Identification of Behaviors in an Agent's Phase Space

Herbert Jaeger
GMD, St. Augustin
herbert.jaeger@gmd.de
herbert@arti.vub.ac.be

October 1995

Abstract: This paper describes a method for analyzing the behavior of an autonomous robot. The robot is viewed as a continuous, stochastic dynamical system. The analysis starts from an empirical phase portrait. In a first stage, elementary regularities are detected. These regularities, called transient attractors, combine properties of attractors with properties of partition cells of phase space. As the system trajectory passes through these regularities repeatedly, a sequence of identifiable events is produced, which can be interpreted as a symbol sequence. This sequence is further analyzed in the second stage, where a finite description of temporal regularities within it is constructed. This description comes in the format of a variety of finite-state automata. In a third stage, higher-order regularities in this finite-state-like description are identified. This yields a hierarchic *behavior model*. At all stages, regularities are defined using criteria of maximal local predictability. Thus, the entire method can be seen as an information theoretic approach to bridging the gap between the non-symbolic, quantitative level of robots and higher-level symbolic models.

Zusammenfassung: Es wird eine Methode zur Analyse des Verhaltens von autonomen Robotern vorgestellt. Letztere werden als kontinuierliche, stochastische, dynamische Systeme aufgefaßt. Die Analyse geht von einem empirisch bestimmten Phasenportrait aus. In einer ersten Stufe werden darin elementare Regularitäten identifiziert, die ich transiente Attraktoren nenne. Sie kombinieren Eigenschaften von Attraktoren mit Eigenschaften von Partitionszellen in Phasenräumen. Indem die Systemtrajektorie wiederholt durch diese Regularitäten hindurchgeht, wird eine Ereigniskette bestimmt, die formal als Symbolsequenz aufgefaßt werden kann. In einer zweiten Analysestufe wird eine endliche Beschreibung von temporalen Regularitäten in dieser Sequenz vorgenommen. Diese Beschreibung ist formal verwandt mit endlichen Automaten. In einer dritten Analysestufe werden in dieser automatenähnlichen Beschreibung Regularitäten höherer Ordnung identifiziert. Es ergibt sich schließlich insgesamt ein hierarchisches *Verhaltensmodell*. In allen Analysestufen werden die jeweiligen Regularitäten durch lokale Optimalitätskriterien der Prozeßvorhersagbarkeit bestimmt. Der gesamte Analyseprozeß erbringt so eine informationstheoretisch orientierte Verbindung der quantitativen Beschreibungsebene mit höheren symbolischen Modellen.

1 Introduction

Recently, continuous dynamical systems theory has been proposed as a methodological framework for behavior-oriented robotics [24] [5]. An agent is viewed as a dynamical system, which consists of a *phase space* spanned by numerical variables that can be used to describe the agent's *states*, and a dynamics which yields *trajectories* in the phase space. The question immediately arises how the basic notion of behavior-oriented robotics, namely, *behaviors*, can be captured within this framework.

Formal models of behaviors are needed for *designing* and *analysing* robots. Both tasks are crucial for progress. They are interdependent, since the development of robots proceeds in a design-analyse-redesign cycle. Formal models of behaviors are likely to differ to some extent, however, for design vs. analysis. The reason is that in design, one usually starts from a known set of system parameters and builds the system out of them, whereas in analysis, one typically has no prior knowledge of the "right" dimensions for understanding the system. Therefore, behavior models for design can be formulated in terms of given system variables, whereas behavior models for analysis should be more or less independent from the selection of particular observables.

Dynamical systems oriented behavior models for design have been proposed by Schöner et al. [21] and myself [17]. In both cases, behaviors are formally treated as subsystems in a comprehensive agent supersystem. Schöner et al. make the additional committment that there exist an attractor in the subsystem. He identifies the behavior with this attractor in particular rather than with the subsystem in general. The techniques presented in these papers rely on the availability of formal descriptions of such subsystems, which renders these techniques applicable to design tasks only.

By contrast, the present article deals with the question of how behaviors can be identified in a phase portrait rendered by the observation of a robot's actions. I.e., it is concerned with the question of behavior models for analysis tasks. Some hints of how this task could be adressed can be gleaned from the literature. Luc Steels [27] presents a case study of an autonomous vehicle where some basic behaviors such as "move forward" appear as point attractors. Randy Beer [5] analyzes walking pattern generators in simulated insects and finds instances where their dynamics can be understood as a transient that alternatingly comes under the influence of either of two attractors. Tim Smithers [23] contributes a more general observation, in that he emphasizes that behaviors must be sought for in the dynamics of variables that belong to the dynamical systems of both the agent and its environment, i.e., that belong to the "interaction space". One should also mention research in psychophysics where observed motor behavior patterns

are formally reconstructed as attractors (e.g. [22], [20]).

With the exception of [23], where the perspective is still too abstract for yielding concrete criteria for identifying behaviors, these contributions concern the description of single behaviors. An essential tacit assumption is that an isolated behavior can be exhibited and observed for a long enough time such that its characteristics proper can surface over transient effects. Essentially it is required that single behaviors occur as stationary, isolated phenomena. The definition of behaviors in terms of attractors in particular seems to be tied to this assumption of stationarity. This is at odds with one of the basic tenets of behavior-oriented robotics, namely, that most of the time many behaviors are active simultaneously and in tight interaction [26]. When one concentrates on identifying behaviors in isolation, the central question of understanding multi-behavior *systems* is shunned.

In the present article I address the task of reconstructing formal models of behavior systems that include many different, interacting behaviors. I propose an answer to a central question that I have raised earlier [15], namely, how can behaviors be described mathematically when most of the time they are exhibited only transiently.

The behavior model that I develop here is rather different from the simpler behavior model for design that I use in [17]. I am currently working out a more elaborated behavior model for design purposes, which is more similar to the analysis model described in this article. I hope but cannot promise that the two types of model will be unified eventually. The field is very much in its infancy, which justifies working from different ends.

The article is organized as follows. Section 2 gives a brief overview on existing approaches to detecting identifiable regularities in continuous systems. Two main routes are being pursued, which rely on attractors and partitioning cells, respectively. They have complementary merits. In section 3, I develop the notion of transient attractors, which combines aspects of these two standard approaches. Transient attractors are defined in terms of locally optimizing the prediction of a stochastic process. An algorithm for detecting transient attractors in empirical (typically quite messy) phase portraits is sketched. The techniques described in this section yield a symbol sequence from a continuous phase portrait. Section 4 deals with the description of higher-level regularities in such symbol sequences. In the literature, finite automata models are often used to this end. In particular, minimal automata are used to measure the complexity of such sequences. This standard approach is, however, not quite satisfactory since finite automata intrinsically contain descriptions of initial transients, whereas one would prefer a model of stationary symbolic processes. As a way out of this situation, I propose an alternative finite-automaton-like model, with an according normal form

representation, called phase generators. The states of phase generators are constructed, again, from a principle of maximizing local predictiveness. An algorithm for deriving a phase generator description from an empirical (i.e., noisy) symbol sequence is sketched. Finally, the entropy of processes generated by a probabilistic version of phase generators is computed. In section 5 I show how higher-order regularities can be defined within phase generators. The phase generator together with these higher-order regularities yields the desired behavior model. Section 6 provides a conclusion.

2 Sequences of discrete observational units in continuous dynamical systems

There are two main routes to describing sequences of discrete observational units in continuous dynamical systems. On the one hand, many investigations concern sequences of *attractor states* which a system runs through under suitable conditions. This perspective characterizes much of the relevant work carried out in neural network research. On the other hand, one can simply partition phase spaces, treat the *partition cells* as observational units, and trace trajectories through the sequence of cells they visit. In this section, I review some examples of these two approaches and point out their complementary merits and shortcomings.

Attractors are natural candidates for discrete, identifiable, representational entities in connectionist systems used for intelligent information processing tasks. In particular, point attractors are often considered to represent concepts, preceptual categories, and sometimes, behaviors. The tradition of such approaches can be traced back to the spreading-activation semantic networks of the pre-connectionist era (e.g., [33]). Waltz and Pollack [31] describe a localist network for sentence comprehension tasks. Equilibrium states (another way of speaking for point attractors) of this network represent coherent "mental states" which lead to the disambiguation of input words. Their work has induced several similar approaches. Smolensky's "harmony theory" [25] also treats linguistic information processing through equilibration processes in neural networks. He develops a rigorous mathematical analysis of these processes in terms of statistical mechanics. Balkenius and Gärdenfors [3] use equilibration processes in neural networks for modeling nonmonotonic inferences yielding concepts as their result. He shows how several formal, logical requirements for such inferences are satisfied by neural network mechanisms. Last but not least, there are several strands of work in psychophysics and behavior-oriented robotics where motor behaviors are modeled by attractors.

I have already mentioned these [22][20][27][21][5] in the introduction.

Limit cycles and chaotic attractors, too, have been identified with concepts, perceptual categories, and representations of motor programs. This view is common in recent work on biologically inspired, artificial neural networks (e.g. [7]) and on neural network models of biological brain systems (e.g. [34], [2]).

Attractors are attractive models for concepts, perceptual categories, motor behaviors, and other descriptive units used by scientists when they describe information processing in animals or robots. The main reason for this attractiveness is that attractors are *discrete* and *stable*, which makes them *identifiable* and *detectable*. Thus, they are natural system-theoretic models for *symbols* [13] [8] [30] in the sense of the physical symbol systems hypothesis [1]. Among other properties, this basic paradigm of symbolic AI requires symbols to be identifiable and temporally persistent.

Attractors have a drawback, however. Strictly speaking, a dynamical system cannot leave an attractor state by the very definition of the latter. But, obviously, concepts, perceptual categories etc. occur in activation/deactivation cycles. Therefore, if one uses attractors as models for symbolic processing units, one has to introduce additional mechanisms to explain how the system trajectory can enter attractor states and leave them again. Several such mechanisms have been proposed. In the context of storing temporal sequences in neural networks, mechanisms based on noise, the semistability of near-limit-subcycles in chaotic attractors, or the decay characteristics of synapses have been investigated (brief overview in [16], section 4.2.1). In non-stationary systems, external input can drive a system through different attractor states (e.g. [34]). This is, of course, a typical situation in animals or robots.

The other main route to deriving symbol sequences from a continuous dynamical system rests on partitioning its phase space. The partition cells are taken as symbolic units. The system trajectory then yields symbol sequences simply by passing through sequences of such cells. This view dominates the information-theoretic analysis of dynamical systems. The definition of a continuous system's (metric and topological) *entropy* rests on a construction of this kind [19]. In their search for a mathematical theory of complexity, theoretical physicists have derived discrete automata models from continuous systems in a similar fashion [11] [10]. I will return to this strand of work in section 4. Recently, there seems to be some interest in neural networks that simulate finite-state automata [28] [29]. Here, a neural network's phase space is partitioned through a learning process, which can lead to cell boundaries with complex shapes. One should also mention that in qualitative reasoning research in AI, partitioning phase spaces is the standard procedure for getting

discrete event sequences (e.g. [18]).

Partitionings of phase spaces yield a direct explanation for temporal sequencing of symbolic units. Furthermore, it is a very simple method, and it makes directly applicable many techniques from ergodic theory. The drawback of partitionings is that they provide no explanation for the apparent stability and identity of symbolic units. A concept or a motor behavior pattern seems to be inherently more than just any odd volume in phase space. There seem to be mechanisms of self-stabilization connected with them, which warrant, for instance, that one can "hold" a concept in one's mind for some time, or that one can correct a temporally extended motor action in the face of perturbations. Self-stabilization gets out of sight when one relies on simple partitioning techniques.

Thus, in sum, attractor and partitioning approaches have complementary merits. The technique for phase space analysis that I will develop in the next section combines aspects of both attractors and partitionings in a way which preserves, hopefully, the advantages of each.

3 Deriving a symbol sequence from a phase portrait

In this section, I outline an approach to extracting a symbol sequence from an empirical phase portrait as yielded by an extended run of a robot. The technique combines ideas from the attractor and partitioning approaches reported in the previous section. I shall first explain the basic idea of combining attractors and partitioning cells in a single kind of mathematical object, namely, in *transient attractors*. This part of the section expands on preliminary ideas sketched in [15] [14]. Secondly, I shall consider the task of effectively detecting transient attractors in empirical phase portraits. The basic idea for an algorithm is presented.

3.1 Transient attractors: theoretical considerations

Transient attractors concern the following phenomenon. Assume that one has a continuous dynamical system, in which one monitors the dynamics of some subsystem. This subsystem is a non-stationary system driven by the dynamics of the variables that couple it into the supersystem. Assume that at a given time, the coupling variables have values that warrant the existance of some attractor in the subsystem. Due to its structural stability, this attractor will persist for some time interval while the coupling variables do not leave a certain range. However, the coupling variables might eventually wander out

of this range, and the attractor in the subsystem will break down. This is the core idea of a "transient attractor".

Before entering into a discussion of certain complications, I give a formal example that illustrates the basic idea. Consider the system specified in polar coordinates by

$$\begin{aligned} \dot{\varphi} &= 1 \\ \dot{r} &= (r - r^2)\sin\varphi \end{aligned} \tag{1}$$

Its trajectories are given by $\varphi = t, r = (1 - Ce^{\cos t})^{-1}$. For $-\infty < C < 1/e$, trajectories are closed loops. Among these, $C = 0$ yields the unit circle (fig. 1).

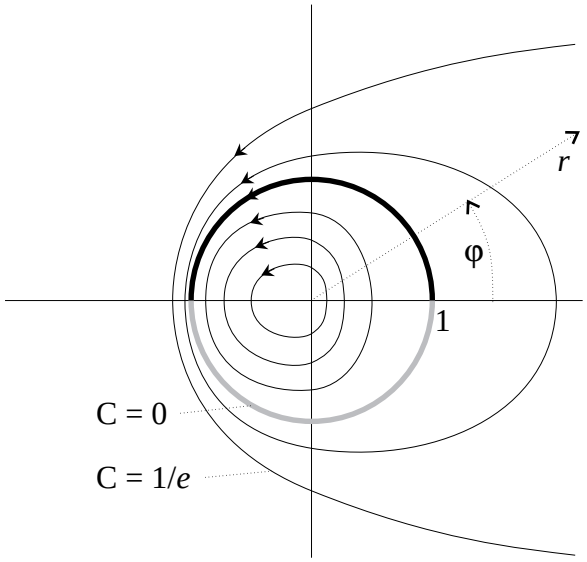

Figure 1: A transient attractor.

When one follows any two trajectories (except the fixed point trajectory $r = 0$) in this system through increasing values of φ, one finds that they come closer to each other in the upper half of the plane (i.e., $0 < \varphi < \pi$), whereas they recede from each other in the lower half.

It is this convergence of trajectories in the upper half that I wish to capture in the notion of a transient attractor. The question is to find a precise definition.

One might attempt a rigorous account of the phenomenon in fig. 1 as follows. Re-interpret $\dot{r} = (r - r^2)\sin\varphi$ as a one-dimensional system with a system variable r and a control parameter φ. If φ is fixed at a value

between $0 < \varphi < \pi$, this system exhibits a point attractor in $r = 1$, which attracts all other trajectories (except again the fixed point in the origin). For $\pi < \varphi < 2\pi$, the point attractor in $r = 1$ becomes a repellor.

One might thus say that the unit circle acts like an attractor in the upper half of the diagram, and as a repellor in the lower half. One might be tempted, as a consequence, to call the unit circle's upper half a "transient attractor".

More abstractly, this attempt to define a transient attractor in a 2-dimensional system

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2) \\ \dot{x}_2 &= f_2(x_1, x_2) \end{aligned} \tag{2}$$

proceeds through the following steps:

- Re-interpret one of the two system equations as a specification of a one-dimensional system with a control parameter. For instance, consider

$$\dot{x}_1 = f_1(x_1, x_2) \tag{3}$$

  as an autonomous system with a system variable x_1 and a control parameter x_2.

- Find some interval $a < x_2 < b$ of the control parameter where the system $\dot{x}_1 = f_1(x_1, x_2)$ exhibits an attractor.

- Observe that while x_2 passes through this interval in the coupled system (2), trajectories "converge" in x_1-direction towards the value of this attractor. Call this a "transient attractor". – It is clear how this procedure generalizes to higher dimensions.

Unfortunately, this seemingly straightforward strategy fails in the general case. The reason is that attractors in systems with control parameters are defined for *fixed* values of the latter, whereas in the coupled system (2) the "control parameter" x_2 changes *dynamically*. If this dynamic change happens on a time scale that is not much slower than that of the autonomous system (3), it can alter the qualitative properties of the latter. In particular, the attractor can disappear. One cannot treat x_2 as a classical control parameter in cases when it exhibits a fast dynamics. I will call x_2 a *coupling variable* instead, in order to emphasize its fast dynamics and its role in coupling the subsystem 3 into the supersystem 2. In [17], I discuss this issue in

some detail, and I describe a "compensation" technique that allows to couple subsystems dynamically without altering their qualitative properties. This technique is mainly useful for *designing* systems. It can be argued, however, that biological behavior systems are often "compensated" to some extent. Inasmuch as they are, it is possible to identify transient attractors in such systems in terms of classical attractors, as suggested in fig. 1.

In sum, the attempt to define transient attractors in terms of classical attractors in certain subsystems works only if the coupling dynamics is slow compared to the autonomous dynamics of the subsystems, or if the subsystems are suitably "compensated".

The example (1) is only intended to make the phenomenon of "converging trajectories" a plausible candidate for transient attractors. In the remainder of this subsection, I will adhere to this basic intuition (fig. 2 a), but I will discuss several ramifications in order to arrive at a more general and realistic definition of transient attractors.

The system (1) is an unrealistically well-behaved, mathematical textbook example. Let us step back a bit and consider the kind of dynamical systems that we are likely to meet with in practice. Empirical phase portraits derived from robot observations differ from textbook systems in several respects:

- Empirical trajectories typically carry high-frequency noise (fig. 2b). In subsequent diagrams, I will draw low-pass filtered trajectories for the sake of clarity, but one must not forget that empirical phase portraits are not rarely smooth.

- Empirical recordings monitor only a few variables of an unknowable multitude of parameters that are causally responsible for the robot's behavior. In other words, one can see only the projection of a high-dimensional dynamical system on a small subspace. I shall call the high-dimensional, unfathomable system the *real system*, the observed low-dimensional subspace the *observation space*, and the dynamical systems observed in this subspace, *observed systems*.

  Several important, mutually related properties of observed systems must be taken into account. First, such systems essentially are stochastic processes. Note that this stochasticity cannot be filtered out like noise, because it is likely to occur at all frequencies. Second, such systems feature crossing trajectories. With respect to our quest for converging trajectories, these two effects imply that we have to deal with "converging webs of trajectories" like in fig. 2c. Trajectories typically arrive at "convergence zones" from different directions, perform some

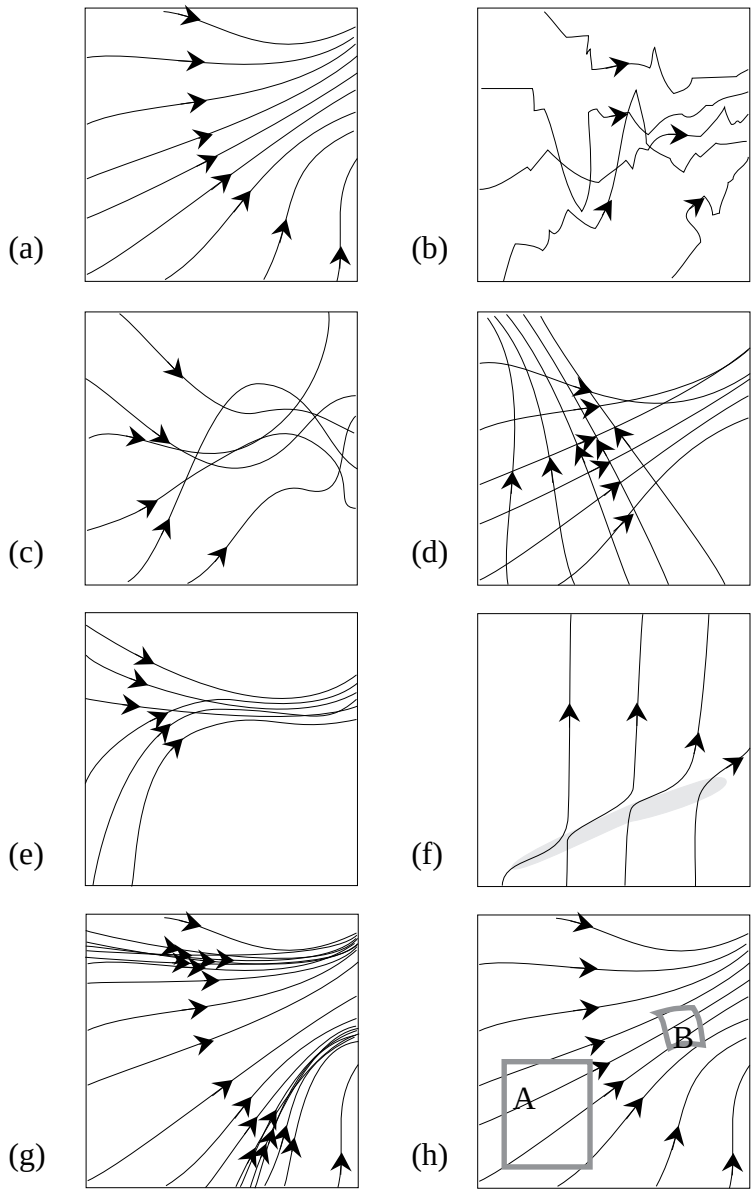

Figure 2: More about the phenomenology of transient attractors. (a) The basic intuition. (b) Noise. (c) Crossing trajectories and arrivals and departures of trajectories from/to impredictable directions. (d) Superposition of transient attractors. (e) Distinguishing two transient attractors by curvature. (f) Shear. (g) Probability effects. (h) Information gain.

crisscrossing, and diverge impredictably to different directions after the event.

- Another consequence of the projection on a subspace is that different transient attractors may occupy the same region in phase space (fig. 2d). In order to separate them from each other, one has to somehow include their direction into the definition.

- Sometimes direction might not suffice, and different transient attractors sharing a single region in phase space may have to be distinguished by their curvature (fig. 2e) or even higher derivatives.

- Yet another consequence of the projection situation is that a zone of convergent trajectories can be sheared in arbitrarily contorted ways (fig. 2f gives a simple example). Such effects can be understood as the result of non-optimal choice of observation dimensions. For each single convergence event, shear could be counteracted by a suitable (often nonlinear) transformation of coordinates. However, in practice one can only try out a restricted number of coordinate choices and transformations, which means that one cannot hope to see all transient attractors in a shear-free version. One has to accept shear and find ways to detect transient, sheared attractors.

- Empirical phase portraits are not filled by trajectories in a homogenous fashion. Rather, certain regions will be visited significantly more often than others. This reflects a probability distribution on the real system, which is projected on the observed system. Probabilities affect our intuitions about transient attractors. Consider the case of a geometrically "clean" transient attractor like in fig. 2a, which features two distinct bands where trajectories pass through much more often than elsewhere (fig. 2g). Should we consider this as a single, or two separate transient attractors? Our intuition (at least mine) doesn't give a clear answer. It would be nice to have a definition of transient attractors that can be used in both ways.

In the remainder of the subsection, I suggest an approach to transient attractors which accounts for the points above. Observed systems will be understood as stochastic processes, where the phenomenon of "converging trajectories" is re-interpreted in terms of temporal predictability.

Remember that our original motivation is to detect behaviors in phase space. On a purely intuitive level, behaviors are dynamic regularities in phase space. The approach I am going to formalize presently is simply an

information-theoretic, formal recasting of the "equation": *behavior = dynamic regularity in phase space = good predictability.*

The underlying intuition is illustrated in fig. 2h. When an observation yields the result "the system is in the region A at time 0", and A is in a region of converging trajectories, then one can predict "the system is in the interval B at time t", where the volume of B is less than the volume of A. This reduction in volume in time means that the process affords of a *good* predictability in A within a time t.

In what follows, I assume that the reader is familiar with the basic theory of stochastic processes. I adhere to the notation from a standard textbook [4]. Readers unfamiliar with the matter can skip the rest of the subsection. The algorithm for detecting transient attractors presented in the next subsection can be understood without the formal stochastic process model.

I consider an observed system as a continuous, stationary stochastic process $(\Omega, \mathfrak{A}, P, (X_t)_{t\in\mathbb{R}})$ with a measure space $(E, \mathfrak{B})$. The measure space is the observation space. E will typically be a compact subset of $\mathbb{R}^n$, and $\mathfrak{B}$ the Borel σ-algebra on it, with Lebesgue measure λ. Trajectories are paths $(X_t(\omega))_{t\in\mathbb{R}}$, where $\omega \in \Omega$.

We wish to make the notion of "good predictability" precise, in order to find a stochastic version of "converging trajectories". We embark on this business by considering first a simplicistic, preliminary version of "good predictability", which is yielded by the distribution of the process, as follows.

Let ν be the distribution of the process $(X_t)_{t\in\mathbb{R}}$ on E. We may assume that ν has a density function $f : E \to \mathbb{R}$ (this assumption is justified, by the theorem of Radon-Nikodym, if every $A \in \mathfrak{A}$ of λ-measure 0 has ν-measure 0, which in turn is granted, e.g., if the process is noisy). f affords us with "good" predictions of $(X_t)_{t\in\mathbb{R}}$ in the following sense. If we were to guess blindly where the process is at an arbitrary time, without knowledge about its past history, then we should point at regions where f takes high values. In a purely statistical sense of absolute, apriori expectancies, the process is "attracted" towards regions where f has high values.

One might wish to correlate, then, transient attractors with regions in phase space where f has high values. This idea could be made fully precise in various ways. For instance, one could define transient attractors in terms of cutoff values for f: a region A is a transient attractor of strength C iff it is connected and maximal such that $f > C$ in A. Other definitions would be likewise plausible. The important thing about this is not the particular way how we single out certain regions where f has high values. Rather, it is important that we have a scalar field f which we can exploit for our purposes.

Using f as a measure for "transient attractiveness" is moderately plausible since f increases when trajectories converge, possibly in a disorderly

fashion, as in the diagrams fig. 2a,b,c,f. Furthermore, f reflects densities of trajectories as in fig. 2g. But f has a serious shortcoming. It cannot disentangle superimposed transient attractors like in fig. 2d,e. Even worse, when we are confronted with several *diverging* strands of trajectories that happen to cross each other at some region, then f would assume high values in this area, without any convergence of trajectories being present. Revert the direction of trajectories in fig. 2d to see an example.

This situation calls for including the *direction* of trajectories into our account. In terms of predicting $(X_t)_{t\in\mathbb{R}}$, this means that we ask the question: given that at some point in time the process is in a state x, in which direction can we make good predictions for its further development?

This question can be stated in a precise way with Markov kernels. Let P_T be the Markov kernel on $(E, \mathfrak{B})$ which describes how the system develops during a time interval of length T under the condition that its state at the beginning of the development is known:

$$P_T : E \times \mathfrak{B} \to \mathbb{R},$$

such that

$$P_T(x, A) \quad = \quad P[X_T \in A | X_0 = x] \tag{4}$$

For fixed x and T,

$$\lambda A\ P_T(x, A) : \mathfrak{B} \to [0, 1]$$

is a probability measure on $(E, \mathfrak{B})$ (I adopt the lambda notation from logics although it is not commonly used in probability theory, because of the transparency it provides for denoting functions). Like before, we may safely assume that it affords of a density function $\lambda y\ f_{x,T}(y) : E \to \mathbb{R}^+$, where $\mathbb{R}^+$ denotes the nonnegative reals. In analogy to the absolute density f, this x-conditioned density tells us in which region in E we should predict the process to be in after a time lapse T, under the condition that at time zero it is in state x.

In order to arrive at a measure for directed predictability which is not confined to a particular starting point x, we let x vary. I.e., we consider the function

$$\lambda x \lambda y\ f_{x,T}(y) : E \times E \to \mathbb{R}^+$$

This function tells us *directions* of good predictability at temporal distance T. $f_{x,T}(y)$ being high means that the process is likely to proceed from

x to y in time T. Again, high values of this function indicate a convergence of trajectories, but this time, we can additionally distinguish between strands of convergent trajectories that cross each other in different directions (cf. fig. 2d), and we are not fooled any more by regions where divergent strands cross.

The selection of T determines the temporal resolution of regularities in phase space that we can detect. In order to detect high-frequency regularities, T must be selected small. Examining the system with increasing T, one can detect phenomena of longer-reaching conditioned predictability, i.e., larger- scale regularities in phase space. Considering $f_{x,T}(y)$ implies a low-pass filtering of the process at a frequency of T^{-1}.

If we observe long-lived robots that repeatedly perform similiar tasks in similar environments, we can assume the observed system to be ergodic. In that case, with increasing T the function $\lambda x \lambda y\ f_{x,T}(y)$ will eventually become independent from x and converge to the apriori density f in its second argument y. f can thus be considered an asymptotic case of $f_{x,T}(y)$.

Transient attractors can be derived from $f_{x,T}(y)$ in analogous fashions as from f, for example, in terms of regions in $E \times E$ where $f_{x,T}(y)$ exceeds some threshold C. Again, the important construct is the scalar field $f_{x,T}(y)$ on $E \times E$.

The fact that we have now a scalar field on $E \times E$ (and not simply on E) may seem bewildering at first sight. But even though one might certainly find other plausible formalizations of directed predictability than the one presented here, one can never make do with a simple scalar field on E. The reason is, obviously, that we wish to distinguish *different* directions that may occur in a *single* point in E.

We can get a graphically nice-looking variant of $f_{x,T}(y)$ on $E \times E$ by collecting, for each function $\lambda y\ f_{x,T}(y)$, the values $y_1, \ldots, y_n$ where the function $\lambda y\ f_{x,T}(y)$ assumes a local maximum (if any). For each x, this yields a collection of vectors $y_1 - x, \ldots, y_n - x$ (where n depends on x). Rescale these vectors in length to the corresponding local maxima values of $\lambda y\ f_{x,T}(y_i)$, and draw them in a diagram from x. A kind of "multi-vector field" appears that clearly shows the direction and the goodness of prediction. Fig. 3 gives an impression.

The task remains to deal with the case depicted in fig. 2e, namely, to distinguish between superimposed, convergent strands of trajectories on the grounds of their curvature. Even more generally, it would be nice to have something like a "Taylor series" of measures for predictability: absolute, directed, curvature-sensitive, etc. I will not further that line of reasoning here. Concerning the practical analysis of robots, the distinction of convergent strands of trajectories on the grounds of their direction should suffice.

Remember that a basic motivation for transient attractors was to re-

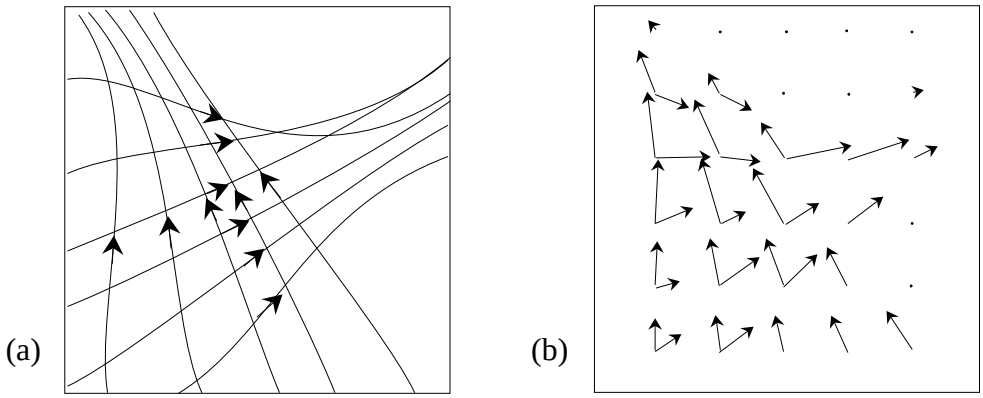

Figure 3: A system (a) and a "multi-vector field" (b) representation of the directed predictability in it.

concile two complementary approaches to fixing discrete observational units in continuous systems, namely, attractors and partitioning cells. This is achieved in transient attractors in that, first, they are regions in phase space where trajectories pass through (like partitioning cells), and in that, second, they exhibit converging trajectories (which makes them resemble attractors). Note that classical attractors can be considered asymptotic cases of regions in phase space of measure 0 where f becomes infinitely high.

This finishes the theoretical considerations about transient attractors. I have outlined a mathematical apparatus for defining transient attractors, which captures much of the complexity and untidiness found in empirical phase portraits. This is only a preliminary sketch, of course. Before one invests much more effort in working it out, one should first see whether the enterprise makes practical sense. To that end, I proceed now to an algorithm designed for practically detecting the areas of good predictability that I have described theoretically so far.

3.2 Sketch of an algorithm for finding transient attractors

In this subsection, I explain the basic idea for an algorithm that detects transient attractors in an empirical phase portrait yielded by the observation of a single long run of a robot, or of several shorter ones.

The algorithm detects regions in phase space that more or less correspond to transient attractors detected by the directed predictability measure $f_{x,T}(y)$ introduced in the preceding section. Presently, I can claim this cor-

respondence only on intuitive grounds. The algorithm's basic mechanism is clearly related to $f_{x,T}(y)$, but the details of the correspondence remain to be worked out. Therefore, I will ignore the issue of relating the algorithm to the preceding theoretical model, and simply present a self-contained explanation.

The algorithm proceeds in 5 steps. The crucial one (step 3) consists in operations on trajectories. The details of how these operations are executed depend on how trajectories are technically represented. I will not treat these issues. Instead, I will only explain the operations' essential effects.

Step 1 Select a time interval T. Just like in the preceding subsection, T determines the temporal resolution of the detection procedure. T should be as small as possible in order to give a good resolution, but great enough to warrant that T^{-1} is a lower frequency than the frequency of fluctuations that one wishes to treat as noise.

Step 2 Low-pass filter the original data for frequencies lower than T^{-1}. The result is a phase portrait whose trajectories are smooth below T^{-1} (for a nice example of how drastically this operation transforms a phase portrait of an infant's motor behavior, see [20]).

Step 3 This is the crucial step. It consists of some substeps and is iterated until convergence is achieved.

Substep 3.1 Select at random a point x on a trajectory in the phase portrait, and determine the point y passed by the trajectory after a lapse of time T after it has passed x.

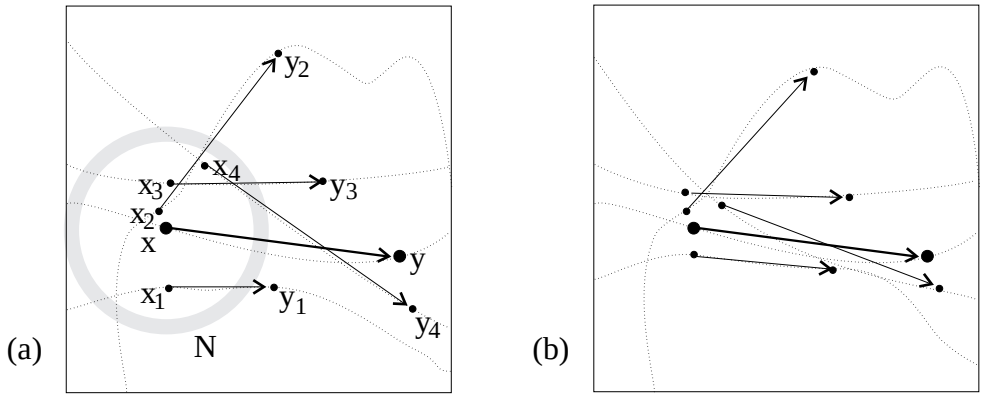

Figure 4: A local modification of the phase portrait. (a) Selection of trajectory segments as candidates for modification. (b) The phase portrait after the modification.

Substep 3.2 Determine the points $x_1, \ldots, x_n$, which lie closest to x on those n trajectories that pass through some neighborhood N of x. Determine the points $y_1, \ldots, y_n$ where those trajectories pass through after time T (fig. 4a).

Substep 3.3 For each $1 \leq i \leq n$, compute a real number $pred_i$ which measures how well the directed line segment through x_i, y_i is predicted from the line segment through x and y. The formula used for computing $pred_i$ is the core of the entire algorithm. Finding a good formula will certainly require some experimentation, or a theoretical clarification of the connections with the theoretical model. It seems reasonable to make the following requirements:

- The sign of $pred_i$ should be the same as the sign of the scalar product $(y - x)(y_i - x_i)$.
- The absolute value of $pred_i$ should inversely correlate with the distance between the two line segments.
- The absolute value of $pred_i$ should positively correlate with the agreement in length of both line segments.

Substep 3.4 Recompute trajectories for which $pred_i$ is positive such that in the vicinity of x they come closer to the reference trajectory through x and y, which remains unchanged. The result will be a locally modified phase portrait that should look somehow like fig. 4b. Again, the modification formula will need some experimentation. It should meet the following requirements:

- The trajectory through x_i, y_i should be modified only if $pred_i > 0$.
- The modification should be the stronger the greater $pred_i$.
- The modification should concern both the absolute distance of the line segment through x_i, y_i from the reference trajectory, and the difference in directions, both of which should be made smaller.
- The modification should not lead to high-frequency bends in the modified trajectories, i.e. it should be "smoothed out" a bit beyond the region locally concerned.

Iterate these substeps until the modifications induced by them fall under some threshold.

Hopefully, the iteration procedure should have compressed into narrow bands such segments of trajectories as have a good mutual, temporally directed predictiveness, as indicated in fig. 5.

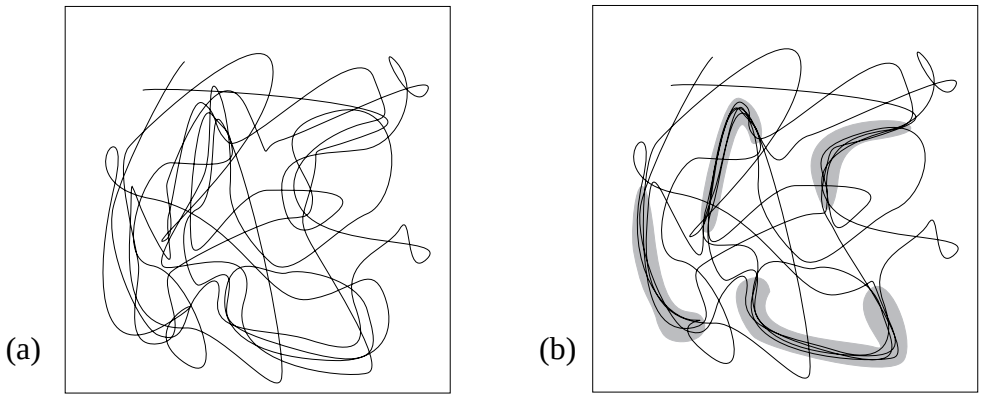

Figure 5: The original phase portrait (a) and the effects of many iterations of the local modification step (b).

Step 4 Identify in the modified phase portrait the compressed strands of trajectories. They represent transient attractors.

Step 5 Name theses transient attractors by symbols and compute the symbol sequence corresponding to the passage of the system trajectory through the transient attractors.

This is, of course, only a preliminary sketch. However, the endeavor seems to make sense since the basic idea is simple and transparent. It could be stated in a nutshell as follows: where trajectories show an inclination to attract each other, let them do it until they meet, and identify the meeting places with transient attractors.

4 Reconstructing a finite state generator for a symbol sequence

In the preceding section I dealt with the question of deriving a symbol sequence from an observed continuous process, in a fashion that takes serious the role of symbols as denoting dynamic "regularities". The natural next step in the analysis is to find higher-level regularities in the symbol sequence. This is usually done by finding a finite description of a generating law, in terms of a grammar or an automaton of some kind.

Grammars and automata belong to the theory of formal languages, i.e. the theory of sets of finite symbol sequences (words). Our working material is not a set of finite words. Rather, it is one potentially infinite symbol

sequence. In order to make the connection to formal languages, one can derive a language from such a single sequence in a straightforward manner: put the set of all finite subsequences to be the set of words of the language. This is the standard way of applying the tools of formal language theory to symbolic processes.

The most regular and simple kind of symbol sequences can be described by regular languages. I will describe symbol sequences generated by robots in terms of regular languages. The basic fact about such sequences or languages is that they are produced by mechanisms with a finite memory. The standard formal model of such mechanisms is finite automata. For almost all physical and biological symbolic processes it is natural to assume that the underlying mechanisms has a finite memory, which can be, however, very large. The same holds for processes generated by robots, which have bounded resources, in bounded environments. It is "ontologically" justified to model them using regular languages.

Regular languages have been extensively investigated within the theory of formal languages and automata theory [12]. Within the ergodic theory of symbolic dynamical systems, the corresponding processes have been termed sofic systems [32]. A somewhat more general class of processes, coded systems, has also been investigated. An introduction can be found in [6], where the connections to the theory of formal languages are discussed, too. Regular languages have found fresh applications in the investigation of self-organization in theoretical physics, where they are used to yield complexity measures for processes undergoing bifurcations [11] [10]. The work carried out in this area is particularly interesting for our purposes, since it explicitly addresses the question of finding simple, standardized, finite descriptions of empirical, symbolic processes.

This section has two subsections. In the first subsection, I describe a certain difficulty that arises when standard normal forms of finite automata are used to describe stationary processes. I show how this difficulty can be resolved by considering a subclass of regular languages that corresponds to stationary processes, and which affords of a canonical automata model including a normal form theorem. In the second subsection, I describe an effective procedure for constructing this normal form automaton from empirical data.

4.1 Phase generators

I assume that the reader is familiar with finite automata and the normal form theorem, which states how a given non-deterministic or deterministic finite automaton can be transformed into an equivalent, deterministic, *minimal* automaton, which is a uniquely determined normal form for all finite

automata generating a given regular language (e.g., [12]).

Grassberger [11] mentions in passing a little problem that occurs when symbolic dynamical systems are modeled by finite automata. I reformulate and accentuate this problem in the following little dilemma:

- On the one hand, when one describes symbolic processes that yield potentially infinite sequences, one is typically *not* interested in transient behavior that occurs only at the onset of such sequences. Quite to the contrary, one wishes to understand the regularities that rule the system after initial special effects have died out and the process has become stationary. Initial transients are uninteresting since they tell us more about the essentially arbitrary starting conditions than about the dynamic mechanism itself.

- On the other hand, finite automata explicitly model initial transients. They feature a special *initial state*, from which transient passages may lead in an irreversible way into a true substructure of the automaton which models the long-term behavior. Unfortunately, this can occur in the minimal automaton even when the language concerned actually does not contain any special initial transients. This situation is illustrated in fig. 6a. In other words, due to the formal requirement of a special initial state, finite automata may contain an annoying artefact that blurs the transparency of this kind of representation. This becomes particularly cumbersome if the number of states of minimal automata is used for measuring the inherent complexity of the symbolic process – which is done all the same (e.g., in [10]), due to the lack of a better alternative.

For the purpose of describing stationary symbolic processes, an automaton representation without a special initial state would be desirable. For instance, the process (or the corresponding language, equivalently) described by the finite automaton in fig. 6a should be represented by a transition graph like in 6b. If only one had a normal form theorem for such transition graphs, the dilemma mentioned above could be resolved.

I have worked out this approach in some detail in [13], where I also developed a normal form for transition graphs like the one in fig. 6b. I called such graphs *generators* then, a usage that I will adhere to now. I shall briefly review the work here (a more elaborated introduction can be found in [14]). In addition, I formulate and prove a new proposition that gives a handy characterization of normal form representations.

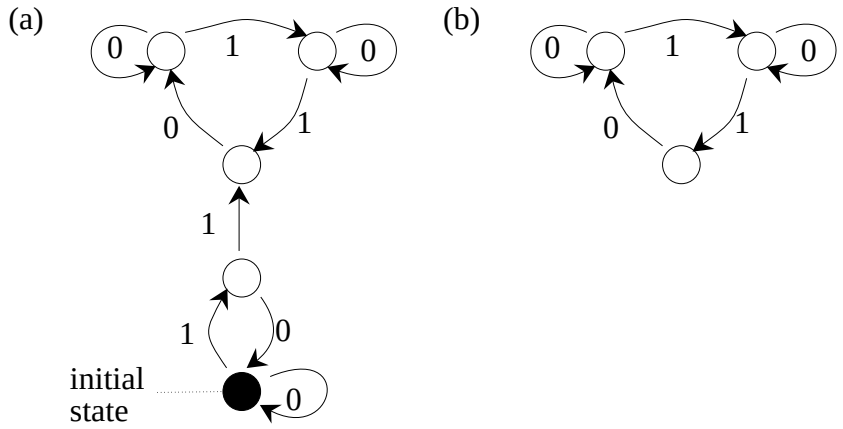

Figure 6: Two representations for the regular language consisting of words that feature 1's and 11's interspaced by arbitray numbers of 0's, such that the number of 1's between two occurences of 11's is even. (a) Minimal deterministic finite automaton. (b) Transition graph without a specified initial state. (Taken from [11])

Let Σ be a finite alphabet. A *generator* is a finite, cyclic, directed graph, whose edges are labeled by symbols from Σ. Fig. 6b gives an example of a generator, where $\Sigma = \{0, 1\}$. More formally, we define:

Definition 4.1 A *generator* (more precisely, a Σ-generator) is a pair $G = (S, trans)$, where S is the set of *states*, and $trans \subseteq S \times \Sigma \times S$ is the set of *transitions*. We use the shorthand $s_i a s_j$ for $(s_i, a, s_j) \in trans$. The *language generated by* G is the set

$$L(G) = \{a_1 \dots a_n | n \geq 0, \exists s_0, \dots, s_n \in S : s_0 a_1 s_1 \wedge \dots \wedge s_{n-1} a_n s_n\}$$

The sequence of transitions $s_0 a_1 s_1, \dots, s_{n-1} a_n s_n$ is called a *derivation* of $a_1 \dots a_n$ in G. G can be interpreted as a directed, edge-labelled graph in an obvious way. If this graph is finite, connected, and cyclic, then $L(G)$ is called a *coherent* language. We use the letter C (instead of L) to denote coherent languages.

Remark For reasons that do not concern us here, I have termed the states of a generator *local states* in [13] and elsewhere.

Notation We use boldface letters **a** to denote words of a formal language, and write **ab** for the concatenation of **a** and **b**. We write **a**s if a derivation

of $\mathbf{a}$ exists that ends in state s; $s\mathbf{a}$ if a derivation of $\mathbf{a}$ exists that begins in state s; and $t\mathbf{a}s$ if a derivation of $\mathbf{a}$ exists that begins in state t and ends in state s.

Coherent languages are a proper subclass of the regular languages. From the remarks above it should be clear how they correspond to stationary processes.

A finite subsequence $a_1 \ldots a_n$ of a symbolic process contains some information about how the process might continue. If such sequences are prolonged to the left (i.e., we learn more about the past), our knowledge about possible continuations into the future can only grow. However, due to their finite memory, in coherent languages the information about the future can always be maximized through knowledge about a *finite* portion of the past. This notion of maximally predictive, finite sequences is formalized in the notion of *phases*:

Definition 4.2 Let C be a coherent language, $\mathbf{a} \in C$. The set

$$\mathit{continue}(\mathbf{a}) := \{\mathbf{b} \in C | \mathbf{ab} \in C\}$$

is the set of *continuations* of $\mathbf{a}$ in C. $\mathbf{p} \in C$ is *phase-fixing* (i.e., maximally predictive in the sense mentioned above) iff $\mathit{continue}(\mathbf{rp}) = \mathit{continue}(\mathbf{p})$ for all $\mathbf{rp} \in C$. $\mathbf{p}, \mathbf{q} \in C$ are *phase-equivalent* iff they are phase-fixing and $\mathit{continue}(\mathbf{p}) = \mathit{continue}(\mathbf{q})$. The equivalence classes are called *phases*. The phase represented by $\mathbf{p}$ is denoted by $\varphi_{\mathbf{p}}$. The set of all phases of C is denoted by $\Phi(C)$.

Some basic properties of phases are collected in the following

Proposition 4.3

1. $\forall \mathbf{a} \in C \exists \mathbf{b} \in C$: $\mathbf{ba}$ is phase-fixing.
2. $\Phi(C)$ is finite.
3. If $\mathbf{p}$ is phase-fixing, and $\mathbf{rp} \in C$, then $\mathbf{rp}$ is phase-fixing, and $\varphi_{\mathbf{p}} = \varphi_{\mathbf{rp}}$.
4. If $\mathbf{p}$ is phase-fixing, and $\mathbf{pr} \in C$, then $\mathbf{pr}$ is phase-fixing.

The simple proof can be found in [13]. Phases can be interpreted as the states of a particular generator:

Definition 4.4 For a coherent language C, $G_\varphi(C) = (\Phi(C), \mathit{trans}_\varphi)$ is the *phase generator* of C, where

$$trans_\varphi := \{(\varphi_{\mathbf{p}}, a, \varphi_{\mathbf{p}a}) | \mathbf{p} \text{ is phase-fixing}, \mathbf{p}a \in C\}.$$

The phase generator is the desired normal form for generators of a coherent language. An algorithm for constructing the phase generator from an arbitrary generator of C is described in [13]. Phase generators are a fundamental tool in the study of coherent languages. In particular, $C_1 \subseteq C_2$ iff $G_\varphi(C_1)$ can be mapped to $G_\varphi(C_2)$ in a certain manner [13].

Let me note in passing that the underlying idea of phase generators, namely, maximizing predicitiveness, nicely agrees with our approach to defining regularities in continuous systems in the preceding section. We have been led there by the very same motive, albeit in a quite different mathematical frame.

I conclude this subsection with a helpful characterization of phase generators:

Proposition 4.5 A generator $G = (S, trans)$ of a coherent language C is the phase generator iff it has the following properties:

1. $trans$ is deterministic in the following sense:

$$\forall a \in \Sigma \, \forall s_1, s_2, s_3 \in S : s_1 a s_2 \wedge s_1 a s_3 \Rightarrow s_2 = s_3$$

2. Every state $s \in S$ is fixed by some $\mathbf{a} \in C$ in the following sense:

$$\forall s \in S \, \exists \mathbf{a} \in C \, \forall t \in S : \mathbf{a}s \wedge \mathbf{a}t \Rightarrow s = t$$

3. States can be distinguished by their futures:

$$\forall s, t \in S : s = t \leftrightarrow s^+ = t^+$$

Here, s^+ denotes the set $\{\mathbf{a} \in C | s\mathbf{a}\}$.

Proof "$\Rightarrow$" is an easy exercise. I treat only the hard direction "$\Leftarrow$". For a phase φ let $\varphi^+ := \{\mathbf{a} \in C | \varphi\mathbf{a} \text{ is a derivation in } G_\varphi\}$. We show first that the following statement holds:

$$\forall s \in S \, \exists \varphi \in \Phi : s^+ = \varphi^+ \tag{5}$$

Let **a** fix s in G. By proposition 4.3(1), some $\mathbf{b} \in C$ and some $\varphi \in \Phi$ exist such that **ba** fixes φ. **ba** still fixes s in G. Since G and G_φ both generate C, we can conclude $s^+ = \varphi^+ = \{\mathbf{c} \in C | \mathbf{bac} \in C\}$, i.e., (5).

Let φ_s denote the phase corresponding to s according to (5). We show what amounts to the converse of (5):

$$\forall \varphi \in \Phi \; \exists s \in S : \varphi = \varphi_s \tag{6}$$

Let $\mathbf{a} \in C$ be a representative of φ, i.e. $\varphi = \varphi_{\mathbf{a}}$. Select $s, t \in S$ such that $t\mathbf{a}s$ in G. Let t be fixed in G by **b**. From the determinism property (premise 1) it follows that s is fixed by **ba** in G. Since **a** is phase-fixing, $continue(\mathbf{a}) = continue(\mathbf{ba})$ holds, from which $\varphi^+ = s^+$ follows, with (6) as an immediate consequence.

From property (3) it follows that (5) and (6) yield a bijection between S and Φ. It is easy to confirm that this bijection is in fact an isomorphism between G and G_φ. □

Using this proposition, one can easily verify that the transition graph in fig. 6b actually is a phase generator.

4.2 Reconstructing the phase generator from an empirical process

Assume that an empirical observation of a stationary, stochastic process has yielded a long symbol sequence $T = c_1 \ldots c_N$. This trajectory has been generated by an unknown mechanism which we assume to have finite memory. Therefore, T can be modeled approximately (i.e., up to what one considers noise) by a coherent language. In this subsection, I describe a procedure for reconstructing the corresponding phase generator from T.

Before we begin, we should be aware of two principal limitations:

- Any effective reconstruction algorithm can only capture regularities that reveal themselves within a bounded temporal range. No algorithm can guarantee anything more than that the reconstructed phase generator accounts for all subsequences of T up to some length l.

- Empirical trajectories are noisy. Since we can work with only a finite trajectory T, we are principally unable to distinguish "true noise" from "rare regularities". I.e., we have to decide for a basically arbitrary cut between which (rare) subsequences we wish to be explained by the generator model, and which others (even more rare) we wish to discard as noise.

Taken together, these two limitations mean that the best we can hope for is reconstructing a phase generator that accounts for a good deal of the regularities in T, but leaves an unexplained rest of nonconforming subsequences. This rest has a "true noise" component and an "overlooked regularity" component, the ratio of which we cannot estimate without further assumptions about statistical properties of the mechanism generating T.

The procedure for reconstructing a phase generator has two stages. In the first stage, information about to predicting bounded futures from bounded pasts is extracted from T, and the data is purged from noise and/or the effects of rare regularities. In the second stage, the phase generator is actually constructed from this preprocessed material.

Stage 1 First we introduce a notational convention: write $\mathbf{a} < \mathbf{b}$ to denote the fact that $\mathbf{a}$ is a subword of $\mathbf{b}$.

Remember that the rationale behind phases and phase generators is the prediction of future developments from a given past sequence. A finite algorithm can consider only finite pasts and futures. Therefore, the first thing to do is to fix two nonzero natural numbers, p and f, which determine the depth of past and future time spans to be taken into account. These parameters should be choosen rather small at the first attempt (say, $p = f = 2$), since this means that a relatively simple phase generator will be constructed. At a later point in stage 1, this choice is tested whether it leads to ignoring important longer-term regularities; if it does, stage 1 loops back to the present point and greater values for p and/or f are choosen.

Let $Pred := \{\mathbf{a} < c_1 \ldots c_{N-f} | |\mathbf{a}| = p\}$ be the set of all subsequences of length p in T that can be used for predicting f further time steps in T. For $\mathbf{a} \in Pred$, let $F_{\mathbf{a}} := \{\mathbf{b} \in \Sigma^f | \mathbf{ab} < T\}$ be the set of continuations of length f of $\mathbf{a}$ which can be observed in T, i.e. the "future of depth f" of $\mathbf{a}$ in T.

At this point, one might find it advisable to purge the data from what one considers noise. A quick and dirty method is to discard from $Pred$ sequences $\mathbf{a}$ that occur below some threshold average frequency, and consequently to discard from each $F_{\mathbf{a}}$ such $\mathbf{b}$ as contain already discarded $\mathbf{a}$ as a subsequence (assuming that $p \leq f$; in case that $f < p$, reverse the order of discarding accordingly).

Now consider the futures of the subsequences $\mathbf{a}$ after they have been continuated by one symbol. More precisely, for every $\mathbf{a} \in Pred$, and $\mathbf{a}a < T$, consider $F_{\mathbf{a}a} := \{\mathbf{b} \in \Sigma^f | \mathbf{a}a\mathbf{b} < T\}$. (Discard, if necessary, such $\mathbf{a}a$ as contain subsequences that have already been discarded earlier.)

Check whether the following holds:

$$\forall F_{\mathbf{a}a}\, \exists F_{\mathbf{a}'} : F_{\mathbf{a}a} = F_{\mathbf{a}'} \;\wedge\; \forall F_{\mathbf{a}}\, \exists F_{\mathbf{a}'a} : F_{\mathbf{a}} = F_{\mathbf{a}'a} \tag{7}$$

A significant violation of (7) means that the future (of f steps) can be predicted significantly better from a past of depth $p+1$ than from a past of depth p. If this situation occurs, increment p by 1, and repeat the entire procedure described so far. Iterate this until (7) approximately holds.

Validity of (7) can have two reasons. First, it may indicate that we have choosen a sufficient depth p of premises $\mathbf{a}$ for prediction. Second, it may also indicate that we have chosen an insufficient depth f to distinguish between different kinds of future developments. In order to exclude this possibility, we increment f by 1 and rerun everything. We repeat the inner loop of incrementing p and the outer loop of incrementing f until (7) stays approximately valid under both kinds of increments.

If this has been achieved, we have a good chance (albeit no certainty) that premises of length p enable good predictions, and that finite futures of depth f are good indicators for "complete", unbounded futures.

Next we define, via some suitable criterion for approximate identity of sets, an equivalence relation $\sim$ on the set $\{F_{\mathbf{a}}|\mathbf{a} \in Pred\} \cup \{F_{\mathbf{a}a}|\mathbf{a} \in Pred, \mathbf{a}a < T\}$ which satisfies $F_{\mathbf{x}} \sim F_{\mathbf{y}}$ iff $F_{\mathbf{x}}$ is approximately equal to $F_{\mathbf{y}}$ ($\mathbf{x}, \mathbf{y} \in Pred \cup \{\mathbf{a}a|\mathbf{a} \in Pred, \mathbf{a}a < T\}$). This yields equivalence classes $\mathcal{F}$. We write $\mathcal{F}_{\mathbf{x}}$ to indicate that $F_{\mathbf{x}} \in \mathcal{F}$. The classes $\mathcal{F}$ represent the "pure" types of future developments that we assume to exist after the effects of noise have been cleared away. We call $Fut := \{\mathcal{F}_{\mathbf{a}}|\mathbf{a} \in Pred\}$ the set of future types in T. As a consequence of (7), every $\mathcal{F}$ should contain representatives both of the kind $F_{\mathbf{a}}$ and $F_{\mathbf{a}a}$.

This ends stage 1. The result can be summed up as follows. A future development out of the class $\mathcal{F}$ is predicted by an observation of $\mathbf{x}$ iff $F_{\mathbf{x}} \in \mathcal{F}$. The effects of noise have been removed, and a fairly justified decision on the proper depths p and f of finite approximations to pasts and futures has been made.

Stage 2 A phase generator $G_\varphi = (\Phi, trans_\varphi)$ which approximately regenerates T can easily be constructed as follows, using the results from stage 1:

Procedure 4.6

Step 1 Define a relation $trans_0 \subseteq Fut \times \Sigma \times Fut$ by

$$(\mathcal{F}, a, \mathcal{F}') \in trans_0 \quad \text{iff} \quad \exists F_{\mathbf{a}} \in \mathcal{F} : F_{\mathbf{a}a} \in \mathcal{F}'.$$

In other words, $trans_0$ defines a Σ-transition graph on the set of nodes Fut.

Step 2 Select a node $\mathcal{F}_{\mathbf{a}}$ such that $F_{\mathbf{a}}$ is minimal as a set, i.e. such that

$$F_{\mathbf{b}} \subseteq F_{\mathbf{a}} \Rightarrow F_{\mathbf{b}} = F_{\mathbf{a}}.$$

Collect all nodes $Fut^* \subseteq Fut$ that are transitively reachable from $\mathcal{F}_{\mathbf{a}}$ via transitions taken from $trans_0$. Put $trans^* := trans_0 \cap Fut^* \times \Sigma \times Fut^*$. This should yield a cyclic transition graph. Finally, put $G_\varphi = (\Phi, trans_\varphi) := (Fut^*, trans^*)$.

If step 2 does not yield a cyclic transition graph, then either the empirical process is not stationary or is not generated by a mechanism with finite memory, or something has gone awry in stage 2. The latter can have several reasons. For instance, a transition that is necessary for making the transition graph cyclic might have been discarded due to its rare occurence. It may also be the case that selecting p or f too small excludes crucial long-range effects. Corresponding strategies for recovery would be to repeat the procedure with a longer T, or to further increment p and f.

The procedure is correct in the following sense:

Proposition 4.7 If the following conditions hold:

1. T is a word from a coherent language C,
2. for all phases φ of C a phase-fixing $\mathbf{a}$ exists such that $|\mathbf{a}| \leq p$,
3. for all $\mathbf{x}, \mathbf{y} \in C, |\mathbf{x}| \geq p, |\mathbf{y}| \geq p$ it holds that $continue(\mathbf{x}) = continue(\mathbf{y})$ iff $\mathbf{x}$ and $\mathbf{y}$ agree in their possible continuations up to a depth f,
4. in stage 1, nothing is discarded, i.e. it is assumed that there is no noise in T, and
5. all words of C of length $p + f$ occur as subwords in T,

then the procedure described above yields the phase generator of C.

Sketch of proof The premises (4) and (5) ensure that what can be learnt about C from considering words of length $p + f$ is faultlessly and completely gleaned from T in stage 1. On this background the proposition can be proved rather easily.

Step 1: Show that G_φ is a substructure of the graph $(Fut, trans_0)$.

To this end, we first observe that a phase φ fixed by $\mathbf{p} \in C$, where $|\mathbf{p}| \leq p$, corresponds uniquely to a future $\mathcal{F}_{\mathbf{p}}$. In this sense, we can write $\mathcal{F}_\varphi$ to denote

the future of φ. Let $(\varphi, a, \psi) \in trans_\varphi$. Use the premises (2) and (3) and proposition 4.3(4) to conclude that $(\mathcal{F}_\varphi, a, \mathcal{F}_\psi) \in trans_0$.

Step 2: Show that $(\mathcal{F}_\varphi, a, \mathcal{F}_\psi) \in trans_0$ implies $(\varphi, a, \psi) \in trans_\varphi$.

$(\mathcal{F}_\varphi, a, \mathcal{F}_\psi) \in trans_0$ implies that some $\mathbf{p} \in C$ exists which fixes φ, such that $\mathcal{F}_\varphi = \mathcal{F}_\mathbf{p}$ and $\mathcal{F}_\psi = \mathcal{F}_{\mathbf{p}a}$. Use property (1) from proposition 4.5 to conclude $(\varphi, a, \psi) \in trans_\varphi$.

The statements of steps 1 and 2 imply that the reduct of $(Fut, trans_0)$ on the nodes $\mathcal{F}_\varphi$ corresponding to phases is isomorphic to the phase generator.

Step 3: Show that in $(Fut, trans_0)$ no transition leads away from the phase generator substructure, i.e., $(\mathcal{F}_\varphi, a, \mathcal{F}) \in trans_0$ implies $\mathcal{F} = \mathcal{F}_\psi$ for some phase ψ.

This is essentially a consequence of proposition 4.3(4).

The node $\mathcal{F}_\mathbf{a}$, selected in step 2 of procedure 4.6 such that $F_\mathbf{a}$ is minimal as a set, corresponds to a phase, i.e., $\mathcal{F}_\mathbf{a} = \mathcal{F}_\varphi$ for some phase φ. This is implied by a basic fact about coherent languages, namely, that if $continue(\mathbf{a})$ is minimal among all such continuation sets in C, then $\mathbf{a}$ is phase-fixing. Therefore, the selection in step 2 of procedure 4.6 hits a node that belongs to the phase generator substructure of $(Fut, trans_0)$. The statements of steps 1 – 3 above ensure that the transitive collection of nodes and transitions in step 2 of procedure 4.6 yields the phase generator, and nothing more. □

4.3 The entropy of processes corresponding to coherent languages

A fundamental characteristic of a process is its *entropy*, which provides a measure for its predictability, or rather, its unpredictability. In this section I show how phase generators can be used for a simple computation of the (metric) entropy of symbolic processes corresponding to coherent languages.

The metric entropy measures predictability in a way that accounts for the effects of transition *probabilities* between successively produced symbols (in contrast to the simpler topological entropy, which shuns probabilistic information). Therefore, before we can start our computation of metric entropies, we must generalize our notion of generators to make them probabilistic:

Definition 4.8 A *probabilistic generator* is a pair $G = (S, trans, P_{trans})$, where S and $trans$ are defined as in definition 4.1, and $P_{trans} : trans \to [0, 1]$ provides each transition $s_i a s_j$with a *transition probability* $P_{trans}(s_i a s_j)$ such that for all states s_i it holds that $\sum_{s_i a s_j} P_{trans}(s_i a s_j) = 1$.

The procedure for constructing phase generators from an empirical tra-

jectory, as described in the previous subsection, can easily be accomodated to yield probabilistic phase generators.

Without going into details here (complete treatment in [19], pp. 51ff), let me note that cyclic generators (i.e., generators of coherent languages) describe ergodic processes, from which in turn it can be concluded that a uniquely determined probability exists which measures with which relative frequency the states of a probabilistic generator are visited on long derivation paths. Formally, this probability is a function $P_{state} : S \to (0,1]$ which satisfies $\sum_{s_i \in S} = 1$. P_{state} can be effectively computed from P_{trans} by a straightforward iteration procedure.

The metric entropy H of a symbolic process T generated by a probabilistic phase generator $G_\varphi = (\Phi, trans_\varphi, P_{trans_\varphi})$ can be computed as follows:

Proposition 4.9

$$H(T) = -\sum_{\varphi \in \Phi}(P_{state}(\varphi) \sum_{\varphi a \psi \in trans_\varphi} P_{trans_\varphi}(\varphi a \psi) \log_2 P_{trans_\varphi}(\varphi a \psi))$$

Proof The proof is an exercise in rearranging (and LATEXing) sums. We start with the well-known fact (cf. [9]) that $H(T)$ can be computed from the block entropies H_n of sequences of length n (logarithms are to base 2 throughout):

$$\begin{aligned} H(T) &= \lim_{n\to\infty}(H_n(T) - H_{n-1}(T)) \\ &= \lim_{n\to\infty} -\sum_{\mathbf{a}\in C^n} P(\mathbf{a}) \log P(\mathbf{a}) + \sum_{\mathbf{b}\in C^{n-1}} P(\mathbf{b}) \log P(\mathbf{b}), \end{aligned} \tag{8}$$

where C^n denotes the set of all sequences $\mathbf{a}$ of length n that can occur in the process, and $P(\mathbf{a})$ denotes the relative frequency of $\mathbf{a}$ among all sequences with the same length as $\mathbf{a}$.

Observing that for $n \to \infty$ almost all sequences $\mathbf{a} \in C^n$ are phase-fixing, i.e., that $\lim_{n\to\infty} \sum_{\mathbf{a}\in C^n, \mathbf{a} \text{ is phase-fixing}} P(\mathbf{a}) = 1$, for large n the following rewriting of the first sum in (8) is admissible:

$$\begin{aligned} &\sum_{\mathbf{a}\in C^n} P(\mathbf{a}) \log P(\mathbf{a}) = \\ &= \sum_{\mathbf{b}\in C^{n-1}} \sum_{a\in\Sigma} P(\mathbf{b}) P_{trans_\varphi}(\varphi_{\mathbf{b}} a \psi) \log P(\mathbf{b}) P_{trans_\varphi}(\varphi_{\mathbf{b}} a \psi) \end{aligned} \tag{9}$$

We can now proceed from (9) with some simpler transformations

$$
\begin{aligned}
&= \sum_{\varphi\in\Phi}\sum_{a\in\Sigma}\sum_{\mathbf{b}\in C^{n-1},\mathbf{b}\text{ fixes }\varphi} P(\mathbf{b})P_{trans_\varphi}(\varphi a\psi)\log P(\mathbf{b})P_{trans_\varphi}(\varphi a\psi)\\
&= \sum_{\varphi}\sum_{a}\sum_{\mathbf{b}} P(\mathbf{b})P_{trans_\varphi}(\varphi a\psi)\log P(\mathbf{b})\\
&\quad + \sum_{\varphi}\sum_{a}\sum_{\mathbf{b}} P(\mathbf{b})P_{trans_\varphi}(\varphi a\psi)\log P_{trans_\varphi}(\varphi a\psi)\\
&= \sum_{\varphi}\sum_{a} P_{trans_\varphi}(\varphi a\psi)\sum_{\mathbf{b}} P(\mathbf{b})\log P(\mathbf{b})\\
&\quad + \sum_{\varphi}\sum_{a}\sum_{\mathbf{b}} P(\mathbf{b})P_{trans_\varphi}(\varphi a\psi)\log P_{trans_\varphi}(\varphi a\psi)\\
&= \sum_{\mathbf{b}\in C^{n-1}} P(\mathbf{b})\log P(\mathbf{b}) \qquad (10)\\
&\quad + \sum_{\varphi}\sum_{a}\sum_{\mathbf{b}} P(\mathbf{b})P_{trans_\varphi}(\varphi a\psi)\log P_{trans_\varphi}(\varphi a\psi)
\end{aligned}
$$

Inserting (10) into (8) yields

$$
\begin{aligned}
H(T) &= \lim_{n\to\infty} -\sum_{\varphi\in\Phi}\sum_{a\in\Sigma}\sum_{\mathbf{b}\in C^{n-1},\mathbf{b}\text{ fixes }\varphi} P(\mathbf{b})P_{trans_\varphi}(\varphi a\psi)\log P_{trans_\varphi}(\varphi a\psi)\\
&= \lim_{n\to\infty} -\sum_{\varphi\in\Phi}\sum_{\mathbf{b}\in C^{n-1},\mathbf{b}\text{ fixes }\varphi} P(\mathbf{b})\sum_{a\in\Sigma} P_{trans_\varphi}(\varphi a\psi)\log P_{trans_\varphi}(\varphi a\psi)\\
&= -\sum_{\varphi\in\Phi} P_{state}(\varphi)\sum_{\varphi a\psi\in trans_\varphi} P_{trans_\varphi}(\varphi a\psi)\log P_{trans_\varphi}(\varphi a\psi) \qquad \square
\end{aligned}
$$

5 Higher-order regularities and the behavior model

The phase generator yields a transparent model of temporal chaining of "elementary" behaviors corresponding to elementary regularities in the phase portrait. Animal behavior is, however, organized hierachically, including behaviors that are complex compounds of elementary ones. For instance, in many social species a behavior element "eye contact" exists, which can occur as part of a more complex "threat" behavior, which in turn can appear within ritual rank order fights. It would be desirable to describe behaviors on several levels in robots, too.

This has been recognized as a central issue in behavior-oriented robotics [26]. The key word is "emergent behavior". As yet, however, no formal criterium exists for what it means for a behavior to emerge from others.

In the third stage of analysis, the phase generator describing elementary behaviors is successively "coarsened" to yield higher-level, complex behaviors. Relatively well predictable patterns of activity, in which several elementary behaviors participate, yield new behavior units on a higher level of description.

The mathematics for the third stage is not yet worked out as well as for the first two stages. Therefore, I shall merely illustrate the main ideas with an example.

Fig. 7a) shows a fictitious phase generator that might have come out of the second stage of analysis. The ciphers at the transitions indicate the empirical transition probabilities.

The edge labels in fig. 7 are "telling names". I have used them in order to make the diagram more readable. Of course, a formal analysis of time series data cannot come up with telling names. They can be substituted for formal labels only after a re-inspection and interpretation of the robot's performance by a human observer. Although it is important from a methodological angle, in this article I will not enter a discussion of questions concerning the intuitive interpretation and subsequent naming of formally detected behaviors.

The behavior model from fig. 7 represents (simplified and adapted) a robot that has actually been built, and finely demonstrated at the KI '95, by students at the University of Bielefeld. A human observer would describe the robot's behavior more or less as follows:

- Most of the time, the robot drives around in the arena "searchingly". It copes with obstacles by bumping into them, retracting, and moving forward again in a different direction.

- Upon contact with a "treadmill", which is recognized through a particular touch sensor, the robot begins to "work" by pushing around in a circle the radial pushbar of the treadmill device. Deviations from optimal pushing position are detected by light sensors, and become corrected. If contact with the mill is lost, the searching behavior is taken up again.

- After a prolonged period of activity, the robot makes its way back to the charging station and recharges. Then in starts anew with its searching behavior.

This kind of intuitive behavior description belongs to a higher, coarser grained level than the fine-grained model yielded by the phase generator from

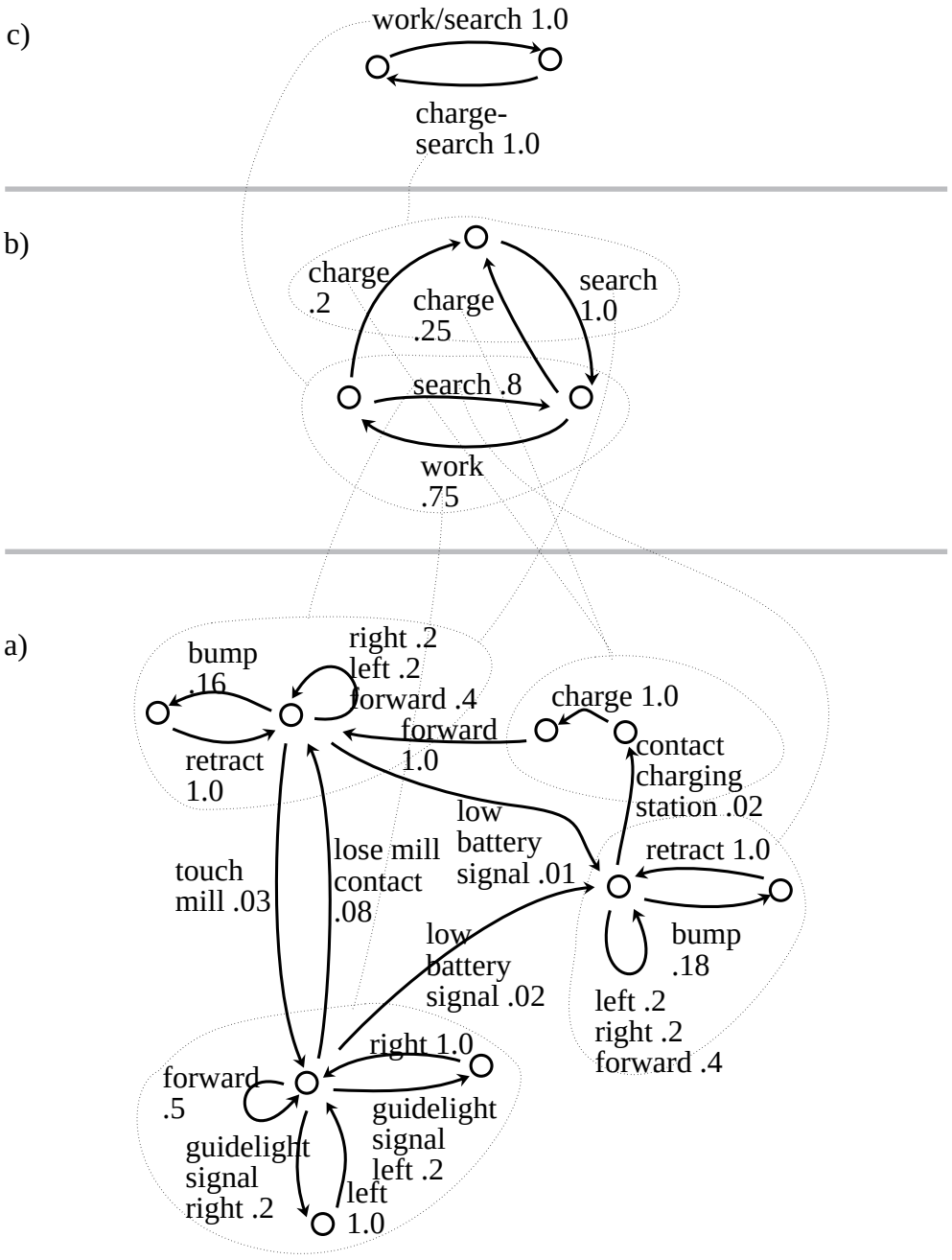

Figure 7: An emergence hierarchy of behaviors. The elementary level (a) is described by a phase generator, as derived in the second stage of analysis. On higher levels (b) and (c), emergent behaviors of increasing complexity are described by other phase generators. See text for explanations.

the second stage of analysis (fig. 7a). Within that fine-grained model, one can identify higher-level regularities by exploiting the transition probabilities. Again, we use a criterium of good predictability to characterize them. The basic idea is to single out *sets* of nodes in the phase generator, and regard them as higher-level units of behavior, according to the following strategy (cf. fig. 8a):

If it is known that the process is in a certain set A of states [$A = \{1, 2\}$ in fig. 8a], then require that

- it can be predicted with high probability that it will transit into a set B of successor states of A [$B = \{3, 4, 5\}$ in fig. 8a], and
- it can be predicted with high probability that it will *not* transit into the set C [here: $C = \{6, 7, 8\}$] of those successor states of A that are not contained in B, and
- the pairwise transition probabilities from A into B all lie in the same order of magnitude.

This criterion intuitively fixes transitions between sets of states, which (the transitions) are well separated from the neighborhood of other possible transitions (points 1 and 2). We require also that the transition between sets is "mixing" from A to B (point 3), i.e., prediction from a single state in A to a single state in B is poor (except if B contains only one state, of course).

Emergent regularities, then, are defined through the transitive iteration of such "group transitions". Each maximally possible sequence of iterations of such group transitions (fig. 8b) yields an emergent regularity in the phase generator, which in turn represents an emergent, higher-level behavior.

Some noteworthy special cases exist: internally tightly linked groupings of states (fig. 8c), "channelled" transients (fig. 8d), the analogue of cyclic attractors (fig. 8e), or the analogue of point attractors (fig. 8f). Very satisfyingly, this leads us back to the basic idea of the first stage of analysis, where we likewise combined transients with attractors in order to arrive at a generalized notion of dynamic regularities.

This criterion of course still has to be worked out in formal detail. However, one can already apply it informally to the phase generator from fig. 7a. The emergent behaviors that can be thus fixed are encircled in fig. 7a by dotted lines. Intuitively, they correspond to the emergent behaviors "searching", "working", and "recharging". Searching and working behaviors are of the kind depicted in fig. 8c, whereas recharging is of the transient type fig. 8d.

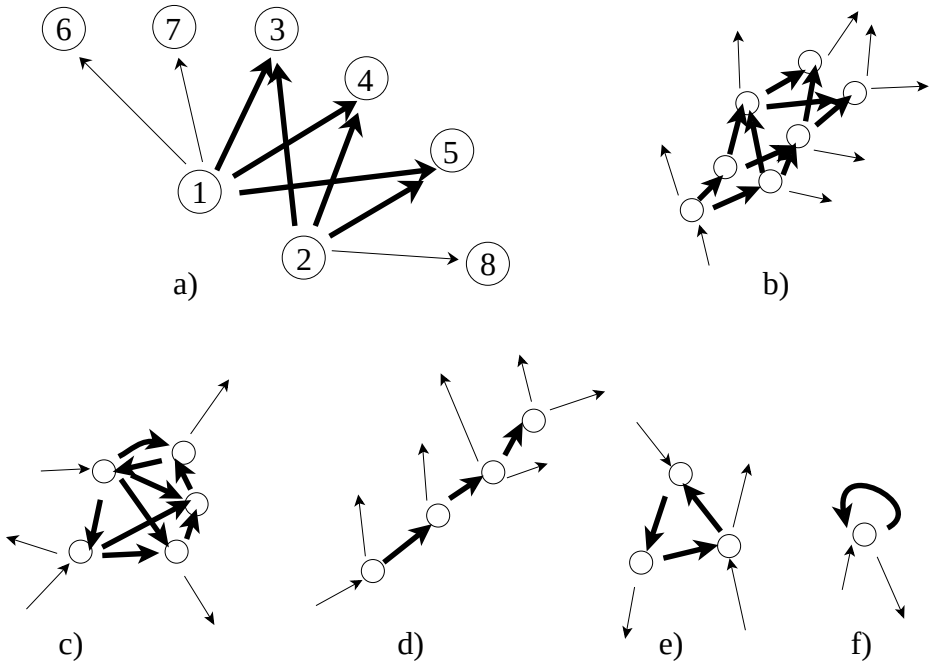

Figure 8: Criterion for detecting higher order regularities. Stylized portions of phase generators are shown: the basic scheme (a), and variations (b – f). The magnitude of transition probabilities is indicated by the arrows' boldness. For further explanations compare text.

Given these emergent behaviors, a higher-level phase generator (fig. 7b) can be constructed straightforwardly, which represents transitions between the emergent behaviors.

The entire procedure can be iterated. On the next (and ultimate, in this example) level we find the phase generator from fig. 7c.

Taken together, the three phase generators from fig. 7 represent the *behavior model*, which is the final result of the three-stage analysis procedure.

6 Conclusion

I have outlined a principled, three-stage approach to detecting regularities in empirical phase portraits of robots. In the first stage, elementary regularities are identified, which can be named and thus yield a symbol sequence. Temporal regularities in this sequence are then detected in the second stage, which yields a normal form automaton model of the robot's behavior se-

quence. In the third stage, higher-order regularities are determined, resulting in a final, hierarchic behavior model. In each of the three stages, I have been led by the idea that "regularity" means "something good for predicting what will happen next". Thus, the entire approach can be considered an information-theoretic attempt to bridging the gap between the quantiative and the symbolic level of describing physical agents.

Behavior-oriented robotics still lacks a principled, mathematical methodology. This article is a contribution to putting the field on a firmer mathematical ground. I believe, furthermore, that the techniques presented in this article are of interest for other subfields of AI where symbol grounding in some sense or the other is at stake.

The algorithms presented here are sketches only. Taking them to practice will undoubtedly require experimentation. Together with Thomas Christaller, I have submitted for funding a research project where these algorithms are to be integrated in an automated analysis tool for the evaluation of behavior-based robots.

Acknowledgments This work has been made possible by a working contract from GMD, St. Augustin. I wish to thank Thomas Christaller for his support. I am indebted to Gregor Schöner for valuable comments on standards in dynamical systems terminology.

References

[1] Vera A.H. and H.A. Simon. Situated action: A symbolic interpretation. *Cognitive Science*, 17(1):7–48, 1993.

[2] A. Babloyantz and C. Lourenço. Computation with chaos: A paradigm for cortical activity. *Proceedings of the National Academy of Sciences of the USA*, 91:9027–9031, 1994.

[3] Ch. Balkenius and P. Gärdenfors. Nonmonotonic inferences in neural networks. In J.A. Allen, R. Fikes, and E. Sandewall, editors, *Principles of Knowledge Representation and Reasoning. Proceedings of the 2nd int. Workshop*, pages 32–39. Morgan Kaufmann, San Mateo, 1991.

[4] H. Bauer. *Wahrscheinlichkeitstheorie und Grundzüge der Maßtheorie.* de Gruyter, Berlin/New York, 3 edition, 1978.

[5] R. Beer. A dynamical systems perspective on agent-environment interaction. *Artificial Intelligence*, 72(1/2):173–216, 1995.

[6] F. Blanchard and G. Hansel. Systèmes codés. *Theoretical Computer Science*, pages 17–49, 1986.

[7] G.A. Carpenter and S. Grossberg. ART 3: Hierarchical search using chemical transmitters in self- organizing pattern recognition architectures. *Neural Networks*, 3(2):129–152, 1990.

[8] D.J. Chalmers. Connectionism and compositionality: Why Fodor and Pylyshyn were wrong. *Philosophical Psychology*, 6:305–319, 1993.

[9] J.P. Crutchfield and N.H. Packard. Noise scaling of symbolic dynamics entropies. In H. Haken, editor, *Evolution of Order and Chaos in Physics, Chemistry, and Biology*, pages 215–227. Springer Verlag, Berlin/Heidelberg/New York, 1982.

[10] J.P. Crutchfield and K. Young. Computation at the onset of chaos. In W.H. Zurek, editor, *Complexity, Entropy, and the Physics of Information*, volume VIII of *SFI Studies in the Sciences of Complexity*, pages 223–269. Addison-Wesley, 1990.

[11] P. Grassberger. Toward a quantitative theory of self-generated complexity. *Int. J. of Theor. Physics*, 25(9):907–938, 1986.

[12] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, Mass., 1979.

[13] H. Jaeger. Dynamic symbol systems. Ph.d. thesis, Faculty of Technology, University of Bielefeld, 1994.

[14] H. Jaeger. An introduction to dynamic symbol systems. In J. Hallam, editor, *Hybrid Problems, Hybrid Solutions. Proceedings of the AISB- 95*, pages 109–120. IOS Press/Ohmsha, Amsterdam, 1994.

[15] H. Jaeger. On modelling behaviors and concepts as attractors. In *On the Role of Dynamics and Representation in Adaptive Behaviour and Cognition (DRABC-94)*, pages 171–173. Universidad del Pas Vasco, San Sebastian, 1994.

[16] H. Jaeger. Dynamische Systeme in der KI und ihren Nachbarwissenschaften. Arbeitspapiere der GMD 925, GMD, St. Augustin, 1995.

[17] H. Jaeger. Modulated modules: designing behaviors as dynamical systems. Arbeitspapiere der GMD 927, GMD, St. Augustin, 1995.

[18] C.X. Ling and R. Buchal. Learning to control dynamic systems with automated quantization. In P.B. Brazdil, editor, *Machine Learning. Proceedings of the ECML-93*, Lecture Notes in Artificial Intelligence 667, pages 372–377. Springer Verlag, Berlin, 1993.

[19] K. Petersen. *Ergodic Theory*. Cambridge University Press, 1983.

[20] S.S. Robertson, A.H. Cohen, and G. Mayer-Kress. Behavioral chaos: Behind the metaphor. In L.B. Smith and E. Thelen, editors, *A Dynamic Systems Approach to Development: Applications*, pages 119–150. Bradford/MIT Press, Cambridge, Mass., 1993.

[21] G. Schöner, M. Dose, and C. Engels. Dynamics of behavior: theory and applications for autonomous robot architectures. *Robotics and Autonomous Systems*, page to appear, 1995.

[22] G. Schöner, H. Haken, and J.A.S. Kelso. A stochastic theory of phase transitions in human hand movement. *Biological Cybernetics*, 53:247–257, 1986.

[23] T. Smithers. What the dynamics of adaptive behavior and cognition might look like in an agent-environment interaction system. In *Proceedings of the Workshop 'On the Role of Dynamics and Represenation in Adaptive Behavior and Cognition'*, pages 135–153. Universidad del Pais Vasco, San Sebastian, 1994.

[24] T. Smithers. What the dynamics of adaptive behavior and cognition might look like in agent-environment interaction systems. In R. Pfeifer, editor, *Practice and Future of Autonomous Agents. Workshop Notes of the NATO ASI on Autonomous Agents, Ascona*. Computer Science Dpt., University of Zurich, 1995.

[25] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1*, pages 194–281. MIT Press, Cambridge, Mass., 1986.

[26] 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.

[27] L. Steels. Mathematical analysis of behavior systems. In P. Gaussier and J.-D. Nicoud, editors, *From Perception to Action*, pages 88–95. IEEE Computer Society Press, Los Alamitos, CA, 1994.

[28] J. Tani. Embedding a grammatical description in deterministic chaos: An experiment in recurrent neural learning. *Biological Cybernetics*, 72:365–370, 1995.

[29] P. Tino, B.G. Horne, and C.L. Giles. Finite state machines and recurrent neural networks – automata and dynamical systems approach. Technical report umiacs-tr-95-1 and cs-tr-3396, Institute for Advanced Computer Studies, University of Maryland, 1995.

[30] T. van Gelder and R. Port. Beyond symbolic: Toward a kama-sutra of compositionality. In V. Honavar and L. Uhr, editors, *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration*, pages 107–125. Academic Press, 1994.

[31] D. L. Waltz and J.B. Pollack. Massively parallel parsing: A strongly interactive model of natural language interpretation. *Cognitive Sience*, 9:51–74, 1985.

[32] B. Weiss. Subshifts of finite type and sofic systems. *Monatshefte für Mathematik*, 77:462–474, 1973.

[33] Y. Wilks. A preferential, pattern-seeking semantics for natural language inference. *Artificial Intelligence*, 6:53–74, 1975.

[34] Y. Yao and W.J. Freeman. A model of biological pattern recognition with spatially chaotic dynamics. *Neural Networks*, 3(2):153–170, 1990.