NATURAL INTELLIGENCE FOR AUTONOMOUS AGENTS

Christian Balkenius

Lund University Cognitive Science
Kungshuset, Lundagård, S–222 22 LUND, Sweden
Christian.Balkenius@fil.lu.se

Abstract: The paper presents a general architecture for behaviour based control systems for autonomous agents. A number of architectural principles are proposed which make it possible to combine reactive control with learning and problem solving in a coherent way. In particular, I investigate the interaction between reinforcement learning, internal world models and dynamic action selection as well as a number of connections to psychological models and biological systems.

1. INTRODUCTION

Traditional computer systems, and artificial intelligence systems in particular, have been built as symbol processing automata where reasoning is implemented as search in a knowledge base. In an attempt to build a control system for autonomous agents, we have been investigating entirely different principles. The general architecture of this agent has been outlined in Gärdenfors and Balkenius (1993). The system is based on a set of highly structured neural network modules. Some of these modules have already been constructed while others are still under development. So far, we have designed modules for motivation and behavioural selection (Balkenius 1993), simple place recognition based on smell cues (see below), perceptual schema formation based on categorization and association (Balkenius 1994), reinforcement learning and self-organizing cognitive maps. Some recent progress concerning reactive problem solving that depends on the interaction between a reactive control system, motivation and reinforcement learning is reported below together with the overall architecture of the system. The main purpose of this paper is to promote a number of general architectural principles for the design of autonomous agents. I will also give an example of a simple but complete agent constructed according to these principles.

It is our goal to design a physical autonomous robot with the type of control system that I described below. Currently, we are working on algorithms for visual inputs (Pallbo 1992, 1993). In the extension, the agent will be guided primarily by vision. Up to this point, the perceptual input used in computer simulations of the agent architecture has either been entirely faked (BERRY II) or has consisted of simulated olfactory and tactile information (BERRY III).

2. AUTONOMOUS AGENTS

In recent years, research directions have emerged that are based on new assumptions about the architecture needed for intelligence. They all share some common properties and "the emphasis in these architectures is on more direct coupling of perception to action, distributedness and decentralisation, dynamic interaction with the environment and intrinsic mechanisms to cope with resource limitations and incomplete knowledge" (Maes 1990).

These approaches aim at natural intelligence, rather than artificial, as they are based on, or at least inspired by, biology. The most important aspect of such architectures is the emphasis on complete creatures or systems that let us make observations that cannot be made from studies of isolated modules (Brooks 1986, 1991a, 1991b). Indeed, the goal of this paper is to present a simple, but complete, agent architecture. We will see a number of interesting properties emerging as a result of the interaction between a number of very simple processes. In constructing this architecture, I have been inspired mainly by three areas of current research.

◊ **The subsumption architecture** Subsumption is a computational model that is based on a network of asynchronously computing elements in a fixed topology. The active elements communicate with each other and with sensors and effectors by

sending and receiving messages without implicit semantics. The meanings of the messages are given by the operations of both the sender and the receiver (Brooks 1991b). Typically, the messages are constrained to be very small values represented in a low number of bits. The communication rate is usually very low, in the order of a few messages every second. This assures that robots built using the subsumption architecture can be controlled by existing and cheap hardware (Horswill 1993). Although this has not been its primary goal, the subsumption architecture shows a number of parallels to models in ethology.

◊ **Neural networks** Even more biologically realistic control mechanisms can be constructed by using neural networks. Architectures of this kind are similar to the subsumption architecture in that they consist of a number of interacting computing units. However, these are usually much simpler than the computing elements in a subsumption architecture. The units are assumed to imitate neurons in the brain. In most cases, however, the similarities with real neurons should not be overstated. The main use of neural networks in technical applications is based on their learning abilities and not their similarity to biological neural systems. This is one of the reasons why neural network research will prove to be important for the construction of autonomous agents. Another reason for using neural networks is that they are, at least in principle, very fast when implemented in parallel hardware. When a neural network is used for reactive control, the calculations can, more often than not, be made in a feed-forward manner. The architecture presented below is based on neural networks. In what follows, however, I will here only describe the mechanisms at a computational level (*cf.* Marr 1982).

◊ **Reinforcement learning** In a typical control situation, it is possible to measure the error in the control scheme, but it is not always possible to work out how to change the parameters of the controlling mechanism in order to improve the control scheme. This is one area where reinforcement learning can be used. This type of learning is based on a reinforcement signal that tells the system how well it is doing (see, for example Baird and Klopf 1993). Using this signal, the system adjusts its parameters in order to maximize the expected reinforcement signal (Barto, *et al.* 1983, Watkins 1992). This, again, is a mechanism that can be found in real animals where it shows up as instrumental or operant learning, *i.e.* learning by reward or punishment (*cf.* Lieberman 1990 and Mackintosh 1983).

These three areas are, of course, not orthogonal. For example, it is quite possible to use neural networks to build a subsumption style control system for a robot. There also exist a number of neural network models that perform reinforcement learning (*e.g.*, Millán and Torras 1992, Williams 1992).

3. Architectural Principles

The architecture is based on a number of organisational principles that I have tailored to fit my needs. These principles are behaviour-based control, subsumption, parallel engagements, central behaviour selection and a functionally layered architecture. I believe that it is possible to construct autonomous agents with quite powerful abilities using these principles. The architecture presented below is an example of this type of agent. Most of its abilities result from a combination of these architectural principles. By making the modules operate on different types of sensory data, the same overall type of architecture can be used for a number of applications.

The control system of the agent is based on behaviours. A behaviour[1] is a subsystem that is responsible for one specific coupling between sensors and actuators (figure 1). This contrasts sharply with the view of traditional AI where control is typically based on a set of goals, a model of the world and a search procedure. The search procedure tries to find an action sequence that changes the state of the world to the desired goal state. If such an action sequence is found, it will be executed in the real world. In a behaviour-based agent, the goal need not be explicitly represented. Instead, behaviours are selected on an immediate sensory basis in such a way that they are likely to move the agent closer to the goal in the real world. Problems are avoided when they occur. As we will see, a behaviour-based agent can be augmented with explicit goal representations and planning, but such abilities are not part of its primary repertoire.

Figure 1. A behaviour is defined as a connection between sensors and actuators.

[1]The term *behaviour* is used here to denote the system internal to the agent that is responsible for the externally observed behaviour. This terminology is admittedly a little confusing but is consistent with the general use of this term in behaviour based robotics.

3.2 Subsumption

The second principle is borrowed from the subsumption paradigm (Brooks 1991a, 1991b). A number of layers in the architecture take care of different behaviours. The lower layers control the basic behaviours of the agent (figure 2). A typical low level behaviour in a subsumption style robot includes activities such as object avoidance, wandering and rudimentary exploration. On a higher level we may find processes like object identification and planning. Each higher layer is able to monitor and control the underlying layers. While traditional systems can be said to be vertically decomposed into processing stages, a system of the present type is horizontally decomposed into behaviours (Schnepf 1991). Because the complexity of the example system described below is kept as low as possible, the subsumption architecture is not used to the extent that it would have been in a more complex agent.

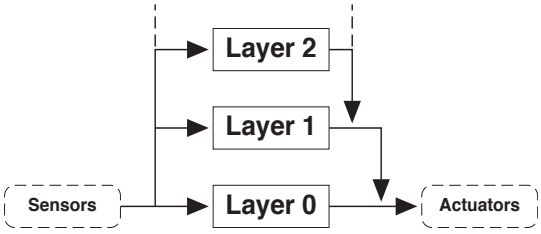

Figure 2. A hierarchy of behaviours.

From ethology we borrow the distinction between approach and consummatory behaviour. Most parallel engagements can be divided into these two components. For the eating behaviour, the approach behaviour consists of searching for or collecting food, while the consummatory behaviour corresponds to the actual eating of the food. For a dish-washing robot, clearing the table can be considered the approach behaviour while washing the dishes is the consummatory behaviour. The distinction between the two is that the first behaviour is instrumental in achieving the second. In most cases, the approach and the consummatory behaviours can be organized in a subsumption style hierarchy (Figure 3). This type of structure can be called an appetence module.

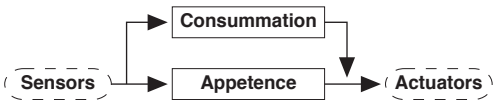

Figure 3. An appetence module consists of approach and consummation

3.3 Parallel Engagements

Different engagements are controlled by parallel modules. In an artificial creature, behaviours such as eating and sleeping may be implemented as parallel engagements. In practical applications of the architecture, vacuum cleaning and dishwashing may constitute parallel engagements. Each engagement is controlled by its own subsumption hierarchy as shown in figure 4.

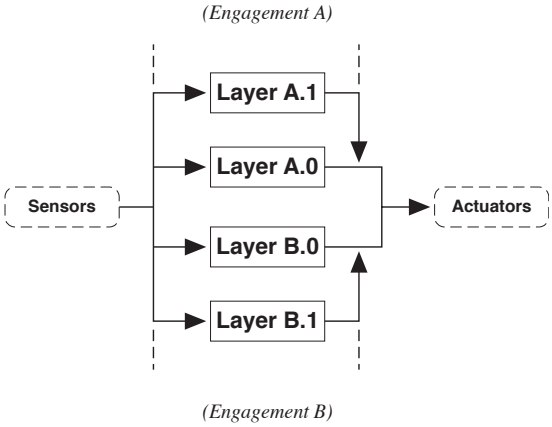

Figure 4. Each engagement has its own behavioural hierarchy.

3.4 Central Behaviour Selection[2]

In two recent papers, I have argued that central control is necessary for complex cognitive operations (Balkenius 1993, 1994). This is in contrast to the view of Brooks (1991), Maes (1991) and others who argue that central control is neither necessary nor suitable for autonomous agents. As the control system of an agent grows larger and more complex, the probability of different behaviours interfering with each other increases rapidly. When this occurs, the existence of a central control module that is able to shut off some behavioural modules and activate others is needed to eliminate this problem. However, this central control module is functional rather than physical. It need not exist in a physically defined place in the architecture. It is quite possible that this central control module is the result of interaction among various behavioural modules. It is important not to misunderstand the principle of central behaviour selection. It simply states that all behaviours cannot be executed at the same time and

[2] A more appropriate name would be "Central Engagement Selection" but the current name is already established in the field.

that the choice of behaviour cannot be made locally. One part of the agent cannot decide to search for food while an other decides to dance. However, most behaviours, and especially parts of behaviours, are best handled in a distributed fashion. Figure 5 shows an architecture where two engagements A and B centrally compete for activation. In section 11, we will come back to this mechanism.

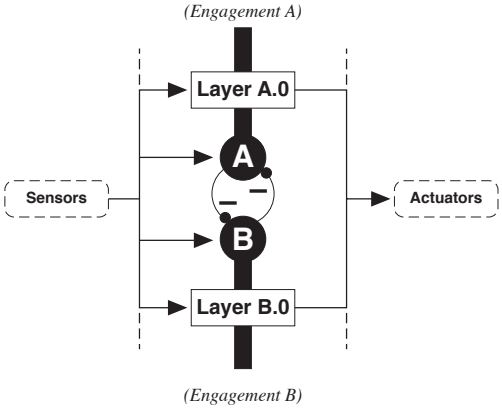

Figure 5. A behavioural hierarchy is selected centrally.

3.5 Functionally Layered Architecture

Finally, the architecture will be organized in a number of layers. This is subsumption on a macro-level. While a subsumption architecture controls the different parts of a behaviour, the layered architecture on a larger scale adds general functionality to the agent. An important feature of the layered architecture is that the agent can operate without the higher layers. They add functionality, but they are never necessary.

For example, the lower layers control the fundamental reactive behaviour. This system equips the agent with a set of elementary abilities that are used as a basis for more complex behaviours.

An intermediate layer may control behaviour based on expected rewards as used in reinforcement learning (Klopf and Morgan 1990, Sutton and Barto 1990, Barto *et al*. 1990, Watkins 1992). Another role of this layer is to chunk actions together into more manageable routines. This kind of mechanism is well handled by traditional approaches (*e.g*. Newell 1990).

The role of the top layer can be to learn about the consequences in the environment of various actions. This knowledge can later be used as an internal environment where actions can be tested before they are confronted with the unforgiving external environment. This layer may also be used for self-supervised learning within the lower layers. Planning and problem solving are instances of this type of process as are daydreaming and worrying. The upper two layers are similar to the DYNA architecture proposed by Sutton (1992). The final layer plays a role similar to the world model in traditional AI systems. Note however that this model is never essential for the behaviour of the agent. If the agent has a perfect model of its environment, then its planning will be perfect, if it has not, the lower layers will let it manage anyway.

Figure 6 shows an architecture with three functional layers, each of which communicates reciprocally with a motivational system (see section 11).

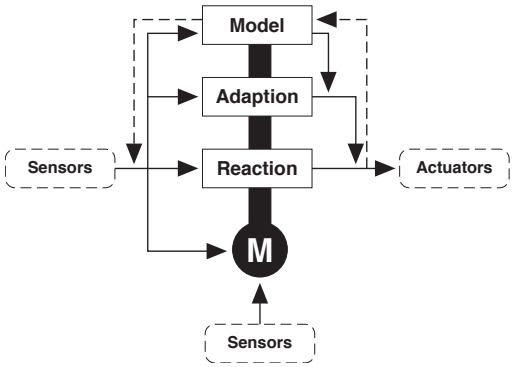

Figure 6. A functional hierarchy.

4. An Example Domain

In 1985, Wilson introduced the so called *animat approach* to intelligence. This approach addresses the problem of a complete artificial animal, or animat, that has to survive in its environment (Wilson 1985, 1987). In this presentation, a two-dimensional spatial domain will be used as an example. The tasks we will consider all fall under the animat approach. A hypothetical creature is situated in a world where it has to fulfill a number of competing needs in order to survive. I have performed a number of computer simulations of a vehicle type creature in a world that consists of a few simple objects (*cf*. Braitenberg 1984).

◊ **Walls** The walls are used to build the environment of the agent. Walls can be recognized by the agent when they touch its whiskers. One important feature of the spatial problems that can be constructed using walls is that we can represent very complex graphs as a complex maze. If the graph we want to represent cannot be represented on a two-dimensional surface, *teleporters* can be added that can make the graph arbitrarily complex. Since search is what

traditional AI is all about, any such problem can be converted into a maze in this way.

◊ **Appetitive Stimuli** When the simulated agent approaches an appetitive stimulus, a consummatory behaviour will be activated. The stimuli are identified by olfactory cues. Different stimuli can have the same or different scents. In general, some reward signal is generated internally by the agent as it reaches and consumes an appetitive stimulus.

◊ **Aversive Stimuli** The second type of stimuli are aversive. They are identical to appetitive stimuli in all respects except that they generate a shock signal to the agent. However, there is no intrinsic difference between rewarding and shocking stimuli, except that the rewarding stimuli increase the overall performance measure of the agent while the aversive decreases it. Any difference in behaviour towards appetitive and aversive stimuli depends on the internal structure of the agent and not on the stimuli themselves.

◊ **Other Agents** The simulated world is a multiagent environment. In principle, the different agents should be able to compete or cooperate. Some very rudimentary examples of involuntary cooperation have been observed during simulations, but a discussion of this phenomenon is beyond the scope of the current presentation.

The basic vehicle uses a very simple reactive control scheme to move around in its world. The simplicity of this control system makes it an ideal starting point for the incremental design of autonomous agents. It has previously been used in a number of studies (*e.g.* Dumeur 1991) and in education (Donnet and Smithers 1991). Since the sensors of the vehicles are most appropriately understood as olfactory, the stimuli in our simulated world are sending out scent signals. In summary, the simulated agent consists of the following components.

◊ **Olfactory sensors** A number of smell sensors are placed in pairs symmetrically along the front of the agent. Each pair of sensors is sensitive to a certain smell signal. By comparing the intensity of the different smells, it is possible to calculate the direction of its source. Olfactory input can also be used to identify places according to their smell (see Appendix C).

◊ **Whiskers** The agent is also equipped with two whiskers in the front. They react on contact with walls or other creatures. Their primary use is to make the agent follow walls instead of trying to go straight through them. Sequences of tactile input can also serve to identify places with some certainty. However, the tacitly generated place representation does depend on how the agent got to the current place.

◊ **Retina** I am planning to use visual input for object avoidance and place recognition. For this purpose, a retina is positioned in the front of the body. The image on the retina is constructed by ray-tracing the simulated world model. Since the retina is fixed in relation to the body, the vehicle has to turn back and forth in order to scan its environment. So far, visual input has not been used in the simulations. I plan to substitute the basic olfactory cues with visual ones as soon as the visual module has been further developed.

◊ **Motors** There is a motor on each side of the creature. Each motor drives a wheel on its side of the body. By varying the speeds of the motors, the agent can go forwards, backwards or turn in any direction. This is the simplest way to get around on a two dimensional surface.

◊ **Control system** Finally, and most important, the sensors are connected with the two motors through the control system of the agent. The role of this system is to let the agent do the right thing at the right time. Indeed, it is necessary for the agent to do anything at all. This system is the topic of the remainder of this paper.

It has been argued (Brooks 1986) that a simulated world should not be used since it is so easy to ignore important aspects of the real world. Instead one should opt for simple behaviours in a real robot in the real world. In principle, I agree with this position. The ultimate test for an agent architecture is in the real world. I believe, however, that the development of algorithms can be made in a quite different time scale if one is allowed to do computer simulations also. But it is important to remember that the simulated architecture is intended for a physical robot. Until it has been tested in the real world, one cannot be sure that it will work under more realistic circumstances.

5. Reactive Control and Goal Gradients

The first layer of the architecture is responsible for reactive control, *i.e.*, control that has its origin in the current sensory input of the system. The simplest type of reactive control for an autonomous agent is that of the basic vehicle as described in Braitenberg (Braitenberg 1984). Let us assume the smell sensors on one side of the body increase the speed of the motor on the other side when the smell intensity increases. This arrangement will make the agent turn towards the source of the smell. This is the simplest possible appetence behaviour. By varying the weights of the connections from the sensors to the motors, the agent

can be made to approach or withdraw from stimuli in various ways.

There are three aspects of behaviour that can be modelled in this manner: (1) approach or avoidance, (2) speeding up or speeding down, (3) turning or moving. The first two aspects are obvious, the third is the difference between increasing the speed of one motor to turn or turning by increasing the speed on one motor and decreasing it on the other, thereby making the speed of the agent constant. Figure 7 shows the different combinations of these aspects and the different paths taken close to a stimulus by the agent.

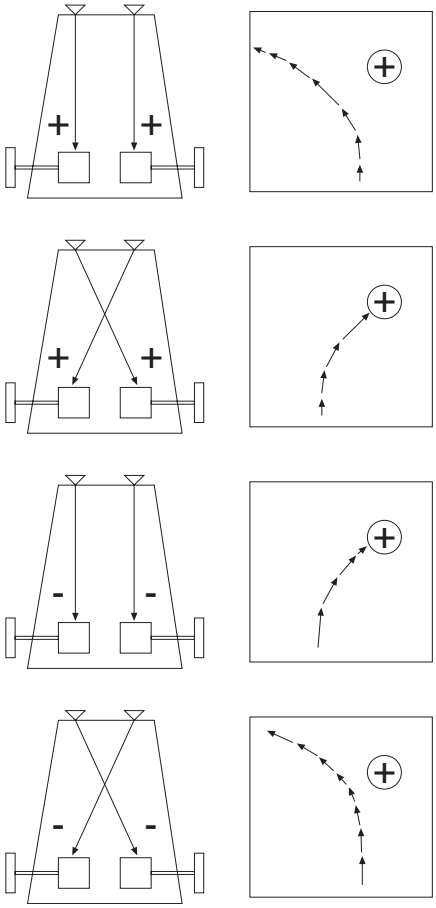

Figure 7. Different vehicle configurations and their behaviours.

It is useful to consider the environment as a vector field. The appetitive stimuli have force vectors pointing towards them while the aversive stimuli have force vectors pointing away from them. The strengths of the forces may either increase or decrease as we get closer to the stimulus. Different motivational factors may also influence the current force field. Arkin (1990) has used this type of force fields to represent motor schemes. Figure 8 presents an example of the force field in an environment with two stimuli. To establish the force fields in an arbitrary environment the agent is placed in every possible point in space and its direction and speed of movement are recorded (*cf*. Arbib 1987).

The use of force fields can be readily generalized to most situations where there is a tendency for someone to either approach or avoid some state of the world. In fact, in the 1930s, the psychologist Kurt Lewin developed a theory for all kinds of human behaviour based on this notion (Lewin 1935, 1936). Unfortunately, his work in this area has mainly been forgotten while his work on social psychology has become standard literature in the field.

By integrating the force field we get something that we may call the rewarding potential of each point in space. The position of an appetitive stimulus may for example have the potential +1 while the position of an aversive stimulus may have a potential of –1. If we call the appetitive stimulus a goal, we may call the forces goal gradients and we are back into psychology again where the use of goal gradients in the explanation of behaviour can be traced back to the works of C. L. Hull (1938), another psychologist[3] whose work has had an impact on psychology that is second only to that of Freud's. In the more general case, we may talk of approach and avoidance gradients generated by positive and negative potentials (*i.e*. reward or punishment).

I will take the extreme view that all problems of behaviour can be reduced to the construction of the appropriate goal gradients. I will use this assumption throughout this paper since it makes it possible to describe both reactive control and various forms of learning and adaptation in a unified way. Learning can be seen as a process for construction of the appropriate goal gradients. Figure 9 shows the potentials on a one dimensional path between two stimuli.

In real animals, there exists an interesting asymmetry between positive and negative potentials. The aversion gradient is typically much steeper that the appetitive. This is rather natural from a biological point of view. It is useful to approach an appetitive stimulus such as food from a very long distance while aversive stimuli do not have to be avoided until the animal is rather close to them. This asymmetry is illustrated in figure 9. In most cases, one would like this asymmetry to carry over to technical applications.

There can be two sources of potentials and gradients.

◊ **External** When the force field is externally generated, it is the result of what I call immediate perception, *i.e*. the current sensory signals are used to build the force field. The path taken by the agents

[3]Hull had an engineering background and was also much concerned with robotics.

in figure 7 is the result of an externally generated force field.

◊ **Internal** When a stimulus cannot be directly observed, it is still possible for the agent to approach or avoid it if it can construct a force field internally. If the agent has learned that the goal is just around the corner, this information can be used to generate an internal potential that makes the agent go the correct way. As we will see below, we may interpret the role of reinforcement learning as a generator of such internal goal gradients.

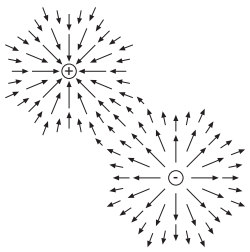

Figure 8. Force-field around two stimuli

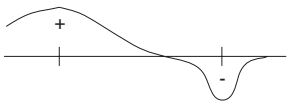

Figure 9. Potential along a path between to stimuli

In our simulated agent, olfaction was used to generate the external force field. Again, I hope to be able to substitute vision for the olfactory sensors later on. In a visually guided agent, desired objects would generate a positive potential while obstacles and walls would generate negative potentials. By viewing sensory information as force fields it is easy to see what happens if we integrate information from several different sensory modalities. In the basic case, the combined behaviour is simply the behaviour generated by the sum of the potentials for each individual modality. A mathematical formulation of the smell oriented reactive agent can be found in the appendices.

6. Reactive Adaptation

The control system described above assumes that the agent already knows which stimuli are appetitive or aversive. This knowledge is reflected in the weights of the connections from sensors to motors. In many cases, one does not know beforehand which stimuli are aversive and which are appetitive. It is also possible that different stimuli change potentials, for example from positive to negative. In these cases, it is useful for the agent to adapt to the new environment.

Let us assume that a positive reinforcement signal is associated with each type of consummatory behaviour. Negative stimuli such as shocks give rise to negative reinforcement signals. We want the agent to approach stimuli that are rewarding and to avoid others. How can we devise a learning rule that would accomplish this?

First we must observe that the only time the agent can learn about the rewarding properties of a stimulus is when it is in contact with it. This implies that the agent must approach all stimuli initially to find out about potential rewards. In other words, the agents should be set up with connection from sensors to effectors that makes it approach all stimuli. This tendency for initial curiosity will gradually go away as the agent learns about the rewarding or punishing properties of different stimuli. Some authors have argued that there is even a special part of the brain concerned with this type of initial curiosity, namely the septo-hippocampal system (*e.g.* Gray 1982)

The weights of the control system must change in a way that makes the agent approach the appetitive stimuli and avoid the aversive. Recall from section 5 that there are a number of ways to approach and avoid objects. This implies that we may either change the connections that cross the centre of the body or the connections that stay on the same side. It is also possible to change all four types of connections to get a number of interesting behaviours.

A learning rule that only changes the crossing connections will make the agent speed up when it approaches an appetitive object but slow down when it avoids an object. A learning rule that operates on the other connections will have the opposite effect. By introducing nonlinearities in the control scheme, the possible complexity becomes very large (Braitenberg 1984).

It is important to realize that there is an intrinsic asymmetry between reward and punishment. If the agent is rewarded for approaching some object, it is likely that it will approach it again. In the long run, the information about the rewarding properties of this object will be very accurate since the agent has probably approached it many times. On the other hand, if the agent is punished for approaching an object, it will not do it again, or at least be reluctant to do so. This implies that its information about the object will be based on one observation only. If the valence of the object changes from aversive to appetitive, the agent will never learn about it. A solution to this problem is to let the agent gradually forget about the punishing properties of aversive stimuli so that it eventually approaches them again. How fast this forgetting sets in should depend on how punishing the stimuli is. For example, if the punishment is very large, in most cases it is best if the stimulus is never approached again. Holland (1975) discusses this problem in the context of the genetic

algorithm and the two-armed bandit. He proves that there is an optimal strategy for the allocation of trails depending on the probability for reward combined with the certainty of that probability. This scheme could possibly be adapted for the use in autonomous agents.

7. Reinforcement Learning

Reinforcement learning can broadly be defined as change in future behaviour as a result of its past consequences. The adaptation scheme presented in the previous section constitutes a simple instance of reinforcement learning. In a more general context we may consider the Q-learning paradigm invented by Watkins (1992). The central concept of this learning paradigm is the Q-function that assigns a scalar reward to each combination of a situation and an action. If the agent is currently in the situation s it is supposed to select the action a with the highest value of $Q(s, a)$. The role of learning is to set up the Q-function appropriately.

This formulation of reinforcement learning is very different from that in the previous section in that it depends on a finite set of actions and situations. Another important difference is that it may make use of *representations*, *i.e.* a mapping from the (a, s)-pair to the reward can be entirely arbitrary. The current situation is not directly mapped on the controlling outputs of the agent. Let us first consider how an appropriate Q-function can be constructed. In the next section, I go on to discuss the relation between an agent with discrete actions and one without.

Assume that the agent is equipped with a set of actions A and can perceive a set of possible situations S. When the agent is exploring its environment it chooses among the actions in A without taking the function Q into account. Eventually the agent receives an externally generated reward R^e. Given that the agent remembers its last few actions and in what situations those actions were performed, it is able to construct the values for the Q-function for the corresponding (a, s)-pairs. The Q-value of the final action is set to R^e, while the other (a, s)-pairs are given a reward as an exponentially decreasing function of the time difference between the reward and the time when the action was performed. In other words, this is an example of delayed reinforcement learning (*cf.* Sutton and Barto 1990).

Since the memory for previously performed actions is typically limited, the agent is only able to learn its last few actions before it received the reward. If we want it to learn longer sequences, we must introduce the concept of internal rewards. This is a reward that is generated internally not because the agent has done something right but simply because it has found an action sequence that leads to a situation where the Q-value is known. In other words, if the agent knows how to get from situation s to the goal g using the Q-function, an action sequence from t to s is internally rewarded with the maximum Q-value for the situation s. While the original Q-learning can be considered a mechanism for action chaining, the internal reward mechanism extends this chunking process to larger sequences.

The internal reward function generates an internal goal gradient that can be followed by the agent. Using this view of the Q-function, it is easy to understand the interaction between a reactive system as described in section 5 and reinforcement learning. The reinforcement learning is used to reshape the goal gradients given by the external stimuli. Such an interpretation will be used further in section 11 to investigate reactive problem solving. The learning scheme used in the computer simulations are presented in appendix C. The convergence proof for general Q-learning can be found in Watkins (1992).

8. From Reaction to Action

Reinforcement learning requires representations of both actions and situations, but our simple agent from section 5 has neither. How can we enhance its architecture to make it ready for full scaled reinforcement learning? We seem to need two distinct processes, one for the creation of actions and one for the creation of situations. Both these processes rely on some sort of categorization process. I have used a simplified form of the ART 2 neural network architecture (Carpenter and Grossberg 1987) to construct both action and situation categories automatically (see Appendix C and D). Such networks have the property that they can be run backwards, *i.e.* by activating an action category node, the network will read out the corresponding action into the motor system.

A further possibility is to use the vigilance parameter of the ART network to generate more categories, and thereby a better discrimination, if the task at hand should require this. This mechanism has not been used so far.

Places, which are equivalent to situations in the current world, are categorized on the basis of their smell. To generate an approximately evenly spaced set of places, the signals from the smell receptors are first processed by a function that is the inverse of the smell diffusion function used to generate the smell intensities at the receptors. This gives the agent a set of signals that approximate the distances to the different stimuli. As a consequence, the place categories formed span the space in an almost evenly manner. In most cases, a place categorization

mechanism as this is not sufficient to construct an exhaustive map of the environment. This type of place learning was used in our simulations, not because it is particularly good, but because it was easy to implement.

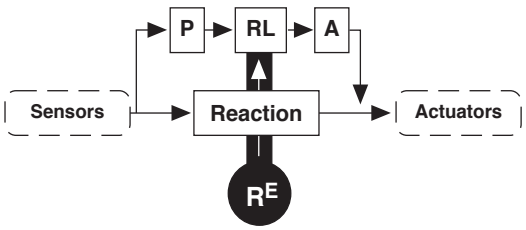

Figure 10. An agent with reinforcement learning. S: sensory signals; M: motor commands; P: perceptual categories; RL: reinforcement learning module; A: action categories; R: reactive system; R^E: external reinforcement.

I have also been investigating various neurological models of place recognition based on visual cues. Such models have been proposed by *e.g.* O'Keefe and others (O'Keefe and Dostrovsky 1971, O'Keefe and Nadel 1978) and Olton *et al.* (1978). There exist a number of computational models that are able to recognize places from visual cues that seem appropriate for an autonomous agent, *e.g.* Zipser (1985, 1986) and Schmajuk and Blair (1993a, 1993b). In these models, the recognition of the cues themselves are not discussed. This is a problem that remains to be solved. One possible algorithm is the one described by Suburo and Shigang (1993), but it seems too intractable for real time use.

Actions are constructed in a similar way. As the agent is allowed to move around the world, driven by the reactive system, a number of action categories are constructed. These actions correspond to different movements of the agent such as a left or a right turn. Of course, the agent could initially be equipped with a number of primitive actions. Such a set is, however, always a limiting factor when action sequences are to be developed. In the type of architecture described here, the agent is in principle able to construct new primitive actions as they are needed. As yet, this has not been tested in simulations.

Figure 10 shows the agent architecture with a reinforcement learning module. This module is placed on top of the reactive system in subsumption style. This implies that the reactive module is always there to take over if the reinforcement module fails to operate or is ignorant of the situation. This is an example of functional layering as described above.

9. Action Based Representations

Above, I have investigated two types of processes operating on actions. One is the process of action creation that divides already established actions or control strategies into smaller subparts. The other is the chaining process governed by reinforcement learning. This is a mechanism that takes atomic actions and links them together in sequences.

These sequences depend on the perceived situations in an interesting way. Each action that is performed is supposed to lead to a new situation in which the Q-value is known. Should this mechanism fail, say, if the environment has changed, the agent may end up in an another situation in which the Q-value is known. If this is the case, the agent can go on as if nothing has happened. In other cases, the agent may end up in an entirely new and unexpected situation. In that case, the reactive system takes over and tries to get it on track again. As soon as the agent finds a situation for which the Q-value is known, it will know what to do next. At the same time, it will have learned how to get out of the unexpected situation.

These properties rely on the fact that the representations of the agent are action based. The agent has no internal model of the environment. All it knows is based on its own behavioural repertoire. It need not know what an external stimulus is or what its sensory signals mean. All it knows is how to select an appropriate action for the current sensory input. The relations between stimuli in the environment are represented in terms of the actions that serve to transform one stimulus into another.

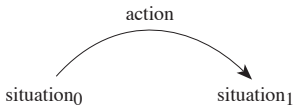

Figure 11. An situation–action–situation arc.

All the situations that the agent may find itself in, together with all the actions it can perform, may be considered a web of situation–action–situation arcs (figure 11). The more the agent has learned, the more this web will begin to look like a map of the environment. This is a very primitive map in many respects. For example, even if the agent knows how to get from A to B, it may not be able to get from B to A, *i.e.* the paths learned are not reversible. On the other hand, this asymmetry resembles the non-commutativity of distances observed in human subjects (Lee 1970). The spaces spanned by these webs have much in common with the *hodological spaces* of Lewin (1935, 1936). This was a formalism that

combined concepts from topology and vector spaces in an attempt to explain certain aspects of human behaviour.

10. Expectation Based Learning

Reinforcement learning in the form above works very well in an environment that does not change. In fact, if all actions are tested with some probability, the agent will eventually find the optimal strategy for the environment (Watkins 1992). However, most natural environments do change. It is important that the agent is able to revise its internal rewards when the environment is not up to expectations. I propose a mechanism for this that relies on two opposing reinforcement modules[4] together with a mechanism for expectation based reward. Expectation based learning has been used in biological models of classical conditioning, notably, the Rescorla–Wagner model (Rescorla and Wagner 1972) and more recently by Barto *et al*. (1990), Klopf (1988) and Klopf and Morgan (1990). See to Lieberman (1990) for an overview of expectation based models. The use of expectation here is somewhat different but depends on the same basic properties as in those models.

A learning system that can adapt to a changing environment works in the following way. Let there exist two opposing learning functions Q^+ and Q^-. Each of these functions operates as the Q-function described above. The role of each of the Q-functions is to construct a positive and a negative goal gradient respectively. The positive goal gradient operates as before. The negative goal gradient, on the other hand, has a somewhat different role. When the positive Q-function predicts a reward but it does not show up, for instance as a result of a new object obstructing the path to the goal, a negative reward is generated internally in the agent. This negative reward signal is used to drive learning in the negative Q-function. When further actions are selected, the difference between Q^+ and Q^- is used to select the appropriate action. Consequently, an action or action sequence that is not successful will be cancelled by the negative Q-function. As a result, the agent will search for a new path to the goal.

The reinforcement signal used in this type of learning is based on the difference between expectations and perceived reality. When expectations are set too high, the negative Q-function will learn to suppress the expectation. When expectations are set too low, the positive Q-function will learn. The balance between these two functions is able to adapt the creature to any changes in its environment. A consequence of this mechanism is that the agent is not rewarded for actions that it already knows will lead to an appetitive situation. Taking an anthropomorphic perspective, we may speak of surprise and disappointment in the agent.

The two opposing learning processes also make it possible to use punishment as a teaching strategy. In most cases, however, this is not an advisable strategy as it induces an avoidance behaviour rather than an approach behaviour. This will make the agent avoid the punishing situation in any number of ways. We may not be able to know what the agent will do. Another problem with punishment is that the actions of the agent will be based on imprecise information. Since the agent will try to avoid the punishing situation, it will not be able to get acquainted with that situation. If the punishment was a bad coincidence, the agent will probably never learn about it. In other words, if one wants the agent to do something, it should be rewarded for doing it. Punishment is only useful if we want the agent not to do something. Very often, reward is better in those cases too as it will induce the agent to make more informed choices. Unfortunately, this point is not as well known as one would wish.

11. Dynamic Action Selection

The presentation so far has assumed that there is only one goal to pursue. In realistic situations, this is almost never the case. For instance, an animal must both eat and drink, sleep, build a nest, find a mate, etc. These are different and competing goals. An agent must be able to select among these goals in a rational way. In Balkenius (1993) a neural network architecture is presented that can handle decisions about what to do and when. I identify this module with the motivational system of an animal. A motivational state is selected as a result of three factors.

- **Internal drives** The agent has a number of primitive needs that vary dynamically over time. Once a need is not fulfilled, an internal drive signal is generated that increases the probability of the agent selecting actions that serve to fulfill that need. For instance, in an autonomous robot, a drive could correspond to the need to recharge the batteries while another drive could make the robot inspect a gate at regular intervals.

- **External incentive** At all times, the agent receives sensory input that tells it about the possibility of fulfilling a need. For instance, the visual view of the electrical outlet would constitute an incentive to recharge the batteries.

[4] Compare these two opposing learning systems with the two types of reactive connections (*i.e.* crossing and non-crossing) from sensors to effectors.

◊ **Internal incentive** Internal incentive has the same role as external incentive except that it does not directly depend on the currently perceived situation. Instead it is generated in the reinforcement learning modules. An internally expected reward corresponding to battery recharge would make the agent more likely to recharge the batteries even if the outlet is not at sight.

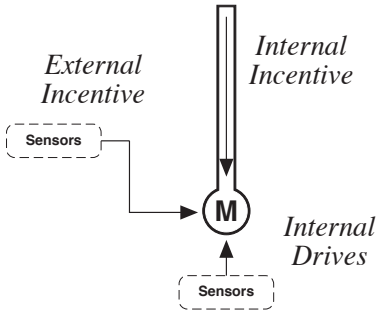

Figure 12. The determinants of motivation.

These three factors are weighted together for each of the engagements of the agent, and a decision emerges about what the agent should do. It is quite possible that this decision is changed as the agent tries to pursue the goal. If our hypothetical robot passes the gate on its way to the electrical outlet, it may very well change its decision and inspect the gate on its way. Once the gate is checked, the robot will continue towards the electrical outlet. This is an example of opportunistic behaviour that results from the interaction between a motivational module and reinforcement learning or a reactive control system.

To implement a system like this, the agent must be equipped with a number of reinforcement modules Q_0, Q_1,...Q_n, *i.e.*, one for each motivational state. At any time, the reinforcement module that corresponds to the current motivational state is used to control behaviour. Of course, all reinforcement modules are simultaneously able to generate internal incentive to the motivational system. This is an example of the principle of central behaviour selection. I have previously argued that a motivational system of this type is necessary in a biological system for higher cognitive function to evolve. The argument and the main evolutionary stages are presented in Balkenius (1993).

In principle, it is possible to use only one reinforcement module to control all behaviours. However, such a system is very cost ineffective. It is able to produce a slightly better performance than a system with separate reinforcement modules but the learning process is many times slower. It has been shown in Tenenberg *et al*. (1993) that a task decomposition of behaviours is a much more practical way to handle many competing goals. Figure 13 shows the architecture of our current agent.

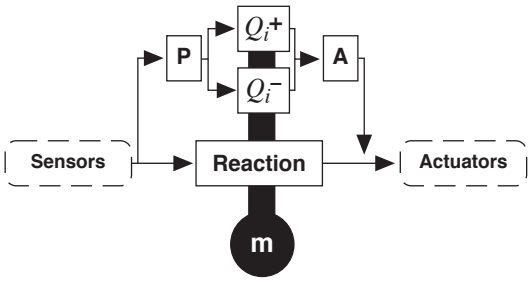

Figure 13. The agent with a motivational module, m, and a more advanced reinforcement learning system, Q_i.

12. Reactive Problem Solving

I now turn to a recent discovery concerning reactive problem solving in terms of a goal gradient. Consider the classic situation in figure 14a. The agent is placed on one side of a barrier and an appetitive stimulus is placed on the other side. The barrier is such that the stimulus is visible from the position of the agent but the agent cannot pass through it. In order to reach the goal, the agent must go around the barrier. This is a complicated situation since the reactive system will guide the agent straight ahead towards the barrier where the agent will get stuck. Somehow the agent must learn to discount of its immediate sensory information and take the path around the barrier.

In the classical works of Hull (1943), Lewin (1935, 1936) and Tolman (1932), this is a problem that was considered to require insight on the behalf of the agent (*cf*. Rashotte 1987). Insight, though, could not be explained in mechanistic terms. As we will see, an analysis in terms of the concepts I have developed in this paper suggests a possible mechanism.

The goal gradient generated by the reactive system will guide the agent directly towards the appetitive stimulus without concern for the obstructing barrier (Figure 14b). Let us assume that an internal reinforcement signal is generated at every place that corresponds to the goal gradient generated by the stimulus. Figure 14c shows the initial internal reinforcement signals generated along the two paths A–B and A–G shown in figure 14a. Note that internal reinforcement is equivalent to the potential discussed in section 5.

As the agent approaches the goal, its expected reward will increase for every step until it reaches the barrier. When the agent is stopped by the barrier, there will be a negative difference between its expected reward and its actual reward that will cause

changes in the negative Q-function for the current motivational state. This function will increase until it has completely cancelled the attracting force of the stimulus. For each action that is performed that does not lead closer to the goal, the Q^- will increase. Figure 14d shows the negative goal gradient that is constructed through this process. When the positive goal gradient resulting from the sensory input is combined with the negative goal gradient constructed in Q^- are added together, the agent will take the correct path around the barrier following the goal gradient in figure 14e.

This is an example of truly reactive problem solving. Nowhere does the agent need the ability to anticipate the consequences of it actions. All changes to the agent are made on a reactive basis. This is a rather general mechanism that can be used to make autonomous agents recover from behavioural strategies with outcomes that are worse than expected. An interesting aspect of the ability to perform reactive problem solving is that it rests entirely on the interaction between a set of already existing modules.

The type of behaviour generated by our simulated agent should be compared with the behaviour of higher animals and small children that are placed in a similar situation. At first they try to pass straight through the barrier and only after much struggle will they take the path around the barrier to approach the goal (Lewin, 1936).

We may finally compare the behaviour of our agent with an experiment performed by Sutton and Barto (1990) in an equivalent situation. Their *temporal difference* (TD) procedure needed about 500 trails to learn the correct path around the barrier and only produced optimal behaviour after more than 5000 trails while the learning mechanism presented here required only one. The success of the current approach has its origin in the interaction between the reactive and the learning modules. Since the simulation by Barto and Sutton did not incorporate a reactive system but used random walk to try out different actions, the comparison is not entirely fair. It does nevertheless show how effective performance can result from several interacting strategies.

The reader may object that most humans would not first try to pass through the barrier and only later take the successful path around it. I want to suggest that in many equivalent situations human behaviour is very similar. It is important to remember that the barrier is only perceived by the agent when its whiskers are in contact with it. In other words, the agent does indeed avoid the obstacle as soon as it is perceived. Similar behaviour is produced when one tries to drive the shortest way to some goal only to find that the road is blocked half way through. However, there exists many situations that cannot be handled in the way described above. This is especially true in environments where some actions are non-reversible. In such environments it is useful to equip the agent with the ability to do model based planning.

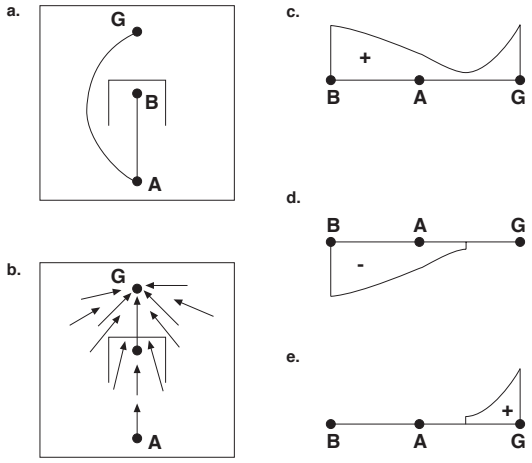

Figure 14. Reactive problem solving (a) the problem situation; (b) forces generated by the goal; (c) the initial goal gradient generated by the goal only along the path from A to G.; (d) the internal negative gradient; (e) the final gradient that solves the problem. The text describes the situation further.

13. Planning

Reactive problem solving in the sense presented above requires the agent to try out different behaviours in the environment to figure out how to solve the problem. What happens if one of the attempted actions has consequences that make the attainment of the goal impossible? It is clear that such situations are rather common in human environments. For example, it would not be very wise to break the key in an attempt to open a locked door. Without any knowledge whatsoever, how is one supposed to know that breaking the key is not the action that in fact opens the door? Of course, the solution is to keep knowledge of previous behaviours and their consequences. This information can be used to predict what will happen when an action is performed. Hopefully, the knowledge can be used to avoid some problems with non-reversible operations. It is also more cost effective to simulate actions internally instead of performing them externally. How can this type of planning ability be added to the already existing architecture?

A general architecture for a planning agent is presented in (Gulz 1991). Planning is considered as internal simulation of external behaviours. Instead of using the external world to generate new sensory information, actions are performed in an internal model. The different actions are simulated in this model and a new sensory input is generated internally. In this view, the internal model is not

independent of the agent itself. It is a module that has the ability to generate the appropriate sensory information that would result if the agent had performed the corresponding external action.

The internal model is used in a very different way than the models used within the symbol processing paradigm. The model is not a description of the world. As far as our agent is concerned, the internal model *is* a world. To be successful, this internal world must parallel the external world. Planning can be seen as behaviour in this internal world instead of the external world (*cf.* Gärdenfors 1992a, Gärdenfors and Balkenius 1993). The only difference between a search in the internal world compared with the external world is that all actions are reversible and can be performed much faster than actions in the external world. The plan constructed from behaviour in the internal world is no different from the paths learned from externally tested action sequences. Both types of learning are made as described above in section 7. By depending on the already existing modules, planning ability is the result of adding a single module (Figure 6). Again we have an example of a true system property that cannot be localized in one specific module of the agent. With the model module, the agent can engage in planning, but this module does not plan in itself. Similar ideas have been used, for example, by Jordan and Rumelhart (Jordan and Rumelhart 1992) to construct an inverse model and by Nguyen & Widrow (Nguyen and Widrow 1989) in adaptive control. In many respects, the proposed architecture is similar to Sutton's DYNA (Sutton 1992).

14. Directions for Further Research

The present architecture is not yet final in any way. The simulations discussed in this paper are the simplest behavioural instances for a complete agent architecture I could conceive of. As already mentioned, the individual modules presented here are among the simplest possible of each kind. In the future, I will aim at extending the different agent modules to handle more complex situations.

The perceptual side of the agent will be extended to include visual input. The place learning method used above is clearly too simple for most problems. In future architectures I will make use of angular relations between visual landmarks to recognize and categorize places.

In more complex environments, it will be necessary to categorize the different situations in more detail. A much more powerful mechanism for perceptual representation will be needed.

Balkenius (1992) presents an architecture for neural networks that are able to self-organize perceptual schemata for static and dynamic percepts. In Gärdenfors (1992b), the inductive properties of this type of architecture are discussed. Some properties that seem to be required of a neural code for successful representations of an environment are reported in Balkenius (1994). It would be interesting to add these more advanced perceptual mechanisms to the present architecture. I believe that a more advanced perceptual system would aid transfer of learning between different similar tasks.

I will also study environments where a planning ability is of use. As yet, I have not attempted an implementation of any planning ability. Mainly because it is hard to think of a domain where planning really makes any difference. Compared to the simple environments that have been simulated so far, the situations where planning would be necessary are far more complicated.

When I consider agents with a larger set of potential behaviours I will also be confronted with problems of coordination of several concurrent actions. In this case, the mechanisms for action creation and chaining must be much more advanced than the modules used in the examples above.

15. Conclusion

A general architecture for a behaviour based control system of autonomous agents has been presented. It is based on a number of architectural principles. Most importantly, the system is organized in a number of layers.

The first layer consists of a number of parallel reactive control systems organized in a subsumption style architecture. These parallel systems can be activated or inhibited by a central behaviour selection mechanism.

The second layer is used for reinforcement learning. There are two opposing learning modules for each motivational state of the agent. These modules interact with the underlying reactive system to produce more complex behaviours. The main operation of the learning modules in the second layer is the dynamic construction of a discrete action set and chaining of those actions into behavioural sequences. Some abilities of the agent to perform reactive problem solving have also been demonstrated.

Finally, a layer is added to the architecture that makes planning possible. The role of this final module is to construct a mapping from an action to its consequences. To the agent, this mapping is considered

an internal environment where actions can be tried before they are confronted with the unforgiving external reality.

The mechanisms presented are mainly based on the type of learning one would find in biological systems. I believe that the type of overall architecture presented here can be used as a starting point for the development of a much more advanced and capable autonomous agent. Insofar as the behaviour of the agent can be considered intelligent, it shows natural rather than artificial intelligence.

Acknowledgements

I would like to thank Paul Davidsson, Agneta Gulz, Peter Gärdenfors, Paul Hemeren, Lars Kopp and Robert Pallbo for their comments on this paper.

The simulations described in this paper were made using the BERRY III simulator. This program is available free from the author for non-commercial purposes. The simulator can also be downloaded by anonymous ftp from LUCS.fil.lu.se in the directory pub/simulators/BERRY–III. The current version of the program requires a Macintosh computer with at least a 020 processor, colour graphics and a math coprocessor. These limitations may change in the future.

Appendix A: Reactive Control

Let s^L be the array of intensities delivered from the sensors on the left of the body and s^R the corresponding sensors on the right side of the body. The connections from sensors to motors will be called w^{XY} where X is the side of the sensors and Y is the side of the motor. In other words, there are four sets of weights, w^{LL}, w^{LR}, w^{RR} and w^{RL}. The output to each of the motors o^L and o^R is calculated as follows.

$$o^L = \sum_{i \in I} s_i^L w_i^{LL} + s_i^R w_i^{RL} \tag{A1}$$

$$o^R = \sum_{i \in I} s_i^R w_i^{RR} + s_i^L w_i^{RL} \tag{A2}$$

These formulas assume that the signals from the sensors do not need to be pre-processed through some non-linear function. A slight reformulation of formula (A1) and (A2) makes it possible to account for the different slopes of the approach and avoidance gradients.

We may reformulate (A1) as,

$$o^L = f^+(p^L) - f^-(q^L), \tag{A3}$$

where p^L is the sum of the positive components of the sum in (1),

$$p^L = \sum_{i \in I} \left[s_i^L w_i^{LL} \right]^+ + \left[s_i^R w_i^{RL} \right]^+ \tag{A4}$$

q^L is the sum of the corresponding negative components,

$$q^L = \sum_{i \in I} \left[-s_i^L w_i^{LL} \right]^+ + \left[-s_i^R w_i^{RL} \right]^+, \tag{A5}$$

and $\left[x\right]^+ = \max(x,0)$. Equation (A2) can be changed in the same way. The two functions f^+ and f^- describe the slope of the appetitive and aversive gradients. In the typical case,

$$\tfrac{d}{dx} f^+(x) < \tfrac{d}{dx} f^-(x) \tag{A6}$$

Appendix B: Simple Reactive Adaptation

Let $R(t)$ be an externally generated reinforcement signal at time t which is positive when the agent reaches appetitive stimuli and negative on contact with aversive stimuli. We want the weights w from equation (A1)–(A5) to change in a way that reflect the values of $R(t)$. There are a number of possible ways to do this. Let,

$$w^{LR}(t+1) = w^{LR}(t) + \Delta w(t), \tag{B1}$$

$$w^{RL}(t+1) = w^{RL}(t) + \Delta w(t). \tag{B2}$$

A linear learning rule can be formulated in the following way,

$$\Delta w_i(t) = \varepsilon R(t) \left(\frac{s_i^L(t) + s_i^R(t)}{2} \right) \tag{B3}$$

The weights that cross the centre of the body are changed with an amount that is proportional to the average sensory input of each type. It is, of course, also necessary to limit the values of w within a suitable range. Weights that are too large would make the movements of the agent oscillate on its way towards the goal. Assume that the suitable range for the w:s is $[-a, a]$. We may use this information to formulate a delta-rule type learning rule (*cf.* Widrow and Hoff 1960/1988).

$$\Delta w_i(t) = \varepsilon\Big(w_i(t) - a\Big)R(t)\left(\frac{s_i^L(t) + s_i^R(t)}{2}\right). \qquad \text{(B4)}$$

Formulas (A7)–(A10) assume that the weights are initially symmetrically organized. If this is not the case, it is necessary to have a different Δw for each side of the agent. Note that the learning rules proposed here has the effect that the agent will increase its speed as it approaches an appetitive stimulus but slow down when it tries to avoid an aversive stimulus. It is easy to extend the learning rules to operate on all weights if we do not want the agent to slow down as described in section 5.

Appendix C: Categorization of Places

This appendix describes the simplest possible mechanism for the categorization of places. The algorithm is derived from the ART 2 neural network architecture (Carpenter and Grossberg 1987) and can be described as follows. Let $x=<x_0, x_1,\ldots, x_n>$ be the activity of a set of neurons n and $w_0, w_1,\ldots, w_n$ a corresponding set of k component weight vectors. Denote the sensory signals by $s=<s_0, s_1,\ldots, s_k>$. Let c be the index of the last committed neuron (0 initially). The value φ is a *vigilance* parameter that determines the number of constructed categories and α is a learning rate. The algorithm for the categorization of places follows:

At each time step:

(0) Get the sensory input s.
(1) For each $i \leq c$ let $x_i = w_i \cdot s$.
(2) Set m to the index of the x_i with the largest activity.
(3) If $x_m < \varphi$ then let $c := c+1$ and $m := c$;
(4) Let $x_i = 1$ if $i = m$ and 0 otherwise.
(5) Let $w_m := (1-\alpha)w_m + \alpha s$.

The index of the current place is given by m. It should be noted that this algorithm is an absolute minimum for place learning. To work, it depends on a nicely structured input space and a bit of luck. The reason for using this algorithm is that it is necessary to test the rest of the architecture. In any realistic situation a more advanced place learning scheme must be used (*e.g.* Schmajuk and Blair 1993a, 1993b, Zipser 1985, 1986).

Appendix D: Action Construction

The construction of action categories (see section 8) can be done in the same way as place learning as described in Appendix C. All we have to do is to replace the sensory input vector with the output signals to the motors from the reactive system. The algorithm will learn a set of actions as a direct result of the motor outputs such as go forward, turn left, etc. These categories are then used as actions in the reinforcement learning system.

Actions are performed by running the network 'backwards'. An action category node n_i is activated and its learned motor pattern is sent to motors for a fixed length of time. Figure D1 shows the relation between the motor system and the action categorization system. Similar architectures can be found for example in Carpenter and Grossberg (1987) Grossberg and Kuperstein)1989).

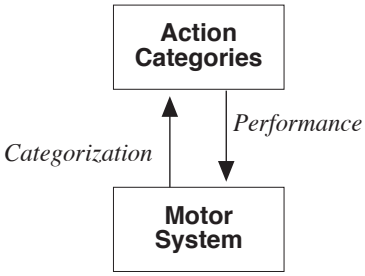

Figure D1. Motor commands are learned in the action categorization module and can later be read out into the motor system by activation of the appropriate action node.

Appendix E: Q-Learning

This appendix describes a more general form of reinforcement learning than the form in appendix B. The learning algorithm, Q-learning, is very general but very slow. In appendix F, another version of reinforcement learning is presented which is much faster. This other type of reinforcement learning is not reversible, however. As a consequence, both algorithms have been used in the simulations described above.

The Q-learning scheme presented here was implemented as a neural network but here it will only be described at the computational level.

Let A be a set of actions and S a set of situations. Situations correspond either to the direct sensory input to the agent or the sensory input in combination with some place representation. It is possible that the actions have been dynamically created as described above in Appendix C.

A Q-function is a function that assigns an internal expected reward value, R^e, to each pair of an action, $a \in A$, and a situation, $s \in S$ (*cf.* Watkins 1992).

Let $s(t)$ denote the situation perceived by the agent at time t and let $a(t)$ be the action performed. This action changes the situation into $s(t+1)$. The value of $Q(s,a)$ is given by the table q_{ij} where i ranges over all situations and j over all actions. Initially, $q_{ij} = 0$ for all i and j, α is a learning rate, and γ is a discount factor that makes a future reward worth less than an immediate reward.

If $R(t+1)>1$, q_{ij} is updated according to the formula,

$$q_{ij}(t+1) = (1-\alpha)q_{ij}(t) + \alpha R(t+1), \qquad \text{(E1)}$$

for $i=s(t)$ and $j=a(t)$. If R(t+1)=0 the following formula is used instead,

$$q_{ij}(t+1) = (1-\alpha)q_{ij}(t) + \alpha\gamma V(s(t+1)), \qquad \text{(E2)}$$

where,

$$V(s) = \max_j q_{sj}. \qquad \text{(E3)}$$

When the agent makes use of the learned Q-function, it has only to select the action that maximizes $Q(s,a)$ in the current situation s. Watkins (1992) proves that this learning algorithm results in the optimal policy in the limit. However, the learning is usually very slow.

Appendix F: Fast Reinforcement Learning

This appendix describes an alternative reinforcement learning schema that is much faster than ordinary Q-learning. So far, it has not been possible to incorporate reversible learning in this model in the general case. It can, however, be used with great utility in fixed environment or in environment with only one goal of each type.

The learning now proceeds in two steps. First the short term memory of the agent is updated and then the q-table is backed-up.

We define the short term memory as a table s_{ij}. All elements of s are initially set to 0. To update the short term memory, we first calculate the situation–action gain, $m_{ij}(t)$, as

$$m_{ij}(t) = \max\left\{\delta x_{ij}(t), n_{ij}(t)\right\}, \qquad \text{(F1)}$$

where,

$$n_{ij}(t) = \begin{cases} 1 & \text{if } i = s(t) \text{ and } j = a(t) \\ 0 & \text{otherwise} \end{cases} \qquad \text{(F2)}$$

Then the short term memory is updated according to the formula,

$$x_{ij}(t+1) = \begin{cases} m_{ij}(t) & \text{if } m_{ij}(t) > \theta \\ 0 & \text{otherwise} \end{cases} \qquad \text{(F3)}$$

The relative sizes of the trace decay, δ, and the memory threshold, θ, determine the memory capacity of the learning system. If q_{ij} is implemented as a lookup table, there is no reason why the threshold could not be set to 0, giving the agent unlimited short term memory. In a neural network implementation, this is less realistic.

We now calculate the current reward as,

$$R(t) = \max\left(R^E(t), R^I(t)\right) \qquad \text{(F4)}$$

where, $R^E(t)$ is the current external reward and $R^I(t)$ is given by,

$$R^I(t) = \max_{a\in A} Q\left(s(t),a\right). \qquad \text{(F5)}$$

Finally we back up the q-table using the formula,

$$q_{ij}(t) = \max\left\{q_{ij}(t-1),\ R(t)x_{ij}(t)\right\}. \qquad \text{(F6)}$$

In a stable world where actions are chosen stochastically according to the Q-function, this learning scheme will generate the optimal Q-function very fast. Since the mechanism incorporates a short term memory, learning is much faster than in ordinary Q-learning.

On the basis of equation (F6) the $Q(s, a)$ can only grow larger. In a changing environment, we also need a mechanism that can decrease the size of $Q(s,a)$. The solution proposed above is to use two Q-functions, one for positive learning and one for negative learning. Investigating this possibility in the general case of a changing environment with multiple goals is one of my current research issues.

References

Arbib, M. A. & House, D. H., (1987), "Depth and detours: an essay on visually guided behavior", in M. A. Arbib and A. R. Hanson (eds.) *Vision, brain and cooperative computation*, 129–163, Cambridge, MA: MIT Press.

Arkin, R. A., (1990), "Integrating behavioural, perceptual and world knowledge in reactive navigation", in P. Meas (ed.) *Designing autonomous agents*, Cambridge, MA: MIT Press.

Baird, L. C. I. & Klopf, A. H., (1993), "Extensions of the associative control process (ACP) network: Hierarchies and provable optimality", in J.–A. Mayer, H. L. Roitblat and S. W. Wilson (eds.) *From animals to animats II*, Cambridge, MA: MIT Press.

Balkenius, C., (1992), "Neural mechanisms for self-organization of emergent schemata, dynamical schema processing, and semantic constraint satisfaction", *Lund University Cognitive Studies*, **14**.

Balkenius, C., (1993), "The roots of motivation", in J.–A. Mayer, H. L. Roitblat and S. W. Wilson (eds.) *From animals to animats II*, Cambridge, MA: MIT Press.

Balkenius, C., (1994), "Some properties of neural representations", in M. Bodén and L. Niklasson (eds.) *Connectionism in a broad perspective,* 79–88, Ellis Horwood, New York.

Barto, A. G., Sutton, R. S. & Anderson, C. W., (1983), "Neuronlike elements that can solve difficult learning control problems", *IEEE Transactions on systems, man, and cybernetics*, **13**, 834–846.

Barto, A. G., Sutton, R. S. & Watkins, C. J. C. H., (1990), "Learning and sequential decision making", in M. Gabriel and J. Moore (eds.) *learning and computational neuroscience: foundations of adaptive networks*, Cambridge, MA: MIT Press.

Braitenberg, V., (1984), *Vehicles: Experiments in synthetic psychology*, Cambridge, MA: MIT Press.

Brooks, R. A., (1986), "Achieving artificial intelligence through building robots", 899, MIT Artificial Intelligence Memo.

Brooks, R. A., (1991a), "How to build complete creatures rather than isolated cognitive simulators", in K. VanLehn (ed.) *Architectures for intelligence*, Hillsdale, NJ: Lawrence Erlbaum Associates.

Brooks, R., (1991b), "Intelligence without reason". *Proceedings of IJCAI–91*, 569–595.

Carpenter, G. & Grossberg, S., (1987), "ART2: Self-organization of stable category recognition codes for analog input patterns", *Applied Optics*, **26**, 4919–4930.

Carpenter, G. A. & Grossberg, S., (1987), "Neural dynamics of category learning and recognition: Attention, memory consolidation, and amnesia", in S. Grossberg (ed.) *The Adaptive Brain*, Amsterdam: North-Holland.

Donnet, J. & Smithers, T., (1991), "Lego vehicles: a technology for studying intelligent systems", in J.–A. Meyer and S. W. Wilson (eds.) *From animals to animats*, Cambridge, MA: MIT Press.

Dumeur, R., (1991), "Extended classifiers for simulation of adaptive behaviour", in J.–A. Meyer and S. W. Wilson (eds.) *From animals to animats*, Cambridge, MA: MIT Press.

Gray, J. A., (1982), The neuropsychology of anxiety: an enquiry into the functions of the septo-hippocampal system, Oxford: Oxford University Press.

Grossberg, S. & Kuperstein, M., (1989), *Neural dynamics of adaptive sensory-motor control*, Elmsford, N.Y: Pergamon Press.

Gulz, A., (1991), "The planning of action as a cognitive and biological phenomenon", *Lund University Cognitive Studies*, **2**.

Gärdenfors, P., (1992a), "Medvetandets evolution". *Blotta tanken, ch 5*, Nora: Nya Doxa.

Gärdenfors, P., (1992b), "Three levels of inductive inference", *Lund University Cognitive Studies*, **9**.

Gärdenfors, P. & Balkenius, C., (1993), "Varför finns det inga riktiga robotar?", Framtider, **12**, 1.

Holland, J., (1975), *Adaption in natural and artificial systems*, University of Michigan Press.

Horswill, I., (1993), "A simple, cheap, and robust visual navigation system", in J.–A. Mayer, H. L. Roitblat and S. W. Wilson (ed.) *From animals to animats II*, Cambridge, MA: MIT Press.

Hull, C. L., (1938), "The goal-gradient hypothesis applied to some "field-force" problems in the behaviours of young children", *Psychological Review*, **45**, 271–299.

Hull, C. L., (1943), *Principles of behavior*, Appleton-Century-Crofts, New York.

Jordan, M. I. & Rumelhart, D. E., (1992), "Forward models: supervised learning with a distal teacher", *Cognitive Science, in press*.

Klopf, A. H., (1988), "A neuronal model of classical conditioning", *Psychobiology*, **16**, 85–125.

Klopf, A. H. & Morgan, J. S., (1990), "The role of time in natural intelligence: implications of classical and instrumental conditioning for neuronal and neural-network modeling", in M. Gabriel and J. Moore (eds.) *Learning and computational neuroscience: foundations of adaptive networks*, Cambridge, MA: MIT Press.

Lee, T., (1970), "Perceived distance as a function of direction in the city", *Envrion. Behav.*, **2**, 40–51.

Lewin, K., (1935), *A dynamic theory of personality: selected papers*, New York: McGraw-Hill.

Lewin, K., (1936), *Principles of topological psychology*, New York: McGraw-Hill.

Lieberman, D. A., (1990), *Learning: behavior and cognition* Belmont, CA: Wadsworth Publishing Company.

Mackintosh, N. J., (1983), *Conditioning and associative learing*, Oxford: Oxford Univeristy Press.

Maes, P., (1990), "Designing autonomous agents", in P. Meas (ed.) *Designing autonomous agents*, Cambridge, MA: MIT Press.

Maes, P., (1991), "A bottom-up mechanism for behvior selection in an artificial creature", in J.-A. Meyer and S. W. Wilson (eds.) *From animals to animats*, 238–246, Cambridge, MA: MIT Press.

Marr, D., (1982), *Vision*, San Fransisco: W. H. Freeman.

Millán, J. d. R. & Torras, C., (1992), "A reinforcement connectionist approach to robot path finding in non-maze-like environments", *Machine Learning*, **8**, 363–395.

Newell, A., (1990), *Unified theories of cognition*, Cambridge, MA: Harvard University Press.

Nguyen, D. & Widrow, B., (1989), "The truck backer-upper: an example of self-learning in neural networks". *Proceedings of the international joint conference on neural networks*, Piscataway, NJ: IEEE Press.

O'Keefe, J. & Dostrovsky, J., (1971), "The hippocampus as a spatial map: Preliminary evidence from unit activity in the freely moving rat", *Brain research*, 171–175.

O'Keefe, J. & Nadel, L., (1978), *The hippocampus as a cognitive map*, Oxford: Clarenon Press.

Olton, D. S., Branch, M. & Best, P., (1978), "Spatial correlates of hippocampal unit activity", *Experimental Neurobiology*, 387–409.

Pallbo, R., (1992), "Neuronal selectivity without intermediate cells", *Lund University Cognitive Studies*, **13**.

Pallbo, R., (1993), "Visual motion detection based on a cooperative neural network architecture", in E. Sandewall and C. G. Jansson (eds.) *Scandinavian conference on artificial intelligence – 93*, 193–201, Amsterdam: IOS Press.

Rashotte, M. E., (1987), "Behaviour in relation to objects in space: some historical perspectives", in P. Ellen and C. Thinus-Blanc (eds.) *Cognitive processes and spatial orientation in animal and man*, Dordrech: Nijhoff.

Rescorla, R. A. & Wagner, A. R., (1972), "A theory of Pavlovian conditioning: variations in the effectiveness of reinforcement and nonreinforcement", in A. H. Black and W. F. Prokasy (eds.) *Classical conditioning II: current research and theory*, New York: Appleton-Century-Crofts.

Schmajuk, N. A. & Blair, H. T., (1993a), "Dynamics of spatial navigation: an adaptive neural network", in J.–A. Mayer, H. L. Roitblat and S. W. Wilson (eds.) *From animals to animats II*, Cambridge, MA: MIT Press.

Schmajuk, N. A. & Blair, H. T., (1993b), "Place learning and the dynamics of spatial navigation: a neural network approach", *Adaptive Behavior*, **1**, 353–385.

Schnepf, U., (1991), "Robot ethology: a proposal for the research into intelligent autonomous systems.", in J.–A. Meyer and S. W. Wilson (eds.) *From animals to animat*, Cambridge, MA: MIT Press.

Suburo, T. & Shigang, L., (1993), "Memorizing and representing route scenes", in J.–A. Mayer, H. L. Roitblat and S. W. Wilson (eds.) *From animals to animats II*, Cambridge, MA: MIT Press.

Sutton, R. S., (1992), "Reinforcement learning architectures for animats", in J.–A. Meyer and S. W. Wilson (eds.) *From animals to animats*, Cambridge, MA: MIT Press.

Sutton, R. S. & Barto, A. G., (1990), "Time-derivative models of Pavlovian reinforcement", in M. Gabriel and J. Moore (ed.) *learning and computational neuroscience: foundations of adaptive networks*, Cambridge, MA: MIT Press.

Tenenberg, J., Karlsson, J. & Whitehead, S., (1993), "Learning via task decomposition", in J.-A. Mayer, H. L. Roitblat and S. W. Wilson (eds.) *From animals to animats II*, Cambridge, MA: MIT Press.

Tolman, E. C., (1932), *Purposive behaviour in animals and men*, New York: Appleton-Century-Crofts.

Watkins, C. J. C. H., (1992), "Q-learning", *Machine Learning*, **8**, 279–292.

Widrow, B. & Hoff, M. E., (1960/1988), "Adaptive switching circuits", in J. A. Anderson and E. Rosenfeld (eds.) *Neurocomputing: foundations of research*, Cambridge, MA: MIT Press.

Williams, R. J., (1992), "Simple statistical gradient-following algorithms for connectionist reinforcement learning", *Machine Larning*, **8**, 229–256.

Wilson, S. W., (1985), "Knowledge growth in an artificial animal", in Grefenstette (ed.) *Proceedings of the first international conference on genetic algorithms and their applications*, Lawrence Erlbaum Assoc.

Wilson, S. W., (1987), "Classifier systems and the animat problem", *Machine Learning*, **2**, 199–228.

Zipser, D., (1985), "A computational model of hippocampal place fields", *Behavioral Neuroscience*, 1006–1018.

Zipser, D., (1986), "Biologically plausible models of place recognition and goal location", in D. E. Rumelhart and J. L. McClelland (eds.) *Parallel distributed processing*, Cambridge, MA: MIT Press.