in: Proc. IJCNN'98 Anchorage, vol. 3, pp.1992-97, IEEE Computer Society Press 1998

A Neural Field Approach to Topological Reinforcement Learning in Continuous Action Spaces

H.-M. Gross, V. Stephan, M. Krabbes
Technical University of Ilmenau, Department of Neuroinformatics
98684 Ilmenau (Thuringia), Germany
homi@informatik.tu-ilmenau.de http://cortex.informatik.tu-ilmenau.de

Abstract— **We present a neural field approach to distributed Q-learning in continuous state and action spaces that is based on action coding and selection in dynamic neural fields. It is, to the best of our knowledge, one of the first attempts that combines the advantages of a topological action coding with a distributed action-value learning in one neural architecture. This combination, supplemented by a neural vector quantization technique for state space clustering, is the basis for a control architecture and learning scheme that meet the demands of reinforcement learning for real-world problems. The experimental results in learning a vision-based docking behavior, a hard delayed reinforcement learning problem, show that the learning process can be successfully accelerated and made robust by this kind of distributed reinforcement learning.**

I. Introduction

A wide class of sequential decision-making problems are those in which the states and actions of a dynamic system must be described using *real-valued variables.* Common examples of these class of problems are found in robotics where agents are required to move manipulators, grasp tools or objects, or navigate to targets or around obstacles. Reinforcement learning (RL) can be used to solve such sequential decision-making problems. The main idea of RL consists in using experiences to progressively learn the optimal value function, which is the function that predicts the best long-term outcome an agent could receive from a given state when it applies a specific action and follows the optimal policy thereafter [9] [10]. The agent can use a RL-algorithm such as Sutton's $TD(\lambda)$ algorithm [9], or Watkins' Q-learning algorithm [11] to improve the long-term estimate of the value function associated with the current state and the selected action. However, in systems having continuous state and action spaces, the value function must operate with real-valued variables representing states and actions. Therefore, the value functions are typically represented by *function approximators*, which use finite resources to represent the value of continuous state-action pairs. Function approximators are useful because they can generalize the expected return of state-action pairs the agent actually experiences to other regions of the state-action-space. In this way, the agent can estimate the expected return of state-action pairs that it has never experienced before. There have been presented many classes of function approximators, each with advantages and disadvantages. The choice of a function approximator depends mainly on how accurate it is in generalizing the values for unexplored state-action pairs, how expensive it is to store in memory, and how well it supports the computation of the one-step search and learning by $TD(\lambda)$ updates [10]. In this paper, we present a novel approach to a state-action function approximator that combines a neural vector quantization technique (Neural Gas [7]) for optimal clustering of a high-dimensional, continuous input space [5] with a neural field approach to topological action coding in continuous action spaces, and a neural implementation of a topologically distributed Q-learning.

Supported by Deutsche Forschungsgemeinschaft, DFG-Grant Gr1378 / 2-1 (SEMRINT-Project), and TMWFK (Gestik-Project)

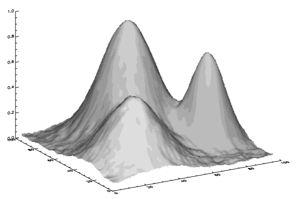
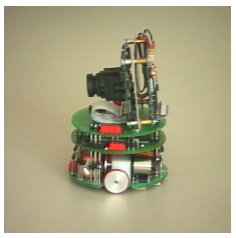

Fig. 1. *Left:* Topographically organized action hypotheses in a 2D neural field that is coding a 2D continuous action space, e.g. for steering angle ϕ (x-dir.) and speed v (y-dir.) *Right:* Experimental platform: mobile miniature-robot Khepera equipped with an on-board color camera (500(H) × 582(V) pixels).

II. The Neural Field Approach to Action Coding and Selection

Dynamic neural fields have been developed as a theoretical concept in computational neuroscience for models of map formation, saccadic motor programming [6], depth perception, etc. This paper proposes the application of the neural field framework to topological action coding, dynamic action selection, and topologically distributed and controlled RL in a two- or higher-dimensional continuous action space (see Fig. 1-left). A neural field can be viewed as a recurrent neural network that receives topographically organized information. The neurons of the field are laterally connected in a way to produce localized clusters of activity (so-called blobs) [1]. Topographically organized action fields are qualified to code alternative action hypotheses simultaneously. This is possible because of the underlying coding principle that transfers the different action hypotheses into spatially coded and separated activity blobs within a two- or higher-dimensional action field. The value and certainty of the action hypotheses are coded in the height and variance of the activity blobs. Figure 1 (left) shows a typical two-dimensional action field that is simultaneously coding three alternative action hypotheses varying in value and certainty. They are localized within the field, for example, by their real-valued *steering angle* ϕ and *speed* v, as Gaussian activity blobs. The neural field used in our model consists of a two-dimensional layer of leaky integrator neurons, whose activations z are changed according to the following differential equation [1]:

$$\tau \frac{d}{dt} z(\underline{\mathbf{r}},t) = -z(\underline{\mathbf{r}},t) - h(t) + x(\underline{\mathbf{r}},t) + \int_R w(\underline{\mathbf{r}} - \underline{\mathbf{r}}')\; S\Big(z(\underline{\mathbf{r}}',t)\Big) d^2\, r' \quad (1)$$

The change of the activation of a neuron at position $\underline{\mathbf{r}}$ in the field is a function of its state $z(\underline{\mathbf{r}},t)$, the global inhibition $h(t)$, the input activity $x(\underline{\mathbf{r}},t)$, and the spatially integrated activity of the neurons in the neighborhood R weighted by the neighborhood function $w(\underline{\mathbf{r}}-\underline{\mathbf{r}}') = w_0 \cdot \exp(-\frac{\|\underline{\mathbf{r}}-\underline{\mathbf{r}}'\|^2}{2\sigma^2}) - H_0$, that is moved in $-y$ direction by a global inhibition H_0. In the following, the main characteristics of neural fields that are of relevance for a topological action coding and selection in RL are summarized:

i) The dimensionality of the field is determined by the dimensionality of the action space. Because of the topological coding and selection principle, it is possible to code any real-valued action in the neural field with just a small number of neurons per dimension. In the same way, the real-valued actions, to carry out, can be easily determined from the center of gravity of the winner-blob. *ii*) Action suggestions, for example, from higher processing levels or from other agents at the same level, can actively modulate the activity distribution of the neural field and influence the action selection dynamics. This is a simple, but very effective method to superimpose the action spaces of several agents in order to coordinate their action selection in a dynamical way. In our implementation, we use an additive superposition of action-value based action suggestions and randomly localized activity blobs to realize a neural field approach to a random exploration strategy.

III. Scenario, Preprocessing, Reward Function

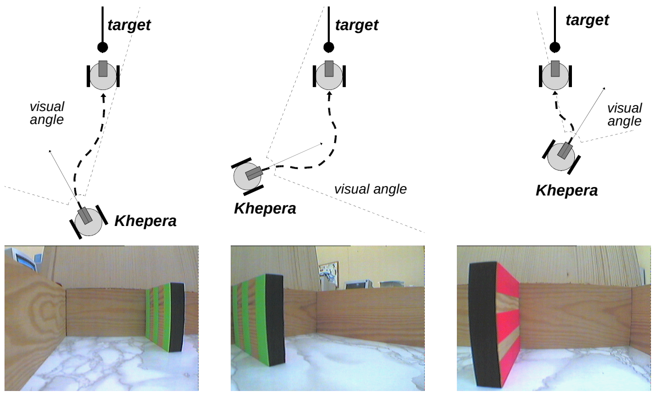

Fig. 2. Real-world scenario to investigate the efficiency of our neural field based reinforcement learning (RL) in the context of a hard delayed RL-problem, a vision-based approaching of the mobile robot KHEPERA to a target object. *Top:* Various situations with different initial positions and the expected docking trajectories. *Bottom:* corresponding views of the target area with the docking station that has a green striped side-wall on the left, a red striped side-wall on the right, and a black front to simplify the image processing.

To investigate the learning dynamics and the efficiency of our neural field based reinforcement learning scheme, we have studied this approach in the context of a delayed RL-problem, a vision-based approaching behavior of a mobile robot to a target object called "docking" station. This is really a hard RL-problem: at the beginning, the robot cannot imagine in which direction and how far the target is, because, a positive reward is received only after the *last* movement, if the robot achieved the docking area successfully. We defined a demanding scenario that requires fine-tuned continuous actions to get to the goal correctly. In this scenario, the robot has to solve two contradictory navigation tasks: on the one hand, it has to move to the docking station as fast as possible, because it passively loses its energy due to a simulated perforation of its "energy tank". On the other hand, it has to carry out a soft docking maneuver to receive a final reward taking into consideration that the docking must occur in the front area of the docking station with a docking angle smaller than φ_{min}. Figure 2 (at the top) shows an illustration of the environment and the expected optimum docking trajectories, Figure 4 presents some correct docking positions that would receive a final reward. The experimental environment of the robot is a rectangular arena of 60×30 cm surrounded by walls. During the learning experiments, the docking station can be moved to any position in the arena (see Fig. 7). By that, we want to avoid that the robot acquires global knowledge about the location of the docking station in the arena that could be used to simplify the learning problem.

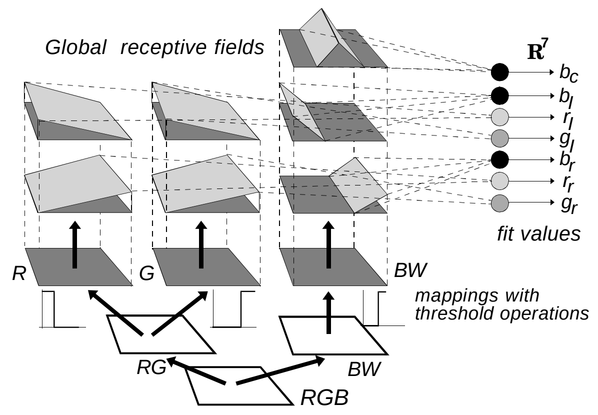

Fig. 3. Visual processing of the RGB-images captured by the on-board camera.

Visual processing: Figure 2 (bottom) presents some typical views of the target area with the docking station captured by the on-board camera. In order to simplify and speed up the image processing, we used opponent colors to mark the front and the side-walls of the docking station in order to make them very easy distinguishable. The input image is first converted into a special neuro-physiologically motivated opponent color space [8] that is formed by a Red-Green(RG)-, Yellow-Blue(YB)-, and Black-White(BW)-dimension (Fig. 3) in order to make the extraction of the target area easy. Then, the image size is reduced to further speed up the image processing. After that, different threshold-operations are applied to the RG- and BW-map of the color space to mark interesting regions in the image. It should be emphasized, that we do not make use of explicit description methods for the position coding of the docking station in the target area, like angle and distance to the target, size of the target, etc. Instead, we use an implicit coding on the basis of receptive fields with characteristic convolution kernels operating globally on the preprocessed maps in the color space (Fig. 3). This way, we achieve a distributed fuzzy coding of the input scene, that provides implicit information about the presence, location, distance, and view of the docking station within the scene. The fit values resulting from these receptive field operations constitute a real-valued input space $\mathbf{R}^7$ (see Fig. 3). Together with the actual amount of energy E, an 8-dimensional continuous input space has to be analyzed in order to learn the desired docking behavior.

Scenario-specific reward function: In our scenario, the robot self-evaluates its actions and determines the following internal rewards, merely on the basis of the fit-values of the visual processing and its present amount of energy: $r(t) = +1.0$ (if the robot achieved the docking area successfully: $(b_c > \theta_{b_1}) \wedge (max(r_l, r_r, g_l, g_r) < \theta_c) \wedge (E > 0)$); $r(t) = 0.0$ (if the robot lost the target from view or lost its energy completely: $(max(b_l, b_c, b_r) < \theta_{b_2}) \vee (E = 0)$); $r(t) = +0.1$ (otherwise). It is to remark, that our reward function does not consider any additional information during the approaching maneuver, like the distance to the goal or the expected docking angle, to speed up the learning process. So, our robot has undoubtedly to solve a delayed RL-problem.

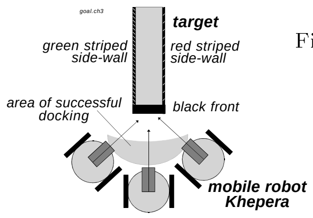

Fig. 4. Docking area with several correct docking positions that can receive a final reward.

IV. Architecture and Learning Algorithm

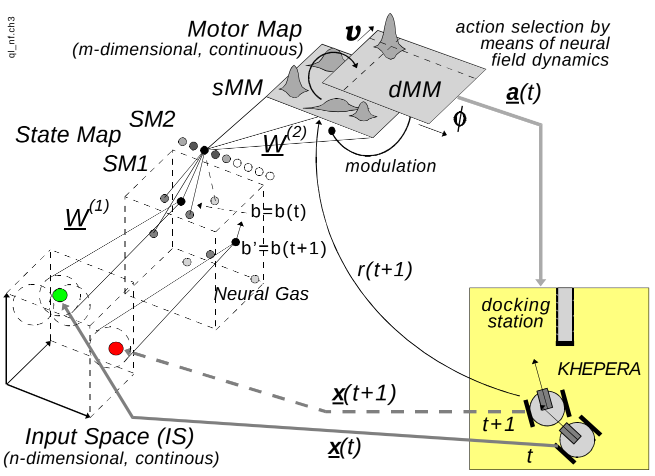

Fig. 5. Neural control architecture for topologically distributed Q-learning embedded in an “Action-Perception-Cycle”.

Based on the original Q-learning algorithm [11] and an earlier neural implementation [5], we developed a neural field approach to distributed Q-learning in continuous state and action spaces that is based on action coding and selection in dynamic neural fields. The neural control architecture (Fig. 5) consists of several neural subsystems for the different subtasks of the RL-based state-action mapping: a double-layered *State Map (SM)* that clusters the n-dimensional continuous *Input Space (IS)* and a double-layered *Motor Map (MM)* that decides, what action is to be selected in this state, and controls the topological RL between *SM* and *MM*. A more detailed explanation of the internal processing and learning steps is given by the following algorithm:

1. Initialize the weights $\underline{\mathbf{W}}^{(1)}$ between the *State Map (SM)* and the *Input Space (IS)* and $\underline{\mathbf{W}}^{(2)}$ between the *Motor Map (MM)* and *SM*.

2. Perceive the current sensory situation $\underline{\mathbf{x}}(t)$.
3. Determine the current state by computing the activity $y_s^{sm1}(t) \in \underline{\mathbf{y}}^{sm1}(t)$ of the neurons s in the first layer of the *State Map (SM1)*, a "Neural Gas" vector quantizer [7], according to a specific neighborhood function that considers similarities in the input space:
$$y_s^{sm1}(t) = e^{-(k_i/\sigma(t))} \tag{2}$$
k_i results of a "neighborhood ranking" of the reference vectors $\underline{\mathbf{w}}_s^{(1)}$ for the current input $\underline{\mathbf{x}}(t)$; $\sigma(t)$ is a time-dependent adaptation range.
4. Compute the activity $y_s^{sm2}(t) \in \underline{\mathbf{y}}^{sm2}(t)$ of the neurons s in *SM2* that project the action-values (Q-values) to the static layer of the *Motor Map (sMM)*. Best results have been achieved by the following interpolating activation that considers the responsibility of all *SM1*-neurons for the current input $\underline{\mathbf{x}}(t)$:
$$y_s^{sm2}(t) = \frac{1/\Big(||\underline{\mathbf{x}}(t) - \underline{\mathbf{w}}_s^{(1)}(t)||\Big)}{\sum_{i\in SM1} 1/\Big(||\underline{\mathbf{x}}(t) - \underline{\mathbf{w}}_i^{(1)}(t)||\Big)} \tag{3}$$
5. Compute the activity $y_{\underline{\mathbf{r}}}^{smm}(t)$ of the neurons $\underline{\mathbf{r}}$ in the static layer of the *Motor Map (sMM)* on the basis of the interpolated Q-value mapping from *SM2* to *sMM*.
$$y_{\underline{\mathbf{r}}}^{smm}(t) = \sum_s w_{\underline{\mathbf{r}}s}^{(2)}(t) \cdot y_s^{sm2}(t) \tag{4}$$
6. Add a randomly positioned Gaussian activity blob $\underline{\mathbf{y}}^{stoch}$ to the activity distribution in *sMM*. This corresponds to a neural field implementation of a random exploration strategy (Boltzmann exploration):
$$\underline{\mathbf{y}}^{smm}(t) := T(t) \cdot \underline{\mathbf{y}}^{stoch} + (1 - T(t)) \cdot \underline{\mathbf{y}}^{smm}(t) \tag{5}$$
T is a control parameter that can be decreased over time to decrease the exploration rate ($T \in [0.0, 1.0]$).
7. Compute the neural field dynamics in the dynamic layer of the *Motor Map (dMM)* and find the winner-blob in $\underline{\mathbf{y}}^{dmm}(t)$ (Fig. 6-right, explanation below).
8. Select the corresponding action $\underline{\mathbf{a}}(t) = f(\underline{\mathbf{y}}^{dmm}(t))$ by determination of the center of gravity within *dMM*.
9. Execute the chosen action $\underline{\mathbf{a}}(t)$, this yields a new situation $\underline{\mathbf{x}}(t+1)$
10. Evaluate the applied action with the immediate reward $r(t+1)$ (see last section).
11. Compute the activity $y_s^{sm1}(t+1)$ of the neurons s in *SM1* for the new sensory situation $\underline{\mathbf{x}}(t+1)$ according to equation (2).
12. Compute the activity $y_s^{sm2}(t+1)$ of the neurons s in *SM2* according to equation (3).
13. Compute the activity $y_{\underline{\mathbf{r}}}^{smm}(t+1)$ of the neurons $\underline{\mathbf{r}}$ in *sMM* according to equation (4).
14. Update the weights $\underline{\mathbf{W}}^{(1)}$ of all neurons s in *SM1* controlled by the activation $y_s^{sm1}(t)$:
$$\begin{aligned}\underline{\mathbf{w}}_s^{(1)}(t+1) &:= \underline{\mathbf{w}}_s^{(1)}(t) + \Delta\underline{\mathbf{w}}_s^{(1)}(t) \\ \Delta\underline{\mathbf{w}}_s^{(1)}(t) &= \eta(t) y_s^{sm1}(t)\Big(\underline{\mathbf{x}}(t) - \underline{\mathbf{w}}_s^{(1)}(t)\Big)\end{aligned} \tag{6}$$
15. Update the weights $\underline{\mathbf{W}}^{(2)}$ (Q-values) between the neurons $\underline{\mathbf{r}}$ of the *dMM*-winner-blob and s in *SM2*:
$$\begin{aligned}w_{\underline{\mathbf{r}}s}^{(2)}(t+1) &:= w_{\underline{\mathbf{r}}s}^{(2)}(t) + \Delta w_{\underline{\mathbf{r}}s}^{(2)}(t) \qquad (7)\\ \Delta w_{\underline{\mathbf{r}}s}^{(2)}(t) &= y_{\underline{\mathbf{r}}}^{dmm}(t) y_s^{sm2}(t) \alpha \Big[r(t+1) + \\ &\qquad \gamma \max_{\underline{\mathbf{p}}} y_{\underline{\mathbf{p}}}^{smm}(t+1) - w_{\underline{\mathbf{r}}s}^{(2)}(t)\Big]\end{aligned}$$
$\underline{\mathbf{y}}^{dmm}(t)$ and $\underline{\mathbf{y}}^{sm2}(t)$ serve as gating functions that control the topological action-value learning between the neurons of the *dMM*-winner-blob and the neighboring *SM2*-neurons representing similar states. α is a constant learning rate (no "freezing") because the agent has to retain its plasticity to cope with a changing environment; γ is the factor discounting future reinforcements ([11]).
16. Switch between time levels:
$\underline{\mathbf{x}}(t) := \underline{\mathbf{x}}(t+1)$; $\underline{\mathbf{y}}^{sm1}(t) := \underline{\mathbf{y}}^{sm1}(t+1)$;
$\underline{\mathbf{y}}^{sm2}(t) := \underline{\mathbf{y}}^{sm2}(t+1)$; $\underline{\mathbf{y}}^{smm}(t) := \underline{\mathbf{y}}^{smm}(t+1)$
17. If $\underline{\mathbf{x}}(t)$ is a final state then terminate else go to **7.**

Action selection by neural field dynamics: For simulation, the differential equation (1) was approximated by the following difference equation:
$$\begin{aligned}z_{\underline{\mathbf{r}}}^{dmm}(k+1) &= (1-\alpha) z_{\underline{\mathbf{r}}}^{dmm}(k) + \alpha\Big(w^I y_{\underline{\mathbf{r}}}^{smm}(t) \\ &\qquad -w_h h_1(k) + w_s \sum_{\underline{\mathbf{r}}'} w_{\underline{\mathbf{r}}\underline{\mathbf{r}}'} y_{\underline{\mathbf{r}}'}^{dmm}(k)\Big) \qquad (8)\\ y_{\underline{\mathbf{r}}}^{dmm}(k+1) &= 1/\Big(1 + e^{-\beta(z_{\underline{\mathbf{r}}}^{dmm}(k+1) - 0.5)}\Big) \\ h_{1/2}(k+1) &= (1-\alpha_{1/2}) h_{1/2}(k) + \alpha_{1/2} \sum_{\underline{\mathbf{r}}'} y_{\underline{\mathbf{r}}'}^{dmm}(k)\end{aligned}$$
In our model, we use different time-scales for the learning process (t) and the relaxation dynamics (k). In this sense, we consider the dynamics in *dMM* to be a sub-dynamics of the learning dynamics on a much faster time-scale. The results of this sub-dynamics are only then transferred to the learning dynamics, if a stable solution has evolved in *dMM* (equation 9). To detect this stable solution, we defined the following simple, but very robust end-detection that is based on the dynamics of two spatially integrating neurons $h_1(k)$ and $h_2(k)$ operating with different time-constants α_1 and α_2:
$$y_{\underline{\mathbf{r}}}^{dmm}(t) = \begin{cases} y_{\underline{\mathbf{r}}}^{dmm}(k)\,, & |h_1(k) - h_2(k)| < \theta_d \\ 0.0\,, & \text{otherwise}\end{cases} \tag{9}$$

Figure 6 demonstrates the selective effect of the neural field dynamics in *dMM*. On the basis of a reliable selection of a winner-blob, a corresponding motor command can be generated by determination of the center of gravity of the activity distribution within the neural field.

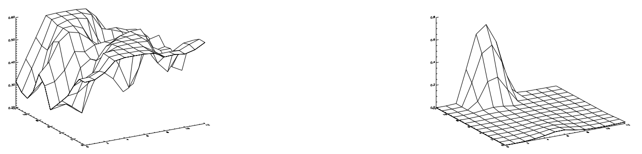

Fig. 6. *Left:* Initial activity distribution in *dMM* as a result of superimposed action-value mappings from *SM2* to *sMM*. *Right:* winner-blob selected by the neural field dynamics within *dMM*: x-direction codes steering angle $\phi \in [-\phi_{max}, +\phi_{max}]$, y-direction codes speed $v \in [0, v_{max}]$.

V. Results of the Learning Experiments

Our approach has been verified by numerous experiments on our mobile robot Khepera in a real-world environment. The experimental results in learning a vision-based approaching and docking behavior are very encouraging. Figure 7 shows typical approaching maneuvers as result of a successful Q-learning. The robot was started at random positions far from the target areas, then it moved directly to the docking station, although, the target was positioned arbitrarily in the arena. The results of the learning dynamics (Fig. 8 - solid line) demonstrate that it is possible to learn such a delayed RL-problem with a number of learning trials that is comparatively small. For example, we did not need more than 500 docking trials with a mean number of 15 steps per trial to learn the docking behavior. Because of the simulated energy loss, the maximum number of steps per trial was set to 20. This small number of real-world

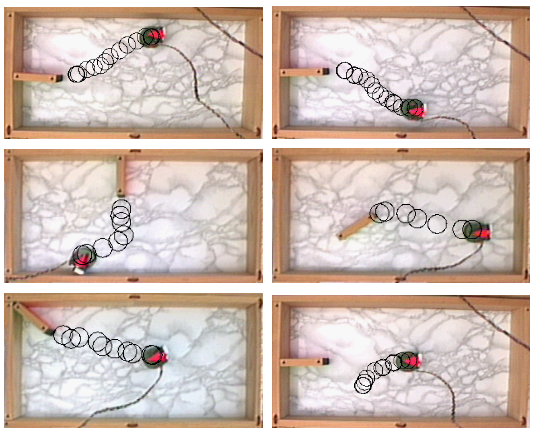

Fig. 7. Temporal traces of successful docking maneuvers to the target located at various positions in the arena (1.-5.) compared to a non-successful docking approach based on a random action selection (bottom right).

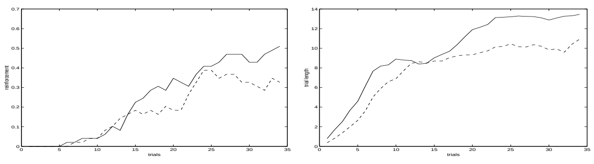

Fig. 8. Temporal evolution of the average reinforcement *(left)* and the mean number of action steps per trial required to achieve the target or done until truncation *(right)* for the neural field based Q-learning (solid line) and the Q-learning model with a discrete action space (dashed line). The diagrams show the mean values of seven independent Q-learning experiments, each of them consists of 35 docking trials.

trials that are necessary to learn the docking behavior is really an encouraging result, since this type of sequential decision-making problems theoretically has exponential time-complexity [2]. Therefore, it is a common practice to modify the learning scheme in such a way that the robot is started in very easy initial situations close to the goal. Later on, the initial situations are shifted into more and more difficult ones. Asada called this simplifying scheme *Learning from Easy Missions (LEM)* [2]. For a couple of reasons (e.g. real-world applicability), we did not apply this kind of biased Q-learning in our experiments. That we nevertheless could achieve such learning results can be just explained by the generalization ability of our topological Q-learning. This learning is localized and controlled by two essential aspects: *i)* the distributed activity coding in the topology preserving *State Map* that is based on the interpolation between the neurons most responsible for the actual input, and *ii)* the topologically organized *Motor Map*, where the "winner-blobs" localize *and* distribute the learning in a local neighborhood. This way, the action value learning within the state-action maps is extended to neighboring regions of the continuous state-action-space. This is not

Start pos.	Action selection	# successf. trials / mean steps	# failed trials / mean steps
A	field	82.6 % / 13.5	17.4 % / 13.1
B	field	97.3 % / 9.8	2.7 % / 6.0
C	field	65.9 % / 14.3	34.1 % / 18.3
A	discrete	65.9 % / 12.0	34.1 % / 12.2
B	discrete	96.0 % / 10.2	4.0 % / 12.1
C	discrete	51.1 % / 14.0	48.9 % / 14.9
B	random	1.9 % / 9.7	98.1 % / 5.2

Fig. 9. Statistical comparison of *i)* average number of successful docking trials after learning and *ii)* mean trial length until success or truncation due to energy loss or target loss for our neural *field* based Q-learning, an approach on the basis of a *discrete* action space (see below), and experiments with random action selection. A, B, C are different starting points.

only of importance for the generalization ability of the function approximator, but it also speeds up the learning significantly. In order to analyze this aspect more in detail, we implemented and investigated a second model with a non-topologically organized, discrete action space (5 different steering angles $\times$ 5 different speed levels). Figure 9 compares the results of the neural field based Q-learning with those ones of the non-topological Q-learning. It compares the success rates of both approaches in the light of differently complex docking maneuvers starting from several initial positions. Because of the generalizing character of the neural field based Q-learning, the success rates of this approach are higher than those ones of the model with the discrete action space. Another point of interest is the learning dynamics of both approaches. Figure 8 shows the evolution of the average reinforcement and the mean number of action steps per trial. The mean trial length of our neural field approach increases faster, since inexpedient actions are avoided earlier because of the generalizing effect of the topological learning in the state-action space. In contrast, the discrete approach cannot extrapolate from single experiences, therefore it must try out much more inexpedient actions in order to learn their consequences. The evolution of the mean reinforcement presents similar results (Fig. 8-left): already after 30 docking trials, our neural field approach shows an obviously higher mean reinforcement. This is a result of a higher rate of successful docking maneuvers. Fig. 10 illustrates the temporal evolution of the action-value maps of three exemplary *SM*-neurons during learning. The corresponding views of the target area are shown at the top of the columns. Finally, it should be emphasized that we have made the robot learn to solve its task without the need of a simulation based initialization of its action-value maps, as it is a common method to speed up the learning.

VI. Conclusion and Outlook

We presented a neural field based approach to distributed RL in continuous state and action spaces. It combines the advantages of a topological action coding and selection in dynamic neural fields with a distributed Q-learning in one neural architecture. The experimental results in learning a vision-based docking behavior, a hard delayed reinforcement learning problem, show that the performance of the learning process can be successfully accelerated and made robust by this kind of topological reinforcement learning. A problem that we have not yet considered is the “state-action deviation problem” [2]. To investigate this problem in the context of our neural field based Q-learning and to analyze the learning dynamics, we plan to adapt our learning procedure according to that of [2]. In this approach, Q-learning occurs only, if the last action caused a state-change.

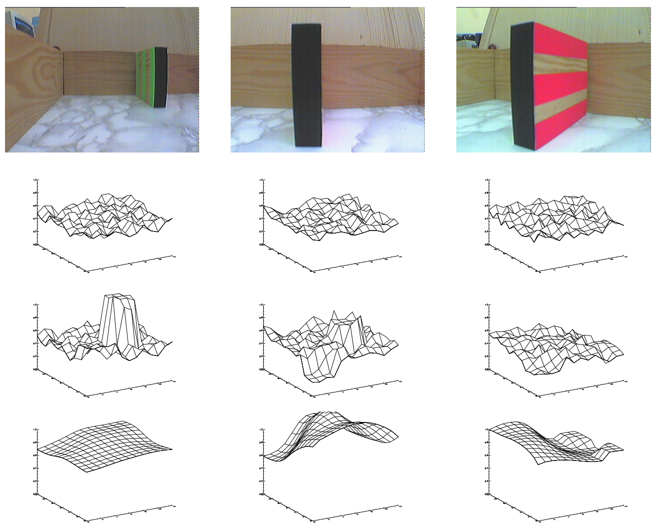

Fig. 10. Temporal evolution of the action-value maps (Q-values) of three exemplarily selected neurons as result of the neural field based distributed Q-learning: *(2. row)* random map-initialization at an optimum ground level, *(3. row)* maps after 20 docking trials, *(bottom row)* final maps that allow a successful docking behavior. *(1. row)* Exemplary views of the target area in different distances to the robot that can optimally activate the selected neurons and their learned action-value maps.

Beyond that, we are planning to extend our neural control architecture by combining incremental vector quantization networks, like the “Growing Neural Gas” [4] or the “Dynamical Cell Structures” [3], with our topological action-value learning. This seems to be a promising approach to a more purposive and task-oriented clustering of the state-action-space.

References

[1] S. Amari. Dynamics of Pattern Formation in Lateral-Inhibition Type Neural Fields, *Biol. Cyb.*, **27** (1977) 77-87.

[2] M. Asada, Sh. Noda, K. Hosoda. Purposive Behavior Acquisition for a Real Robot by Vision-Based Reinforcement Learning, *Machine Learning*, **23** (1996) 2&3, 279-303.

[3] J. Bruske, I. Ahrns, and G. Sommer. Practicing Q-Learning, *Proc. of ESANN'96*, 25-30.

[4] B. Fritzke. A Growing Neural Gas Network Learns Topologies, In *Proc. NIPS*, vol.7, MIT Press 1995.

[5] H.-M. Gross, V. Stephan, H.-J. Böhme. Sensory-based Robot Navigation using Self-organizing Networks and Q-learning, *Proc. WCNN'96*, 94-99, Lawrence Erlb. Publ. 1996.

[6] K. Kopecz, and G. Schöner. Saccadic motor planning by integrating visual information and pre-information on neural dynamic fields, *Biol. Cyb.*, **73** (1995) 49-60.

[7] Th. Martinetz, K. Schulten. A Neural Gas Network learns Topologies, *Proc. ICANN'91*, 397-402, Elsevier 1991.

[8] T. Pomierski, H.-M. Gross. Biological Neural Architectures for Chrom. Adapt. *Proc. ICNN'96*, 734-39, IEEE Press 1996.

[9] R.S. Sutton. Learning to predict by the methods of temporal differences' *Machine Learning*, **3** (1988) 9-44.

[10] J.C. Santamaria, and R.S. Sutton. Experiments with Reinforcement Learning in Problems with Continous State and Action Spaces, *Techn. Report UM-CS-1996-088.*

[11] Ch. Watkins, and P. Dayan, “Q-learning”, *Machine Learning*, **8** (1992) 279-292.