learning is capable of solving a complex task, but it is not a priori clear how the discrete synaptic weight updates employed in the model could be realizable by a synapse operating in continuous time. A third category of studies investigates aspects of TD learning in the context of spiking activity; however, they focus only on the prediction problem of when rewards can be expected and do not address the control problem of what actions to take (Rao & Sejnowski, 2001) or only simple control problems where a reward or penalty is awarded after every decision (Izhikevich, 2007; Farries & Fairhall, 2007).

We present for the first time a spiking neuronal network model that concurrently solves the prediction and control problems by implementing a complete actor-critic TD learning agent (see section 2). The synaptic plasticity underlying the learning process relies on biologically plausible measures of pre- and postsynaptic activity and a global reward signal. We further show the equivalence between the synaptic dynamics and the discrete time formulation of a TD algorithm (reviewed in section 1.1), yielding a quantitative mapping between the parameters of the discrete time formulation and those of the neuronal formulation (see section 3). The neuronal network can solve a nontrivial gridworld task with sparse rewards almost as fast as the equivalent discrete time algorithmic implementation and with the same equilibrium performance (see section 4.1). We show that the derived mapping has a high degree of accuracy (see section 4.2). Performance and mapping remain robust for a wide range of parameters. The mapping accuracy decreases if the agent occupies each state for only very short periods, but the learning behavior of the network remains robust until a limiting time span is reached (see section 4.3). The synaptic update rules investigated here represent only one possible implementation of TD learning; we discuss alternative implementations in section 5.

Preliminary work has been presented in abstract form (Potjans, Morrison, & Diesmann, 2007a, 2007b).

1.1 Actor-Critic Temporal-Difference Learning. The goal of a reinforcement learning agent is to maximize the amount of reward it receives from the environment over time. When the agent receives a reward (or penalty), a major problem is how to distribute the reinforcement among the decisions that led to it; this is known as the temporal credit assignment problem. Temporal-difference learning represents a particularly effective method of solving this problem. Here, we focus on a variant of TD learning known as the actor-critic architecture (Witten, 1977; Barto et al., 1983). A thorough introduction to TD learning and the actor-critic algorithm presented here can be found in Sutton and Barto, 1998. The two modules of the actor-critic architecture (see Figure 1) are so called because the actor selects which actions to take in a given state, and the critic evaluates the consequences of the chosen action. In each discrete time step, the environment transmits its state to the agent. The actor selects an action by making

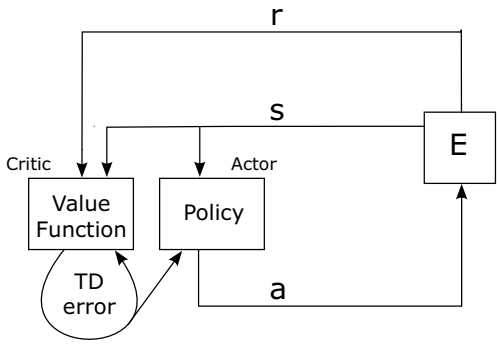

Figure 1: Actor-critic architecture. In each state, the actor selects an action (a) on the basis of its policy. As a result of this action, a new state is entered. The environment (E) transmits the new state information (s) to the actor and the critic and transmits the reward (r) associated with the new state to the critic. The critic evaluates whether the new state is better or worse than expected on the basis of its value function and the reward and emits an error signal (TD error) that expresses this disparity. The TD error is then used to update both the value function and the policy. (Figure adapted from Sutton & Barto, 1998.)

use of a policy $\pi(s,a)$, which gives the probability of selecting action a in state s. As a result of the action, in the next time step, the environment will transmit a new state to the agent and also any reward associated with the new state. Additionally, in each time step, the critic evaluates to what extent the current state is an improvement over the previous state by making use of a value function, $V^\pi(s)$. The value function can be understood as the expected discounted summed future reward obtained when starting from a given state s at time step i and following policy π:

$$\begin{aligned} V^\pi(s) &= E\left\{r_{i+1} + \gamma r_{i+2} + \gamma^2 r_{i+3} + \ldots \mid s_i = s\right\} \\ &= E\left\{r_{i+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{i+k+2} \mid s_i = s\right\} \\ &= E\left\{r_{i+1} + \gamma V^\pi(s_{i+1}) \mid s_i = s\right\}, \end{aligned} \tag{1.1}$$

where $E\{\cdot\}$ denotes an expectation value, r_{i+k} is the reward delivered at the state visited k time steps later, and γ is a discounting factor between 0 and 1. During the learning process, only estimates $V(s)$ for the actual value function $V^\pi(s)$ are available. Successive estimates of the value function will generally not fulfill the self-consistency relationship given in equation 1.1. The discrepancy is termed the TD error:

$$\delta_i = r_{i+1} + \gamma V(s_{i+1}) - V(s_i). \tag{1.2}$$

The sign and magnitude of δ_i contain information on how the estimate for $V^\pi(s)$ must be adjusted with respect to self-consistency. If the TD error is positive, this means that the new state s_{i+1} is better than expected, so $V(s_i)$ needs to be increased. Conversely, if the TD error is negative, the new state is worse than expected, so $V(s_i)$ should be decreased. The magnitude of δ_i gives an indication of how far adrift the current estimate of $V^\pi(s)$ is. This results in an update rule for $V(s_i)$:

$$V(s_i) \leftarrow V(s_i) + \alpha\delta_i, \tag{1.3}$$

where α is a small positive step-size parameter. This is known as the TD(0) algorithm (Sutton, 1988), and it converges for a given policy with probability 1 for sufficiently small α (Dayan, 1992; Dayan & Sejnowski, 1994). This solves the prediction problem of determining the value function V^π for a particular policy π. To solve the control problem (i.e., determine an optimal policy for the task), the actor also makes use of the information encoded in the TD error signal to adjust the policy accordingly. A positive TD error indicates that the selected action a resulted in a better state than expected; therefore, the probability of selecting this action the next time the agent is in state s should be increased. Likewise, a negative TD error implies that the probability of selecting this action the next time the agent is in state s should be reduced. For example, if the probability of selecting an action is given by the Gibbs softmax method,

$$\pi(s,a) = \frac{e^{p(s,a)}}{\Sigma_b e^{p(s,b)}}, \tag{1.4}$$

where $p(s,a)$ is a measure of the preference for action a in state s, then π can be updated toward the optimal policy by means of the update

$$p(s_i,a_i) \leftarrow p(s_i,a_i) + \beta\delta_i, \tag{1.5}$$

where β is another small step-size parameter. In this way, an optimal policy and associated value function can be learned incrementally, purely by the agent's own exploration of the environment.

The simplest way to implement the value function is by using a look-up table, with one entry per state. The memory required for this structure increases linearly with the number of states. Similarly, in the case of discrete actions, the policy can be represented by a look-up table in which each state has one entry for each action. Consequently the amount of memory required for this structure increases linearly with the product of the number of states and the number of possible actions. Moreover, as each state has to be visited many times in order to obtain a good approximation of the value function, increasing the number of states generally results in slower learning. As a

result, the algorithm presented above is well suited for tasks with small to moderate numbers of states and actions. To address problems with large numbers of states and actions, some kind of generalization is required. One approach is to interpret the actor-critic algorithm as a stochastic gradient algorithm where the critic implements TD learning with an approximation architecture determining the update of a parameterized policy (Konda & Tsitsiklis, 2003), for which convergence criteria have been identified. As our aim in this article is to show proof of principle for spiking neuronal networks implementing TD learning, in the following we restrict ourselves to the simple tabular algorithm outlined above.

2 Spiking Neuronal Network Model

The structure of the network—the neuronal modules and the synapses that encode value function and policy—is presented in section 2.1. For clarity, the synaptic dynamics underlying the convergence of the value function and policy to optimal values are presented separately (section 2.2). All simulation details are given in appendix A.

2.1 Network Architecture. The network structure shown in Figure 2 is inspired by the actor-critic architecture and is similar to the nonspiking network investigated by Foster et al. (2000). The agent consists of three modules of current-based integrate-and-fire neurons: the actor module, the critic module, and the states module. The agent interacts with the environment, which for the purposes of this work is implemented purely algorithmically. The environment activates the representation of a state by providing the corresponding neurons with a DC stimulus, causing them to fire with a rate of 42.63 Hz; their firing rate in the inactivated condition is 0.01 Hz. The environment simultaneously inhibits the actor module for a short time (action suppression period) to allow the activity of the state neurons to build up. The state neurons project to the actor module and the critic module. The stronger the synaptic weights between the activated state and a given actor neuron, the greater the probability that it will fire first. Whichever actor neuron fires first in response to the activation of a state is interpreted by the environment as the chosen action (for a review of first-spike coding, see VanRullen, Guyonneau, Thorpe, 2005). Thus, the synaptic weights between the state neurons and the actor neurons encode the policy of the agent. As a result of the chosen action, the environment deactivates the previous state and activates a new state as determined by the environmental model. If the new state is associated with a reward, the environment emits a constant reward signal R until the agent leaves the state again. The synaptic weights between the state neurons and the critic neurons encode the value function of the agent. Thus, if the agent moves from a state associated with a low value to a state associated with a high value, the rate of the critic neurons

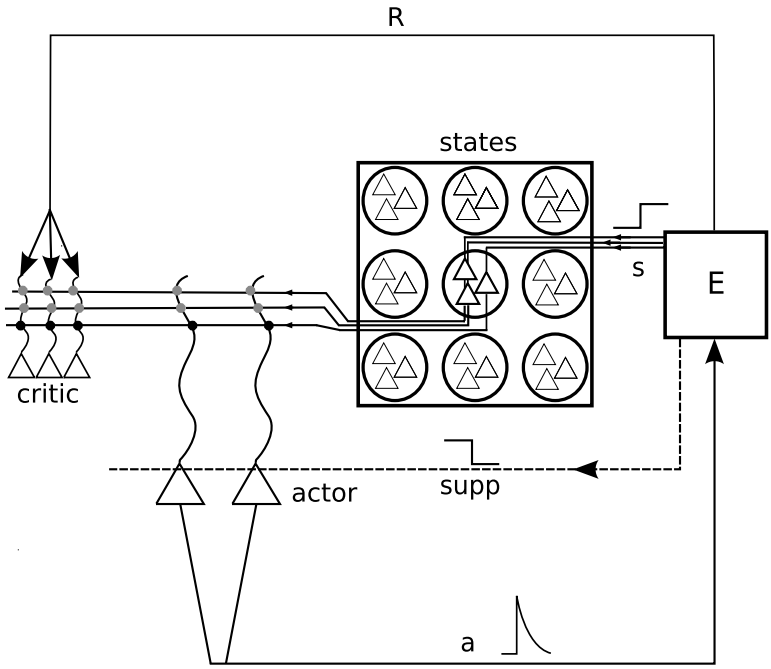

Figure 2: Neuronal implementation of the actor-critic architecture shown in Figure 1. The states (circles; 9 shown) are represented by pools of 40 neurons (triangles; 3 shown per state), the critic by one group of 20 neurons (3 shown), and the actor by 1 neuron for each possible action (2 shown). The state signal (s) consists of a positive DC stimulus from the environment (E) to the appropriate group of state neurons. The action signal (a) is defined as the first spike emitted by one of the actor neurons after entering a new state. The real-valued reward signal (R) has a modulatory effect on the state-critic synapses (see section 2.2.1). The action suppression signal (supp) consists of a negative DC stimulus to the actor neurons.

increases. Conversely, a move from a high-value to a low-value state results in a decrease in the critic firing rate.

2.2 Plasticity.

2.2.1 State-Critic Plasticity. In this section, we formulate an update rule for the synaptic weights between state and critic neurons that encode the value function of the neuronal agent, that is, a biologically plausible realization of the value function update of the TD(0) algorithm, equation 1.3. To obtain such a rule, we must consider three important properties of the value

function update. First, not all values are updated at the same time; when the agent moves from state s_i to state s_{i+1}, only $V(s_i)$ is updated. Second, the update depends on the values of the current and the successive states, $V(s_i)$ and $V(s_{i+1})$, respectively. Third, the update depends on an external reward r associated with state s_{i+1}. In the neuronal framework, we can interpret these properties as follows:

1. State-critic synapses are negligibly plastic except for a short time period when the agent has just left the corresponding state; the plasticity interval is smaller than the time period the agent typically stays in one state.
2. State-critic synapses are sensitive to a characteristic dynamic response of the critic neurons, which encodes the change in stimulus.
3. State-critic synapses are sensitive to a global signal representing the reward.

In addition, the synaptic update rule should be biologically plausible, in that it is formulated continuously, does not require a clock signal, and relies largely on local information. In the following, we present and test a candidate synaptic update rule; however, the constraints do not entail a unique specification, and so the rule presented here should be considered as only one example of a class of suitable rules. Further examples are discussed in section 5.

In order for a synapse to detect when the agent has left the state with which the synapse is associated, a locally available signal encoding this information must be identified. One such signal is the presynaptic rate: this drops from a high to low value when the agent leaves a state as the environment stops stimulating the state neurons associated with that state (see section 2.1). Although the rate of the presynaptic neuron is not a directly measurable quantity from the point of view of the synapse, individual spikes of the presynaptic neuron can be considered to be directly measurable. We can therefore approximate the rate for a particular realization of spikes times by an activity trace, defined as

$$\dot{\Lambda}_j^{\mathrm{s}}(t) = -\frac{1}{\tau_{\mathrm{s}}}\left(\Lambda_j^{\mathrm{s}} - \sum_{t_j^n < t}\delta\left(t - t_j^n\right)\right), \tag{2.1}$$

where t_j^n is the nth firing time of the presynaptic neuron j and τ_{s} is the time constant of the activity trace. This is a low-pass filtered version of the presynaptic spike train. For a review of plasticity rules implemented by such variables, see Morrison, Diesmann, and Gerstner, 2008. We propose that the synapse can be in three activity-dependent states: high activity, low activity, and plastic (see Figure 3A). The transition from one activity-dependent state to another occurs when its activity trace passes the corresponding

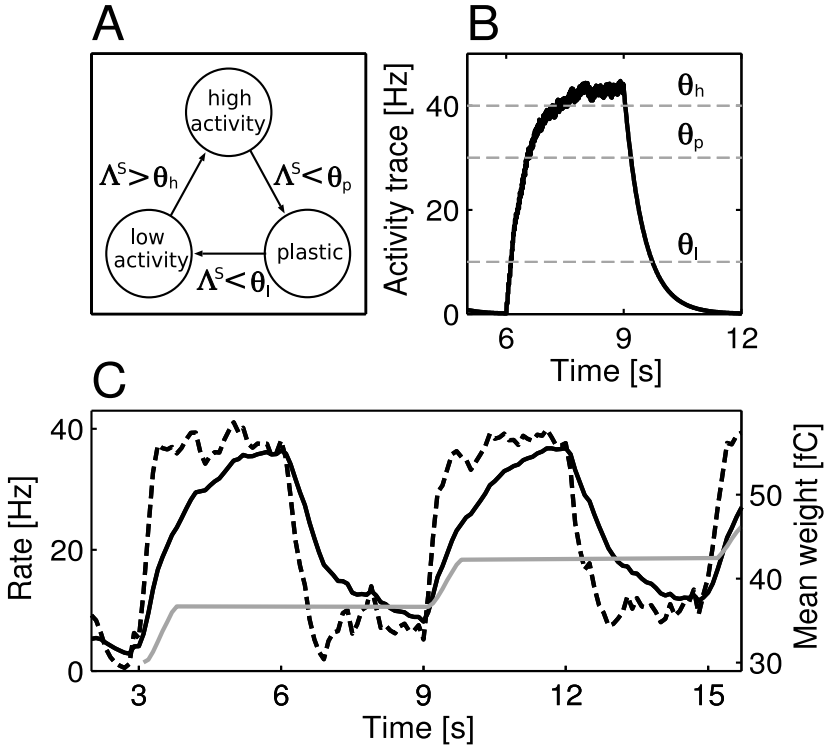

Figure 3: Plasticity of the state-critic synapses. (A) A synapse can adopt three states: low activity, high activity, and plastic. The transitions are controlled by thresholds (θ_h, θ_p, and θ_l) on the presynaptic activity trace Λ^s. (B) A presynaptic activity trace Λ^s as a function of time. At 6 s the presynaptic neuron is stimulated to produce a firing rate of 42.6 Hz; at 9 s the stimulus is turned off. The gray dashed lines show the values of θ_h, θ_p, and θ_l, which trigger the transition conditions depicted in *A*. (C) Postsynaptic activity traces and resultant state-critic weight development as functions of time. The state in question is represented by 40 presynaptic neurons, which project to a critic module of 20 neurons (see Figure 2). The agent moves out of the state represented by the presynaptic neurons into a higher-valued state at 3s, 9s, and 15s. The synapses become plastic when the agent leaves the state (see *B*), and the change in weight depends on the difference between two postsynaptic activity traces. The rapid postsynaptic activity trace (dashed black curve, averaged over all postsynaptic neurons) adapts more quickly to the changes in rate than the laggard postsynaptic activity trace (solid black curve, averaged over all postsynaptic neurons). The weight of the state-critic synapses (gray curve, averaged over 800 synapses) consequently increases each time the agent leaves the state and is otherwise constant.

threshold. A realization of an activity trace for a state-critic synapse, together with the three thresholds, is shown in Figure 3B. In the following, let us assume that the state neuron j is part of the group of neurons representing state x. Before the agent enters x, the presynaptic firing rate is low, and so the synapse is in the low-activity state. When the agent enters x, all neurons representing that state are stimulated, causing an increase in firing rate and thus an increase in the activity trace Λ_j^{s} for all state-critic synapses of neuron j. When Λ_j^{s} exceeds a threshold θ_{h}, the synapses enter the high-activity state. When the agent leaves x and enters a different state, the neurons representing x are no longer stimulated, so their firing rate decreases. Λ_j^{s} decreases exponentially below a threshold θ_{p}, and the synapses enter the plastic state. When the activity decreases further below a threshold θ_{l}, the synapses reenter the low-activity state. In the low- and high-activity states, the synapse is static; the transitions into and out of the plastic state define an interval of plasticity triggered by the agent leaving x, and thus fulfill the first property of the value function update discussed above. Note that only the state-critic synapses associated with x will be plastic when the agent leaves x, all other state-critic synapses will be in the low- or high-activity states. If the action chosen does not result in a new state, the state-critic synapses associated with x remain in the high-activity state, and no learning takes place. This is a deviation from the traditional discrete time TD(0) algorithm, in which the value function is updated every time step independent of whether the action chosen results in a new state. However, for tasks with a moderate number of states and sparse rewards, this deviation is unlikely to result in a worse performance. For problems where the reward is sparse, a good choice for the discount factor γ is close to 1, as it allows the agent to be farsighted. According to equations 1.1 and 1.2, the value function update when an action is chosen that does not lead to a new state is proportional to $(\gamma - 1)V(s_i)$ that is, it is negligibly small. We show in section 4.1 that this deviation does not lead to a worse performance on our test task.

The second property of the value function update in TD(0) learning requires a comparison of the values of two successive states. In order to fulfill this property in the neuronal continuous time formulation, the synapse needs information that encodes the values of the previous and the current states during its plasticity interval. As the value of a state is encoded by the strengths of the corresponding state-critic synapses, an appropriate locally available signal expressing the value of a state is the rate of the critic neurons: the greater the strength of the synapses, the higher the firing rate of the critic neurons. Assuming that the postsynaptic spikes are directly measurable by the synapse due to backpropagation or other mechanisms, the postsynaptic rate can be approximated by a low-pass filter of the spike train, as in equation 2.1.

To provide the synapse with information about two successive states at the same time, we posit that the spikes of a postsynaptic neuron k contribute

to two postsynaptic activity traces at the synapse: a rapid (or fast) trace Λ_k^{r} and a laggard (or slow) trace Λ_k^{l}, with unequal time constants $\tau_{\mathrm{r}} < \tau_{\mathrm{l}}$. As the laggard activity trace contains information about the previous state for longer than the rapid activity trace, a comparison of the two traces reveals whether the strength of the stimulus has recently increased or decreased. This is depicted for an example in Figure 3C. Here, the agent moves from state x to state y at 3 s, 9 s, and 15 s and from y to x at 6 s and 12 s. For the purposes of illustration, the network is initialized such that the estimate of $V(y)$ is greater than the estimate of $V(x)$: the state-critic synapses associated with y are stronger than the state-critic synapses associated with x. Consequently, the firing rate of the critic neurons increases when the agent moves from x to y and decreases when the agent moves from y to x. As the rapid activity trace adapts to changes in the postsynaptic activity more quickly than the laggard activity trace, Λ^{r} is greater than Λ^{l} during the plasticity interval of the state-critic synapses of x (see Figure 3B). Conversely, during the plasticity interval of the state-critic synapses of y, Λ^{r} is less than Λ^{l}. Biophysical candidates for all three activity traces are discussed by Pfister and Gerstner (2006) in the context of spike-timing dependent plasticity (STDP). We can therefore make the following ansatz for a continuous formulation of the value function update in analogy to the discrete time algorithm 1.3:

$$\dot{w}_{kj}(t) = \begin{cases} R + A\left(\tilde{\gamma}\,\Lambda_k^{\mathrm{r}}(t) - \Lambda_k^{\mathrm{l}}(t)\right) + C & \text{if plastic} \\ 0 & \text{otherwise} \end{cases}, \tag{2.2}$$

where R is a real-valued signal representing the reward associated with the successive state (see section 2.1), A and $\tilde{\gamma}$ are coefficients, and C is a constant; the plasticity condition is determined by the presynaptic activity trace Λ_j^{s} (see equation 2.1). Note that all terms rely on information local to the synapse with the exception of the global reward signal R, which is unspecific (it is the same for all synapses). A possible biological implementation is the release of a neurotransmitter such as dopamine, which has been shown to act as a third factor in the plasticity of corticostriatal synapses (see Reynolds & Wickens, 2002, for a review). In this study, we are not concerned with the reproduction of specific neurotransmitter dynamics, so we implement the reward signal in the simplest possible way: if the agent enters a state that is associated with a reward of R, a reward signal of R is effective until the agent leaves the state. The development of the mean weight of the state-critic synapses associated with the low-valued state x is shown for the example in Figure 3C. In this example, no state is rewarded ($R = 0$). As the difference between Λ_k^{r} and Λ_k^{l} is positive during the plasticity interval of synapses associated with state x, the synaptic weights between the neurons representing x and the critic increase. Conversely, when the agent moves from y to x, the negative difference between Λ_k^{r} and Λ_k^{l} during the

plasticity interval results in a decrease of strength for the synapses between the neurons representing y and the critic (data not shown). This is clearly analogous to the value function update defined in equation 1.3; the choice of parameters R, A, and $\tilde{\gamma}$ required for equivalence of the two formulations is demonstrated in section 3.1.

Our rule can be considered as belonging to the class of differential Hebbian learning rules, which depend on rate of change of neuronal activity, rather than on simultaneous pre- and postsynaptic activity, as is the case for classical Hebbian rules. Differential Hebbian rules were introduced by Klopf (1986) and Kosko (1986) in the form $\dot{w}_{kj} \propto \dot{\lambda}_j \dot{\lambda}_k$—the change of weight is proportional to the product of the changes in the pre- and postsynaptic rates. Such a rule can account for classical conditioning (Klopf, 1988). More recently, an alternative variant of a differential Hebbian rule, in which the change of weight is proportional to the correlation of the presynaptic signal and the derivative of the output signal, was successfully applied to RL control problems (Porr & Wörgötter, 2003, 2007; Wörgötter & Porr, 2005). In our rule, the change of weight is proportional to the rate of change of the postsynaptic rate, which is expressed by the difference between the two activity traces Λ_k^{r} and Λ_k^{l}, but not to the rate of change of the presynaptic rate. The presynaptic rate determines when the synapse is plastic, but not the magnitude of weight changes; its influence is multiplicative in the weak sense. Roberts (1999) demonstrated that under certain constraints, the experimentally observed phenomenon of STDP (Markram, Lübke, Frotscher, & Sakmann, 1997; Bi & Poo, 1998; Zhang, Tao, Holt, Harris, & Poo, 1998) implies differential Hebbian learning in the sense that the mean change in synaptic weight is proportional to the rate of change of the postsynaptic spike probability. It is outside the scope of this study to investigate to what extent our rule is compatible with STDP or other empirical plasticity data.

2.2.2 State-Actor Plasticity. The synaptic weights between state and actor neurons represent the policy in the neuronal actor-critic architecture. Here we present a biologically plausible realization of the plasticity rules given by equation 1.5. Analogous to the value function update, equation 1.3, in the discrete time algorithm, preference values are updated only when the corresponding action has just been chosen. In the case of an update of a preference value, the size of the update is proportional to the size of the update in the value function (compare equations 1.3 and 1.5). In the neuronal framework, these properties can be interpreted as follows:

1. State-actor synapses are negligibly plastic except for when the corresponding action has just been chosen.
2. State-actor synaptic updates are proportional to state-critic synaptic updates.

To enable the correct timing of weight updates, we assume that the plasticity is gated by the postsynaptic activity; a synapse between a state neuron

j and an actor neuron l is plastic, while the postsynaptic activity trace Λ_l^{a} with time constant τ_{a} implemented as in equation 2.1 exceeds a threshold value θ^{a}. For a sufficiently high value of θ^{a}, this condition is met for only a brief period after an actor neuron has fired—just after that particular action has been chosen (see section 2.1). To fulfill the proportionality criterion, we assume that while the synapse is plastic, it is subject to an axonal heterosynaptic spread of potentiation or depression from the synapses between the state neuron and the critic neurons:

$$\dot{w}_{lj}(t) = \begin{cases} \dfrac{B}{N_k} \displaystyle\sum_k \dot{w}_{kj}(t) & \text{if } \Lambda_l^{\mathrm{a}} > \theta^{\mathrm{a}} \\ 0 & \text{otherwise} \end{cases}, \tag{2.3}$$

where B is a scalar coefficient, N_k is the number of synapses each state neuron makes to critic neurons, and $\dot{w}_{kj}$ is the rate of change of the synaptic weight between neuron j and critic neuron k, as defined in equation 2.2. The weight is restricted to the range 30 fC to 90 fC. The continuous weight update is analog to the preference update in the discrete time algorithm, equation 1.5. It is local in the sense that it relies solely on postsynaptic spikes and information available to the axon of the presynaptic neuron.

Axonal or presynaptic spread of potentiation and depression has been observed in a variety of preparations. Bonhoeffer, Staiger, and Aertsen (1989) demonstrated that if long-term potentiation (LTP) is induced in a synapse in rat hippocampal slice cultures, enhancement can also be observed in other axonal synapses of the presynaptic neuron within a range of at least 150 μm. Presynaptic spread of potentiation was confirmed for rat visual cortex by Kossel, Bonhoeffer, and Boltz (1990), in acute hippocampal slices within a range of 300 μm by Schuman and Madison (1994), and in sparse cultured hippocampal networks by Tao, Zhang, Bi, and Poo (2000). Presynaptic spread of LTD was first shown in Xenopus nerve-muscle cultures by Cash, Zucker, and Poo (1996) and in sparse cultures of rat hippocampal cells by Fitzsimonds, Song, and Poo (1997). Here, we postulate a form of heterosynaptic plasticity that is also dependent on the participation of the postsynaptic neuron of the affected synapse. The evidence supporting this is ambiguous. In the case of LTD, voltage clamping such a neuron at -80 mV did not prevent the heterosynaptic depression of its synapses (Fitzsimonds et al., 1997), which suggests that no participation of the postsynaptic neuron is required. In the case of LTP, voltage clamping such a neuron -95 mV combined with whole-cell dialysis and Ca^{2+} chelation did inhibit the heterosynaptic potentiation of its synapses (Schuman & Madison, 1994), which suggests that participation of the postsynaptic neuron is indeed required. Similar forms of heterosynaptic plasticity have been used to model the development of delay-line topologies that can detect the delay between two stimuli (Kempter, Leibold, Wagner, & van Hemmen, 2001;

Leibold, Kempter, & van Hemmen, 2001; Leibold & van Hemmen, 2002). However, in these studies, the activity of the postsynaptic neuron of the affected synapse modulated the synaptic weight in an additive rather than multiplicative fashion.

3 Mapping Between Continuous and Discrete Time Update Rules

3.1 Value Function Mapping. The weight update for the synapses between state and critic neurons presented in equation 2.2 is heuristically motivated by the value function update of the discrete time TD(0) algorithm, equation 1.3. In this section, we show the equivalence of the continuous and the discrete time formulation of the value function update.

In the neuronal network, the value of each state is represented by the set of synapses connecting the pool of the corresponding state neurons with the pool of neurons representing the critic. As each synapse belonging to this set undergoes slightly different weight changes, we consider the dynamics of the mean weight of the state-critic synapses associated with a state s:

$$\dot{w}(s,t) = \frac{1}{N_j N_k} \sum_{k,j} \dot{w}_{kj}(t),$$

where the dynamics of the weight w_{kj} between a neuron j representing state s and a critic neuron k is given by equation 2.2, N_j is the number of neurons representing state s, and N_k is the number of critic neurons. The dynamics for the mean synaptic weight in the plasticity interval is

$$\dot{w}(s,t) = R + A(\tilde{\gamma}\lambda_{\mathrm{r}}(t) - \lambda_{\mathrm{l}}(t)) + C, \tag{3.1}$$

where $\lambda_{\mathrm{r/l}}$ denotes the mean of the activity trace $\Lambda^{\mathrm{r/l}}$, which can be obtained by averaging over the dynamics of the activity trace $\Lambda^{\mathrm{r/l}}$:

$$\dot{\lambda}_{\mathrm{r/l}} = -\frac{1}{\tau_{\mathrm{r/l}}}\left(\lambda_{\mathrm{r/l}} - \lambda(t)\right),$$

where $\lambda(t)$ denotes the postsynaptic firing rate. The solution of this inhomogeneous differential equation is

$$\lambda_{\mathrm{r/l}}(t) = \lambda_{\mathrm{r/l}}(t_0)\, e^{-(t-t_0)/\tau_{\mathrm{r/l}}} - \frac{1}{\tau_{\mathrm{r/l}}} \int_{t_0}^{t} \lambda(t') e^{-(t-t')/\tau_{\mathrm{r/l}}}\, dt'. \tag{3.2}$$

For simplicity, we consider $\lambda(t)$ to be constant while the agent is in state s. If the agent moves from s_i to s_{i+1} at time t_0, the rate of the critic neurons

changes from $\lambda(s_i)$ to $\lambda(s_{i+1})$, and the adaptive rate functions $\lambda_{r/l}$ describe the development of the postsynaptic rate according to equation 3.2 as

$$\lambda_{r/l}(t) = \lambda(s_i)\, e^{-(t-t_0)/\tau_{r/l}} + \lambda(s_{i+1})\left(1 - e^{-(t-t_0)/\tau_{r/l}}\right). \tag{3.3}$$

When the agent moves from s_i to s_{i+1}, the state-critic synapses associated with state s_i are plastic for a short time interval $\Delta t = t_2 - t_1$ (see section 2.2.1) after s_i is left. According to equations 3.1 and 3.3, the total change of the mean synaptic weight $w(s_i, t)$ during this time interval is

$$\begin{aligned}\int_{t_1}^{t_2} \dot{w}(s_i, t)\, dt = R\Delta t + A\big\{\tilde{\gamma}[(\lambda(s_i) - \lambda(s_{i+1}))\hat{\tau}_r + \lambda(s_{i+1})\Delta t] \\ - [(\lambda(s_i) - \lambda(s_{i+1}))\hat{\tau}_l + \lambda(s_{i+1})\Delta t]\big\} + C\Delta t\,,\end{aligned} \tag{3.4}$$

where $\hat{\tau}_r = \tau_r(e^{-(t_1-t_0)/\tau_r} - e^{-(t_2-t_0)/\tau_r})$ and $\hat{\tau}_l = \tau_l(e^{-(t_1-t_0)/\tau_l} - e^{-(t_2-t_0)/\tau_l})$. The period Δt in which the synapse is plastic is determined by the presynaptic activity thresholds θ_p and θ_l, as described in section 2.2.1: t_1 is the time at which the presynaptic rate reaches the upper threshold θ_p and t_2 the time at which it reaches the lower threshold θ_l. These times can be calculated by formulating an expression for the development of the presynaptic rate λ_s analogous to equation 3.3 and solving for $\lambda_s = \theta_p$ and $\lambda_s = \theta_l$, respectively:

$$\begin{aligned} t_1 &= -\tau_s \ln\left(\frac{\theta_p - \lambda_{in}}{\lambda_{ac} - \lambda_{in}}\right) + t_0 \\ t_2 &= -\tau_s \ln\left(\frac{\theta_l - \lambda_{in}}{\lambda_{ac} - \lambda_{in}}\right) + t_0. \end{aligned} \tag{3.5}$$

Here, λ_{ac} is the rate of state neurons in the active condition, and λ_{in} is the rate of state neurons in the inactive condition.

To compare the total change in $w(s_i)$ to the total change in $V(s_i)$ given by equation 1.3 after the agent's movement, we require a transformation between the units of synaptic weight and the units of the value function. We therefore start with two linear variable transformations:

$$V(s_i) = m_V \lambda(s_i) + c_V \tag{3.6}$$

$$\lambda(s_i) = m_\lambda w(s_i) + c_\lambda, \tag{3.7}$$

where $\lambda(s_i)$ is the firing rate of the critic neurons corresponding to state s_i. A linear relationship between the mean synaptic weight w and the firing rate of the critic neurons λ is a good fit for weights in the range 30 fC to 90 fC ($m_\lambda = 0.65$ Hz/fC, $c_\lambda = -13.7$ Hz). For the transformation from λ to V, we require that the extremal values of V lie within the range of appropriate weights: V_{min} corresponds to $w_{min} \geq 30$ fC, and V_{max} corresponds to $w_{max} \leq 90$ fC.

The extremal values of V are fully determined by the task and the value of γ. Parameters m_V and c_V can therefore be obtained by solving the linear set of equations,

$$V_{\min} = m_V m_\lambda \cdot w_{\min} + (m_V c_\lambda + c_V) \tag{3.8}$$

$$V_{\max} = m_V m_\lambda \cdot w_{\max} + (m_V c_\lambda + c_V), \tag{3.9}$$

for the particular task. The discrete time value function update rule 1.3 transformed into the units of synaptic weights according to the linear variable transformations 3.6 and 3.7 is

$$w(s_i) \leftarrow w(s_i) + \alpha \delta_i' \tag{3.10}$$

with

$$\delta_i' = \frac{1}{m_\lambda m_V} \left[r_{i+1} + m_V \left(\gamma \lambda(s_{i+1}) - \lambda(s_i) \right) + c_V (\gamma - 1) \right]. \tag{3.11}$$

The continuous and the discrete time formulations of the value function update are equivalent if the total change of the synaptic weight due to the continuous time update, equation 3.4, is the same as the discrete time update of the value function transformed into the units of synaptic weights, equation 3.10:

$$\int_{t_1}^{t_2} \dot{w}(s_i, t)\, dt \stackrel{!}{=} \alpha \delta_i'. \tag{3.12}$$

Comparing coefficients in equations 3.11 and 3.4 leads to the following mapping of parameters from the traditional discrete time algorithmic implementation of TD(0) to the continuous time neuronal implementation:

$$\begin{aligned} R &= \alpha \frac{1}{\Delta t} \frac{1}{m_\lambda m_V} r_{i+1} \\ \tilde{\gamma} &= \frac{\Delta t + \hat{\tau}_l (\gamma - 1)}{\Delta t + \hat{\tau}_r (\gamma - 1)} \\ A &= -\alpha \frac{1}{m_\lambda} \frac{\Delta t + \hat{\tau}_r (\gamma - 1)}{\Delta t (\hat{\tau}_r - \hat{\tau}_l)} \\ C &= \alpha \frac{1}{\Delta t} \frac{1}{m_\lambda m_V} c_V (\gamma - 1). \end{aligned} \tag{3.13}$$

3.2 Policy Mapping. In this section, we demonstrate that the synaptic weights between the state and the actor neurons are equivalent to the policy of a discrete time algorithmic implementation of TD(0). As described in section 2.1, whichever neuron fires first after the action suppression period determines the selected action for that state. Therefore, the policy of the agent $\pi(s, a)$ is equivalent to the probability that actor neuron a fires first

when the agent is in state s. As each actor neuron receives identical spike trains from the state neurons, the probability that an actor neuron fires first is dependent on the strength of its own synaptic weights and the strength of the synaptic weights of the other competing actor neurons. As each synapse undergoes slightly different weight changes, we calculate the probability of choosing a specific action with respect to the mean synaptic strength: $w_l = \frac{1}{N_j}\Sigma_j w_{lj}$, where w_{lj} is the strength of the synapse between state neuron j and actor neuron l and N_j is the number of neurons corresponding to state s. Given that the strengths of the incoming synapses to actor neurons p and q are w_p and w_q, respectively, the probability that p fires before q is

$$P\left(t_p^{\text{fs}} < t_q^{\text{fs}} \mid w_p, w_q\right) = \int_0^\infty f_{w_p}(t)\left[1 - \int_0^t f_{w_q}(t')\,dt'\right]dt, \tag{3.14}$$

where $t_{p/q}^{\text{fs}}$ is the first spike time of the corresponding neuron and $f_w(t)$ is the first spike time distribution of a neuron with synaptic weights of strength w. This expression is easily extendable to more than two weights. To obtain an analytical expression, we can fit empirical first spike time distributions with gamma probability density functions

$$f\,(t \mid \kappa, \theta) = \frac{1}{\theta^\kappa \Gamma(\kappa)} t^{\kappa-1} e^{-\frac{t}{\theta}}, \tag{3.15}$$

where Γ is the gamma function. For the purpose of this study, we determined parameters κ and θ for synaptic strengths between 30 fC and 90 fC in steps of 1 fC and a presynaptic rate of 42.63 Hz at 40 synapses, corresponding to the input received from the state neurons in the activated condition. Figure 4 shows two examples of empirical distributions and the corresponding fits. This results in the following expression for the probability of neuron p firing before neuron q:

$$P\left(t_p^{\text{fs}} < t_q^{\text{fs}} \middle| w_p, w_q\right) = \int_0^\infty f(t \mid \kappa(w_p), \theta(w_p))\left[1 - \gamma\left(\frac{t}{\theta(w_q)}, \kappa(w_q)\right)\right]dt.$$

Here $\gamma(t, \kappa)$ is the incomplete gamma function, with $\gamma(t, \kappa) = \frac{1}{\Gamma(\kappa)}\int_0^t e^{-t} t^{\kappa-1} dt$. As equation 3.14 is equivalent to the probability $\pi\,(s, a)$ of choosing action a (represented by neuron p) in state s, we can quantitatively compare the probabilities of taking given actions in a given state in the neuronal implementation of TD(0) with those learned by a discrete time algorithmic implementation.

The plasticity dynamics for the state-actor synapses given in equation 2.3 implies a change in weight proportional to the mean change in weight of the state-critic synapses and is thus analogous to the update of the preferences for the softmax action selection method given in equation 1.5; by combining equations 2.3 and 3.12, we obtain $w_a\,(s_i) \leftarrow w_a\,(s_i) + \beta'\delta_i'$, where $\beta' = \alpha B$.

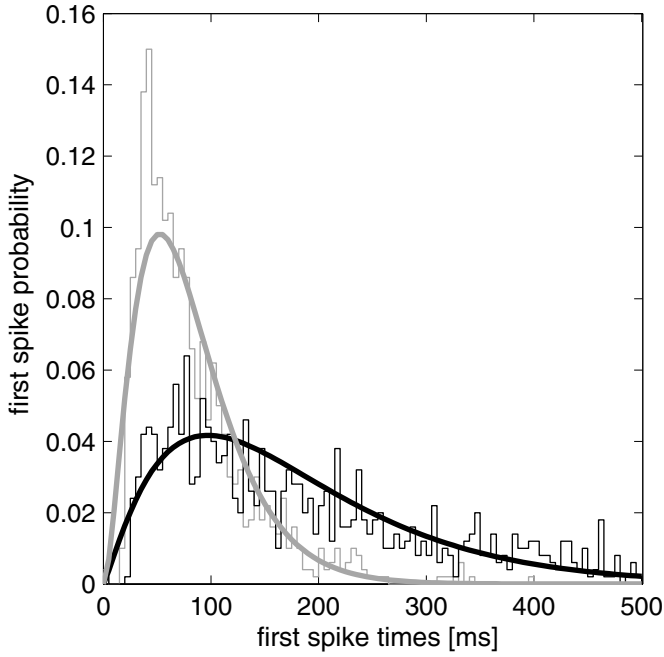

Figure 4: First spike time distribution for synaptic weights of 30 fC (black stairs) and 40 fC (gray stairs) collected over 1000 trials with a bin size of 5 ms. The solid curves indicate gamma probability function fits (see equation 3.15) to the two distributions, with parameters $\kappa = 2.2$, $\theta = 820$ (black curve) and $\kappa = 2.8$, $\theta = 287$ (gray curve), respectively.

However, a given weight change will not produce the same change in policy in the two implementations, as the neuronal action selection method described above and softmax action selection are different nonlinear functions. Therefore, the parameters β and β' in the two implementations are analogous but not equivalent.

4 Performance

4.1 Latency in a Gridworld Task. A common test case for reinforcement learning algorithms is that of the gridworld (Sutton & Barto, 1998), in which an agent can move from one state to the next in four different directions (see Figure 5, inset). In our variant of this task, only one state is associated with a reward. If the agent enters this state, it receives the reward. When the agent selects its next action, it is moved to a random state on the grid, independent of the direction chosen. Therefore, in order to maximize the amount of reward the agent receives, it must learn the quickest path from

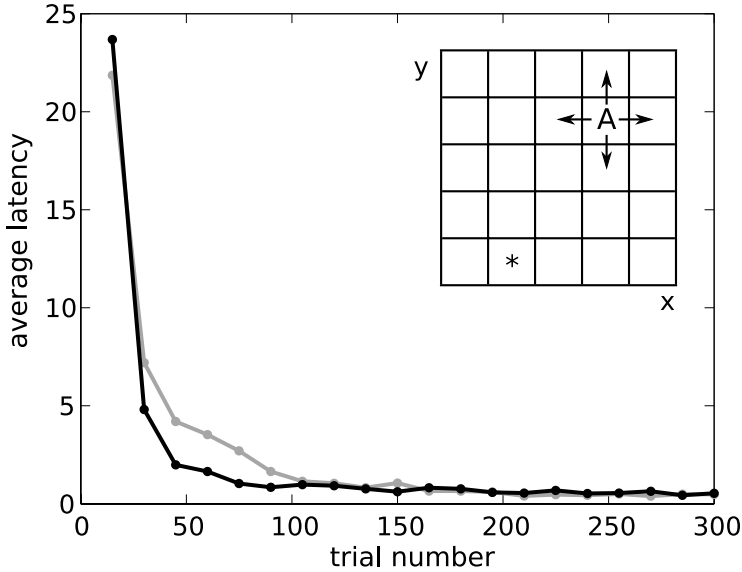

Figure 5: Learning curves for gridworld task. Average latency over 10 runs for $\alpha = 0.4$, $\gamma = 0.9$, and $\beta' = 0.8$ in the neuronal implementation (gray curve) and $\beta = 0.3$ for the corresponding algorithmic implementation (black curve). Each data point represents the average latency over 15 successive trials. Inset: The gridworld task. The agent (A) can move in four different directions, indicated by arrows. If the agent is at the edge of the world and selects an action that would take it outside the world, no movement is made. If the agent reaches the rewarded state (*), a reward $r = 12$ is awarded in the case of the discrete time algorithmic implementation, and the equivalent reward $R = 13.1$ fA (see section 3.1) is awarded in the case of the neuronal implementation. Independent of the direction of the next action selected, the agent is transported to a randomly chosen state.

any given state to the rewarded state. This is a nontrivial task, as the rewards are sparse; correct actions must also be learned for states that the agent passes through considerably earlier than the reward is obtained.

To measure performance, we consider the latency of the agent in finding the reward. The shortest distance from a state s with coordinates (x, y) to the rewarded state with coordinates (x_r, y_r) is given by the Manhattan distance: $d^M = \mid x - x_r \mid + \mid y - y_r \mid$. We define the latency of a trial as the difference between the number of steps d an agent takes to get from its initial position to the rewarded state and the Manhattan distance between those two states: $l = d - d^M$. The minimum latency for any state is therefore 0.

By performing a parameter scan, we established that the optimal parameters for a discrete time algorithmic implementation of the actor-critic learning agent (see appendix A) on this task are $\alpha = 0.4$, $\gamma = 0.9$, and $\beta = 0.3$. In analogy to the neuronal algorithm, we implemented the discrete time agorithm such that the value function is updated only when the action chosen resulted in a new state. This deviates from the textbook TD(0) algorithm, where the value function is updated every time step independent of whether the action chosen results in a new state. However, no improvement of performance on the gridworld task could be observed if the discrete time algorithm also performed value function and policy updates after actions that did not lead to a state transition (data not shown).

Choosing a reward of $r = 12$ results in extremal values for the value function of $V_{\min} = 14.8$ and $V_{\max} = 30.9$ (simulated results averaged over 10 runs). Setting $w_{\min} = 44.1$ fC and $w_{\max} = 68.99$ fC (see equations 3.8 and 3.9) results in particularly simple mapping parameters $m_V = 1$ s and $c_V = 0$, and thus to the following coefficients for the plasticity of the state-critic synapses: $R = 13.1$ fA, $A = 4.75$ fC, $\tilde{\gamma} = 0.98$, and $C = 0$ fA according to section 3.1 (see appendix A for the activity trace time constants and thresholds). The optimal performance for the neuronal implementation was achieved for $\beta' = 0.8$, resulting in $B = 2.0$ for the weight dynamics of the state-actor synapses (see section 3.2).

Figure 5 shows the learning curves for the continuous time neuronal implementation and the discrete time algorithmic implementation with optimal parameter settings averaged over 10 runs. Initially the two implementations learn equally fast: the average latency drops from more than 20 to less than 10 within 30 trials. The algorithmic implementation then learns slightly faster and improves its latency to approximately 1 by 75 trials, whereas the neuronal implementation requires 105 trials to reach this level of performance. Increasing the network size to 80 neurons per state and 40 critic neurons does not improve the learning speed of the neuronal implementation. In both cases, the performance improves hyperbolically, and both attain the same equilibrium performance, which remains stable over at least 600 trials.

4.2 Accuracy of Mapping. As the shortest path between two states in the gridworld environment is often not unique, there is no unique optimal policy. In the neuronal network, the policy for state s is determined by the average synaptic weights between the state neurons representing s and the actor neurons as described in section 3.2. Figures 6A to 6C show three substantially different policies learned by three runs of the neuronal algorithm with different initial seeds for the random number generator. The preferred movement direction for each state is indicated by an arrow $\vec{p}(s)$, where the x-component gives the difference between the probabilities of choosing east and west and the y-component the difference between the

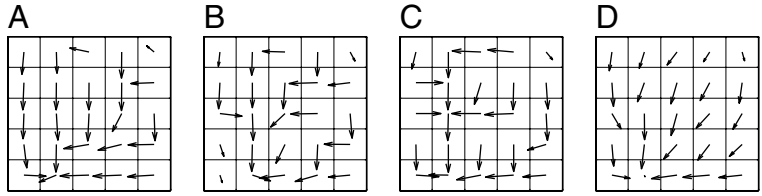

Figure 6: Policy learned by the neuronal network for the gridworld task (see Figure 5, inset). (A–C) Preferred movement direction for each state for different runs of the neuronal implementation. (D) Preferred movement direction for each state averaged over 10 runs. All parameters as in Figure 5.

probabilities of choosing north and south:

$$\vec{p}(s) = \begin{pmatrix} \pi(s, \text{east}) - \pi(s, \text{west}) \\ \pi(s, \text{north}) - \pi(s, \text{south}) \end{pmatrix},$$

where the probability of choosing an action in a given state is obtained from the state-actor weights as described in section 3.2. The policy averaged over a number of runs shown in Figure 6D gives a clearer indication of the general trend. In order to quantitatively compare the policies and value functions learned by the neuronal and algorithmic implementations, we restrict ourselves to consideration of these quantities averaged over several runs.

The average value function and policy for the two implementations is shown in Figure 7. Both implementations learn the same qualitative value function: a pronounced gradient can be seen, with states that are close to the reward state having higher values than states farther away. A quantitative measure of similarity is given by the root mean square error of the average value function learned by the continuous time neuronal (ctn) implementation $V^{\text{ctn}}(s)$ with respect to that learned by the discrete time algorithmic (dta) implementation $V^{\text{dta}}(s)$:

$$\sigma_{\text{V}} = \left(\frac{\sum_s (V^{\text{dta}}(s) - V^{\text{ctn}}(s))^2}{N_{\text{s}}} \right)^{1/2}, \tag{4.1}$$

where N_{s} is the number of states. For the value functions depicted in Figure 7, the root mean square error is 2.7 fC, or 3.9% of the maximum weight of $V^{\text{dta}}(s)$.

The average policies learned by the two implementations are also similar. In both cases, the preferred directions are generally directly toward the

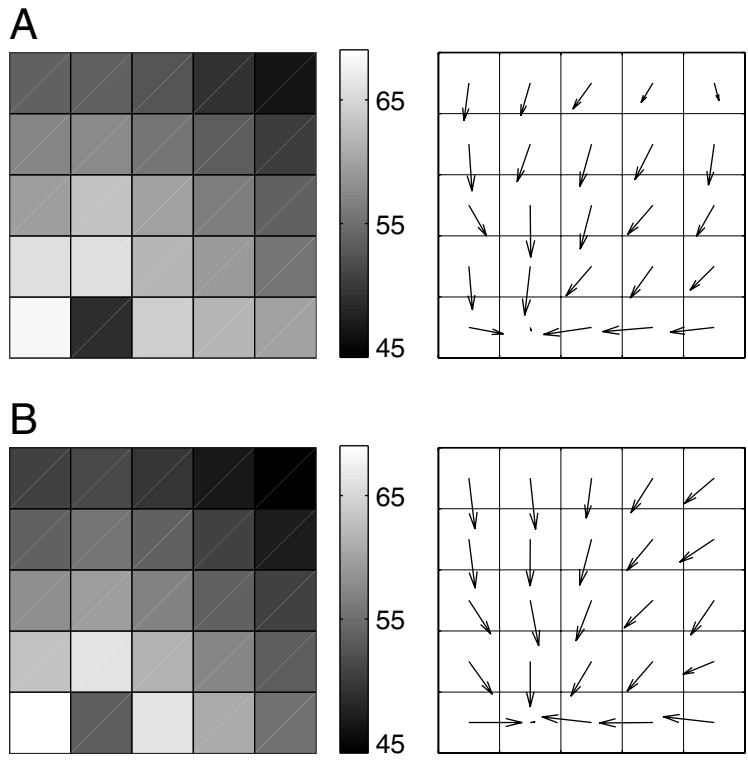

Figure 7: Comparison of the average value function and the average policy learned by the neuronal implementation and the algorithmic implementation. (A) Neuronal implementation. The value function $V(s)$ (left) is expressed as the average synaptic weight between the respective state neurons and the critic neurons in units of fC. The preferred direction for each state is determined by the synaptic weights between state neurons and actor neurons (see section 3.2). (B) Algorithmic implementation. As in *A*, the value function is transformed into units of synaptic weight as described in section 3.1. Parameters as in Figure 5; all plots averaged over 10 runs.

rewarded state, but the algorithmic implementation learns stronger preferences for the top row of states than the neuronal implementation.

4.3 Robustness to Parameters. The mapping derived in section 3.1 transforms the discrete time TD(0) parameters into the parameters of the synaptic plasticity rule as a function of various free parameters such as the time constants of the rapid and laggard activity traces, τ_r and τ_l. The range of possible time constants is limited by the period of time during which the actor module is inhibited after a new state is entered: the action

suppression period τ_{asp}. The results presented in the previous section are obtained with a comparatively long action suppression period of 1000 ms, permitting the use of large time constants: $\tau_{\text{l}} = 500$ ms, $\tau_{\text{r}} = 250$ ms. This is advantageous for the estimation of spike rates by the activity traces $\Lambda^{\text{r/l}}$ but has the disadvantage that the agent moves rather slowly. In this section, we investigate whether the network can be accelerated without loss of performance and mapping accuracy. Theoretically, the same discrete time TD(0) parameters can be realized with any choice of $\tau_{\text{r}} < \tau_{\text{l}} < \tau_{\text{asp}}$, as this choice just determines a different set of continuous time parameters A, $\tilde{\gamma}$, R, and C. To test this, we investigate the robustness to the relationship of the rapid and laggard time constants $\tau_{\text{r}}/\tau_{\text{l}}$ and the action suppression period τ_{asp}.

Figure 8 shows the robustness of the network behavior to a systematic variation in τ_{r} while keeping τ_{l} constant. The average summed reward the agent accumulates over 4400 steps exhibits a weak dependence on $\tau_{\text{r}}/\tau_{\text{l}}$. The best performance is seen for $\tau_{\text{r}}/\tau_{\text{l}} = 0.5$, but a robust learning behavior is exhibited for the entire range $0.05 \leq \tau_{\text{r}}/\tau_{\text{l}} \leq 0.99999$. Similarly, the root mean square error of $V^{\text{ctn}}(s)$ with respect to $V^{\text{dta}}(s)$, equation 4.1, remains low across the range, with a minimum at $\tau_{\text{r}}/\tau_{\text{l}} = 0.3$. Note that at $\tau_{\text{r}}/\tau_{\text{l}} = 0.05$, the time constant of the rapid activity trace is in the same order of magnitude as the average interspike interval of the critic neurons, and so the trace itself conveys very little information about the postsynaptic rate. However, the synapse integrates the activity trace throughout the entire plastic interval, equation 2.2. For synaptic time constants much smaller than the average interspike interval, the integral of the postsynaptic trace between two postsynaptic spikes is 1; consequently, the integral of the trace over the plasticity interval gives the sum of postsynaptic spikes in this interval. As a result, the synapse can estimate the postsynaptic rate even for very small time constants. Learning still takes place in the limit $\tau_{\text{r}} = 0$ ms, albeit with the minimum average summed reward recorded. Similarly, the error $\sigma_V / V^{\text{dta}}_{\text{max}}$ is maximal for $\tau_{\text{r}} = 0$ ms at 16%. At the other end of the range, robust learning is still exhibited for $\tau_{\text{r}}/\tau_{\text{l}} = 0.99999$, although the difference between the fast and the slow postsynaptic activity traces is almost 0. The very small differences between the two postsynaptic activity traces due to similar time constants can still be detected, as the multiplicative factor A increases in inverse proportion to the difference between the time constants (see equation 3.13). We also tested the limiting case $\tau_{\text{r}} = \tau_{\text{l}}$, using the parameters A, $\tilde{\gamma}$, R, and C, obtained for $\tau_{\text{r}}/\tau_{\text{l}} = 0.99999$. The resultant network is not capable of learning the task. The average reward accumulated over 4400 steps was 52 ± 5 and $\sigma_V / V^{\text{dta}}_{\text{max}} = 0.62$. This demonstrates that both traces are needed to implement TD learning; however, the relationship between the time constants of the traces has very little influence on the learning performance and mapping accuracy.

Shorter action suppression periods entail smaller time constants and thus shorter time intervals of plasticity Δt. As the difference in the number of critic neuron spikes Δ_{sp} evoked by states with weights of w_1 and w_2

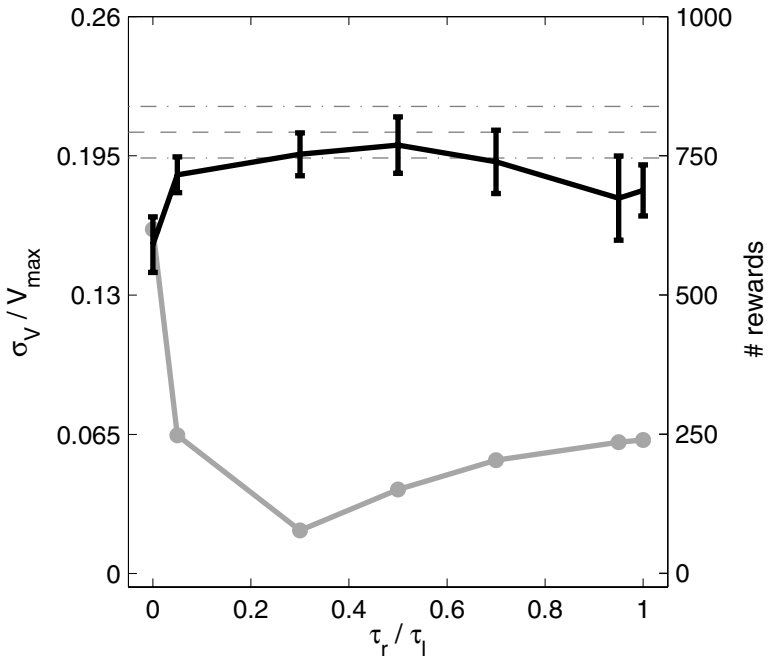

Figure 8: Mapping accuracy and performance as functions of the relationship between the time constants of the rapid and laggard activity traces τ_r/τ_l for constant $\tau_l = 500$ ms and $\tau_{asp} = 1000$ ms. Root mean square of the average value function of the neuronal implementation divided by the maximum value of the function learned by the algorithmic implementation (gray curve). Number of times the neuronal implementation locates the reward state in 4400 steps averaged over 10 runs (black curve; error bars show ±1 standard deviation). The gray dashed line shows the mean number of rewards (±1 SD; dash-dotted lines) obtained by the algorithmic implementation over the same period. Reinforcement learning parameters α, γ, r, and β, β' as in Figure 5; for all other parameters, see appendix A.

decreases linearly with the plasticity interval, $\Delta_{sp} = m_\lambda(w_1 - w_2)\Delta t$, the ability of the system to distinguish between the values of states decreases for shorter plasticity intervals. To investigate the impact of this, we vary the action suppression period systematically while maintaining the relationships $\tau_s = \tau_l = 0.5 \cdot \tau_{asp}$ and $\tau_r = 0.5 \cdot \tau_l$. Figure 9 shows the average summed reward accumulated by the agent over 4400 steps and the root mean square error of $V^{ctn}(s)$ with respect to $V^{dta}(s)$ as functions of the action suppression period. Between 1000 ms and 600 ms, the mapping accuracy deteriorates only slightly, but between 600 ms and 200 ms, the root mean square error increases rapidly to 16.95 fC, or 24.57% of the maximum weight

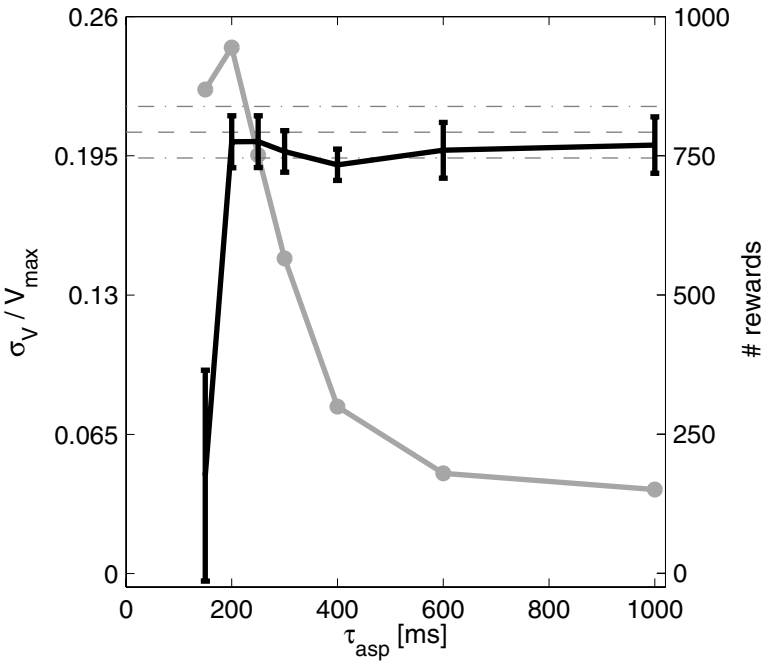

Figure 9: Mapping accuracy and performance as functions of the action suppression period τ_{asp}. All curves and markings as in Figure 8. Reinforcement learning parameters α, γ, r and β, β' as in Figure 5. For all other parameters see appendix A.

of $V^{\text{dta}}(s)$. However, despite the deterioration of the mapping accuracy, the neuronal implementation still performs well on the gridworld task even for very short action suppression periods. Between 1000 ms and 200 ms, there is no loss in performance; the neuronal implementation finds the reward state as often as the algorithmic implementation. Between 200 ms and 150 ms, there is a sudden drop in performance. Note that for $\tau_{\text{asp}} = 150$ ms, τ_{l} and Δt are shorter than the average spike interval of the postsynaptic spikes. As a result, the synapse can no longer detect the difference between the rapid and the laggard activity trace, and TD learning is no longer implemented. Due to this fundamental limit in representing the change of postsynaptic rate, it is not to be expected that time averaging of the rate can be replaced by spatial averaging in a larger network; indeed, doubling the number of state and critic neurons did not significantly increase the performance of the neuronal implementation for an action suppression period of 150 ms.

5 Alternative Plasticity Rules Implementing TD Learning

In sections 2.2.1 and 2.2.2, we presented a set of properties for state-critic and state-actor synapses that enable the implementation of value function and policy updates for actor-critic TD(0) learning. The motivation for the specific plasticity rules we formulated was to map the properties of the discrete time algorithm as accurately as possible to continuous time plasticity mechanisms. In this section, we relax this constraint and consider some other biologically plausible plasticity rules with the described properties.

The first property of the state-critic neurons is that they exhibit negligible plasticity except for a short period just after the agent has left the corresponding state. To fulfill this criterion, we introduced a presynaptic activity trace Λ_j^s (see equation 2.1) and made the plasticity contingent on passing certain thresholds. As an alternative timing mechanism, let us consider a presynaptic efficacy trace ε_j, which is set to 0 at every presynaptic spike and recovers exponentially to 1 with a time constant τ_ϵ:

$$\dot{\varepsilon_j}(t) = -\frac{\varepsilon_j - 1}{\tau_\varepsilon} - \sum_{t_j^f} \varepsilon_j \delta\left(t - t_j^f\right), \tag{5.1}$$

where t_j^f denotes the fth spike of the presynaptic neuron j. Such traces were introduced by Froemke and Dan (2002) to define a model for STDP that accounts for data obtained from triplet and quadruplet spike protocols. We can now define an alternative formulation for the plasticity of state-critic synapses:

$$\dot{w}_{kj}(t) = \varepsilon_j(t)\Lambda_j^s(t)\left(R + A\left(\tilde{\gamma}\,\Lambda_k^r(t) - \Lambda_k^l(t)\right) + C\right). \tag{5.2}$$

The correct timing properties of this rule can be understood as follows. Before the agent enters a state, the neurons representing this state receive very little stimulation and thus have an extremely low rate. While the agent is in the state, the presynaptic rate is high, but the mean efficacy is effectively 0 if τ_ϵ is much longer than the interspike interval. The product $\varepsilon_j(t)\Lambda_j^s(t)$ is significant only when the agent has just left the state and the efficacy is recovering to 1. The time interval of nonnegligible plasticity is determined by the decay time of the presynaptic activity trace back to 0. If the time constant τ_s of the presynaptic activity trace is small in comparison to τ_{asp}, the interval of nonnegligible plasticity is short enough such that only one state transition is taken into account. This is depicted in Figure 10 for $\tau_\varepsilon = 4000$ ms and $\tau_s = 200$ ms.

This plasticity rule is not such a strict implementation of TD(0) learning as equation 2.2 because the synapse is weakly plastic thoughout. However, if we assume that the changes in the synaptic weights outside the time interval τ_{asp} after the agent has left the corresponding state are negligibly small, it is once again possible to derive a mapping from the parameters of

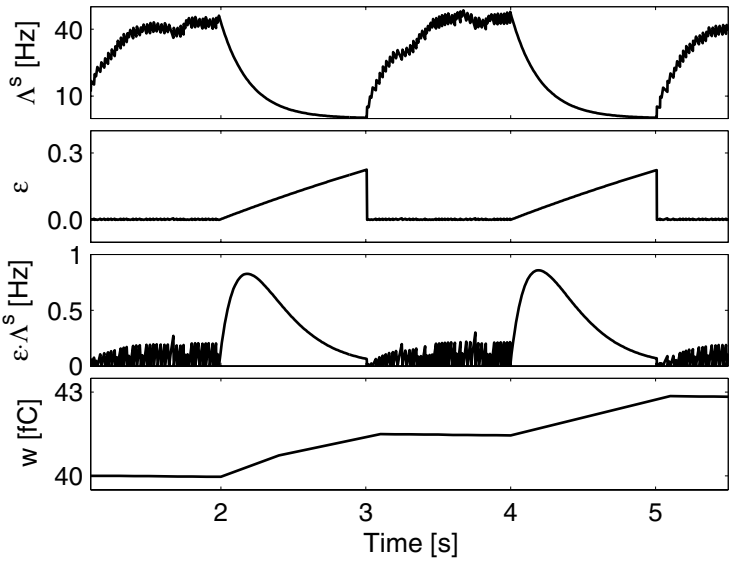

Figure 10: Plasticity of state-critic synapses: alternative formulation. The agent leaves a lower-valued state for a higher-valued state at 2 s and 4 s and returns to the lower-valued state at 3 s and 5 s. The top three panels show for one state-critic synapse associated with the lower-valued state the presynaptic trace Λ^s, the efficacy ϵ, and the product $\epsilon \cdot \Lambda^s$, respectively, as functions of time. The bottom panel shows the average weight (over 800 synapses) of the state-critic synapses associated with the lower-valued state; the synaptic weight remains essentially constant except for the period shortly after the agent leaves the corresponding state.

the discrete time value function update of TD(0) learning (see appendix B). We tested the learning behavior for this synaptic plasticity rule in simulations in the gridworld task (see section 4.1) with $\tau_\varepsilon = 1000$ ms, $\tau_s = 300$ ms, $\tau_{asp} = 1000$ ms, $\tau_l = 500$ ms, and $\tau_r = 0.5 \cdot \tau_l$. For the discrete time TD(0) learning parameters $\alpha = 0.4$, $\gamma = 0.9$, and $r = 12$ with $c_V = 0$, we obtain for the synaptic plasticity parameters $A = 1.1$ fC/Hz^2, $\tilde{\gamma} = 0.98$, $R = 2.8$ fC, and $C = 0$ fC. Over 10 runs, the average measure for the accuracy of the mapping σ_V (see equation 4.1) is 2.1 fC, or 3.3% of the maximum value of $V^{dta}(s)$, and the average summed reward after 4400 steps is 774.2 ± 35.5 (data not shown). This is a comparable performance to that obtained by the plasticity rule given by equation 2.2 ($\sigma_V = 2.7$ fC; average summed reward after 4400 steps: 769 ± 50.8). The plasticity rule given by equation 5.2 therefore also represents a biologically plausible implementation of value function updates for TD(0) learning in spiking neuronal networks. This rule can also be considered as belonging to the class of differential Hebbian rules (see

section 2.2.1); this time it depends on the correlation between presynaptic activity and the rate of change of postsynaptic activity. It is very similar to rules used in isotropic sequence order learning (Porr & Wörgötter, 2003) and nonspiking network models of TD learning (Barto, 1995; Houk, Adams, & Barto, 1995; Foster et al., 2000).

The second property of the state-critic neurons is that they are sensitive to a characteristic dynamic response of the critic neurons. In section 2.2.1 we made the simple assumption that the rate of the critic neurons encodes the value of the state, and formulated a synaptic plasticity rule that is sensitive to changes in postsynaptic rate. However, other dynamic responses could also encode a change in input stimulus, for example, a transient rate excursion before relaxing to the new rate (e.g., Gazeres, Borg-Graham, & Frégnac, 1998; Muller, Buesing, Schemmel, & Meier, 2007).

Similarly, alternative rules can be found that implement the actor-critic policy update. The first property of the state-actor synapses is that they be negligibly plastic, except when the corresponding action has just been chosen. We implemented this by making the plasticity contingent on the postsynaptic activity Λ_l^{a} being above a certain threshold. An alternative formulation of the state-actor update rule that fulfills this timing criterion can be obtained analogous to equation 5.2 by multiplying the weight update rule, equation 2.3, by the postsynaptic rate:

$$\dot{w}_{lj}(t) = \frac{B}{N_k} \Lambda_l^{\mathrm{a}} \sum_k \dot{w}_{kj}(t). \tag{5.3}$$

The second property of the state-actor synaptic update rule is that of proportionality to the updates of the state-critic synapses. This entails the existence of a mechanism that is not strictly local, as information is required that cannot be derived from pre- and postsynaptic activity alone; this is the price of the modular structure of the actor-critic architecture. We implemented this directly in equation 2.3, by positing that synaptic changes spread along the axon of the state neurons. However, the updates in the state-critic synapses are dependent on the dynamic response of the critic neurons. Therefore, an alternative approach is to assume that the critic neurons project to the state-actor synapses and influence their plasticity as a neuromodulatory third factor, as has been demonstrated in the case of dopamine for corticostriatal synapses (for a review, see Reynolds & Wickens, 2002).

6 Discussion

In this study we demonstrated that a network of spiking neurons can implement an actor-critic TD learning agent. To this end, we derived a mapping that permits the discrete time algorithm to be performed by a pulse-coupled dynamical system operating in continuous time. As in the traditional discrete time algorithm, the agent moves in a discrete state space and takes

discrete actions. However, many interesting real-world problems, such as learning how to catch a ball, cannot easily be reduced to such a framework. Once it is shown that it is possible to implement this simple case, a natural next step would be to extend the network model to be able to solve continuous space and time tasks. Nonspiking algorithms for continuous time problems have already been been formulated, for example, in Munos (2006) and Doya (2000b). We investigated the actor-critic method rather than actor-only techniques such as the REINFORCE algorithm, as the latter is dependent on frequent reward, whereas many real-world problems are characterized by sparse, delayed, and unreliable rewards (Williams, 1992; Seung, 2003). The basis of the actor-critic is that the critic generates an internal reward that allows the actor to adapt its policy even when no external reward is present. This division of labor enables an actor-critic learner to solve the temporal credit assignment problem even for tasks with sparse rewards, such as the gridworld used as an example here. We investigated the actor-critic method in preference to TD learning approaches based on action values such as Sarsa and Q-learning (see Sutton & Barto, 1998), which are also capable of learning a task with sparse rewards, because the modular layout of the actor-critic architecture already suggests an appropriate network structure. Indeed, there is experimental evidence to support an actor-critic paradigm in which different areas of the brain fulfill the roles of actor and critic (O'Doherty et al., 2004; Tricomi, Delgado, & Fiez, 2004). The modular structure of the actor-critic architecture has also inspired numerous theoretical studies of the basal ganglia (Barto, 1995; Houk et al., 1995; Suri & Schultz, 1999; see Joel, Niv, & Ruppin 2002, for a recent review). However, recent evidence also supports other modes of TD learning such as those based on action values (Morris, Nevet, Arkadir, Vaadia, & Bergman, 2006) or an actor-director model (Attalah, Lopez-Paniagua, Rudy, & O'Reilly, 2007; Lerchner, La Camera, & Richmond, 2007), so further research into neuronal implementations of these techniques is needed.

Here we have focused on neural mechanisms that are capable of implementing one specific algorithm for system-level learning, but we do not claim that this is the algorithm used by the brain. Indeed, it seems probable that the brain employs a variety of learning strategies depending on the problem to be solved. Various parts of the brain have been associated with different categories of learning, such as the cerebellum with supervised learning, the basal ganglia with reinforcement learning, and the cerebral cortex with unsupervised learning (see Doya, 2000a, for a review). It is also possible that the brain uses combinations of strategies that have yet to be formalized.

Reinforcement learning is often criticized for taking too long to converge to be practical for biological purposes. In our example, an agent requires approximately 100 attempts to learn the task. However, the task is much harder than it may initially seem, as the agent has no information about the underlying gridlike structure. Consider the following thought experiment.

You are presented with a card with an easily recognizable image, such as is used in the game memory. You may choose a number between 1 and 4, at which point the card is (deterministically) replaced with a different card. You may keep choosing numbers until you find the card with a picture of a doughnut on it, for which you receive a reward. You then start again from a random card. Informal experiments with such a setup in our lab reveal that it is very difficult to learn a fast route to the rewarded card, even if the subjects are aware that each card has been assigned to a position on a grid and each number has been assigned to a direction. Much faster convergence times can be achieved by reinforcement learning techniques if knowledge of the underlying problem structure is incorporated in the algorithm, for example, the contiguity of space in a navigation task (Foster et al., 2000). But in some respects, the task is also easier than it seems, as the agent has perfect knowledge of its state and policy. It makes no identification errors, whereas the authors sometimes mistake the second floor of their office building for the third floor, and no execution errors, whereas the authors know they are living in Japan but still often look the wrong way when crossing the road. Another limitation of reinforcement learning in its basic form is that the learning agents are purely reactive. Once the learning has stabilized, the agent will have the same probabilistic response to a given stimulus at each occurrence. This is reflected in the feedforward architecture of our model. A more sophisticated system would be able to switch between different internal states as a result of its own dynamics, for example, due to motivation or attention, and thus generate different probabilistic responses to the same stimulus.

We have shown that a reasonably small network with biologically realistic spike rates can perform the necessary calculations to update its value function and policy in as little as 200 ms. This seems to be compatible with the reaction times required for survival. Due to the paucity of information at the synaptic level for response times of less than 200 ms, the performance of the network cannot be improved by increasing the number of neurons in the network. The performance of the network is also robust with respect to the time constants of the activity traces, even in the limits as τ_r approaches 0 or τ_l. In all experiments, the parameters A, $\tilde{\gamma}$, C, and R were carefully chosen (see appendix A), which may give the impression that the system is highly sensitive to parameter choice. In fact, the careful choice of parameters was motivated by the requirement of equivalence with the discrete time algorithmic implementation; the network learns for a wide range of parameters in the same way that the traditional discrete time algorithm can solve a task for a range of the parameters α and γ. This raises the question of how such parameters might be determined by the brain given that different parameters are optimal for different tasks. Does the brain employ one set of parameters that are "good enough" or are appropriate parameters learned through metaplasticity (Doya, 2002)? This question can be addressed only by carefully designed behavioral experiments.

Our model relies on two particular types of synaptic plasticity: differential Hebbian in the critic module and heterosynaptic axonal plasticity in the actor module. Aspects of both mechanisms have been discussed in the experimental and theoretical literature, but there is no direct evidence for the complete forms of the weight dynamics postulated here. The plasticity of the critic module is local, depending on only pre- and postsynaptic spikes, except for its sensitivity to a global signal encoding the reward. An experimental setup to reveal this kind of plasticity could be realized by stimulating a pre- and postsynaptic neuron pair for a certain interval so that they fire at constant rates. At the end of the interval, the presynaptic stimulation should be stopped while the strength of the postsynaptic stimulation is increased or decreased. The prediction of the plasticity rule proposed in section 2.2.1 is that the synaptic strength remains constant during the simultaneous stimulation and subsequently undergoes potentiation or depression depending on whether the postsynaptic stimulation is increased or decreased. The plasticity of the actor module is local in that it depends on the postsynaptic activity, but it also depends on information available to the axon of the presynaptic neuron, which is not local in the strict sense. The experimental prediction of the rule proposed in section 2.2.2 is that axonal spread of long-term potentiation and long-term depression has a much greater effect on synapses if their respective postsynaptic neurons are active than if they are not. However, an alternative plasticity for the actor module, discussed in section 5, is that the output of the critic module influences the state-actor directly as a neuromodulatory third factor. In this case, the experimental prediction is that such synapses undergo the greatest changes when there has been recent pre- and postsynaptic activity and that potentiation is induced by a transient increase in the neuromodulator and depression by a transient decrease. An obvious candidate for the neuromodulatory third factor is dopamine. In further work, we will investigate to what extent the plasticity rules postulated here are compatible with or can be replaced by well-established synaptic plasticity mechanisms.

The action selection mechanism implemented here is both extremely simple and rather vulnerable: if the neuron representing west dies, the agent will never be able to go west again. However, it is easy to conceive of extending the network to incorporate a more robust action selection mechanism, for example, via attractor networks (Amit, 1989), competing synfire chains (Hayon, Abeles, & Lehmann, 2004), or rate-based coding of action populations (Georgopoulos et al., 1982).

One interesting technological aspect of this work is that we have taken a discrete, serial algorithm and reformulated it in continuous time exploiting finely grained parallelism. In our model, there is no central instance adjusting the value function and policy; the synapses adjust themselves to the appropriate values on the basis of their individual dynamics. Adopting standard algorithms to the finely grained parallelism of the biological substrate does not give us an immediate practical advantage. If

the biological substrate is simulated on standard hardware, this is likely to be less efficient than a direct implementation of the standard algorithm. However, intense research is under way to create hardware emulating the capabilities of the neural tissue, prominently FPGAs and analog VLSI (Guerrero-Rivera, Morrison, Diesmann, & Pearce, 2006; Philipp, Grübl, Meier, & Schemmel, 2007). Fundamental problems have been solved, but limitations still exist in the number of neurons and synapses. Once these technologies can produce devices with an appropriate number of neurons and synapses, they can make full use of microscopically parallel algorithms like the one developed in this article and solve problems with at least the same speed and accuracy as natural brains. However, almost like standard computers, these devices are general-purpose engines and need to be programmed to solve a particular task. Thus, from a technological perspective, this work can also be interpreted as an exploration into neuronal software for the upcoming neuromorphic hardware.

Appendix A: Parameters and Simulation

The neuron model used in this study is a current-based leaky integrate-and-fire neuron. In the subthreshold range, the dynamics of the membrane potential v is

$$\frac{dv}{dt} = -\frac{v}{\tau_m} + \frac{I(t)}{C},$$

where τ_m is the membrane time constant, C is the capacity, and $I(t)$ is the input current to the neuron, which is the sum of any external current and the synaptic currents. The synaptic current is given by

$$I_{syn}(t) = \sum_j w_j \sum_{t_j^n < t} \delta\left(t - t_j^n\right),$$

where t_j^n is the nth spike of the presynaptic neuron j and w_j the weight of the corresponding synapse. The arrival of a single spike of weight $w = 50$ fC evokes an instantaneous postsynaptic potential of amplitude 0.2 mV, which decays with the time constant of τ_m. If the membrane potential passes the threshold θ, a spike is emitted, and the neuron is clamped to the reset potential V_{reset} for the duration of the refractory period τ_r. The spike is delivered to the downstream neurons with a synaptic propagation delay d. The neuronal parameters used in our simulations are given in Table 1.

In addition to input received from other neurons in the network, each neuron receives both excitatory (82.1 Hz; 10 fC) and inhibitory (43.2 Hz; −10 fC) background stimulation in the form of independent Poissonian spike trains. The neurons representing the active state receive a further

Table 1: Neuronal Parameters.

τ_m	C	θ	V_{reset}	τ_r	d
10 ms	250 pF	20 mV	0 mV	2 ms	5 ms

Table 2: Activity Thresholds for Different Action Suppression Times Δt_{asp}.

Δt_{as} [ms]	θ_h [Hz]	θ_p [Hz]	θ_l [Hz]	θ_a [Hz]
150	40	35	20	2.6
200	40	30	15	2
250	40	30	15	1.6
300	40	30	15	1.3
400	40	30	15	1
600	36	31	10	0.7
1000	36	31	10	0.4

stimulus from the environment in the form of a DC current of amplitude 160 pA. When the agent enters a new state, the actor neurons receive a DC current of -250 pA for the duration of the action suppression period.

Unless otherwise stated, the time constants for the activity traces (see sections 2.2.1 and 2.2.2) were chosen with respect to the action suppression period τ_{asp} as follows: $\tau_s = \tau_l = \tau_a = 1/2 \cdot \tau_{asp}$ and $\tau_r = 1/2 \cdot \tau_l$. Appropriate activity thresholds to determine the plasticity window were chosen as in Table 2; all synaptic weights were initialized at 50 fC. The network was simulated with NEST (see www.nest-initiative.org and Gewaltig & Diesmann, 2007) using a time step of 0.1 ms.

For the purposes of comparison, we implemented an actor-critic TD learning agent with softmax action selection equation 1.4 as a traditional discrete time algorithm in C++. To make the comparison as fair as possible, we deviated from the textbook approach in the following two ways. First, in the neuronal implementation, the synaptic weights are restricted to the range [30 fC, 90 fC], which results in a maximum probability of choosing a given action of 96.7% and a minimum probability of $3.1 \cdot 10^{-5}$%. We therefore restricted the preferences $p(a, s)$ in the discrete time algorithmic implementation to the range [1, 5.47], which results in the same maximum probability as in the neuronal implementation but a minimum probability of 0.38%. Second, analogous to the neuronal implementation, value function updates were carried out only in the discrete time algorithm when the action chosen resulted in a new state. If the preferences are not restricted, the discrete time algorithmic implementation attains a marginally better equilibrium performance. If the agent can also learn from actions that do not result in a state transition, no improvement in performance can be observed.

Appendix B: Value Function Mapping for an Alternative Plasticity Rule

If we assume that the synapses following the weight dynamics given in equation 5.2 are negligibly plastic except for a time period equal to the action suppression time τ_{asp} after the agent has left the corresponding state, we can derive a mapping from the parameters of the discrete time value function update of TD(0) learning to the synaptic parameters as follows.

As in section 3.1, we consider the mean weight change for the state-critic synapses associated with state s_i when the agent moves from state s_i to state s_{i+1} at time t_0. If we assume that the rate of the state neurons representing s_i is 0 when the agent is in s_{i+1}, the averaged value for the corresponding efficacy trace, equation 5.1, and the presynaptic activity trace, equation 2.1, are

$$\varepsilon_{s_i}(t) = 1 - e^{-\frac{t-t_0}{\tau_\varepsilon}}$$
$$\lambda_{s_i}(t) = \lambda(s_i)\, e^{-\frac{t-t_0}{\tau_s}}.$$

The averaged postsynaptic activity traces λ_{r} and λ_l are given by equation 3.3, and the mean synaptic weight has the following dynamics:

$$\dot{w}(s_i, t) = \varepsilon_{s_i}(t)\, \lambda_{s_i}(t)\, (R + A(\tilde{\gamma}\lambda_{\text{r}}(t) - \lambda_l(t)) + C).$$

The synaptic weight update is equivalent to the discrete time value function update expressed in synaptic weights (see equation 3.10), if the following equation if fulfilled:

$$\int_{t_0}^{t_0+\tau_{\text{asp}}} \dot{w}(s_i, t)\, dt =^{!} \alpha\delta'_i.$$

A comparison of the coefficients leads to the following mapping:

$$R = \alpha \frac{1}{m_V m_\lambda \lambda(s_i)} \frac{1}{T^{\text{s}}} r$$
$$A = \alpha \frac{1}{m_\lambda \lambda(s_i)} \frac{\gamma T_1^{\text{r}} + T_2^{\text{r}}}{T_1^{\text{l}} T_2^{\text{r}} - T_1^{\text{r}} T_2^{\text{l}}}$$
$$\tilde{\gamma} = \frac{T_2^{\text{l}} + \gamma T_1^{\text{l}}}{T_2^{\text{r}} + \gamma T_1^{\text{r}}}$$
$$C_V = \alpha \frac{1}{m_V m_\lambda \lambda(s_i)} \frac{1}{T^{\text{s}}} (1-\gamma)\, c_V,$$

with $\hat{\tau}(x) = 1/x \cdot (1 - e^{-x\tau_{\text{asp}}})$ and

$$
\begin{aligned}
T^{\text{s}} &= \hat{\tau}\left(\frac{1}{\tau_{\text{s}}}\right) - \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\varepsilon}}\right) \\
T_1^{\text{r}} &= \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\text{r}}}\right) - \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\text{r}}} + \frac{1}{\tau_{\varepsilon}}\right) \\
T_1^{\text{l}} &= \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\text{l}}}\right) - \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\text{l}}} + \frac{1}{\tau_{\varepsilon}}\right) \\
T_2^{\text{r}} &= \hat{\tau}\left(\frac{1}{\tau_{\text{s}}}\right) - \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\text{r}}}\right) - \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\varepsilon}}\right) + \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\varepsilon}} + \frac{1}{\tau_{\text{r}}}\right) \\
T_2^{\text{l}} &= \hat{\tau}\left(\frac{1}{\tau_{\text{s}}}\right) - \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\text{l}}}\right) - \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\varepsilon}}\right) + \hat{\tau}\left(\frac{1}{\tau_{\text{s}}} + \frac{1}{\tau_{\varepsilon}} + \frac{1}{\tau_{\text{l}}}\right).
\end{aligned}
$$

Acknowledgments

We warmly thank Sirko Straube for his substantial contributions to the initial phase of this study. We also acknowledge Tobias Potjans and Marc-Oliver Gewaltig for fruitful discussions and Tilo Schwalger for comments on an earlier version of the article. This work was partially funded by DIP F1.2, BMBF Grant 01GQ0420 to the Bernstein Center for Computational Neuroscience Freiburg, and EU Grant 15879 (FACETS).

References

Amit, D. J. (1989). *Modeling brain function*. Cambridge: Cambridge University Press.

Antonov, I., Antonova, I., Kandel, E. R., & Hawkins, R. D. (2003). Activity-dependent presynaptic facilitation and Hebbian LTP are both required and interact during classical conditioning in *Aplysia*. *Neuron*, *37*(1), 135–147.

Attalah, H. E., Lopez-Paniagua, D., Rudy, J. W., & O'Reilly, R. C. (2007). Separate neural substrates for skill-learning and performance in the ventral and dorsal striatum. *Nat. Neurosci.*, *10*(1), 126–131.

Baras, D., & Meir, R. (2007). Reinforcement learning, spike-time-dependent plasticity, and the BCM rule. *Neural Comput.*, *19*, 2245–2279.

Barto, A. G. (1995). Adaptive critic and the basal ganglia. In J. C. Houk, J. Davis, & D. Beisser (Eds.), *Models of information processing in the basal ganglia* (pp. 215–232). Cambridge, MA: MIT Press.

Barto, A., Sutton, R. S., & Anderson, C. W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Trans. Sys. M. Cybern.*, *13*, 834–846.

Bertsekas, D. P., & Tsitsiklis, J. N. (1996). *Neuro-dynamic programming*. Belmont, MA: Athena Scientific.

Bi, G.-q., & Poo, M.-m. (1998). Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type. *J. Neurosci.*, *18*, 10464–10472.

Bonhoeffer, T., Staiger, V., & Aertsen, A. (1989). Synaptic plasticity in rat hippocampal slice cultures: local "Hebbian" conjunction of pre- and postsynaptic stimulation leads to distributed synaptic enhancement. *Proc. Natl. Acad. Sci. USA*, *86*(20), 8113–8117.

Cash, S., Zucker, R., & Poo, M.-m. (1996). Spread of synaptic depression mediated by presynaptic cytoplasmic signaling. *Science*, *272*(5264), 998–1001.

Dayan, P. (1992). The convergence of TD(λ) for general λ. *Machine Learning*, *8*, 341–362.

Dayan, P., & Sejnowski, T. (1994). TD(λ) converges with probability 1. *Machine Learning*, *14*, 295–301.

Doya, K. (2000a). Complementary roles of basal ganglia and cerebellum in learning and motor control. *Curr. Opin. Neurobiol.*, *10*, 732–739.

Doya, K. (2000b). Reinforcement learning in continuous time and space. *Neural Comput.*, *12*(1), 219–245.

Doya, K. (2002). Metalearning and neuromodulation. *Neural Networks*, *15*, 495–506.

Farries, M. A., & Fairhall, A. L. (2007). Reinforcement learning with modulated spike timing-dependent synaptic plasticity. *J. Neurophysiol.*, *98*, 3648–3665.

Fitzsimonds, R., Song, H.-j., & Poo, M.-m. (1997). Propagation of activity-dependent synaptic depression in simple neural networks. *Nature*, *388*, 439–448.

Florian, R. V. (2007). Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural Comput.*, *19*, 1468–1502.

Foster, D. J., Morris, R. G. M., & Dayan, P. (2000). A model of hippocampally dependent navigation, using the temporal difference learning rule. *Hippocampus*, *10*, 1–16.

Froemke, R. C., & Dan, Y. (2002). Spike-timing-dependent synaptic modification induced by natural spike trains. *Nature*, *416*(6879), 433–438.

Gazeres, N., Borg-Graham, L., & Frégnac, Y. (1998). Phenomenological model of visually evoked spike trains in cat geniculate nonlagged X-cells. *Vis. Neurosci.*, *15*, 1157–1174.

Georgopoulos, A., Kalaska, J. F., Caminiti, R., & Massey, J. T. (1982). On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. *J. Neurosci.*, *11*(2), 1527–1537.

Gewaltig, M.-O., & Diesmann, M. (2007). NEST (neural simulation tool). *Scholarpedia*, *2*(4), 1430.

Guerrero-Rivera, R., Morrison, A., Diesmann, M., & Pearce, T. C. (2006). Programmable logic construction kits for hyper real-time neuronal modeling. *Neural Comput.*, *18*, 2651–2679.

Hayon, G., Abeles, M., & Lehmann, D. (2004). Modeling compositionality by dynamic binding of synfire chains. *J. Comput. Neurosci.*, *17*, 179–201.

Houk, J. C., Adams, J. L., & Barto, A. G. (1995). *A model of how the basal ganglia generate and use neural signals that predict reinforcement*. Cambridge, MA: MIT Press.

Izhikevich, E. M. (2007). Solving the distal reward problem through linkage of STDP and dopamine signaling. *Cereb. Cortex*, *17*(10), 2443–2452.

Joel, D., Niv, J., & Ruppin, E. (2002). Actor-critic models of the basal ganglia: New anatomical and computational perspectives. *Neural Networks*, *15*, 535–547.

Kempter, R., Leibold, C., Wagner, H., & van Hemmen, J. (2001). Formation of temporal-feature maps by axonal propagation of synaptic learning. *Proc. Natl. Acad. Sci. USA*, *7*(98), 4166–4171.

Klopf, A. (1986). A drive-reinforcement model of single neuron function. In J. Denker (Ed.), *Neural networks for computing: AIP Conference Proceedings* (Vol. 151, pp. 265–270). New York: American Institute of Physics.

Klopf, A. (1988). A neuronal model of classical conditioning. *Psychobiology*, *16*, 85–125.

Konda, V., & Tsitsiklis, J. (2003). On actor-critic algorithms. *SIAM Journal on Control and Optimization*, *42*(4), 1143–1166.

Kosko, B. (1986). Differential Hebbian learning. In J. Denker (Ed.), *Neural networks for Computing: AIP Conference Proceedings* (Vol. 151, pp. 277–288). New York: American Institute of Physics.

Kossel, A., Bonhoeffer, T., & Boltz, J. (1990). Non-Hebbian synapses in rat visual cortex. *NeuroReport*, *1*(2), 115–118.

Leibold, C., Kempter, R., & van Hemmen, J. (2001). Temporal map formation in the barn owl's brain. *Phys. Rev. Lett.*, *87*(24), 248101.

Leibold, C., & van Hemmen, J. (2002). Mapping time. *Biol. Cybern.*, *87*, 428–439.

Lerchner, A., La Camera, G., & Richmond, B. (2007). Knowing without doing. *Nat. Neurosci.*, *10*(1), 15–17.

Markram, H., Lübke, J., Frotscher, M., & Sakmann, B. (1997). Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. *Science*, *275*, 213–215.

Montague, P., Dayan, P., Person, C., & Sejnowski, T. (1995). Bee foraging in uncertain environments using predictive Hebbian learning. *Nature*, *377*, 725–728.

Montague, P. R., Dayan, P., & Sejowski, T. J. (1996). A framework for mesencephalic dopamine systems based on predictive Hebbian learning. *J. Neurosci.*, *16*(5), 1936–1947.

Morimoto, J., & Doya, K. (2001). Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. *Robotics and Autonomous Systems*, *36*, 37–51.

Morris, G., Nevet, A., Arkadir, D., Vaadia, E., & Bergman, H. (2006). Midbrain dopamine neurons encode decisions for future action. *Nat. Neurosci.*, *9*(8), 1057–1063.

Morrison, A., Diesmann, M., & Gerstner, W. (2008). Phenomenological models of synaptic plasticity based on spike timing. *Biol. Cybern.*, *98*, 459–478.

Muller, E., Buesing, L., Schemmel, J., & Meier, K. (2007). Spike-frequency adapting neural assemblies: Beyond mean adaptation and renewal theories. *Neural Comput.*, *19*, 2958–3010.

Munos, R. (2006). Policy gradient in continuous time. *Journal of Machine Learning Research*, *7*, 771–791.

Niv, Y., Joel, D., Meilijson, I., & Ruppin, E. (2002). Evolution of reinforcement learning in uncertain environments: A simple explanation for complex foraging behaviors. *Adaptive Behavior*, *10*(1), 5–24.

O'Doherty, J. P., Dayan, P., Friston, K., Critchley, H., & Dolan, R. J. (2003). Temporal difference models and reward-related learning in the human brain. *Neuron*, *28*, 329–337.

O'Doherty, J., Dayan, P., Schultz, J., Deichmann, R., Friston, K., & Dolan, R. J. (2004). Dissociable roles of ventral and dorsal striatum in instrumental conditioning. *Science*, *304*, 452–454.

Pessiglione, M., Seymour, B., Flandin, G., Dolan, R., & Frith, C. (2006). Dopamine-dependent prediction errors underpin reward-seeking behaviour in humans. *Nature*, *442*, 1042–1045.

Pfister, J.-P., & Gerstner, W. (2006). Triplets of spikes in a model of spike timing-dependent plasticity. *J. Neurosci.*, *26*, 9673–9682.

Philipp, S., Grübl, A., Meier, K., & Schemmel, J. (2007). Interconnecting VLSI spiking neural networks using isochronous connections. In *Proceedings of IWANN2007* (pp. 471–478). Berlin: Springer.

Porr, B., & Wörgötter, F. (2003). Isotropic sequence order learning. *Neural Comput.*, *15*, 831–864.

Porr, B., & Wörgötter, F. (2007). Learning with relevance: Using a third factor to stabilize Hebbian learning. *Neural Comput.*, *19*(10), 2694–2719.

Potjans, W., Morrison, A., & Diesmann, M. (2007a). Reinforcement learning in an actor-critic spiking network model. *Neuroforum*, *8*(1).

Potjans, W., Morrison, A., & Diesmann, M. (2007b). A spiking neural network model for the actor-critic temporal-difference learning algorithm. In *Proceedings of the 37th SFN Meeting*. San Diego, CA: Society for Neuroscience.

Rao, R. P. N., & Sejnowski, T. J. (2001). Spike-timing-dependent Hebbian plasticity as temporal difference learning. *Neural Comput.*, *13*, 2221–2237.

Reynolds, J. N., & Wickens, J. R. (2002). Dopamine-dependent plasticity of corticostriatal synapses. *Neural Networks*, *15*, 507–521.

Roberts, P. D. (1999). Computational consequences of temporally asymmetric learning rules: I. Differential Hebbian learning. *J. Comput. Neurosci.*, *7*, 235–246.

Schultz, W. (2002). Getting formal with dopamine and reward. *Neuron*, *36*, 241–263.

Schultz, W., Dayan, P., & Montague, P. R. (1997). A neural substrate of prediction and reward. *Science*, *275*, 1593–1599.

Schuman, E., & Madison, D. (1994). Locally distributed synaptic potentiation on the hippocampus. *Science*, *263*, 532–536.

Seung, H. S. (2003). Learning spiking neural networks by reinforcement of stochastic synaptic transmission. *Neuron*, *40*, 1063–1073.

Seymour, B., O'Doherty, J., Dayan, P., Koltzenburg, M., Jones, A., Dolan, R., et al. (2004). Temporal difference models describe higher-order learning in humans. *Nature*, *429*, 664–667.

Suri, R., & Schultz, W. (1999). A neural network model with dopamine-like reinforcement signal that learns a spatial delayed reponse task. *Neuroscience*, *91*(3), 871–890.

Suri, R. E., & Schultz, W. (2001). Temporal difference model reproduces anticipatory neural activity. *Neural Comput.*, *13*, 841–862.

Sutton, R. (1988). Learning to predict by methods of temporal difference. *Machine Learning*, *3*, 9–44.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Tao, H.-z. W., Zhang, L. I., Bi, G.-q., & Poo, M.-m. (2000). Selective presynaptic propagation of long-term potentiation in defined neural networks. *J. Neurosci.*, *20*(9), 3233–3243.

Tesauro, G. (1994). TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Comput.*, *6*(2), 215–219.

Tricomi, E. M., Delgado, M. R., & Fiez, J. A. (2004). Modulation of caudate activity by action contingency. *Neuron*, *41*, 281–292.

VanRullen, R., Guyonneau, R., & Thorpe, S. J. (2005). Spike times make sense. *TINS*, *28*(1), 1–4.

Williams, R. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, *8*, 229–256.

Witten, I. H. (1977). An adaptive optimal controller for discrete-time Markov environments. *Information and Control*, *34*, 286–295.

Wörgötter, F., & Porr, B. (2005). Temporal sequence learning, prediction, and control: A review of different models and their relation to biological mechanisms. *Neural Comput.*, *17*, 245–319.

Xie, X., & Seung, H. S. (2004). Learning in neural networks by reinforcement of irregular spiking. *Phys. Rev. E*, *69*, 41909.

Zhang, L. I., Tao, H. W., Holt, C. E., Harris, W. A., & Poo, M.-m. (1998). A critical window for cooperation and competition among developing retinotectal synapses. *Nature*, *395*, 37–44.

Received August 20, 2007; accepted May 5, 2008.

This article has been cited by:

1. Christoph Kolodziejski, Bernd Porr, Florentin Wörgötter. On the Asymptotic Equivalence Between Differential Hebbian and Temporal Difference LearningOn the Asymptotic Equivalence Between Differential Hebbian and Temporal Difference Learning. *Neural Computation*, ahead of print1-30. [Abstract] [PDF] [PDF Plus]