Computer Science
University of Manchester

Autonomous Acquisition of Sensor-Motor Couplings in Robots

Ulrich Nehmzow

Technical Report UMCS-94-11-1

Autonomous Acquisition of Sensor-Motor Couplings in Robots*

Ulrich Nehmzow

Department of Computer Science
University of Manchester
Oxford Road, Manchester, UK.
u.nehmzow@cs.man.ac.uk

November 1994

Recent technical reports issued by the Department of Computer Science, Manchester University, are available by anonymous ftp from ftp.cs.man.ac.uk in the directory pub/TR. The files are stored as PostScript, in compressed form, with the report number as filename. They can also be obtained on WWW via URL http://www.cs.man.ac.uk/csonly/cstechrep/index.html. Alternatively, all reports are available by post from The Computer Library, Department of Computer Science, The University, Oxford Road, Manchester M13 9PL, UK.

Abstract

Fixed robot controllers are suitable for tasks whose main characteristics are known *a priori*. However, for tasks such as exploration of unknown territory, fixed controllers tend to be too brittle as they rely on predefined knowledge, supplied by the designer. For such tasks (i.e. tasks in which *a-priori* knowledge is limited) we propose to use self-organising controllers which allow the robot to acquire the necessary competences autonomously ([Nehmzow *et al.* 89]).

The paper describes experiments with mobile robots, in which the robots autonomously determine effective connections between their sensory input and their motor response to it.

In these experiments, artificial neural networks are used to store associations between sensory input and motor responses, the networks are trained in a self-supervised way, making use of so-called *instinct-rules* which govern the robots' behaviour. The fast learning achieved enables the robots to adapt to changing circumstances such as changes in their environment or their morphology (e.g. sensor failure).

1 Introduction

This paper presents experiments on autonomous competence acquisition, conducted with two small mobile robots, *Alder* and *Cairngorm*, both fully autonomous, equipped with on-board controllers, actuators and tactile sensors.

The questions it addresses are: How can flexibility in situations unforeseen by the designer be achieved? How can robots autonomously adapt to changing circumstances in world, task, or their own morphology? How can they improve their performance, based on past experiences?

The experiments described in this paper use an adaptable mapping between incoming sensor signals and outgoing actuator responses to approach these questions: an artificial neural network is used to store sensor-motor couplings and is trained in real time, using the information that is obtained through the robot's interaction with the environment. The learning is rapid, tasks such as obstacle avoidance or contour following are acquired in a couple of learning steps, within seconds.

1.1 Background: Reinforcement Learning

Reinforcement Learning — the learning mechanism used here — covers techniques of "learning by trial and error through performance feedback, i.e. from feedback that evaluates the behaviour, ...but does not indicate correct behaviour" ([Sutton 91]). As it uses a measure of overall performance only, it is relatively easy to implement and has indeed been used in robot control, for example to achieve phototaxis ([Kaelbling 91, Colombetti & Dorigo 93]), boxpushing ([Mahadevan & Connell 91]), obstacle avoidance ([Nehmzow *et al.* 89]) or walking ([Maes & Brooks 90]). One problem in reinforcement learning is that if the search space (e.g. the space of possible actions) is large, learning may take a long time to "take off", simply because very little positive reinforcement is received in the early stages of the learning process. Kaelbling describes this vividly by pointing out that her light-seeking robot needed to be pushed towards the light manually in the early stages of learning, in order to subsequently acquire the competence of phototaxis ([Kaelbling 90]). Similarly, Mahadevan and Connell report that their best reinforcement learning algorithm used for box-finding (Q-learning with statistical clustering) received an average reward of about 0.16 after 2000 learning steps, whilst a hand-coded box-finding agent was able to achieve an average award of about 0.25.

The point is: the obvious benefits of reinforcement learning in robots — the ability to adapt to changing environments, to cope with sensor failure, decreasing battery charge, to acquire new competences, etc. — can be lost if too long a learning phase is required, particularly for tasks fundamental to the robot's operation, such as for instance obstacle avoidance. Careful design of input and output space (for example by using quantised input data —[Mahadevan & Connell 91] — or a restricted set of motor actions — [Kaelbling 91, Maes & Brooks 90]) is a way to alleviate this problem. In addition to these two methods we have chosen a "Pattern Associator" as core component of the controller — an artificial neural network that can learn only a restricted set of functions, but these extremely fast (one or two orders of magnitude faster than other learning schemes, such as for instance the backpropagation network). This enables our robots to acquire fundamental sensor-motor couplings rapidly, requiring a couple of learning steps only.

The following sections describe the robots, the controller architecture and the experiments we conducted regarding learning in mobile robots. More details about these experiments can be found in [Nehmzow 92].

2 Competence Acquisition, using Artificial Neural Networks

2.1 The Robots

The experiments described in this paper were conducted with two small mobile robots, shown in figure 1. *Alder* is controlled by an ARC52 controller, based on the 8052 microprocessor, having 16k of RAM. *Cairngorm* is controlled by a 68000-based controller card, having 128k of memory (programmed in C). Both robots have binary tactile sensors which detect contact with nearby objects. In addition to this they are — when desired for certain experiments — equipped with forward motion sensors (a microswitch that is triggered by the passive caster wheel of the robots being aligned with the main axis of the robots), or an ultrasonic range sensor.

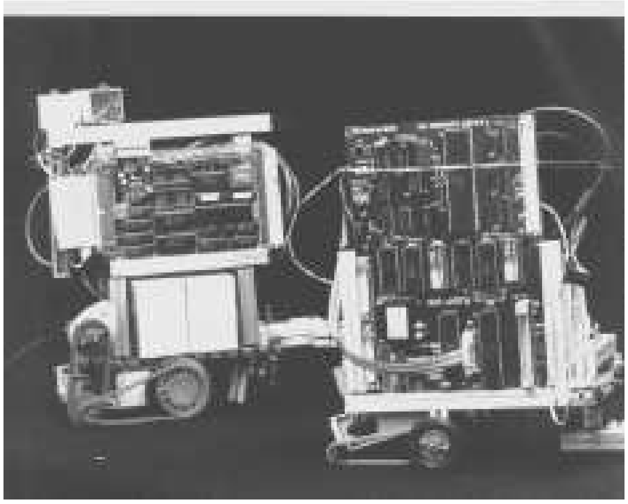

Figure 1: Alder (left) and Cairngorm.

2.2 The Controller Architecture

As mentioned in section 1, the goal of the experiments described in this paper was to achieve flexibility in unforeseen situations in mobile robot control. This, we believe, requires the robot's ability to learn from experience. We therefore chose to build a controller with a learning capability, so that the robot could determine for itself an effective wiring between sensors and actuators. In Alder's and Cairngorm's controllers these associations between sensors (input stimuli) and actuators (responses) are stored in a Pattern Associator (see below).

Figure 2 shows the general structure of the controller used in all experiments discussed in this paper. The controller consists of fixed and plastic components, fixed components being the instinct-rules, the robot morphology and various parameters within the controller; the plastic component is the Pattern Associator.

The Pattern Associator

The Pattern Associator is an artificial neural network that is easy to implement, low in computational cost and fast in learning; networks of this type are sometimes referred to as *linear associators* ([Kohonen *et al.* 81]). It consists of two layers of computational units: the input and the output layer (see figure 3). The fact that it learns very quickly is bought at the price that it can only learn functions that are linearly separable ([Minsky & Papert 88]). The functions Alder and Cairngorm had to learn in the particular experiments discussed in this paper are linearly separable (see [Nehmzow 92] for a discussion of this point), it was therefore possible to use the Pattern Associator in these experiments as an associative memory, associating stimuli with robot motor actions.

The function of input and output units is as follows: input units simply pass the received input signals $\vec{i}$ on to all output nodes, the output o_j of output unit j is determined

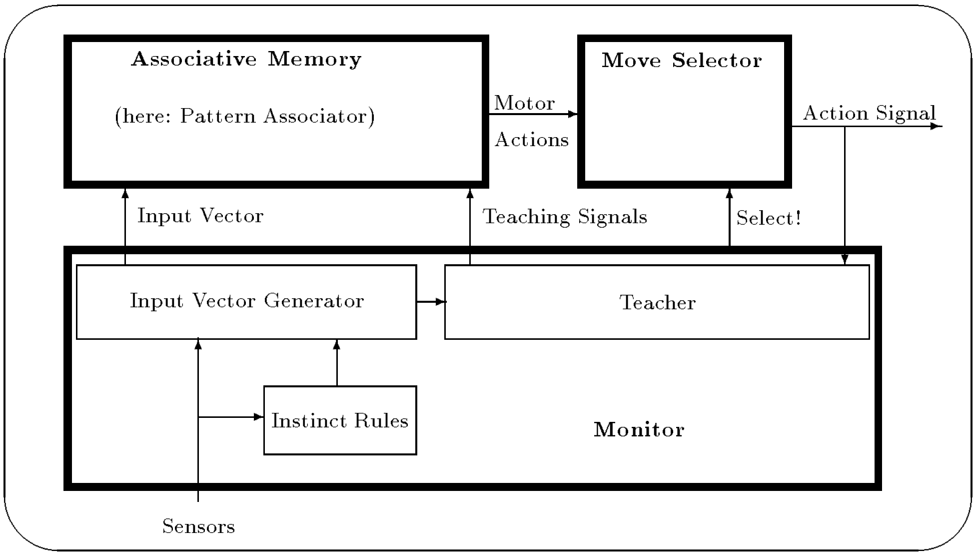

Figure 2: Computational Structure of the Self-organising Controller.

by

$$o_j = \sum_{k=1}^{M} w_{jk} i_k = \vec{w_j} \cdot \vec{\imath} \tag{1}$$

where $\vec{w_j}$ is the individual weight vector of output unit j and M the number of input units. If the desired output, the target vector $\vec{\tau}$, is known, updating the weight vectors is done according to the following rule:

$$\Delta \vec{w_j} = \eta(\tau_j - o_j)\vec{\imath} \tag{2}$$

$$\vec{w_j}(t+1) = \vec{w_j}(t) + \Delta \vec{w_j}. \tag{3}$$

η is the so-called gain or learning rate, a parameter that determines how big the changes to the weights are, and therefore how quickly the net learns. A big η (for example 0.8) will result in a network that adjusts very quickly to changes, but which will also be "neurotic" (it will forget all it has learned and learn something new as soon as a couple of freak signals occur). A small η (for example 0.1), on the other hand, will result in a "lethargic" network that takes a long time before it learns a function.

How the target vector $\vec{\tau}$ was determined in the experiments described in this paper is explained below.

2.3 Instinct-Rules

As the Pattern Associator is trained by 'supervised learning', a means of evaluating the robot's behaviour and determining teaching signals for the Pattern Associator had to be found. We use fixed rules for this purpose which we call *instinct-rules*. They are similar, but not identical to *instincts* as defined by [Webster 81]: an instinct is a "complex and specific response on the part of an organism to environmental stimuli that is largely hereditary and unalterable though the pattern through which it is expressed may be modified by learning, that does not involve reason, and that has as its goal the removal of a somatic tension or excitation". This describes behaviour and is therefore different to the instinct-rules used in the experiments described here, as instinct-rules are not behaviour, but constants (sensor states) that guide the learning of behaviour. The goal of instinct and instinct-rules, however, is the same: the removal of a somatic tension (in the case of the robot such a 'somatic tension' is an external sensor stimulus, or, in some experiments, the lack of it).

Each instinct-rule has a dedicated sensor in order that it can be established whether it is violated or not. This sensor can be a physical, external sensor (for example a whisker), or an internal sensor (for example a clock that is reset every time some external sensor stimulus is received). Even memory can serve as a trigger for an instinct-rule (touch alternate walls!).

2.4 Input and Output

Current and previous sensor signals constitute the input signals to the Pattern Associator. Information about violated instinct-rules could similarly be used; but this has not been done in the experiments described in this paper.

The output of the network denotes motor actions of the robot. Examples for such motor actions are swift left turn (i.e. right motor moving forward while left motor is moving backward — in other words a left turn on the spot), swift right turn, forward and backward movement.

The idea behind this controller setup is that effective associations between sensor signals and motor actions arise over time through the robot's interaction with the environment.

2.5 Determining the Teaching Signal (The Learning Mechanism)

As said earlier, the Pattern Associator requires a teaching signal to develop meaningful associations between its input and its output. This teaching signal is provided by the *monitor*, a "teacher" that uses the instinct rules to assess the robot's performance and teach the network accordingly: as soon as any of the instinct-rules become violated (i.e. as soon as a specified sensor status is no longer maintained), an input signal is generated by the *input vector generator*, sent to the associative memory and the output of the network is computed (fig. 2). The *move selector* determines which output node carries the highest output value and which motor action this output node stands for. That motor action is then performed for a fixed period of time. For Alder and Cairngorm a typical value is about four seconds. If the violated instinct-rule becomes satisfied within this period of time, the association between original input signal and output signal within the Pattern Associator is taken to be correct and is confirmed to the network (this is done by the monitor). If, on the other hand, the instinct-rule remains violated, a signal is given from the monitor to the move selector to activate the motor action that is associated with the second strongest output node. This action is then performed for a slightly longer period of time than the first one (in Alder's and Cairngorm's case the increase is typically two seconds) to compensate the action taken earlier; if this motor action leads to satisfaction of the violated instinct-rule the network will be taught to associate the initial sensor state with this type of motor action; if not the move selector will activate the next strongest output node. This process continues until a successful move is found. Because the monitor is part of the robot controller, the process of sensor-motor competence acquisition is completely independent from an operator's supervision: the robot acquires its competences autonomously.

The fact that motor actions are tried in turn and with increasing duration is important for making this skill acquisition scheme work. If, to give an example, the robot had two possible motor actions available (left and right turn) and was trying to touch a wall, due to a built-in boredom instinct-rule being triggered, it would initially turn left and perform this action for four seconds. Supposing the wall *was* on the left of the robot, but the robot did not touch it within the first four seconds, the right turn would be initiated and executed for six seconds. This would leave the robot pointing a further "two-seconds-worth" away from the wall. After these six seconds, the robot would again move to the left, this time for eight seconds, and touch the wall. This will lead to the association of a left turn with the boredom instinct-rule, which happens to be the correct association. So, although the very first left movement of the robot was unsuccessful, the robot nevertheless was able to discover the correct motor action.

Figure 3 shows the general structure of the Pattern Associator used. The actual input to the network may vary from experiment to experiment (the input vectors are described at the relevant places in this paper), the output nodes denote motor actions (the figure shows the more general case with four possible motor actions; usually a network with just two motor actions — left and right turn — is used).

The result of this process is that effective associations between input stimuli and output signals (motor actions) arise. The instinct-rules and sensors we used are described in more detail in the following sections, where we will discuss a number of experiments on motor competence acquisition.

3 Obstacle Avoidance

One of the most important tasks to achieve on a mobile robot is obstacle avoidance. When implemented on a robot, it becomes clear that even this seemingly simple task is not an easy one. Sensors give inconsistent readings, actuators depend on changeable factors such as supply current and voltage, wear, temperature, humidity and others. 'Avoiding Obstacles' is by no means a bygone research topic (see, for example [Freund *et al.* 91] and [IEEE 91]).

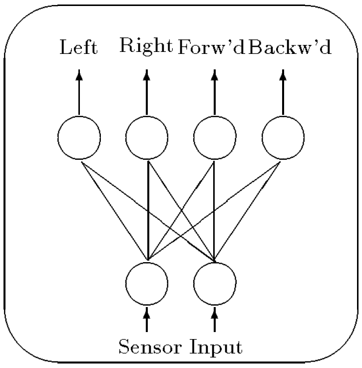

Figure 3: Association between sensor stimuli and motor actions.

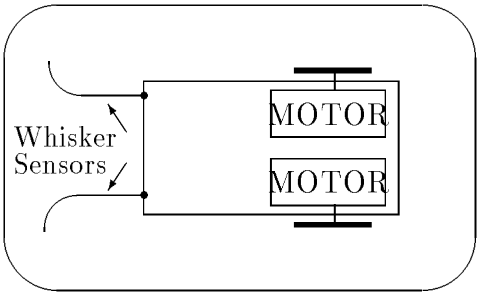

Figure 4: A simple vehicle.

Assume a robot had two whiskers mounted at the front of the vehicle and two independent motors that allowed the robot to move forward and backward as well as turn left or right. Such a robot is shown in figure 4.

If the task of this vehicle was to turn away from a touched whisker, that is to turn left if the right whisker is touched and vice versa, one could simply let the whiskers reverse the motors as shown in figure 5.

This would result in a vehicle avoiding obstacles. It could cope with new obstacles (it would, for example, move backwards if both whiskers were touched simultaneously, although the wiring may have been designed without this situation in mind), however it would not be able to cope with certain other changes. If, for example, its whiskers were swapped, the vehicle shown in figure 5 would move towards an obstacle instead of away from it!

As described earlier in this paper we have replaced the coupling between sensor signals and motor actions — box "A" in figure 5 — by an artificial neural network. It is therefore possible for the robot to determine the wiring between sensors and actuators for itself. As a result of this Alder and Cairngorm are able to cope with unforeseen situations, such as swapped whiskers. This example may seem far fetched and unlikely to occur in

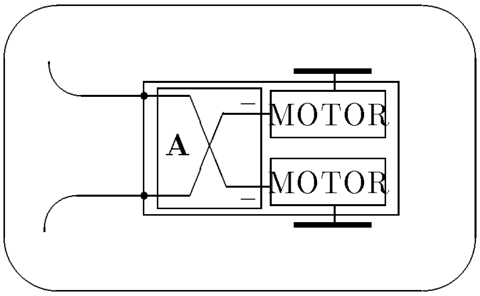

Figure 5: A simple obstacle avoiding vehicle.

reality, but the wiring of sensors can be error prone if a large number of sensors is to be connected. If the robot can determine the best wiring autonomously, malfunctions due to wrong sensor connections can be avoided. [Maes & Brooks 90]:

> "For more complicated robots prewiring solutions become either too difficult or impractical. ...Additionally, it is often too difficult for the programmer to fully grasp the peculiarities of the task and the environment, so as to be able to specify what will make the robot successfully achieve the task."

This is one of the experiments conducted: after having swapped the whiskers the robots are confused for a short time (typically they make about two wrong decisions before they are able to avoid obstacles again), but then they re-train the network and acquire the obstacle avoidance competence again[1].

We performed two experiments on obstacle avoidance — in the first one Alder and Cairngorm learn to avoid obstacles, using only their whisker sensors, in the second one Alder uses its whiskers, as before, but also its forward motion sensor and therefore learns both to move forward and to avoid obstacles.

3.1 Obstacle Avoidance without using the Forward Motion Sensor

In the first experiment, which was conducted both with Alder and with Cairngorm, the robots were not able to detect forward motion. The first instinct-rule, mentioned below, therefore could not be monitored[2]. The robots merely "assumed" that they were indeed moving forward when a 'move forward!' command was given.

The structure of the controller used for this experiment is shown in figure 6, the instinct-rules contained in the monitor are shown in figure 7.

The network had only two input nodes, the input vector presented to the network is shown in figure 8.

[1]Interestingly, [Waterman 1989, p. 135] reports that North American desert ants *Novomersor* are able to follow a pheromone trail, albeit crookedly, if their antennae (their chemoreceptors) are swapped.

[2]We have indicated this by writing this instinct-rule in brackets.

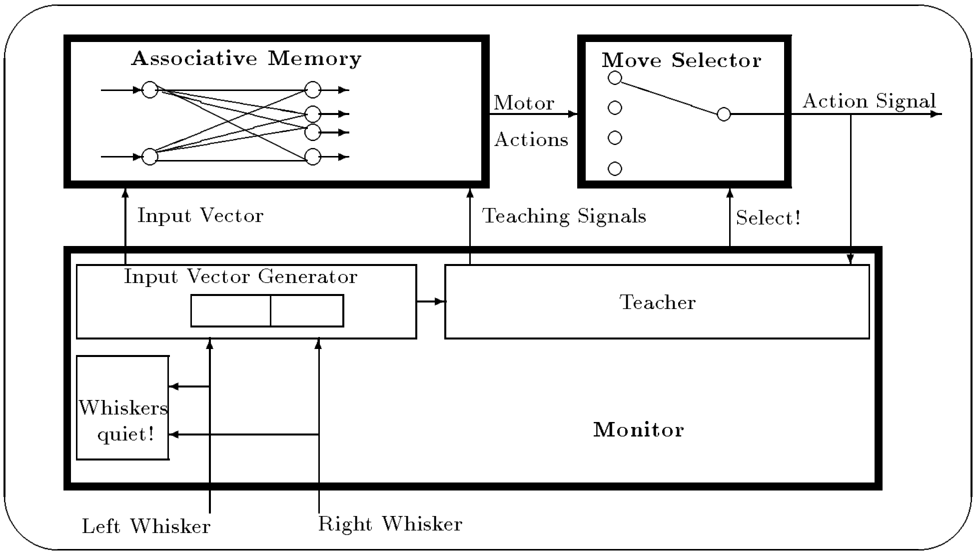

Figure 6: Controller used for obstacle avoidance I.

1. (Move forward!)
2. Keep whiskers 'quiet'!

Figure 7: Instinct-rules used for obstacle avoidance I.

Both robots learned the effective wiring that would make them satisfy their instinct-rules very quickly. Within two to three learning steps the robots were able to turn away from obstacles. Learning took much less than a minute.

Using these two instinct-rules for assessing their behaviour, and the input vector shown in figure 8 the robots are not only able to acquire the obstacle avoidance competence and to re-learn it in the case of something drastic changing, like, for instance, the position of their whiskers; the robots are also able to cope with changed environments.

One example of such a changed environment is that of Alder and Cairngorm first

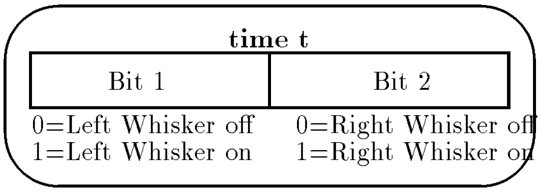

Figure 8: The input vector for obstacle avoidance behaviour I.

learning to avoid obstacles, and then encountering a dead end: although the robots enter the dead end with an obstacle avoidance behaviour (i.e., they turn in the opposite direction of a whisker that is touching something), they come out of the dead end turning in the same direction, regardless of which whisker is signalling. The effective wiring of the robots before and after the encounter of the dead end is shown in figure 9, we call the resulting behaviour *dead end escape behaviour*.

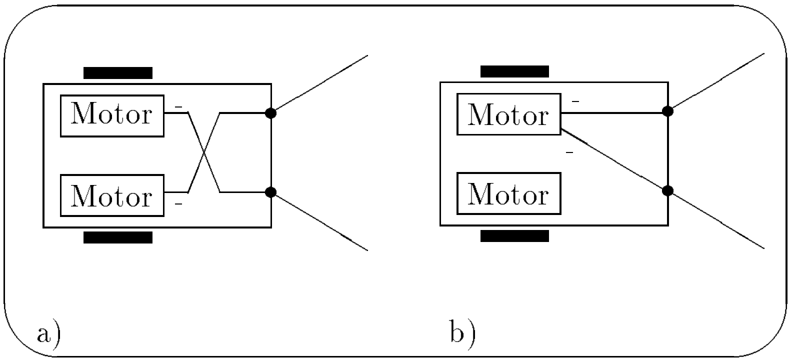

Figure 9: The effective wiring of Alder and Cairngorm before (a) and after (b) encountering a dead end.

3.2 Learning to Avoid Obstacles and to Move Forward

In the second experiment Alder was equipped with a sensor to detect forward motion. Both instincts shown in figure 11 were therefore monitored — the controller for this experiment is shown in figure 10.

As before, Alder was able to acquire the obstacle avoidance competence and cope with swapped whiskers. In addition to that the robot could now also cope with a reversal of the motor connections[3]. In fact, it now didn't matter which way round the motors were connected, nor did it matter which way the sensors were connected. Because of the self-organising structure of the controller Alder was able to find the right behaviour to meet its instinct-rules. The input vector used for this second experiment is shown in figure 12. As the Pattern Associator can only process non-zero inputs, a two bit encoding of the forward motion sensor is used. Strictly speaking, the whisker sensor signals ought to be encoded in two bits as well, but because the no-whisker-touched state is not to be associated with any motor action other than forward (which the signal from the forward motion sensor takes care of) this is not necessary. If a Perceptron with input signals of ± 1 was used, the whole question would not arise, of course.

[3]The "reversal" of motor connections results in the output units of the network standing for different motor actions. If, for example, the connections of each of the two motors are reversed in polarity, a previously "forward" move will have become a "reverse" move, and a left turn a right turn (and vice versa).

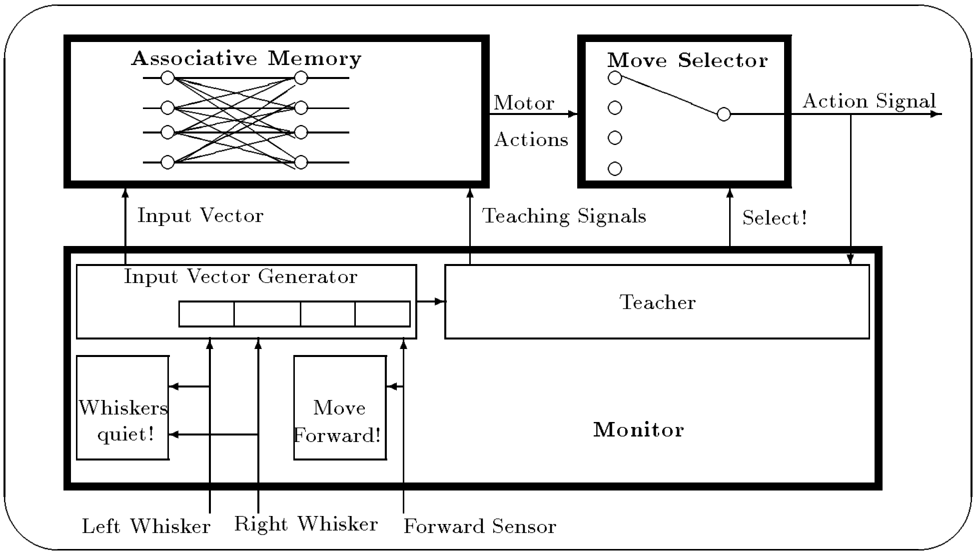

Figure 10: Controller used for Obstacle Avoidance II

1. Move forward!
2. Keep whiskers quiet!

Figure 11: Instinct-rules used for obstacle avoidance II.

Again, as in the experiment mentioned in the previous section, Alder quickly learned the effective wiring between sensors and actuators. If the caster wheel of Alder is not in the 'forward' position when the robot is switched on, Alder will try the motor action associated with the first output node for a while. If in the course of this the forward motion sensor comes on, all instinct-rules are satisfied and the robot will move forward until one of the whiskers touches something. When this happens, Alder determines the effective obstacle avoidance behaviour as in the previous experiment. Note that in this experiment the motor actions associated with each of the four output nodes depend on the wiring of the motor power supply. For all other experiments described in this paper the motors are wired up such that the nodes mean left turn, right turn, forward (or left turn) and backward (or right turn) in that order, but here deliberately the wires are connected arbitrarily, which means that the effect of each node has to be found out by the robot. To learn to move forward takes the robot about two to three learning steps (which again takes much less than a minute), to learn obstacle avoidance takes about the same amount of time.

Left Whisker	Right Whisker	1=Moving forward 0=not moving forward	1=not moving forward 0=moving forward

Figure 12: The input vector used for obstacle avoidance behaviour II.

4 Wall Following

A further advantage of a self-organising controller as shown in figure 2 is that the behavioural repertoire of the robot can quite easily be expanded by adding further instinct-rules (sometimes the input vector to the associative memory also needs to be changed).

The second experiment we conducted showed this: the robot's task still was to move away from obstacles, but in addition to that the robot now also had to stay close to a wall and follow it. This was achieved by using one additional instinct-rule, "boredom", so that the whole set of instincts looked like figure 13.

1. (Move forward!)
2. Keep whiskers quiet!
3. Make whiskers signal after a while!

Figure 13: Instinct-rules used for wall following

The input vector used now contained some history, it is shown in figure 14, the controller for this experiment is shown in figure 15.

Time t		Time t-1		
1=Lft. Wh. on 0=Lft. Wh. off	1=Rgt.Wh. on 0=Rgt.Wh. off	1=Lft. Wh. on 0=Lft. Wh. off	1=Rgt.Wh. on 0=Rgt.Wh. off	$\neg(B1 \vee B2 \vee \ldots$ $\ldots \vee B3 \vee B4)$
B1	B2	B3	B4	B5

Figure 14: Input vector used for wall following and corridor following.

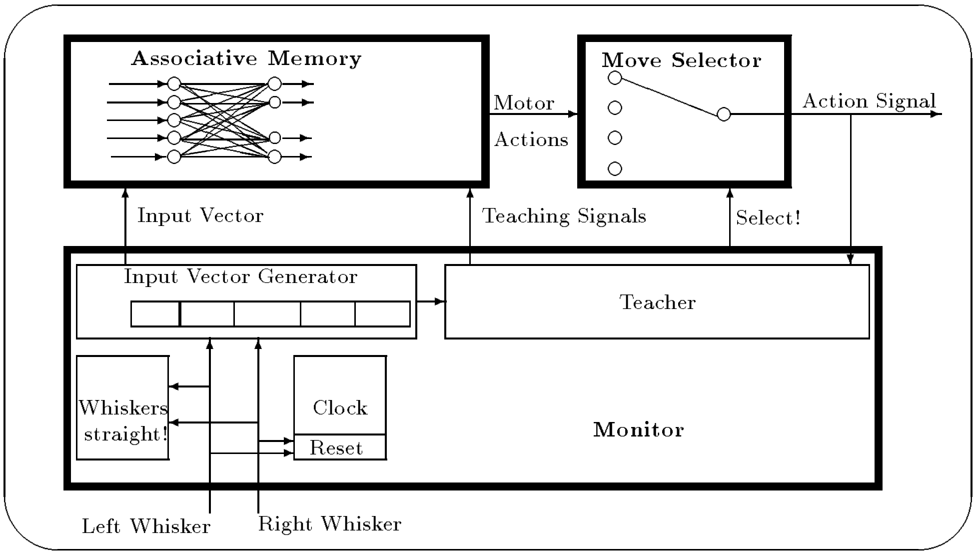

Figure 15: Controller used for wall following and corridor following.

Using this set of instinct-rules and this input vector, the robots were able to learn to follow a wall successfully. When the robot is started, having been placed near a wall, but not touching it, the internal clock that triggers instinct-rule number 3 (see figure 13) is reset. This means that neither instinct-rule number 2 nor number 3 trigger, but number 1 does: the robot starts moving forward. After about four seconds the clock triggers rule number 3, and a wall seeking movement is initiated. Eventually the robot will touch the wall, thus satisfying rule 3 (and taking it off the "agenda"). Immediately, however, rule 2 will be violated, and the robot will start an obstacle avoidance action. Every time an instinct-rule is violated, an input vector as shown in figure 14 is generated and a motor action sought that will satisfy this particular instinct-rule. It does not matter if in the course of trying to satisfy this one instinct-rule other rules are violated, too. The robot attends to the first violated instinct-rule until it is satisfied again, and then continues.

The experimental result of this was that after very few teaching signals Alder and Cairngorm learned which way to turn in order to touch the wall, and which way to turn in order to get away from it. As in the previous experiment in obstacle avoidance, the robots learn the correct moves after about two teaching steps. The whole wall following competence is acquired in under one minute.

Because of their ability to acquire the wall following competence, the robots are not dependent on any particular wiring or experimental setup. The way the whiskers are connected to the interface and the side on which the wall is can be chosen arbitrarily, the robots will nevertheless learn how to follow the wall. The wall following experiment shows another advantage. Because of the structure of the controller, without any especially given knowledge, the robot can follow walls on either its left or its right hand side. This makes the robot more flexible, and better able to cope with situations such as corridors, where the wall on one side suddenly stops (because of a door, for example), but continues on the other side. At such places the robot would be temporarily "confused", but then learn to follow the wall on the opposite side and continue its path. Also, the experiment shows that the behavioural repertoire of the robots can easily be expanded by adding instinct-rules, an observation that will be amplified by the following experiment (see also [Nehmzow 93]).

5 Corridor Following

As just stated in section 4, it is easy to extend the behavioural repertoire of the robot by adding further instinct-rules. In order to obtain a corridor following behaviour, literally the only thing we had to do was to add a fourth instinct-rule (about five lines of program code) — the input vector used was the same as the one used for wall following, shown in figure 14, so was the controller architecture. The set of instincts to acquire the corridor following competence are shown in figure 16.

---

1. (Move forward!)
2. Keep whiskers quiet!
3. Make whiskers signal after a while!
4. Make alternate whiskers signal!

Figure 16: Instinct-rules used for corridor following.

The robot now learns which way to turn to touch alternate walls of a corridor, and which way to turn to move away from either wall. As before, learning is fast, the robots acquire the corridor following competence in less than a minute. This last experiment demonstrates how easy it can be to "program" the robot to do new tasks, using the self-organising controller described here.

6 Phototaxis and Boxpushing

In order to show how the behavioural repertoire of a robot can be expanded to more complicated tasks, changing the instinct-rules only, we conducted experiments with an *R2* mobile robot, which was equipped with light sensors and range finding infrared sensors (for a description of this robot see [Nehmzow & McGonigle 94]).

Phototaxis was acquired within less than ten learning steps, using an instinct-rule stipulating that the light sensors mounted at the front of the robot must return the highest value (i.e. face the brightest light).

Similarly, by using a instinct rule requiring the front facing infrared sensors to be "on" constantly (i.e. to return sensor values that indicate an object was situated in front

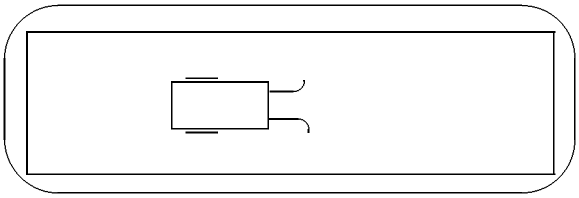

Figure 17: The experimental setup for context-dependent learning.

of the robot), the robot acquired a box-pushing or object following competence in two learning steps, one for each of the two situations where the box is placed to the right or the left of the robot's center respectively. The learning time to acquire a box-pushing competence is less than one minute.

7 Context-dependent Learning

In section 3.1 we described an experiment in which the robot had first learned to avoid obstacles, and then encountered a dead end. The obstacle avoidance behaviour is badly suited for leaving dead ends, because inevitably there will be the situation where the robot turns back into the dead end instead of turning away from it. Alder and Cairngorm discover this through the interaction with their environment and change their behaviour: after having encountered a dead end they turn in the same direction regardless of which whisker is touched. This is what we called dead end escape behaviour.

In the process of acquiring the dead end escape behaviour the previously acquired knowledge about obstacle avoidance is replaced by knowledge about dead end escape. Consequently, when Alder and Cairngorm leave a dead end and encounter an obstacle, they have to re-learn obstacle avoidance behaviour. They lack a notion of context. In this section we describe an experiment done on context dependent learning in which Cairngorm learns to distinguish two different contexts and behave appropriately in both.

The experimental setup is as shown in figure 17. The robot was placed in an enclosure which consisted of two dead ends and a corridor in between. The robot's task was to follow the corridor and to get out of the cul-de-sacs, resuming the corridor following behaviour.

To find out whether contextual information could be exploited by the robot we conducted the following experiment. Cairngorm was controlled by the same controller that was previously used to acquire the corridor following competence, as described in section 5. The only difference to the controller described there was an additional bit in the input vector to the Pattern Associator (see figure 18).

This last bit in the input vector was set whenever a switch on the robot was on, and it was reset whenever the switch was off. As this switch was manually operated, the operator could identify different contexts and make them available to the learning algorithm.

Time t		Time t-1		
1=Left Wh. on	1=Right Wh. on	1=Left Wh. on	1=Right Wh. on	Switch
0=Left Wh. off	0=Right Wh. off	0=Left Wh. off	0=Right Wh. off	

Figure 18: Input vector for context–dependent learning.

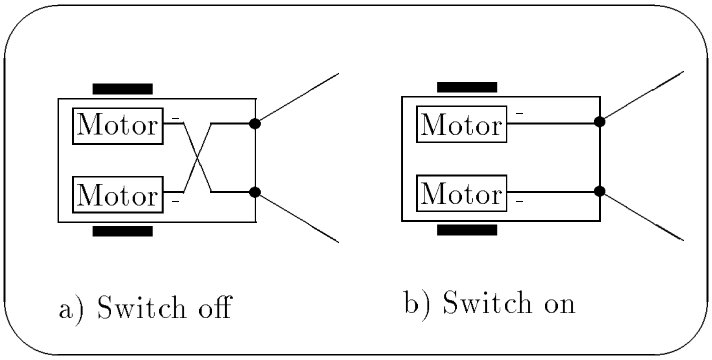

Figure 19: Results of context–dependent learning experiment.

7.1 Experimental Results

The robot was placed in the enclosure shown in figure 17. Whenever the robot was in one of the two dead ends, that is in a dead end context, the switch was on. Whenever the robot was in between dead ends, that is in a corridor following context, the switch was off. After the robot had learned corridor following (and, in the process, encountered dead ends several times), the effective wiring of Cairngorm was analysed. The results are shown in figure 19: whenever the switch is off (corridor following context), the robot turns left when the right whisker is on, and right when the left whisker is on (figure 19a). When the switch is on (dead end escape context) the behaviour is reversed: the robot turns towards a touched whisker (figure 19b). Both these behaviours are appropriate for their respective contexts. When following a corridor, whiskers will only be on after a wall seeking movement. A turn away from a touched whisker will take the robot back into the middle of the corridor. In the dead end context this is different, here it is a turn towards a touched whisker that will take the robot to the exit fastest.

The purpose of this experiment was to investigate whether the controller could successfully exploit contextual information. This is why the robot was provided with contextual information. A whole series of experiments is conceivable in which this contextual information is determined by the robot itself, not by the operator. Here are some examples: when the robot follows a corridor, it is very unlikely that the same whisker will signal twice consecutively, because the robot turns left, say, but then turns back right and when initiating the next wall seeking move will turn right, therefore the last touch will have been on the left, the current one on the right. The only situation where the same whisker will signal more than once consecutively is if the robot is in a dead end: the fact that one whisker has signalled twice in a row can therefore be used as a detector for a dead end context. Similarly, if the robot had three front whiskers, one mounted at the left, one in the middle and one at the right, then the middle whisker would normally only be on in a dead end, not in a corridor. Instead of a middle whisker, a central range finding sensor (e.g. sonar) would serve the same purpose.

8 Discussion

Our experiments show that it is possible to let robots acquire basic sensory-motor skills autonomously, and that this does increase the robot's ability to cope with unforeseen situations, such as changes in the environment, the task or the agent itself. These results are not dependent of specific hardware or software configurations. Independent research, using the self-organizing controller described in this paper, has been conducted in Edinburgh ([Daskalakis 91]), Brussels ([Ramakers 93]) and Bremen ([Körtge & Monnerjahn]), with similar findings.

Some questions, however, are not yet conclusively answered. The instinct-rules used in the experiments here, for instance, have been arrived at by "designer's intuition", and the use of three simple rules:

1. Instinct-rules are always associated with a sensor which monitors or measures some specified condition. Sensors in this sense can be:
   - Physical sensors (e.g., whiskers),
   - internal sensors (e.g., a clock), and
   - memory (e.g., "Make alternate whiskers signal!").
2. Instinct-rules are not behaviours. "Move towards the light!", therefore, is not an instinct-rule, but "Increase the reading from a light sensor!" is. The most suitable instinct rules, in our opinion, are binary ones, i.e. instinct rules which are either satisfied or violated. If, for example, light following is the desired behaviour, a good instinct rule would be "The light sensor at the front must see the brightest light", rather than "Keep as close as possible to the light".
3. Instinct-rules generate the desired behaviours, the question therefore must be "Which sensor signal can be associated with the desired behaviour?".

The question, however, whether the determination of instinct rules can be expressed in algorithmic form or even be automated is as yet open.

We have not yet found a natural limit in the number of independent competences the robot can acquire in the way described here (see [Nehmzow 93]); how competences that interfere with each other can be accomplished is another open research area.

Other issues concern the choice of "magic numbers" such as the learning rate (see, concerning this, [Ramakers 93]), the size of network, the encoding of sensor signals and the encoding of motor actions. We intend to address these questions in future work.

Acknowledgements

Part of the work reported here was supported by a grant from the UK Science and Engineering Research Council (GR/F/5852.3). Other support and technical facilities were provided by the Department of Artificial Intelligence at the University of Edinburgh. The work reported in section 6 was conducted by the author at the Department of Psychology, Edinburgh University.

I thank my colleagues, in particular Tim Smithers, for stimulating and helpful discussions. John Hallam and Peter Forster were also involved in the funded project, and I acknowledge their contribution.

The publication of this report was supported by the Department of Computer Science at Manchester University. I thank my colleague Jon Shapiro for helpful discussions regarding the final version of this report.

References

[Colombetti & Dorigo 93] Marco Colombetti and Mario Dorigo, *Learning to Control an Autonomous Robot by Distributed Genetic Algorithms*, in J.A. Meyer, H. Roitblat and S. Wilson (eds.), *From Animals to Animats 2*, MIT Press 1993.

[Daskalakis 91] Nikolas Daskalakis, *Learning Sensor-Action Coupling in Lego Robots*, MSc Thesis, Department of Artificial Intelligence, Edinburgh University, 1991.

[Freund *et al.* 91] E. Freund, R. Mayr, F. Dierks, U. Judaschke, U. Kernebeck, B. Lammen, *Safety Aspects for Autonomous Robot Systems*, in G. Schmidt (ed.), *Information Processing in Autonomous Mobile Robots*, Springer Verlag, Berlin, Heidelberg, New York, 1991.

[IEEE 91] Proc. IEEE Conference on Robotics and Automation, Sacramento 1991, pp. 674-701, 898-925.

[Kaelbling 90] Leslie Pack Kaelbling, *Learning in Embedded Systems*, Stanford Technical Report TR-90-04, June 1990.

[Kaelbling 91] Leslie Pack Kaelbling, *An Adaptable Mobile Robot*, Proceedings of 1st European Conference on Artificial Life 1991, MIT Press Cambridge Mass. and London, England, 1991.

[Körtge & Monnerjahn] N. Körtge and J. Monnerjahn, Forschungsberichte Informatik der Universität Bremen, No. 7/92, ISSN -0722-8996, 1992.

[Kohonen *et al.* 81] T. Kohonen, P. Lehtio and E. Oja, *Distributed Associative Memories*, in G. Hinton and J. Anderson (eds.), Lawrence Erlbaum Publishers, 1981.

[Maes & Brooks 90] Pattie Maes and Rodney Brooks, *Learning to Coordinate Behaviors*, Proc. AAAI 1990.

[Mahadevan & Connell 91] Sridhar Mahadevan and Jonathan Connell, *Automatic Programming of Behavior-based Robots using Reinforcement Learning*, 9th National Conference on Artificial Intelligence, AAAI 1991.

[Minsky & Papert 88] Marvin Minsky and Seymour Papert, *Perceptrons*, MIT Press Cambridge Mass. and London, England, 1988.

[Nehmzow *et al.* 89] Ulrich Nehmzow, John Hallam and Tim Smithers, *Really Useful Robots*, in T. Kanade, F.C.A. Groen and L.O. Hertzberger (eds.), *Intelligent Autonomous Systems 2*, Proceedings of IAS 2, ISBN 90-800410-1-7, Amsterdam 1989.

[Nehmzow 92] Ulrich Nehmzow, *Experiments in Competence Acquisition for Autonomous Mobile Robots*, Ph.D. thesis, Edinburgh University, 1992.

[Nehmzow 93] Ulrich Nehmzow, Tim Smithers and Brendan McGonigle, *Increasing Behavioural Repertoire in a Mobile Robot*, in: J.A. Meyer, H. Roitblat & S. Wilson (eds.), *From Animals to Animats 2*, MIT Press 1993.

[Nehmzow & McGonigle 94] Ulrich Nehmzow and Brendan McGonigle, *Achieving Rapid Adaptations in Robots by Means of External Tuition*, Proc. "Simulation of Adaptive Behaviour 3", Brighton, August 1994.

[Ramakers 93] Wilfried Ramakers, *Investigation of a Competence Acquiring Controller for a Mobile Robot*, Licentiate thesis, Vrije Universiteit Brussel, 1993.

[Sutton 91] Richard S. Sutton, *Reinforcement Learning Architectures for Animats*, Jean-Arcady Meyer and Stewart Wilson (eds.), *From Animals to Animats*, Proc. 1st Intern. Conf. on Simulation of Adaptive Behaviour, MIT Press, Cambridge Mass. and London, England, 1991.

[Waterman 1989] Talbot H. Waterman, *Animal Navigation*, Scientific American Library, New York 1989, p.183.

[Webster 81] *Webster's Third New International Dictionary*, Encyclopaedia Britannica Inc., Chicago 1981.