# Dual Dynamics: Designing Behavior Systems for Autonomous Robots[1]

Herbert Jaeger[2] and Thomas Christaller
German National Research Center
for Information Technology (GMD), FIT.KI
Schloss Birlinghoven, D-53754 Sankt Augustin, Germany
phone +49 2241 14 2253, fax +49 2241 14 2384
email: herbert.jaeger@gmd.de, thomas.christaller@gmd.de

December 18, 1997

[1]to appear in *Artificial Life and Robotics*
[2]to whom correspondence should be addressed

**Abstract**

This paper describes the "dual dynamics" (DD) design scheme for robotic behavior control systems. Behaviors are formally specified as dynamical systems using differential equations. A key idea for the DD scheme is that a robotic agent can work in different "modes", which lead to qualitatively different behavioral patterns. Mathematically, transitions between modes are bifurcations in the control system.

# 1 Introduction

The "behavior based" approach to designing mobile robots has been very fertile (Brooks[1], Steels[2], Pfeifer and Scheier[3]). However, the field suffers somewhat from a certain lack of formal theory, which in turn hampers the understanding and the design of robots with increasingly complex behavioral repertoires. This paper introduces a formal model of complete behavior control architectures for mobile robots, the "dual dynamics" (DD) scheme.

Behaviors are construed as dynamical systems, which interact through shared variables. There is no global control; the overall functionality of the system arises from the interactions of the individual behavior subsystems.

Like in many other behavior-oriented control architectures (Brooks[4]), behaviors are ordered in levels in the DD scheme, too. Normally, the organization of the levels is left to the designer's intuition. DD is distinguished by providing *formal* criteria for this ordering. These criteria grow from the rigorous dynamical systems approach. They are expressed in terms of timescales and bifurcations.

The basic assumption on which DD rests is that a situated agent can work in different "modes". Modes are coherent, relatively stable "frames of mind", so to speak, which enable the agent to tune into different situations and tasks. Specifically, agents respond to sensory signal differently in different modes. For instance, when being in `feeding` mode, an agent will show orienting reactions toward food sources, which are ignored in `work` mode. The DD approach rests on the assumption that transitions between modes can be formally captured by bifurctions of dynamical (behavior) systems.

DD helps to advance behavior-based robotics in two ways: (i) theoretically, by offering a rigorous model of behavior control systems in terms of dynamical systems and bifurcations, and (ii) practically, in that the differential equations of DD models can be run directly as actual control programs on robots. In that case, the mathematical transparency of a DD model greatly aids in the design-test-redesign cycle of robot development.

The DD approach has been partially inspired by ethological (e.g. Baerends [5]) and biocybernetical (e.g. Ewert et al[6]) models of hierarchical behavior control systems. Methodologically, it is related to current efforts to exploit dynamical systems theory for behavior-based robotics (e.g. Beer[7], Schöner et al[8]).

The main part of this paper presents a concise description of the DD model (section 2). A brief report of a DD-based robot is given in section 3.

Section 4 concludes with a discussion.

# 2 The DD scheme

In this section, we shall first clarify the notion of modes, then given an informal sketch of the DD scheme, and finally provide the mathematical specification.

## 2.1 Modes

When one observes animals or humans (or even robots), one will find that they behave differently on different occasions. Importantly, this happens even if the environment does not change. For instance, it has been reported that eagles attack glider airplanes in breeding season, whereas they peacefully co-operate in searching for thermal lifts at other times (von Kalckreuth[9]). Observations of this kind demonstrate that different behavior cannot be explained by different external stimuli alone, but must also be caused by some agent-internal component.

An obvious way to account for such findings is to say that agents have internal *states* in the sense of system theory or control theory (Stengel[10], Narendra[11]). Such states might be linked to slowly changing somatic state variables, like hormone concentrations, or to quicker neuronal variables, like arousal.

A consequence of modeling modes by internal state variables would be that modes should change into one another more or less smoothly, following the changes of the state variables. Specifically, one should expect that intermediate modes can be observed. However, this is not supported by common observations. Eagles either attack gliders, or they co-operate. They don't mix these options, for instance by quickly oscillating between them, or by displaying strangely "morphed" motor patterns.

We believe that internal states are not sufficient to model the apparent stability and non-mixability of behavioral modes. We propose to model them by a stronger system-theoretic concept instead, which is called (inconsistently) *regimes* or *modes* in mathematics, *modes* in physics, or *phases* in chemistry. In fact, people in these fields are less interested in the regimes/modes/phases themselves but rather in the boundaries thereof: *bifurcations*.

The modern theory of bifurcations has led to a deep understanding of phenomena of qualitative changes in dynamical systems (Strogatz[12] and Abraham & Shaw[13] are highly recommendable introductions; Kelso et al[14] is a crash course). The general picture is that almost every dynamical system displays abrupt changes between qualitatively different patterns of behavior, when certain *control parameters* transit across *critical values*, but keeps displaying the same qualitative type of behavior in while control parameters shift inside these critical values.

A classical example is the sudden onset of a stable oscillation in an electronic circuit when voltage (= control parameter) surpasses a critical threshold. Importantly, voltage can change considerably beyond this threshold without disrupting the oscillation: the regime of oscillations is *qualitatively stable*. Another well-known example is provided by the three phases of water: when temperature and pressure of a water system surpass certain critical values, water abruptly changes from liquid to gas to solid. Probably every reader has seen in his or her life a *phase diagram* of water. This is a two-dimensional graphic which has pressure $P$ and temperature $T$ as its coordinates, and displays the phase transitions as boundary lines in the $P$-$T$-plane.

The DD scheme builds on the mathematics of qualitatively stable regimes, and bifurcations between them, to model behavioral modes. In this view, state variables (like hormones) are seen as control parameters, which sometimes induce qualitative transitions between coherent patterns of behaviors, while most of the time leaving the overall system in a stable mode.

From the aspect of behavior sciences, the notion of modes has many facets. It is related to *behavior systems* in ethology (Baerends[5], Tyrrell[15]), i.e. patterns of activities that can be identified on phenomenological and functional grounds. Functionally, modes allow the agent to "tune in" to situations, by establishing particular, adaptive filtering and expectation mechanisms for perception, by facilitating particular motor responses and inhibiting others, etc.

The DD approach is not committed to a particular kind of control parameters. In particular, they need not be internally regulated parameters like hormones in animals. In the view of DD, an agent can also get into a particular mode due to sensory input, or by being put into a particular environment. For instance, when a human falls from a ship into the water, he/she will immediately start behaving differently in a very consistent fashion, due to sensory input.

The DD design scheme for a robot control architecture is intended to en-

able a transparent design of behavioral modes, by specifying certain guiding constraints on the interactions between behavior subsystems.

## 2.2 Informal sketch of the DD architecture

The basic building blocks of a DD robot architecture are *behaviors*. They are ordered in levels (fig. 1a). At the bottom level, one finds *elementary* behaviors: sensomotoric coordinations with direct access to external sensor data and actuators. Typical examples are `move_forward` and `turn_left`. At higher levels, there are increasingly comprehensive behaviors, which also have access to sensoric information but cannot directly activate actuators: their task is to regulate modes. Typical examples of higher-level behaviors in a robot are `work` and `replenish_energy`.

Elementary behaviors are different from all higher-level behaviors in that they are made from two subsystems (fig. 1a), which serve quite different purposes. This has given the approach its name, "dual dynamics".

The first of these subsystems is called the *target dynamics*. It calculates target trajectories for all actuators which are relevant for the particular behavior. For this calculation, the target dynamics has access to every available sensor information, and may include specific preprocessing. The output of the target dynamics consists of as many variables as there are motoric degrees of freedom to be controlled.

A requirement for the target dynamcis is that this system should not undergo bifurcations. This is what makes elementary behaviors elementary, and provides the first formal criterium for level organization, namely, for deciding which level is elementary. For instance, the target trajectories of `turn_left` in a simple 2-wheeled robot which moves on a flat, clean surface are likely to remain qualitatively unchanged in different instances of the maneuver. Thus, `turn_left` would be a good candidate for an elementary behavior in such a simple robot. By contrast, in a walking machine which has to cope with different surfaces, it is likely that there will be qualitatively different maneuvers for turning left in different circumstances. Each of them would thus yield a separate elementary behavior.

The other subsystem of an elementary behavior is its *activation dynamics*. It regulates a single variable, the behavior's *activation*. The equation ruling this variable should be written in a way that the variable displays a dynamic range between 0 and 1. Intuitively, a value of 1 means that the behavior is fully active, whereas 0 means that it is completely inhibited. Technically,

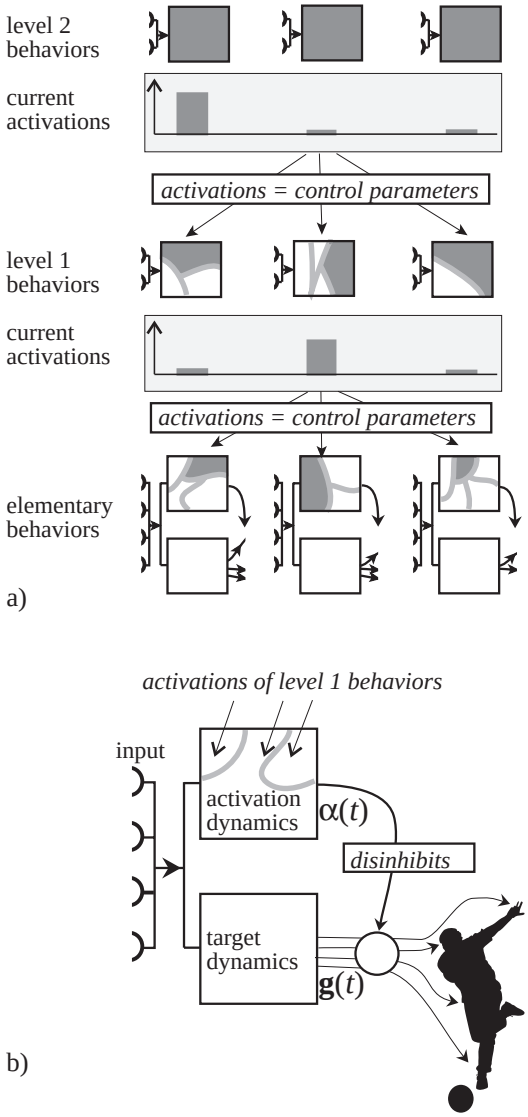


Figure 1: (a) Global structure of a DD behavior control system. At any time, every behavior has an activation. Activations of higher-level behaviors (depicted in shaded boxes) act as control parameters for the activation dynamics of lower levels. The dynamical system which maintains a behavior's activation can undergo bifurcations; this in indicated by depicting these systems as stylized "phase diagrams" (boxes with irregular partitions). A mode of the entire system is thus determined by the activations of all higher-level behaviors. (b) The target and activation subsystems of an elementary behavior.

inhibition is the standard case, and rising values of the activation lead to a disinhibition. Disinhibition means that the target dynamics is passed to the actuators.

The activation dynamics is allowed to undergo bifurcations. *The control parameters which induce these bifurcations are the activation variables of higher-level behaviors.* This is the core idea behind DD. It will be shown in the next subsection how these bifurcations are technically effected.

To illustrate this central point, assume first that the robot is "hungry", i.e. that some high-level behavior `replenish_energy` has a high activation, and other high-level behaviors like `work` have a low activation. In this condition, an elementary behavior `turn_left` should raise its activation variable to 1 when a charging station is perceived at the left. Now assume that the robot becomes eager to work, i.e. on higher levels `replenish_energy` gets deactivated and `work` gets activated. Then, the activation dynamics of `turn_left` should work in a qualitatively different manner. It should now rise toward 1 when an opportunity to work is perceived at the left but not when a charging station is seen. In other words, the activation dynamics of `turn_left` should undergo a bifurcation when higher-level activations of `replenish_energy` and `work` change in a certain way.

To reiterate, only the activation dynamics subsystem undergoes bifurcations in a properly designed DD scheme. The fact that bifurcations (which are inherently difficult to master from a designer's perspective) are confined to these one-dimensional subsystems contributes greatly to the transparency of DD behavior control systems.

Higher-level activation variables yield control parameters for lower-level activation dynamics. Now, in the theory of dynamical systems it is assumed that control parameters change on a (much) slower timescale than the systems they control. This implies that behaviors on different levels in a DD architecture must have different timescales, with higher-level behaviors being long-term and lower-level behaviors become active/inactive on a short-term scale. This provides the designer with the second formal criterium for level organization, namely, how higher-level behaviors are related to each other "vertically".

Concluding this subsection, we would like to emphasize that an elementary behavior is not "called to execute" from higher levels. The level of elementary behaviors is fully operative on its own and would continue to work even if the higher levels were cut off. The effect of higher levels is not to "select actions", but to change the overall dynamics of the entire elementary

level, by inducing bifurcations in that level.

## 2.3 The formal model

Here we present a core version of the formal DD model. A more elaborate version is described in more detail in Jaeger[16].

By convention, the level of elementary behaviors is numbered the level 0, with the next higher level being level 1, etc.

First we describe a single elementary behavior.

The target dynamics of an elementary behavior $B_j$ yields a vector-valued target trajectory $\mathbf{g}_j(t)$, where each vector component represents the target trajectory of a particular effector. The target dynamics is expressed via ordinary differential equations:

$$\dot{\mathbf{g}}_j = G(\mathbf{g}_j, \alpha_j, I_j(t)) \tag{1}$$

The variable $\alpha_j$ is the activation of $B_j$ (see below). $I_j(t)$ represents time-varying input to $B_j$, such as sensor input.

The activation dynamics of $B_j$ consists in the trajectory of a single parameter, $\alpha_j$, which determines whether the behavior is inhibited ($\alpha_j \sim 0$) or active ($\alpha_j \sim 1$), or something in between.

The activation dynamics can bifurcate, yielding different modes. This is achieved by utilizing the activations $\alpha'_1, \ldots, \alpha'_m$ of the level-1 behaviors $B'_1, \ldots, B'_m$ as control parameters for the activation dynamics of $B_j$. It looks as follows:

$$\begin{aligned} \dot{\alpha}_j = {} & \alpha'_1 T_{j,1}(\alpha_j, \mathbf{g}_j, I_j(t)) + \ldots + \\ & + \alpha'_m T_{j,m}(\alpha_j, \mathbf{g}_j, I_j(t)) - \alpha_j k \prod_{i=1,\ldots,m} (1 - \alpha'_i)^2 \end{aligned} \tag{2}$$

$T_{j,i}(\alpha_j, \mathbf{g}_j, I_j(t))$ is a function of $\alpha_j$, the behavior's target trajectory $\mathbf{g}_j$, and possibly other input $I_j(t)$. Each $T_{j,i}$ corresponds to a particular mode of (2), which is entered when $\alpha'_i$ is roughly equal to 1, and the other $\alpha'_k$ roughly equal to 0. If $\alpha'_i$ goes to zero and another $\alpha'_r$ rises, the dynamics of (2) changes from being determined by $T_{j,i}$ to being determined by $T_{j,r}$, which will typically result in a bifurcation of (2). The terms $T_{j,i}$ can be designed explicitly and independently from one another. Thus, bifurcations, which are

generally hard to master, are "tamed" here simply by explicitly providing different dynamical laws $T_{j,i}$ for the different regimes/modes/phases of the dynamics of $\dot{\alpha}_j$.

The decay term $-\alpha_j k \prod_{i=1,\ldots,m}(1-\alpha'_i)^2$ brings $\alpha_j$ back to zero in case all $\alpha'_i$ vanish to zero.

As to the question of what kind of input $I_j(t)$ is permissible for an elementary behavior $B_j$, DD has an iron rule: *the only input which comes top-down from higher-level behaviors is the mode control parameters $\alpha'_i$ in the activation dynamics*. In turn, this rule implicitly admits to use every conceivable non-top-down source for $I_j(t)$, e.g., sensor input, activations of other elementary behaviors, or their target trajectories.

In order to yield an output signal from an elementary behavior to the actuators, $\mathbf{g}_j(t)$ and $\alpha_j(t)$ are combined via the following product assignment:

$$\dot{\mathbf{u}}_j = k_j \alpha_j (\mathbf{g}_j - \hat{\mathbf{z}}_j), \tag{3}$$

where $\hat{\mathbf{z}}_j$ is the estimated current state of the actuators, $k_j$ is a gain constant, and $\mathbf{u}_j$ is the signal issued from the behavior to the actuators. This *product term* implements a simple closed loop control (of P-controller type) which tries to make the actuators follow the target trajectory $\mathbf{g}_j$. If this propertional control proves inefficient, the product term can be augmented to more complex control schemes, e.g. PID-control. The DD scheme is not committed to a particular kind of control realized in this term. The product term corresponds to the open circle in fig. 1b.

Taken all together, the DD model of an elementary behavior consists of the equations (1), (2), and a suitable version of (3).

Now let us turn to the complete picture. The DD scheme allows to construct multi-level architectures with any number of levels. Since higher levels – the top level excepted – are formally similar to each other, we can restrict this presentation to the case of a three-level architecture with $l$ level-2 behaviors $B''_1, \ldots, B''_l$, $m$ level-1 behaviors $B'_1, \ldots, B'_m$, and $n$ elementary behaviors $B_1, \ldots, B_n$. The latter have already been treated. The activation dynamics of the level-1 behaviors is similar to (2) and looks as follows:

$$\begin{aligned} \dot{\alpha}'_i &= \alpha''_1 T'_{i,1}(\alpha'_i, I'_i(t)) + \ldots + \\ &\quad + \alpha''_l T'_{i,l}(\alpha'_i, I'_i) - \alpha'_i k \prod_{h=1,\ldots,l} (1-\alpha''_h)^{r'_i} \end{aligned} \tag{4}$$

Since we assume here that there are no levels higher than 2, the activation dynamics of level-2 behaviors does not bifurcate (since there are no control parameters available from higher up). Therefore, the activation dynamics of $B''_h$ has the simple form

$$\dot{\alpha}''_h = T''_h(\alpha''_h, I''_h(t)) \tag{5}$$

The iron rule concerning inputs mentioned in the previous section transfers to higher-level behaviors. Thus, $I'_i(t)$ and $I''_h(t)$ can be virtually anything provided it does not come from higher levels.

# 3 Practical experiences with DD

A reference implementation of the DD scheme has been carried out as part of cooperative work between GMD and the VUB AI Lab. The VUB AI Lab offers a large experimental arena which has been devised as a model of an ecosystem (McFarland[17]). Robots in this arena have to "survive" by performing a robotic analogue of a "parasite fighting" task in an energy-efficient fashion, and by recharging in due intervals. The arena is a challenging environment since it offers no controlled lighting conditions (lighting ranges from direct sunshine in the morning to distant artificial lighting at night hours), while the robots have to rely on simple light sensors for achieving their tasks.

One of the Lab's Lego-based 2df robots, the "Black Knight" (BK) was used as a platform for a reference implementation for the DD scheme[1]. A repertoire of behaviors that is standard for many robots running in the "ecosystem" arena was re-implemented using the DD scheme. The differential equations of DD were simulated on the robot's microcontroller hardware using the PDL language (Steels[2], Spenneberg et al[18]) , which in turn is written in C.

A detailed account of experiences with designing BK's behavior control system is not feasible here. A close-up investigation on a particular behavior can be found in Jaeger[19]. Suffice it to say that specifying the behavior control system in terms of some 50 differential equations took the first author a week, programming and getting rid of C errors 10 days. These times include learning the basics of the C programming, and overcoming the platonic

[1]Cf. ftp://ftp.gmd.de/GMD/ai-research/Publications/1996/jaeger.96.dd14.c for the heavily documented code.

misconception of the existance of reliable sensor readings – this author (mathematician by training) was at that time a novice in practical robotics. The work which usually takes most of the time, namely, testing and fine-tuning, then took only another 2 days. It was found that the conceptual clarity of the DD design allowed, in almost all cases, to conclude from apparent faulty external behavior where the fault in the internal control system resided.

Current work is centered on augmenting DD by planning and representation capabilities (Hertzberg et al[20]). The guiding idea is to use the history of activations as the robot's memory, from which its world model is made. Thus, there is an intrinsic connection between representation and action. This project is being implemented on a B14 robot platform.

# 4 Discussion

The DD scheme has several obvious limitations, among them the following two. First, it is a very general scheme which helps to organize a complete system of many behaviors, but it does not say something very specific about how the target and activation dynamics of each behavior should be designed. Second, the mathematical format of ordinary differential equations is confined to handling sensoric input which consists in a few variables. Specifically, this excludes video input.

The advantages of DD lie in its conceptual and mathematical clarity, which in turn leads to transparent, and hence efficient, design. After all, DD made it possible for a newcomer to practical robotics to finish a design and implementation job within three weeks, achieving a behavioral complexity that is state of the art in behavior-based robotics. This holds some good promise for DD to lead beyond the limitations in behavioral complexity with which the field seems to have gone stuck since some years.

## Acknowledgements

The authors feel deeply grateful toward Luc Steels and his collaborators for a very rewarding co-operation.

# References


[1] R. A. Brooks. Intelligence without reason. A.I. Memo, ftp'able at http://www.ai.mit.edu/ 1293, MIT AI Lab, 1991.

[2] L. Steels. Building agents out of autonomous behavior systems. In L. Steels and R.A. Brooks, editors, *The "Artificial Life" Route to "Artificial Intelligence": Building Situated Embodied Agents*. Lawrence Erlbaum, 1993.

[3] R. Pfeifer and Ch. Scheier. *An Introduction to New Artificial Intelligence*. To appear in MIT Press, 1997.

[4] R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23, 1986.

[5] G.P. Baerends. On drive, conflict and instinct, and the functional organization of behavior. In M.A. Corner and D.F. Swaab, editors, *Perspectives in Brain Research. Proc. of the 9th Int. Summer School of Brain Research, Amsterdam, August 1975*, pages 427–447. Elsevier, Amsterdam, 1975.

[6] J.-P. Ewert, T.W. Beneke, E. Schürg-Pfeiffer, W.W. Schwippert, and A. Weerasuriya. Sensorimotor processes that underlie feeding behavior in tetrapods. In V.L. Bels, M. Chardon, and P. Vandewalle, editors, *Biomechanics of Feeding in Vertebrates*, volume 18 of *Comparative and Environmental Physiology*, pages 119–161. Springer Verlag, 1994.

[7] R.D. Beer. The dynamics of adaptive behavior: a research program. *Robotics and Autonomous Systems*, 20:257–289, 1997.

[8] G. Schöner, M. Dose, and C. Engels. Dynamics of behavior: theory and applications for autonomous robot architectures. *Robotics and Autonomous Systems*, 16(2):213–246, 1995.

[9] J. von Kalckreuth. *Das stille Abenteuer*. Motorbuch-Verlag, 1975.

[10] R.F. Stengel. *Stochastic Optimal Control*. John Wiley and Sons, 1986.

[11] K.S. Narendra. Identification and control. In M.A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 477–480. MIT Press/Bradford Books, 1995.

[12] S.H. Strogatz. *Nonlinear Dynamics and Chaos*. Addison Wesley, 1994.

[13] R.H. Abraham and C.D. Shaw. *Dynamics: The Geometry of Behavior*. Addison-Wesley, Redwood City, 1992.

[14] J.A.S. Kelso, M. Ding, and G. Schöner. Dynamic pattern formation: A primer. In L.B. Smith and E. Thelen, editors, *A Dynamic Systems Approach to Development: Applications*, pages 13–50. MIT Press/Bradford Books, 1993.

[15] T. Tyrrell. The use of hierarchies for action selection. *Adaptive Behavior*, 1(4):387–420, 1993.

[16] H. Jaeger. The dual dynamics design scheme for behavior-based robots: A tutorial. Arbeitspapiere der GMD 966, GMD – Forschungszentrum Informationstechnik GmbH, St. Augustin, 1996.

[17] D. McFarland. Towards robot cooperation. In D. Cliff, editor, *From Animals to Animats III: Proceedings of the 3rd International Conference on Simulation of Adaptive Behavior*, pages 440–444. Bradford/MIT Press, 1994.

[18] D. Spenneberg, E. Schlottmann, T. Höpfner, and Th. Christaller. PDL programming manual. Arbeitspapiere der GMD 1082, GMD, 1997. http://www.gmd.de/FIT/KI/CogRob/Publications/CogRob.Publications.html.

[19] H. Jaeger. Brains on wheels: Mobile robots for brain research. *Manuscript, available at http://www.gmd.de/People/ Herbert.Jaeger/Publications.html*, 1996.

[20] J. Hertzberg, H. Jaeger, P. Morignot, and U.R. Zimmer. A framework for plan execution in behavior-based robots. http://www.gmd.de/People/Herbert.Jaeger/Publications.html, 1998.