in: Proc. WCNN'96, World Congress on Neural Networks 1996, San Diego, pp. 94-99,
Lawrence Erlbaum Associates, Inc. Publishers 1996

Sensory-based Robot Navigation using Self-organizing Networks and Q-learning *

H.-M. Gross, V. Stephan, H.-J. Boehme

Technical University of Ilmenau, Department of Neuroinformatics
D-98684 Ilmenau (Thuringia), Germany
email: homi@informatik.tu-ilmenau.de

Abstract

We present a rapidly learning neural control architecture for sensory-based navigation of a mobile robot and compare the learning dynamics and the navigation behavior in the context of different implemented network approaches and learning schemes. Our control architecture is a combination of i) alternative vector quantization techniques (Neural gas and Kohonen feature map) for optimal clustering and categorizing of continuous input data spaces and ii) a neural implementation of the Q-learning, a very efficient reinforcement learning method for the choice of the appropriate actions. Our simulation experiments in an artificial environment of changeable geometrical complexity demonstrate that a robot, utilizing this control scheme, can learn the desired behavior rapidly, irrespective of the chosen contradictory navigation tasks. Moreover, we can show that only simultaneous learning schemes develop a kind of 'functional categorizing' of sensory situations. Only they are capable of acquiring knowledge about the sensorial consequences of executed actions from the beginning.

1 Introduction

Reinforcement Learning (RL) refers to a class of learning tasks and algorithms in which the learning system learns an associative mapping π from sensory situations X to appropriate motor actions A by maximizing a scalar external or internal evaluation of its performance. A variety of parametric function approximation methods has been employed so far to solve RL problems practically. These methods have the advantage of being able to generalize beyond the training data and hence give reasonable performance also on unvisited parts of the input space. Among these, neural methods are the most popular ones, especially methods based on adaptive clustering the input space [2], [5]. The advantage of neural networks is to overcome the essential problems: memory requirement for storing all possible situation-action utility values and generalization of sensory situations in continuous input data spaces.

2 Q-learning Based Neural Control Architecture

The global architecture of our neural system is illustrated in Figure 1 (on the left). Two neural subsystems are used: a *Sensory Map (SM or sm in equations)* that codes the sensory-based input state (situation) and a *Motor Map (MM or mm)* that decides what action (movement) should be selected in this state. In our model, the *Sensory Map* gets information from infrared distance sensors of the robot, constituting a n-dimensional *Input space (IS or is)*. Each neuron r of the *Sensory Map* has an associated reference vector $\underline{\mathbf{w}}_r^{sm\text{-}is} \in \mathbf{R}^n$. The reference vectors can be regarded as positions of the corresponding units in input space. In comparison with the real mobile robot *Khepera* [2] - our target platform for the following 'real-world' experiments - the robot in our simulation possesses only three infrared sensors. They are disposed in a somewhat circular fashion (to the left, to the front, to the right) and allow to measure distances only in a short range: 2 to 5 cm, similar to the real *Khepera*.

Both approaches of our *Sensory Map* are based on vector quantization techniques [4]. In the *Kohonen feature map* [3] implemented first as *Sensory Map*, the neurons weights specify clusters that sample the input space such that the point density function of the clusters tends to approximate the probability density function of the input vectors. In addition, the weights are organized such that topologically close neurons are sensitive to inputs that are physically similar. The main principle of the *Neural gas* algorithm [4], we implemented as alternative method for vector quantization of the *Input Space*, is similar but neglects topological relations

*Supported by Deutsche Forschungsgemeinschaft (Gr 1378/1-1, Project SEMINT).

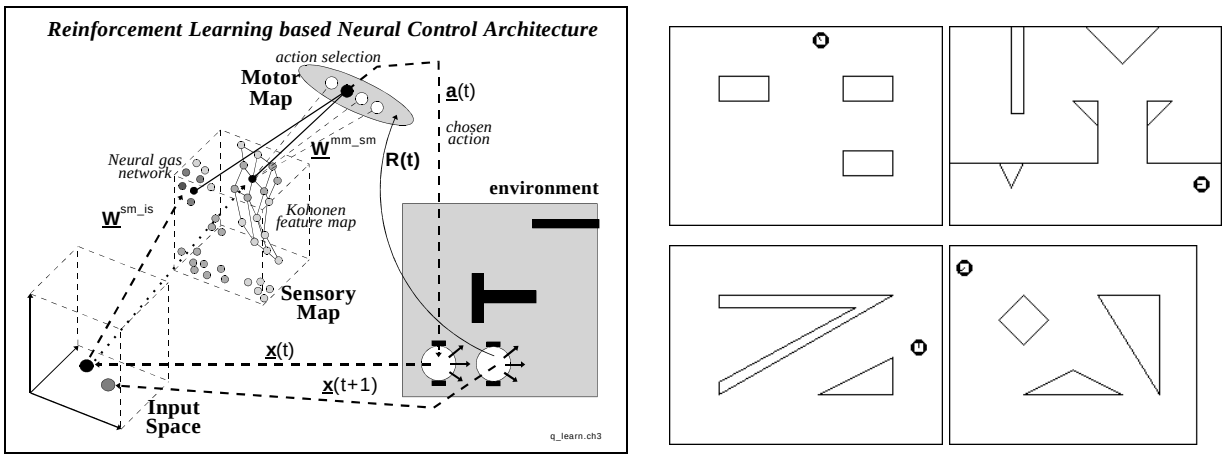

Fig. 1: *(**Left**) The neural control architecture. (**Right**) Some examples of used typical "non-grid"-worlds of variable geometrical complexity composed of walls, obstacles and dead-ends.*

within the *Sensory Map*: For each input signal $\underline{\mathbf{x}}$ only the k nearest centers in the input space are adapted whereby k is decreasing from a large initial to a small final value.
The actions - in this application movements of the robot in its environment - are chosen among m possible ones in the *Motor Map*. Despite the chosen contradictory navigation tasks (moving forward and straight ahead as fast and as long as possible *and* avoiding obstacles) we implemented only four elementary movements: turn left, turn right, go straight ahead, and go backward, each of them a fixed quantity. Which action is chosen in a concrete situation, is the result of a selection process based on reinforcement learning between the *Motor Map* and the *Sensory Map*. The philosophy of this learning process is very simple: from a given sensory situation, if the chosen action caused pain, then the link between the sensory situation in the *Sensory Map* and the chosen action in the *Motor Map* should be inhibited. If it caused pleasure, it should be reinforced. If no pleasure and no pain happened, then it should slightly be reinforced (see equation 5 in paragraph 3). With the appropriate learning rule for adaptation of the weight matrix $\underline{\mathbf{W}}^{mm\text{-}sm}$ connecting the *Memory Map* with the *Sensory Map* (see Fig. 1 (left)) it is possible to avoid making movements that cause pain when a similar situation has been met before. Based on the original Q-learning algorithm of [6], we developed a modified neural implementation of this delayed reinforcement learning algorithm.

3 Learning and Control Algorithm

1. Initialize the weights $\underline{\mathbf{W}}^{sm\text{-}is}$ between *Sensory Map (SM)* (Kohonen feature map or "neural gas"-network) and the $n-$dimensional *Input Space (IS)* with random values and the weights $\underline{\mathbf{W}}^{mm\text{-}sm}$ between *Motor Map (MM)* and *Sensory Map (SM)* with fixed values.

2. Present the current sensory input $\underline{\mathbf{x}}(t) = (x_1, \ldots, x_n)^T$ with $\underline{\mathbf{x}}(t) \in \underline{\mathbf{X}}$

3. Select the best-matching neuron r' in the *Sensory Map (SM)* (neuron with the minimum Euclidean distance between reference weight vector and input vector)

$$\|\underline{\mathbf{w}}_{r'}^{sm\text{-}is}(t) - \underline{\mathbf{x}}(t)\| \quad = \quad \min_r \|\underline{\mathbf{w}}_r^{sm\text{-}is}(t) - \underline{\mathbf{x}}(t)\| \tag{1}$$

4. Compute the activation $y_{rr'}^{sm}(t)$ of the "neighbor neurons" in the *Sensory Map*: $\underline{\mathbf{y}}^{sm}(t) = f(\underline{\mathbf{x}}(t))$. Goal is to enable only a localized learning in the *Sensory Map* (Kohonen feature map) or in the *Input Space* (Neural gas) according to two different "diffusion functions". In this way, the situations 'near' $\underline{\mathbf{x}}$ are modified similarly while the values of states 'far' from $\underline{\mathbf{x}}$ remain unchanged.

$$\textbf{Kohonen map: } y_{rr'}^{sm}(t) = \exp\left(-\frac{\|r - r'\|^2}{2b(t)^2}\right) \qquad \textbf{Neural gas: } y_{rr'}^{sm}(t) = \exp\left(-\frac{k_i}{b(t)}\right) \tag{2}$$

k_i are the result of a "neighborhood ranking" of the reference vectors $\underline{\mathbf{w}}_r$ for the actual input vector $\underline{\mathbf{x}}(t)$ in the input space, with $\underline{\mathbf{w}}_{r'}^{sm\text{-}is} :\rightarrow k_0 = 0$; $b(t)$ is a decreasing adaptation range.

5. Select a suitable action a_i out of a finite set $A = a_1, a_2, a_i, \ldots, a_m$ on the basis of the accumulated weights (representing merits or Q-values for the exploration policy) between the neurons of the *Motor Map* and the best-matching neuron r' of the *Sensory Map*

   - *max selector*: $a_i \Longleftarrow \max_{1 \le i \le m} w_{ir'}^{mm\_sm}(t)$
   - *Controlled stochastic action selector*: a popular stochastic action selector is based on the Boltzmann distribution,

$$P(a_i) \quad = \quad \exp\left(\frac{w_{ir'}^{mm\_sm}(t)}{T(t)}\right) \Big/ \sum_{k=1}^{m} \exp\left(\frac{w_{kr'}^{mm\_sm}(t)}{T(t)}\right) \tag{3}$$

   where T is a nonnegative real parameter (temperature) that controls the stochasticity of the action selector. When $T \to \infty$ all actions have equal probabilities and, when $T \to 0$ the stochastic policy tends towards the greedy policy in the *max selector*. To learn, T is started with a suitable large value and is decreased to values near zero using an annealing rate. This way, exploration takes place at the initial large T values.
   To choose the action, select a random value Z in interval $[0, 1]$

$$a_i = \begin{cases} i = 1 & : & 0 & \le & Z & \le & P(a_1) \\ i = 2 & : & P(a_1) & < & Z & \le & P(a_1) + P(a_2) \\ \vdots & : & & & \vdots & & \\ i = m & : & P(a_1) + \ldots + P(a_{m-1}) & < & Z & \le & 1 \end{cases} \tag{4}$$

6. Execute the chosen action a_i in the environment, this yields a new sensory situation $\underline{\mathbf{x}}(t+1)$.

7. Evaluate the chosen action with an internal or external reinforcement value $R(t)$, for instance in our robotics scenario with $R(t) = R_a(t) + R_b(t)$

$$R_a(t) = \begin{cases} -1.0, & \text{movement led to a collision (pain)} \\ 0.0, & \text{if no collision occurred (pleasure)} \end{cases} \quad R_b(t) = \begin{cases} +0.1, & \text{robot drove straight ahead} \\ -0.8, & \text{robot drove backward} \end{cases} \tag{5}$$

8. Determine the best-matching neuron r'' in *Sensory Map* associated with the new sensory situation $\underline{\mathbf{x}}(t+1)$ after executing a_i in $\underline{\mathbf{x}}(t)$ and select the corresponding Q-values $w_{ir''}^{mm\_sm}(t)$.

9. Update the weights $\underline{\mathbf{w}}^{sm\_is}$ of r' and its neighbors (topological neighbors in *Kohonen map*, k nearest centers in input space in *Neural gas*) towards $\underline{\mathbf{x}}(t)$ controlled by activation $y_{rr'}^{sm}(t)$ to make these neurons more responsive to the last input

$$\begin{aligned} w_{ri}^{sm\_is}(t+1) &= w_{ri}^{sm\_is}(t) + \Delta w_{ri}^{sm\_is}(t) \\ &= w_{ri}^{sm\_is}(t) + \eta(t) \cdot y_{rr'}^{sm}(t)(x_i(t) - w_{ri}^{sm\_is}(t)) \end{aligned} \tag{6}$$

   Decrease learning rate $\eta(t)$ and adaptation range $b(t)$. To prevent an overfitting, we modified the NG-learning rule as follows: Update Neural gas-weights only if $\|\underline{\mathbf{w}}_{r'}^{sm\_is} - \underline{\mathbf{x}}(t)\| > d_{min}$

10. Update the weights $\underline{\mathbf{w}}^{mm\_sm}$ (Q-values) between the motor neurons and the neighborhood of r'

$$w_{ir}^{mm\_sm}(t+1) \quad = \quad w_{ir}^{mm\_sm}(t) + \alpha \Delta w_{ir}^{mm\_sm}(t) \tag{7}$$

$$\text{with} \quad \Delta w_{ir}^{mm\_sm}(t) \quad = \quad y_{rr'}^{sm}(t)\Big(R(t) + \gamma V(t) - w_{ir}^{mm\_sm}(t)\Big) \tag{8}$$

   α - constant learning rate (in contrast to the original Q-learning algorithm) , $V(t)$ - evaluation function

$$V(t) = \begin{cases} \max\limits_{1 \le i \le m} w_{ir''}^{mm\_sm}(t) & : & \text{maximum learning rule - optimistic evaluation} \\ \min\limits_{1 \le i \le m} w_{ir''}^{mm\_sm}(t) & : & \text{minimum learning rule - pessimistic evaluation} \\ 1/m \sum\limits_{i=1}^{m} w_{ir''}^{mm\_sm}(t) & : & \text{averaging learning rule} \end{cases} \tag{9}$$

11. Switch between time levels: $\underline{\mathbf{x}}(t) = \underline{\mathbf{x}}(t+1), \quad r' = r''$

12. If a stopping criterion (e.g., performance measure) is not fulfilled yet go to step 4.

4 Navigating the Robot: Experiments and Simulation Results

In order to illustrate the learning properties of the different versions of the control architecture, we investigated the robot's navigation behavior in an artificial environment, where obstacles of varying complexity can be introduced or removed. Figure 1 (right) illustrates some typical artificial worlds for simulation experiments.

a) General navigation behavior in unknown and known environment

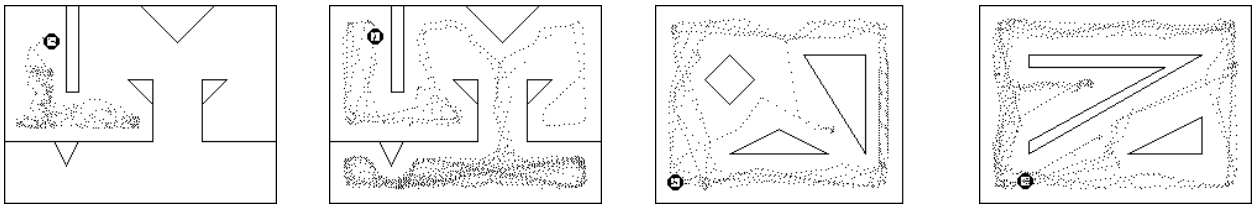

Fig. 2: **1.** *Random trace of motion before learning in the first world.* **2.** *Trace of a typical collision avoiding exploration behavior in a known environment beyond learning.* **3.** *&* **4.** *Significant traces of motion in unknown environments on the basis of the sensorimotor knowledge acquired only in the first world.*

The aim of the first experiment is to demonstrate how our simulated robot learns to solve the conflict between the two contradictory tasks: moving forward and straight ahead as fast and as long as possible and avoiding obstacles by local navigation with the help of an internal reward or punishment (pain and pleasure). Figure 2 illustrates this learning and navigation behavior and shows that unforeseen abilities such as escaping from a dead-end even emerge without explicit planning by the designer.

b) Clustering of input data space

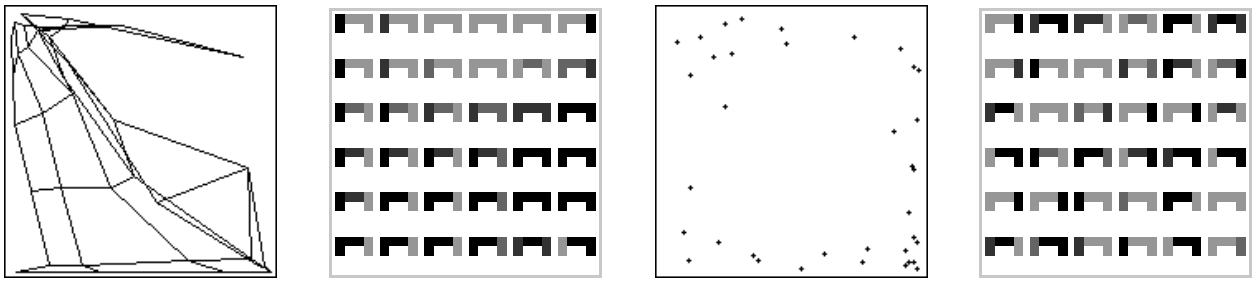

Fig. 3: *(From left to right)* **1.** *Adaptation of the two-dimensional Kohonen feature map and* **3.** *of the "Neural gas" network to the probability distribution of the sensory input data from distance sensors. The positions of the reference vectors in* $\mathbf{R}^3$ *input space are shown as projections onto a* $\mathbf{R}^2$ *subspace after 1000 learning steps (x-direction: distance sensor to the left, y-direction: forward sensor; minimum distances lie in the upper left corner).* **2.** *frame shows the synaptic weights of the neurons in the Kohonen map, the* **4.** *frame of the Neural gas neurons after adaptation, coding all frequently experienced sensory situations. We use three gray-filled small bars for visualization of the synaptic weights per neuron (black: small distance). This corresponds to the arrangement of the distance sensors on the robot (see als Fig. 1).*

Figure 3 illustrates the results of clustering the three-dimensional *Input Space* by the investigated neural vector quantization methods, the *Kohonen feature map* and the *Neural gas network*. Although most of the input data are localized near the boundaries of the *Input Space* (frequently experienced sensory situations), some weights of the Kohonen neurons (first frame from left) are shifted gradually to the center of the *Input Space* due to the neighborhood operations during *SM*-learning. This worsens the quality of the vector quantization, the clustering error increases. In contrast, the Neural gas neurons float in the input space like gas particles (third frame from left). Their weights approximate the probability density function of the input signals better than the Kohonen weights, the clustering error is comparatively small (see also Fig. 4 (left)).

c) Characteristic Q-learning results

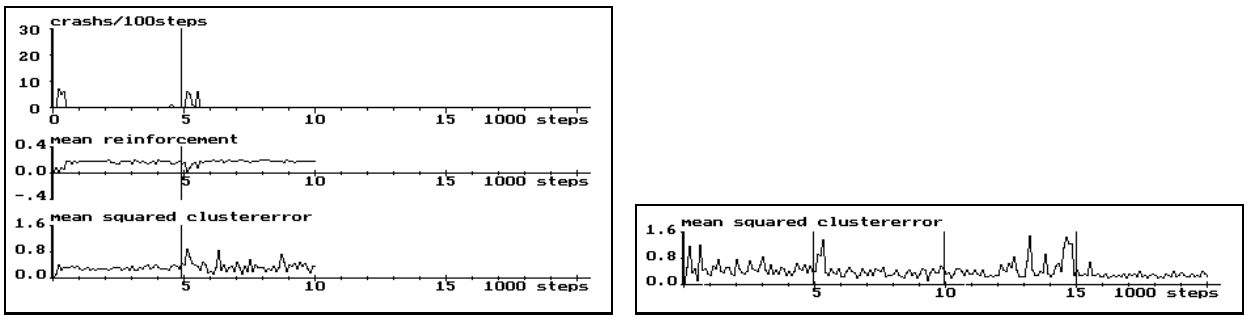

Fig. 4: *(**Left**) Development of Q-learning in intervals of 5.000 steps each. The first 5.000 steps show Neural gas based Q-learning, the second period shows learning on the basis of Kohonen mapping between IS and SM. The number of collisions and the mean reinforcement value behave very similar, but in the second, the Kohonen-period, the clustering error is larger and fluctuates stronger in contrast to the first period. This is expression of a non-optimal clustering of the input space in Kohonen learning. (**Right**) Results of sequential and simultaneous learning schemes for the Sensory and the Motor Map. First 5.000 steps show the development of the mean input space clustering error during sequential learning of SM- and MM-weights with Kohonen based Sensory Mapping. Next 5.000 steps are expression of a simultaneous SM- and MM-learning (with Kohonen map), the third period shows once more a sequential learning (but now with a Neural gas mapping) and the last period a simultaneous learning (with NG). It is to seen that simultaneous learning of SM and MM leads to better results (referring to the clustering error). This results from the fact, that sensory situations, achieved by randomly selected movements during sequential learning, are not adequately to those situations obtained by an immediately controlled navigation during simultaneous learning. The peaks after 2.000 and 12.000 steps are expression of these learning problems.*

Figure 4 (left) shows the time behavior of the sliding average of collisions, of the mean reinforcement value, and of the mean squared clustering error $e_k = 1/100 \sum_{k=t-99}^{t} |\underline{\mathbf{x}}(k) - \underline{\mathbf{w}}_{r'}^{sm-is}(k)|$ in learning intervals of 5.000 steps each. In these experiments we investigated the influence of the different vector quantization techniques to the quality of the Q-learning and to the navigation behavior. Figure 4 (right) illustrates the influence of a phase segregation in organizing the *Sensory Map* and the *Motor Map* on the navigation behavior of the robot. We can show, that only control architectures with simultaneous learning schemes develop a kind of 'functional categorizing' of sensory situations since only they are suited to acquire knowledge about the sensorial consequences of executed actions in the environment from the beginning. Sequential learning is not able to categorize all relevant sensory situations in this action oriented sense. In this case, all presented sensory input data result only from robot movements selected randomly since no sensorimotor knowledge can be acquired during this first phase. Therefore numerous sensory situations are rather hypothetical than of practical relevance for real navigation. Since these situations nevertheless are mapped onto the *Sensory Map* we get only a data-driven categorizing of input space, but not a task-specific or functional one. This is non-optimal for the desired navigation tasks.

Figure 5 (left) illustrates the learning dynamics and the time behavior of describing parameters (average collisions, mean reinforcement and mean clustering error) dependent on the chosen method for action selection (controlled stochastic selection (Boltzmann) and maximum). We studied the influence of the temperature T on the stochasticity of the action selection. The larger the temperature T the longer the robot shows poor navigation behavior (higher collision rate, decreased mean reinforcement values, increased clustering error, slower learning). The maximum selection in any case chooses the action with the largest Q-value in the current sensory situation. This leads to rapid learning and reliable obstacle avoiding behavior but no exploration behavior evolve. Figure 5 (right) finally depicts the navigation behavior when different learning rules (maximum, minimum, average) are chosen. The maximum rule leads to an optimistic navigation behavior, the robot often maneuvers itself in situations not easy to solve. The minimum rule creates pessimistic behavior. The robot takes no risks, it already searches for a way out at the smallest danger of collision. So far, the minimum rule stands for a risk-free learning.

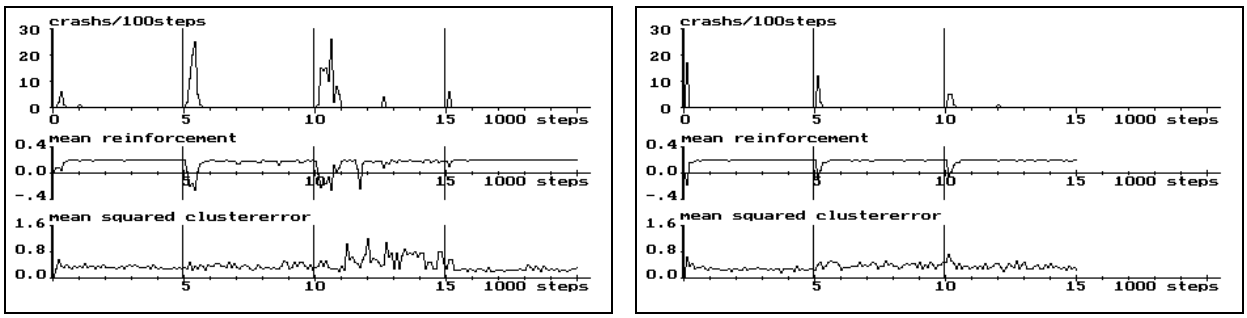

Fig. 5: (**Left**) *Development of Q-learning using different action selection methods: first three periods of 5.000 steps each use the "Boltzmann selection" with different temperatures ($T_0 = 0.5, T_0 = 5.0, T_0 = 10.0$), the fourth period is based on the maximum selection.* (**Right**) *Comparison of the learning behavior in dependence of the implemented evaluation function in the Q-learning rule (maximum, averaging and minimum for 5.000 steps each)*

d) Compatibility between simulated robot and Khepera
Because of the intended compatibility of the control architecture for simulated and real robot navigation we oriented on equivalent sensory signals and movements in the simulator and on the *Khepera* from the beginning. So, we were able to learn much faster in our simulator and to load down the adapted weights onto *Khepera*. It was unexpected that the real *Khepera* was able to navigate without problems in reality only on the basis of knowledge acquired externally on the simulated robot.

5 Conclusion and Future Work

Experiments in learning a navigation behaviour on our simulated robot and on *Khepera* illustrate the efficiency of the different versions of neural control architectures in a situation-action space of considerable size. Altogether, we achieved the best results with the Neural gas based control architecture. Although the results obtained up to now are very promising, it is necessary to investigate the performance of the network for more complex problems than the ones presented here. Another promising direction of research is the combination of incremental neural networks, like the "Growing Neural Gas"-network of FRITZKE [1], with neural reinforcement learning described in this paper. An advantage over the NG method of MARTINETZ is the *incremental character* of the GNG-model which eliminates the need to pre-specify the network size and which allows to implement never-ending learning also in the *Sensory Map*. This way, the whole *SM*- and *MM*-learning process can be continued indefinitely or until an internal (robot-defined) performance criterion is met. The first results of this incremental Q-learning network are very promising and we are further investigating this currently.

References

[1] **Fritzke, B. (1995).** A Growing Neural Gas Network Learns Topologies. In *Advances in Neural Information Processing Systems 7*, MIT Press, Cambridge MA

[2] **Gaussier, Ph. and Zrehen, St. (1993).** Emergence of Behaviors on a Mobile Robot: Learning with Neural Networks. *Proc. of Learning Days in Jerusalem, Heb. University, June 1993*, pp. 1-15

[3] **Kohonen, T. (1982).** Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43: 59-69

[4] **Martinetz, Th.M. et al. (1993).** "Neural-Gas" Network for Vector Quantization and its Application to Time-Series Prediction. *IEEE Trans. on NN*, 4 (1993) 4, pp. 558-569

[5] **Sehad, S. and Touzet, C. (1994).** Self-Organizing Map for Reinforcement Learning: Obstacle-Avoidance with Khepera. *Proc. of PerAc'94 - From Perception to Action*, 420-423, IEEE Press

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