Advances in Complex Systems
Vol. 16, Nos. 2 & 3 (2013) 1350002 (14 pages)

DOI: 10.1142/S0219525913500021

World Scientific
www.worldscientific.com

A GOAL-ORIENTATION FRAMEWORK FOR SELF-ORGANIZING CONTROL

FRANK HESSE* and FLORENTIN WÖRGÖTTER†

Bernstein Center for Computational Neuroscience Göttingen and III. Physics Institute - Biophysics, Georg-August-Universität, Friedrich-Hund-Platz 1, 37077 Göttingen, Germany
**fhesse@physik3.gwdg.de*
†*worgott@physik3.gwdg.de*

Received 21 February 2012
Revised 9 October 2012
Accepted 15 October 2012
Published 13 March 2013

Self-organization, especially in the framework of embodiment in biologically inspired robots, allows the acquisition of behavioral primitives by autonomous robots themselves. However, it is an open question how self-organization of basic motor primitives and goal-orientation can be combined, which is a prerequisite for the usefulness of such systems. In the paper at hand we propose a goal-orientation framework allowing the combination of self-organization and goal-orientation for the control of autonomous robots in a mutually independent fashion. Self-organization based motor primitives are employed to achieve a given goal. This requires less initial knowledge about the properties of robot and environment and increases adaptivity of the overall system. A combination of self-organization and reward-based learning seems thus a promising route for the development of adaptive learning systems.

Keywords: Self-organization; reinforcement learning; autonomous robots.

1. Introduction

Manually preprogrammed (primitive) behaviors, as widely used for the control of autonomous robots, require detailed knowledge about the robotic system and the field of application, especially when considering that the complexity of future devices will further increase. If autonomous robots could acquire a repertoire of behaviors by themselves these efforts could be drastically reduced. Self-organization, especially in the framework of embodiment in biologically inspired robots, offers a promising route to the generation of behavioral repertoires which are adapted to the properties of the robotic device and its environment [20, 13, 16].

*Corresponding author.

An open question is how self-organization of basic motor primitives and goal-orientation can be combined, which is a prerequisite for the usefulness of such systems. This is the key question in guided self-organization [11, 17].

In the work at hand, we will utilize homeokinesis [2] for the generation of behavioral primitives. In previous works it could be shown that the error function used in homeokinetic control can be directly modulated, based on a given reward [11] or context information [6]. Furthermore, the parameters of the controller can be adapted in order to minimize objective functions for self-organization and goal orientation (problem-specific error functions) in parallel [10]. However, in order to not vitiate self-organization or goal-orientation it is important to find the right balance between them, since both depend on the same parameters.

Here we propose a general framework (i.e., also applicable to other controller) for a more independent combination of self-organization and goal orientation. Self-organization will be used to generate body- and environment-related activity in robotic devices. Once this has been achieved pre- and post-processing of sensor values and motor commands, respectively, are used in order to achieve goal-oriented behaviors. This pre- and post-processing framework allows a decoupling of the generation of primitive behaviors and goal orientation. Hence mutual interference of self-organization and goal-orientation parameters is drastically reduced. Furthermore changes in the robot's properties (e.g., by way of defects) can be overcome by the self-organization approach and will have less influence on the goal-oriented behaviors than in the usual case where predefined behaviors are used to achieve a goal.

In the next section, the homeokinetic principle will be shortly described before introducing the pre- and post-processing framework in Sec. 3. Two case studies, with a simulated anthropomorphic hand and a simulated four-wheeled robot, will be presented in Secs. 4 and 5, respectively. Results are summarized and discussed in Sec. 6.

2. The Homeokinetic Principle

Homeokinesis [2, 4] is a general domain invariant principle for the generation of primitive behaviors in autonomous robots based on a dynamical systems formulation [14, 18]. Instead of a designer-provided objective function measuring the distance between the current and a desired behavior, its objective function is derived from the dynamics of the system itself. Based on predictability, in terms of an internal representation of its current behavior, and sensitivity to sensor values this objective aims at smooth and predictable kinetic regimes. Hence, there is no desired behavior, reference value or goal fulfilled by the robot from the point of view of an external observer. From the general principle, simple learning rules for the parameters of a closed-loop robot controller can be derived. Homeokinesis will now be briefly discussed, for more details see e.g., Refs. [3] and [7].

Homeokinetic control is defined by a function K mapping the sensory input $x_t^C \in \mathbb{R}^l$ to the motor values $y^C \in \mathbb{R}^k$: $y^C = K(x^C)$, where all variables refer to time step t and the superscript C indicates that the sensor and motor values are provided to or generated by the controller, respectively. For simplicity x^C, y^C will be used as x, y for the remainder of this section. Sensor values and motor commands are assumed to be in the range $[-1, 1]$. The controller with $K_i(x) = g(z_i)$, $i = 1, \ldots, k$, where $z_i = \sum_j c_{ij} x_j + h_i$, $j = 1, \ldots, l$, is adaptive and depends on a set of adjustable parameters c_{ij} and h_i. In the work at hand $g(\cdot) = \tanh(\cdot)$ is used. An internal model $F : \mathbb{R}^l \times \mathbb{R}^k \to \mathbb{R}^l$ is used to predict the sensory input of the next time step based on the current sensory information and the last motor command: $x_{t+1} = F(x_t, y_t) + \xi_t$. The difference between the predicted and the true sensor value is the modeling error ξ, which is used to train the model F by supervised learning using an online gradient descent. The task of the model is to roughly reflect the momentary relation between sensor values and motor commands. In this study, the internal model is implemented as a linear model with parameters a_{ji}: $x_{t+1,j} = \sum_i a_{ji} y_{t,i} + \xi_{t,j}$, $j = 1, \ldots, l$ and $i = 1, \ldots, k$. Using this equation for the internal model we can write the sensorimotor loop in closed form as $x_{t+1} = \psi(x_t) + \xi_t$, where the dynamics of the system is expressed by the loop function $\psi_j(x_t) = \sum_i a_{ji} K_i(x_t)$, which depends on the parameters of both the model and the controller.

The controller parameters are adapted by an online gradient descent in order to minimize the so called Time Loop Error $E_t = ||x_t - x_t^P||^2$. The reconstructed sensor value x^P is calculated based on the inverse of the loop function: $x_t^P = \psi^{-1}(x_{t+1})$. Note, since ψ is not always invertible, x_t^P cannot be exactly calculated in all cases, which can be overcome by a convenient regularization. The difference between x_t and x_t^P is thus the error arising in a time loop. Using gradient descent with a learning rate ε, the parameters of the controller follow the dynamics $\Delta c_t = -\varepsilon \frac{\partial E_t}{\partial c_t}(x_t, c_t)$, $\Delta h_t = -\varepsilon \frac{\partial E_t}{\partial h_t}(x_t, h_t)$. These parameter dynamics do not accomplish a learned system in the sense of a final constellation of parameters, instead the dynamics of model and controller parameters are essential for the behavior of the robot. Minimizing E is on the one hand seen to increase the sensitivity to the sensor values, which is the source of activity and explorative behaviors in the controlled systems. On the other hand, the exploration is moderated with respect to smoothness of the generated behaviors, since E is also small if the prediction error is small. The generated behaviors are thus a compromise between the two opposing goals: sensitivity and predictability.

3. Pre- and Post-Processing Framework

The paper at hand proposes a possibility to generate goal-oriented behaviors based on self-organized, and hence goal-free, systems. The goal orientation is realized by a pre- and post-processing framework. Basic motor primitives generated by a controller can be modified in order to achieve a given goal. The basic idea is that

the "wires" between robot and controller are cut and the pre- and post-processing are plugged in. Since only the sensor values and motor commands but no internal parameters of the controller are modulated the principle can be applied to basically every controller. Eventually, the pre- and post-processing has to be provided with some context information about the goal, e.g., the direction to the goal.

Let us consider a robot that provides at each instant of time, $t = 0, 1, \ldots$, a vector of sensor values $x_t^R \in \mathbb{R}^l$. The robot is controlled by the motor values $y_t^R \in \mathbb{R}^k$. The superscript R indicates that the sensor and motor values are gathered/executed by the robot. The controller generates in each time step a vector of motor commands $y_t^C \in \mathbb{R}^k$ based on the sensory input $x_t^C \in \mathbb{R}^l$. We assume the same number of sensor and motor values at the robot and the controller and a direct connection from x_i^R to x_i^C and y_i^C to y_i^R.

The post processing is a mapping $M : \mathbb{R}^k \to \mathbb{R}^k$, which depends on the parameter vectors $p, q \in \mathbb{R}^k$ (elements of p being larger than zero) defining a linear mapping of the motor command before it is executed by the robotic system: $y^R = M(y^C)$ with

$$y_{t,i}^R = p_i y_{t,i}^C + q_i, \quad p_i > 0 \text{ for all actuators } i. \tag{1}$$

The parameters p allow a scaling of the amplitude of the motor command, while q defines a shift of the center of motion of the self-organizing control for the respective degree of freedom.

The task of the pre-processing is to compensate the modulation of the post-processing in the sensor space, where an additional factor, the response strength of the sensors, has to be considered. The pre-processing is a mapping $N : \mathbb{R}^l \to \mathbb{R}^l$ with:

$$x_{t,i}^C = \sum_{s=0}^{k} \left(v_{is} \frac{x_{t,i}^R - \sum_{j=0}^{k} b_{ij} q_i}{p_s} \right), \quad i = 1, \ldots, l, \tag{2}$$

where $v_{is} = \frac{y_{t-1,s}^C p_s b_{is}}{\sum_{j=0}^{k} y_{t-1,j}^C p_j b_{ij}}$ reflects the relative contribution of the post-processed motor commands $y_{t-1,s}^C$ to the sensor value $x_{t,i}^R$, which is required to compensate the scaling parameters p_s of the post-processing properly. p and q are known from Eq. (1) and b is a matrix representing the response strength of the sensors to the corresponding motors. The latter can be obtained from a model $G : \mathbb{R}^l \times \mathbb{R}^k \to \mathbb{R}^l$ mapping sensor values and motor command at time t to the sensor values at time $t+1$ with $x_{t+1}^R = G(x_t^R, y_t^R) + \zeta$. The difference ζ between actual and predicted sensor value can be used to train G by supervised learning with an online gradient descent. G is similar to the internal model F introduced in Sec. 2. However, F operates on the sensor values x^C and motor commands y^C delivered to and obtained from the controller, while G operates on the sensor values x^R and motor commands y^R directly received from and sent to the robot. With this in mind and also for generality (e.g., using a different controller instead of homeokinetic control) separate

models are used. G can already be adapted when the pre- and post-processing are not active, e.g., when the self-organizing control is in its initial phase.

Assuming an ideal sensor $x^R_{t+1,0}$ measuring a sum of the results of immediately executed motor commands $y^R_{t,0}$ and $y^R_{t,1}$ with a response strength of e.g., 0.8 and 0.9, respectively ($x^R_{t+1,0} = 0.8y^R_{t,0} + 0.9y^R_{t,1}$). From Eqs. (1) and (2) we find

$$\begin{aligned} x^C_{t+1,0} &= \sum_{s=0}^{1} \left(v_{0s} \frac{x^R_{t+1,0} - \sum_{j=0}^{1} b_{0j} q_j}{p_s} \right) \text{ with } v_{0s} = \frac{y^C_{t,s} p_s b_{0s}}{\sum_{j=0}^{1} y^C_{t,j} p_j b_{0j}} \\ &= \sum_{s=0}^{1} \left(\frac{y^C_{t,s} p_s b_{0s}}{\sum_{j=0}^{1} y^C_{t,j} p_j b_{0j}} \frac{0.8(p_0 y^C_{t,0} + q_0) + 0.9(p_1 y^C_{t,1} + q_1) - \sum_{j=0}^{1} b_{0j} q_j}{p_s} \right) \\ &\quad \text{and assuming } b_{00} = 0.8 \text{ and } b_{01} = 0.9 \text{ we get} \\ &= b_{00} y^C_{t,0} + b_{01} y^C_{t,1}. \end{aligned}$$

Hence, with a conveniently learned b, there is no influence caused by the pre- and post-processing from the point of view of the controller.

If there is no direct relation between sensory signals and motor commands a more complex predictor might be necessary. However, in the reminder of this work we used an even simpler predictor, assuming that sensor x^R_i is measuring a quantity controlled by motor y^R_i, for $i = 0, \ldots, l$, and that this wiring is known beforehand. Then Eq. (2) simplifies to

$$x^C_i = \frac{x^R_i - b_i q_i}{p_i}, \quad i = 1, \ldots, l, \tag{3}$$

where b is a vector now, representing the response strength of a sensor to the corresponding motor.

The parameters p, q of the pre- and post-processing can be set by hand or adapted by a learning paradigm as shown in two case studies, a simulated artificial hand Sec. 4, and a four wheeled robot Sec. 5, respectively.

4. Case Study 1: Guiding Object Manipulation of an Anthropomorphic Hand

Self-organized finger movements of a simulated anthropomorphic hand show interesting behaviors with respect to object manipulation and rich tactile information can be gathered from such manipulations. However, objects need to be grasped or placed in the hand, e.g., dropped onto the palm of the horizontal (open) hand. With a goal-free control of the fingers this becomes difficult, since there is no preference for an open hand. Hence, objects are often placed on the moving fingers and slide down without actually being manipulated. The framework proposed here allows to open the hand temporarily in order to properly place objects in the hand, but does not disturb the self-organizing controller.

4.1. *Setup*

We programmed a model of a human hand with 15 degrees of freedom in a physics simulation environment [12], see Fig. 1. All joints are controlled by bidirectional motors that mimic the interplay between flexor and extensor muscles. The effect of a motor action is measured by angular position sensors, which serve as input to the homeokinetic controller. The controller perceives sensor values and generates a new motor command every 0.1 s. The learning rate is set to $\varepsilon = 0.01$. Controller outputs are averaged over 20 time steps to favor lower-frequency behaviors, as these are more appropriate for object manipulation.

With this setup vivid movements of the fingers are generated which can be interpreted as an exploration of the dynamical range. If an object is present in the hand, the fingers show object manipulation sequences where different ways of manipulation are generated and tactile information can be explored. However, dropping a new object in the hand does rarely work, because the homeokinetic control has no preference for opening the hand. So we introduce the proposed framework in order to open the hand temporarily before a new object is dropped onto the palm. A sensor indicating the presence of an object in the hand is available to the pre- and post-processing to trigger the opening. During normal movement the parameters of the pre- and post-processing are manually set to $p_i = 1$ and $q_i = 0$, $i = 1, \ldots, 15$, causing no change compared to a system without this framework. When no object is present in the hand and a new object should be dropped in the hand, the parameters change to $p_i = 0.4$ and the bias to $q_i = -0.4$ using a smooth ramp function. This decreases the range of the joint movements and shifts the center of the movements toward an open hand posture. If an object is present in the hand again, the parameters change back to the initial values $p_i = 1$ and $q_i = 0$, $i = 1, \ldots, 15$ (using a ramp function). The vector b representing the response strength of the sensors (see Sec. 3) is obtained by supervised learning of a simple linear predictor $G(x_t^R, y_t^R) = by_t^R$. The adaptation of this predictor and the adaptation of the homeokinetic controller going on during the whole time of the experiment.

4.2. *Results*

The system with the proposed framework was tested by dropping objects onto the palm and we compared the number of successfully placed objects to a pure homeokinetic system. After the homeokinetic control was settled, new objects were dropped from above the palm as soon as there was no object in the hand. Ten objects were dropped in one trial. For each setup (with/without framework) ten trials were executed.

Using homeokinetic control in the mean 5.5 out of 10 objects could be successfully placed onto the palm, as shown in Fig. 1. In the other cases at least one finger blocked the path so that the object could not be properly placed. With the additional framework in the mean 9.8 out of 10 objects could be successfully dropped onto the palm.

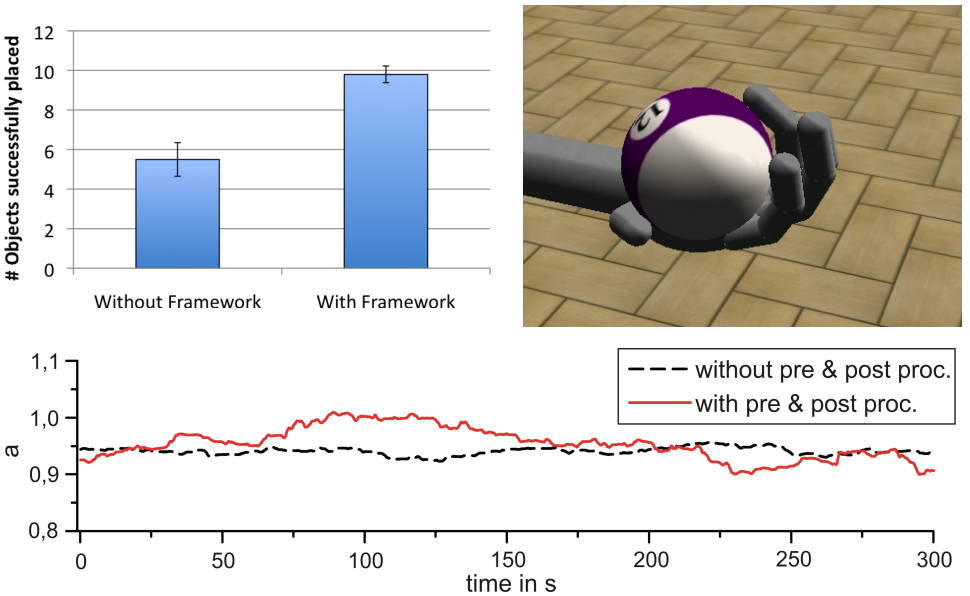

Fig. 1. (Color online) Top left: Number of successfully placed objects in the mean over ten trials with and without the proposed framework. Ten objects were dropped in each trial. Error bars represent the standard deviation. Top right: Simulated anthropomorphic hand manipulating a spherical object. Bottom: Representative example (distal interphalangeal joint of index finger) of a parameter of the internal model before and during usage of the pre- and post-processing framework.

To check the influence of the pre- and post-processing on the controller the parameters a of the internal model, representing the relations of sensor values (x^C) and motor commands (y^C) from the point of view of the homeokinetic system, were investigated. The parameter development three minutes before activation of the pre- and post-processing and three minutes after activation, was compared. To minimize disturbances of the model, objects were not dropped onto the palm during this time. It could be observed that the mean values of the model parameters stayed the same while the standard deviation increases, see Fig. 1 for a representative example. Hence, there is only small influence on the self-organizing control in the current realization. The self-organizing controller still generates coordinated movements of the controlled degrees of freedom which appear and decay, showing that the exploration of the behavioral repertoire of the robotic system still continues like without using the proposed framework. Given the simple way b (the model parameter of the pre- and post-processing, representing the relations of the sensor values (x^R) and the motor commands (y^R) from the point of view of the robot) was obtained, less influential setups are possible when employing more sophisticated predictors $G(x_t^R, y_t^R)$. Note that, changes in the model parameters can also occur if the behavior of the self-organizing system changes over time (e.g., coordination of degrees of freedom, resulting in decreased correlation (as obtained from the internal

model) between motor and attached sensor, while the correlation with a not connected sensor increases). These cases were not considered here. Note furthermore that the influence on the controller arises only from changes of the parameters q. Modulations caused by the scaling parameters p can be completely counterbalanced by the pre-processing because there is no dependence on the response factor b.

These results show that the proposed framework is able to guide the behavior of the self-organizing system, while only slightly influencing the latter, as shown by way of internal model parameters. Hence, the object manipulation is not degraded.

5. Case Study 2: Goal-Orientation for a Wheeled Robot

The proposed mechanism for the generation of goal-oriented behaviors based on self-organized systems will be investigated for the case of a four-wheeled robot in this section. The parameters of the pre- and post-processing can be predefined as described in the previous section. It is also possible to adapt them online. In this case study an example of the latter case will be shown, where reinforcement learning [23, 19] acts on the parameters of the proposed framework in order to guide the self-organized system in a navigation task with a simulated four-wheeled robot.

5.1. *Setup*

Experiments were conducted in the physics simulation environment already used in the previous experiment [12]. The used robot has a capsule as body (length 0.4 units) with four wheels attached, see Fig. 2. The environment consists of a square arena (side length 32 units) with a wall as boundary and four positions defining a sequence of goals for the goal-oriented behavior of the robot, see e.g., Fig. 3. The wheel velocities are controlled independently according to the motor command $y_t^R \in \mathbb{R}^4$. The vector of sensor values $x_t^R \in \mathbb{R}^5$ consists of four measured wheel velocities $(x_0^R, \ldots, x_3^R)$ and the angle between forward direction of motion and goal location (x_4^R). The angle was preprocessed to start with 0° in front and go up to an absolute value of 180° at the rear of the robot, on the right with positive and on the left with negative sign, all with respect to the forward direction, see Fig. 2(left).

Each wheel of the robot is controlled by an individual homeokinetic controller. Hence a coordination of the wheels, which is required for locomotion (at least of the two wheels on one side), can only occur through the environment. The learning rate is set to $\varepsilon = 0.1$ and the bias is set to $h_i = 0$ to favor continuous locomotion. Note that, the probability for the generation of forward or backward movement by the self-organizing controller is still equal. The controller perceives sensor values and issues a new motor command every 0.1 s.

Reinforcement learning is used for the parameter adaptation of pre- and post-processing. In this study, Q-Learning [22, 21] was chosen in order to emphasize the framework and not the details of a complex learning mechanism. Q-learning is a unifying algorithm for simultaneous value function and policy optimization. It has

been widely used for the control of autonomous robots [15, 1, 5, 8, 9] since no robot–world interaction model is required for behavior generation. Here the simplest form, one-step Q-learning, is used. It is defined by

$$Q_t(s,a) = \begin{cases} (1-\beta_t)Q_{t-1}(s,a) & \text{if } s = s_t \text{ and } a = a_t \\ \quad +\beta_t(r_t) + \gamma \max_a(Q_{t-1}(s_{t+1},a)) & \\ Q_{t-1}(s,a) & \text{otherwise,} \end{cases}$$

where $Q_t(s,a)$ is an element of the learned action-value table, storing the utility of executing the action a in the situation s; r being the reward for the executed action; β and γ being the learning rate and discount factor, both positive and smaller than one; t indicates the current time step. Since the reward for an executed action in a given state is stored in the corresponding Q-value and, in dependence on the discount factor, further propagated through the Q-table a "utility landscape" is generated. Selecting actions in order to ascend this landscape with a greedy behavior policy $\pi(s) = \arg\max_a Q(s,a)$ leads to the achievement of the goal (defined by a positive reward), if the Q-table was properly learned. To ensure this, usually an exploration rate is introduced, defining a probability for the execution of a randomly selected action in a given state, in order to visit all state action pairs, not only those receiving high utility values during the first updates.

The sensory information provided by the robot is used for the calculation of the current state and the reward of the Q-learner. The learner has six states, defined by the direction of motion (measured wheel velocities) and the angle α ($\alpha = x^{R_4}$) between the robot's forward direction and the direction to the goal:

s_0: goal at the left side ($-175° < \alpha < -5°$),
s_1: goal directly in front ($-5° < \alpha < 5°$) and moving forward,
s_2:goal directly in front ($-5° < \alpha < 5°$) and moving backward,
s_3: goal at the right side ($5° < \alpha < 175°$),
s_4: goal directly behind ($\alpha > 175°$ or $\alpha < -175°$) and moving forward,
s_5: goal directly behind ($\alpha > 175°$ or $\alpha < -175°$) and moving backward,

see also Fig. 2 (left). Actions of the Q-Learner are braking and not braking the wheels, realized as parameter settings of the pre- and post-processing ($p^i = 0.5$ or $p^i = 1$). Hence, in every state 16 actions are possible, reflecting all combinations of the two possible actions per wheel. In each time step the reward r_t is defined as $r_t = -1$ if the goal moves to a side ($\alpha_t < -6°$ and $\alpha_t < \alpha_{t-1}$, or $\alpha_t > 6°$ and $\alpha_t > \alpha_{t-1}$) and $r_t = 0$ otherwise. An additional penalty of -0.3 is added in each time step in which a brake action is used.

With this setup and the discount factor $\gamma = 0.9$, an exploration rate of 0.5 and the learning rate $\beta = 0.1$ the experimental runs were started. During the first minute the goal-oriented control was not active ($p_i = 1$ for all i), to allow the generation of activity by the self-organizing control. During the next five minutes the Q-learner was updated (five times per second). Every 50 s (or whenever the currently active goal was reached) a new goal was activated. In the last two-and-a-half minutes of

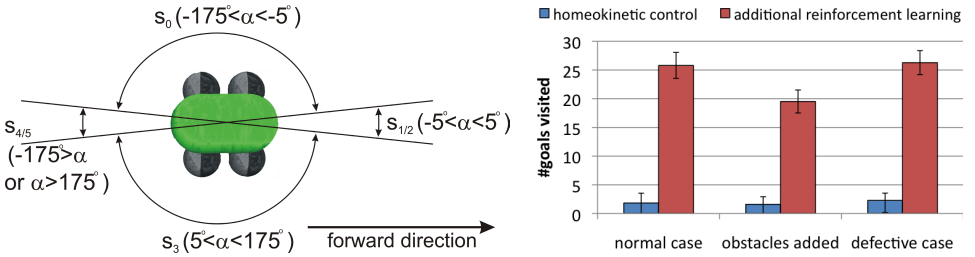

Fig. 2. (Color online) Left: The states of the Q-Learner indicated in a top view of the robot. Right: Mean and standard deviation of the number of visited goals with and without the proposed framework in 25 trials of ten minutes each.

the learning period the learning and exploration rate were linearly decreased to zero and kept zero. Hence, from this point on only a greedy action selection was used for the remainder of the experimental run. The adaptation processes of the slef-organizing controller and the model of the pre- and post-processing are active during the whole time of the experiment.

One experimental run consists of adaptation (only homeokinetic control active), learning (Q-Learning active) and the execution of the learned task (Q-Learner with greedy behavior policy, no learning) in three conditions. The first task (*normal* case) is to let the robot follow a sequence of goal positions for 10 min. A new goal is activated when the currently active goal is reached. The total number of visited goals is counted. The second task (*obstacle* case) is like the first, but obstacles are added on the paths between the goals. The third task (*defective* case) is also like the first, but just before the execution of the task (the Q-Learner has learned already) the robot's properties are manipulated. The sense of direction of one motor is inverted, mimicking a system defect. Hence, in order to drive straight, the motor command y^R has to consist of three values with the same sign and one value with opposite sign. For both controller setups (with/without proposed framework) 25 experimental runs were conducted.

5.2. *Results*

During the adaptation process at the beginning of the experiments homeokinesis was able to generate active behaviors of the robot. This requires coordination of the motor commands since at least two wheels on one side of the robot have to have motor commands with the same sign to bring the robot into motion. The ability to coordinate different wheels is a result of the sensitivity to sensor values, generated by the homeokinetic principle.

After learning, the Q-learners were able to set the parameters of the pre- and post-processing framework properly in order to achieve goal-orientation. The numbers of visited goals for the three conditions for homeokinetic control with and without the proposed framework are shown in Fig. 2. In the *normal* case homeokinetic control visited 1.8 goals in the mean, since goals are only visited by chance.

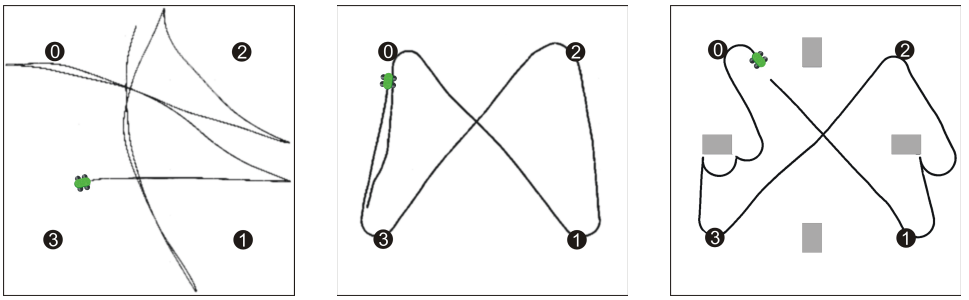

Fig. 3. (Color online) Traces of the robot, black dots represent the sequence of goals: 0, 1, 2, 3, 0 and so on (robot size scaled up for visibility). Left: pure homeokinetic control in *normal* case, Center: homeokinetic control with proposed framework in *normal* case and (right) in case with obstacles added.

It is in the nature of the homeokinetic approach that it is not focusing on the goal (and in this case can not even perceive it), which leads to the low number of 1.8 visited goals in the mean in the *normal* case. Even though the proposed pre- and post-processing framework is based on homeokinetic control, it clearly shows goal-oriented behavior which manifests in 25.8 goals visited in the mean. With additional obstacles on the path between the goals (*obstacle* case) the mean number of visited goals drops slightly. The homeokinetic control reacts to collision situations by inverting the direction of motion of the robot (see Fig. 3). However, with the proposed framework still 19.5 goals could be visited, with pure homeokinetic control 1.6 goals. In the *defective* case both systems visit in the mean nearly the same number of goals as in the *normal* case (2.3 and 26.2, respectively), since the underlying self-organization process can adapt the motor command to the "defect" wheel in order to keep the system active. This is caused by the internal parameter dynamics of the homeokinetic controller.

Sample traces of the robot controlled with and without the proposed framework in the *normal* case and with added obstacles are depicted in Fig. 3. For the proposed framework deviations from an ideal trajectory (e.g., straight lines to the goal) can be clearly seen. These are caused by the dynamics of the self-organizing control and the sensory noise (added in all simulations to achieve more realistic sensor readings). However, the robot shows very strong goal directed behavior compared to pure homeokinetic control. The results show that, by combining self-organization and learning paradigms, robust and adaptive (with respect to changes of the robot and in the environment), body and environment related systems with goal-oriented behavior can be achieved.

6. Conclusion

Self-organization allows the adaptation of a controller to the properties of a given system. This way body- and environment related basic behaviors can be generated without requiring detailed knowledge about the robot or its environment. However,

desired behaviors from the point of view of an external observer can usually not be achieved (over longer timescales), since a treatment of desired values is missing.

This paper proposes a framework for the guidance of self-organized basic motor primitives in order to achieve a goal. In the first case study it was shown that guidance of an anthropomorphic hand controlled by the homeokinetic principle is possible. In order to drop objects onto the palm the hand was temporarily opened by appropriately setting the parameters of the framework. Even a linear model, as part of the pre- and post-processing, can thereby provide the necessary information about the response strength of the sensor. However, the latter is only required if, additionally to the scaling, also the center of the joint motion is modulated. In the second case study goal navigation in a self-organization based four wheeled robot could be achieved by learning the framework parameters using Q-Learning. The learned system showed adaptability to changes of the robot (*defective* case) and the environment (*obstacle* case).

The (successful) adaptation to changes is of cause limited to scenarios which the self-organization paradigm, or the used controller in general, can handle. However, if for example pure Q-Learning with manually predefined actions should be applied to the setup used here, the robot would with high probability already get stuck at the walls in the beginning of the learning period and not learn a proper Q-table, except if additional states for these situations are introduced (and additional sensor values provided to the Q-Learner). Two short thought experiment using Q-Learning without self-organizing control will make this more clear. The task stays the same: obstacle avoidance and goal-navigation with a four-wheeled robot. In a first scenario six predefined behavioral primitives (move straight ahead, move straight back, turn left and turn right while driving forwards, and turn left and turn right while driving backwards) are used. Compared to the state-action space using the pre- and post-processing framework, at least one additional velocity-based sensor/state is required to allow the system to cope with obstacles. The resulting state-action space would have 56 elements (7 states $\times$ 6 actions). Such a system should have a bit shorter learning time than the pre- and post-processing framework (which has a state-action space with 96 elements), but requires pre-knowledge of the actuation system and can not deal with the *defective* case, even not when the Q-Learner is relearned. The predefined primitives simply do not work anymore in the *defective* case. In a second scenario the motor signals are directly used as actions of the Q-Learner. This would drastically increase the size of the state-action space. Assuming four action per wheel (low and high forward and backward velocity) results in 256 actions. Taking only the seven states from the previous scenario (and not all sensors seperately) generates a state-action space with 1792 elements. In this scenario, learning would take much longer than with the proposed method. Furthermore, the *defective* case can only be solved by re-learning the Q-Learner, because a different action has to be chosen for the defect wheel. The proposed mechanism can deal with the *defective* case without re-learning of the Q-Learner, because the self-organizing controller with its ongoing adaptation process solves this

problem following its intrinsic objective function. Using a controller with predefined primitives, able to go straight and reverse its velocity when it hits an obstacle, instead of the homeokinetic controller in this framework, it would be possible to achieve the goal-orientation, but the adaptation to the *defective* case is not possible, since the internal adaptation process of the self-organizing controller is missing.

Compared to existing methods of guided self-organization [11, 6, 10] the proposed framework has the benefit of decoupling self-organization and goal-orientation. On the one hand this means that the internal dynamics of the controller cannot be stimulated to stay in or go to specific regimes, but on the other hand a goal-oriented behavior of the system can be achieved and a mutual interference of self-organization and goal-orientation parameters is strongly reduced. The framework parameters can be predefined or adapted/learned with basically any approach.

Furthermore, the proposed system can also be applied to other intrinsically motivated systems (or predefined controller), since the modulation of inputs and outputs is done in a transparent way from the point of view of the self-organizing controller. Limitations are only reached, if the intrinsically motivated controller already generates complex behaviors which impair the goal-oriented behavior generation.

The presented approach provides a possibility to combine self-organization and goal-orientation for the control of autonomous robots in an independent fashion. Self-organization based basic motor primitives can be used to achieve a given goal. This requires less initial knowledge about the properties of robot and environment and allows adaptation to changes of the robot's morphology without relearning or adaptation of the goal-oriented behavior. A combination of self-organization and reward-based learning seems thus a promising route for the development of adaptive learning systems.

Acknowledgments

Helpful discussions with Poramate Manoonpong are gratefully acknowledged. This research was supported by the BFNT Göttingen (project 3B, 01GQ0811).

References

[1] Asadpour, M. and Siegwart, R., Compact Q-learning optimized for micro-robots with processing and memory constraints, *Robot. Auton. Syst.* **48** (2004) 49–61.
[2] Der, R., Self-organized acquisition of situated behaviors, *Theor. Biosci.* **120** (2001) 179–187.
[3] Der, R., Hesse, F. and Martius, G., Rocking Stamper and Jumping Snakes from a dynamical systems approach to artificial life, *Adapt. Behav.* **14** (2006) 105–115.
[4] Der, R. and Martius, G., *The Playful Machine - Theoretical Foundation and Practical Realization of Self-Organizing Robots* (Springer, 2012).
[5] Guo, H. and Meng, Y., *Dynamic Correlation Matrix Based Multi-Q Learning for a Multi-Robot System*, 201 (IEEE, 2008).
[6] Hesse, F., Der, R. and Herrmann, J. M., Modulated exploratory dynamics can shape self-organized behavior, *Adv. Complex Syst.* **12** (2009) 273–291.

[7] Hesse, F., Martius, G., Der, R. and Herrmann, J. M., A sensor-based learning algorithm for the self-organization of robot behavior, *Algorithms* **2** (2009) 398–409.
[8] Lee, D.-H., Park, I.-W. and Kim, J.-H., *Q-Learning Using Fuzzified States and Weighted Actions and its Application to Omni-Direnctional Mobile Robot Control*, 1 (IEEE, 2009).
[9] Liu, Y., Pan, Z., Stirling, D. and Naghdy, F., Q-learning for navigation control of autonomous blimp, in *Australasian Conf. Robotics and Automation (ACRA)*, December 2–4, 2009, Sydney, Australia.
[10] Martius, G. and Herrmann, J., Taming the beast: Guided self-organization of behavior in autonomous robots, in *From Animals to Animats 11: Proc. 11th Int. Conf. Simulation of Adaptive Behavior, SAB 2010, Paris-Clos Lucé, France, August 25–28, 2010* (Springer, 2010), ISBN 3642151922, p. 50, doi: 10.1007/978-3-642-15193-4_5.
[11] Martius, G., Herrmann, J. M. and Der, R., Guided self-organisation for autonomous robot development, *Lect. Notes Comput. Sci.* **4648** (2007) 766.
[12] Martius, G., Hesse, F., Güttler, F. and Der, R., LpzRobots: A free and powerful robot simulator (2011), http://robot.informatik.uni-leipzig.de/software.
[13] Nolfi, S., Behaviour as a complex adaptive system: On the role of self-organization in the development of individual and collective behaviour, *Complexus* **2** (2004) 195–203.
[14] Ott, E., *Chaos in Dynamical Systems* (Cambridge University Press, 1993).
[15] Park, K.-H., Kim, Y.-J. and Kim, J.-H., Modular Q-learning based multi-agent cooperation for robot soccer, *Robot. Auton. Syst.* **35** (2001) 109–122.
[16] Pfeifer, R., Lungarella, M. and Iida, F., Self-organization, embodiment, and biologically inspired robotics, *Science* (*New York*) **318** (2007) 1088–1093.
[17] Prokopenko, M., Guided self-organization., *HFSP J.* **3** (2009) 287–289.
[18] Strogatz, S. H., *Nonlinear Dynamics and Chaos* (Westview Pr, 2001).
[19] Sutton, R. S. and Barto, A. G., *Introduction to Reinforcement Learning* (MIT Press, Cambridge, MA, USA, 1998).
[20] Tani, J., Learning to generate articulated behavior through the bottom-up and the top-down interaction processes, *Neural Netw.* **16** (2003) 11–23.
[21] Touzet, C. and Santos, J. F., Q-learning and robotics, in *IJCNN, European Simulation Symposium*, Watkins 1989 (Marseille, France, 2001).
[22] Watkins, C., *Learning From Delayed Rewards*, Phd thesis (Cambridge University, Cambridge, England, 1989).
[23] Wörgötter, F. and Porr, B., Reinforcement learning, *Scholarpedia* **3** (2008) 1448.