Artificial Intelligence 72 (1995) 173–215

# A dynamical systems perspective on agent-environment interaction

Randall D. Beer*

*Dept. of Computer Engineering and Science and Dept. of Biology, Case Western Reserve University, Cleveland, OH 44106, USA*

Received September 1992; revised March 1993

**Abstract**

Using the language of dynamical systems theory, a general theoretical framework for the synthesis and analysis of autonomous agents is sketched. In this framework, an agent and its environment are modeled as two coupled dynamical systems whose mutual interaction is in general jointly responsible for the agent's behavior. In addition, the adaptive fit between an agent and its environment is characterized in terms of the satisfaction of a given constraint on the trajectories of the coupled agent-environment system. The utility of this framework is demonstrated by using it to first synthesize and then analyze a walking behavior for a legged agent.

## 1. Introduction

This paper is concerned with properly characterizing the interaction between an autonomous agent and its environment. By *autonomous agent*, I mean any embodied system designed to satisfy internal or external goals by its own actions while in continuous long-term interaction with the environment in which it is situated. The class of autonomous agents is thus a fairly broad one, encompassing at the very least all animals and autonomous robots. An animal, for example, may simply be trying to survive, while a robot might be designed to carry out specific tasks, such as keeping some designated area clean or exploring the surface of another planet. The task is thus to abstract over particular details of implementation and embodiment (e.g., nerve cells vs. finite state machines or muscles vs. motors) in order to understand the essential character of this class of systems.

* E-mail: beer@alpha.ces.cwru.edu


*SSDI* 0004-3702(94)00005-L

However short of this ambitious goal the present paper may fall, the long-term aim is nothing less than a general theoretical framework for the explanation and design of autonomous agents.

The central problem for any autonomous agent, and thus the primary concern in this paper, is the generation of the appropriate behavior at the appropriate time as both its internal state and external situation continuously change. For an embodied agent, action must always take precedence over any other activity. Abstract reasoning, when it can be afforded at all, is profitable only insofar as it is ultimately reflected in improved behavior. This does not necessarily imply that an embodied agent must be purely reactive, reflexively responding only to its immediate situation. Rather, it means an autonomous agent must be able to flexibly combine its immediate circumstances with its long-term goals so as to continuously adjust its behavior in ways appropriate to both. An animal moving throughout its environment, for example, needs to adopt many different modes of behavior as it becomes hungry or tired and encounters potential food, predators and mates, all the while adjusting its posture and leg movements to the constantly changing terrain which it is traversing.

Traditionally, such "low level" concerns of embodiment have not played a major role in AI research. Instead, work in AI has tended to emphasize "high level" intellectual skills, such as language, problem solving and abstract reasoning. Embodied agents, when they have been considered at all, have been viewed as merely symbolic reasoning engines with sensors and effectors attached. Accordingly, the problems of embodied agents have usually been formulated as special cases of the problems of disembodied intelligent systems. Of course, it has long been realized that embodiment raises certain additional technical issues. Sensors, for example, introduce the problem of constructing, and maintaining the consistency of, internal representations of the environment from physical signals, while effectors introduce the problem of translating representations of action into actual motor commands. Furthermore, physical embodiment introduces real-time constraints on an agent's action that limit the amount of time that can be spent reasoning. However, these technical problems are usually seen as merely complicating, rather than invalidating, the classical picture.

In recent years, however, a growing number of AI researchers have begun to appreciate the fundamental importance of embodiment. There are a number of reasons for this change in perspective. Designing agents that can interact with the real world with the versatility and robustness of even simple animals has turned out to be considerably more subtle and difficult than originally realized, and approaches developed for disembodied agents have not in general translated well to the noisy, unpredictable, open-ended and dynamic nature of the real world. Furthermore, many problems that seemed intractable for disembodied systems have turned out to be considerably simplified by active participation in an environment. Work on animate vision, for example, has demonstrated that a number of visual problems are drastically simplified if the agent is given the ability to control its own gaze direction [8]. Likewise, work on situated agents has shown that the potentially brittle and combinatorially explosive nature of general

planning can be significantly alleviated by using the immediate situation to guide behavior [3, 20]. Indeed, there is a growing realization that, far from being a mere complication for a disembodied intellect, embodiment may in fact be the more fundamental issue. Certainly, from an evolutionary perspective, the human capacity for language and abstract thought is a relatively recent elaboration of a much more basic capacity for situated action that is universal among animals.

This reassessment of the importance of embodiment has led to an explosion of recent work on autonomous agents. Brooks, working in the area of mobile robotics, was one of the first AI researchers to point out the limitations of classical AI techniques in the face of real-world complexity and the need for a renewed emphasis on embodiment and situated action [18, 20]. Agre and Chapman's work on routine activity grew out of a similar frustration with the limitations of classical planning in realistic environments and led to similar conclusions [2, 3, 22]. Building on Rosenschein's situated automata theory [56], Rosenschein and Kaelbling developed methods for deriving finite state machine controllers for an agent from a formal specification of its knowledge and goals [42]. My own work [11, 14] and that of Cliff [24] has demonstrated the significant potential for interaction between autonomous agent research and work on the neural basis of animal behavior. Biological design principles have also been stressed by Arbib and Liaw [5]. Surveys of recent work in autonomous agents can be found in the collections edited by Maes [45] and Meyer and Wilson [49].

This body of work is loosely characterized by a number of shared ideas. It emphasizes the primacy of actually taking action in the world over the abstract descriptions that we sometimes make of it. It focuses on the development of complete agents capable of carrying out open-ended tasks in unconstrained environments rather than isolated cognitive skills in restricted task domains. It emphasizes behavior and the fundamental importance that the immediate situation plays in guiding an agent's behavior, ideas historically associated with behaviorism, over reasoning and symbolic models of the world. Another common theme of this work has been that a significant fraction of behavior must be seen as emerging from the ongoing interaction between an agent and its environment, an idea often associated with cybernetics. This work has also begun to question the central role that internal representation has been assumed to play in intelligent behavior by most work in cognitive science.

In this paper, I will attempt to show that these and other ideas that are emerging from recent work on autonomous agents, as well as work on the neural basis of animal behavior, can be naturally cast into the language of dynamical systems theory. Furthermore, I will argue that this language can form the basis for a powerful theoretical framework for the explanation and design of autonomous agents in general. Section 2 reviews some of the basic concepts of dynamical systems theory that are required to present this framework. In Section 3, I sketch the theoretical framework itself and draw out some of its conceptual consequences. Section 4 demonstrates the utility of this framework by illustrating in some detail its application to the synthesis and analysis of a walking behavior for a simulated legged agent. Finally, Section 5 discusses the assumptions of the

proposed framework, considers its broader implications, and suggests some directions for future work.

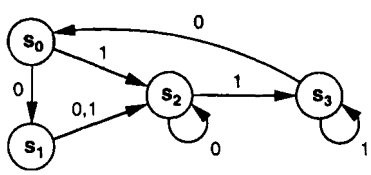
## 2. Dynamical systems

This section will briefly review the essential concepts and terminology of the qualitative theory of dynamical systems that will be required for the theoretical framework to be presented in Section 3. The presentation is necessarily informal and incomplete. The reader interested in a more thorough treatment should refer to one of the many available texts on dynamical systems including, in order of increasing mathematical sophistication, the books by Abraham and Shaw [1], Hale and Koçak [33] and Wiggins [66].

Consider the following three mathematical systems:

$$x_{n+1} = \mu x_n(1 - x_n) \tag{1}$$

$$ml\frac{d^2\theta}{dt^2} + \gamma\frac{d\theta}{dt} + mg\sin\theta = A\cos(\omega t) \tag{2}$$




(3)

At first glance, these three systems may appear to have very little in common. The first equation is an example of an iterated map. The second system is a second-order differential equation describing the motion of a damped, sinusoidally-driven pendulum. The third system is a finite state machine with input. However, all of these systems are also instances of *dynamical systems* and the underlying similarity of many of the questions one might ask about each of these systems only becomes apparent in this formalism.

For our purposes here, a dynamical system is characterized by a set of *state variables* $\boldsymbol{x}$ and a *dynamical law* $\mathcal{F}$ that governs how the values of those state variables change with time. The set of all possible values of the state variables constitutes the system's *state space*. For simplicity, I will assume here that the state space is continuous (i.e., the state variables are real-valued), though most of the concepts that I will be introducing hold in some form for any metric space. Often the geometry of the state space is assumed to be simply Euclidean. Sometimes, however, other spaces arise. Perhaps the most common examples are cylindrical and toroidal state spaces, which occur naturally when some of the state variables are periodic (e.g., $\theta$ in Eq. (2)).

If the dynamical law depends only upon the values of the state variables and the values of some set of fixed *parameters* $\boldsymbol{u}$, then the system is said to be

*autonomous*.[1] In a continuous-time dynamical system, the dynamical law is given in terms of a set of differential equations: $\dot{\boldsymbol{x}} = \mathcal{F}(\boldsymbol{x}; \boldsymbol{u})$. In this case, the dynamical law defines a *vector field* on the state space. In a discrete-time dynamical system, the dynamical law is simply a map from current state to next state: $\boldsymbol{x}_{n+1} = \mathcal{F}(\boldsymbol{x}_n; \boldsymbol{u})$. A dynamical system is said to be *linear* or *nonlinear* according to the linearity or nonlinearity of $\mathcal{F}$ in the state variables.

As a concrete example, consider the following system of equations, which describe the behavior of a fully-interconnected network of two simple model neurons:

$$\begin{aligned} \dot{y}_1 &= -y_1 + w_{11}\sigma(y_1 - \theta_1) + w_{21}\sigma(y_2 - \theta_2)\,, \\ \dot{y}_2 &= -y_2 + w_{12}\sigma(y_1 - \theta_1) + w_{22}\sigma(y_2 - \theta_2)\,, \end{aligned} \tag{4}$$

where $y_i$ is the state of the $i$th neuron, $\sigma(\xi) = (1 + e^{-\xi})^{-1}$ is the standard sigmoidal (S-shaped) activation function, $\theta_i$ controls the offset or threshold of the activation function and $w_{ij}$ is the weight of the connection from the $i$th to the $j$th neuron. This system is a simplification of a common neural network model that will be employed in Section 4. Note that this is a nonlinear system due to the nonlinearity of the activation function $\sigma$.

Starting from some *initial state* $\boldsymbol{x}_0$, the sequence of states generated by the action of the dynamical law is called a *trajectory* of the system and is often denoted $\phi_t(\boldsymbol{x}_0)$. A trajectory has the property that its tangent at each point is equal to the value of the vector field at that point. Some representative trajectories of the two-node network (4) are shown in Fig. 1. The set of all such trajectories through every point in the state space is called the *flow*, denoted $\phi_t$. In the classical theory of differential equations, one is typically interested only in individual solutions, which correspond to individual trajectories of the dynamical system. In contrast, in the qualitative theory of dynamical systems, one is usually more interested in the geometrical or topological structure of the entire flow.

Of particular interest is the possible long-term behavior of a dynamical system. The state of some systems will simply diverge to infinity (e.g., the system $\dot{y} = y$), while others will eventually converge to *limit sets*. A limit set is a set of points that is invariant with respect to the dynamical law, so that if the state of a dynamical system ever falls on a limit set, the action of the dynamical law will keep it there indefinitely. The *stable* limit sets or *attractors* are especially important. An attractor has the property that the trajectories passing through all nearby states converge to the attractor. This means that if the state of the system is perturbed a sufficiently small distance away from an attractor, the action of the dynamical law will bring the state back to the attractor. The open set of initial states that converge to a given attractor is termed its *basin of attraction*. Those portions of the trajectories through such points which do not lie on the attractor itself are

[1] This is a technical term in dynamical systems theory whose meaning is unrelated to its use in the term "autonomous agents"

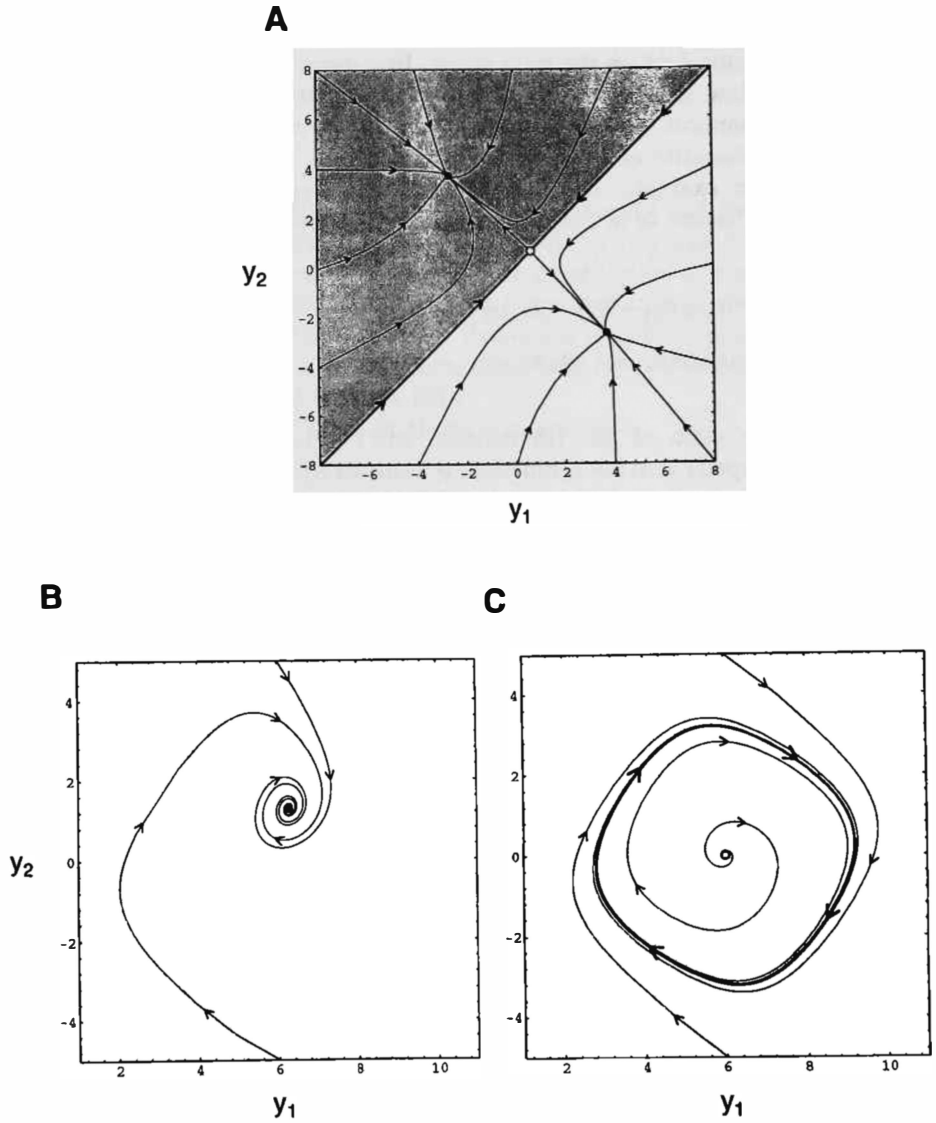



Fig. 1. Phase portraits for the two-neuron system (Eq. (4)) for several parameter settings. All of these systems are structurally stable, that is, the qualitative structures of their phase portraits persist for small variations in the parameters. (A) Here the system exhibits two stable equilibrium points near $(-3, 4)$ and $(4, -3)$ with basins of attraction shown in gray and white respectively. The open point near $(0, 0)$ is a repellor and the inset of this repellor (dark diagonal line) forms the separatrix between the two basins of attraction. The parameter values are $w_{11} = w_{22} = 4$, $w_{12} = w_{21} = -3$, $\theta_1 = \theta_2 = 0$. (B) Here the system exhibits a single equilibrium point attractor. The parameter values are $w_{11} = 2.75$, $w_{12} = -6$, $w_{22} = w_{21} = 6$, $\theta_1 = 6$, $\theta_2 = 0$. (C) Here the system exhibits a limit cycle (dark oval) and a repellor (open point near $(6, 0)$). This phase portrait was derived from the one shown in B by increasing the single parameter $w_{11}$ from 2.75 to 6.

called *transients*. The solid point near (−3,4) in Fig. 1(A) is an example of an attractor, and its basin of attraction is shown in gray.

*Repellors* are limit sets that are *unstable*. Repellors have the property that at least some nearby trajectories diverge from them. Despite the fact that the repellor itself is invariant, if the state of the system is perturbed even an infinitesimal distance from the repellor, the action of the dynamical law will serve to carry it away. The open point near (0, 0) in Fig. 1(A) is an example of a repellor. Attractors are important because they govern the long-term behavior of any physical system. Regardless of the initial state, a physically embodied dynamical system will always be found near an attractor after transients have passed. Due to their instability, repellors can only be observed by starting a dynamical system on a repellor and then never perturbing it. Since any physical system has some noise, it could never stay on a repellor indefinitely.

Four major classes of attractors are usually distinguished. *Equilibrium point* attractors are stable limit sets which consist of single points, such as the solid point near (4, −3) in Fig. 1(A). An equilibrium point $\boldsymbol{x}^*$ represents a constant solution to the system: $\phi_t(\boldsymbol{x}^*) = \boldsymbol{x}^*$. *Periodic* attractors or *limit cycles* are stable limit sets which are closed trajectories in the state space. These correspond to periodic or oscillatory solutions, with the property that $\phi_t(\boldsymbol{x}^*) = \phi_{t+T}(\boldsymbol{x}^*)$ for some minimum period $T > 0$ and any point $\boldsymbol{x}^*$ lying on the attractor. An example of a limit cycle is shown in Fig. 1(C).

The remaining two classes of limit sets, *quasiperiodic* attractors and *chaotic* attractors, are much more complicated than either equilibrium points or limit cycles. Chaotic attractors, for example, possess a fractal structure and they exhibit a sensitive dependence on initial conditions. No matter how closely two unequal initial states are chosen, their resulting trajectories can diverge exponentially even while remaining bounded on the attractor until they become completely uncorrelated. For this reason, despite the underlying determinism of its dynamical law, the behavior of a chaotic attractor is in many ways indistinguishable from a random process. While I will not discuss quasiperiodic or chaotic attractors further in this paper, it is important to realize that such complicated behavior is quite common in higher dimensional nonlinear dynamical systems.

In general, the state space of a dynamical system will contain multiple attractors, each surrounded by its own basin of attraction. These basins are separated by unstable manifolds called *separatrices*. The dark diagonal line separating the white and gray basins of attraction of the two equilibrium point attractors in Fig. 1(A) is an example of a separatrix. Thus one can visualize these separatrices as dividing the flow of a dynamical system into a number of "cells" each containing an attractor of some type. A global characterization of this cellular structure is called the *phase portrait* of the system (Fig. 1).

We have been holding the parameters $\boldsymbol{u}$ of the dynamical law $\mathcal{F}$ constant and considering the global structure of the resulting flow. What happens when these parameters are changed? Since $\mathcal{F}$ is a function of $\boldsymbol{u}$, the vector field that it determines, and hence the resulting flow $\phi_t$ that this vector field induces on the state space, will most certainly change as these parameters are varied. Thus a

parameterized dynamical law actually defines a family of dynamical systems, with any particular flow corresponding to a single setting of the parameters.

Just as we were previously interested in the structure of any given flow in state space, we can now inquire into the structure of a family of flows in parameter space. Most dynamical systems are *structurally stable*, that is, for most parameter settings, small changes in the parameter values will produce small changes in the flow. Limit sets and basins of attraction may deform and move around a bit, but the new flow will be qualitatively similar (i.e., topologically equivalent, or *homeomorphic*) to the old one. However, at certain parameter values, dynamical systems can become *structurally unstable*, so that even infinitesimal changes in parameter values can cause drastic changes in the flow, producing phase portraits that are qualitatively different from the original. These qualitative changes in the types of limit sets are called *bifurcations*.

For example, as the parameter $w_{11}$ in our example system (4) is increased from 2.75 to 6, the equilibrium point attractor shown in Fig. 1(B) loses its stability and bifurcates into the repelling point and limit cycle shown in Fig. 1(C) (the actual bifurcation, and therefore the structurally unstable flow that separates these two structurally stable flows, occurs around a $w_{11}$ value of 3.25). Much more complicated bifurcations can occur. Thus, just as we can visualize separatrices as dividing the state space of any given dynamical system into basins of attraction of different attractors, we can think of the sets of bifurcation points corresponding to structurally unstable flows as dividing the parameter space of a family of dynamical systems into different structurally stable flows.

Up to this point, we have only considered autonomous dynamical systems, that is, systems in which the parameters have been held fixed for the duration of any particular trajectory. What happens when these parameters are allowed to vary in time as the trajectory evolves? A *nonautonomous* dynamical system is one in which one or more parameters are allowed to vary in time: $\dot{\boldsymbol{x}} = \mathcal{F}(\boldsymbol{x}; \boldsymbol{u}(t))$. We can think of such parameters as *inputs* to the system. Because, as described above, the flow is a function of the parameters, in a nonautonomous dynamical system the system state is governed by a flow which is changing in time (perhaps drastically if the parameter values cross bifurcation points in parameter space). Nonautonomous systems are much more difficult to characterize than autonomous ones unless the input has a particularly simple (e.g., periodic) structure. In the nonautonomous case, most of the concepts that we have described above (e.g., attractors, basins of attraction, etc.) apply only on timescales small relative to the timescale of the parameter variations. However, one can sometimes piece together a qualitative understanding of the behavior of a nonautonomous system from an understanding of its autonomous dynamics at constant inputs and the way in which its input varies in time.

## 3. A theoretical framework

The qualitative theory of dynamical systems allows one to build up a global understanding of both the possible behaviors of a dynamical system and the

dependence of those behaviors on external parameters even when the solutions have no closed-form expression in terms of elementary mathematical functions. In this section, I will use this formalism to sketch a theoretical framework for characterizing the interaction between autonomous agents and their environments. Only the basic framework will be described here. Some sample applications of the framework will be presented in Section 4, and Section 5 discusses the assumptions of the proposed framework and considers some of its broader implications and directions for future work. This framework owes a great debt to the perspective that the cybernetic tradition has long taken on many of these same questions. I have been particularly influenced by the work of Ashby [6] and Maturana and Varela [47, 48].

### *3.1. Agents and their environments*

Following Ashby [6], I will model an agent and its environment as two dynamical systems $\mathscr{A}$ and $\mathscr{E}$, respectively. I will assume that $\mathscr{A}$ and $\mathscr{E}$ are continuous-time dynamical systems: $\dot{\boldsymbol{x}}_{\mathscr{A}} = \mathscr{A}(\boldsymbol{x}_{\mathscr{A}}; \boldsymbol{u}_{\mathscr{A}})$ and $\dot{\boldsymbol{x}}_{\mathscr{E}} = \mathscr{E}(\boldsymbol{x}_{\mathscr{E}}; \boldsymbol{u}_{\mathscr{E}})$. In addition, I will assume that both $\mathscr{A}$ and $\mathscr{E}$ have convergent dynamics, that is, the values of their state variables do not diverge to infinity, but instead eventually converge to some limit set. Note that the division between an agent and its environment is somewhat arbitrary (e.g., is an artificial limb or a tool part of the agent or part of the environment?) and therefore our theoretical framework should not depend overly much on the exact nature of this division. Our first act as scientific observers is to partition the world into individual components whose interactions we seek to understand, and there are many different ways to do this. For example, it will sometimes be convenient to view an agent's body as part of $\mathscr{A}$ and sometimes as part of $\mathscr{E}$.

An agent and its environment are in constant interaction. Formally, this means that $\mathscr{A}$ and $\mathscr{E}$ are coupled nonautonomous dynamical systems. In order to couple two dynamical systems, we can make some of the parameters of each system functions of some of the state variables of the other. I will represent this coupling with a sensory function $\boldsymbol{S}$ from environmental state variables to agent parameters and a motor function $\boldsymbol{M}$ from agent state variables to environmental parameters. $\boldsymbol{S}(\boldsymbol{x}_{\mathscr{E}})$ corresponds to an agent's sensory inputs, while $\boldsymbol{M}(\boldsymbol{x}_{\mathscr{A}})$ corresponds to its motor outputs. Thus, we have the following (Fig. 2):

$$\begin{aligned} \dot{\boldsymbol{x}}_{\mathscr{A}} &= \mathscr{A}(\boldsymbol{x}_{\mathscr{A}}; \boldsymbol{S}(\boldsymbol{x}_{\mathscr{E}}); \boldsymbol{u}'_{\mathscr{A}}) , \\ \dot{\boldsymbol{x}}_{\mathscr{E}} &= \mathscr{E}(\boldsymbol{x}_{\mathscr{E}}; \boldsymbol{M}(\boldsymbol{x}_{\mathscr{A}}); \boldsymbol{u}'_{\mathscr{E}}) , \end{aligned} \tag{5}$$

where $\boldsymbol{u}'_{\mathscr{A}}$ and $\boldsymbol{u}'_{\mathscr{E}}$ represent any remaining parameters of $\mathscr{A}$ and $\mathscr{E}$ respectively that do not participate in the coupling. I will assume that this coupled agent-environment system also exhibits only convergent dynamics.

Note that I am using the terms "sensory input" and "motor output" in a fairly broad sense here. $\boldsymbol{S}$, for example, is intended to represent *all* effects that $\mathscr{E}$ has on $\mathscr{A}$, whether or not this influence occurs through what is normally thought of as a sensor. This breadth of usage is justified by the observation that any such effect

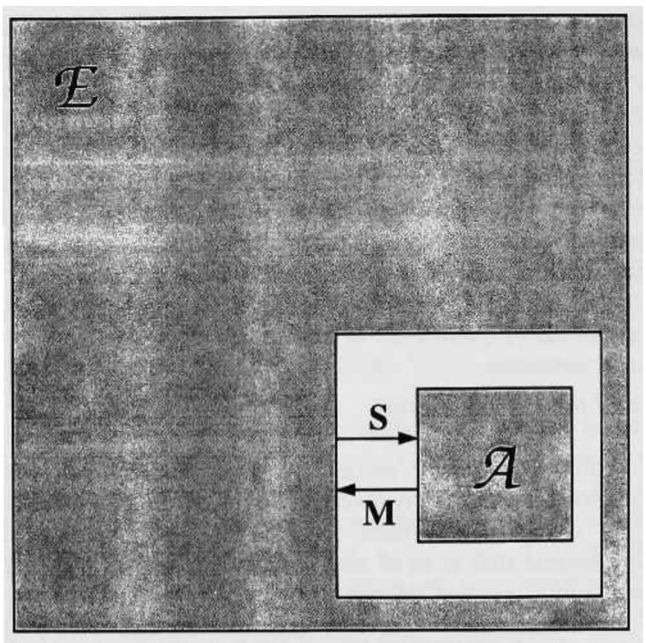



Fig. 2. An agent and its environment as coupled dynamical systems.

can influence the subsequent trajectory of $\mathcal{A}$. Likewise, $M$ is intended to represent all effects that $\mathcal{A}$ has on $\mathcal{E}$, whether or not they occur through what is normally thought of as an effector.

I cannot overemphasize the fundamental role that feedback plays in this relationship. Any action that an agent takes affects its environment in some way through $M$, which in turn affects the agent itself through the feedback it receives from its environment via $S$. Likewise, the environment's effects on an agent through $S$ are fed back through $M$ to in turn affect the environment itself. Thus, each of these two dynamical systems is continuously deforming the flow of the other (perhaps drastically if any coupling parameters cross bifurcation points in the receiving system's parameter space), and therefore influencing its subsequent trajectory. Note that one dynamical system cannot in general completely specify the trajectory of another dynamical system to which it is coupled. Rather, a dynamical system follows a trajectory specified by its own current state and dynamical laws. By varying some of the parameters of these laws, a second dynamical system can certainly bias the intrinsic "tendencies" of the first (or even cause qualitative changes in behavior if bifurcations occur). However, one dynamical system cannot in general "steer" the trajectory of another dynamical system along any desired path. It is therefore perhaps most accurate to view an agent and its environment as mutual sources of perturbation, with each system continuously influencing the other's potential for subsequent interaction.

Given this tight coupling between an agent and its environment, we can equally

well view the two coupled nonautonomous systems $\mathcal{A}$ and $\mathcal{E}$ as a single autonomous dynamical system $\mathcal{U}$ whose state variables are the union of the state variables of $\mathcal{A}$ and $\mathcal{E}$ and whose dynamical laws are given by all of the internal relations (including $S$ and $M$) among this larger set of state variables and their derivatives. Neither of these perspectives is intrinsically better than the other, and one could switch between them as necessary. Given everything that has been said in Section 2, any trajectories observed in the interaction between the nonautonomous dynamical systems $\mathcal{A}$ and $\mathcal{E}$ must be trajectories of the larger autonomous dynamical system $\mathcal{U}$. Furthermore, after transients have died out, the observed patterns of interaction between $\mathcal{A}$ and $\mathcal{E}$ must represent an attractor of $\mathcal{U}$.

We thus have the basis for a dynamical understanding of one of the central themes of recent autonomous agent research, namely the idea that an agent's behavior arises not simply from within the agent itself, but rather through its interaction with its environment. Due to the higher dimensionality of its state space, a dynamical system formed by coupling two other systems can generate a richer range of dynamical behavior than either system could individually, and properties of the coupled system can therefore not generally be attributed to either subsystem alone. Therefore, an agent's behavior properly resides only in the dynamics of the coupled system $\mathcal{U}$ and not in the individual dynamics of either $\mathcal{A}$ or $\mathcal{E}$ alone. This suggests that we must learn to think of an agent as containing only a latent potential to engage in appropriate patterns of interaction. It is only when coupled with a suitable environment that this potential is actually realized through the agent's behavior in that environment.

### *3.2. Adaptive fit*

What constitutes an "appropriate" pattern of interaction between an agent and its environment? It is often said that the behavior of animals is amazingly well adapted to the environments in which they must live. While, strictly speaking, evolution directly selects only for reproductive success, it is only animals whose behavior "fits" the dynamical and statistical structure of their environments that survive long enough to reproduce. We would like the behavior of the autonomous agents that we design to be similarly well-adapted to the environments in which they must function. Thus, the notion of adaptive fit is crucial to understanding the relationship between an agent and its environment. But what does it mean for an agent to be adapted to its environment?

Let us focus for the moment on animals, whose adaptive fitness is related to their survival. Then we can temporarily reformulate the question What does it mean for an agent to be adaptively fit to an environment? to the question What does it mean for an animal to survive? In order to answer this question, it will be useful to begin with a simple analogy to autonomous dynamical systems. As we have seen in Section 2, a dynamical law induces change on the state variables of a dynamical system. However, not all states are treated equally. While most states will be changed into other states through the action of the dynamical law, some

states will persist indefinitely because they are invariant with respect to the changes caused by the dynamical law (i.e., an equilibrium point attractor of the system). Invariant states "survive" in the same way that rocks do, by resisting change.

Unlike rocks, animals actively engage their environments in order to stably maintain their existence. Similarly, we expect the agents that we design to accomplish particular tasks in their environments, not to sit immobile and ignore the world around them. In order to capture this more dynamic notion of survival, we can extend our analogy to periodic trajectories, which persist in a far more interesting way than do equilibrium points. In the case of a limit cycle, no single state is invariant with respect to the dynamical law. Rather, all of the states along a limit cycle have the property that the action of the dynamical law carries them to other states along the limit cycle, forming a closed curve which is itself invariant. Thus, the persistence of a limit cycle is achieved only by coordinating the effects of the dynamical law on all of the state variables of the system in such a way that a closed trajectory is formed.

Even such a dynamically maintained invariant as a limit cycle does not quite capture the notion of survival that we are after. It falls short in two ways. First, as explained in Section 3.1, an animal is not an autonomous dynamical system, but rather a nonautonomous one which is constantly perturbed by its environment. Second, an animal does not really have to maintain any *particular* trajectory in order to survive, as suggested by the limit cycle metaphor. Rather, in order to survive, any living organism must maintain the integrity of the network of biochemical processes that keep it alive. Maturana and Varela [47, 48] have termed this network of processes an *autopoietic* system.[2] If an animal's autopoiesis is sufficiently disrupted, either as a result of its own internal dynamics or as a result of environmental perturbations that it cannot properly compensate for, then the animal will cease to exist. Thus, an animal's autopoiesis serves as a crucial constraint on its behavioral dynamics. We can visualize this constraint as a (perhaps very complex and time-varying) volume in an animal's state space (Fig. 3; [6]). An animal is adaptively fit to an environment only so long as it maintains its trajectory within this constraint volume despite the perturbations that it receives from its environment.

In order to elaborate this basic account of adaptive fit, the nature of the constraint volume would need to be more completely characterized. For any real animal, this volume must obviously be very complicated, varying in time with its internal state. Indeed, the separation between the animal's behavioral dynamics and its constraint volume is fundamentally somewhat artificial, because any given animal's behavioral dynamics is clearly related to the particular way in which its autopoiesis is realized and this itself changes through evolution [47, 48]. However, for our purposes here, I take the existence of an agent (living or otherwise) for

[2] An autopoietic (lit. self-producing) system is a network of component-producing processes with the property that the interactions between the components produced generate the very same network of interactions that produced them.

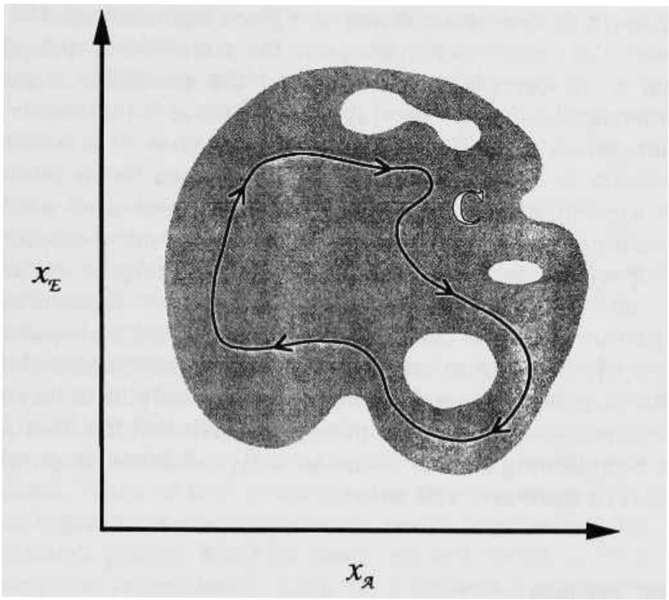



Fig. 3. An illustration of adaptive fit. This simple example assumes that both the agent and its environment are one-dimensional dynamical systems. As the state of the environment $x_E$ moves up and down, the state of the agent $x_A$ must move back and forth accordingly so that the trajectory of the coupled system always remains within the constraint volume $\mathbb{C}$. Here the two-dimensional coupled agent-environment system exhibits a limit cycle that satisfies the given constraint, but this is only one of an infinite number of possible agent-environment trajectories that satisfy $\mathbb{C}$.

granted and focus instead on the behavioral dynamics required to maintain that existence. This focus justifies the separation of behavioral dynamics from autopoietic constraints, and allows me to assume that the constraint volume is given *a priori*.

The adaptive fit of natural animals to their environments results from an evolutionary process involving reproduction with heritable variation and natural selection. When animals reproduce, mutations and sexual recombination of their genetic material lead to variations in their design. Because the genetic material of those animals which do not survive long enough to reproduce is not passed on to descendants, inappropriate designs are pruned away, while successful designs proliferate. In terms of our framework, we can think of evolution as trying out many different agent dynamics and retaining only those that, on average, are capable of satisfying their autopoietic constraints long enough to reproduce.

How might this notion of an autopoietic constraint apply to artificial agents? Homeostatic processes may be involved even for a robot. For example, an autonomous robot may need to regularly replenish its energy and avoid any potentially damaging situations. However, since the agents that we design are not living organisms, their existence does not strictly depend upon autopoiesis. Instead, the success of an artificial agent is typically measured in terms of its

ability to accomplish some desired task in a given environment. For our purposes here, however, this external criterion plays the same role as autopoiesis in living animals, that is, it serves as a constraint on the admissible trajectories of the agent-environment dynamics. The only real difference is that, while autopoiesis is largely an intrinsic constraint on an animal's own state, it is usually an artificial agent's effects on its environment that are constrained by an external designer.

Thus, we can immediately generalize the above notion of adaptive fit to an arbitrary constraint $\mathbb{C}$ on the dynamics of a coupled agent-environment system (Fig. 3). I will say that an adaptive fit exists between an agent and its environment as long as the trajectory of the agent-environment dynamics satisfies this constraint, that is, as long as their interaction results in an adequate performance of the task for which the agent was designed. As a somewhat fanciful example, we might consider a robot vacuum cleaner to be adaptively fit to its environment as long as its interactions with its environment are such that the floor remains clean, despite the complicating factors found in a typical home (e.g., children, pets, rearrangement of furniture, and so on).

## 4. A concrete example

The basic theoretical framework sketched in the previous section is rather abstract in nature. In order to make the general framework that I have proposed more concrete, this section will show how it can be applied to examples of each of the two major problems in autonomous agents research, namely the *synthesis* problem and the *analysis* problem. Loosely speaking, the synthesis problem is the problem of constructing an agent that does what we want in a given environment, while the analysis problem is the problem of understanding how a given agent does what it does in a given environment. I will show how a walking behavior for a simulated legged agent can be synthesized and analyzed from the dynamical systems perspective of this framework. Along the way, I will point out some of the distinct advantages of this approach.

### *4.1. Synthesis of a walking agent*

A major concern of much of the work on situated agents has been how to design agents that engage in some desired interaction with their environments. In terms of our framework, we can state this synthesis problem somewhat more formally as follows:

*The Synthesis Problem.* Given an environment dynamics $\mathscr{E}$, find an agent dynamics $\mathscr{A}$ and sensory and motor maps $\boldsymbol{S}$ and $\boldsymbol{M}$ such that a given constraint $\mathbb{C}$ on the coupled agent-environment dynamics is satisfied.

In order to illustrate the advantage of a dynamical systems perspective on the

synthesis of autonomous agents, let us consider the problem of designing a dynamical neural network that will make a simulated insect-like agent walk (Fig. 4; [11]). In terms of our framework, the dynamics of the agent's body is $\mathscr{E}$ and the dynamics of the neural network that controls it is $\mathscr{A}$. $\boldsymbol{M}(\boldsymbol{x}_{\mathscr{A}})$ gives the transformation from neural activity to body effectors, while the transformation from body sensors to neural inputs is given by $\boldsymbol{S}(\boldsymbol{x}_{\mathscr{E}})$. Here $\mathbb{C}$ is a constraint on $\mathscr{E}$ only, namely that the average velocity of the body be greater than zero (where positive velocities correspond to forward motion and negative velocities correspond to backward motion). We will assume that $\boldsymbol{S}$ and $\boldsymbol{M}$ are given *a priori* in this example and the problem is to design a neural network controller whose dynamics are such that, when coupled to the agent's body, they cause it to walk. Note that the design of locomotion controllers for hexapod robots is currently a problem of some practical interest [15, 19, 26]. For a complete description of this work, as well as additional examples, see [13].

The body operates as follows. There are six legs, each with a foot that may be either up or down. When its foot is down, a leg provides support to the body and any forces that it generates contribute to the body's translation under Newtonian dynamics (the stance phase). When its foot is up, any forces generated by the leg cause it to swing (the swing phase). Each leg is controlled by three effectors: one governs the state of the foot and the other two determine the clockwise and counterclockwise torques about the leg's single joint. Each leg also possesses a single sensor that measures its angle relative to the body. The body can only move when it is stably supported, that is, when the polygon formed by the supporting feet contains the body's center of mass.

The agent is controlled by a continuous-time recurrent neural network. Such networks were briefly introduced in Section 2. In their most general form, an interconnected network of $N$ such neurons is described by the following system of equations:

$$\tau_i \dot{y}_i = -y_i + \sum_{j=1}^{N} w_{ji}\sigma(y_j - \theta_j) + I_i(t) \quad i = 1, 2, \ldots, N \tag{6}$$

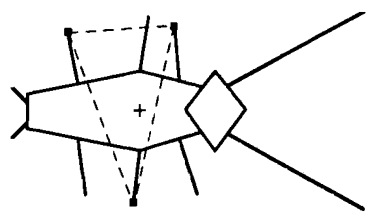


Fig. 4. The body model of the simulated insect. The legs can swing about their single joint with the body. A supporting foot is denoted by a black square. The body can only move when it is statically stable, i.e., when the polygon formed by the supporting feet (dashed line) contains the center of mass (cross) of the body.

where $y$ is sometimes interpreted as the mean membrane potential of the neuron, $\sigma(\xi) = (1 + e^{-\xi})^{-1}$ is a sigmoidal (S-shaped) function which can be interpreted as its short-term average firing frequency, $\theta_j$ is a bias term associated with the cell's firing threshold, $\tau$ is a time constant related to the passive properties of the cell membrane, $w_{ji}$ represents the strength of the connection from the $j$th to the $i$th neuron, and $I(t)$ represents a time-varying external input to the network (such as from a sensor). By restricting the matrix of connection weights to be zero-diagonal symmetric, Hopfield [38] demonstrated that such networks could be used as associative memories, with each pattern stored as a different equilibrium point attractor of the network dynamics. When no such restriction is placed on the connection weights, these networks are capable of exhibiting a much wider range of dynamical behavior. This is the form in which they will be used here. Note that no claim is being made about the general applicability of this particular neural model. It was merely selected to illustrate the framework due to its simplicity and widespread use.

Each leg of the agent was controlled by a 5-neuron fully-interconnected network (Fig. 5(A)). Three of these neurons are motor neurons whose outputs drive the three effectors of the leg, while the other two neurons are interneurons whose role is unspecified. All five neurons received a weighted input from the leg's angle sensor. Six copies of the single leg controller were combined in an architecture loosely based upon the organization of the neural circuitry underlying insect locomotion [11] to form a full locomotion controller with 30 neurons (Fig. 5(B)). Symmetry considerations were used to reduce the number of free parameters in this circuit to 50 (5 thresholds, 5 time constants, 25 leg controller weights, 5 sensor weights, 5 crossbody connection weights and 5 intersegmental connection weights). We wish to find settings of these 50 parameters such that the dynamics of the network causes the agent to walk when coupled to the body shown in Fig. 4.

Regardless of the particular control mechanism used, the majority of current work on situated agents relies on a human designer to manually construct a controller that will cause the agent to engage in some desired interaction with its environment. However, a number of researchers have begun to realize that manual design may not be the best approach. What is difficult about the synthesis problem is that the constraint to be satisfied may be very complex and its specification may be very far removed from the actual agent dynamics required to satisfy it. For example, the constraint that the average velocity of the body be greater than zero does not immediately specify what signals need to be sent to the body's eighteen effectors in order to satisfy this constraint. In addition, natural environments are rather complicated and somewhat unpredictable. Manual design often fails because designers, in trying to anticipate the possible opportunities and contingencies that might arise, build too many unwarranted assumptions into their designs. For these reasons, a number of researchers have begun to explore automated techniques for autonomous agent design, such as reinforcement and other forms of learning (e.g., [9, 10, 23, 41, 46]) or genetic algorithms (e.g., [17,

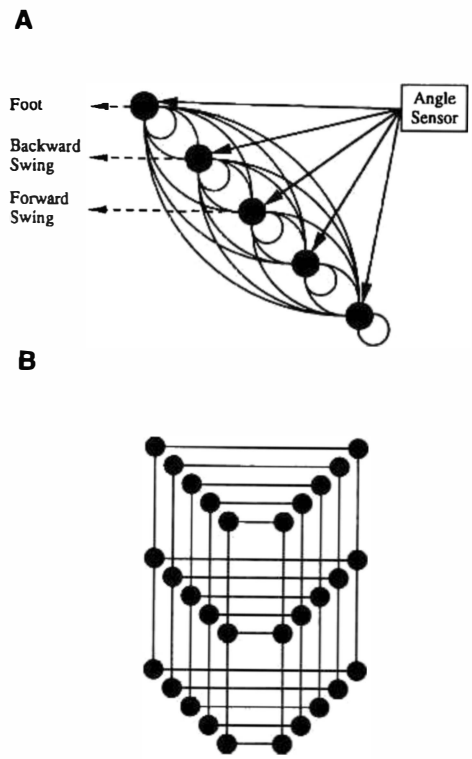



Fig. 5. Network architecture of the locomotion controller. (A) A leg controller. (B) Coupling between adjacent leg controllers.

40, 65, 67]). We used a genetic algorithm to search the space of network parameters for networks whose dynamics make the body shown in Fig. 4 walk.[3]

A genetic algorithm (GA) is a search technique whose operation is loosely based on natural evolution [31, 37]. The basic cycle of a genetic algorithm operates as follows. The space to be searched (in our case, the 50 network parameters) is usually encoded as a binary string. An initially random population of such strings is maintained. At each iteration, the performance of each individual is evaluated. A new generation of individuals is then produced by applying a set of genetic operators to selected individuals from the previous generation. Individuals are selected for reproduction with a probability propor-

[3] We employed a public GA simulator known as GAucsd (version 1.1). At the time of this writing, the latest version of GAucsd is available by anonymous ftp from cs.ucsd.edu in the pub/GAucsd directory. All network parameters were encoded in four bits, with time constants in the range [0.5, 10] and both thresholds and connection weights in the range [−16, 16]. The crossover rate was set to 0.6 and the mutation rate was set to 0.0001. Population sizes of 500 were used and good locomotion controllers typically took on the order of 100 generations to evolve. Full details of these experiments can be found in [13].

tional to their fitness. The standard genetic operators are mutation (in which bits are randomly flipped) and crossover (in which portions of the genetic strings of two individuals are exchanged). By iterating the processes of selection, recombination and mutation, the population accumulates information about the distribution of fitness in the search space. This information focuses subsequent search into fruitful subspaces.

In order to guide its search, a genetic algorithm requires a real-valued measure of performance rather than a rigid constraint to be satisfied. In this case, the constraint $\mathbb{C}$ can be thought of as some minimum acceptable level of performance. We used the total forward distance traveled by the agent in a fixed amount of time as the performance measure to be optimized.[4] Note that, by optimizing distance traveled in a fixed amount of time, we are not only demanding that the insect walk, but that it walk as quickly as possible. Because the insect can only make forward progress when it is statically stable, the GA must find a network dynamics that not only appropriately generates the three control signals required to operate each leg, but also properly coordinates the independent movements of the six legs so that stability is continuously maintained in order to satisfy the constraint that the average velocity of the body be greater than 0.

There are two different ways that we can think about these experiments. Abstractly, we can think of continuous-time recurrent neural networks as simply a basis dynamics out of which to build whatever agent dynamics is required and we can think of GAs as simply a technique for searching the family of flows defined by the parameterized network architecture for one whose dynamics cause the agent to walk when it is coupled to the body. More concretely, we can think of our neural network as a simple model of a nervous system and the genetic algorithm as a simple model of evolution. This second perspective can actually be quite useful because it allows comparisons to be made between the model and biology.

However, we must be careful not to lose sight of the many simplifications involved in this latter perspective. Both nervous systems and evolution are considerably more complicated than these simple models would suggest. To take just one example, while we have externally imposed a notion of fitness on the GA, no such external fitness measure exists in natural evolution. Indeed, because an animal's environment includes many other animals that are simultaneously evolving, the relationship of a given behavior to reproductive success may change significantly over time. Fitness is something intrinsic to natural environments rather than being externally specified. The significance of this difference between extrinsic and intrinsic fitness is that, by favoring particular behaviors over others in a fixed, *a priori* fashion, extrinsic fitness functions limit the range of behaviors that can possibly evolve in a way that intrinsic fitness does not.

We evolved eleven different locomotion controllers in all. Though the specific

[4] Because GAucsd is formulated to minimize an error measure rather than maximize a fitness measure, the actual measure used was the square of the difference between the maximum attainable distance and the actual distance covered in a given length of time.

parameter values found by the GA were quite different in these eleven networks, the dynamics of all of them have the property that, when coupled to the insect-like body shown in Fig. 4, they cause it to walk in such a way that stability is continuously maintained. The behavior of a typical controller is shown in Fig. 6(A). All of these controllers generate a pattern of leg movements known as the *tripod gait*, in which the front and back legs on each side of the body swing in unison with the middle leg on the opposite side. The tripod gait is ubiquitous among fast-walking insects [32].

As the networks evolved, they passed through several more or less distinct stages. Very early on, agents appeared that put down all six feet and pushed until they fell. These agents thus exhibit roughly the proper phasing of the three signals controlling each leg, but lack the ability to recover a leg after a stance phase as well as the ability to coordinate the motions of the different legs. In the next stage, agents evolved the ability to rhythmically swing their legs in an uncoordinated fashion. Such agents made forward progress, but they fell quite often. Finally, agents utilizing statically stable gaits began to appear, but their coordination was still suboptimal. Subsequently, the efficiency of locomotion slowly improved.

During these experiments, we discovered that the nature of the environment in

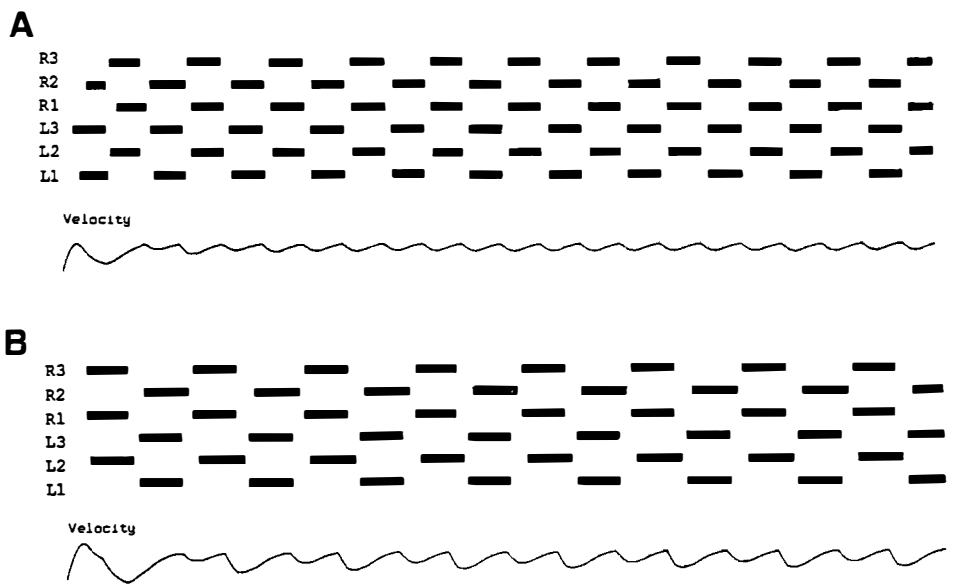



Fig. 6. Behavior of a typical mixed locomotion controller with (A) and without (B) its sensors. Black bars represent the swing phase of a leg and the space between bars represents a stance phase. The legs are labeled L for the left and R for right and numbered from 1 to 3 starting at the front of the agent. Note that the stepping frequency is higher, the swings of the three legs in each tripod are more tightly coordinated and the velocity varies less when the sensors are intact (A). However, this controller can generate a reasonably good tripod gait even in the complete absence of any sensory feedback from the body (B).

which the locomotion controllers were evolved had a major impact on their functional organization. In particular, the relative contributions of $\mathscr{A}$ and $\mathscr{E}$ to the generation of a walking pattern varied according to the dependability of sensory feedback during evolution. Three different classes of locomotion controllers were found:

(1) If sensors were enabled during the GA search, then *reflexive patterns generators* always evolved (5 trials). A reflexive controller is one which depends upon sensory feedback for its operation. If the sensors are later removed, reflexive controllers exhibit inappropriate phasing of motor outputs, and some cannot even oscillate in isolation. A reflexive controller is therefore not robust to sensor loss. Reflexive controllers take advantage of the fact that there is no point putting into the agent any dynamics that already appear to be in the environment. All that matters is that the coupled agent-environment system satisfy the given constraint.

(2) If the sensors were disabled during the GA search, then no access is provided to $\mathscr{E}$. In this case, so-called *central pattern generators* (CPGs; [25]) always evolved (4 trials). Even though the individual neurons are not oscillatory, a CPG is capable of generating the rhythmic control signals required for walking. The drawback of a CPG is that its output is stereotyped. It can make no use of sensory feedback to fine-tune its operation.

(3) Finally, if the presence of sensors was unreliable during the GA search (i.e., sensors were sometimes available and sometimes not), then *mixed pattern generators* evolved (2 trials). A mixed locomotion controller is one that works better with its sensors intact, but is quite capable of generating the basic motor pattern required for walking even without its sensors (Fig. 6). Though mixed controllers are robust to sensory damage, they are capable of using sensory feedback to improve their performance when it is available. Such mixed organizations are the most typical among biological pattern generators.

In this section, I formulated the problem of designing a walking agent as a search through a space of dynamical systems for those that, when coupled to a given body, maximize the forward distance that the body travels in a fixed amount of time. This same general approach has also been used to evolve a variety of chemotactic agents that were capable of using chemical signals to find their way to a patch of food [13]. The most notable result from these chemotaxis experiments were agents that utilized a bifurcation in their network dynamics to switch between distinct strategies depending upon the intensity of the chemical signal (which in turn depended upon the agent's distance from the food patch). Furthermore, I have demonstrated how manipulating characteristics of the environment (i.e., sensor dependability) puts selective pressure on the development of controllers with very different functional organizations. This ability to automatically tailor agent dynamics to fit the dynamical and statistical structure of a given environment is a significant advantage of automated agent design techniques.

### 4.2. *Analysis of a walking agent*

Given that some agent already exists, we might like to explain its behavior in a given environment. This is in fact the major problem faced by Neuroethologists, who seek to explain an animal's observed behavior in terms of its nervous system, its body and its environment. In terms of our framework, we can state this analysis problem somewhat more formally as follows:

***The Analysis Problem.*** Given an environment dynamics $\mathscr{E}$, an agent dynamics $\mathscr{A}$, and sensory and motor maps $S$ and $M$, explain how the observed behavior $M(x_{\mathscr{A}})$ of the agent is generated.

In order to illustrate the utility of a dynamical systems perspective on the analysis of autonomous agents, we would like to understand the operation of the evolved locomotion controllers described in the previous section. Unfortunately, a dynamical analysis of these 30 neuron networks would be far too complicated for our illustrative purposes here. However, in a set of closely related experiments, we also evolved five-neuron controllers for single-legged insects [13]. Note that, for the purposes of evolving single leg controllers, we had to modify the stability criteria so that a single-legged insect could move whenever its single foot was down. Except for the lack of an interleg coordination problem to be solved, these experiments were in every way analogous to those described in the previous section. These leg controllers passed through similar evolutionary stages and we also found reflexive, central and mixed pattern generators depending upon the conditions under which they were evolved. Fig. 7 shows the activity of a mixed leg controller with and without its sensors. Because these five-neuron networks are much more amenable to a dynamical analysis, we will focus on them here. For additional information on this analysis, see [12, 29].

#### 4.2.1. *Analysis of a central pattern generator*

Because they have no sensory input, central pattern generators are autonomous dynamical systems. For this reason, CPGs are in some sense the simplest leg controllers to understand. In the case of a CPG, the dynamics of the neural network simply exhibits a limit cycle whose motor space projection $M(x_{\mathscr{A}})$ causes the insect's single leg to rhythmically stance and swing in a fashion appropriate to walking. The three-dimensional motor space projection of the five-dimensional limit cycle exhibited by one CPG is shown in Fig. 8. This limit cycle repeatedly takes the state of the system through the regions in motor space associated with stance phase (upper left-hand corner) and swing phase (lower right-hand corner). Since a limit cycle is a primitive concept in dynamical systems theory, there is really nothing more to be said at this level of discussion about the operation of a CPG (though there is of course much more that might be said about the way in which this limit cycle is realized in this particular circuit).

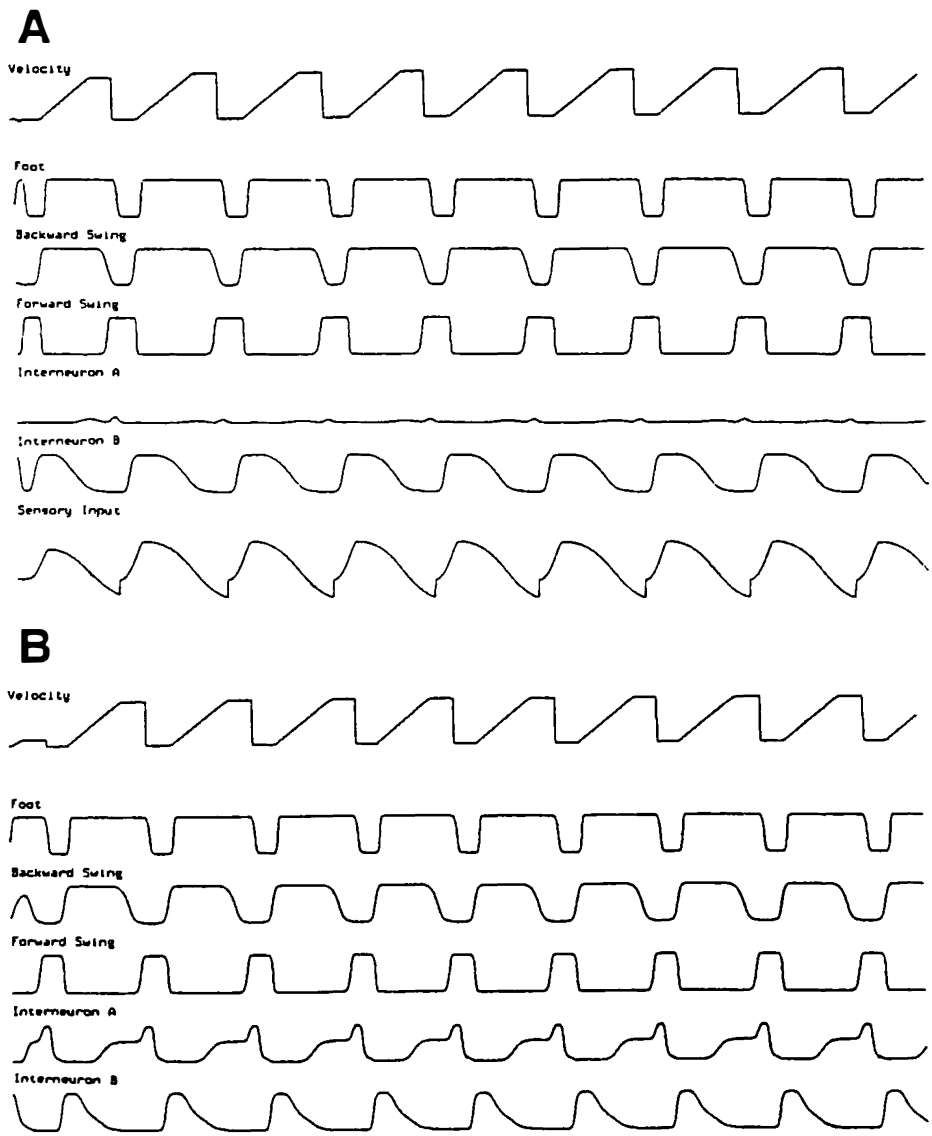



Fig. 7. Activity of a typical mixed leg controller with (A) and without (B) its sensor. Each group of plots shows the forward velocity of the body, the output of the foot, backward swing and forward swing motor neurons and the output of the two interneurons. The velocity ramps up to a maximum value during each stance phase and then drops to zero when the insect lifts its single leg each swing phase and falls. The top plot also shows the output of the leg angle sensor. In both groups of plots, the leg was initialized at 95% of its full backward position (i.e., near the point where a swing phase should begin). Note that, with its sensor intact (A), this controller almost immediately begins a swing phase. However, without its sensor, this controller inappropriately attempts to generate a stance phase, effectively wasting a step, because it has no access to the leg's angular position. Note also that Interneuron A appears to play a much larger role in the walking pattern when the sensor is absent (B) than when it is present (A).

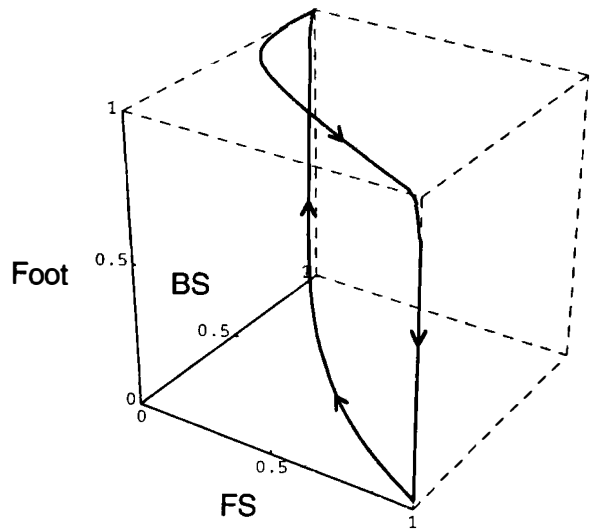



Fig. 8. A motor space projection of the five-dimensional limit cycle generated by a typical central pattern generator. The output of the foot, backward swing (BS) and forward swing (FS) motor neurons are plotted. Note that the foot is considered to be down when the output of the foot motor neuron is above 0.5 and up otherwise. A stance phase (foot and backward swing motor neurons active, forward swing motor neuron inactive) corresponds to a region near the back, upper left-hand corner of the state space, while a swing phase (forward swing motor neuron active, foot and backward swing motor neurons inactive) corresponds to a region near the front, lower right-hand corner of the state space.

### *4.2.2. Analysis of a reflexive pattern generator*

Due to the presence of a sensory feedback signal, reflexive leg controllers are nonautonomous dynamical systems. When coupled to the body, the motor space projection of the dynamics of a reflexive controller also exhibits a suitable limit cycle (Fig. 9). However, we already know that reflexive controllers do not produce appropriate rhythmic output when their sensory input is removed. Unlike a CPG, the limit cycle of a reflexive leg controller arises only when it is coupled to the body. Technically, this limit cycle is a three-dimensional projection of a higher dimensional trajectory of the coupled agent-environment system $\mathcal{U}$. How does the interaction between a reflexive controller's autonomous dynamics and the sensory feedback that it receives from the body produce the observed limit cycle?

One way to approach this question is to think of a reflexive controller as an autonomous dynamical system whose flow is parameterized by the sensory input that it receives from the leg's angle sensor. At any given point in time, the network's state is flowing toward the attractor in whose basin it finds itself. However, because the angle of the leg is constantly changing, the structure of the network's flow is changing also, perhaps even undergoing bifurcations. We can visualize the instantaneous phase portrait of the autonomous network dynamics corresponding to any given leg angle. We can also visualize the network's state and the trajectory that it is instantaneously following at any point in the limit

cycle. Of course, the system state generally never completely traverses these instantaneous trajectories because the phase portrait continuously changes as the leg moves. However, by piecing together these instantaneous pictures at many different points in time, we can build up a picture of the dynamics underlying the limit cycle observed when a reflexive controller is coupled to the body. Note that the leg actually passes through any given angle twice; once in swing phase and once in stance phase. While the phase portrait is the same in each case (since it depends only on the leg angle), the system's state, and hence the trajectory that it is following, will in general be different.

Such an analysis of one reflexive controller is presented in Fig. 9. The visualization of this particular controller is simplified by the fact that once transients have passed, the outputs of its two interneurons become constant. In other words, once the limit cycle is established, the dynamics of this network are essentially three-dimensional. The limit cycle that this reflexive controller exhibits when it is coupled to the body is shown at the center of Fig. 9. Surrounding this central plot are smaller plots showing the instantaneous autonomous dynamics of the network at different points in the swing/stance cycle. At (1), the foot has just been put down and a stance phase begun. At this point, the network's state is flowing toward the equilibrium point attractor in the upper left-hand corner of the state space. The position of this attractor corresponds to a situation in which the foot and backward swing motor neurons are active and the forward swing motor neuron is inactive (i.e., a stance phase). Due to the dynamics of the body, this pattern of motor neuron activity means that the foot is down and the leg is applying a force to the body that causes it to move forward, changing the leg angle and thus the output of the leg angle sensor. As the leg continues to stance at (2), the system state has essentially reached the equilibrium point. As the leg passes from (2) to (3), however, this equilibrium point suddenly disappears and is replaced by another equilibrium point near the lower right-hand corner of the state space that now begins to attract the system state. The position of this attractor corresponds to a state in which the foot and backward swing motor neurons are inactive and the forward swing motor neuron is active (i.e., a swing phase).

The system state now begins to flow toward this new attractor (3). Between (3) and (4), the output of the foot motor neuron falls below the activation threshold of the foot (0.5) and the foot is lifted, actually beginning a swing phase. As the leg passes from (4) to (5), the equilibrium point attractor in the lower right-hand corner of the state space disappears and the earlier equilibrium point attractor in the upper left-hand corner reappears. The network state now moves toward this attractor through (6) until the output of the foot motor neuron goes above the activation threshold for the foot at (1) and the foot is once again put down, beginning a new stance phase. Thus we can see how the limit cycle observed in the coupled network/body system arises as the network's state is alternately attracted by the two equilibrium points.

We can now explain the reason that this controller is a reflexive pattern generator by observing that, when its sensor is removed, the autonomous

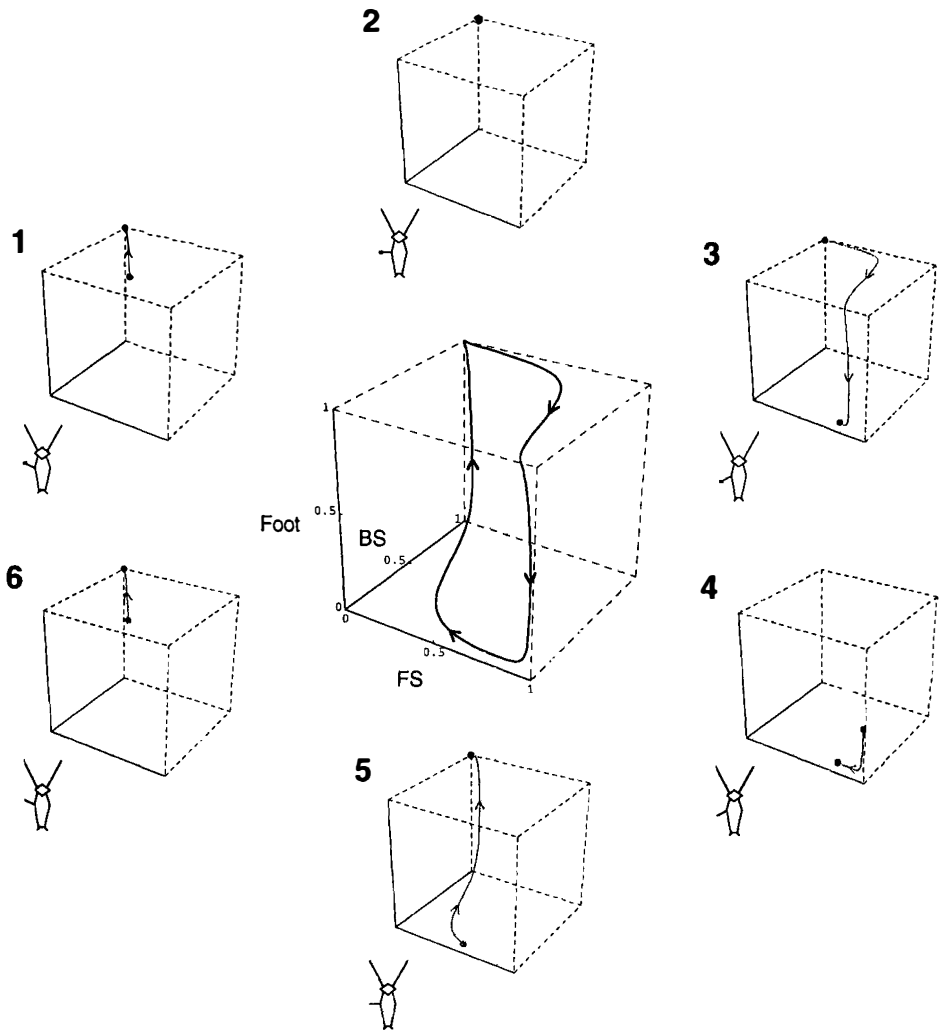



Fig. 9. Operation of a typical reflexive pattern generator. The output of the foot, backward swing (BS) and forward swing (FS) motor neurons are plotted. The limit cycle generated when this controller is coupled to the body is shown at the center. Surrounding this central plot are plots of the instantaneous autonomous dynamics of the network at different points in the step cycle. In each case, the solid point denotes an equilibrium point attractor, the gray point denotes the instantaneous system state, and the gray line shows the trajectory that the system would follow if the leg angle were to remain at its present angle. The top three plots roughly correspond to the beginning (1), middle (2) and end (3) of a stance phase, while the bottom three plots roughly correspond to the beginning (4), middle (5) and end (6) of a swing phase. Vertical columns of plots (i.e., (1) and (6), (2) and (5), and (3) and (4)) correspond to approximately the same leg angle and therefore the same phase portrait, though the system state and therefore the trajectory it is following differs between the upper and lower plots of each column.

dynamics of this controller is governed by an equilibrium point. By convention, a leg that is perpendicular to the long axis of the body is assigned a leg angle of 0. Since we modeled the removal of a sensor by setting its output to 0, the autonomous dynamics exhibited when the sensor is removed is identical to that exhibited when the leg is fixed in a horizontal position (i.e., plots (2) and (5) in Fig. 9). Thus, if the sensor is removed, the system state will flow toward the equilibrium point attractor in the upper left-hand corner of the state space and remain there, causing the leg to go into a permanent stance phase.

The switch between equilibrium points that occurs between (2) and (3) in Fig. 9, and again between (4) and (5), appears to be essential for the operation of this reflexive controller. How is this switch actually accomplished? This question is answered in Fig. 10, which shows a sequence of bifurcations that occur in the autonomous dynamics of this network as the leg moves from an angle of about 23 degrees past horizontal to 16 degrees past horizontal during swing phase (between plots (4) and (5) in Fig. 9). The sequence begins with a single equilibrium point attractor at the bottom of the state space (1). At (2), a second equilibrium point attractor appears at the top. Note that, during swing phase, the system state is still in the basin of attraction of the lower attractor at this point. At (3), the lower equilibrium point bifurcates into a limit cycle, which then begins to expand (4). Eventually, this limit cycle disappears, leaving behind only a single equilibrium point at (5). Now the system state is attracted by this upper equilibrium point. This sequence of bifurcations is reversed during stance phase. A complete bifurcation diagram for this network can be found in [12].

Because these bifurcations take place in such a narrow range of leg angles (approximately 7 degrees), the system state never really "sees" the intermediate attractors. For example, the limit cycle that briefly appears plays no functional role whatsoever in the network's dynamics because the system state never has a chance to get near it, let alone go around it. During the normal operation of this controller, this bifurcation sequence occurs in about 10 integration steps, during which the system state moves only an average Euclidean distance of 0.025 in the state space. However, the net effect of this sequence of bifurcations is to alternately switch the network's phase portrait between the two equilibrium points that are crucial to its operation. This particular sequence of bifurcations is unique to this controller, and was not observed in any of the other controllers that were analyzed.

From this dynamical analysis we can summarize the nature of the interaction between $\mathscr{A}$ and $\mathscr{E}$ that underlies the operation of this reflexive controller. The autonomous dynamics of $\mathscr{A}$, and its parameterization by $\boldsymbol{S}$, is such that $\mathscr{E}$ can deform it, via a series of intermediate bifurcations, into essentially two kinds of flows. In one of these flows, there is a single fixed-point attractor near the upper left-hand corner of the state space, while in the other there is a single fixed-point attractor near the lower right-hand corner. The nature of $\mathscr{E}$, and its parameterization by $\boldsymbol{M}$, is such that, when the network state is in the neighborhood of the upper left-hand attractor, the state of the body is changing in such a way that $\boldsymbol{S}$ will cause the lower right-hand attractor to appear in $\mathscr{A}$. Likewise, when the

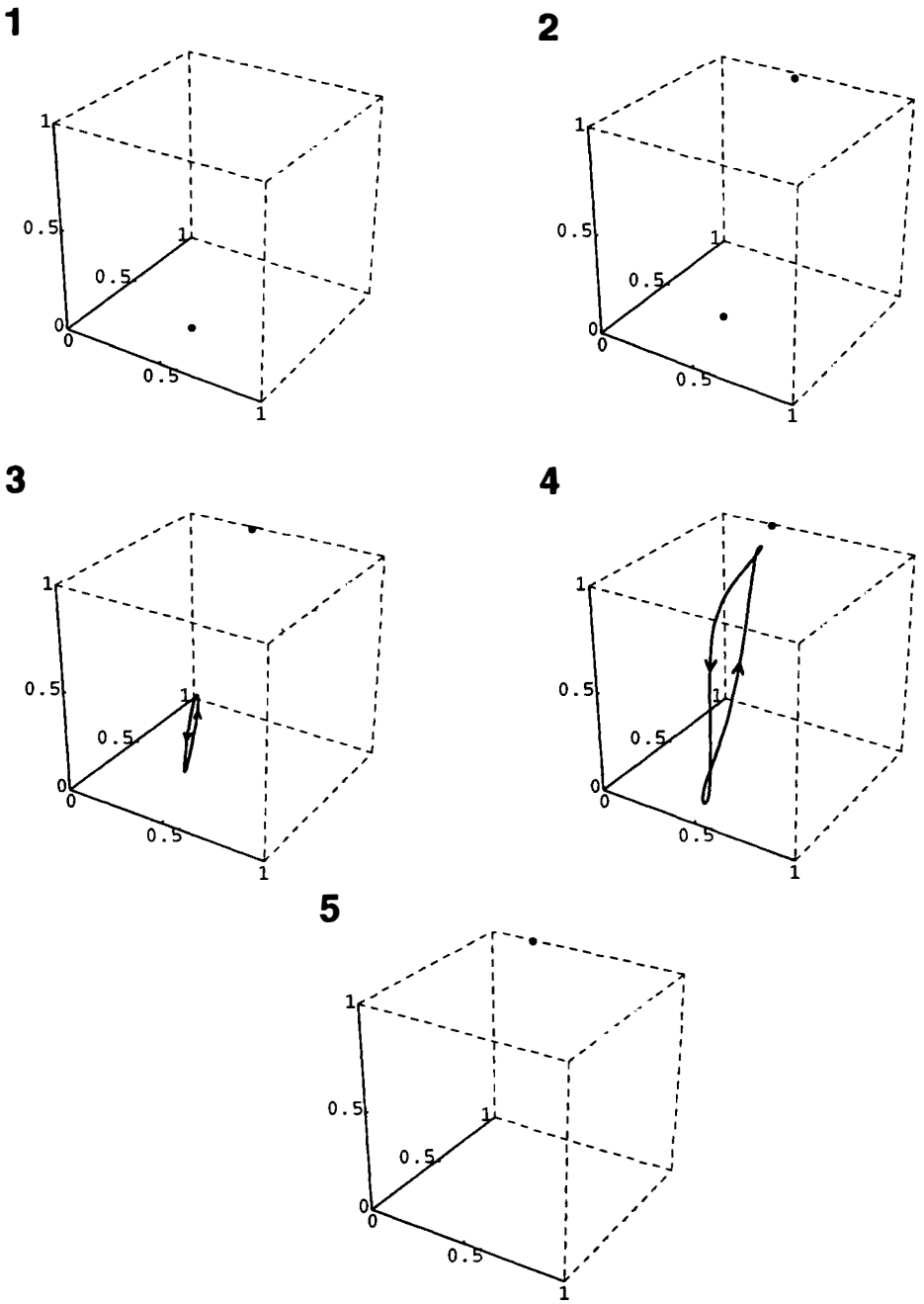



Fig. 10. A sequence of bifurcations underlying the operation of the reflexive pattern generator shown in Fig. 9. Only attractors are shown. During swing phase (between plots (4) and (5) in Fig. 9), the autonomous dynamics of the network undergoes the sequence of bifurcations shown here. During stance phase (between plots (2) and (3) in Fig. 9), this sequence of bifurcations is reversed.

network state is in the neighborhood of the lower right-hand attractor, the state of the body is changing in such a way that $S$ will cause the upper left-hand attractor to appear in $\mathscr{A}$. This reciprocal relationship between $\mathscr{A}$ and $\mathscr{E}$ is what gives rise to the observed rhythmic walking pattern. Therefore, both $\mathscr{A}$ and $\mathscr{E}$ play absolutely essential and deeply intertwined roles in the operation of this reflexive controller.

*4.2.3. Analysis of a mixed pattern generator*

As shown in Fig. 11, we can approach the analysis of a mixed leg controller in a fashion similar to our analysis of the reflexive controller. However, because the dynamics of this circuit is fundamentally five-dimensional, we only plot the three-dimensional motor space projection of this system's trajectories. The limit cycle that this controller exhibits when it is coupled to the body is shown in the center. While the shape of this limit cycle is somewhat different from the one generated by the reflexive controller discussed above, they both exhibit the proper phasing of motor outputs necessary to make the leg walk. Surrounding this central plot are smaller plots that show the instantaneous autonomous dynamics at various points along this limit cycle. As for the reflexive controller, we can understand the dynamics of the coupled network/body system by piecing together these instantaneous snapshots. However, unlike the reflexive controller, the autonomous dynamics of this mixed controller exhibits limit cycles rather than equilibrium points through most of the cycle. When the mixed controller is coupled to the body, this limit cycle is continuously deformed as shown in Fig. 11 as the leg angle changes, and the system state is constantly attracted by this deforming limit cycle. The reason that this mixed controller can tolerate the loss of its sensor is because the autonomous limit cycle that it generates when the sensory input is set to 0 (see plots (2) and (5) in Fig. 11) is appropriate to make the leg walk. In the absence of any sensory input, the system state would follow this limit cycle rather than the limit cycle shown in the center of Fig. 11.

Since a mixed controller is capable of autonomously generating an appropriate limit cycle, what role if any is the sensory feedback it receives from the body actually playing in its operation? In order to explore this question, we examined how the controller responds when it is artificially driven with sinusoidal sensory input whose frequency is higher or lower than normal. Under these conditions, we found that the motor output pattern that the controller generates speeds up or slows down accordingly (Fig. 12). The sensory signal is thus capable of entraining the intrinsic oscillation produced by the controller itself. Despite the fact that the activity pattern of the interneurons changes considerably throughout this range of operating frequencies, the amplitude, shape and phasing of the motor outputs remains appropriate for walking. Within a significant range about its normal operating frequency, the motor pattern remains 1:1 phase-locked with the sensory signal. We have observed other ratios of phase-locking at higher or lower driving frequencies. Entrainment by sensory feedback is a common feature of biological pattern generators. For example, the pattern generator underlying locust flight can be entrained by rhythmic stimulation of wing stretch receptors [64].

This entrainment has an interesting functional consequence. Suppose that the

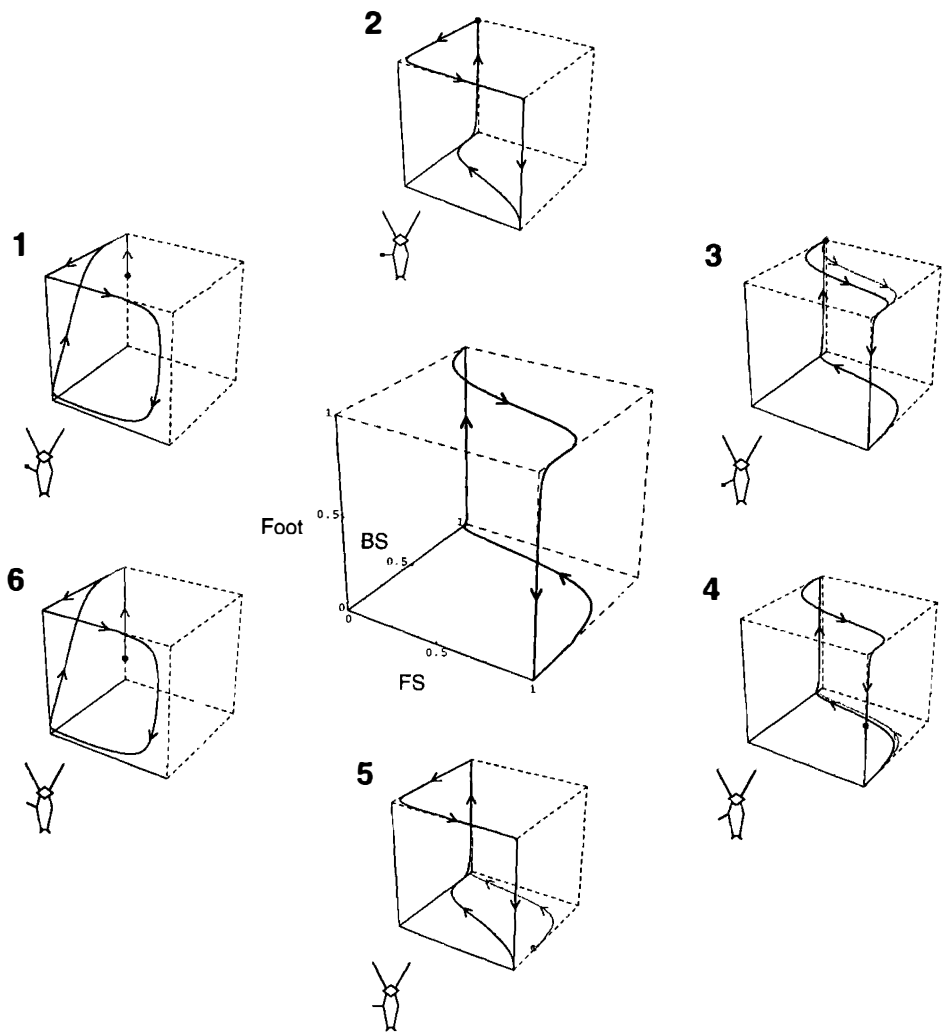



Fig. 11. Motor space projection of the operation of a typical mixed pattern generator. The layout of this figure is the same as for Fig. 9.

legs of the agent were to grow during its "life". For a given amount of applied torque, longer stancing legs will take more time to swing through a given angle than shorter legs. Thus, the sensory feedback signal from a longer leg will be spread out in time relative to that of a normal length leg. Since the mixed controller is entrained by the sensory feedback that it receives from the body, the sensory feedback from a longer leg will cause the leg controller to slow down its motor output pattern accordingly. Adapting their output to a changing periphery is a general problem that pattern generators have to deal with, for example in development or following peripheral damage. Note, however, that this adaptation

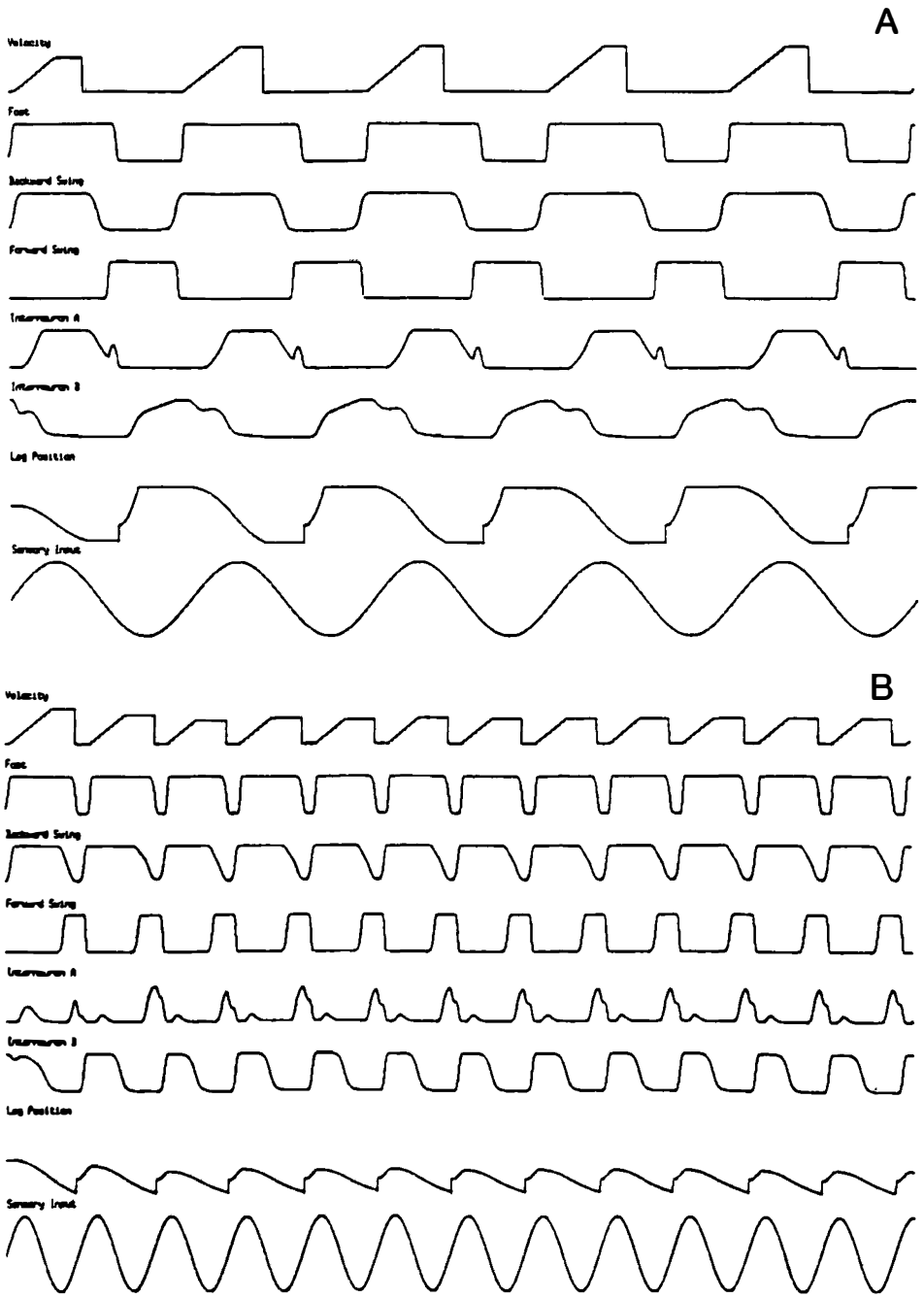



Fig. 12. Entrainment of the mixed pattern generator shown in Fig. 11 by sinusoidal sensory input. When the driving frequency is lower (A) or higher (B) than normal, the motor pattern slows down or speeds up accordingly.

does not come about through any structural change to the neural network itself, nor does it require a separate learning algorithm to modify the network parameters. Rather, it arises as a natural consequence of the dynamics of the mixed controller and its coupling with the body. This is a kind of functional plasticity which is quite different from what is normally thought of as learning. However, there are many examples of such plasticity in biology (for instance, the ability of an insect to immediately adjust its gait following the loss of a leg [32]).

The dynamical sophistication of this mixed leg controller is truly remarkable. With only *five neurons* (three of which are motor neurons), this controller can (1) generate the basic swing/stance leg movements necessary for walking; (2) take advantage of sensory feedback if it is available but can tolerate its absence with only a slight degradation in performance; and (3) adapt its operation to morphological changes in the body without requiring a separate learning mechanism. The likelihood of anyone designing such a flexible and compact leg controller by hand is probably rather low.

### *4.2.4. Conclusion*

This section has illustrated how some of the tools of dynamical systems theory can be applied to the analysis of an agent interacting with its environment. For example, we have been able to explain why some locomotion controllers are robust to loss of sensory feedback and others are not in terms of the appropriateness or inappropriateness of their autonomous dynamics to the walking task. Furthermore, we have been able to gain significant insight into the specific nature of the interaction between the network dynamics and the body that gave rise to a walking pattern in each of the three controllers analyzed. For example, while the reflexive controllers make use of the body dynamics in a fundamental way in the generation of the limit cycle necessary for walking, the CPGs are capable of generating appropriate limit cycles completely autonomously and the mixed controllers use rhythmic sensory feedback to fine-tune autonomous limit cycles. Along the way, we discovered that, because it is entrained by sensory feedback, the mixed controller can adapt its motor output to a changing periphery.

What general conclusions can we draw from this analysis? One is struck by the variety of agent dynamics which evolved. Not only are the actual network parameters of each controller different, but the underlying dynamics vary widely. Yet they all lead to virtually indistinguishable walking behavior when coupled to the body under normal conditions. Furthermore, lesion studies of these controllers (not described here) have demonstrated that their operation is dynamically distributed, usually making it impossible to assign specific functional roles to individual interneurons. About the only thing that can be said about all of these controllers is that the coupled agent-environment systems in which they are embedded do what we asked them to do under the conditions in which they evolved. Indeed, as in natural evolution, this is all that they were selected for in the first place. We are thus led to conclude that it is simply inappropriate in

general to attempt to impose our functional preconceptions on the organization of evolved systems.

## 5. Discussion

In this paper, I have attempted to use the language of dynamical systems theory to sketch a general theoretical framework for the design and analysis of autonomous agents. The framework focuses on the problem of generating the appropriate behavior at the appropriate time as both an agent's internal state and its external environment continuously change. The two key ideas of this framework are (1) that an agent and its environment must be understood as two coupled dynamical systems whose mutual interaction is jointly responsible for the agent's behavior and (2) that the adaptive fit between an agent and its environment can be characterized in terms of the satisfaction of a given constraint on the trajectories of the coupled agent-environment system. I have particularly emphasized that an agent's behavior is, strictly speaking, a property of the coupled agent-environment system only and cannot in general be attributed to either the agent or environment individually.

A concrete application of these ideas to the synthesis and analysis of a walking behavior for a six-legged agent was used to illustrate the framework. While I feel no particular commitment to either continuous-time recurrent neural networks or genetic algorithms, they represent at least one way in which agent dynamics satisfying a given constraint on the coupled agent-environment system can be designed. Such an "evolutionary" approach to agent design allows the agent's organization to be tailored to the particular dynamical and statistical structure of its environment and leads to remarkably adaptive, robust and compact controllers. I have also demonstrated that, despite the fact that they often exhibit no clean functional organization, the operation of these evolved systems can be understood from the perspective of interacting dynamical systems. Furthermore, this perspective can provide significant insight into the specific nature of the interaction between the agent and its environment that gives rise to the observed walking behavior in the various controllers.

### *5.1. Related work*

There is currently a growing interest in dynamical explanations in the behavioral and brain sciences. The central idea dates back at least to cybernetics [6, 63]. More recently, concepts from dynamical systems theory have been making a substantial impact in such fields as neuroscience, cognitive science and mobile robotics. Within neuroscience, dynamical analyses have been applied to single neurons (e.g., [55]), small circuits (e.g., [62]) and complete brain systems (e.g., the model of olfactory cortex formulated by Skarda and Freeman [58], in which a chaotic attractor plays a central role). The concepts of dynamical systems are also beginning to play a major role in understanding the biological control of

movement (e.g., [57]). Connectionism has made dynamics one of its founding principles [60] (though it is not clear that the full implications of this principle have yet been appreciated in connectionist research [36]) and has, for example, begun to propose theories of language in its terms [27, 53]. Both van Gelder [61] and Giunti [30] have begun to formulate a dynamical conception of cognition in cognitive science more generally. Finally, Smithers [59] has recently argued for a role for dynamical ideas in mobile robot research.

Within the autonomous agents literature, the theoretical framework that I have proposed is perhaps most closely related to Rosenschein's work on situated automata [56]. He models an agent and its environment as two interacting automata and he has emphasized that knowledge need not be explicitly encoded within an agent in order for it to engage in sophisticated interactions with its environment. He has used concepts from automata and formal language theory to characterize the behavior of such systems. In a sense, I have generalized this perspective to arbitrary dynamical systems and demonstrated how concepts from dynamical systems theory can be used to characterize the behavior of such systems, especially emphasizing continuous systems. On the other hand, while Rosenschein's major concern has been how propositional content can be assigned to correlations between an agent's internal states and the states of its environment, my interest is in how an agent can generate the appropriate behavior at the appropriate time.

### *5.2. Assumptions and extensions*

This paper has focused on continuous, deterministic, convergent and low-dimensional dynamical systems as models for agents and their environments. In this section, I briefly consider the motivation behind these assumptions, their impact on the framework, and a number of ways in which they might be relaxed.

#### *5.2.1. Continuity*

The emphasis on continuous dynamics (i.e., continuous state spaces and continuous time) in this paper is motivated by the fact that the dynamics of both nervous systems and the macroscopic physical world are continuous in nature. For example, though many nerve cells fire action potentials (a seemingly discrete event), the current flows that underlie action potentials are continuous quantities. Furthermore, it has long been known that many nerve cells do not produce action potentials but instead communicate using graded potentials [52]. Likewise, the more discrete behavioral phenomena that we observe (e.g., decision making) must eventually be explained in continuous terms. In addition, it is my belief that the versatility and robustness of animal behavior resides in the rich dynamical possibilities of continuous state spaces.

However, it should be noted that any system with finite state which evolves deterministically can be described using the concepts of dynamical systems. Most of the concepts we have used hold in discrete-time systems as well, and many also hold in systems defined on discrete state spaces. For example, the transition table

of a finite state machine defines a flow on a discrete state space. The lack of a metric on this state space limits the dynamical behavior that a finite state machine can exhibit, but such concepts as initial state, trajectory, flow, attractor, equilibrium point, limit cycle, basin of attraction, autonomous and nonautonomous still apply. Thus the present framework may still be useful even if the continuity assumption should turn out to be inappropriate.

*5.2.2. Determinism*

The theoretical framework sketched in this paper is purely deterministic in nature. This determinism derives from the common assumption in science that the macroscopic physical world is in principle completely determined by a knowledge of its state and the dynamical laws that govern its evolution. However, we often say that real environments, real sensors, etc. are somewhat unpredictable. What this usually means is that, because we have only incomplete knowledge of a system's state and laws, we are forced to use stochastic models to describe its behavior. In other words, regardless of whether or not the macroscopic physical world is deterministic in principle, we must sometimes treat it as stochastic in practice.

It should be noted that continuous-time recurrent neural networks show every sign of being robust in the face of unpredictable environmental contingencies. The locomotion controllers can often tolerate the loss of an interneuron or the loss of sensory feedback, and preliminary studies indicate that they are extremely robust to noise on the sensory feedback signal as well. However, regardless of the robustness of these controllers, the question naturally arises as to how the theoretical framework itself might be applied to unpredictable systems.

As I see it, there are two possibilities, depending upon whether a deterministic or stochastic model of unpredictability is adopted. Recall that nonlinear dynamical systems can exhibit dynamics that is completely deterministic in principle but unpredictable in practice (so-called chaotic dynamics). Significant progress has been made on extending the qualitative theory of dynamical systems to the analysis of chaotic dynamics [66]. If the unpredictability of a given system can be modeled with chaotic dynamics, then such techniques can be applied.

Otherwise, we must deal with a fundamentally stochastic model of unpredictability. In this case, we must consider stochastic dynamical systems (see, for example, [7]). Typical concerns in stochastic dynamical systems are understanding how some probability density function over the states of the system evolves with time and determining the asymptotic form of this distribution. The application of such techniques to autonomous agent problems is clearly an important research direction for the future development of the framework.

*5.2.3. Convergence*

In this paper, I have assumed that both the agent and environment dynamics are convergent, that is, the values of their state variables eventually converge to limit sets rather than diverging to infinity. In fact, even a flow that contains divergent regions is acceptable as long as the dynamics of interaction between the

agent and environment never enters such a region. Divergent dynamical systems are a mathematical abstraction anyway. Due to resource limitations and saturation effects, no real physical system is truly divergent. Thus, the assumption of convergent dynamics is, I think, a fairly reasonable one.

However, it is important not to confuse this convergence assumption with the claim that the dynamics of agents and environments must settle onto limit sets before the framework applies. Indeed, the dynamics of the reflexive and mixed pattern generators never settle on an autonomous attractor, but instead are always in a transient because their flow is constantly perturbed by the sensory feedback that they receive. Of course, the dynamics of the coupled agent-environment system in these examples does eventually settle into a limit cycle, but even this need not be the case in general. If either the environment or the agent contains dynamics on time scales that are long relative to the lifetime of the agent, then the entire trajectory of interaction between them will take place on an extended transient.

Even in the case of an extended transient, however, the framework described in this paper still applies. The dynamics of interaction between the agent and its environment is still determined by the global structure of the flow of the coupled system and this structure is itself largely determined by the types and locations of its limit sets. Furthermore, in a system with multiple time scales, there may be a great deal of recurrence to the system's behavior over sufficiently short time scales. Under such conditions, it is often possible to treat the slower state variables as being approximately constant parameters of the faster dynamics and to study the dynamics of this reduced system. Though the attractors of this reduced system are not true attractors of the full system (because the slower state variables are in fact changing), they represent patterns of interaction that may show up repeatedly over sufficiently short time scales (cf. Agre and Chapman's notion of *routines* [3]).

*5.2.4. Low dimensionality*

Our ability to visualize the dynamics of the leg controllers described in Section 4 depended strongly on their relatively low dimensionality. Clearly, we will often need to analyze systems whose dimensionality makes direct visualization of the complete flow impossible. In this case, we must find ways to simplify the dynamics or rely upon nonvisual techniques. One obvious approach (utilized in Section 4) is to visualize higher dimensional dynamics with a set of carefully selected lower dimensional projections. More generally, one can sometimes find a series of coordinate transformations that map the dynamics of a higher dimensional system to a lower dimensional system while preserving most of its global structure. For example, such a technique has been used to reduce the four-dimensional Hodgkin-Huxley model of action potential generation to a two-dimensional system that preserves not only the qualitative behavior of the original system, but most of its quantitative behavior as well [43].

A number of other techniques are available for simplifying dynamical systems by reducing either their dimensionality or complexity in some spatial or temporal

region of interest (e.g., Poincaré maps, center manifolds, normal forms, symbolic dynamics and the use of symmetries [66]). In addition, many of the analytical techniques of dynamical systems theory (e.g., computing limit sets, stability analysis, computation of Lyapunov exponents, etc.) do not require a global visualization of the flow. Finally, it should be pointed out that some progress has even been made on extending the qualitative theory of dynamical systems to infinite-dimensional systems (e.g., those arising from sets of partial differential, delay-differential and integro-differential equations; [34]). These and other techniques will undoubtedly need to be explored as larger systems are studied.

### *5.3. Modularity and design*

Because the only requirement on an agent's dynamics is that the coupled agent-environment system satisfy a given constraint, there is a great deal of freedom both in how a desired dynamics of interaction is divided between the agent and its environment and in the internal organization of the agent's dynamics. We saw examples of this freedom in our analysis of the evolved leg controllers where, despite the fact that both their individual dynamics and the nature of their interaction with the body vary widely, all of these leg controllers produce virtually indistinguishable walking behavior when coupled to the body under normal conditions. Likewise, there is a growing realization within neuroscience that the neural circuits mediating nontrivial behaviors in even simpler animals are highly distributed and nonhierarchical and that traditional engineering principles may not apply to their design [4].

Because evolution only directly selects *against* agent dynamics that do not satisfy their constraint (rather than selecting *for* some optimal design), evolution tends to produce designs that take full advantage of the available freedom. This can lead to designs whose organization is very different from engineered systems. When an engineer designs a complex system, he or she typically performs a hierarchical decomposition of the problem to be solved, resulting in simpler subsystems with clean, well-defined functions and interfaces. Such a modular decomposition is necessary to ensure a correct, reliable and maintainable implementation, and it also appeals to a certain aesthetic sense of parsimony on the part of the designer. Evolution, however, operates under no such constraints. Natural selection preserves those animals that, as a package, work and discards those that do not. Of course, this is not to say that evolved systems are completely unstructured. Certainly, evolved systems exhibit modularity and it is likely that evolution would be unable to produce systems with the complexity of animals without it. However, because the internal organization of an evolved system does not reflect the conceptualization of any designer, whatever modules do exist are under no requirement to exhibit the sort of clean functional organization that we expect from an engineered system.

One might argue that this "messiness" is really just an implementation detail reflecting the blindness of evolution and not a fundamental part of the design. For example, we might be tempted to describe the leg controller dynamics as "really"

just implementing a simple finite state machine that switches between two states labeled "swing phase" and "stance phase". However, it is important to realize that this is not a predictive explanation of these networks, but merely a descriptive summary of their normal operation. The predictive (and therefore explanatory) power of this summary is severely limited by the fact that it fails to capture the underlying dynamical laws that actually govern the system's operation. For example, the temporal patterns of motor neuron activations required to actually make the leg walk are not deducible from such a description. It also fails to capture how sensory feedback is capable of fine-tuning a walking pattern, or why some networks are robust to sensor loss while others are not. Yet these and other features are completely and succinctly explained by the qualitative dynamical analysis summarized in Figs. 8, 9, 10 and 11.

The fact that evolution can produce such messy designs is typically viewed as a shortcoming. However, it is also a source of freedom to cobble together counterintuitive but highly effective, versatile, robust and compact designs. Furthermore, this negative assessment overlooks a crucial difference between the task of an engineer and that of evolution. Engineers solve well-defined problems, and it is the detailed *a priori* specifications of those problems that allow modular solutions to be designed and their parsimony or optimality to be evaluated. In contrast, evolution has no clear specification of the "problem" that any given animal should solve. Even if such a specification existed, it would do little good since the "problem" itself is constantly evolving as the physical environment changes and the coevolution of conspecifics and other species modifies the relationship between behavior and reproductive success. Designing an artificial agent capable of autonomously accomplishing open-ended tasks in an unconstrained real-world environment is much closer to the sort of "problem" solved by evolution than it is to the problems for which traditional engineering methods have been successful.

While evolutionary design shows great promise as a practical method for designing autonomous agents, it is well known that the performance of the standard genetic algorithm does not scale well with the size of the parameter space using the direct encoding of neural network parameters employed here. These scaling problems will need to be addressed before such techniques can be applied to the design of more complicated agents. Toward that end, many different parameter encodings and many different variations of the basic genetic algorithm have been proposed (e.g., [16]) and their application to autonomous agent design needs to be explored. Where appropriate, other search techniques, such as simulated annealing and gradient techniques, should also be explored. When possible, biological data and symmetry considerations can be used to reduce the number of parameters to be searched, as in the adjacent coupling of the leg controllers in the full locomotion controller described in Section 4.1. Another strategy is to search not the parameter space of the agent's dynamics directly, but rather the parameter space of a developmental process that constructs the agent's dynamics (e.g., [35, 50]).

Yet another approach to the scaling problem is to evolve a complex agent

dynamics incrementally. One possibility is to decompose a complex task into a set of simpler subtasks and to independently evolve solutions to these. A complete solution can then be obtained by evolving the coupling between these subnetworks on the original task. Another possibility is to evolve solutions to a simpler version of a difficult problem, successively increase the problem complexity, and then re-evolve the controllers to solve the harder versions of the problem. Finally, attempts to evolve controllers for physical robots instead of simulated agents introduce additional problems, since evolutionary search can only be carried out in simulation at present and there are nontrivial issues involved in transferring controllers from simulated to physical environments [21, 40].

### *5.4. Internal state and representation*

Due to its emphasis on the unpredictable nature and real-time requirements of the real world, a great deal of recent work on autonomous agents has focused on the development of reactive agents whose actions are completely determined by their immediate situation. A purely reactive agent is a degenerate case of the present framework because it maintains no internal state. Rather, it is simply a function from sensory inputs to motor outputs. A reactive agent has no true autonomy because it is constantly pushed around by its environment.

In contrast, a dynamical systems perspective on autonomous agents emphasizes the importance of internal state to an agent's operation. Unlike a reactive agent, an agent with internal state can initiate behavior independently from its immediate circumstances and organize its behavior in anticipation of future configurations of its environment. This ability relies upon the fact that, while it is true that the real world is complicated and somewhat unpredictable, natural environments also exhibit a great deal of structure that a properly designed agent can depend upon and even actively exploit [39]. As a simple example, consider the way in which the reflexive controllers exploit the structure of the body dynamics to achieve walking.

The importance of internal state in the present framework raises an interesting question: Does the framework imply a commitment to internal representation? Much of the debate between proponents and critics of situated agent research has tacitly assumed the equation of internal state with representation, with proponents using criticisms of representation to argue the need for reactivity and critics using the limitations of state-free systems to argue the need for representation (e.g., [20, 44]). But are internal state and representation the same thing? Though this question clearly deserves a more detailed treatment than I can give it here, let me briefly explain why I believe that the answer must be no.

The problem with equating internal state and representation is that computationalism, the theoretical claim that a system's behavior derives from its instantiation of appropriate representations and computational processes [28, 51, 54], then becomes a tautological theoretical position. A scientific hypothesis must be falsifiable, that is, it must be formulated sufficiently clearly to be empirically tested and it must be possible that the test will come out negative. But

all physical systems possess internal state on a variety of timescales. We would presumably hesitate in accepting, say, the temperature of the fuel-air mixture in a cylinder of an automobile engine as a representation of anything, or a thunderstorm as performing any computation. If, on the other hand, we did admit such a broad definition of representation and computation, then computationalism would become true by definition rather than by demonstration and would therefore be making no interesting theoretical claim about the nature of the processes underlying behavior. Similar problems exist with other commonsense notions of representation (e.g., correlation) and computation (e.g., systematicity of the relationship between input and output). For a more detailed discussion of these issues, see [12].

For this reason, representation must require additional conditions above and beyond the mere possession of internal state (or correlation, etc.) and computational systems must therefore be special cases of dynamical systems more generally [30, 61]. There is a great deal of controversy about the particular form of these extra conditions, but the details fortunately need not concern us here. What matters is that the framework's emphasis on internal state, while allowing it to transcend the limitations of purely reactive systems, does not necessarily imply a commitment to internal representation. Rather, the question of whether or not the notion of representation is appropriate for understanding the operation of any particular agent must be settled by an empirical investigation of the internal organization of that particular agent's dynamics. The framework is thus, strictly speaking, agnostic about the theoretical roles of representation and computation in the design and analysis of autonomous agents.

### *5.5. Conclusion*

This paper has largely been presenting an argument about language, namely the language that we use to talk about autonomous agents and their environments. In particular, my primary goal has been to demonstrate that many of the ideas emerging from recent work on autonomous agents (as well as work on the neural basis of animal behavior, though I have not emphasized this aspect here) can be naturally cast into the language of dynamical systems theory.

One must never underestimate the power of language in shaping the way we think about the world. Our theoretical languages provide the metaphors we use to conceptualize the very phenomena we seek to understand. A computational language invites us to think about representations and their manipulation. Using this language, we become concerned with the structure of representations, where they are stored, how they are retrieved, what they mean, etc. From a computational perspective, observed regularities in an agent's behavior become windows into its program. If, for example, an agent persistently acts toward some end, then, computationalism tells us, it must be by virtue of possessing an internal representation of that goal. From a computational perspective, perception becomes a problem of reconstructing an accurate internal representation of the external environment. Taking action becomes a problem of constructing and

executing representations of the actions to be performed. Learning becomes a problem of modifying existing representations and accumulating new ones. And so on.

This paper has only just begun the difficult task of developing a dynamical language for these and other phenomena exhibited by autonomous agents. A great deal of work remains to be done in developing this framework into a full-fledged theory. If this framework is to succeed in providing a foundation for our understanding of autonomous agents, then specific dynamical accounts of perception, action, goal-oriented behavior, decision-making, sequential behavior, learning, etc. will need to be developed and these accounts will need to be applied to specific agents, both natural and artificial. I strongly suspect that a dynamical perspective on these phenomena will significantly change the way we think about them.

Ultimately, like all work on embodied agents, the framework must face the fact that people can deliberately form and reason with conceptual representations. While I have taken the position that such intellectual capabilities are relatively recent elaborations of a far more fundamental capacity for situated action (and are therefore not nearly so crucial as is usually assumed for even highly complex but nondeliberative behavior), they must nevertheless eventually be explained. Will attempts to extend a dynamical perspective to such cognitive behavior as language and abstract reasoning turn out to require the implementation of computational processes on top of the dynamical substrate responsible for situated action? Or will the very way we think about such cognitive behavior, and the notions of representation and computation that currently seek to underwrite it, also have to change in the process?

## Acknowledgments

I would like to thank Phil Agre, Hillel Chiel, John Gallagher, Ken Loparo, Leslie Picardo, and Beth Preston for many useful discussions and for their comments on earlier drafts of this paper. I would also like to thank the reviewers, as well as the attendees of the Workshop on Computational Theories of Interaction and Agency, for their many helpful suggestions. The simulation experiments reviewed in Section 4 were carried out in collaboration with John Gallagher. This work was supported by grant N00014-90-J-1545 from the Office of Naval Research. Additional support was provided by the Howard Hughes Medical Institute and the Cleveland Advanced Manufacturing Program through the Center for Automation and Intelligent Systems Research.

## References

[1] R.H. Abraham and C.D. Shaw, *Dynamics—The Geometry of Behavior* (Addison-Wesley, Redwood City, CA, 2nd ed., 1992).

[2] P. Agre, The dynamic structure of everyday life, Technical Report 1085, MIT AI Lab., Cambridge, MA (1988).
[3] P.E. Agre and D. Chapman, Pengi: an implementation of a theory of activity, in: *Proceedings AAAI-87*, Seattle, WA (1987) 268–272.
[4] J.S. Altman and J. Kien, Highlighting *Aplysia*'s networks, *Trends Neurosci.* **13** (3) (1990) 81–82.
[5] M.A. Arbib and J.-S. Liaw, Sensorimotor transformations in the worlds of frogs and robots, *Artif. Intell.* **72** (1995) 53–79.
[6] W.R. Ashby, *Design for a Brain* (Wiley, New York, 2nd ed., 1960).
[7] K.J. Åström, *Introduction to Stochastic Control Theory* (Academic Press, New York, 1970).
[8] D.H. Ballard, Animate vision, *Artif. Intell.* **48** (1991) 57–86.
[9] A.G. Barto, S.J. Bradtke and S.P. Singh, Learning to act using real-time dynamic programming, *Artif. Intell.* **72** (1995) 81–138.
[10] K. Basye, T. Dean and L.P. Kaelbling, Learning dynamics: system identification for perceptually challenged agents, *Artif. Intell.* **72** (1995) 139–171.
[11] R.D. Beer, *Intelligence as Adaptive Behavior: An Experiment in Computational Neuroethology* (Academic Press, San Diego, CA, 1990).
[12] R.D. Beer, Computational and dynamical languages for autonomous agents, in: T. van Gelder and R. Port, eds., *Mind as Motion* (MIT Press, Cambridge, MA, to appear).
[13] R.D. Beer and J.C. Gallagher, Evolving dynamical neural networks for Adaptive behavior, *Adaptive Behav.* **1** (1992) 91–122.
[14] R.D. Beer, R.E. Ritzmann and T. McKenna, eds., *Biological Neural Networks in Invertebrate Neuroethology and Robotics* (Academic Press, San Diego, CA, 1993).
[15] R.D. Beer, H.J. Chiel, R.D. Quinn, K.S. Espenschied and P. Larsson, A distributed neural network architecture for hexapod robot locomotion, *Neural Comput.* **4** (3) (1992) 356–365.
[16] R.K. Belew and L.B. Booker, eds., *Proceedings of the Fourth International Conference on Genetic Algorithms* (Morgan Kaufmann, San Mateo, CA, 1991).
[17] L.B. Booker, Classifier systems that learn internal world models, *Mach. Learn.* **3** (1988) 161–192.
[18] R.A. Brooks, A robust layered control system for a mobile robot, *IEEE Trans. Rob. Autom.* **2** (1986) 14–23.
[19] R.A. Brooks, A robot that walks: emergent behaviors from a carefully evolved network, *Neural Comput.* **1** (2) (1989) 253–262.
[20] R.A. Brooks, Intelligence without representation, *Artif. Intell.* **47** (1991) 139–159.
[21] R.A. Brooks, Artificial life and real robots, in: *Toward a Practice of Autonomous Agents: Proceedings of the First European Conference of Artificial Life*, Paris, France (1992) 3–10.
[22] D. Chapman, *Vision, Instruction and Action* (MIT Press, Cambridge, MA, 1991).
[23] D. Chapman and L.P. Kaelbling, Input generalization in delayed reinforcement learning: an algorithm and performance comparisons, in: *Proceedings IJCAI-91*, Sydney, Australia (1991) 726–731.
[24] D. Cliff, Computational neuroethology: a provisional manifesto, in: *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, Paris, France (1991) 29–39.
[25] F. Delcomyn, Neural basis of rhythmic behavior in animals, *Science* **210** (1980) 492–498.
[26] M. Donner, *Real-Time Control of Walking* (Birkhauser, Boston, MA, 1987).
[27] J.L. Elman, Distributed representations, simple recurrent networks and grammatical structure, *Mach. Learn.* **7** (1991) 195–225.
[28] J.A. Fodor, *The Language of Thought* (Harvard University Press, Cambridge, MA, 1975).
[29] J.C. Gallagher and R.D. Beer, A qualitative dynamical analysis of evolved locomotion controllers, in: J.-A. Meyer, H. Roitblat and S. Wilson, eds., *From Animals to Animats 2: Proceedings of the Second International Conference on the Simulation of Adaptive Behavior* (MIT Press, Cambridge, MA, 1993) 71–80.
[30] M. Giunti, Computers, dynamical systems, phenomena and the mind, Ph.D. Thesis, Indiana University, Bloomington, IN (1992).

[31] D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning* (Addison-Wesley, Reading, MA, 1989).
[32] D. Graham, Pattern and control of walking in insects, *Adv. Insect Physiol.* **18** (1985) 31–140.
[33] J.K. Hale and H. Koçak, *Dynamics and Bifurcations* (Springer-Verlag, New York, 1991).
[34] J.K. Hale, L.T. Magalhaes and W.M. Oliva, *An Introduction to Infinite Dimensional Dynamical Systems—Geometric Theory* (Springer-Verlag, New York, 1984).
[35] S.A. Harp, T. Samad and A. Guha, Towards the genetic synthesis of neural networks, in: *Proceedings Third International Conference on Genetic Algorithms*, Fairfax, VA (1989) 360–369.
[36] I. Harvey, Untimed and misrepresented: connectionism and the computer metaphor, Technical Report CSRP 245, School of Cognitive and Computing Sciences, University of Sussex (1992).
[37] J.H. Holland, *Adaptation in Natural and Artificial Systems* (University of Michigan Press, Ann Arbor, MI, 1975).
[38] J.J. Hopfield, Neurons with graded response properties have collective computational properties like those of two-state neurons, *Proc. Nat. Acad. Sci.* **81** (1984) 3088–3092.
[39] I. Horswill, analysis of adaptation and environment, *Artif. Intell.* **73** (1995), to appear.
[40] P. Husbands and I. Harvey, Evolution vs. design: controlling autonomous robots, in: *Proceedings Third Annual Conference on AI, Simulation and Planning*, Perth, Australia (1992) 139–146.
[41] L.P. Kaelbling, *Learning in Embedded Systems* (MIT Press, Cambridge, MA, 1993).
[42] L.P. Kaelbling and S.J. Rosenschein, Action and planning in embedded agents, *Robotics and Autonomous Systems* **6** (1–2) (1990) 35–48.
[43] T.B. Kepler, L.F. Abbott and E. Marder, Reduction of conductance-based neuron models, *Biol. Cybern.* **66** (1992) 381–387.
[44] D. Kirsch, Today the earwig, tomorrow man?, *Artif. Intell.* **47** (1991) 161–184.
[45] P. Maes, ed., *Designing Autonomous Agents* (MIT Press, Cambridge, MA, 1990).
[46] S. Mahadevan and J. Connell, Automatic programming of behavior-based robots using reinforcement learning, in: *Proceedings AAAI-91*, Anaheim, CA (1991) 768–773.
[47] H.R. Maturana and F.J. Varela, *Autopoiesis and Cognition* (Reidel, Boston, MA, 1980).
[48] H.R. Maturana and F.J. Varela, *The Tree of Knowledge* (Shambhala, Boston, MA, 1987).
[49] J.-A. Meyer and S.W. Wilson, eds., *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior* (MIT Press, Cambridge, MA, 1991).
[50] G.F. Miller, P.M. Todd and S.U. Hegde, Designing neural networks using genetic algorithms, in: *Proceedings Third International Conference on Genetic Algorithms*, Fairfax, VA (1989) 379–384.
[51] A. Newell and H.A. Simon, Computer science as empirical inquiry: symbols and search, *Commun. ACM* **19** (1976) 113–126.
[52] K.G. Pearson, Nerve cells without action potentials, in: J.C. Fentress, ed., *Simpler Networks and Behavior* (Sinauer, Sunderland, MA, 1976).
[53] J.B. Pollack, The induction of dynamical recognizers, *Mach. Learn.* **7** (1991) 227–252.
[54] Z.W. Pylyshyn, *Computation and Cognition* (MIT Press, Cambridge, MA, 1984).
[55] J. Rinzel and G.B. Ermentrout, Analysis of neural excitability and oscillations, in: C. Koch and I. Segev, eds., *Methods in Neuronal Modeling* (MIT Press, Cambridge, MA, 1989).
[56] S.J. Rosenschein, Formal theories of knowledge in AI and robotics, *New Gen. Comput.* **3** (4) (1985) 345–357.
[57] G. Schöner and J.A.S. Kelso, Dynamic pattern generation in behavioral and neural systems, *Science* **239** (1988) 1513–1520.
[58] C.A. Skarda and W.J. Freeman, How brains make chaos in order to make sense of the world, *Behav. Brain Sci.* **10** (1987) 161–195.
[59] T. Smithers, Taking eliminative materialism seriously: a methodology for autonomous systems research, in: *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, Paris, France (1992) 31–40.
[60] P. Smolensky, On the proper treatment of connectionism, *Behav. Brain Sci.* **11** (1988) 1–74.
[61] T. van Gelder, What might cognition be if not computation?, Technical Report 75, Indiana University Cognitive Science, Bloomington, IN (1992).

[62] X.-J. Wang and J. Rinzel, Alternating and synchronous rhythms in reciprocally inhibitory model neurons, *Neural Comput.* **4** (1992) 84–97.
[63] N. Wiener, *Cybernetics* (MIT Press, Cambridge, MA, 2nd ed., 1961).
[64] G. Wendler, The influence of proprioceptive feedback on locust flight coordination, *J. Comput. Physiol.* **88** (1974) 173–200.
[65] G.M. Werner and M.G. Dyer, Evolution of communication in artificial organisms, in: C.G. Langton, C. Taylor, J.D. Farmer and S. Rasmussen, eds., *Artificial Life II* (Addison-Wesley, Reading, MA, 1991).
[66] S. Wiggins, *Introduction to Applied Nonlinear Dynamical Systems and Chaos* (Springer-Verlag, New York, 1990).
[67] S.W. Wilson, Knowledge growth in an artificial animal, in: K.S. Narendra, ed., *Adaptive and Learning Systems* (Plenum Press, New York, 1986).