React fast, think slow

The relation between pro-activity and environmental structure in autonomous behavior

Jeroen Koomen

0806978

juli 2002

begeleiders:

Lambert Schomaker (KI)

Eric Postma (IKAT, Universiteit Maastricht)

Kunstmatige Intelligentie

Rijksuniversiteit Groningen

968

CONTENTS

1 Introduction

Behavior in the natural world has been described in many different ways and the descriptions have been attacked from many different angles. This gave rise to a multitude of theories about biological behavior. One of these theories that is receiving more and more attention from the cognitive science society is called the *Enactivist Paradigm* (Varela, Thompson & Rosch, 1991). Within the enactivist paradigm the definition of behavior is centered around the notion of *interaction* between an agent and its surroundings. With interaction, every action on part of the agent leads to a change in the agent itself and/or in its perceived surroundings and those changes are used for subsequent action. Theoretical biologist Jacob von Uexküll defined behavior as follows:

> *The sensation of the mind become properties of things during the construction of the world, or, one could also say, the subjective qualities construct the objective world.*
> [*J.v.Uexküll*, taken from Prem (1996)] [1]

Within this *enactivist* definition of behavior (to be discussed in chapter 2) we distinguish between *reactive* and *pro-active* influences on behavior. The difference between reactive and pro-active behavior is causal. In reactive behavior, the actions of an agent are caused by sensory patterns picked up from the environment. In pro-active behavior, however, an internal source of activation causes the actions. The internal source can be a belief, an emotional state, or any other motivational force. Reactive and pro-active behavior can be characterized as environment-driven and agent-driven, respectively. In biological organisms both types of behavior have to be appropriately balanced in order to survive. Presumably, the optimal balance depends on the complexity of the organism and its environment.

In this thesis a model of behavior is investigated which distinguishes between reactive and pro-active processes by focusing on the following research questions:

(1) Is the balance between the level of pro-activity and the level of reactivity dependent on the nature of the environmental task?

(2) In what way is the level of pro-activity dependent upon the environment in which overall behavior is evolved?[2]

To answer these questions two experiments with a simulated autonomous robot have been performed. In these experiments, the robot has to learn to manage the resources that are present in the environment. The behavior of the robot is controlled by a neural network. The space of possible neural weight configurations is searched by means of an evolutionary algorithm. The search is guided by a fitness function that rewards appropriate behavior for the task at hand.

The purely agent-driven pro-active behavior is more ignorant of sensory patterns and is therefore more or less blind behavior. But it adds an additional degree of freedom to overall behavior and can become functional under certain circumstances. The agent moves in the environment and this produces changes in the sensory patterns. Based on this change in external information, or *external dynamics*, the level of pro-activity changes accordingly. The pro-active behavior can be said to have a certain *structure*, or shape, depending on the shape and weight configuration of the neural network that produces it. Also, the environment in which the agent finds itself has a certain structure. Some of this structure could be *usable* for the agent, which makes it *usable structure*. The relation between the structure of the pro-activity and of the environment could say something about the usefulness of the pro-active behavior.

The outline of the thesis is as follows. Chapter 2 describes the theoretical background and the neural-network modeling of behavior in an autonomous agent. In chapter 3 the two experiments are described. In particular, the neural-network architecture and evolutionary algorithm are outlined. In chapter 4 the results of the 3 experiments are discussed and related to the two research questions. Finally, chapter 5 concludes by stating that the use of motivation and the resulting pro-active behavior depends on the structure, and more precisely, the *usable* structure in the environment.

---

[1] Quote by Uexküll, translated from German by Prem, E.

[2] The term 'evolutionary task' is used for describing a task that is embedded within an environment and is evolved over time. So with this term more is meant than merely the abstract or logical description of a task.

2 Theoretical background

This chapter is divided in three parts. The first part begins with an outline of the enactive paradigm. The second part considers various behavioral issues from an enactivist point of view. These issues will be crucial for understanding the model of behavior, presented in part three, in which behavior is argued to be dynamic interaction between an agent and its environment. A special emphasis is put on the role of motivation within such interaction.

2.1 The enactivist paradigm

Nowadays, cognitive science is a well-founded science encompassing several paradigms. We distinguish three paradigms. The first is the *cognitivist paradigm*, often associated with the symbolic approach to cognition. The second is the *connectionist paradigm*, which came out of the connectionist 'revolution' in the 1980's. The third paradigm is relatively new to the field of cognitive science and is known as the *'enactive' paradigm*. The enactive paradigm views mind as embodied action. In the book "The Embodied Mind" (Varela, Thompson and Rosch, 1991) the enactive paradigm is described and visual perception is taken to illustrate enaction. The problem with visual perception, as argued in the book, is that two stances can be taken: the objectivist stance and the subjectivist stance. According to the objectivist stance 'color', for instance, is a property of some object 'out there', whereas according to the subjectivist stance, color is a quality of the internal world of the perceiver projected upon some object. The enactive paradigm combines both stances by assuming that world and perceiver specify each other. Like color categories, perceived qualities depend on our perceptual and cognitive capacities but also on our biological and cultural environment. So this implies a mutual specification of the 'external' world and the 'internal' perceiver, or mind. Because of this mutual specification, the external world is not merely a source of information to be used by an agent, but becomes *fundamental* to the overall behavior of the agent. The mutual dependence of animal and its environment was already addressed by the theoretical biologist Jakob von Uexküll (Uexküll, 1982). He claimed that there is no possibility to understand an animal's behavior without its environment, nor the environment without any animals put therein. Another way of saying that the agent and environment specify each other is to say that the agent is *embedded* in the environment. The importance of the environment will be further described in 2.2.3 and will become apparent when the results of the experiments are given in chapter 4.

A concise definition of the enactive view is given by Varela (1993):
"In a nutshell, the enactive approach consists of two points: (1) perception consists in perceptually guided action and (2) cognitive structures emerge from the recurrent sensorimotor patterns that enable action to be perceptually guided." We see that perception plays a central role in the definition of the enactive view.

According to the enactivist paradigm, the mutual specification of agent and environment, which can be described as the *structural coupling* of agent and environment, is not merely an aspect of cognition but *is* cognition itself. This sets it apart from the cognitivist paradigm, which defines cognition as information processing as symbolic computation and rule-based manipulation of symbols. The connectionist paradigm views cognition as the emergence of global states in a network of simple components. Although the same could happen within a enactivist approach to cognition, it misses the fundamental link with the environment in which the global states were allowed to form.

Clark (1997) gives a similar comparison between different approaches to cognition. He describes three different stages in the development of cognitive science and gives some key characteristics. Clark clearly seems to be an enactivist when he states that real embodied intelligence is a means of *engaging* with the world. He further states that real embodied intelligence uses active strategies that leave much of the information out in the world and then uses *iterated, real-time sequences of body-world interactions to solve problems in a robust and flexible way*. Clark sees this kind of intelligence as one produced by the joint activity of two coupled complex systems (the agent and the environment) and in such cases it would make little sense of speaking of one system's *representing* the other.

The enactivist paradigm sometimes has been called the emergentist approach or the embodiment/situatedness approach to cognition and intelligence. Since the paradigm is rather new,

variations in definitions and labels are very common.

Perception plays a central role within the enactivist paradigm. Therefore the concept of perception, along with the associated concept of *representation* will be addressed.

2.1.1 Perception and Representation

"The Eye altering alters all."
William Blake, *The Mental Traveler*, (1800-10)

Perception deals with how animals or agents connect with their surrounding environment. Representation deals with how, once the connection between agent and environment made, aspects of this connection are sustained over time within the agent itself.

What is perception? We can define perception as the interaction with the collection of stimuli on which an organism can (or must) (re)act. Stimuli can be anything from visual retinal activation to an internal feeling of hunger. Everything on which an organism or agent can base its actions can be seen as perceptual information. As these actions lead to changes in the environment and/or in the agent itself and the agent can perceive these changes, it can be said that these changes in perception mirror or even stand for the interaction between the agent and its environment and/or itself. In other words, by behaving, the agent partly defines its own perception. When the agent changes its perception, the environment will also change *according to the agent*. One can see that this description of perception is an enactivist one. Franklin (1995) has put it this way:

"Mind operates on sensations to create information for its own use. I do not think of minds as information-processing machines in the sense of taking information from the environment and processing it to arrive at the next action. Rather, I think of information as not existing out there in the environment. Information comes into being when minds process sensations (Oyama 1985). The same scene can provide quite different information to different minds."

In his book, Franklin gives an example of a case study that gives support to the enactivist approach:

"In a case study by Held and Hein, kittens were raised in the dark and were exposed to light only under controlled conditions. A first group of animals was allowed to move around normally, but each of them was harnessed to a simple carriage and basket that contained a member of the second group of animals. The two groups therefore shared the same visual experience, but the second group was entirely passive. When the animals were released after a few weeks of this treatment, the first group of kittens behaved normally, but those who had been carried around behaved as if they were blind: they bumped into objects and fell over edges. This beautiful study supports the enactive view that objects are not seen by the visual extraction of features but rather by the visual guidance of action."

In the cognitivist paradigm, perception is seen as the extraction of perceptual information from the environment. The information is then transformed into some representation of the object perceived. In the enactivist paradigm, the fact that an agent is *embedded* in its environment plays a crucial role. In the experiments outlined in chapter 3, we will see that neural controllers are created by evolving the connection weights of a neural network. Over time, the weights in the network will come to *represent* the behavior of the agent. Such a representation is much more abstract. The weights of the network will come to represent the evolutionary task *given* a world in which such a task is *possible*. Without the world, the weights do not mean very much. So representing becomes more a mechanism of enacting. Instead of representing an independent world, the agent brings forth a world by interacting with it. And the representation that is present in the network is not planned or controlled by a central system. It emerges from the interactions among the elements of the network and between the network and the environment. Such emergence is labeled 'indirect emergence' (Clark, 1997), because it is heavily dependent upon environmental structures. The emergence of this representation of the evolutionary task enables the agent to carry out its task giuded by perception. Or as Varela (1993) stated in the definition of the enactivist paradigm: "cognitive structures emerge from the recurrent sensorimotor patterns that enable action to be perceptually guided."

In 2.2 the fundamental behavioral issues that will be used in the model described in 2.3 and in the experiments in chapter 3 will be addressed. As we will see, the descriptions of the various behavioral concepts are enactivist in nature.

2.2 Behavior

What is meant with the term 'behavior'? According to the enactivist paradigm, behavior means interaction. Biological organisms are in constant interaction with the world and with themselves. The presence alone of an organism in an environment can be seen as interaction in the sense that its presence has impact on the environmental situation. (For instance, its presence can be perceived by other organisms, air currents change according to its presence, the organism absorbs light photons and so on.)

So behavior is interaction. This interaction can be internal and external. Internal interaction means that an agent reacts on internal processes, e.g. heart-rhythm regulation. External interaction means interaction with the environment, the outside world. The quality of interaction could be defined by the usability of sensory patterns it produces. So the world influences the behavior of organisms but organisms, through their structure, will define the way this is achieved.

As the agent interacts with the environment, a collection of stimuli that are either internal or external drives perception, and therefore behavior. A classification within the collection of stimuli that I use is one of *positive* and *negative* stimuli. I will define positive stimuli as stimuli that give information about the *presence* of something. Negative stimuli are then stimuli that give information about the *absence* of something. An example of this distinction will follow when I discuss the model of behavior used in the experiments.

These comments on behavior suggest a purely *reactive* nature of behavior. Stimuli drive perception and behavior and there seems to be no room for free will or self-motivated behavior. But natural behavior displays many features of self-motivated, even creative behavior. This kind of behavior arises from some internal source of activation. In the next sections, aspects of natural behavior are described that will ultimately be used in the model of behavior outlined in 2.3.

2.2.1 Motivation and pro-active behavior

Motivation is a central characteristic of the model of behavior proposed in this thesis. Motivation is the internal mechanism that provokes pro-active behavior. Pro-activity typically refers to behavior that is determined by an internal source of activation and at most only partially determined by external stimuli. Wooldridge (1995) defines pro-activity as *goal-directed behavior by taking the initiative*. He relates pro-activeness to the measure of flexibility an agent is able to attain. He describes a purely reactive agent as one that decides what to do based entirely on the present (without reference to its history).

Aaron Sloman has investigated the subjects of motivation and free will within behavior. Both concepts will be discussed in terms of Sloman's work.

Motivation

Early work on motivation in behavior includes Aaron Sloman's "Motives, Mechanisms and emotions" (Sloman, 1987). In this paper, Sloman gives an analysis of the role of emotions and motives in mind and behavior. He gives design constraints for intelligent machines and shows how they can be related to the structure of human motivation and to computational mechanisms underlying emotional states. He claims that no special subsystems are required to account for emotions and that mechanisms underlying intelligence suffice. Therefore he challenges the gap between emotion and cognition. In terms of the paradigms discussed before, Sloman is somewhere in between cognitivism and connectionism. Sloman offers a computational theory of emotions, attitudes, moods, and motivation. Despite the quite cognitivist nature of his work, many of his beliefs and definitions seem

useful. In particular, the term 'motivator' will be used in chapter 3 to refer to the node in the neural network that produces the neural activation underlying the pro-active part of behavior. Sloman gives the following definition of the term 'motivator':"...I use the term 'motivator' to refer to mechanisms and representations that tend to produce or modify or select between actions, in the light of beliefs."

In this definition, the presence of 'beliefs' is crucial for setting *motivated* behavior apart from *non-motivated* behavior. Although the term 'beliefs' is quite a vague one, we could make it less vague by interpreting the way certain conditions activate the *internal motivator* as a belief about what kind of behavior is functional. So the way the internal motivator is implemented can be seen as representing a belief about when it is functional to become pro-active. (Note: the *internal motivator* is the implementation of the internal source of activation, responsible for creating pro-active behavior. More on this follows in section 2.3 where we discuss the model of behavior.)

Another term that is often associated with motivation is *free will*. When behavior is not totally reactive but is also determined by some internal source of activation, it is argued that the behavior is exhibiting a form of free will.

According to Sloman, free will is a matter of degree, not an all-or-none phenomenon. Sloman illustrates this in terms of a list of design distinctions for the design of intelligent machines. He states that by examining a range of possible designs for intelligent systems the distinction between systems with or without free choice becomes very obscure. He comes to the conclusion that there are many "lesser distinctions corresponding to design decisions that a robot engineer might or might not take". Thus, free will is argued to be continuous in nature, rather than an all-or-none phenomenon. The same can be said for pro-activity. It is not the question whether behavior is pro-active or not, but *how much* pro-active it is.

The continuous nature of free will, and also, pro-active behavior, is fundamental to the way pro-active behavior is implemented in my experiments, which are described in the chapter 3.

The next section deals with the distinction between reactive and pro-active behavior.

2.2.2 Reactive vs. Pro-active behavior

In the previous section, pro-active behavior was described as behavior that is not purely based on external cues, but also on some internal source. *Reactive* behavior can be defined in a similar way. The distinction between reactive and pro-active behavior can be based on the *source* of such behavior. This source can be either external or internal. For instance, an external source is a collection of sensory patterns created by interaction with the environment. In that case, an agent is reacting to sensory patterns from the environment. In case the source of behavior is internal, the agent does not react to external sensory patterns. Rather, the internal source *motivates* the agent to become (pro-) active, more or less independent of external stimuli.

When Sloman (1987) defines the term 'motivator' as a mechanism that selects between actions in the light of beliefs, he means that the activity that follows such selection is based on some internal state or structure (internal with respect to the agent of course). Another word for activity that is based on an internal source of activation is *pro-activity*. Roughly speaking, pro-activity may be useful in three situations:

- when the environment is 'poor', i.e., when the sensory patterns received from the environment are insufficient to generate adequate or functional behavior,
- when the pro-activity can add functionality to overall behavior to exploit some feature of the interaction which pure reactive behavior can not,
- when it is triggered by some internal physiological signal, such as hunger or thirst.

In the first (poor environment) case, the pro-activity could be triggered by an absence of stimuli, rather than a presence. Previously, in section 2.2, a distinction was made between *positive* and *negative* stimuli. We can now state that pro-active behavior can be triggered by *negative* stimuli, in which case reactive behavior has a shortage of *positive* stimuli. So in such a case pro-active behavior can replace the reactive behavior. When an organism becomes pro-active, its behavior is not totally environment-driven, but is also driven by an internal source. This behavior resembles a kind of internal drive or force, avoiding the organism to become too passive. The internal drive can also be seen as the belief of the organism that behavior, produced by that internal drive, will be functional (e.g. will increase its chances of survival).

In the second case, the question is more whether the pro-active behavior can add functionality

to overall behavior that can exploit some feature of interaction. In some cases, the pro-active behavior can have a quality that becomes useful in the light of some task performed within a certain environment. In that case, pro-activity increases performance.

The third case is maybe the most intuitive when thinking about motivated biological behavior. When an organism gets hungry, the sensation of hunger will tend to drive the organism to search for food.

So we can divide behavior of an organism into two categories: reactive and pro-active. Reactive has been argued to be based on (positive) environmental stimuli whereas pro-active has been argued to be based on an internal source of activation (triggered by negative stimuli). Both can be seen as perception-driven behavior, because the agent automatically perceives the internal source of activation since it is part of the agent. Also, the functionality of any pro-active behavior will depend upon the environment in which behavior takes place, and more particular, upon the *structure* of the environment.

We know that the stimuli that drive perception determine for a large part the success of the behavior. Therefore, some subclasses of stimuli will allow more functionality of pro-active behavior than others. It would be advantageous for an agent if it could somehow *seek out* these subclasses and increase the functionality of its pro-active behavior. Nolfi *et al.* (Nolfi and Parisi, 1993; Nolfi, 1999, 2000) have shown that agents can develop this ability through learning.

Self-selection of stimuli

The concept of 'self-selection of stimuli' is an important explanatory tool that I use in chapter 4. It is a term that comes from the field of *selective attention*. One of the people that has done research in the field of 'selective attention' is Stefano Nolfi. Nolfi (2000) described the ability of a reactive agent to *select* sub-classes of sensory patterns purely by sensory-motor co-ordination. In particular, Nolfi shows that agents can take advantage of their sensory-motor abilities in various ways. For instance, agents that exploit the power of sensory-motor co-ordination can overcome the perceptual aliasing problem. This problem arises in ambiguous environments where different environmental situations generate the same sensory patterns. As a result, the agent is not able to distinguish between the two situations purely based on its sensory stimuli. One solution would be to use action to look for additional sensory patterns that are not ambiguous and would solve the problem (see Pfeifer and Scheier, 1999). The process in which action is used to select sensory patterns is called *active perception*. Examples of this process have been discovered throughout nature. The fruit fly *drosophila* moves its body with respect to an object in order to shift the object within a certain area in the visual field (Nolfi, in press).

In Nolfi and Parisi (1993), the authors argue that agents behaving in an environment can improve performance not only by improving their ability to react efficiently to stimuli, but also by acquiring an ability to expose themselves to a sub-class of stimuli to which they know how to respond efficiently. In one of their experiments, agents controlled by evolved neural networks live in a grid world containing food elements. Performance was rated by the agent's ability to find food in an efficient way. The evolved behavior was analyzed by looking at the performance increase due to two factors: (a) selecting a restricted class of stimuli to which the agents know how to react and (b) the ability to approach food. They found that early in evolution the agents rely more on their ability to select a favorable sub-class of stimuli. The performance due to the ability to approach food increased more gradually. Their results suggest that agents that have an inherited ability to seek out favorable sub-classes of stimuli have an evolutionary advantage.

These results showed that agents could learn to seek out stimuli on which they know how to react. But there is no reason to assume that the same does not hold for pro-active behavior. Agents can learn to seek out sub-classes of stimuli on which they know how to *pro*-act. Such agents will interact with their environment to increase the likelihood of entering behavioral regions in which pro-active behavior is more functional.

2.2.3 Environment and structure

"Our body is in the world as the heart is in the organism, it forms with it a system."
-- Maurice Merleau-Ponty, Phenomenology of Perception

Meaning emerges only when something is placed into a context. When behavior is placed outside an environmental context, it has no meaning. So the role of the environment is essential in evaluating certain behavior of an agent. We have seen that according to the enactivist paradigm, behavior is interaction and mind is seen as embodied action. This makes the environment a necessary component of behavior.

How must we treat the environment? The classical approach is to view the environment as a collection of information that can be extracted by a perceptive agent. Another view which is closely related to von Uexküll's notion of *Umwelt* states that the environment as an objective resource of data is not very functional for investigating animal and human cognition. Rather, the environment *according to the agent* (Umwelt) becomes the definition of an environment when considering cognition. The mutual specification of agent and environment can also be described with the term 'structural coupling' (Varela *et al*, 1993).

Structural coupling and feedback

Structural coupling means the coupling of the agent's structure to the world. If the structure is changed, its relation to the world is changed, so the world is changed. Structural coupling is intricately connected with the notion of feedback. Cyberneticists distinguish two different kinds of feedback: self-balancing (or 'negative') feedback and self-regulatory (or 'positive') feedback. An example of such a feedback loop is given in (Capra, 1996). Capra gives a description of a device called the 'centrifugal governer'. The mechanical device is illustrated in figure 2.1 and was used as a system for the control of the flow of steam in a steam engine. It consists of a rotating spindle with two flyballs attached to it. When the rotational speed of the spindle increases as a result of the increased flow of steam, the two balls move outwards. When this happens, a piston is pulled upwards by the outward movement of the two flyballs and this cuts off part of the steam flow. This results in diminished force acting upon the rotating spindle and so the flyballs move inwards again, allowing the flow of steam to increase. In this manner a balance will emerge letting the steam engine work with a constant flow of steam.

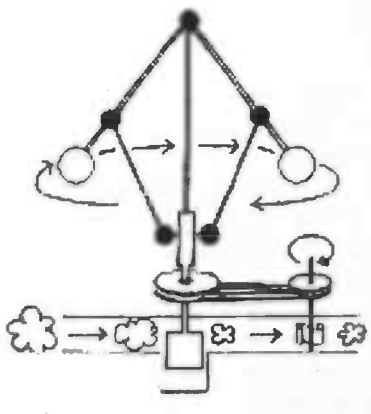


FIGURE 4-4 Centrifugal governor

Fig. 2.1[1] The Centrifugal Governer

The dynamics of this feedback loop can be easily drawn as a loop diagram shown in fig. 2.2. Each part of the chain has a plus- or minus sign, according to the kind of feedback involved (positive of negative).

[1] The figure is taken from *The Web of Life*, Capra F.

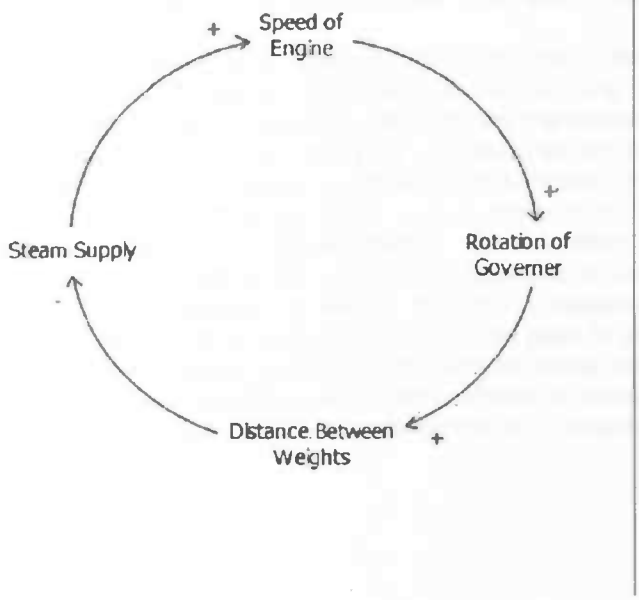


Fig. 2.2 Loop diagram of the centrifugal governer

A similar kind of feedback loop can be drawn for an embedded agent. For example, an organism running through a terrain. The increase in running speed increases air friction, which has a negative feedback on the effort required for maintaining the running speed. Also, oxygen supply to muscled will increase until shortage of oxygen decreases the amount of work produced by the muscles. This is another example of a self-regulatory mechanism. Similar examples are abundant through nature and we will come across another such example when discussing the implementation of pro-active behavior in chapter 3.

Structure

Any environment has certain characteristics such as size, shape, regularity, diversity and so on. All these characteristics define the *structure* of the environment, or *environmental structure*. This may be seen as intrinsic to the environment and has an objective character. But as we have seen, within the discussion of behavior, it is not the *objective* environment that is important, but the environment *according to the agent*. Part of the *environmental structure* will be *useful* to the agent, while another part will be not. The part of the environmental structure that can be used by an agent we will call *usable structure*. This structure has a more subjective nature, since it is usable only *according to the agent*. So there is a difference between the, more objective, *environmental* structure that is present in the world, and the, more subjective, *usable* structure that is functional for the agent. This difference is closely related to the difference between *Umwelt* and *Real World*, described by von Uexküll. The *usability* of environmental structure is created by the agent-environment interaction and is not an intrinsic quality of the environment. The same environmental structure may have completely different levels of usability to different agents.

A similar distinction was made by in (Fletcher, Zwick and Bedau, 1996) in which the way a population of agents can adapt to environments with different levels of structure is described. In this paper, it is said that it is not the amount of intrinsic structure that is present in the environment that really counts, but how much of this structure is useful to the agent. Two aspects of the usefulness of environmental structure are distinguished: *ambiguity* and *value*. The ambiguity of an environment reflects the level in which the same environmental setting will have the same adaptive significance to the agent's behavior. If for instance a certain area of the environment allows optimal behavior of the agent in one instance, but not in another, the environment is ambiguous. The value of the environment is the amount of gain that can be achieved by adapting to that environment.

A difference with the definitions of usability of structure I use is that I state that the *usability* of environmental structure is determined by interaction and is not an intrinsic quality of the environment

itself. In this respect, Fletcher places himself in a cognitivist light. He describes how the environment can *give* the population ambiguous *information*, whereas I use a description in which it is the agent-environment interaction that *creates* information which can be ambiguous or not.

In another work done by Fletcher (Fletcher, 1996), the relationship between adaptability and environmental structure is investigated in an evolving artificial population of sensorimotor agents. There he argues that adaptability depends on the amount of detectable structural information in the environment along with the ambiguity and value of this information. In his words: "...i.e., whether the information accurately signals a difference that makes a difference." This is quite an intuitive notion. When the environment is too poor, it does not supply the population with enough information to be exploited, so adaptation will be difficult. On the other hand, when the environment becomes too complex, it could swamp the population with too much information, inhibiting adaptation. So it is hypothesized that adaptation will be maximal somewhere in between these two extremes. A very simple example of an environment that is too poor is an empty world. In that world there is a total absence of information, so there is nothing to adapt to. An example of an environment that is too complex is an environment in which the outcome of certain behavior changes randomly every now and then. This gives information, but the information is useless because it gives no direction to the adaptive process.

2.3 The Model

With the descriptions of behavior and environment in mind we can now make a basic model of behavior that can serve as a basis for investigating the questions posed in chapter 1:

1. Is there an optimal balance between *pro-active* and *reactive* behavior within artificial evolved behavior?
2. How is this balance related to the environment in which behavior was evolved?

The model of behavior used in the experiments is based on the distinction between reactive and pro-active behavior. Figure 2.3 shows an illustration of the model.

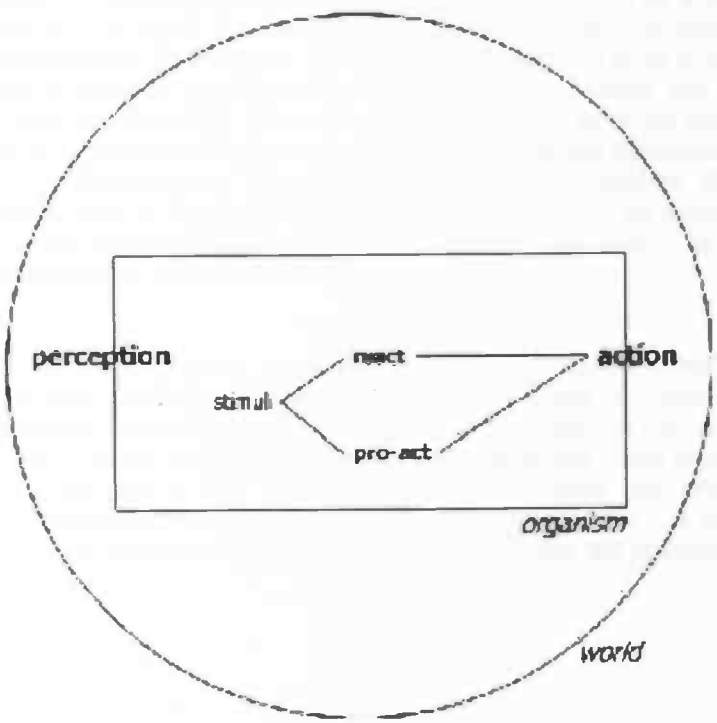


Fig. 2.3 Model of behavior

In the figure, the circle represents the *world* in which the *organism*, indicated by the box, finds itself. Behavior, as seen as interaction, is the creation of **perception** and **action**. Perception results in the creation of stimuli or sensory patterns. These stimuli are used as a basis for the two components of behavior, the **react** component and the **pro-act** component. These two components can be seen as two processes running in parallel. The combination of these two components creates a new action to be carried out by the agent. Because the two processes run in parallel there always exists a certain *balance* between the two. The words 'perception' and 'action' are deliberately spelled *across* the agent-environment boundary to emphasize the enactivist nature of the model. Perception is not purely a quality of the agent alone. It is also not an environmental feature. Rather, as we have seen, it is the *coupling* of agent and environment that result in sensory stimuli. The same can be said about action.

The organism is deliberately shown *inside* the world to emphasize the *embeddedness* of the organism. Also you can notice the absence of arrows. The idea behind this is to discard any hierarchy in the flow of information. One could say that perception leads to information. This information is then processed leading ultimately to an action. One could also say that an action creates a new perception through the use of information like infrared values. Perception, according to the enactivist paradigm, consists in perceptually guided action. This definition also emphasizes that it is *action* that is responsible for perception and can therefore not be regarded separately. Although these are details and have no real consequences for the control of the robot, they are important for the mindset in which the experiments took place. Further, the organism has two important characteristics: an internal energy level and age. The internal energy level is assumed to decline as time elapses, modelling basic maintenance of physiological functions such as body temperature and heart rhythm. Age is implemented as a simple incremental timer. When age reaches a certain value, the organism dies (of old age).

The model in figure 2.3 is a conceptual model of behavior. In chapter 3 a neural implementation of the model will be given. The two experiments use slightly different implementations of the basic model. The two variations of the model are discussed in the next chapter. In each of these models, the *source* of pro-active behavior is implemented by an internal source of activation, which will be called the **internal motivator**, relating to the motivational nature of pro-active behavior, described by Sloman (1987).

Earlier, we discussed pro-active behavior and asked the question in which situations it could be functional to become pro-active. One answer was when the environment is said to be 'poor'. Another way of putting this is by saying that *external dynamics* are low. This means that changes in sensory patterns created by perception are too poor to be used as a basis of functional reactive behavior. The idea that the level of pro-activity should increase at moments in which external dynamics are low is implemented. This is done by making the activation level of the 'internal motivator' dependent on the level of external dynamics. So when external dynamics are low, the activation value of the internal motivator will be high. This is a control that comes from the interaction between agent and environment and not purely from either one, because the external dynamics arise from the movements of the agent in the environment. More detail of the implementation of the model will follow in chapter 3. Furthermore, a low level of external dynamics imply an absence of (change in) stimuli. This can be seen as 'negative' stimuli. So the internal motivator reacts on negative stimuli, becoming active when the change in perceived stimuli is low.

In this chapter, a model of behavior has been outlined that is based on the enactivist paradigm. Two main processes within the model have been distinguished: *reactive* and *pro-active* influences. Pro-active behavior is behavior where at least part of the source of activity is internal. This internal source can be seen as internal motivation implying some preconception of the agent about the world. Also, the role of the environment within behavior has been described. In the next chapter two experiments will be presented which investigate some key aspects of pro-activity and behavior. The role of the environment is investigated by changing the environmental settings in each experiment.

3 Experimentation

In the previous chapter a model of behavior was given with the distinction between reactive and pro-active processes as its main characteristic. In this chapter the implementation of the model is given that will be used in the experiments outlined in section 3.5. In the experiments, a simulated robot has to learn to manage its resources in an efficient way. The robot has an internal energy level which declines over time. It can increase its energy level by eating food elements that are present in the environment. But besides food elements, the environment also carries poison elements, which will decrease the energy level of the robot when it consumes them. So the robot has to learn to avoid the poison elements and approach food elements. The more efficient this is done, the higher performance will be.

The chapter is divided into five parts. In the first part the term 'evolutionary robotics' is explained, which is an experimental technique used for investigating artificial behavior. The second part describes the agent, or more specifically, the controller of the agent: a feedforward neural network. Special attention is given to the 'motivator' node of the network, which implements the source of pro-active behavior. The third part deals with the world or environment in which the agent operates. The fourth part outlines the evolutionary algorithm, used as the technique for optimizing artificial behavior. The fifth and final part describes the actual experiments in which motivation, the balance between reactive and pro-active behavior, and the role of the environment are investigated.

3.1 Evolutionary Robotics

The design and development of robotic controllers is a complicated and delicate task. Many techniques exist, all with specific advantages. For many tasks that involve interaction with a complex world, a simple straightforward solution is not clear. Franklin (1995) stated: "If you are going to build artificial creatures with artificial minds, and you want them to survive and procreate, you will have to endow them with some way to produce novel behavior. Not all important contingencies can be predicted, on either an individual's or evolution's time scale". Because not all important contingencies can be predicted, many researchers of natural and artificial behavior use a technique called *evolutionary robotics*. In evolutionary robotics, a neural network is used as the control system for a robot, and an evolutionary algorithm is used for searching for the right network architecture. Different variations of this idea are possible. Different things can be evolved by the evolutionary algorithm. One could evolve the structure of the network. One could also evolve the connection weights within a specific network architecture. One could also do both. In the current experiments, behavior that is evolved is straightforward enough to deduce that the architecture used in the experiments should work given the right connection weights. Therefore, only the connection weights are evolved and not the structure of the network.

One reason for using the Evolutionary Robotics approach is the use of evolution as the main *engineer*. By making use of evolutionary techniques, solutions of control problems can be discovered that would not likely be found by a human engineer. Wooldridge (1995) acknowledged this fact when he recognized that behavior emerges from the interaction of different 'component behaviors' when placed into an environment. This suggested to him that the relationship between individual behavior, the environment, and overall behavior is not understandable. This makes it very hard for an engineer to design a control system for a certain task with enough certainty about the success of such a control system. And even if it were possible, chances are very high that the human solution would be much more elaborate than its evolutionary counterpart. Evolution can see things humans can not and will use everything that is usable, even it would not seem 'logical' to the human eye.

3.2 The Agent

The agent used in the experiments is a simulated robot, based on a real robot, called the *Khepera* (fig. 3.1). The Khepera was developed by K-Team SA in Lausanne, Switzerland

(http://www.k-team.com/index.html). The Khepera robot is cylindrical in shape and about 6 cm in diameter. It has eight IR (infra red) proximity sensors of which two are mounted on the back and six on the front. The Khepera can sense its surroundings via these IR sensors. Furthermore it has two accurate stepper motors attached to wheels, with which it can move around.

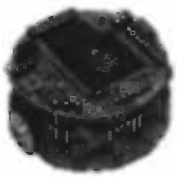


Fig. 3.1 The Khepera robot

A simulated version of the Khepera robot is used, which is shown in figure 3.2 In this figure, the eight different infra-red (IR) sensors are indexed in the same way that was used in the experiments.

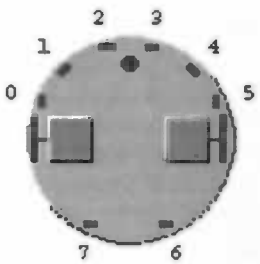


Fig. 3.2 Simulated robot

The controller of the robot (that is, the mechanism responsible for the behavior of the robot) takes as input the IR signals from the sensors and gives as output two speed-values of the two motors. In the experiments, the controller comes in the form of a neural network. The basic idea of neural networks will be assumed as common knowledge. (For more on neural networks, check out the bibliography.) The neural network is an implementation of the model that was given in 2.3. The internal source of activation, called the *internal motivator*, is one of the nodes in the network. This internal motivator is the implementation of the pro-active process in the behavior of the robot. The other nodes in the input layer of the network form the reactive process. This will be outlined in more detail in the next section.

3.2.1 The Neural Network

The neural network that is an implementation of the model outlined in chapter 2 is shown in figure 3.3.

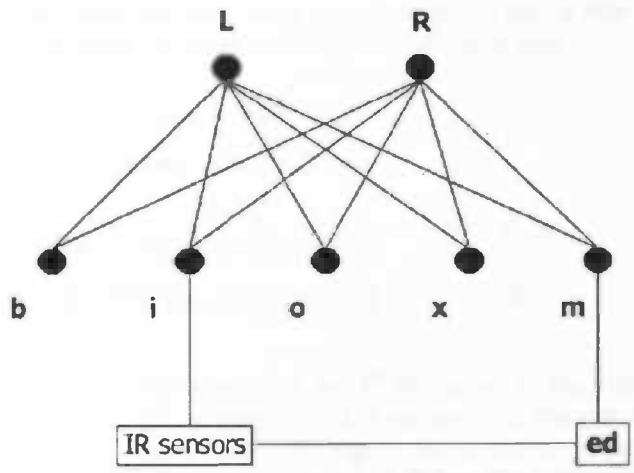


Fig. 3.3 The neural network

The output layer consists of two neurons, or nodes, one for each of the two motors. One node for the left motor, the **L**-node, and one for the right motor, the **R**-node. **IR sensors** represent the infra-red readings taken from the eight sensors of the robot. These values are given to one neuron in the input layer of the network and to **ed**, which stands for *external dynamics*. The external dynamics takes part in the calculation of the activation of the **m**-node, which will be described below. The input layer has five nodes. The **b**-node is a bias node. The **i**-node gets sensory information from the **IR sensors**. The **o**-node is the object node and carries *type* information about a perceived object. The **x**-node is a node that is necessary for dealing with input patterns that are not linearly seperable (XOR-problem). These four nodes can be seen as forming the reactive part of the network. The **m**-node is the internal motivator and forms the pro-active part of the network. The five nodes will be described in more detail below.

When we look at the network we can see that it forms an implementation of the model of behavior given in 2.3. To clarify this lets cut the network in two parts. The first part consists of the nodes **b** through **x** along with their connections and weights and the second part is the **m**-node with its two connections and weights. This is illustrated in figure 3.4. These two processes come together at the two motor outputs of the network, thus together forming the behavior of the robot.

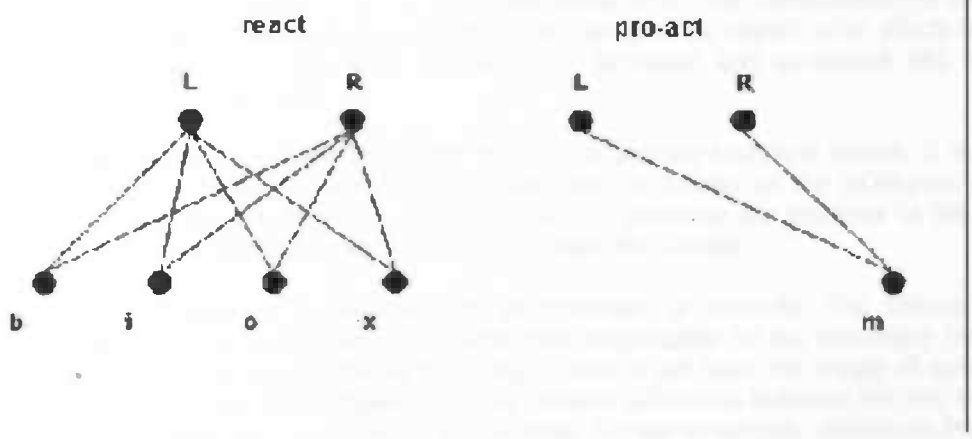


Fig. 3.4 The network split up in a reactive and pro-active part

In this figure, the same names were used for the reactive and pro-active processes as were used in the model of behavior given in figure 2.3.

The four nodes **i** through **m** will now get a more detailed description.

The **i**-node gets its activation from the eight IR-sensors of the robot. The eight IR-values are combined into one value *IR*. This value says something about how much obbjects the robot senses on either its right or left side. *IR* is calculated in the following way:

$$IR = \begin{cases} l + r & \text{when } r > l \\ -l - r & \text{when } l > r \end{cases} \qquad (3.1)$$

$$l = 2S(0) + 3S(1) + 4S(2) + S(7) \qquad (3.2)$$

$$r = 4S(3) + 3S(4) + 2S(5) + S(6) \qquad (3.3)$$

In this equation, $S(i)$ represent the i^{th} IR-sensor of the robot. The sensors are located on the robot as shown in figure 3.2. In equation (3.1) we see that the value of *IR* is made up of combining two values *l* and *r*, representing 'left' and 'right'. So a single value gives a notion of the amount of proximity. The weighting of the components of *IR* has the following reason. Objects that are perceived more head-on bring with them a higher risk of a collision, so they are to be avoided more drastically. So the activation of the i-node gives a notion of *how much is perceived* and *which side has the greatest contribution*. A few simple tests with the robot showed that this single value was sufficient as a basis for simple object-avoidance behavior.

In the evolutionary task, the robot has to distinguish between different types of objects. This is a classification task. A problem with classifying the objects that are present in the robot's environment is that the sensory patterns that result from interaction with the objects are too poor as a basis for classification. In other words, the visual input received is too poor for making a good representational separation of the different objects. This is known as the *aliasing problem*. One solution of this problem is using not only external (visual) information, but also information about the state and behavior of the robot itself. For instance, Pfeiffer and Scheier (1999) devised a method in which a robot classifies objects of different size by interacting with them. In their experiment, a robot learns to circle around an object when it encounters one. While driving around it the robot can then link its angular velocity with the identity of the object, since objects of different size will result in different angular velocities. They called this method *sensory motor coordination classification*. But using a method like that would include learning an extra task alltogether and the classification itself is not the focus of this thesis.

Therefore, a third node in the network gives type-information about the objects in the environment. This node is called the Object node, or o-node. Food-objects are classified as a +1 activation of the o-node and poison-objects as a -1 activation. If neither of both objects are present, the activation of the o-node is set to its default value of 0. The classification of the objects is done when the robot comes within a certain threshold radius of the object, after which type-information of the object is taken from the world image in the simulator and translated into the corresponding activation value of the o-node.

The reason for adding the fourth node is of a more technical nature. It does not add extra useful information about the world, but it solves what is known as the XOR-problem. This problem arises when only the i-node and o-node are used to generate the behavior of the robot. The XOR-problem is treated in the appendix. This node is called the **x**-node.

The fifth node is called the *internal motivator*, or **m**-node. The internal motivator is the implementation of the internal source of activation responsible for the pro-active behavior. Since, the pro-active behavior is only produced by a single node, it will have the shape of turning behavior. The angle of this turning behavior depends on the relative difference between the two connection weights between the **m**-node and resp. the **L**- and **R**-node. So the **m**-weights determine the *structure* of pro-active behavior.

Three cases are discerned in which this node will become active. The first case is the situation in which the environment is poor. Movement in such an environment does not lead to (large) changes in the perception of the agent. For instance when the agent is driving in an open space with no objects or when it bumps against a wall and stays there or drives parallel to it, the IR-values will not change

very much from moment to moment. By becoming pro-active the agent could pull itself out of such a behavioral impasse. The pro-active behavior adds an additional degree of freedom which can be functional in certain situations. And by becoming pro-active only under the condition that external dynamics are low, a situation can be avoided in which pro-active behavior prevents the agent to react adequately to objects. Because when objects are near, external dynamics will tend to be high, inhibiting pro-activity.

A second case in which the agent must become pro-active is the case in which its energy-level is low. In other words, the agent is hungry. The robot has an internal battery indicating the energy-level. When it is low this could mean that the current behavior is not functional, that is, does not lead to high energy-levels. By making the battery-level inversely proportional to the activity of the internal motivator the agent will become more pro-active at moments when its energy is low. Because of the special nature of the internal motivator, the calculation of its activation value is described in a separate section 3.2.2.

In 2.2.2 three cases were distinguished in which pro-activity may be useful. Two have already been addressed. The third case was the one in which the pro-active behavior could exploit some aspect of the agent-environment interaction that reactive behavior could not, thus adding extra functionality to overall behavior. This could occur when the *structure* of pro-active behavior can somehow be coupled to the *structure* in the environment. When this coupling results in behavior that is evolutionary functional or fit, pro-active behavior can be beneficial. But this remains to be seen from the actual agent-environment interaction and is not implemented forehand.

The activation values of the input nodes all are real numbers in the range [-1.0,1.0]. The excitation of an output node is made up of the weighted sum of activation values of all input nodes. The activation of the output nodes is then achieved by taking the sigmoid function of the excitation value.

$$a_j = \tanh(\sum_{i=0}^{n} w_{ij} I_i) \qquad (3.4)$$

In (3.4) a_j is the activation value of the j^{th} output node, w_{ij} is the connection weight between input node i and output node j, and I_i is the excitation value of input node i. **Tanh** is the hyperbolic tangent function, which has a *sigmoid* shape. This function normalizes its argument value to a value in the range of (-1,1).

3.2.2 The motivator

The activation of the internal motivator is determined by three things: an internal source of activation, the external dynamics and energy level of the organism. In the previous section two situations were described in which the internal motivator becomes active. One in which the environment is poor and another in which the robot is hungry. These two situations are implemented by making the activation level of the internal motivator dependent on the external dynamics and the energy level of the robot.

The activation of the internal motivator is calculated in the following way:

- a *target* value of the activation of the motivator is calculated by:

$$m_{target} = (b + \frac{1}{bat} + e^{-a(ed/w)^2}) \qquad (3.5)$$

in which b is a bias activation, *bat* is the current energy level of the robot (*bat* is short for battery), *ed* is the external dynamics and a and w are two parameters that determine the shape of the gaussian function.

- then, the *actual* activation value is determined by the id-parameter as:

$$m(t) = m(t-1) + id * (m_{target} - m(t-1)) \qquad (3.6)$$

where *m(t-1)* is the previous activation value of the motivator. The **id**-parameter (*id* stands for *internal dynamics*) controls how much the activation of the internal motivator is allowed to shift towards the taget value given its previous value.

The external dynamics **ed** is a measure of change in sensory information. It is the rate of change in the IR-readings from one point in time to another. This measure is calculated by taking the derivative of the IR-input. The internal dynamics, or **id**, reflects a sort of viscosity of the activation change of the **m**-node. The higher the internal dynamics, the more the activation is allowed to change from one moment to the next. When **id** is zero no activation will be formed in the **m**-node. So this parameter stands for a sort of quickness in response of the internal motivator.

Higher **id**-values mean fast adaptation of the internal motivator. This implies also a fast decay of any influence of past activation. So the internal motivator implements a simple form of memory. The faster it is allowed to decay, the less past activation values can 'linger' in the robots behavior. Low values of the **id**-parameter lead to slow decay of earlier activation values, meaning better memory. In neurobiology *memory* refers to the relatively enduring neural alterations induced by the interaction of an organism with its environment (Haykin, 1994). According to that defenition the internal motivator certainly has a form of memory.

Equation (3.5) shows that when external dynamics are low, the target value for the activation of the motivator is high. When the activation of the motivator increases, overall behavior becomes more pro-active. This could result in the robot abandoning the kind of behavior that has led it into stimulus poor areas. When this happens, the robot will come into more rich grounds and external dynamics will rise as the robot interacts with the increased level of stimuli. This will set a lower target value for the activation of the motivator, resulting in a decrease in pro-activity. So the internal motivator can take care of the agent until the environment takes over again. This causal loop within behavior incorporating the external dynamics and the motivator has a self-regulatory nature. It resembles a kind of feedback loop.

In chapter 2, the notion of structural coupling and feedback was introduces and illustrated by the example with the centrifugal governer. The loop diagram in figure 2.3, in which the interaction of the centrifugal governer with the overall system was shown, can be easily used as a basis for a similar kind of loop diagram for the internal motivator and the external dynamics **ed**. The result of this is shown in fig. 3.5:

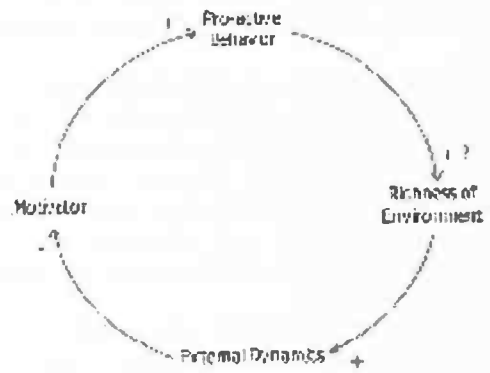


Fig. 3.5 Loop diagram of the relation between the motivator and the external dynamics

In figure 3.5, a question mark is shown next to the positive causal connection between the pro-active behavior and the richness of the environment. The reason for this is that the positive nature of the connection is hypothetical. It heavily depends on the complexity of the network and the environment. As we will see in 3.3, the different testing environments used in the experiments are not very rich or complex environments. This will results in the robot moving through empty space for the larger part of its life. Empty space will decrease the external dynamics and this will, as we have seen,

result in more pro-active behavior. Pro-active behavior then almost becomes an additional bias. This adds an additional degree of freedom to overall behavior. Whether this additional degree of freedom is functional depends on the shape of the pro-active behavior in relation to the structure of the environment.

The following section deals with two important charcteristics of the agent: its *battery*, representing the internal energy level, and *age*. How these parameters change over time and what information they carry will be described.

3.2.3 Battery

Since the evolutionary task of the agent consists of managing its resources, the energy level of the agent will carry important information about the success of behavior. As we will see in this section, the fitness information about the quality of behavior follows directly from the way the energy level of the agent changes over time.

The internal battery starts at the 'birth' of the robot at a maximum value. In every cycle in the simulation, so in every step of behavior, the energy-level drops, following a simple decay-function:

$$\text{battery} = batdecay * \text{battery} \quad \text{, with } 0 < batdecay < 1$$

A simple decay-function such as this one still has some useful properties. By the following example it will become clear that with the use of this simple decay-function, the energy-level at the end of the robots life carries information about the efficiency with which the robot has managed its resources.

Example

When an organism feeds itself, it is more effective to do this when it experiences 'hunger', that is, when its energy-level is low. If it would go look for food when it has just eaten, it would carry on looking for food all the time, missing maybe other important tasks like mating or pretecting its territory. Also, when the organism eats all its available food at once, the relative effect of eating one food unit will be smaller than when it spreads the consumption over time. This becomes even a more pressing matter when food is scarse or limited. So the way an organism manages its resources *over time* says something about the efficiency with which it takes care of its energy-level.

Now lets look at two different scenarios in which the robot eats three food-objects. In the first scenario the robot eats the three objects one after another in a very short period of time at a moment that its energy-level is still high. This means that it eats everything when it does not really feel hunger. In the second scenario the robot eats the three objects at moments when its energy-level is low, that is, when it does experience hunger. Two simulations with the robot were carried out which implement these two scenarios. One food-element carries an energy-level of 200. When the robot eats a food-element, the energy stored in the food-element is added to the energy level of the robot.

In figure 3.6, the energy-level is plotted against time. The dashed line shows the energy-level of the first scenario and the solid line of the second scenario. Because the decay-function subtracts the same percentage of its value every cycle, the absolute effect of the decay is higher when the energy-level is high. So the effect of the consumption of food at times of high energy-levels is much smaller than at times of low energy-level, because the gain in energy will be spent much faster when energy-level is high.

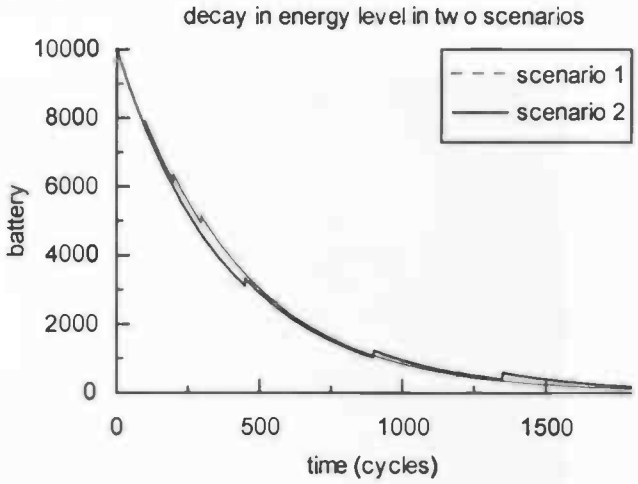


Fig. 3.6 Battery level when three food elements are consumed

You can see that the final energy-level at the end of the robots life is higher in scenario 2 (about 394 vs. 224 in scenario 1). This is because the gain in energy at lower energy levels 'last' longer so to speak. They have a bigger relative impact. So the energy-level at the end of the robots life carries information about the efficiency with which the robot manages its resources. Just this value would even be a useful fitness-function for the evolutionary algorithm. Actually, the energy level at the end of life is taken as a multiplication factor in the fitness function, which magnifies the effect.

The way the energy level at the end of life reflects quality of resource management introduces the importance of *timing*. The plots above show that a good timing of the consumption of food increases its effect dramatically. Even more, the energy level of an agent that eats one food-element *just* prior to dying of old age can end up with a higher energy level than an agent that has consumed 5 elements more early in life. Because *timing* becomes important, so does *speed*. When an agent moves faster it can come across more elements theoretically. But chances are high it will encounter them more early in life than when it would move more slowly. So there seems to exist an unavoidable trade-off between the chance of encountering an element and the relative effect it has on energy level at death.

Although the agent does not have access to its energy-level directly, it can sense it indirectly through the increased level of pro-activity, the activation of the internal motivator. So the robot can not feel its energy-level, but it can feel a heightened urge for self-motivation because it is 'hungry'.

3.3 The World

In the experiments, the world in which the robot operates has the form of a 1m • 1m enclosed square space. In that space 3 kinds of objects are to be found: food elements, poison elements and wall-elements. The food- and poison elements are scattered through space, while the wall-elements make up the enclosure of the space. The food- and poison elements are small round objects that are similar in shape and size. In figure 3.7 the yellow (or lighter) objects represent food and the green (or darker) objects represent poison. These objects have a fixed place in space and will vanish when the robot collides with them. That means that the robot has 'eaten' the element. When the individual (robot) dies the eaten elements are put back on their original location. Before the robot collides with an object it can sense them through its IR-sensors. Also, type information of the object is given to the o-node in the network. So the robot has a certain time-span in which it can react on the object.

To investigate the role the environment plays in the evolution of autonomous behavior, three different environments, or *worlds*, are introduced. The first world as the default world with 6 food elements and 6 poison elements scattered over space. This world is called *rich_world*, indicating a richness of resources. This world is depicted in fig. 3.7.

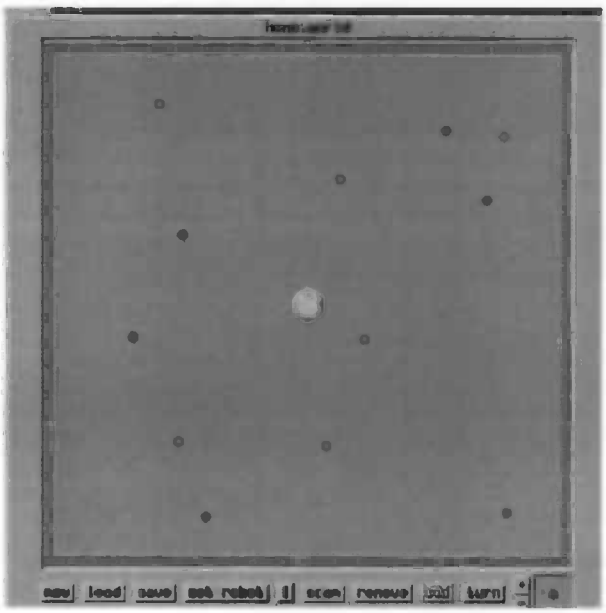


Fig. 3.7 *rich_world*

Special attention will be given to the relation between the motivator and the character of the world in which the robot finds itself. One question would be if certain worlds will offer more functionality to the motivator than others. In other words, what kind of environments have a quality that can be actively exploited by the motivator. We know that the motivator represents a certain kind of 'belief' about the environment. The question then is if and how much correspondence there is between the character of the environment and the belief about the environment represented by the pro-active part of the network. As the behavior resulting from the motivator is a kind of turning behavior, we could then look at the average turning angle produced by the motivator. Then we could see if the structure of the environment favors certain turning angles over others. If that is so, the motivator can add extra functionality.

Lets take a look at the two other worlds: *left_world* and *poor_world*.

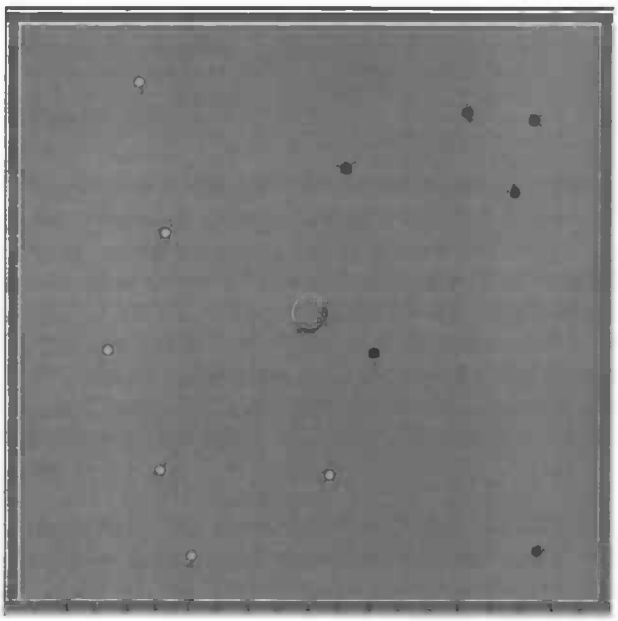


Fig. 3.8 *left_world*

In *left_world*, the environment displays more structure, as the elements are clustered. But the question remains if the structure is one that can be used. In other words, will the agents be able to use the structure of the environment so that the turning behavior produced by the internal motivator is beneficial?

A third world is one in which the number of elements is divided by two. This world has much less resources and so it is called *poor-world*.

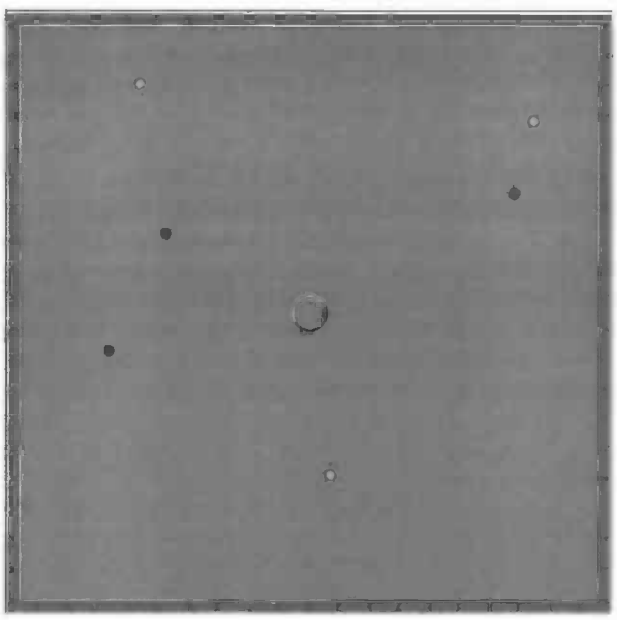


Fig. 3.9 *poor_world*

What effects will this have on behavior? Will individuals evolve higher default driving speed in order to cover more area? In a poorer environment, the notion of *timing* would be more relevant than

in an environment which offers higher chances of encountering objects, such as *rich_world*. We will look at these questions when we come to the experiments.

3.3.1 Structure

In chapter 2, *structure* was introduced as the amount of information or regularity that is present in the environment. A difference was made between *environmental structure* and *usable structure*. The former says something about the objective nature of the environment, like the distribution of elements, or the size and shape of the environment. The latter says something about the level in which the environmental structure can be *used* by the agent. Also in chapter 2, Fletcher's notions of *value* and *ambiguity* were given. If an agent in the experiment is set in an environment that offers high levels of gain to the use of an internal motivator in the interaction with that environment, and this is done in an unambiguous way, that environment is said to have a high level of *usable* structure. If no *structural coupling* between the agent and the environmental structure is possible, this environmental structure is not usable.

Lets have a look at some different 'structural settings' which offer different levels of *usable structure*. The coupling of environment and agent consists of two components. So the structural settings used in the experiments also have two components. The first one is the environment component, or *world* component. As different *worlds* are used as an evolutionary background, different *worlds* offer different amounts of structure. For instance, *left_world* clearly gives more structure than *rich_world*. It is conceivable that the agent will learn to make use of this structure by limiting its living space to the left part of the environment.

The other component of the structural settings is the way the robot is reset every time it has died. More particularly, the *angle* with which the robot is put into its *world*. In the experiments, 3 different alternatives are used: *changed_angle, passed_angle, and fixed_angle*. With *changed_angle*, the starting angle of the robot is changed over 360 degrees over the number of lives. With this setting, the robot will approach the environmental structure from another angle each life. This will limit chances for the robot to make use of the present environmental structure, since it is approached from a different angle each life.

In the *passed_angle* setting, the angle of the robot in the environment at the moment of its death is *passed on* to its next life. When behavior is evolved, an individual could learn to make use of this principle. When a certain functional behavior has more or less the same starting angle as 'death' angle, behavior can go into a functional loop. In that case, the chance that the agent will encounter the same subset of stimuli (Nolfi, 2000) each life will grow dramatically. When this subset leads to high fitness, the agent has successfuly taken advantage of the *passed_angle* setting. This behavior can be represented as an attractor in behavior space.

The third alternative, *fixed_angle*, for placing the robot is to let the robot start of with exactly the same angle each life. This third mode gives the robot the most opportunity to optimize its behavior according to the structure that is present in the world. It optimizes the amount of *usable structure*. In chapter 4 we will see this is exactly what happens.

In figure 3.10 the three different *angle* settings are depicted.

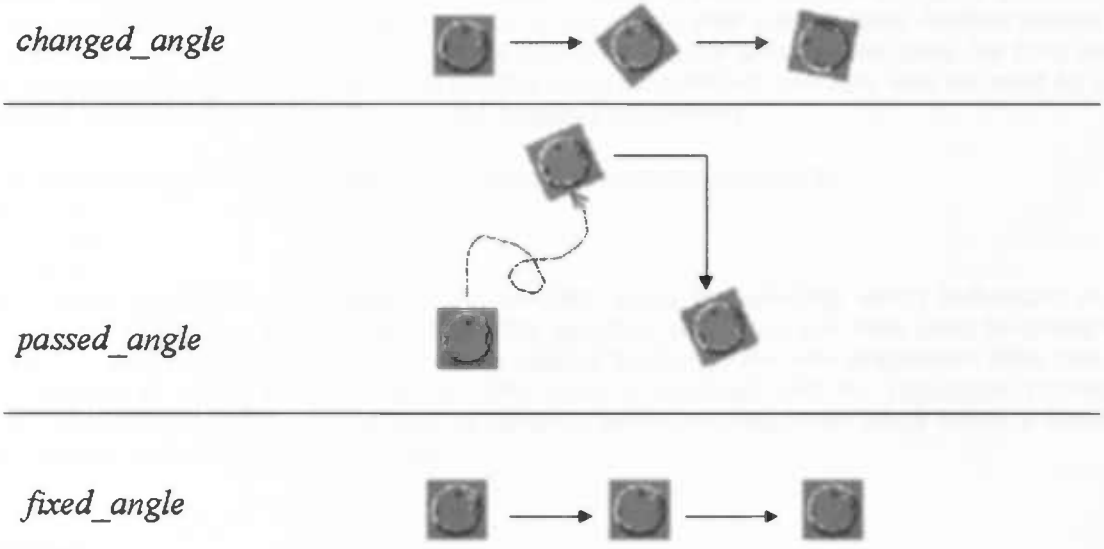


Fig. 3.10 Three different angle settings

The two components of the structural setting, *world* and *angle*, determine the level of *usable structure*, and they can take on the following values:

World	*Angle*
poor	*changed*
rich	*passed*
left	*fixed*

What influence different levels of *usable structure* have on evolved behavior will become apparent when we discuss the results of the experiments in chapter 4.

3.4 The Evolutionary Algorithm

"From the war of nature, from famine and death, the most exalted object which we are capable of conceiving, namely, the production of the higher animals, directly follows."
Charles Darwin, *The Origin of Species* (1959)

The ideas of Darwin about the evolution of organisms is what inspired researchers to use these ideas as a technical engineering technique called the *evolutionary algorithm*. What the evolutionary algorithm (EA) does in the current case is search the n-dimensional space, where n is the number of connection weights in the network. Each point in this space can be seen as an instance of the robot. So it searches the space of instances of robots. Each instance implies certain behavior. This behavior has a certain chance of being successful. The search is guided by what is called an 'evaluation' function or 'fitness' function. This function specifies functional behavior. When an instance of the robot is tested in the simulated world, the fitness function will specify how well the performance of that individual robot is. So taken the space of possible behavior and the fitness function we could construct a fitness landscape. This will be a chance landscape since the behavior is created in a non-deterministic way.

The advantage of this approach is that the system does not have to be hand-designed to perform a certain task, but the task will emerge gradually during the evolution process. The algorithm

exploits what information it has and uses it to construct the appropriate control system. This makes the evolutionary algorithm very robust because it does not need to understand the parameters being optimized. Many other search algorithms for instance depend on exact information like derivatives or other mathmatical processing of the parameters for optimizing their current state. Another reason that makes EA's robust is that it uses probabilistic transition rules, not deterministic rules. So EA's search many different points in decision space in parallel using randomized operators with no need for extra information other than the direct codings of the system's parameters.

The evolutionary algorithm in the experiments uses three genetic operators:
- selection
- crossover
- mutation

In each cycle of evolution, or *generation*, *selection* starts by selecting which individuals in the population are allowed to produce offspring. The selected individuals are then used to create new offspring by using *crossover*. Then *mutation* is applied to change the new population. After that, the new population is tested on performance. This cycle is repeated until the population *converges*. Convergence occurs when the total level of variance within the population drops below a threshold value. Lets turn to the process of *selection*.

Selection

Selection is the process with which individuals are chosen for reproduction. This process is like the survival of the fittest we see in nature. Only the strong survive. This implies a measure of 'strongness', or fitness. Fitness is determined by an objective fitness-function. The fitness-function is perhaps the most important part of an evolutionary algorithm for it guides the search process. It tells the algorithm what areas of the searchspace are of interest. The amount of fitness that an individual receives will determine the probability it will contribute offspring to the next generation.

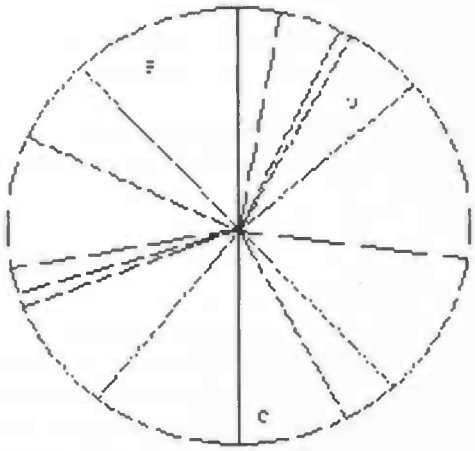


Fig. 3.11 Roulette wheel

The way in which selection is done based on individual fitness is by implementing the so called 'roulette wheel' method. In this method each slot of a roulette wheel (e.g. a in figure 3.11) corresponds to an individual in the population and the size of the slot is proportional to the fitness of the individual. Actually the size is proportional to the rank of the individual and its rank in the population is determined by its fitness. Offspring is then created by spinning the wheel and copying the chromosome corresponding to the slot where the ball stops. An individual chromosome x_i has a fitness f_i. The rank of the individual r_i is then determined by f_i. The individual with the highest fitness is highest in rank. This scales the fitness values prior to selection after which the probability of making offspring is proportional to rank, not fitness. This prevents the occurance of two extreme cases:
- when all individuals have similar fitness all individuals will have the same probability of making offspring so that evolution will amount to random search,
- when 1 individual has much higher fitness than the rest, that individual will soon dominate the entire population and cause premature convergence.

When a new population is created, 75 percent of the worst individuals in the old population is replaced by 75 percent of the best children in the new population. Not 100 percent of the old population is replaced for preservation of diversity in the population – by keeping some of the old population, chances are higher that some characteristics that may become useful in the future but were not useful in the current situation will remain within the population.

In the experiments an individual has the form of a string of real numbers that lie in the region of [-1.0,1.0]. Each value stands for a certain weight of a connection in the neural network. So a typical individual in a population will look like:
[1.000,0.784,-0.321,-0.005,0.992,....,0.562].

When producing a new population, the children that come out of the old population are created by crossing over two parents.

Cross-over

Crossover is a procedure by which a pair of chromosomes swap genetic material. This occurs with a certain probability, the crossover-probability p_C. Various types of crossover exist, e.g.:
1-point crossover → 1 point along the chromosome is randomly chosen and the genetic information from this location on is swapped between the two chromosomes.
multi-point crossover → more than 1 point is chosen and the same procedure as with 1-point crossover is executed at each chosen location.
uniform crossover → for each point along the chromosome the crossover probability determines if genetic information at that location is swapped between the two chromosomes.

In the experiments 1-point crossover was tested against uniform crossover and 1-point crossover gave better results. Given parent strings x and y of length l, first a random number decides if crossover occurs given probability p_C. If crossover will occur, another random number determines the location k along the chromosome. Then 1-point crossover will look like:

$$(x_1 \ldots x_k, x_{k+1} \ldots x_l) \Rightarrow (x_1 \ldots x_k, y_{k+1} \ldots y_l)$$
$$(y_1 \ldots y_k, y_{k+1} \ldots y_l) \Rightarrow (y_1 \ldots y_k, x_{k+1} \ldots x_l)$$

Studies have been made that focus specifically on the way crossover disrupts and constructs so-called *schemata*. More on this topic can be read in the appendix.

Mutation

Mutation consists of changing the value of a location in a chromosome by adding or subtracting a random value. Mutation occurs with a mutation probability p_M. For each newly created individual each location has probability p_M of being mutated. The value being added or subtracted lies within [-0.5,0.5]. Mutation is an important operator because the algorithm works with real valued codings. This means that to be able to gain access to *all* of the solution space, all values between [-1.0,1.0] must be possible. With a binary coding, crossover can do this provided that for each location there is at least one individual within the population that has either a 1 or 0 as a value for that location (in other words, all possible information is present in the population).

The mutation operator determines for a great part the balance between the exploitative and explorative nature of the algorithm. If no mutation occurs, no change in values will happen, so search will have a more exploitative character. When a lot of mutation occurs, search becomes more explorative, since the values at most locations in the chromosome undergo a lot of change. It may be intuitive that one looks for a balance between these two effects.

Fitness

"Either dance well or quit the ballroom."
-- Greek proverb --

The way in which the behavior is evolved depends upon the use of the *fitness function* in the evolutionary algorithm. The behavior that is evolved will be an optimization of the fitness function under the constraints of the environment. When the environment allows, the individuals in the population will, generation after generation, seek out ways to satisfy the selective pressures given by the fitness function. If it is resource management we want, we will need a measure of the quality of resource management of individual robots. This measure will be part of the fitness function so that the behavior being evolved will actually *be* the resource management we want. As was stated in 3.2.3, the internal energy level, or battery level of the robot is an important part of this measure.

Good resource management is behavior in which food- and poison-objects are classified and consumption of the right objects is spread over time. What would be a suitable fitness function? One could count the number of food-objects that the robot eats and use that number as a function of fitness. Then the evolutionary algorithm will almost certainly evolve behavior in which the robot is attracted to food-objects. But what about the poison-objects? One individual could eat 2 food-objects and 0 poison-objects while another individual could eat 3 food-objects but also 4 poison-objects and be the fitter of the two. That would not produce very good resource-management behavior, since the consumption of poison does not imply a very good classification of objects. So the number of poison-objects an individual robot would eat could be used as a negative factor in the fitness-function. So the fitness-function could have the following form:

$$F = food - poison$$

where *food* is the number of food-objects eaten and *poison* the number of poison-objects.

However, the fitness-function used in the experiments has the following form:

$$F = battery * \left(\frac{(food+1)}{(\gamma * poison+1)}\right) + \frac{1}{wall} \qquad (3.7)$$

where *battery* is the energy-level of the robot the moment he dies and *wall* is a number that says something about the number of times the robot has run into a wall. The γ (where γ = [1,2]) is a parameter that indicates that poison has a bigger influence on fitness than food. The reason of this is that it is more important to avoid poison than it is to approach food. (The addition of 1 to the number of poison- or food elements prevents any danger of multiplication with 0.)

The addition of *wall* is purely a way of speeding up the evolution process. If the robot minimizes the times it spends time against a wall, it can use this time driving around in the space where the food- and poison-objects are. The more it comes into contact with these objects, the more its architecture can be tested for its resource-management qualities. And this means more information for the evolutionary algorithm to use. So the wall factor encourages wall-avoidance behavior. The influence of the *wall* component in the fitness function will inpractice be limited to the first few generations. When performance is increased and the robot manages a battery level greater than zero at the end of its life, the influence of the wall-component will be greatly diminished, rendering it irrelevant for further progress.

The reason that the negative influences on the fitness take the form of a fraction and not for instance the form of a subtraction, is that the selection method works on the assumption that fitness always has a positive value. If subtraction was used, the fitness value could come out negative, having disruptive effect on selection.

One can see the importance of choosing the right fitness function, because in the evolution process of populations of artificial organisms, individual organisms generate offspring as a function of the degree that they satisfy the fitness criterion. The wrong fitness function lets the wrong individuals reproduce, producing wrong behavior. An important question is how much the fitness function pre-defines the task. By what degree does the fitness function imply specific behavior? Well, given the structure of the control system, the way it interacts with the environment and the environment itself, it can imply a lot. Taken the fitness function alone, it does not imply as much. Even if one has information about structure, communication methods and environment, the evolutionary algorithm can still exploit behavior that is sub-optimal but still functional.

For instance, the neural weights encoded in a certain individual could result in the robot driving backwards (this is the result of negative valued bias weights). Although this driving direction could lead

to behavior in which the robot correctly classifies food and poison, the detail in perception is much lower on the backside of the robot than on the front. So although backward driving individuals can evolve functional behavior a large gain can be made by adopting positive valued bias weights. This would result in a forward bias driving direction and thus in a larger detail in perception. And a larger richness in perception can lead to a larger richness in behavior, since perception drives behavior.

In the following section, the experiments are described, making use of the neural network and evolutionary algorithm outlined above.

3.5 Experiments

The experiments were done by evolving nine populations of 50 individual robots. The nine populations were created using a random generator (a random generator is a function that gives a random number within a specified value domain). Each robot has nine lives, so performance is averaged over nine runs (note: each robot in *each* population has nine lives. So one *generation* of the evolutionary algorithm has 50x9x9 = 4050 runs). At the end of evolutionary process (the population has converged or 100 generations have been executed) the performance of evolved behavior is investigated. Performance is then related to the two main questions, already given in chapter 1:

- Is the balance between the level of pro-activity and the level of reactivity dependent on the nature of the environmental task?
- In what way is the level of pro-activity dependent upon the environment in which overall behavior is evolved?

The balance between pro-active and reactive behavior is investigated in the first experiment in 3.5.1. In the second experiment in 3.5.2 the balance between pro-active and reactive behavior is put under the control of the robot itself. The second question about the role of the environment is addressed in both experiments. In chapter 4 results are given of these experiments.

3.5.1 The Balance

By creating a trade-off between the reactive and pro-active part of the behavior forming process we can investigate the influence of the reactive/pro-active balance on the success of the evolved behavior. This is done by evolving populations with different balance values and looking at the quality of the evolved behavior. The parameter that controls the balance between the amount of influence from the reactive and pro-active parts of the controller is called the *dynamics-parameter* or **d**-parameter in short. So this parameter will define how much of the behavior of the agent will be reactive and how much it will be pro-active. If we call **R** the reactive part of behavior (react in figure 3.4) and **P** the pro-active part (proact in figure 3.4), the resulting behavior of the agent (B) can be defined as:

$$B = d*P + (1-d)*R \qquad (3.8)$$

where $0 < d < 1$. In this formula we can see a trade-off between the reactive and pro-active influences on behavior. The processes in an organism that form behavior have a limit to the amount of information that can be taken into account. The more one influence demands its share in the behavior forming process, the less room will be left for additional influences.

The main purpose of the experiment is based on the hypothesis that the optimal balance between reactive and pro-active behavior of an autonomous agent depends on the nature of the environmental task. After the evolution process, the individuals are tested for the quality of their behavior and these values are related to the values of the **d**-parameter. In this way we should get a clear picture of the influence of the **d**-parameter. After that an analysis of the resulting behavior is given, as an attempt to account for the results.

The model of behavior now changes to the one shown in figure 3.12:

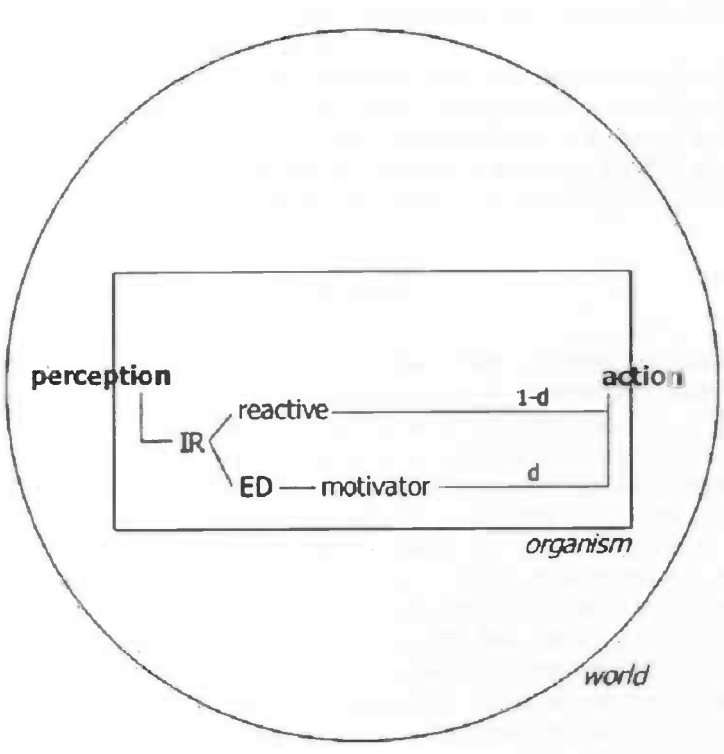


Fig.3.12 Model of behavior in experiment 1

This model is about the same as the conceptual model described in figure 2.3. **IR** represents the infra-red readings taken from the IR-sensors. **ED** stands for external dynamics, which is the measure for the amount of change in the IR-readings. **reactive** is the reactive part of the controller (see 3.2) and **motivator** is the internal motivator in the neural network, implementing the source of pro-active behavior. The trade-off is shown by the **d** and **1-d** parameters in the model.

For each value of the **d**-parameter 9 different populations are evolved. Each population carries 50 individuals. The populations are created using a random generator. Each individual is 'downloaded' onto the robot and is allowed to 'live' for 1500 cycles (MAX_AGE) after which its fitness is calculated. Its initial battery value (MAX_BAT) is 10000. The robot is reset after each life and starts at a different starting angle according to the *angle* setting that was used. This is repeated 9 times and the average fitness is calculated.

The population converges when the sum of the variances of the weights within the population drops below a certain threshold value. For each population the mean and best fitness values are recorded after each generation.

In this experiment, four different environmental settings were used. They are:

	world	angle
1	*rich*	*changed*
2	*poor*	*changed*
3	*left*	*changed*
4	*rich*	*passed*

For each setting, the experiment is repeated for eleven values of the **d**-parameter: {0.0, 0.1, 0.2, ..., 1.0}.

In chapter 2 'self-selection of stimuli' was explained. The idea behind this was that as agents are being evolved they learn to behave in such a way that they only encounter a sub-class of stimuli on which they know how to react. It almost seems as if they *select* the kind of stimuli they want. Actually it is a natural process within evolution. Individuals that interact with their environment in such a way that increases the chance of encountering 'functional' stimuli will perform better over all.

'Functional' stimuli mean stimuli which produce functional behavior. As these individuals perform better over all, they will start dominating the population with the result of having a population of seemingly 'smart' stimuli selectors.

Since the internal motivation can add an additional degree of freedom, agents that use their internal motivation should be better equipped in selecting subclasses of stimuli that lead to better performance. On the other hand, since there is a trade-off between reactive and pro-active behavior, the exploitation of the additional degree of freedom inhibits exploitation of effective reaction to stimuli. And without adequate reaction to stimuli, it is not likely that behavior will be very well adapted.

3.5.3 The Speed of Balance

In this experiment the robot itself controls the balance between reactive and pro-active behavior in more or less the same way as it controls motivator activation, that is, based on the level of external dynamics. So the balance value between reactive and pro-active behavior shifts from static to dynamic. As this dynamic balance is capable of adapting to perceptual change, it is to be expected that performance will increase with respect to the previous experiment.

In this experiment the robot can actively shift from more reactive behavior towards more pro-active behavior and vice versa, given the level of external dynamics. In the first experiment the dynamics parameter was fixed, so **d** = const. Now **d** *itself* is dynamic and becomes **d** = f(ed). First, a target value for **bal**, target_bal, is calculated. (In order to prevent confusion with the first experiment, the new dynamic **d**-parameter is called **bal**, which stands for 'balance-parameter'.) Then the new value for **bal** takes on a value somewhere between the taget value and the previous value, prev_bal. The main difference with the activation value of the motivator lies in the self-control of the balance parameter. As we will see, this simple mechanism gives a desirable effect.

$$\text{target\_bal} = e^{-a\left(\frac{ed}{w}\right)^2} \qquad (3.9)$$

$$\mathbf{bal} = \text{prev\_bal} + \mathbf{bal} * (\text{target\_bal} - \text{prev\_bal}) \qquad (3.10)$$

$$\text{prev\_bal} = \mathbf{bal} \qquad (3.11)$$

In (3.9) *ed* = external dynamics and *a*, *w* are constants. Target_bal is the target value for the **bal** parameter and prev_bal is the previous value of the **bal** parameter. The target value for **bal** takes the level of external dynamics and calculates a gaussian value. The actual value that **bal** will take depends on the value of **bal** itself. So the balance parameter also functions as its own dynamics parameter! When **bal** = 1, **bal** will always take on its target value. The lower **bal** gets, the less it will be allowed to divert from its previous value. Obviously, it is important that **bal** never acquires the value of zero, since that would keep the parameter on zero. This will never happen though, because the parameter can only *approach* zero. After that **bal** defines the balance between the reactive and pro-active influences of the network on the motor behavior by:

$$\text{Motor} = (1 - \mathbf{bal}) * \text{react} + \mathbf{bal} * \text{pro-act}$$

When **bal** = 1, behavior will be ruled by pro-active influences, which is the motivator in the network. When **bal** = 0, behavior will be purely reactive, driven by the other four nodes in the network.

So the robot does not just change the value of activation of its internal motivator. It also does not just change between reactive and pro-active influences on behavior. Now it changes the balance between reactive and pro-active behavior and the speed with which this is done. To be more precise, it is the *interaction* between the robot and the environment that controls the balance. The robot uses the external dynamics to calculate the balance value, but the external dynamics is a result of the interaction between robot and environment.

The model of behavior used in this second experiment is shown in figure 3.13.

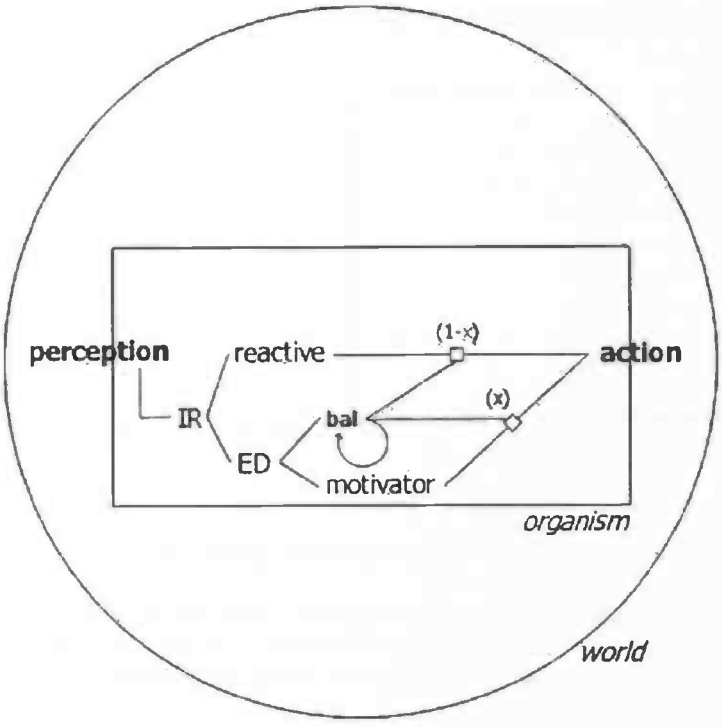


Fig. 3.13 Model of behavior in experiment 2

In this model, **bal** controls the balance between reactive and pro-active behavior and itself. (1-x) and (x) represent the trade-off that is created by the **bal**-parameter.

Why the self-controlling nature of **bal** gives a desirable effect is illustrated in the next section.

'React fast, think slow'

"Act quickly, think slowly."
-- Greek proverb --

The self-control of its own dynamics of **bal** results in qualitative difference between the transition from reactive to pro-active behavior and the transition from pro-active to reactive behavior. This is best shown by an example.

Lets say the robot begins its 'life' in a part of the environment with almost no diversity. This means that moving through such an environment results in low external dynamics so that the balance of behavior lies towards pro-active. **Ed** is low, so **bal** is high, which sets the balance of behavior towards pro-active. The robot will move until for instance it comes to a wall. When it approaches the wall, its infra-red readings will change, giving rise to a higher value of the external dynamics, **ed**. This will set a lower target value for **bal**. Since the current value of **bal** is high, the amount of change allowed towards its new target value is large. So **bal** drops and behavior shifts towards the reactive regime. Behavior is then based more on the sensor readings, making it more reactive. This could cause the robot to steer away from the wall. The robot then drives away from the wall into more 'empty' space. This causes the infra-red readings to drop and and stay low so that **ed** drops accordingly. The lower **ed** sets a higher target value for **bal**. But since **bal** is low, change in its value will be slow. So this time behavior changes more slowly and it will take more time to shift back to the pro-active regime. In figure 3.14 you can see the change of **bal** as a reaction to changes in external dynamics.

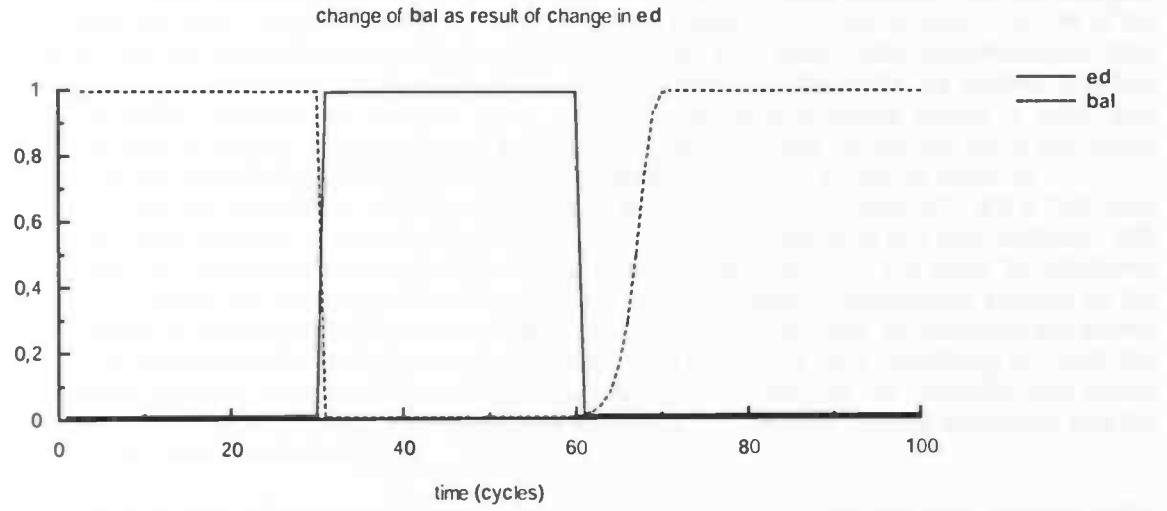


Fig. 3.14 Change of **bal** as a result of changing **ed**

This shows how a simple feedback mechanism can result in a qualitative difference between the two directions of a parameter-shift. The behavior will shift fast from pro-active to reactive but will shift more slowly back to pro-active. So it's 'react quick', but 'think slow'.

The experiment is divided into three parts. In the first part, two sets of populations are evolved. The first set is evolved with **bal** initialized at 0. This means that behavior will be 100% reactive and **bal** is not allowed to change. The second set is evolved with an initial **bal** of 1. In that case **bal**>0 and the balance between reactive and pro-active behavior is allowed to change dynamically. Both sets of populations are evolved in *rich_world* and with the *changed_angle* setting. The hypothesis is that fitness of individuals with a dynamic balance between reactive and pro-active behavior will be higher than that of individuals that are purely reactive. This is because the dynamic balance allows pro-active behavior, which can add an additional degree of freedom to overall behavior.

In the second part of the experiment the amount of influence the balance-parameter **bal** has on behavior is controlled by a damping parameter **bald**, standing for **bal**ance **d**amping. After a new value for the **bal** parameter is calculated, it is multiplied by **bald**.

bal → **bald***bal

The **bald** parameter limits the range of **bal**. When **bald**=1.0, no damping occurs and **bal** can do its work. Such a run would be the same as in the first part of the experiment. This parameter is given to the evolutionary algorithm. This allows evolution itself to determine the importance of the dynamic balance.

With this evolving damping factor, sets of populations are evolved with four different world and angle settings.

	world	angle
1	*rich*	*changed*
2	*left*	*changed*
3	*left*	*passed*
4	*left*	*fixed*

The evolved **bald**-parameter is important, because it gives an indication of how important pro-active behavior is. A high value of **bald** at the end of the evolutionary process suggests a positive contribution of the pro-active behavior. We will look at the relation between the values of **bald** and the fitness of various populations at the end of the evolutionary process, together with the different environmental settings in which evolution took place.

The first main research question of the thesis is about the balance between reactive and pro-active behavior and how it depends on the task. The second research question is about the role of the environment. We can combine these two questions into a new one: *under what circumstances does pro-active behavior add to the functionality of overall behavior?* Hypothetically, as usable structure increases, pro-active behavior will become more functional, resulting in higher values of bald. And fitness values will be higher in the case of increased usable structure, since the extra pro-active behavior has more functionality and adds an additional degree of freedom to overall behavior.

What could be expected if we allowed evolution to also control the trade-off? Will it first favor agents that are more reactive so that reaction to stimuli will be poison-avoiding and wall-avoiding? Will it start exploiting pro-active behavior only later when reactive behavior does not allow for additional performance increase? Or will it start favoring agents that show added effectiveness caused by the additional degree of freedom that internal motivation offers? As we shall see, as evolution will exploit everything that leads to performance increase, most gain will first come from classifying correctly the food and poison elements and changing behavior accordingly. Only after that will evolution take further gain from the performance caused by the additional degree of freedom the internal motivation and the corresponding pro-active behavior offers.

In the third part of the experiment evolved individuals from the second part, evolved under three different settings, are transferred onto a new world, called *right_world*. This world has all the food elements on the east (right) side of the environment. The individuals are evolved in this world with the *changed_angle* setting. This part of the experiment investigates how well individuals evolved under different (structural) circumstances can adapt to new environmental situations. Another way to put this is how well different individuals can cope with novelty. It can also give an answer to how much individuals come to rely on environmental structure as a basis for functional behavior.

The individuals are taken from the converged populations of part two of the experiment that were evolved in *left_world*. The transfer can be illustrated by:

left_world *left_world* *left_world*	*changed_angle* *passed_angle* *fixed_angle*	⇒	*right_world*	*changed_angle*

It can be hypothesized that individuals evolved under circumstances with the least amount of *usable* structure have come to rely the least on environmental structure and will be the best able to adapt to the new world.

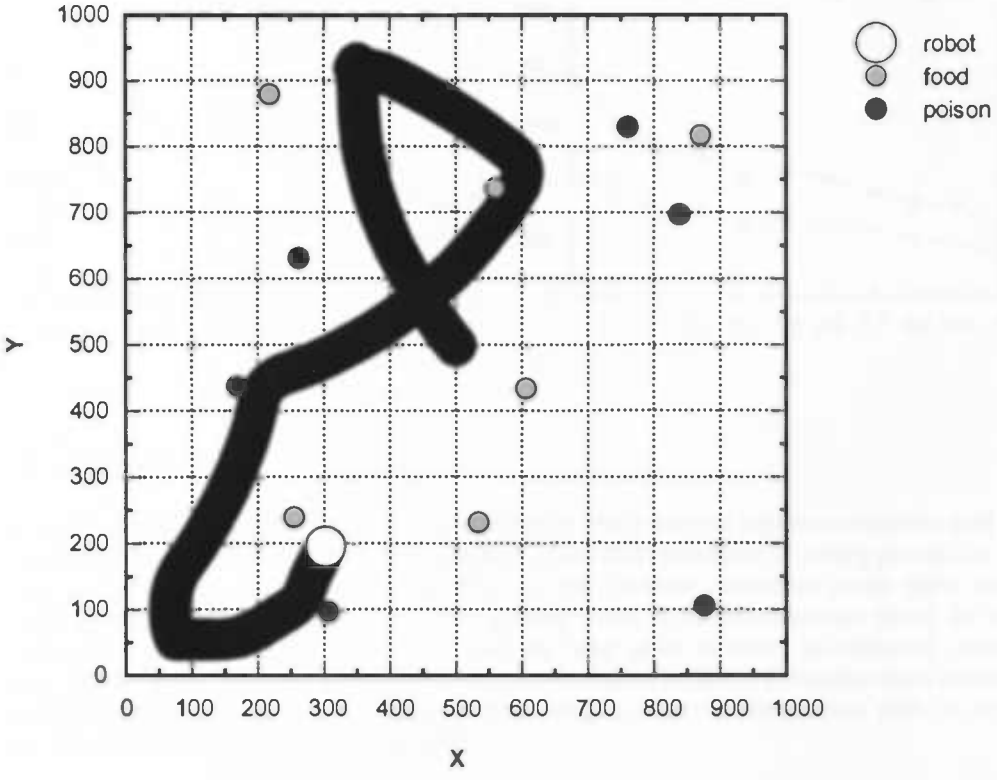
Now the main question will be if the motivator offers additional functionality for the evolutionary algorithm to exploit. We will see later on that it very much depends on the environment in which evolution takes place.

4 Results

In this chapter the results are given of the experiments as outlined in chapter 3.

4.1 Experiment 1: The balance

First, lets look at *what* kind of behavior is being evolved. The fitness function of the evolutionary algorithm guides evolutionary search towards behavior in which food elements are approached and poison elements are avoided. To illustrate this a plot of a typical path produced by an



evolved individual is plotted in figure 4.1.

Fig. 4.1 Path of evolved behavior

In the plot, the starting position is indicated by the red arrow. This plot clearly shows the agent has learned to avoid poison-elements and approach food-elements.

As stated in chapter 3, the main question in the first experiment was whether a certain balance between reactive and pro-active behavior optimizes behavior. This balance is investigated by implementing a trade-off between reactive and pro-active behavior and evolving populations with different levels of reactive resp. pro-active behavior. Below are the fitness plots for the four different settings that were given In chapter 3.

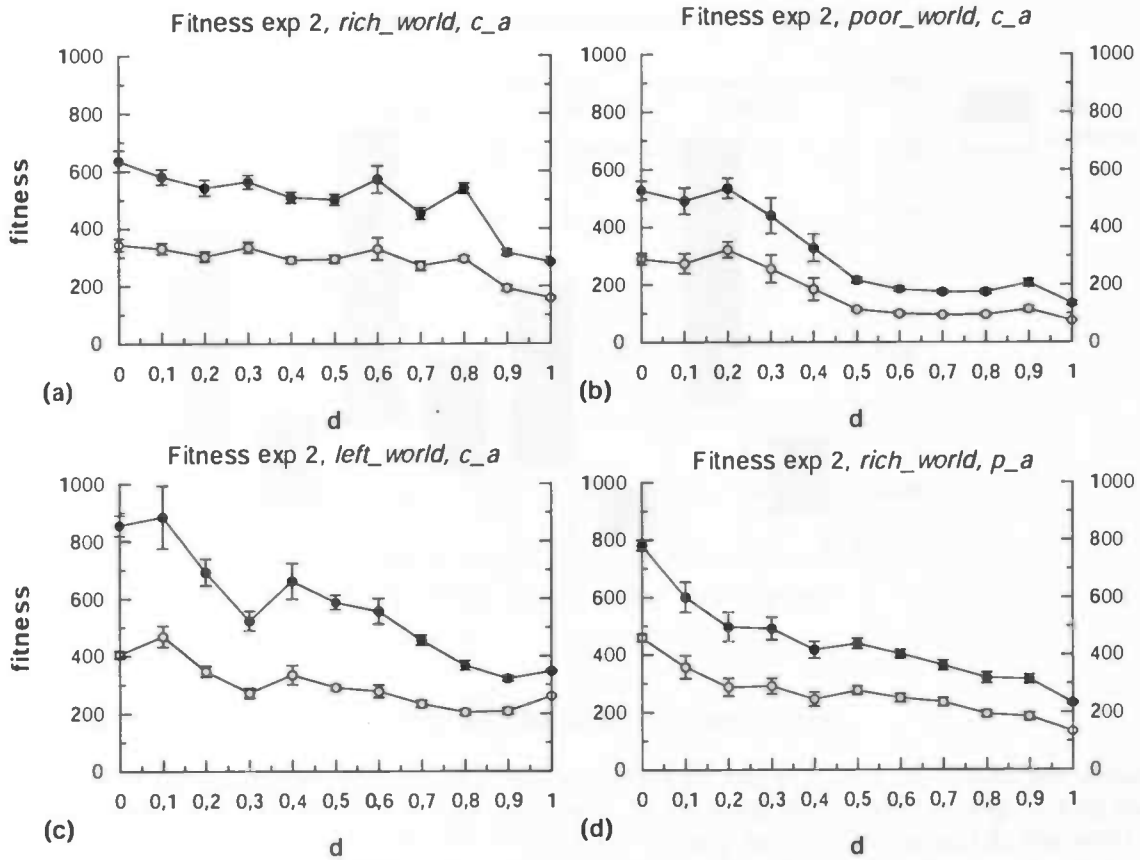


Fig. 4.2 Fitness results in experiment 1

In these graphs, **d** is the parameter that controls the trade-off between reactive and pro-active behavior. When **d**=0, behavior is purely reactive and when **d**=1, behavior is purely pro-active.

The overall image in these plots is that as the internal motivator takes over more of the behavior, the quality of behavior declines. This overall trend is to be expected since as behavior is more dominated by the motivator, the robot will be less able to react to different objects in the environment. When **d**=1, behavior is purely based on motivator activity. Evolution then comes down to finding motivator weights which let the robot move along a minimal dangerous path. It says 'let the robot move' since the robot is more or less blind.

To get a better picture of the differences in the usefulness of reactive and pro-active behavior between the different settings, figure 4.3 shows the fitness for the different settings for **d**=0 and **d**=1 (representing resp. reactive and pro-active behavior). As is clear from both figure 4.2 and 4.3, becoming pro-active has a negative effect on performance. We could subtract the average fitness for pro-active behavior from the average fitness for reactive behavior and use it as a measure for the the penalty for becoming pro-active in the specific setting. When we do that we get a picture in which the setting that has the best overall performance for pro-active behavior (performance in *left_world*) has the smallest penalty for becoming pro-active. The setting that has the best overall performance for reactive behavior (performance with *passed_angle*) has the largest penalty.

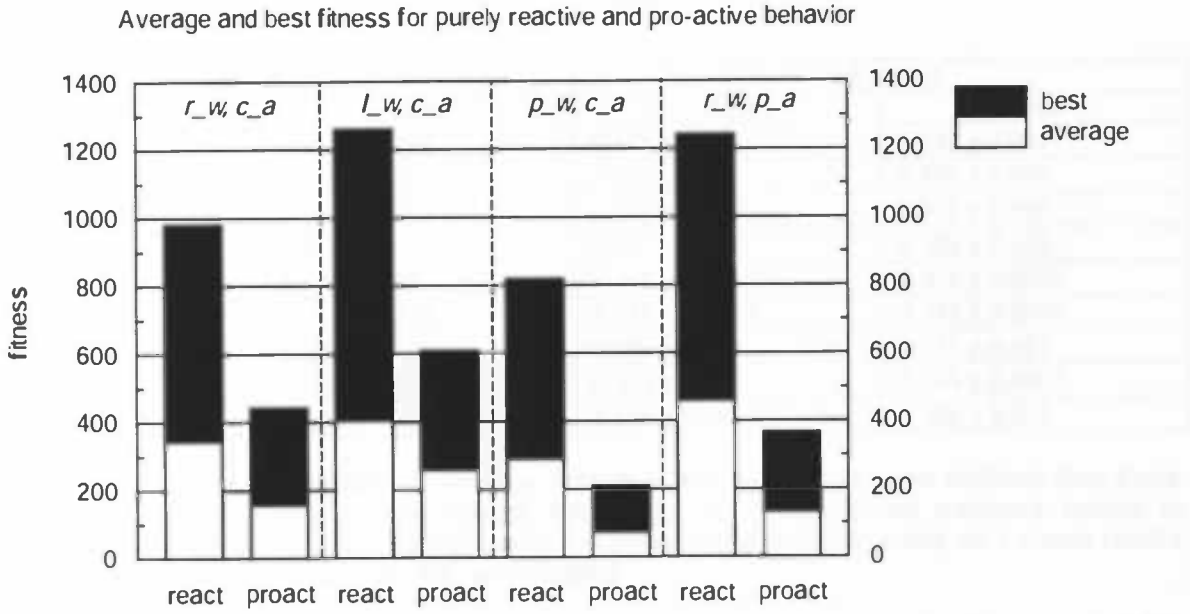


Fig. 4.3 Reactive and pro-active behavior for the different settings

A first comparison of data will be between figure 4.2a and 4.2d. In 4.2a, results are shown for individuals evolved in *rich_world* with *changed_angle*, which is the default setting. Figure 4.2d shows results with *passed_angle*, or *p_a*. So the difference is the way the robot is placed into the world. We can see that fitness in 4.2d is higher for more reactive behavior (d<0,2), but decreases more in the *passed_angle* setting as behavior becomes more pro-active. This means that the difference in functionality between reactive and pro-active behavior is larger with the *passed_angle* setting. This can also be seen in figure 4.3. Reactive behavior seems to elicit the best results in the *passed_angle* setting. But pro-active behavior has the most disturbing effect in the *passed_angle* setting.

One conclusion could be that behavior that is more reactive has a larger chance to come into a functional loop in which the angle of one good life is more or less the same at death as it is at birth. That way the robot learns to 'select' a certain subclass of stimuli which it knows will lead to higher chances for success in following lives. For this to happen, it must be able to interact with its environment and *react* upon sensory patterns with a minimum of distubance. The pro-active behavior has the form of turning behavior, so the more pro-active the robot becomes, the more this turning behavior can disturb the seeking-of-a-favorable-angle-at-death.

To see whether behavior can go into a functional loop, nine successive runs of one evolved individual with a d value of 0 were analysed which showed that five out of nine angles at death were in about the same direction. This suggests that the agent learns to make use of the passed angle.

A second comparison is between the results in figure 4.2a and those in *left_world* (4.2c). First, we see that fitness is higher for d=0.1 than for d=0.0, indicating the added pro-activity caused by the motivator is useful. This means that the *balance* between reactive and pro-active behavior is different in *left_world* than in *rich_world*. The motivator is more useful when a certain default turning angle in the behavior can increase the chances of encountering food-objects, which boosts fitness. This effect is more present in a world in which the distribution of resources is more biased, that is, shows clusters, which is the case in *left_world*. This is also seen in figure 4.3, in which pro-active behavior is most beneficial in the *left_world* setting. The structure of *left_world* is such that coupling between the environmental structure and the 'structure' of the pro-activity is possible and productive.

It is interesting to look at the evolved weights of the connections between the motivator and the motor output nodes in the network of individuals evolved with d=1. These individuals are all purely pro-active. If this pro-active behavior is more productive in *left_world* than in *rich_world*, as the higher fitness for *left_world* would imply, we would expect evolution to have found a solution for using pro-active behavior in *left_world*. When a solution is found, the weight values for the individuals in the evolved populations will converge and look alike. When pro-active behavior is less functional (as

seems to be the case in *rich_world*), we would expect more variation in the weight values, meaning more, but less optimal, solutions. Below are the average weight values for the nine populations in *rich_world* and *left_world*.

rich_world		*left_world*	
Wml	Wmr	Wml	Wmr
-0.374 ± 0.007	-0.619 ± 0.006	-0.452 ± 0.004	-0.437 ± 0.001
0.827 ± 0.06	0.555 ± 0.04	-0.489 ± 0.007	-0.460 ± 0.004
-0.555 ± 0.004	-0.682 ± 0.003	-0.397 ± 0.005	-0.379 ± 0.002
0.562 ± 0.1	0.393 ± 0.07	-0.392 ± 0.002	-0.380 ± 0.002
0.371 ± 0.08	0.709 ± 0.09	-0.394 ± 0.004	-0.379 ± 0.0006
-0.598 ± 0.003	-0.713 ± 0.005	-0.388 ± 0.0003	-0.378 ± 0.0000
0.616 ± 0.001	0.895 ± 0.004	-0.384 ± 0.003	-0.375 ± 0.003
-0.954 ± 0.008	-0.636 ± 0.004	-0.393 ± 0.002	-0.378 ± 0.0007
-0.863 ± 0.009	-0.623 ± 0.006	-0.395 ± 0.003	-0.380 ± 0.003

In this table we can clearly see that the weights in *left_world* show more similarity than those in *rich_world*. This suggests that an optimal solution for using the internal motivator existed in *left_world*. Figure 4.3 supports the usefulness of pro-active behavior in *left_world*, as it shows results for pro-active behavior are best in the *left_world* setting.

In figure 4.2b, results are shown for individuals evolved in *poor_world*. The fitness at d=0,2 is higher than at d=0,0, indicating added functionality of the internal motivator. The optimal balance between reactive and pro-active behavior now lies at d=0,2. Apparently, the pro-active behavior can successfully compensate for some of the loss in performance caused by the impaired reactive behavior.

Striking is the much larger difference between the fitness at low and high values of d in this graph as compared to results in 4.2a with the *rich_world, c_a* setting. As *poor_world* has much less resources, reacting on the resources that are present becomes even more pressing than in *rich_world*. As behavior becomes less reactive, the more important reactive behavior falls away, preventing the robot to react adequately, letting fitness drop. The difference between low and high d-values becomes clear when we look at the graph in figure 4.4, in which the fitness in *poor_world* is plotted as a percentage of the fitness in *rich_world*.

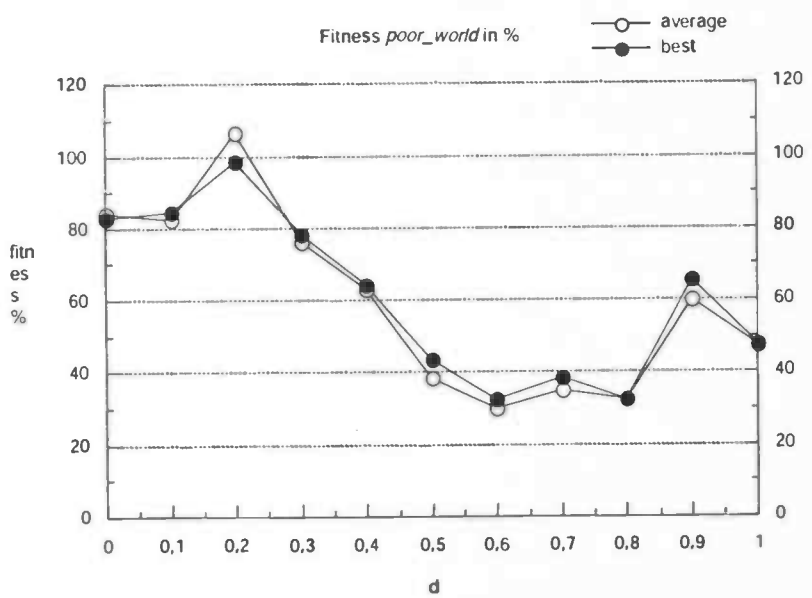


Fig. 4.4 Fitness of behavior in *poor_world* as a percentage of fitness in *rich_world*

This graph clearly shows a qualitative difference between low and high values of **d**. This implies that pro-active behavior is more disruptive in a poor environment. This can be explained by the fact that pro-active behavior is more 'blind', as it inhibites reactive behavior. Blind behavior is more dependent on *chance* for exploring for food. A rich environment offers more chance of encountering food elements than a poor environmnent.

In 3.2.3 the importance of *timing* was introduced as a result of the relative effect of the consumption of food elements on the energy level of the robot. We saw that a good temporal distribution of resources resulted in higher fitness. It was hypothesized that the concept of timing would become even more important in an environment in which resources were scarce, such as in *poor_world*. To investigate whether this is true, the connection weights from the bias node to the motor nodes are plotted with the accompanying fitness value for the particular individual. This is done for the evolved populations with a **d** value of 0 in both the *rich_world* setting as the *poor_world* setting. The bias weights determine the basic driving speed of the robot. The driving speed could be optimized for the environmental setting in which behavior is evolved. If timing is more important in *poor_world*, we would expect smaller variance in the weight values in *poor_world*. The two plots in figure 4.5 and 4.6 show these results.

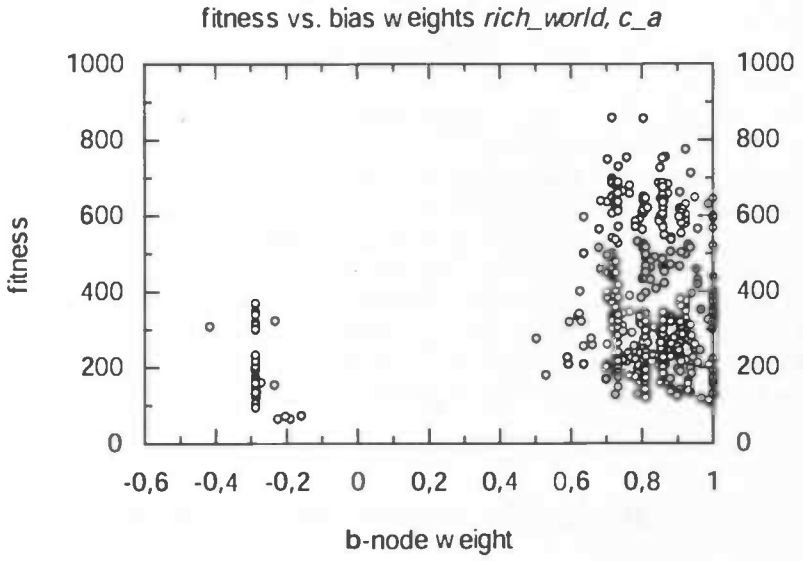


Fig. 4.5 Bias weights for *rich_world*

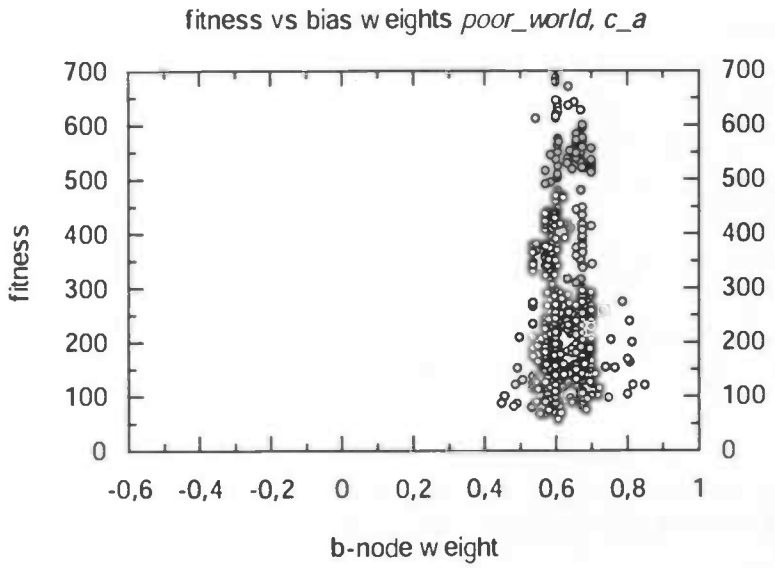


Fig. 4.6 Bias weights for *poor_world*

As we can see, the solutions in *poor_world* are much closer together in terms of bias weight values. In *rich_world*, we even see a few solutions with negative bias weights, meaning a backward bias driving direction. When a solution is found for satisfying selective pressure in the evolutionary process, this usually is accompanied by a small variance in value for the specific parameter. In this case, the selective pressure for finding a right bias weight is bigger in *poor_world* than in *rich_world*. So we would expect a smaller variance in bias weight in *poor_world*. The average weight variance for the bias weight in *rich_world* of the nine final populations is 0,0024. For *poor_world* this value is 0,0018. This indeed suggests that timing is more important in a poor environment.

4.2 Experiment 2: The speed of balance

In this experiment, the balance between the reactive and pro-active behavior is no longer a static factor, but is actively controlled by the agent. The parameter that controls the trade-off between recative and pro-active behavior is called **bal**.

In the first part of this experiment, individuals were evolved that were purely reactive (**bal** was initialized at 0) next to individuals that were evolved which had a dynamic balance between reactive and pro-active behavior (**bal** was initialized at 1). In the fitness plots below the average and best fitness of the nine evolved populations are plotted against time. The continuous line shows the average fitness of a population, averaged over the nine populations that were used. The dashed line shows the fitness of the best individual in the population, also averaged over the nine populations.

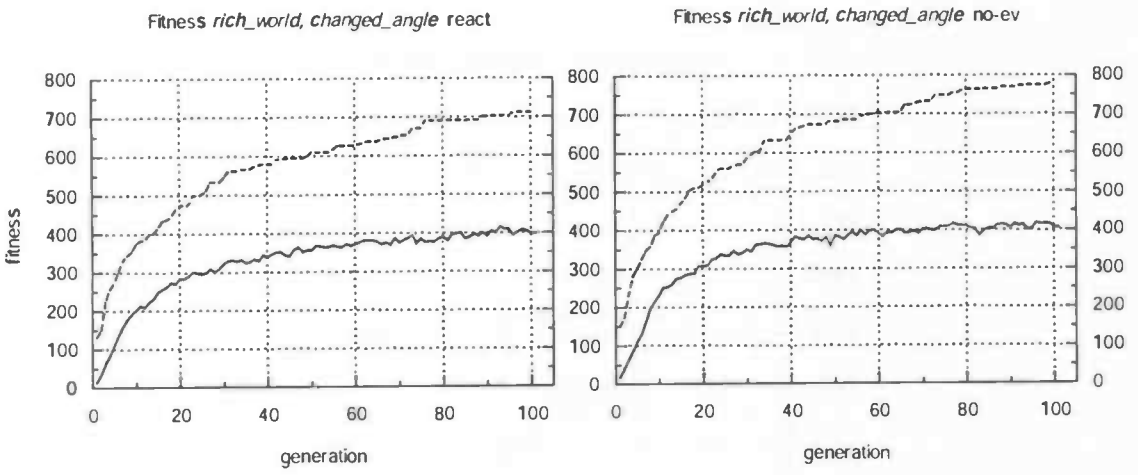


Fig. 4.7 (a) Fitness of behavior that is purely reactive (b) fitness of behavior in which a dynamic trade-off between reactive and pro-active behavior exists

Figure 4.7a shows results for agents that are purely reactive, since **bal** was initialized at 0, allowing itself no future change. Figure 4.7b shows results for agents that have an active dynamic balance between reactive and pro-active behavior, since **bal** was now initialized at 1.

These results suggest that at the end of evolution, individuals with an active motivator perform on average about the same as without. An explanation for this can be that under the conditions in which the individuals are evolved (*rich_world* and *changed_angle*), the motivator can only sometimes be advantageous. But there is a qualitative difference between the two plots. In the first 10 generations, we can see that performance rises more quickly in 4.7b than in 4.7a. Apparently, the dynamic balance benefits the evolutionary process and boosts fitness. A downside of a faster convergence however, is that it decreases variability within the population more quickly. This will have a negative effect on further performance increase, resulting in eventually about the same average fitness level in both figures. The beneficial effect of the dynamic balance is in the last few generations only observable in the higher best fitness value in 4.7b.

What would happen if we let evolution choose those individuals that have learned to make use of their motivation over those who have not? This is what was investigated in the second part of the experiment. In this part an additional parameter **bald** that functions as a damping factor for **bal** is co-evolved along with the network weights. This damping factor **bald** will decide how dynamic the balance betwen reactive and pro-active behavior can be and how well **bal** can do its job. A low damping factor will decrease the influence of **bal** and keep **bal** low in which case behavior 'stays' in the reactive regime.

Below are the average fitness results of the populations evolved with the evolving damping factor. The world and angle settings are the same as in the first part, that is, *rich_world* and *changed_angle*.

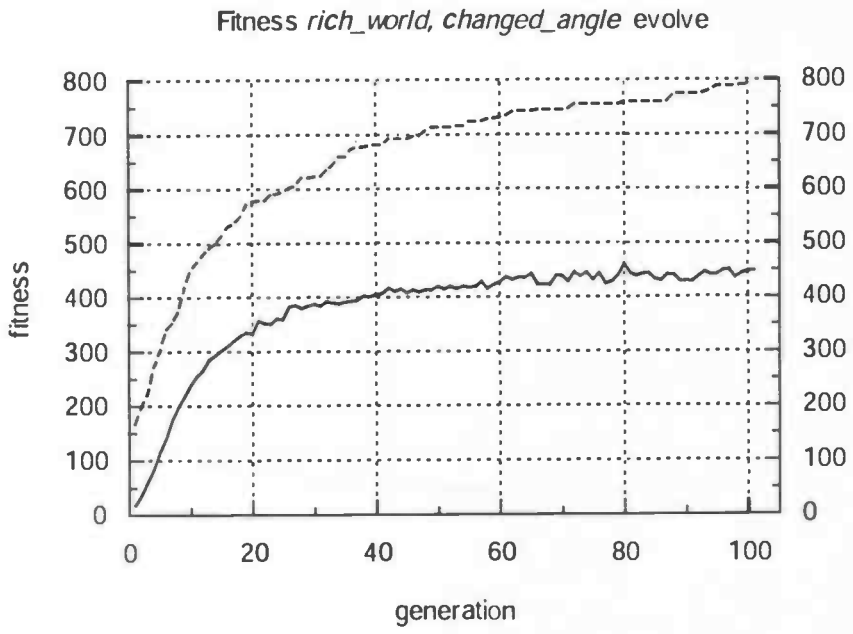


Fig. 4.8 Fitness of behavior with a co-evolved damping factor **bald**

These results (as compared with results in figure 4.7) show that fitness is increased when evolution can allow for more or less trade-off between recative and pro-active behavior. This will allow evolution to keep functional reactive behavior while in the same time exploit individuals that have found ways to use their motivator beneficially. Some individuals will have good reactive behavior but a disturbing motivator. Those individuals that evolve a low **bald** value will experience less disturbing influence from the motivator than those that evolve a high **bald** value. Individuals that have a beneficial motivator that adds functional pro-active behavior to overall behavior will benefit from evolving high **bald** values, since high **bald** values allow a higher range for the **bal** parameter. The higher the range in which **bal** can acquire its value, the more pro-active behavior is allowed.

It is intuitive that learning to make use of the internal motivator is a much more complex and difficult task than learning to react adequately to their environment. Figure 4.9 shows the average weight variances of the populations over time.

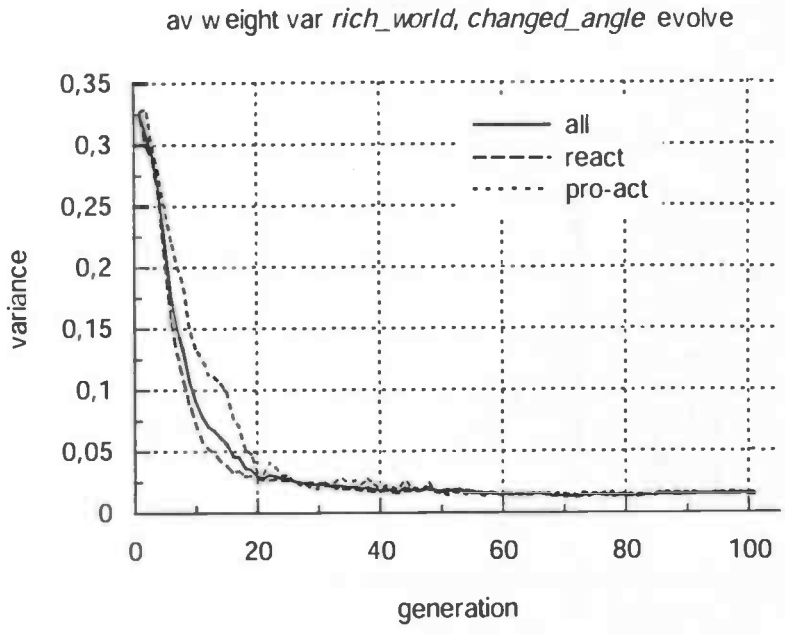


Fig. 4.9 Average weight variances of the populations evolved in *rich_world*, with the *changed_angle* setting and with the co-evolved damping factor **bald**

In this graph, the solid line shows the weight variances averaged over *all* weights in the network. The dashed line shows the average weight variance averaged over the weights belonging to the reactive part of the network (see figure 3.4). The dotted line shows the average weight variance averaged over the weights belonging to the pro-active part of the network. We can see in figure 4.9 that the variance of the pro-active weights take more time to converge than the reactive weights. This affirms that learning to use the internal motivator is a more difficult task. If we would separate the reactive and pro-active parts and evolve them seperately, the pro-active part would probably take even more time to converge. But the convergence of the reactive part limits the variability in values of the pro-active part, forcing it to converge earlier. However, the separation is only theoretically possible, because the two processes are coupled and behavior with only one of them is qualitatively different than with both processes working in parallel.

One way of investigating whether the internal motivator adds to the quality of the evolved behavior is to look at the relation between the evolving damping factor for the balance between reactive and pro-active behavior **bal**, which is **bald**. The higher **bald** is, the more **bal** can influence behavior and the more pro-active behavior is allowed to occur. In the following graph 4.10 the values for **bald** are shown with the accompanying fitness values of the individuals which they belonged to.

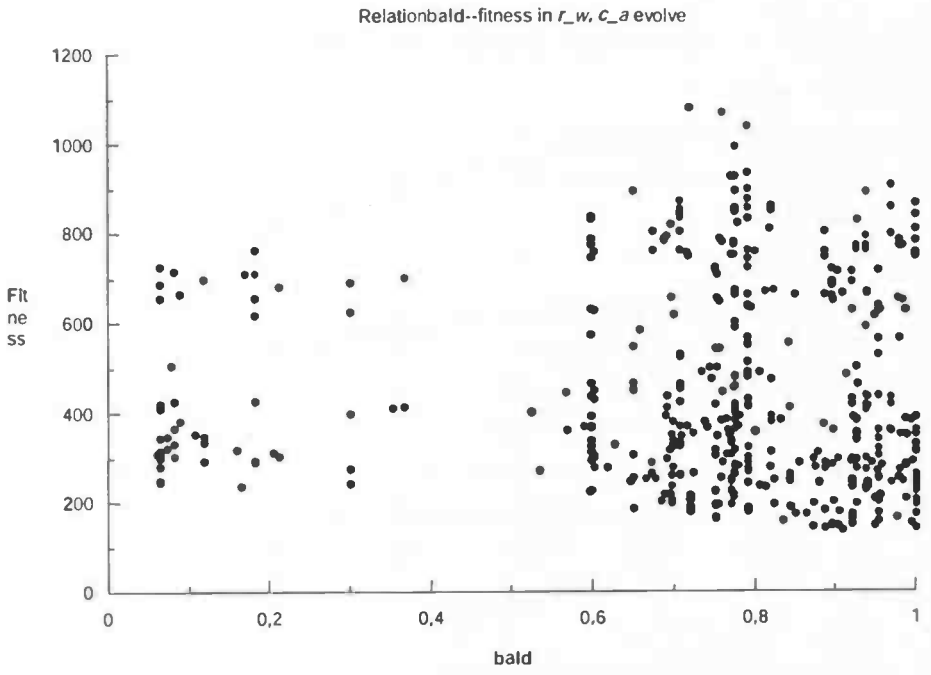


Fig. 4.10 Fitness values for different evolved values of **bald** taken from the populations evolved in *rich_world, changed_angle*

We can see that most individuals evolve a high **bald**-value. This suggests that a solution was found for the 'how-to-use-my-internal-motivator' problem. When the internal motivator would always be a disturbing factor on performance, allowing pro-active behavior would always impair the quality of behavior. In that case evolution would quickly filter out those individuals that allowed this pro-active behavior, resulting in a population with low values for the **bald** parameter. Since this is not the case, the pro-active behavior can be functional and beneficial.

The next graphs will show the results for the populations evolved in *left_world* under three different 'angle' conditions: *changed_angle, passed_angle* and *fixed_angle*. In all three cases, the damping factor **bald** was co-evolved.

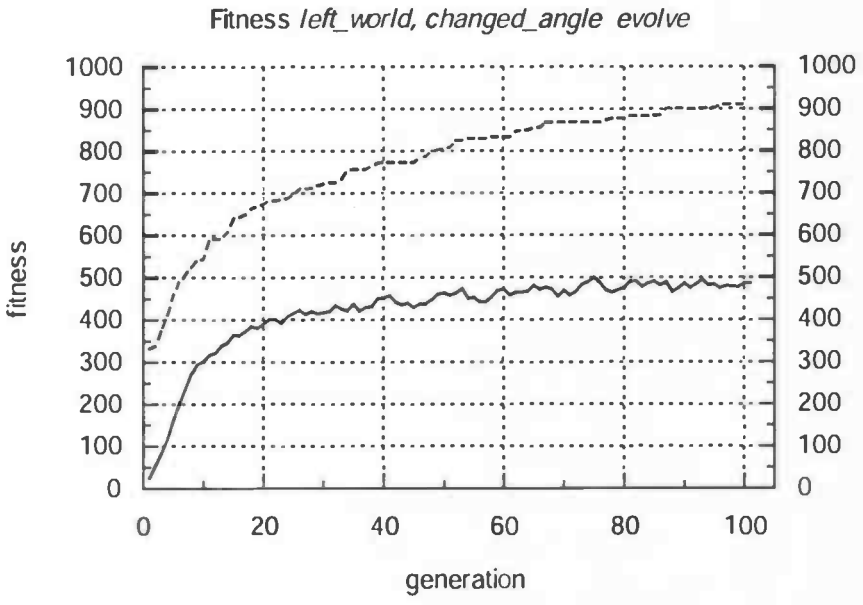


Fig. 4.11 Fitness of behavior in *left_world*, with the *changed_angle* setting and co-evolving **bald**

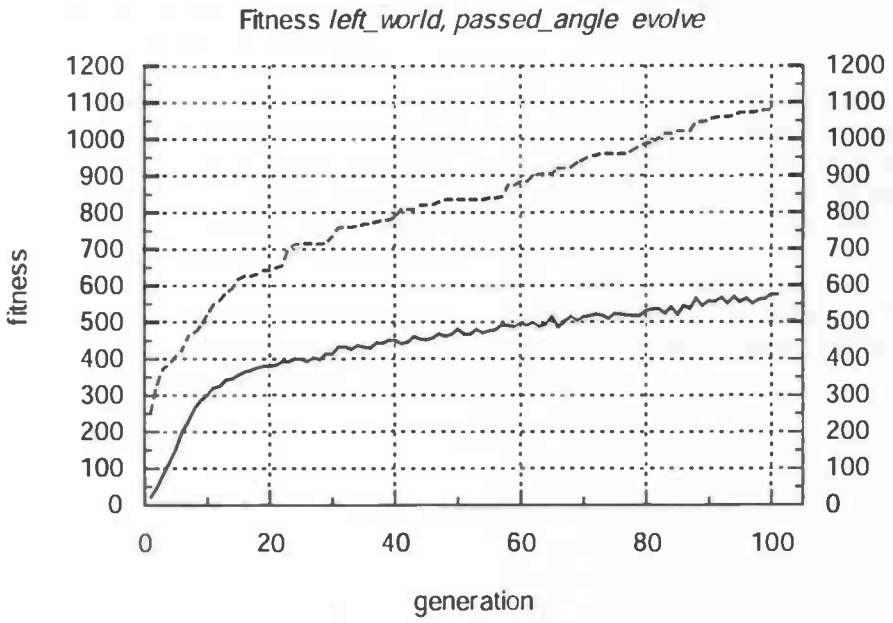


Fig. 4.12 Fitness of behavior in *left_world*, with the *passed_angle* setting and co-evolving **bald**

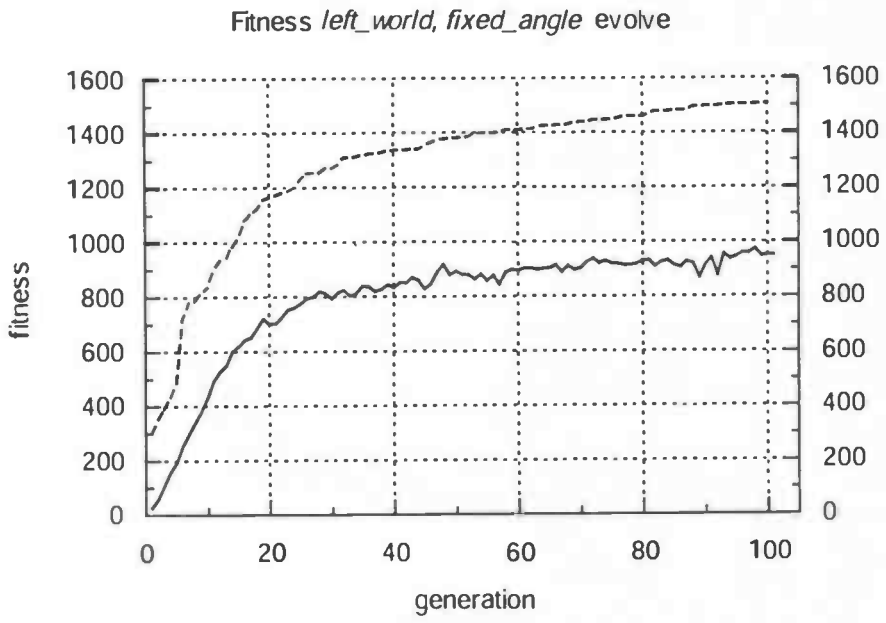


Fig. 4.13 Fitness of behavior in *left_world*, with the *fixed_angle* setting and co-evolving **bald**

The first noticable thing about these results is that fitness increases as the amount of *usable* structure in the environment increases. The more usable structure the agent has at its disposal, the easier it will be for the agent to seek out the sensory patterns within the structure that can be exploited by the use of the internal motivator.

Another thing we can see in the results for the *passed_angle* setting is that fitness keeps rising more after the first apparent moment of converging than is the case in the other two results. This is because in this setting, using structure is more difficult and takes more time to learn, since the structure depends on how the robot is placed in the world at the time of his death. The structure must be learned as information *over multiple lives*, instead of over just one life. As this is more difficult, evolution takes more time to learn to exploit this.

When we look at the variances in weights from the *passed_angle* setting and compare them with those in the *fixed_angle* setting we can see that variance for especially the pro-active part of the network stays higher in the *passed_angle* setting for a longer period of time. Below are the plots for the variance in weight values for the two settings.

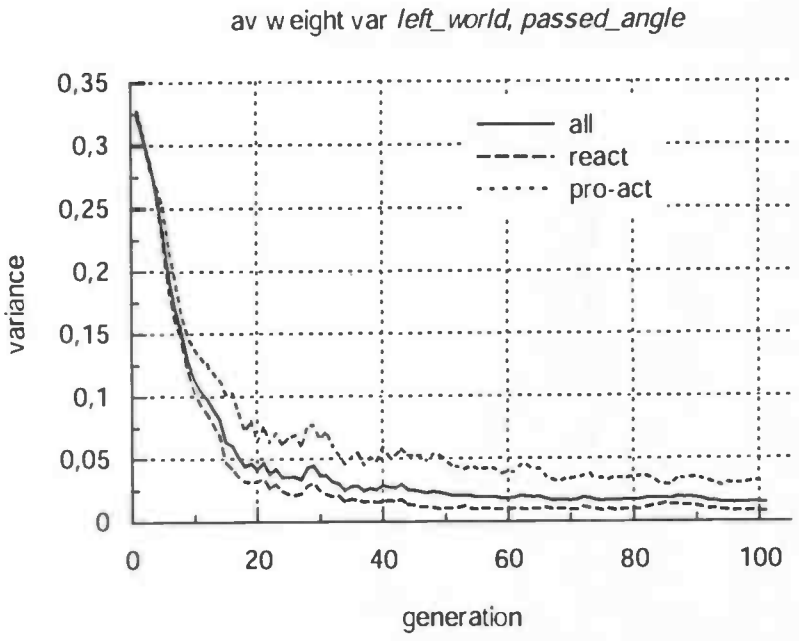


Fig. 4.14 Average weight variance in the populations evolved in *left_world*, with the *passed_angle* setting

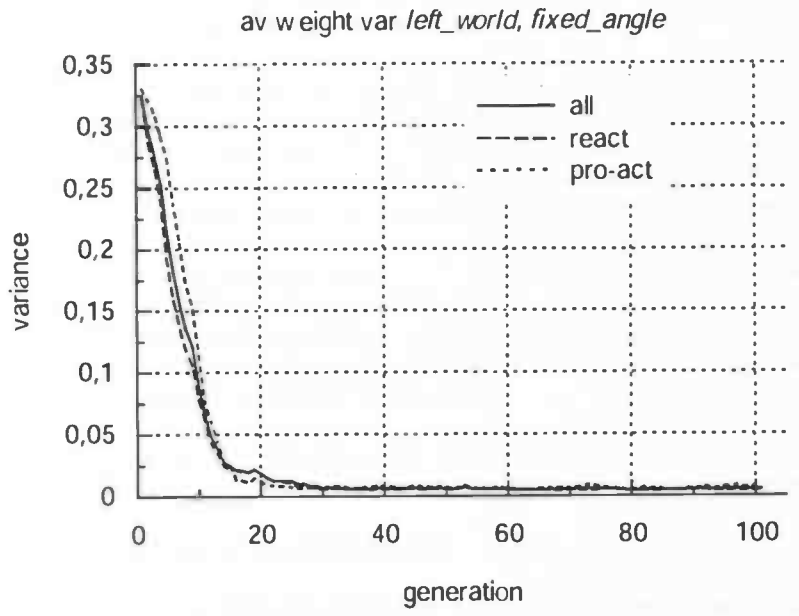


Fig.4.15 Average weight variance in the populations evolved in *left_world*, with the *fixed_angle* setting

Figure 4.14 and 4.15 indeed show that with the *passed_angle* setting, the pro-active weights take more time to converge than in the *fixed_angle* setting.

With *fixed_angle* one could suspect that performance is purely due to learning to react to sensory patterns as to learn to drive through some optimal path, so that the intenal motivator would not play an important role. We can look again at the relation between the damping factor for **bal**, **bald** and the fitness of the accompanying individuals. This is shown in figure 4.16 below.

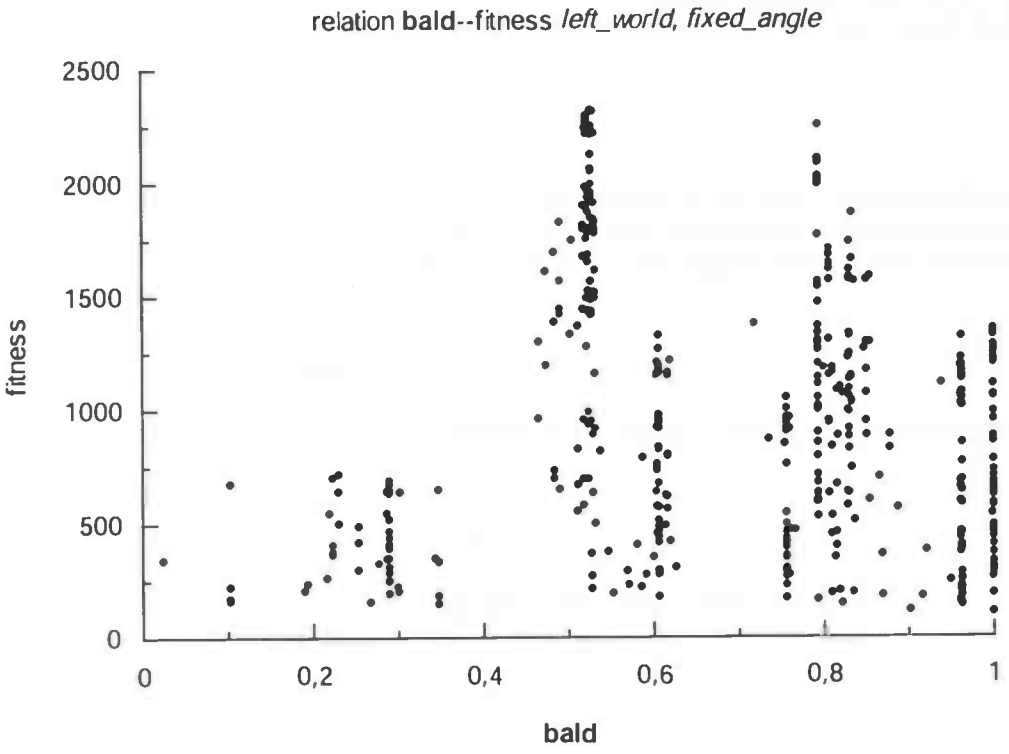


Fig. 4.16 Fitness values for different evolved values of **bald** taken from the populations evolved in *left_world, fixed_angle*

When we compare this graph with that in figure 4.11, we can see that more solutions are found at lower values for **bald**, which suggests that pro-active behavior is less important. But again in 4.16, most values of **bald** are > 0.5. The reason for the difference lies in the complexity of the task. In the *rich_world, changed_angle* setting the task of managing resources efficiently is more difficult. This means that the variation within the population, and in the reactive weights in particular stays high for a longer time. This gives the evolutionary algorithm more time to select those individuals who have learned to seek out sensory patterns in which the use of the internal motivatir is functional. In the *left_world, fixed_angle* setting in which usable structure is much higher, the resource-management task is much easier and so reactive weights will converge much more quickly. This time evolution has less time to exploit the use of the motivator, which makes it less important. This is shown in the relatively lower values for **bald** in figure 4.16.

To illustrate how the pro-active behavior can become functional, the next section describes the way in which an evolved individual successfully makes use of environmental structure.

Making use of environmental structure

As was described, the motivator node in the network causes turning behavior. The question is if and how such behavior could become useful. The usefulness of any behavior will depend on the environmental settings in which such behavior is placed. So, in what kind of environment would the pro-active turning behavior become useful? One answer would be an environment in which the distribution of food-elements show a certain arc-shaped structure. Individuals that can use this structure by using their pro-active turning behavior will certainly have an evolutionary advantage, since it increases chances of encountering food elements.

To test this, the behavior of an evolved individual that was evolved in *left_world* with the *fixed_angle* setting was investigated. From the 9 populations that were evolved, the 6th population showed a very high **bald** value of 1.0. This means that no damping was applied to the operation of the dynamic balance parameter **bal**. So **bal** could operate within full range. This suggests that evolution somehow discovered a way to use pro-active behavior. This is strengthened by the fact that the 6th

population started of with an average **bald** value of 0.46, shifting to 1.0 towards the end of the evolution process.

Now lets take a look at the turning behavior of a typical individual from the 6th population. To get a measure of the radius of the turning behavior, the linear velocity was stored during a few test runs. This velocity was calculated with:

$$v = \sqrt{(\Delta x^2 + \Delta y^2)} \qquad (4.1)$$

in which v is the linear velocity and (x,y) are the coordinates of the robot. This calculates the velocity in environmental terms of ep/sc, which stands for *environmental points/simulation cycles*. Also, the angle of the robot in relation to the x-axis of the environment was logged. The change of this angle, α, over time gives the angular velocity, ω:

$$\omega = d\alpha / dt \quad \text{, in rad/sc} \qquad (4.2)$$

To calculate the radius of the circling behavior of the robot, a formula was used taken from circular motion mechanics:

$$r = v / \omega \qquad (4.3)$$

From the data of the test runs the following values for v and ω were calculated:

$$\omega = 0.008 \text{ rad/sc}$$
$$v = 2.203 \text{ ep/sc}$$

This gives:

$$r = v / \omega = 274.7 \text{ ep}$$

Now lets take a look at a typical path produces by an evolved individual from the 6th population evolved in *left_world* with the *fixed_angle*.

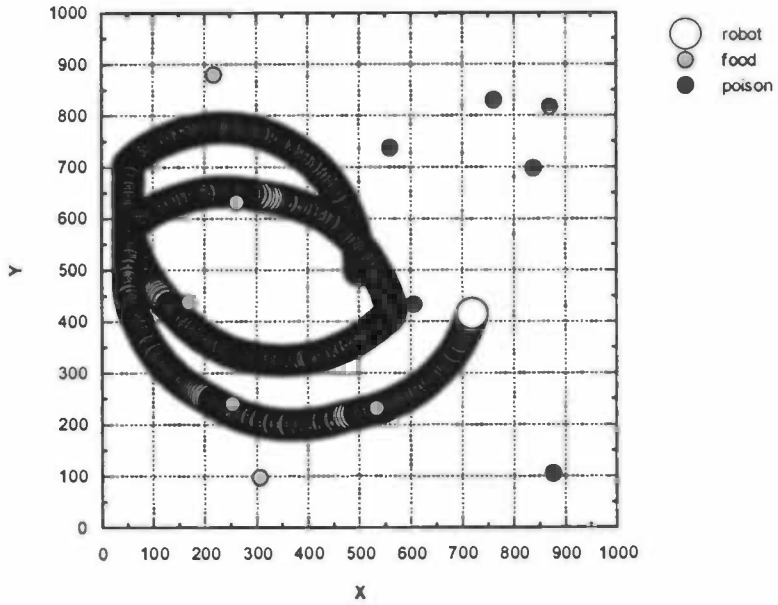


Fig. 4.17 Path of evolved behavior of a typical individual from the 6th population evolved in *left_world*, with the *fixed_angle* setting

In this picture we can see how the robot has learned to use the environmental structure. It uses a turning angle which increases chances that it will encounter the poison element in the middle of the environment. Encountering this element will result in turning back towards the left side of the environment. Because it stays on the left side of the environment more, chances will increase that it will find food elements. Making use of its pro-active turning behavior, the robot has effectively increased chances to encounter food elements. Lets have a look if we can find a similar radius in the structure of *left_world* as we have calculated from the behavior of the robot.

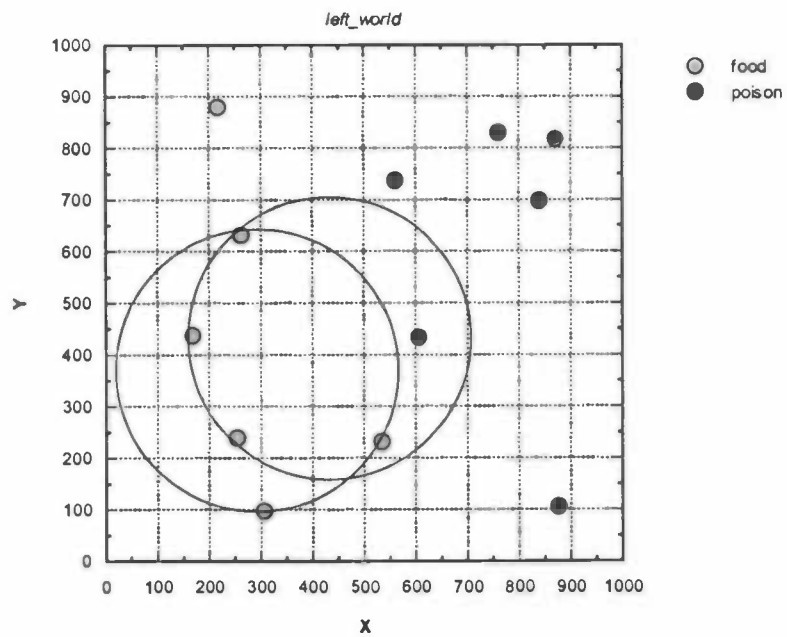


Fig. 4.18 Two instances of circles in *left_world* that have the same radius as the evolved pro-active behavior evolved

In fig. 4.18 two circles have been drawn that have the same radius as the calculated radius of the circling behavior of the robot. We can see that the structure in *left_world* is such that circling behavior with this particular radius has an advantage. So using the internal motivator in a way that produces circling behavior with such a shape will offer evolutionary advantage.

In the third part of experiment 2 the best populations for three settings evolved in *left_world* were put in an alternative world, *right_world*. The populations were evolved and performance was measured. The results of the different populations are shown in figure 4.19 through 4.21 below.

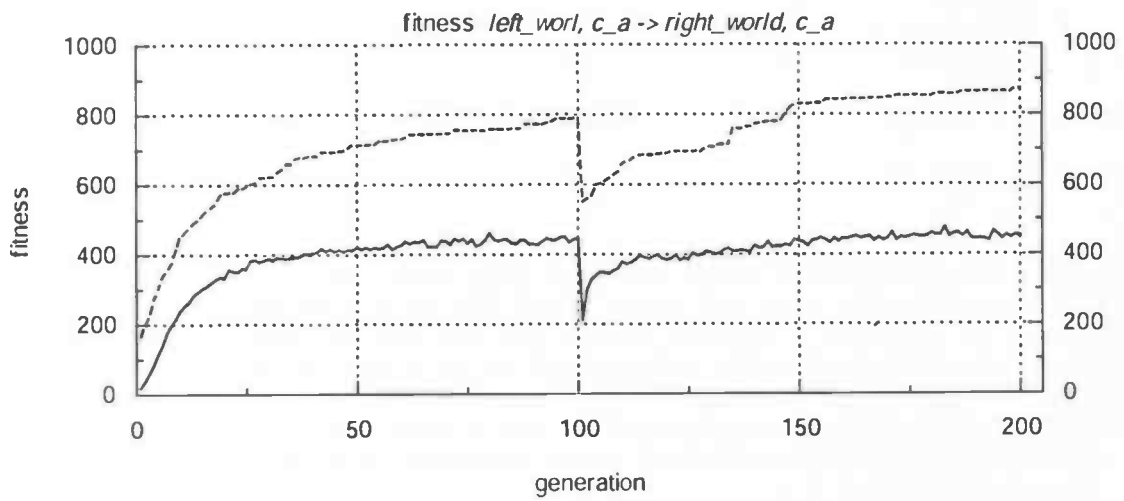


Fig. 4.19 Fitness of behavior when transferred from *left_world, changed_angle* to *right_world, changed_angle*

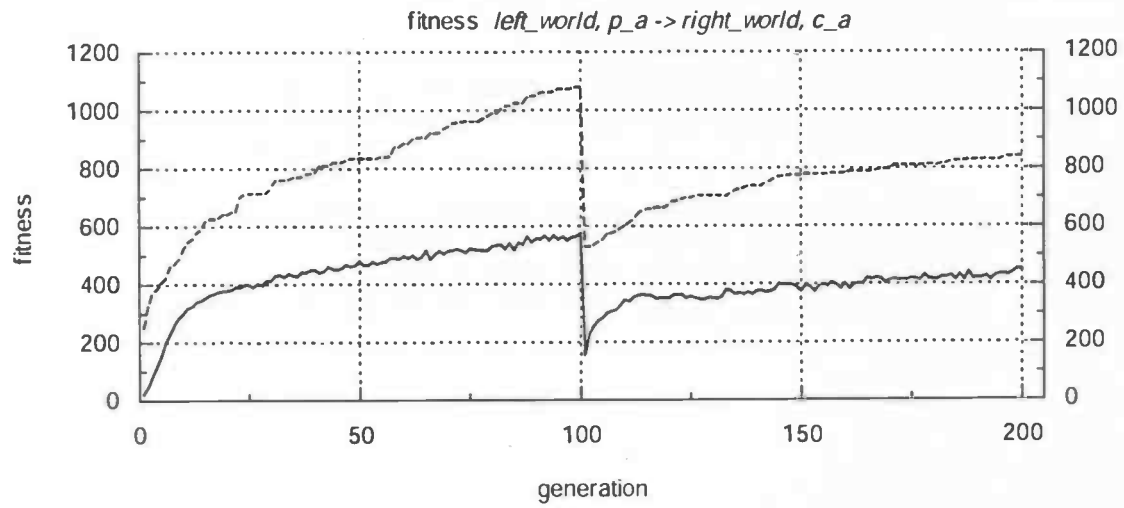


Fig. 4.20 Fitness of behavior when transferred from *left_world, passed_angle* to *right_world, changed_angle*

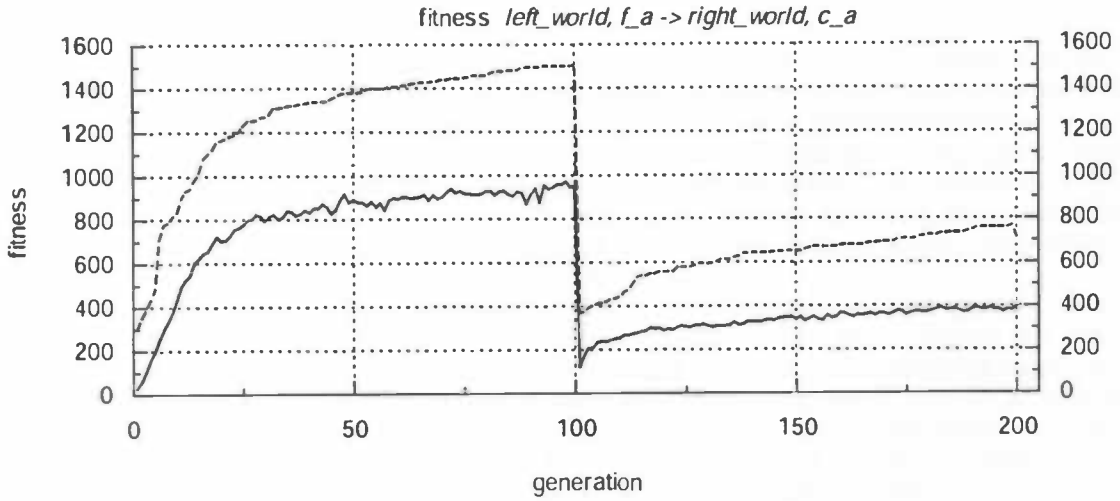


Fig. 4.21 Fitness of behavior when transferred from *left_world, fixed_angle* to *right_world, changed_angle*

In figures 4.19-4.21 fitness values up to generation 101 are taken from the results of evolution in *left_world*. From generation 102 to 202, fitness results are shown when the populations were transferred onto *right_world*. We can see very different changes in fitness for the three different settings. When we take the average of the last 10 generations at the end of evolution in *left_world* and compare them with the average fitness of the last 10 generations at the end of evolution in *right_world*, we can calculate the average final fitness in *right_world* as a percentage of the average final fitness in *left_world*. The same is done for the best fitness values. The results are shown in the table below.

	4.19 *r_w, c_a*	4.20 *l_w, p_a*	4.21 *l_w, f_a*
average fitness in %	102	77	41
best fitness in %	111	78	50

For the *changed_angle* setting this value is 102%, which means that the increase in fitness when populations are shifted from *left_world_ to right_world* is 2%. The individuals evolved in *rich_world* do not seem to have large problems adapting to the new environment. Populations with the *passed_angle* setting have a decrease in fitness of about 23%. But the fitness for populations evolved with *fixed_angle* drops a staggering 59%! These results clearly show that the higher the level of usable structure that was present in the environment in which evolution took place, the less individuals were able to adapt to a novel environment.

So the more usable structure evolution can use, the better performance will get, but also the worse performance will react to novelty. This is quite intuitive, since a higher level of structure will increase the level of dependance *upon* structure of the organisms that evolve in it. The more there is to use, the more it will be used, but also the more dependent behavior will become.

5 Conclusions

In the previous chapters a picture has been created of how a simple model of behavior in which reactive and pro-active influences can be identified behaves in various environmental settings. Different aspects of the basic model have been investigated. A central characteristic of the model was the presence of an internal motivator, which embodied a kind of internal belief. The internal motivator could add extra internal activation which made behavior more pro-active. The experiments showed that the optimal balance between reactive and pro-active behavior is greatly determined by the environmental settings in which behavior was evolved. Settings that gave smaller penalties for the use of the internal motivator usually had a more pro-active balance.

Also, we have seen that the way individual robots were introduced to their environment greatly influenced the balance. The *angle* setting was used to investigate this. Angle settings that optimized the *predictability* of the environment (such as *fixed_angle*) made the environmental structure more usable, increasing the functionality of pro-active behavior. On the other hand, with angle settings that resulted in more unpredictable or uncertain approaches of the robot to its environment, performance became more dependent on reactive behavior. In those settings (such as *passed_angle*), pro-active behavior had the largest penalty as it disturbed the more crucial reactive behavior.

Overall, we saw that as the amount of usable structure increases, performance increased as well. When we gave control over the balance between reactive and pro-active behavior to the evolutionary algorithm and made this balance a dynamic parameter in the control of the agent itself, performance increased even further. The evolutionary algorithm learned to exploit the internal motivation given the level of use it had. This depended on the amount of usable structure available. When the algorithm became too much dependent on the available structure, performance also became more rigid in the sense that it was less adaptable to new environmental situations. We saw this when individuals that were evolved under situations with high levels of usable structure performed very badly when transferred onto a new environment.

The results in this thesis embody a common psychological or behavioral principle. When behavior comes to rely too much on its environment, adaptation to new environments will be more difficult. And, the dynamic balance between reactive and pro-active behavior resulted in behavior in which the agent could react quickly on sensory patterns and would become pro-active in a more slow or hesitant way, but only as long as the environmet did not exhibit much change. As performance with this dynamic balance was much higher than that in which the balance was fixed, this shows that pro-active behavior can be seen more as a luxury, that can add some functionality when the environment doesn't call for immediate action. When the environment provides enough sensory patterns on which functional behavior can be based, it is more wise to react to those sensory patterns. But when the environment does not provide enough sensory patterns, it can be useful to behave more pro-actively in a way that will increase chances to encounter useful sensory patterns in the future. So there is wisdom in *'react fast, think slow'*.

5.1 Related work

Motivation and Emotions

Velásquez

Velásquez (1998) sees emotions as a sort of biasing mechanisms during the action-selection process. In his model of behavior, a Drive System along with an emotion generation system influence the behavior system. The behavior system ultimately produces an action to be carried out. An intersting aspect of the model is that within the Drive System, each drive represents a motivational system or urge. The drive identifies special conditions in the sensory data that can either increase or decrease the magnitude of the drive. This is closely related to the influence of the external dynamics on the value of the internal motivator in my model. A main influence in his research is Damasio (1994)

who states that the reasoning process includes the use of a covert, nonconscious mechanism that *directs* us towards the "right" decision.

Ventura

Ventura (2000) has done research based on Damasio's claim that rationality cannot be understood seperately from emotion. And secondly, emotions involve the body, a physical part of the organism.

In his paper, Ventura gives a description of the biological basis of the role of emotions in human intelligence. Ventura refers to Damasio, who suggests that emotions are critical in the rational (intelligent) thought processes. According to Damasio, sensory stimuli that are mapped onto topographic maps in the brain (called *images*) are marked with a somatic (as relative to the *body*) representation. These representations stand for a kind or gut feeling about the perceived data. This gut feeling or emotion can influence the kind of action that will follow from the perception. The magnitude and form of the influence of the emotion depends on the level of *relevance* of the representation to the external stimuli.

A similar process occurs within the implementation of the model proposed in this thesis. In the neural network the internal motivator represents a certain belief of gut feeling about the outside world. The relevance of this feeling is modelled by the external dynamics parameter. The higher the external dynamics, the lesser reason for the agent to become pro-active, so the less relevant the belief represented by its internal motivator will be. So the internal motivator can be compared with an emotion influencing behavior. The external dynamics represent the level of relevance of the internal motivator to the external stimuli. When the robot find himself in a stimulus-rich environment, his internal belief will have a different level of relevance to this environment than when it would find itself in a stimulus-poor environment.

Prem

In (Prem, 1996), Erich Prem describes how the notion of motivation and emotion can be used in the design of autonomous agents. In his article, Prem uses the ideas of J.v.Uexküll. Prem gives a model of behavior in which different behaviors compete in parallel controlled by different motivations. The motivations are themselves activated by drives. Motivations are concerned with the kind of behavior that the agent should have given its current needs. The needs (or drives) represent the relative importance or urgency of an engagament at a certain time.

This notion of 'needs' is closely related to the Drive System in (Velásquez, 1998). In both cases the drives or needs resemble a measure of relevance of a certain motivation to the current environment. The idea of 'needs' in Prem's article could be translated into the external dynamics parameter ed in my model, since the external dynamics gives a measure of relevance of the internal motivator to the current environmental situation.

Prem states: "Motivation and, even more, emotion contribute to a system behavior which will be less predictable than their purely reactive counterpart. ...emotional systems tend to process information that is only derived from observing internal states of the agent. In the case of motivations, the input can also come from external incentives, e.g. food in front of the agent and thus drive the behavior on a more reactive basis."

His statements about the role of motivations closely resemble the way in which the internal motivator influences behavior in my model. The second sentence even describes more or less the behavioral dynamics that occurs in my experiment. When the agent is in front of food, changes in its perception will rise, resulting in a heightened level of external dynamics. The increased external dynamics then drive the behavior on a more reactive basis because it inhibites the activation of the internal motivator and thus driving behavior away from pro-activity.

Cecconi and Parisi

Cecconi and Parisi (1993) too have done research with neural networks with motivational units. In their experiments a motivational unit was added to the input layer of a neural network and provided a binary value which represented one of two motivational states: 'hunger' or 'thirst'. They showed that the addition of this motivational unit increased the autonomy of the organisms. The difference with my experiments is that in their case there is a very direct link between the motivational unit and the fitness function used to determine the success of the organisms. In my case this link isn't as direct. The internal motivator represents a certain belief about the world. Whether this belief is functional or not is greatly determined by the structure of the environment itself as we have seen.

Another difference is that their world and motivational unit are discrete and my world and internal motivator are continuous. Both the environment-driven aspect and the continuous representation of the system could add to the biological plausibility of my experiments.

Evolution

Nolfi

In chapter 2, we discussed the notion of self-selection of stimuli (Nolfi 1993; Nolfi 2000). Within evolution of reactive agents, Nolfi found two different processes taking place. On the one hand, evolution selects individuals that can react adequately to sensory patterns and whose patterns are less affected by the aliasing problem or ambiguity or by the lack of regularities within groups of sensory patterns. On the other hand, evolution will select individuals that can exploit their sensory-motor coordination as to avoid ambiguous sensory patterns or patterns with a low level of regularity. So, evolution will select those that have somehow learned to react efficiently to their environment and at the same time select those who can more actively use their interaction so as to seek out more favorable sensory patterns.

Translated to my experiment, evolution typically will first select individuals that have learned to avoid poison-elements and approach food-elements. Parallel to this process, evolution will select those individuals who can use their sensory-motor coordination to seek out sensory patterns in which the use of their internal motivator has positive effect on overall performance. When we looked at the variances in the weight-values of an evolving population, we saw that the variances belonging to the weights in the reactive part of the network converged within a few generations. Those belonging to the pro-active weights took more time. The benefit of the motivator depends on how well the agent can create good conditions for its use by reacting in a certain way to sensory patterns.

Environment

Menczer and Belew

The world provides a context in which the agent can link its internal structure to an external one. In that sense the world is partly responsible for the behaviour of the agent. Some have argued that the environment is greatly responsible for the nature and complexity of evolved behaviour. Menczer and Belew (1996) argue that the complexity of the environment shape the selective pressures that control the adaptive (evolutionary) process. In the light of the current experiments, this complexity could be expressed by the number of elements in the environment. Then the number of elements present in the world would characterize the selective pressure that is placed upon the population of individual organisms. Another metric for environmental complexity is patchiness, that is, the level to which similar elements are clustered together. When patchiness is large, similar elements are more clustered, resulting in more spatial structure and therefore complexity.

In my experiments, complexity could be compared with the notion of *usable structure*. We saw that when the amount of usable structure increased, pro-active behavior had more opportunities to become functional in the light of the evolutionary task. In the words of Menczer and Belew, more complex environments ask for more complex behavior. Behavior in which the internal motivator adds an additional degree of freedom is more complex and could therefore be more suitable in more complex environments.

Lund and Miglino

The relation between behavior and environment was also studied by Lund and Miglino (Miglino 1996; Lund & Miglino 1998) where they set up an experiment in which a Khepera robot was evolved to navigate toward a target area within a rectangular world. The experiment was based on another experiment done with rats in which the rats had to look for a food patch in an open box. The researchers used the experiment to show that rats build a map of the environment and use it to navigate. In Miglino's experiment, in order to be successful, the robot had to distinguish between the long and the short wall in the environment. The way it solved the problem was by making use of a certain angular trajectory in which the robot always ended up against the longer wall. The shape of this angular trajectory, however, depended heavily on the dimensions of the environment. So the ability of the robot to get to the target area was due not only to how the robot reacted to sensory patterns but also on the structure (in this case the dimensions) of its environment.

Structure

Garner

In his book, Wendell R. Garner (1962) speaks about internal and external structure of a closed system. Internal structure is defined as the set of relations between parts of the system. External structure is defined as the totality of invariant relations between the internal structure and some other system. In Garner's terms, the *environmental structure* in this thesis can be seen as internal (to the environment) structure and *usable structure* as external structure.

5.2 Future work

Internal dynamics

In section 3.2.2 the internal motivator was discussed. The creation of the activation value of the motivator node in the network involved a parameter that controlled the *plasticity* of the activation value. This parameter, called the **internal dynamics** parameter or id-parameter, was set to 1 in the two main experiments. In the course of the experimentation an experiment was done before the two experiments outlined in this thesis were performed. In that preceding experiment behavior was evolved with various levels of internal dynamics. However, the setting in which behavior was evolved did not give a very high level of functionality to the pro-active behavior. Therefore, varying the level of internal dynamics did not influence performance very much. When the experiment were to be performed again in a setting with higher levels of functionality for pro-active behavior we may expect different results. For completeness, the experiment along with the accompanying model of behavior is given in the appendix.

Implementation

The network that was used to implement the basic model of behavior as discussed in chapter 2 was only a very simple and crude implementation. As the implementation gets more complex and can take in more complex sensory information, variations within the collection of possible behavioral patterns will probably increase. This could increase the number of tasks that can be learned.

This especially applies to the implementation of internal motivation. The implementation in the network included just one node. The resulting behavior therefore is not very exiting. But even the simple behavior could become functional given the right environmental setting. An implementation is perceivable that is more complex and dynamic and could adapt itself to environmental and historical information. Such an implementation could show its use in a more varied collection of environmental situations as well as be more adaptable to novelty.

Also, further study of the time window used for the calculation of the external dynamics could shed more light on how much history should be taken into account in order to limit the disturbing effect of pro-active behavior.

The analysis of behavior in which two of more systems are coupled is a very difficult problem. A lot of extra work could be done to investigate the precise behavioral dynamics present in the experiments. Some questions are: how do certain weights change just prior and after a convergence point in evolution? Does the change embody a qualitative change or just a quantitative one? How many solutions of the resource management problem are found? and so on. As more analytic data is collected, more insight into the nature of the evolution of behavior in which reactive and pro-active elements can be identified will be gained. But the relation between usable structure and the use of motivation in behavior will probably only be strengthened.

Adaptation

The concept of *adaptation* is closely related to the balance between reactive and pro-active behavior. A system is called adaptive when it can keep its parameters within physiological boundaries. For instance, when a cat approaches a fire it can just walk into the fire, which will likely kill it. It can also react on the heat and stay away from the fire. The second behavior is called adaptive, because in that case the cat succeeded in keeping some parameter (body temperature) within physiological boundaries. The first behavior therefore is not adaptive. An interesting experiment for instance could

be to create an environment with different temperature gradients and place an agent in that environment. The agent has a certain internal temperature but will gradually take over the temperature of the environment. When it stays in a spot that is too cold, its body temperature will drop below physiological boundaries. So it will need to respond on environment temperature. We can then compare a purely reactive agent with an agent that will pro-actively seek out an optimal spot in the environment.

Final word

Many more such experiments could be conceived and much more results can be derived from them. Experiments in which the intricate couplings of agent and environment play an important role and in which behavior of that agent is a combination of reactive and pro-active components are not very common, if not rare. Some works have been investigating the role of the amount of complexity of the environment in the evolution of behavior. Others have investigated behavioral models in which motivation or emotions play a central role. I have tried to bring these two areas together. I hope that as more results from both areas are made, more of this kind of experimental marriages will occur.

Appendix

Simulation vs. Real world

The experiments are performed in a simulated environment and not in the real world. The main reason for using a simulation of the robot and its environment is time. In the experiments an evolutionary algorithm is used. Such algorithms use large populations of individuals, which all have to be tested in a certain world for every generation. Doing just one such run with the real robot would take approximately one minute, including placing the robot back to its original position. A typical run of one of the experiments include about 225.000 individual runs of the robot. Such a run would be very time consuming in the real world. A second reason for doing the experiments using a simulator is the way the robot interacts with food objects. Once it has 'eaten' an object it disappears from the world. Doing this in the real world would demand quite sophisticated testing environments.

Many claim that simulation and the real world have such distinctly different levels of complexity that real-world experiments should be chosen above all. In my view, what really matters to cognition, is not the objective physical world, but the world *according to the agent*. So the important aspects about the world are only those that matter to the agent's mind. And these can be only as complex as the agent's structure allows them to be. Therefore, the 'simulation-vs-real-world debate' is not very useful. The focus should lie on the *agent's* world (*Umwelt*), not the *real* world.

When using simulation, we must ask the question if using a simulated robot and environment would weaken any claims, based on the results, about behavior of a real autonomous robot. For instance, Jakobi, Husbands and Harvey (1995) argue that for robot simulations to have any correlation with the real world, great care has to be taken with the noise levels used in such simulations. They have showed that applying a realistic amount of noise to their simulations, transfers of evolved neural controllers from the simulation to a real robot generated about the same behavior on the real robots. With an absence or excess of noise, performance after transfer to the real robot was much more impaired. Version 2.0 of the Khepera simulator takes in account these problems and a realistic as possible amount of noise is being added to the sensors of the robot.

Recurrency and sensory-motor coordination

When designing neural networks for control tasks, one of the issues is whether to make use of a purely feedforward structure or a structure with recurrent connections. A recurrent structure is a structure with one or more feedback-loops in which activation also flows back to a lower level in the network, affecting indirectly the source of that activation. The question whether one should make use of a recurrent structure very much depends on the task at hand. The use of recurrency doesn't necessarily mean better performance. Nolfi and Miglino (in press) suggested that when evolving networks, the extra recurrent connections can also disturb the quality of behavior. In some (if not many) cases the task can be performed by means of a simple feedforward network. As the evolution process runs, a pure feedforward solution of the problem could be discovered. The recurrent connection weights are also evolved, but will tend to be neutralized because they offer no additional functionality to the solution. But further mutations of the recurrent weights can have a negative effect on resulting behavior. So recurrency should be used only when it is fundamental to the behavioral problems for which the network is evolving.

Recurrent structures are often used for representational purposes. By feeding back activation a kind of memory of the internal state is created. This could in principle lead to better performance in many tasks since more information is available about the environment. But Nolfi and Miglino also showed that the absence of any internal representation created by recurrent structures can be overcome by exploiting the power of sensory-motor coordination (Nolfi and Miglino, *in press*; Nolfi and Parisi, 1993). In artificial evolution, networks improve in selecting favorable stimuli and making useful coupling between those stimuli and the internal representation of the evolutionary task. Again, this notion of representation does not result from recurrent feedback structures, but is present in a more abstract way in the total weight-matrix of the network. After time, the network will come to represent a

solution of coupling the kind of interactions typically encountered to the nature of the evolutionary task. But care should be taken with this notion of representation. The kind of representation that is present in pure sensory-motor systems is only functional *given* the structure of the environment in which the agent interacts, because only *certain* environments allow for the kind of interaction in which a successful coupling by the network between the environment and the task is established.

The XOR-problem

The XOR-problem is a specific case of a more general problem, namely that of classifying points in a hyperspace, stretched by the state-parameters of a system. In this case, the system is a neural network and the state-parameters are the activation values of the **i**-, **o**- and one of the outputnodes. The **i**- and **o**-node are the two nodes that receive their information from the outside world (see fig. 3.2). To get adequate behaviour of the robot, it has to be able to produce the right actions based on the information it gets from these two nodes. There are two actions the robot takes each cycle: changing the value of the left motor and changing the value of the right motor. This is represented by a sub-network, as shown in figure 1. This network has the shape of two perceptrons (a perceptron is a network that has two input nodes and one output node).

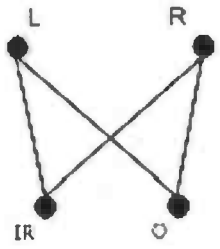


Fig. 1 Subnetwork

In table 1 different situations that can occur are stated and represented by positive or negative values of the two inputnodes **i** and **o**. On the right side of the table the correct action is stated represented by a positive or negative value of the outputnode **L**, which gives its activation to the left motor. In table 2 the same is done for the other sub-network in which **R** is the outputnode.

Situation	i	o	L	Action
food left	-	+	-	left motor slower, turn towards food
food right	+	+	+	left motor faster, turn towards food
poison left	-	-	+	left motor faster, turn away from poison
poison right	+	-	-	left motor slower, turn away from poison

Table 1

Situation	i	o	R	Action
food left	-	+	+	right motor faster, turn towards food
food right	+	+	-	right motor slower, turn towards food
poison left	-	-	-	right motor slower, turn away from poison
poison right	+	-	+	right motor faster, turn away from poison

Table 2

If we would create a hyperspace spanned by i and o, there would be two classes of points in that space; one class would be negative L and one would be positive L (see figure 2, in which the white points are positive L and the grey points are negative L). What a perceptron does is create a line in the hyperspace spanned by its inputnodes. So the subnetwork for L draws a line in figure 2. It is obvious that a line will never be able to separate the two classes.

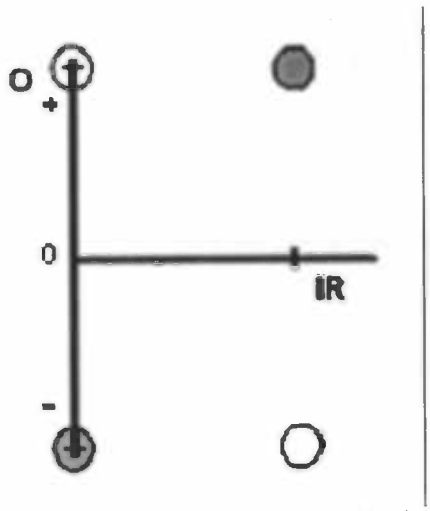


Fig. 2 Hyperspace spanned by i and o

A solution of this problem is adding an additional dimension to the hyperspace, making it three-dimensional. In that way it can become possible to separate the different classes with a single plane. One way of doing this is by adding an extra hidden layer to the network. But this also increases the number of weights in the network extensively and those weights are the parameters that are evolutionary optimized using the evolutionary algorithm. The more parameters have to be optimized, the slower the algorithm gets. Another solution is adding an extra node to the inputlayer that represents the multiplication of the i- and o-node. The activation of this node will add the extra information to the network necessary to separate the two classes. That is the reason for adding the x-node (see fig. 3.2).

Schemata

Artificial evolution of neural controllers uses strings of information given to the evolutionary operators. *Schemata* are similarity templates describing a subset of strings with similarities at certain string positions (Goldberg, 1989). If we take binary strings it will be easy to explain the effects of crossover on the composition of these strings. Lets say we have a population of binary strings of length 5. So one individual could look like (1,0,0,1,0). A schema (*,0,0,*,*) denotes the subset of strings that have a 0 on the second and third location. So (1,0,0,1,0) is a member of this subset, but so is (0,0,0,1,1).

Schemata are useful if we think that some locations in the string have a certain relation to each other, so that the values at those locations have a mutual influence. For instance, in the coding of parameters in the experiments, the weights from the neural network are represented in a chromosome string whereby weights of the connections from one input node to the two output nodes are adjacent in the string. It is clear that such a pair of weights embody the working of one such an input node.

The crossover operator has certain tendencies to prefer one search outcome over another. Such tendencies are called inductive biases. For 1-point crossover such a bias is *positional bias*. Positional bias characterizes how much the probability that a schema is transmitted intact during a crossover event depends on the relative position of the parts of the schema on the chromosome. It indicates which schemata are more likely to appear in new offspring. It may be shown that for 1-point crossover, the shorter the defining length of a schema, the more likely it will be preserved during crossover (defining length of a schema is the distance between the first and the last location within the schema; for instance the schema (*0,0,*,*) has defining length 1, while schema (1,*,*,*,0) has defining length 4. This last schema has a bigger chance of being cut by crossover the schema with defining

length of 1). So 1-point crossover has strong positional bias. One reason of encoding two related weights of connections from input to output nodes next to each other is to minimize the disturbing effect of 1-point crssover.

On the implementation of internal motivation

The implementation of internal motivation in the experiments took the form of a node in e neural network. One can question the value of such an implementation. What if a random generator could produce similar results? Would that not give doubt to the value of the more complex implementation used in the experiments? The question is if the direction in which the pro-active influence on behavior urges the behavior is important. As must be clear, the internal motivator urges the behavior in a pre-defined direction, determined by the weights of its connections to the output neurons. The random generator misses a pre-defined direction of stimulation. To investigate this, an additional experiment was done, in which the internal motivator is removed and replaced by two random generators, giving their activation values to the motor output values of the robot (figure 4). So now pro-activity means behavior based on an internal source of noisy activation. The model used in this experiment is shown in figure 3.

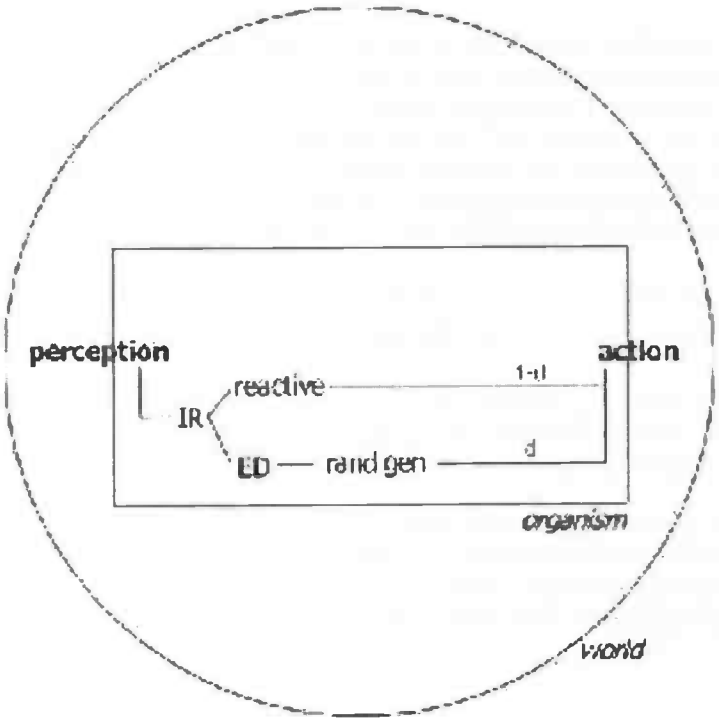


Fig. 3 Alternative model of behavior with a random generatot implementation of pro-activity

In figure 3 we can see that the only difference is the *form* in which the the pro-active influence on behavior comes. The rest of the model looks exactly like the one in the first experiment, shown in figure 3.12. Again, the level of noise is influenced in the same way as the activation level of the internal motivator in the previous experiments. That is, the external dynamics plays a similar role.

An alternative network

In the experiment an alternative network was used. The pro-activity comes in the shape of a *random generator*. This random generator creates random values in the range of [-1.0,1.0], which is the range of the activation values of the neurons in the network. In figure 4 the architecture is shown.

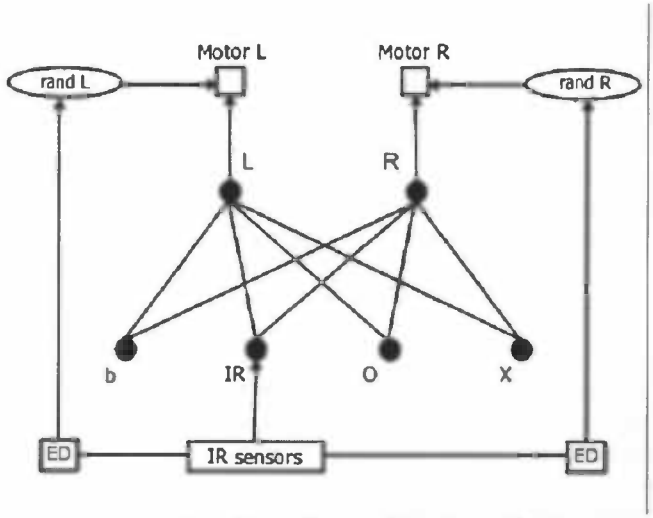


Fig. 4 The alternative architecture with the random generators

The neural network functions in exactly the same way as in the architecture in the main three experiments. The difference lies in the random generators. The values created by the random generators is bounded by the external dynamics. The higher the **ed** value is, the smaller the values created by the random generator will be. The motors of the robot receive a weighted combination of the output activations of the neural network and the values created by the random generators. The parameter controlling this trade-off is the **d**-parameter explained in chapter 2.

The values of the motors of the robot are calculated by:

$$\text{rand} = 0.5 + (rand(-1,1) * e^{-ed}) / 2 \qquad (5.1)$$

$$\text{motor} = ((1-\mathbf{d}) * o_{net} + \mathbf{d} * \text{rand}) * 20 - 10 \qquad (5.2)$$

in which rand is the value created by the random generator and motor is the value given to one of the two motors of the robot. *Rand*(x,y) is a function that gives a random value lying between x and y. *ed* is a value in which the external dynamics are incorporated (see (3.5)). The value o_{net} is the output value of the network.

In figure 5 the values of the best and average fitness of populations from the extra experiment are shown. The dynamics parameter d determined the balance between the reactive influence (the influnce from the network) and the pro-active influence (the random generator). The values of the best and average fitness are average value taken over nine different starting populations.

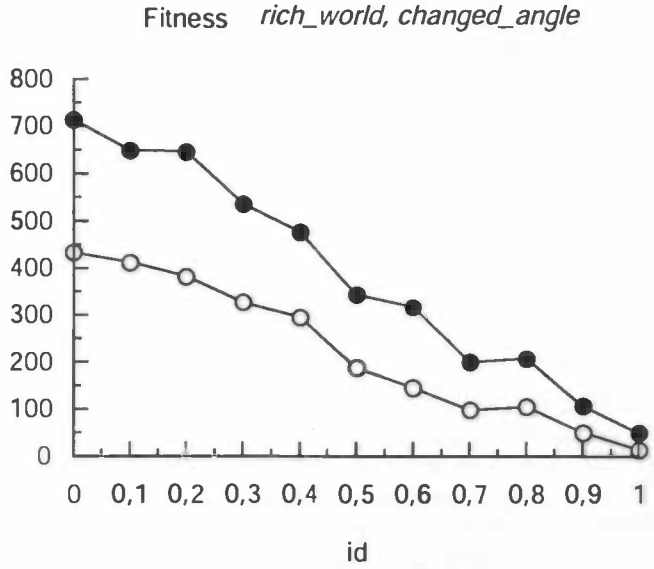


Fig. 5 Fitness of behavior evolved with the random generatot implementation of pro-activity

When we compare these results with those in the first graph in figure 4.3, in which results are shown for the same setting, but with the internal motivator modelled by the node in the network, we see that performance for the latter is much better. This result shows that the use of the motivator as a node in the network in experiment 2 is functional. When it would not be, performance would be ignorant of the use of such an internal motivator and performance would be as good as when using a random generator as a model for internal motivation.

Internal dynamics experiment

In the model of behavior given in chapter 2 two main processes were distinguished: the reactive part and the pro-active part. The internal source of activation that produces the pro-active behavior can be more or less active. It has a certain degree of dynamics. The parameter that controls the amount of dynamics that is allowed for the pro-active part of the controller is the *internal dynamics parameter* or id-parameter. The higher this parameter, the more dynamic the pro-active part of the behavior forming process can be. This experiment addresses the question about the speed with which the agent can adjust its level of pro-activity, or, the level of influence from its internal motivation.

In the experiment populations of agents are evolved with different values of the id-parameter. At the end of this evolutionary process performance is evaluated in relation to the different id-values. This will produce a picture of the relation between the amount of pro-active dynamics of the agent and the quality of behavior. The question will be if certain amounts of pro-activity will optimize the behavior. In figure 6 the model of behavior that is used in this experiment is depicted.

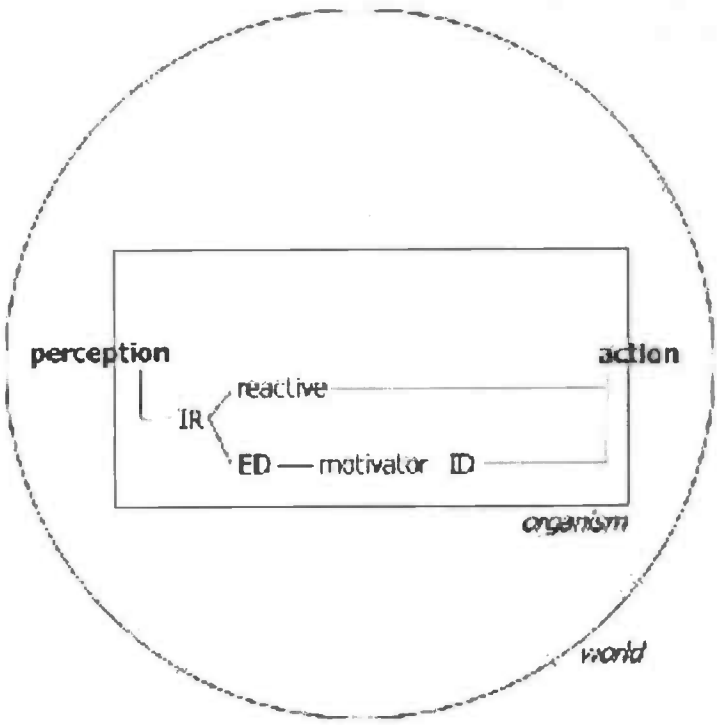


Fig. 6 Model of behavior

In this model **IR** stand for the infra-red readings taken from the robots sensors. These readings are used as input for the reactive part of the network. And they are used to create a measure for the external dynamics **ed**. The **ed** value influences the activation value of the motivator. And it is the motivator that has a certain value of internal dynamics **id**, which is the focus here. The rest of the model was already explained in chapter 2, so **reactive** stands for the reactive process, and thus, for the reactive part of the neural network (see 3.2).

For each value of the id-parameter 9 different populations are evolved. Each population carries 50 individuals. Each individual is 'downloaded' onto the robot and is allowed to 'live' for 1500 cycles (MAX_AGE) after which its fitness is calculated. Its initial battery value (MAX_BAT) is 10000. The robot is reset after each life and starts at a different starting angle according to the *angle* setting that was used. This is repeated 9 times and the average fitness is calculated. The population converges when the sum of the variances of the weights within the population drops below a certain threshold value. For each population the mean and best fitness values are recorded after each generation.

The 9 starting populations are first evolved in the default world, *rich_world*. After that the same starting populations are evolved in *poor_world*. A third time the populations are evolved in *rich_world* again but this time the angle of the agent at the moment of death is taken as the starting angle for its successive life. This third setting will allow for more use of any structure present in the environment.

So there are three environmental settings:

setting	world	angle
1	*rich*	*changed*
2	*poor*	*changed*
3	*rich*	*passed*

At the end we can look at the fitness values plotted against the id-values and see if some trend has emerged. The fitness results for the different environmental settings will be compared.

In figure 7 the average and best fitness of the final populations are given. The first graph shows the results of the populations evolved in *rich_world*. The second graph shows results from *poor_world*. And the third graph shows results in *rich_world* with the *passed_angle*.

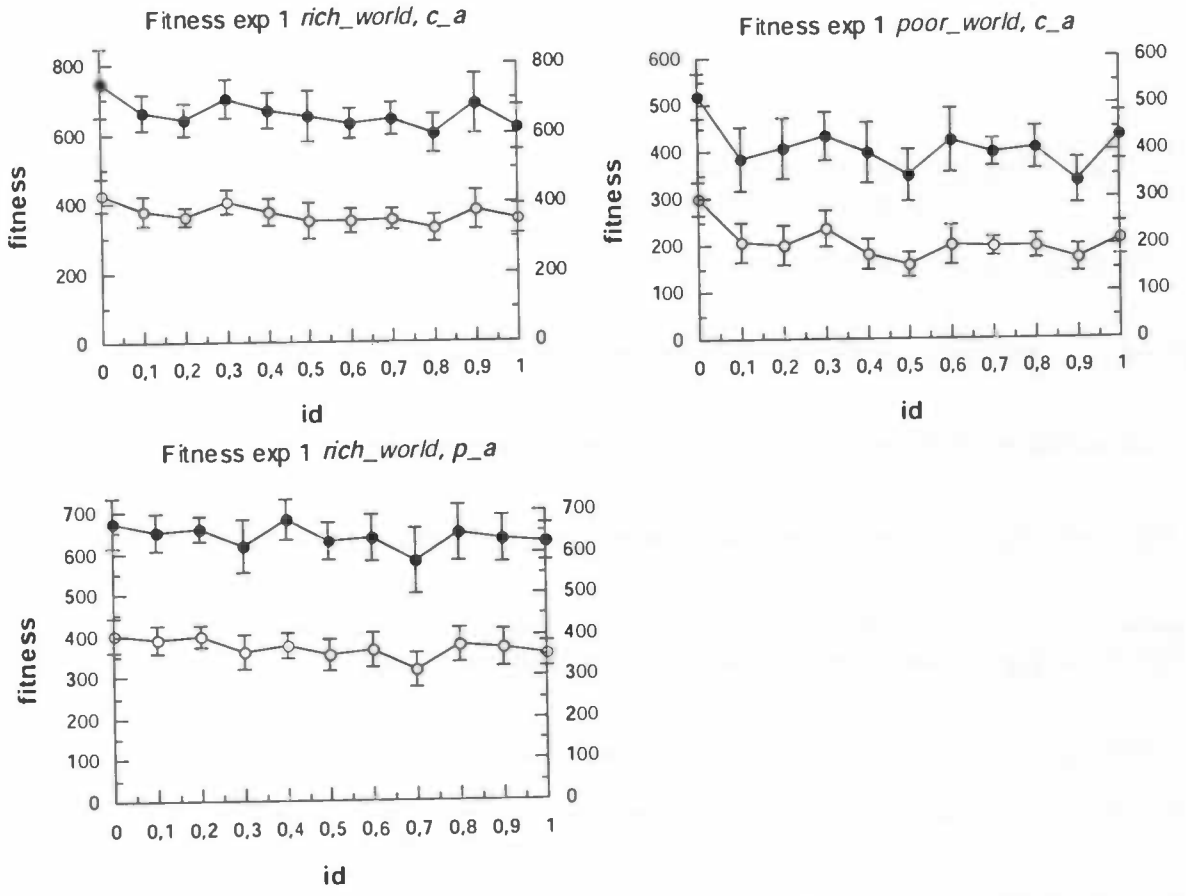


Fig. 7 Fitness results for different levels of internal dynamics

The x-axis shows the different id-values at which the starting populations were evolved. The white data point show the *average* fitness and the black data point show the fitness of the *best* individual. All values were averaged over nine populations.

We can see that as the id-value increases, which means that the agent will be able to change the activation of its internal motivator more quickly, the fitness drops slightly. This suggests that the effect of the internal motivator on the quality of behavior is minimal and even slightly negative. The results suggest that the speed of change has no real influence, since fitness values for populations evolved with different id-values do not vary much. Apparently, the internal motivator has only small influence on the quality of behavior. As a result, changing the speed with which the internal motivator can alter its activation will not have great impact either.

References

Anderson, E. J., and Ferris, M. C., 1994, *Genetic Algorithms for Combinatorial Optimization: The Assembly Line Balancing Problem*, In: ORSA Journal on Computing, vol 6, pp. 161-173

Belew, Richard K.; Mitchell, Melanie, 1996, *Adaptive individuals in evolving populations : models and algorithms*, Reading, Mass., Addison-Wesley

Capra, Fritjof, 1996, *The web of life : a new scientific understanding of living systems*, New York, Anchor Books

Cecconi, F., Parisi, D., 1993, *Neural networks with motivational units*, In: Meyer, Roitblat, Wilson, From Animals to Animats II, pp. 346-355

Clark, Andy, 1997, *Being there : putting brain, body, and world together again*, Cambridge, Mass., MIT Press

Dautenhahn, K., 1999, *Investigations into Internal and External Aspects of DynamicAgent-Environment Couplings*, In: Tschacher, Dauwalder, Dynamics, Synergetics, Autonomous Agents, Singapore, World Scientific, pp. 207-223

Eiben, A. E., Hinterding, R., and Michalewicz, Z., 1999, *Parameter Control in Evolutionary Algorithms*

Fletcher, J., Zwick, M., Bedau, M. A., 1996, *Dependence of adaptability on environmental structure in a simple evolutionary model*, In: Adaptive Behavior, vol 4, pp. 275-307

Floreano, Dario; Mondada, Francesco, 1996, *Evolution of Homing Navigation in e real Mobile Robot*, In: IEEE Transactions on Systems, Man, and Cybernetics--Part B: Cybernetics, vol 26, issue 3, pp. 396-407

Floreano, Dario; Mondada, Francesco, 1998, *Evolutionary Neurocontrollers for Autonomous Mobile Robots*, In: Neural Networks, vol 11, pp. 1461-1478

Franklin, Stan, 1995, *Artificial minds*, Cambridge, Mass., MIT Press

Garner, Wendell R., 1962, *Uncertainty and Structure as Psychological Concepts*, John Wiley and Sons, New York

Haykin, Simon, 1994, *Neural Networks, A Comprehensive Foundation*, Prentice-Hall

Jacobi, N.; Husbands, P.; Harvey, I., 1995, *Noise and the Reality Gap: The Use of Simulation in Evolutionary Robotics*, In: ECAL 1995, pp. 704-720

Lund, H. H., 1995, *Evolving Robot Control Systems*, In: Alander, J. T., Proceedings of 1NWGA, University of Vaasa, Vaasa

Lund, H. H., Miglino, O., 1998, *Evolving and Breeding Robots*, In: Proceedings of the First European Workshop on Evolutionary Robotics, Springer-Verlag

Menczer, F., Belew, R. K., 1996, *From complex environments to complex behaviors*, In: Adaptive Behavior, vol 4:3-4, issue Winter-Spring

Michel, O.,1996, *Khepera Simulator 2.0 User Manual*, http://diwww.epfl.ch/lami/team/michel/khep-sim/index.html

Nolfi, S.; Elman, J.L.; Parisi, D., 1994, *Learning and evolution in neural networks*, In: Adaptive Behavior, vol 3, issue1, pp. 5-28

Nolfi S. & Miglino O. (in press). Studying the emergence of grounded representations: exploiting the power and the limits of sensory-motor coordination. In G. Mulhauser (Ed.), *Evolving Consciousness*, Advances in Consciousness Research Series, John Benjamins, Amsterdam

Nolfi, S.; Parisi, D., 1995, *Genotypes for neural networks*, In: Arbib, A., Handbook of brain theory and neural networks, Cambridge, MA, MIT Press, pp. 431-434

Nolfi S. & Parisi D., 1993, *Self-selection of input stimuli for improving performance*, In: G. A. Bekey, Neural Networks and Robotics, Kluwer Academic Publisher, pp.403-418

Nolfi, S., 1996, *Adaptation as a more powerful tool than decomposition and integration*, In: Fogarty, T.; Venturini, G., Proceedings of the Workshop on Evolutionary Computing and Machine Learning, University of Bari, Italy

Nolfi, S., 1999, *How learning and evolution interact: The case of a learning task which differs from the evolutionary task.*, In: Adaptive Behavior, vol 7, issue 2, pp. 231-236

Nolfi, S.; Parisi, D., 1999, *Exploiting the power of sensory-motor coordination*, In: D. Floreano, J-D. Nicoud, F. Mondada, Advances in Artificial Life, Proceedings of the Fifth European Conference on Artificial Life, Berlin, Springer Verlag, pp. 173-182

Nolfi, S., in press, *Power and limits of reactive agents.*, to appear in: Neurocomputing

Pfeifer, Rolf, 1996, *Building Fungus Eaters: Design Principles of Autonomous Agents*, In: Maes, P. et al., From Animals to Animats 4, Cambridge, MA, MIT Press

Pfeifer, Rolf,2000, *From animals to animats 6 : proceedings of the Sixth International Conference on Simulation of Adaptive Behavior*, Cambridge, Mass., MIT Press

Prem, E.,1996, *Motivation, Emotion and the Role of Functional Circuits in Autonomous Agent Design Methodology*, Wien, Austrian Research Institute for Artificial Intelligence, Tech. Rep. 04-96

Ross Ashby, W., 1956, *An Introduction to Cybernetics*, London/New York, Methuen

Scheier, Christian; Pfeifer, Rolf, 1995, *Classification as Sensory-Motor Coordination*, http://www.ifi.unizh.ch/ailab/publications/1995.html

Seth, A. K., 1998, *The Evolution of Complexity and the Value of Variability*, In: Adami, C., Belew, R., Kitano, H., Taylor, C., Artificial Life VI: Proceedings of the Sixth International Conference on the Simulation and Synthesis of Living Systems, Cambridge, MA, MIT Press, pp. 209-221

Sloman, A., 1987, *Motives Mechanisms and Emotions*, In: Emotion and Cognition 1,3, pp. 217-234

Todd, P. M., Wilson, S. W., 1993, *Environment structure and adaptive behavior from the ground up*, In: Meyer, Roitblat, Wilson, From Animals to Animats 2: Proceedings of the Second International Conference on Simulation of Adaptive Behavior, Cambridge, MA, MIT Press/Bradford Books, pp. 11-20

Uexküll, J. von, 1982, *The theory of meaning*, In: Semiotica, vol 42, issue 1, pp. 25-82

Varela, Francisco J.; Thompson, Evan; Rosch, Eleanor; 1993, *The embodied mind, cognitive science and human experience*, Cambridge, Mass., MIT Press

Velásquez, J., 1998, *When Robots Weep: Emotional Memories and Decision-Making*, In: Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98), Madison, MIT/AAAI Press

Ventura, R., 2000, *Emotion-based agents*, Instituto Superior Técnico, Lisbon, Technical University, Master's thesis

Whitley, D., 1993, *A genetic algorithm tutorial*, Fort Collins, Colorado State University, pp. 93-103, March, Tech. Rep. CS

Wooldridge, M. J., and Jennings, N. R., 1995, *Intelligent agents:Theory and practice*, In: Knowledge Engineering Review, vol 10, issue 2, pp. 115-152