Self-organized robot behavior from the principle of homeokinesis

Ralf Der
Universität Leipzig
Institut für Informatik

May 1, 2000

Abstract

The paper introduces homeokinesis as a general principle for the self-organization of robot behavior. Homeokinesis is a domain-invariant principle which induces specific, seemingly goal–oriented behaviors of an agent in a complex external world. The principle is based on the assumption that the agent is able of (i) learning an internal representation (self-model) of its current behavior and (ii) to adapt its behavior in such a way that the complexity gap between model and true behavior is minimized. Driven by the basic impetus of always moving and acting the agent in a noisy complex world develops smooth controlled behaviors as the minimal compromise between the high dynamical complexity of the world and the low complexity of the model. In these terms behavior may be called a byproduct (epiphenomenon) of the internal "desire" of the agent to be able of interpreting the world in simple terms.

1 Introduction

Can the self-organization of behavior from a practical point of view be more than wishful thinking? In order to answer this question let us have first a look into physics which provides many examples of self-organization (SO) together with a well developed theory of such phenomena. Open physical systems driven by external influences into a steady state far from equilibrium may show the spontaneous creation of patterns in space, in time or both. Standard examples are given by the Bénard cells, chemical oscillations, or the creation of waves in active media (like reaction-diffusion systems), respectively. The patterns reflect a new dynamical organization of the system which is inscribed neither into the microscopic dynamics nor into the external conditions. So the system may be called to have created this organization "out of nothing" by itself.

Although largely different in detail, there is a common principle guiding this process of SO. The systems concerned are characterized by (i) observing global symmetries (like the homogeneity and isotropy of a chemical fluid) and (ii) the presence of feed-back mechanisms able of amplifying local symmetry breakings caused by fluctuations up to macroscopic scales. The patterns are the result of a symmetry breaking arising as the minimal compromise of violating versus conserving the global symmetries of the system.

In the language of artificial intelligence this symmetry breaking scenario is a domain invariant principle. In the above sense the movement of an agent (robot or living being) in a complex external world is a spatio-temporal pattern. Can we find common, i.e. domain invariant principles which drive the self-organization of such patterns? Little is known here, a notable exception being the principle of homeostasis first introduced in physiology by Cannon as early as 1939 [1]. His aim was to find a basic understanding of the goal-oriented reactions (behavior) of the body in response to external perturbations. The reactions are produced by the orchestrated action of many interweaved control circuits. According to the principle of homeostasis, the function of the individual circuit is not controlled by the behavior of the system as a whole. Instead all the circuit does is to try to keep itself in a stable stationary state. Of course this is done best if the system is able to react to external disturbancies. Hence the behavoir of the body so to say is a byproduct of this internal drive of the system to uphold its stationary state. This is a very general, unspecific principle not tailored to any specific process.

However, the principle has not found much applications in practice. In the present paper we argue that it is the focus on the stationary (homeostatic) state of the system which restricts its applicability for the construction of realistic controllers. The principle of homeokinesis to be introduced is the dynamical *pendant* of homeostasis. Coarsely speaking it means that the internal goal of the agent is not the stabilization of a stationary state but of a definite internal kinetic regime. We will give this general idea a concrete formulation in the follwing.

2 Self-model and kinetic regime

The most striking feature of both the homeostatic and the homeokinetic principle is the dominance of the internal perspective. Behavior is not bred by externally fixed goals but is a epiphenomenon of some internal drive of the agent. The internal world of the robot is built on its sensor values so that the above mentioned kinetic regime is to be formulated in sensor space. The controller of the robot may be viewed as a function mapping sensory inputs $x \in R^n$ to some outputs $y \in R^m$ governing the reaction of the robot to its specific environmental situation. The output determines the new situation (position) of the robot which on its hand corresponds to a new sensoric input. Thus, through the sensory-motor loop, each controller produces its specific trajectory $x(t)$ in sensor space which depends both on the environment and the starting position of the robot.

Figure 1: A typical environment for our robot equipped with a homeostatic controller.

By way of example, let us assume the robot is instructed to keep a fixed velocity, the controller output determining the current turn speed w of the robot in physical space. Let us write the controller [1] as

$$w = g\left(\sum_{i=1}^{n} c_i x_i\right) \tag{1}$$

where $g\left(\right) = \tanh\left(\right)$ and the c_i are the parameters of the controller which we assume to be conveniently chosen for the moment. We assume $x_i \in [0,1] \quad \forall i$ whithout restriction of generality. Let us assume that the sensor values are read at regular intervals and that Δx is the change in sensor values in the current time step. Then the above mentioned dynamics in sensor space may be written as

$$\Delta x = F\left(x; c\right) \qquad x \in R^n, \quad c \in R^p \tag{2}$$

c being the parameter vector of the controller. The (stochastic) function F can be seen as the generator of a time evolution of x, i.e.

$$x\left(t+1\right) = x + F\left(x; c\right) \tag{3}$$

where $x = x\left(t\right)$.Eqs. (2), (3) may be interpreted as the update rule of a time discrete dynamical system, the structure of the function reflecting the complexity of the dynamics.

Now, every set of parameters c corresponds to a definite robot behavior. If the controller supports a stable wall following behavior then after starting from any position in physical space the robot after some time will encounter a wall and follow it at a fixed distance. This means that its sensors values are converging towards $x = x_{wall}$. In the language of dynamical systems we may say that x_{wall} is an attractor of the dynamics in sensor space. In the same way, trapping and following an object at a constant distance corresponds to an attractor dynamics. Now we might be tempted to use this as a general tool for the domain invariant definition of "reasonable" behaviors in a given world. However this principle is not general enough. Consider a controller which is able of guiding the robot smoothly through a complicated maze like the one inf Fig. 1. This controller produces trajectories of a moderate (dynamical) complexity. What we mean by this is that they are somewhere between chaotic trajectories and trajectories produced by a kind of trivial controller like going straight ahead.

[1] We assume that the controller does not refer to earlier sensor values as a PD controller would do. However the latter case can easily be integrated into the present concept.

So we might formulate a domain-invariant principle for the learning of behavior in terms of a measure of dynamic complexity. However this is difficult to realize in practice in a domain-invariant way. The problem is that the level of complexity must be specified somehow. Our approach consists in the following. We introduce a self-model of the agent which is realized by a learning time series predictor

$$\Delta x^P = \Phi\left(x; u\right) \tag{4}$$

where Δx^P is the expected change in sensor values evaluated on the basis of the current values x. The parameters $u \in R^m$ are to be adapted such that under the current controller of the robot the prediciton is as good as possible. Due to the learning the complexity of the model behavior depends on the **structure** of the function Φ alone. From the point of view of homeokinesis the structure of the function Φ defines the kinetic regime which the agent tries to reach and stabilize.

By way of example we consider the models given by

$$\Phi = 0 \tag{5}$$

and (writing Φ for each sensor explicitly)

$$\Phi_i = w \sum_j u_{ij} x_j \tag{6}$$

Eq. (5) underlies the hypothesis that there is no change in sensor values at all. The structure of the model so to say is the trivial or null structure. Eq. (6) states that every change in sensor values is proportional to the turn speed of the robot as given by the controller, the structure of the model being the bi-linear form of the expression. Of course these are very simple models but we will see that they are sufficient for the self organization of various kinds of nontrivial behaviors.

In all cases of practical interest the complexity of the model behavior is inferior to that of the true behavior. Thus we speak of a dynamical complexity gap between model and reality. The idea now is to use the width of the complexity gap as learning signal for the adaptation of both the model and the controller. So the aim is to adapt behavior such that the complexity gap between true and model behavior is minimized.

3 The learning algorithm

This principle is simply realized by introducing an error function E which measures the distance (complexity gap) between true and model behavior as

$$E = \frac{1}{2} \sum D_i^2 \tag{7}$$

where

$$D_i = \Delta x_i - \Delta x_i^P \tag{8}$$

and more explicitly

$$D_i = \Delta x_i - \Phi_i \left(x; u \right) \tag{9}$$

The observed change in sensor values Δx_i is a function of both the current sensor values x and the controller parameters c. Hence $E = E\left(x, c, u\right)$ so that the parameters c and u can bel learned by gradient descent. The update for u is

$$\Delta u_i = \epsilon \sum_j \left(\Delta x_i - \Phi_i \left(x; u \right) \right) \frac{\partial}{\partial u_i} \Phi_j \left(x; u \right) \tag{10}$$

the partial derivatives being defined explicitly by the model. In order to update c we need the derivatives

$$\frac{\partial}{\partial c_i} E = \frac{\partial E}{\partial w} \frac{\partial w}{\partial c_i} \tag{11}$$

The last derivative can be carried out explicitly using eq. (1), so that we are left with the derivative w.r.t. to w. This gradient information is gained by the following trick. We add a small perturbation $\phi\left(t\right)$ to the output w of the controller. This may be a noise with zero mean or a periodic oscillation. Assuming that the change of ϕ is small in the time interval between two sensor recordings, we may assume that the change in sensor values Δx and hence E is a function of the current value of ϕ. Writing E as $E = E\left(x, c, u\right) = E\left[x, w, u\right]$ the effect of the perturbation is

$$E\left[x, w + \gamma\phi, u\right] = E + \gamma\phi \frac{\partial E}{\partial w} + \ldots \tag{12}$$

where $E = E\left[x, w, u\right]$ and γ is a strength parameter. Now ϕ is to be chosen such that in the average over a longer time interval we have $\overline{\phi} << \overline{\phi^2}$. Then we may write

$$\overline{E\phi} \approx \gamma \overline{\phi^2} \frac{\partial E}{\partial w} \tag{13}$$

so that the gradient can be obtained by a time average over the product of $\phi(t)\,E$.

Collecting the results we have the following learning algorithm: We read the sensor values x_i at regular time intervals of length τ. The new turn speed of the robot is evaluated as

$$w = g\left(\sum c_i x_i\right) + \gamma\phi(t) \tag{14}$$

where $x_i = x_i(t)$.The update of the controller parameters is

$$\Delta c_i = -\varepsilon g' x_i \phi(t)\, E \tag{15}$$

where $x_i(t+\tau) - x_i(t) = \Delta x_i$ and $E = \sum_j (\Delta x_j)^2$ using the trivial model $\Phi = 0$. Equations (14) and (15) constitute the complete algorithm for the homeostatic robot controller.

4 Experiments with physical Khepera robots

We have implemented the algorithm given by eqs. (14) and (15) on chip of a Khepera miniature robot. In the experiments the parameters of the controller were initialized as $c_i = 0 \quad \forall i$, so that initially the robot will crash usually after a short time. Then it must be removed from the wall by hand. After several such actions the robot very soon learns to either avoid the wall or to follow it. The latter behavior is stable as long as there are no dramatic changes in the environment. This could be corroborated in a long time experiment lasting two days during which the robot covered about 8 km. It should be mentioned that during the long time experiment there was no change in learning parameters, in particular we did not use any "cooling" of the learning rate ϵ.

With the learning parameters used the learning of a specific behavior is very fast, typically we need a few minutes (real time) to train a new behavior. Training means to help the robot by hand to recover from unconvenient situations, see above. A typical wall following experiment is exemplified in Fig. 2. It is interesting that when following the wall for some time the controller will change the behavior mode in the sense that it finds the distance to the wall under which control is most stable. For smooth walls this means that it steers with two sensors, choosing the distance to the wall such that the sensor closest to the wall is at maximum value and the next one is at

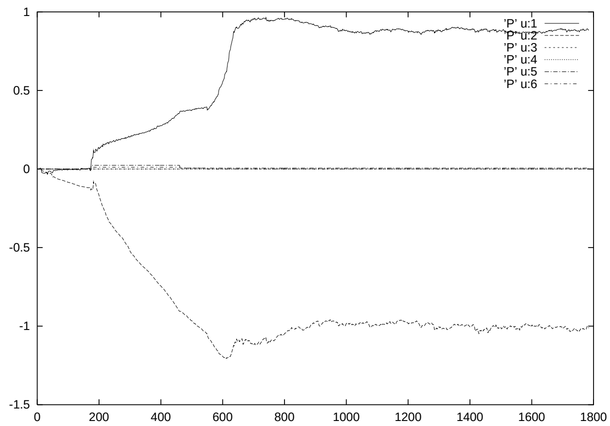

Figure 2: Wall following behavior. The robot sees the wall to its left. Plotted are the values of the controller parameters c_1 through c_6 over robot path. One step in the graph corresponds to about 2 cm. After a very short initial learning period the robot followed the inner wall of a circle of about 80 cm. After about 200 steps the robot was transmitted to the outer wall of a circle of about 60 cm. The robot rapidly adapts to the new situation and stays stable in the new behavior for about 400 steps. Initiated by an external perturbation starting by step 600 there is a transition into a more stable (closer) wall following modus. The latter behavior is completely stable.

half maximum. This is a nontrivial effect and can be understood in terms of minimizing the influence of the noise as explained in [2].

A further point of general interest is that the robot changes its behavior depending on the environmental conditions. Typically one can learn it to balance a ball, cf. Fig. 4, and than put the learned robot under some angle close to the wall. Of course the robot will show the same reaction as with a ball, i.e. it turns until the wall is in front of it and then bangs against the wall. However after helping it severals times out of this crash situation it will learn to follow the wall afterwards.

It is also interesting that the controller may cope with the (simulated) damage of a sensor as demonstrated with the ball balancing behavior which the robot also learns if one of the sensors involved is damaged.

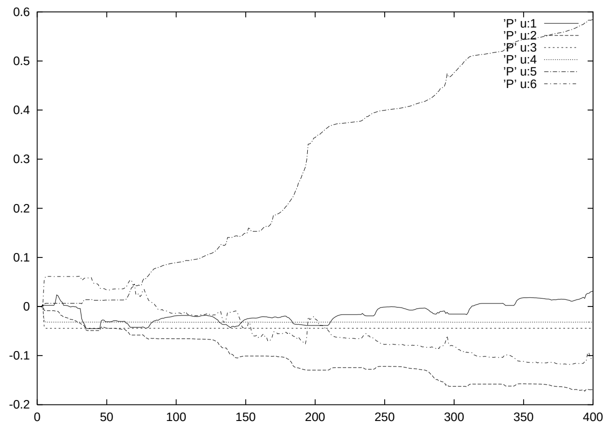

Figure 3: Corridor following: The robot moves in a circular corridor with diameter of about 80 cm. The width of the corridor varies from twice the robot diameter to a few millimeters wider than the robots diameter. The robot nearly immediately learns to follow the corridor. In the closer parts it keeps to the middle of the corridor while in the wider parts it keeps to the inner wall of the corridor. Parameter values (depicted are c_1 through c_6) are not symmetrical because of the different sensor characteristics.

5 Conclusions

Our experiments with a real Khepera robot clearly demonstrated that the hoemokinetic principle is a safe basis for the self organization of behavior with an autonomous robot. Depending on the environmental conditions [2] we have observed a number of different behaviors with always the same adaptive controller and learning parameters. In particular we observed behaviors like

- Collision avoidance,
- Ball balancing,
- Navigation through corridors,

[2]The environmental conditions include the human interferences like choosing the starting position of the robot and the way of restarting it after helping it out of a crash situation.

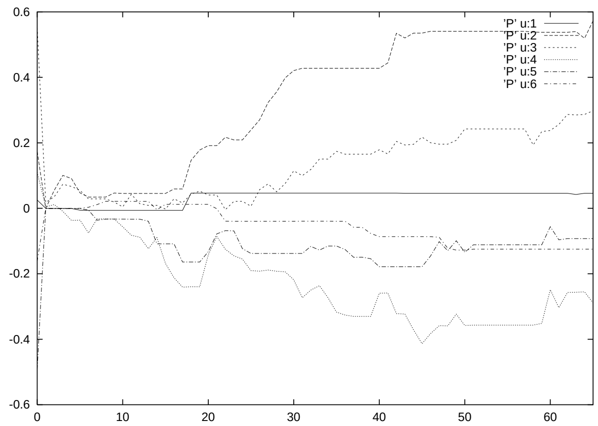

Figure 4: Ball balancing behavior. The robot moving with a constant velocity is presented a table tennis ball which is stabilized for some time by hand in front of it. After about 40 cm (one step in the picture covers about 2 cm) the robot is able of balancing the ball as long as its speed is sufficient to keep up with it. The plot displays the parameters c_1 through c_5 of the controller.

- Different modi of wall following,
- Navigational capabilities in chaotic mazes.

Moreover we observed the phenomenon that the controller converged towards the best working conditions, i.e. the behavior where control is most safely realized. The latter effect results from a nontrivial combination of nonlinearity and noise effects. Learning is found to be extremely fast and permanently alert while in static environments it is convergent and reproducible.

Our future work will be devoted to testing the current controller under new environmental conditions while including the camera and the ambient light sensors. We will also study emerging collective behaviors with several roboters in a common environment each one being controlled by the adaptive homeokinetic controller. Moreover we also would like the robot to develop the "desire" to move which at the time being is given it from outside since we put the forward velocity to a prefixed value. This can be achieved by a predictor looking backward in time which leads to an amplification of the noise effects, cf. [2].

References

[1] W. B. Cannon. *The wisdom of the body.* Norton, New York, 1939.

[2] R. Der, U. Steinmetz, and F. Pasemann. Homeokinesis - a new principle to back up evolution with learning. In M. Mohammadian, editor, *Computational Intelligence for Modelling, Control, and Automation*, volume 55 of *Concurrent Systems Engineering Series*, pages 43 – 47. IOS Press, 1999. URL=http://www.informatik.uni-leipzig.de/ der/Veroeff/wienfin3.ps.