The Prospective Student's Introduction to the Robot Learning Problem

Why is "learning" in robots an interesting problem, and why should it be tackled using real robots, rather than numerical models of them? The purpose of this paper is to give an introduction to the problems research in robot learning addresses, to present some attempts at solving them, and, most of all, to point out what remains to be done.

This means that the paper can be read in a number of ways: firstly, by the interested novice as an introduction to the field of robot learning. Secondly, as a survey of work done so far. And thirdly, by the prospective student as a guide towards interesting and unsolved problems within the field of robot learning.

The paper first gives an introduction to robot learning, discussing the actual goings-on inside a robot as well as the additional components needed to make the robot do what it is supposed to do. It then introduces the idea that for complex tasks performed in semi-structured environments, *learning* is necessary to make the robot successful in performing a specified task. A tentative curriculum of robot learning is presented.

The paper then discusses reinforcement learning in particular, because this learning method is, to date, the most widely used learning paradigm applied to robotics. We then suggest mechanisms to improve the performance of reinforcement learning systems. The paper concludes by discussing unsolved problems, the "good thesis topics" the prospective student has been waiting for.

1 Introduction

Figure 1 shows a mobile robot[1], equipped with various sensors to perceive its environment, and actuators that allow it to perform actions in the world. The robot has a particular task to achieve, this is what the box labelled "task description" is meant to signify.

The aim of this paper is to replace the questionmark in figure 1 with some appropriate structure. In order to do this the following two questions need to be answered:

1. What happens inside the robot, after sensor signals have been received and before actuator signals have been generated, independent of the particular control strategy adopted?

2. Which functions ought to be computed by the controller in order to make the robot achieve a specified task?

[1]The term "robot" is used to describe a physical device, interacting with the real world, that has the following three properties:

1. the ability to measure properties of the environment through sensors,
2. the ability to perform physical actions of an object-manipulation or locomotion type in the environment, and
3. presence of a control structure that determines the action to be performed at any given moment.

I can see at least two ways to approach the second question. One way to determine the adequate control structure would be to model as closely as possible the known aspects of the control strategies used by the most successful agents on earth, living beings. The purpose of such an approach is to verify or refute models of animal cognition.

For the purpose of this paper I adopt a different approach: the primary interest here is to establish ways to make the robot perform a given task as reliably and accurately as possible. Biologically inspired control strategies may be used for this purpose, but anything else that works well will be acceptable, too. I call this the engineering approach to robot learning.

2 What goes on?

The fundamental principles of robot-environment interaction are depicted in figure 2; I call them fundamental, because they are independent of a particular controller implementation.

As the robot interacts with the environment, certain properties of this environment are sensed with the robot's sensors. What makes the robot learning problem so much more difficult than other (non-robot) machine learning problems is the fact that there are a number of unknowns: the true state of the world is unknown, as is the function g that maps true world state on to perceived world state; g is a function of many variables, such as sensor characteristics, environment characteristics, and characteristics of the information processing devices used.

At this point I propose to take a small detour, to look at the problem of numerical modelling ("simulation"). This will help to answer the question whether it is possible to model g accurately, and whether physical experiments could be abandoned altogether and numerical models be used instead (which would be faster).

Numerical modeling

Verification (establishment of truth) and validation (establishment of legitimacy) are *impossible* for any system other than a closed one ([Oreskes *et al.* 94]). *Every* natural system is open, therefore numerical models of natural systems can never be verified or validated (using the above definitions), merely confirmed (having their probability supported by observations).

This fact has the following implications on the robot learning problem:

- True (i.e. verified) numerical modeling of robot perception is impossible,
- true (i.e. verified) numerical modeling of robot action is impossible, and
- it is not possible to build a verified numerical model of the interaction between robot and environment.

When formalizing the robot learning problem, we are always faced with two issues: firstly, there is the classical machine learning problem (what should the association between perceived state and action be?) and secondly, what is the relationship between *actual* state (of world and robot) and *perceived* state?

3 What else should go on?

In order to define what *should* go on, it is necessary to define the objectives of the process. This is the "task description" of figure 1. The most precise way to describe the objectives of the robot is to define sensor states that are to be achieved, and sensor states that are to be avoided. This is a description of goals at the level of robot hardware, and may not be suitable to describe all conceivable goals (for instance, some goals may only be describable by multiple sensor states, including past sensor readings). However, when it is applicable it provides a precise and deterministic way of encoding what the robot is supposed to achieve.

It follows from having defined the goals of the robot's actions that its success is measured in terms of these goals. This is important: in section 5.1 I will discuss how the performance of robots depends on the task description. For the purpose of this paper, I measure success in terms of specified goals, not in terms of 'elegance of solution' or other performance-related criteria.

The one component of figure 2 which we can fill with whatever function we wish is the block labelled "control function f". This function f maps perceived state of robot and world on to actuator commands. It may (and usually will) contain sensor proprocessing commands, as well as the encoding of the control strategy.

Our goal is to make the robot perform a specified task in its environment in as reliable and efficient a manner as possible. Our means is to select f accordingly.

There are at least two choices for selecting f. The simplest way is to preprogram a fixed strategy that maps perception to action, a "hardwired" control program. This works satisfactorily for many basic robotics applications. However, as the complexity of the task (or of the environment) increases, the *problem of perceptual discrepancy* also increases. This is the problem that the human designer inevitably has a very different perception of the world compared with the robot's view, a fact that makes him less and less able to determine a priori which of the sensor signals will be relevant for a particular task, and which won't. Eventually, this discrepancy between human and robot perception can only be resolved through trial and error.

To give an example: in 1989, I used simple mobile robots for experiments in robot learning. The robots were able to learn fundamental motor skills such as obstacle avoidance by associating perceptions with motor responses in an artificial neural network ([Nehmzow *et al.* 89]). One would rightly argue that to make a robot with two whiskers avoid convex obstacles, a simple hardwired control function of "move away from the side where a whisker touches" would suffice. However, because of their learning capability these robots were able to escape from dead ends, even though I had never anticipated this scenario (perceptual discrepancy problem!): they quickly learned to keep turning in one direction whenever *any* whisker touched ([Nehmzow 94]).

The conclusion is that the more complex task, agent or environment are, the more unsatisfactory it is to use hardwired control functions. A second way of answering the question for an adequate control function f is, therefore, to *make the robot learn*.

4 Learning

The robot learning problem is to design a robot so that it improves its performance through experience. To be precise, we must specify what *performance* and what *experience*.

Suppose the robot's performance is to be evaluated in terms of its ability to achieve some set of goals G. More precisely, suppose each goal is of the form $X \rightarrow Y; R$, where X and Y are both conditions representing a set of possible states, and where R is some real valued reward. The goal $X \rightarrow Y; R$ is interpretted as "if the robot finds itself in a state satisfying condition X, then the goal of reaching a state satisfying condition Y becomes active, for which a reward R is received." For example, the goal of recharging the battery when it is low can be represented in this way, by setting X to the sensory input "battery level is low," Y to the sensory input "robot senses that it is electrically connected to the battery recharger," and R to 100[2]. Given a set of such goals, we can define a quantitative measures of robot performance such as the proportion of times that the robot successfully achieves condition Y given that condition X has been encountered, or the sum of the rewards it receives over time. If we wish, we might further elaborate our measure to include the cost or delay of the actions leading from condition X to condition Y.

Given this definition of *robot performance*, relative to some set of goals G, we can say that the robot learning problem is to improve robot performance through experience. Thus, robot learning is also relative to the particular goals and performance measure. A robot learning algorithm that is successful relative to one set of goals might be unsuccessful with respect to another. Of course we are most interested in general-purpose learning algorithms that enable the robot to become increasingly successful with respect to a wide variety of goal sets.

4.1 What and How to Learn

What and how should we design robots to learn? Two important dimensions along which approaches vary are (1) the exact function(s) to be learned and (2) the nature of the training information available. Here we consider a few possible learning approaches, then summarize some of the more significant dimensions of the space of possible approaches.

The most direct way to attack the robot learning problem is to learn the control function f directly, from training examples corresponding to input-output pairs of f. Recall that f is a function of the form $f : S \rightarrow A$, where S is the perceived state, and A is the chosen control action. One system that learns f directly is Pomerleau's ALVINN system. It learns the control function f for steering a vehicle driving at 65 mph. The system learns from training data obtained by watching a human steer the vehicle for a few minutes. Each training example consists of a perceived state (a camera image of the road ahead), along with a steering action (obtained by observing the human driver). A neural network is trained to fit these training examples. The system has been successful in learning to drive on a variety of public roads.

In other cases, training examples of the function f might not be directly available. Consider, for example, a robot with no human trainer, with only the ability to determine when the goals in its set G are satisfied and what reward is associated with achieving that goal. For example, a robot with the goal of docking to the battery charger might require a sequence of many actions to succeed. However, if it has no external trainer to suggest the correct action as each intermediate state, its only training information will be the delayed reward it eventually achieves when the goal is satisfied. In this case, it is not

[2]Note the importance of the condition X in stating the goal – without it we could only represent "full-time" goals such as "*always* seek to be electrically connected to the battery charger."

possible to learn the function f directly because no input-output pairs of f are available. An alternative approach that has been used successfully in this case is to learn a different function called Q, and use it to indirectly compute the control function f. The Q function is a mapping of the form $Q : S \times A \rightarrow V$, where V is the value of performing action A in state S. More precisely, the value V of a pair S, A is the expected sum of future rewards (each discounted exponentially by its delay) obtained by executing action A from state S, then following optimal actions from there forward. In short, the Q function is simply an evaluation function over $S \times A$. The control function f can then be computed from Q as $f(s) = max_{a \in A} Q(s, a)$. How can the robot obtain training values for learning Q? Given a state s_t at some time t and action a_t, the robot can estimate the training value of $Q(s_t, a_t)$ by performing action a_t to obtain the next perceived state s_{t+1}. It can then estimate the training value for $Q(s_t, a_t)$ as $\max_{a \in A} \hat{Q}(s_{t+1}, a)$ where $\hat{Q}$ is its currently learned approximation to Q. In this way, the currently learned evaluation function $\hat{Q}$ applied to state s_{t+1} is used to estimate training values for the preceding state s_t. Surprisingly, it can be shown that under certain conditions, this method converges to the optimal control procedure [?]. This approach of Q learning is one style of *reinforcement learning*. In general, reinforcement learning involves learning to control the robot based on a possibly delayed scalar training reward, rather than training examples specifying the desired action for each state.

Figure 3 shows the general robot reinforcement learning scenario: the mapping between perceived state and motor commands is acquired by the learner, perhaps by learning a different function such as Q. Learning is based on a reinforcement training signal received from the teacher to alter its control policy f[3].

The above examples illustrate two key dimensions of the robot learning problem: the nature of training information available to the learner, and the choice of specific target function to be learned.

4.2 The Target Function

Above we saw two ways to formulate the robot control learning task: learning the function $f : S \rightarrow A$ and learning the function $Q : S \times A \rightarrow V$. Other functions that might be learned and relevant include

- An evaluation function over states, $H : S \rightarrow V$ where V is the expected sum of discounted future rewards acheivable beginning at state S. Defined in terms of the earlier Q function, $H(s) = \max_{a \in A} Q(s, a)$. This evaluation function, like the Q function, can be used to compute the desired control function f. It has been used, for example, in Sutton's Adaptive Heuristic Critic /citesutton. However, computing f from H requires the ability to predict the effects of actions before executing them, whereas computing f from Q does not require this ability.

- A next state function to predict the effect of actions on the state, $NextState : S \times A \rightarrow S'$, where S' is the state resulting from applying action A to state S. Given a perfect next state function, one could compute f via look-ahead search to determine which action leads to the greatest reward. Even an imperfectly learned $NextState$ has been shown to be helpful background

[3] In figure 3 the teacher is part of the robot controller, enabling the robot to learn autonomously. Alternatively, the teacher could be outside, in which case the robot would learn from external feedback. A robotics implementation of such a system is discussed in [Martin & Nehmzow 95].

knowledge to improve learning the function Q [O'Sullivan *et al.* 95]. One intriguing property of $NextState$ is that it is task-independent (i.e., unlike f, Q, and H, it is not specific to any goal or reward function). Therefore a robot that learns $NextState$ can use it regardless of the current goal. A second nice property of $NextState$ is that it is particularly easy to collect training examples, since each action performed by the robot provides a training example of this function.

- A function to map from raw sensory input and actions to a useful representation of the robot's state, $Perceive : Sensor^* \times A^* \rightarrow S$. This corresponds to learning a useful state representation, where the state S is computed from possibly the entire history of raw sensor inputs $Sensor^*$, as well as the history of actions performed A^*. It appears particularly difficult to obtain direct training example of this function. While no current systems successfully learn this function, it is possible that recurrent neural networks might be relevant to learning this.

Other functions are possible as well. The issue of what collection of target functions should be learned by a robot is one of the key open research questions regarding robot learning. It is closely related to the question of what is the best general architectural for a robot. The above list is intended to suggest some of the possibilities, but is by no means complete.

4.3 Available Training Information

As is clear from the above discussion, the type of training information available has a strong impact on the choice of learning method. The nature of the training data available can be understood along two orthogonal axes: supervised versus unsupervised learning on the one hand, and learning by immediate feedback versus learning by delayed feedback on the other hand.

In the general case of *supervised learning*, training information about values of the target function are presented to the learning mechanism. A prototypical example is the supervised training data obtained from the human trainer for the ALVINN system described above. The backpropagation algorithm for training artificial neural networks is one common technique for supervised learning ([Rumelhart]). The training information provided in supervised control learning may be examples of the action to be performed, as in ALVINN, or more limited information in the form of a scalar, performance-related reward, not indicating the correct action itself ([Sutton 91]). This is the case in reinforcement learning as described above. Note that in supervised learning, the supervisor providing the training information may be either an external trainer (as in ALVINN), or a module within the robot itself (as in situations where the robot assigns rewards to goal states). We refer to the later as *self-supervised* learning, because one component of the robot is acting a a teacher for another component.

Unsupervised learning, on the other hand, performs a clustering of incoming information without using input-output pairs for training. Kohonen's self-organizing feature map is a well known example of an unsupervised learning mechanism ([Kohonen 88]). Whilst this paper is concerned with making robots achieve specified goals (implying that supervised learning will be the chosen method), unsupervised learning also has a role to play here. For example, unsupervised learning can identify clusters of similar data points, enabling the data to be re-represented in terms of more orthogonal features. This reduces the effective dimensionality of the data, enabling more concise data representation and supporting more accurate supervised learning.

Learning by *immediate reward* has been applied to robot control for a number of decades by now. Using electronic circuits whose characteristics were altered through operant conditioning (instrumental learning), Walter, in the 1950s, had small mobile robots learn to move towards light sources ([Walter 5?]). Using artificial neural networks, rather than electronic circuits as in Walter's case, mobile robots have learned a variety of different tasks in addition to phototaxis, for example obstacle avoidance, wall following or box pushing ([Nehmzow 94]).

As we have seen, training feedback for control learning may be *delayed* (e.g., the reward is provided only at the end of a long sequence of actions leading to the goal), or *immediate* (e.g., the reward or desired action is provided at each step in the sequence leading to the goal). In general, it is more difficult to learn from delayed feedback because the system faces the *credit assignment problem* is the reason for this: how much did each action of the sequence of actions actually contribute towards accomplishing the desired goal (e.g., [?], [Connell & Mahadevan 93]). Standard ways of addressing this problem of learning by delayed rewards are the bucket brigade ([Samuel 59, Holland *et al.* 86] algorithm or Q-learning ([Watkins 89]). Some attempts have been made to make real robots learn from delayed rewards, but the number of training examples required is substantially more in this case, leading some researchers to turn to computer simulations ([Kaelbling 90]), and others to seek new methods to overcome this problem ([?], [O'Sullivan *et al.* 95]).

4.4 The robot learning curriculum

Learning aims for lasting change in order to achieve a particular goal. It therefore requires a *curriculum*, an agenda of what is to be learned, and in what order. Is it possible to identify a curriculum of robot learning?

At the bottom level, there is probably the learning of relexive sensor-motor associations, expressed in competences such as obstacle avoidance, wall following, and following attractors. Competences at this level are state of the art in mobile robotics by now (see [Nehmzow 92] for a review). How to combine several fundamental competences to achieve more complicated tasks, how to identify components of plans and how to assemble them to form complete plans, these are elements of the same curriculum, at a higher level.

Here is an attempt to define a curriculum of robot learning:

- Reflexive actions
- Attentive competences
  - Detection of salient features of the environment
  - Detection of correlation in perceptual patterns over time
  - Ability to focus attention
  - Differentiation between relevant and irrelevant events with respect to the current goals
- Identification of sequences of actions as plan components
- Planning capabilities
  - The assembly of plan components to plans
  - Planning of actions independent from their actual execution
  - Ability to modify or abandon plans

- Identification of macro-plan components, and
- their assembly to complex macro plans
- Cumulative learning
- Forming abstract concepts, and
- Integration of several of these levels into one system.

If the robot is to achieve arbitrary, user-defined goals there is the requirement for the robot to identify the accomplishment of a goal. Although robots can learn certain sensor-motor competences without any internal representation of goals, for example by some type of Hebbian learning through mere correllation of perceptions and actions, we argue that it is impossible to implement a learning robot controller that could learn *any* behaviour without explicitly specified goal states.

5 Learning Mechanisms

Because the emphasis of this paper is on making robots achieve certain, user-defined goals I concentrate on supervised learning mechanisms here, i.e. learning mechanisms that use an external reinforcement signal to control the learning process. The clearest categorization of learning mechanisms in robotics I see is based on when this signal is received: immediately after an action is performed (immediate reward), or only after a sequence of actions has been performed (delayed reward).

If immediate reward is available, learning in robots can be achieved for example through operant conditioning (instrumental learning). In Grey Walter's turtles of the 1950s ([Walter 5?]) changing charges in a capacitor, according to learning rule using immediate feedback, led to light-seeking behaviour. Whilst Walter was dependent on electronic circuitry to implement his learning controller, and robots were therefore built for specific tasks, we now have controllers that allow a robot to acquire different tasks by merely redefining the reward function (see, for instance, [Nehmzow 92], for an overview).

Learning from immediate reward is the most effective way of learning. Competences such as phototaxis can be acquired within minutes by robots ([Nehmzow 94]). In some cases, however, immediate reward is not available. If the task has been described in such a way that reward is only obtained occasionally (i.e. upon reaching a goal state), the learning process becomes very ineffective.

Learning by trial and error from performance feedback, i.e. from feedback that evaluates behaviour, but does not indicate correct behaviour ([Sutton 91]) is the most accessible and most widely used method of robot learning. This learning from scalar feedback, after having executed a particular action is called *reinforcement learning*.

The best known reinforcement learning architecture that allows learning from delayed rewards is Q-learning ([Watkins 89]. Here, as in the "Bucket Brigade" algorithm, reward is trickled backwards to all actions involved in reaching the goal position. Pairs of current state and possible action at any given moment are given a 'Q-value' (a probablity of success). Therefore, what happens is that delayed reward is translated into immediate reward through the learning mechanism.

5.1 The phototaxis example

That learning from delayed rewards is slow can be seen in this example from robotics: suppose the goal were to build a light-seeking robot, and the performance of this robot was to be measured exclusively in terms of its light finding behaviour. If the task description awards positive reinforcement upon reaching the actual light source, the learning process is extremely slow in taking off, because initially all actions appear equally good to the robot, and the search space is enormous. [Kaelbling 90] has implemented such a system and reports that the learning process was so slow that she returned to simulations of her learning controller[4]. If, on the other hand, the task description awards reward whenever the robot performs a move towards the currently brightest part of the room, learning is extremely fast and phototaxis is achieved within minutes ([Nehmzow 94]). Purely from a light-seeking performance point of view, the latter system is superior. The question is: can we learn from this example how to make learning from delayed rewards more effective?

One obvious way to improve the "translation" of delayed reward into immediate reward is that of improving the teacher (see figure 3). If, for example, the teacher has the ability to learn itself, it might be possible to build suitable models that allow immediate assignment of appropriate reward to the robot's actions.

5.2 Advanced robot learning

The robot learning curriculum identified some higher level competences in learning robots. Here are some suggestions about how to move towards these more complex competences:

1. In addition to learning the robot control function $State \rightarrow Action$, have the robot learn a forward prediction model $State \times Action \rightarrow NextState$. Becuase this forward prediction model is *task-independent* knowledge, it can be used to improve the ability of the robot to learn strategies for new goals or tasks. For example, if this learned forward prediction model is perfectly correct and complete, it can be used to compute the control policy for a new goal simply by searching for a sequence of actions that leads to a goal state (i.e., traditional AI-style planning). More realistically, any learned forward prediction model is likely to be only partly correct. Even in this case it can be used to improve the accuracy of learning control policies for new goals. One method for accomplishing this is Explanation-based neural network (EBNN) learning, in which approximate, previously learned forward prediction models have been shown to significantly improve the accuracy of Q learning ([O'Sullivan *et al.* 95]).

2. Allow previously learned control behaviors to be used as primitive actions for subsequent learning. For example, imagine a robot with primitive actions Forward(d) and Turn(theta), that learns a control strategy for the goal "exit the room by going into the hallway" and a second control strategy for "traverse the hallway." Given a third goal such as "navigate from room A to room B" it would be much easier to solve the goal by using these previously learned behaviors as primitive actions. This kind of hierarchical organization offers one possible route to scaling up robot learning to tasks of more realistic complexity. Recent attempts to develop

[4] It should be noted, though, that the purpose of Kaelbling's system was not primarily to build a light-seeking robot, but to investigate learning from delayed rewards.

such approaches include [?] and [?], but this remains an issue in need of further research.

3. Temporal aspects as well as perceptual ones need to be represented in the controller, for instance by constructing rich internal states from the temporal sequence of perceptions (*dynamic robot learning*). There is a similarity here to early research in computer vision: it became eventually clear how difficult it is to identify an object using a single frame, whilst the task became much simpler through using sequences of images ([Dickmanns & Graefe 88], p.4f). "Active vision" is a further advancement of this idea, the vision system now being able to control the focus of attention as well. Similarly, it is a hard task to identify particular world states from one observation, but using temporal knowledge and active sensing the task can be made easier.

4. Anticipation and prediction, surprise/novelty detection both in temporal sequences and in single perceptions are needed to achieve the necessary sensor acuity. The more complex task, robot and environment are, the higher will be the demand on conputing resources. One way to tackle this problem is to focus attention even before sensor signals are obtained, i.e. select the signals that are to be processed further, rather than take all data and analyze afterwards. This principle has successful been used in vehicle road following ([Dickmanns & Graefe 88]).

5. In order to make plans of whole sequences of actions, the controller needs to be able to generate subgoals. Schmidthuber and Wahnsiedler ([Schmidthuber & Wahnsiedler 92]) address the problem of "compositional neural sequence learning" in a path-planning simulation, in which a path of minimal cost has to be found if previous paths and their costs are known.

6 Open Questions

We identify the following areas as problems that require further investigation.

6.1 Task Description

In the phototaxis example (section 5.1) it became apparent how different task descriptions (reward when reaching the light versus reward when turning towards the brightest spot) resulted in drastically different performance. Defining a task description is still much of a black art, and dependent on the designer's ingenuity. Mechanisms (algorithms) to establish optimal task descriptions are as yet lacking, but are needed if autonomous adaptation of the teacher, as proposed in section 5.1, is desired.

6.1.1 Learning from delayed rewards

Current machine learning paradigms that allow learning from delayed rewards let discounted reward trickle backwards to reinforce all actions that led to the final, successful action ([Watkins 89, Samuel 59, Holland *et al.* 86]).

There are a number of problems with this. Firstly, the state space in which the robot operates needs to be discretized. This is not trivial, as too fine a scale will introduce much redundant data, whilst too coarse a scale will miss necessary detail. Scenarios are conceivable in which the success or failure of a learning robot is dependent on the correct scale, so that in this case one could argue that the problem has shifted from one of finding the right learning paradigm to one of finding the right scale.

Secondly, Q-learning ([Watkins 89]) requires the agent to encounter all possible states infinetely often to be guaranteed to find the optimal learning strategy — a prerequisite that is impractical.

Thirdly, not all actions that form part of a sequence do necessarily contribute towards the eventual success of the agent. Some actions may be completely unrelated to success, and therefore ought not to receive any positive (nor negative) reinforcement.

The question is: can we find better ways to learn from delayed rewards (such as developing concepts about the world, task, and agent, and using these to attribute reward and punishment)?

This is the (difficult) credit assignment problem ([Connell & Mahadevan 93]). Temporal credit assignment is the hard part here: which actions of a sequence of actions were instrumental in reaching a goal state, and should therefore be reinforced? The standard way of solving this is the the bucket brigade algorithm or variants of it: the state immediately preceding a goal state is reinforcement. In a subsequent epoch, the state immediately preceding this state is reinforced, and so on ([Samuel 59, Holland *et al.* 86]). This method has the obvious disadvantage of being slow and requiring frequent repetition of the same series of actions, furthermore there is no guarantee that a particular sequence learned is actually a good one (let alone optimal).

How to avoid these problems, how to attribute reinforcement based on delayed rewards in a more efficient manner is an open question.

6.1.2 Modification of the reward function through learning

Consider a controller like the one given in figure 3. The relationship between learner and teacher is an intricate one and determines the performance of the whole controller. Although it is conceivable that the learner acquires competences that are not explicitly taught by the teacher (see, for instance, the dead-end example of section ??), the teacher plays an important part regarding the possible complexity of acquired control function. Therefore, it might be beneficial to have an adaptable teacher as well as an adaptable learner. The questions are: how can it be decided in a particular situation whether the learner or the teacher should be modified, and how can we determine in what way to modify the teacher, should this be the chosen action?

In the phototaxis example (see section 5.1) it became clear that the method of providing reinforcement made the greatest difference to the robot's performance. Providing immediate reward rather than delayed reward, robots learn phototaxis very quickly indeed. The research issue here is: how can the teacher be trained to convert delayed reward into immediate reward?

6.2 Concept learning

Correlation between actions

In learning from delayed rewards, we may require the development of internal concepts that allow the machine to attribute reward and punishment, following delayed reward (see above). The question is: what is such a concept, and how can it be developed? Such a concept would have to determine whether temporarily disjunct actions are related or not.

Feature recognition

One aspect of concept formation is the differentiation between salient features and dynamical features of the environment, both in terms of temporal patterns and perceptual patterns. Anticipation and prediction (i.e. some sort of generalization) can be useful mechanisms here (see [O'Sullivan *et al.* 95] for an example of an explanation based learning system on a mobile robot).

Macro learning and planning

In order to move towards complex functions, fundamental building blocks of such functions need to be identified, and subsequently assembled. The questions are: how can the learning controller identify fundamental building blocks, and how can it assembly these to form (various) complex plans? (See [Singh 92] for an example of how concatenating elemental sequential decision tasks leads to more complex sequential decision tasks. See also [Lin]).

6.3 Dynamic robot learning

The robot learning problem must not be viewed statically. The robot should not 1) obtain a sensory perception, 2) process the data and 3) make a decision to perform an action (sense-think-act cycle). The interaction between robot and environment is a dynamic one, in which *sequences* of perceptions and *sequences* of events should determine future actions. An example from the area of navigation illustrates this point: room 3601B at Carnegie Mellon University is an open plan office with lots of identical cubicles. The identification of one's location based on the current sensory perception alone is impossible. Considering a sequence of previous actions (moves), however, it is no problem any more to establish one's current position.

Like in computer vision, a temporal sequence of perceptions provides more information about the environment than a single observation ([Dickmanns & Graefe 88]). It is one of the yet unachieved goals of robot learning to develop a learning architecture that allows the robot to learn from complete temporal sequences of perceptions, rather than from single perceptions.

6.4 Validation

If learning robot controllers are to be used outside the experimental laboratory scenario, valiation of the robot's behaviour is needed. Such validation needs to answer these questions: Under what circumstances will the robot leave safe operational limits? Under what circumstances will the robot act in a goal-oriented way, and can it be proven that the robot will achieve the goal under these circumstances?

7 Summary

To make a mobile robot (such as the one shown in figure 1) achieve certain goals (specified by the user) in a particular environment, it may be sufficient to use fixed control strategies, as long as the tasks are relatively simple.

Due to the problem of perceptual discrepancy designing such controllers becomes increasingly difficult as the complexity of tasks increases (complexity here is seen with respect to the robot learning curriculum outlined on page 7). This is why we believe that for complex tasks *learning* robot controllers have to be used.

The thesis of this paper can be summarized by the following three statements: In order to achieve tasks in mobile robot control that go beyond fundamental reflexive responses to sensory stimuli, we need to

1. use learning controllers, because the problem of perceptual discrepancy gains in influence as the complexity of tasks increases,
2. consider both perceptual *and temporal* aspects of the robot-environment interaction, and
3. utilize high level competences like prediction, feature detection (correlation in perceptual and temporal streams of data), and concept formation.

Using a classical reinforcement learning architecture, we believe that this gain in complexity cannot be achieved by altering the learner alone (for instance by developing new learning paradigms), but has to concentrate on adaptable teachers (i.e. an adaptable reinforcement function) as well.

References

[Colombetti & Dorigo 93] M. Colombetti and M. 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.

[Connell & Mahadevan 93] J. Connell and S. Mahadevan (eds.), *Robot Learning*, Kluwer 1993.

[Dickmanns & Graefe 88] E. Dickmanns and V. Graefe, *Dynamic monocular machine vision* and *Applications of dynamic monocular machine vision*, Universität der Bundeswehr, Munich, Technical Report UniBwM/LRT/WE 13/FB/88-3, 1988.

[Fikes 72] Fikes, [Shakey and the triangle tables]

[Holland *et al.* 86] J. Holland, K. Holyoak, R. Nisbett and P. Thagard, *Induction*, MIT Press 1986.

[Kaelbling 90] L. Kaelbling, *Learning in Embedded Systems*, PhD thesis, Stanford Technical Report TR-90-04, 1990.

[Kohonen 88] T. Kohonen, ??, Springer Verlag 1988.

[Lin] Some paper on assembling simple functions into complex ones.

[Littman 93] M. Littman, *An optimization-based categorization of reinforcement learning environments*, in: J.A. Meyer, H. Roitblat and S. Wilson (eds), *From Animals to Animats 2*, MIT Press 1993.

[Mahadevan & Connell 91] S. Mahadevan and J. Connell, *Automatic Programming of Behavior-based Robots using Reinforcement Learning*, Proc. AAAI 91.

[Mitchell 80] T. Mitchell, *The need for biases in learning generalizations*, Rutgers University TR CBM-TR-117, 1980.

[Mitchell 82] T. Mitchell, *Generalization as Search*, Artificial Intelligence Vol 18, No 2, March 1982.

[Nehmzow *et al.* 89] U. Nehmzow, J. Hallam and T. 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] U. Nehmzow, *Experiments in Competence Acquisition for Autonomous Mobile Robots*, PhD thesis, University of Edinburgh 1992.

[Nehmzow 94] U. Nehmzow, *Autonomous acquisition of sensor-motor couplings in robots*, University of Manchester technical report UMCS-94-11-1, 1994.

[Martin & Nehmzow 95] P. Martin and U. Nehmzow, *"Programming" by teaching: neural network control in the Manchester mobile robot*, Conf. Intelligent Autonomous Vehicles IAV 95.

[O'Sullivan *et al.* 95] J. O'Sullivan, T. Mitchell and S. Thrun, *Explanation based learning for mobile robot perception*, in *Symbolic Visual Learning*, Oxford University Press, 1995.

[Oreskes *et al.* 94] N. Oreskes, K. Shrader-Frechette and K. Belitz, *Verification, Validation, and Confirmation of Numerical models in the Earth Sciences*, Science, Vol 263, 4 Feb 1994, pp. 641-646.

[Rendell 86] L. Rendell, *A General Framework for Induction and a Study of Selective Induction*, Machine Learning, 1, 1986.

[Rumelhart] D. Rumelhart, **PDP book**

[Russell 89] S. Russell, *A Sketch of Autonomous Learning using Declarative Bias*, in P. Brazdil and K. Konolige (eds), *Machine Learning, Meta Reasoning and Logics*, Kluwer 1989.

[Russell 89] S. Russell, *The Use of Knowledge in Analogy and Induction*, London: Pitman 1989.

[Schmidthuber & Wahnsiedler 92] J. Schmidthuber and R. Wahnsiedler, *Planning simple trajectories using neural subgoal generators*, in J.A. Meyer, H. Roitblat and S. Wilson (eds.), *From Animals to Animats 2*, MIT Press 1993.

[Samuel 59] A.L. Samule, *Some studies in machine learning using the game of checkers*, IBM Journal on Research and Development, 1959.

[Singh 92] S. Singh, *Transfer of learning by composing solutions of elemental sequential tasks*, Machine Learning 8:323-339, 1992.

[Simon & Lea 74] H. Simon and G. Lea, *Problem Solving and Rule Induction: A Unified View*, in L.W. Gregg (ed.), *Knowledge and Cognition*, Erlbaum, 1974.

[Simon 81] H. Simon, *The Sciences of the Artificial*, 2nd edition, MIT Press 1981.

[Sutton 91] R. Sutton, Reinforcement Learning Architectures for Animats, in J.A. Meyer and S. Wilson (eds), *From Animals to Animats*, MIT Press 1991.

[Walter 5?] Grey Walter, Scientific American 195? and 195?.

[Watkins 89] C. Watkins, *Learning with delayed rewards*, Doctoral dissertation, Cambridge University 1989.

[Dorigo] **Special issue of Man, Systems and Cybernetics**

8 Reinforcement Learning

In figure 3 depicts the reinforcement learning problem in robotics, as it has been discussed in this paper so far: true world state and true robot state are mapped on to perceived states by the unknown functions ga and gw. The "learner" inside the function f receives the perceived state as an input signal, as well as the reinforcement signal (a scalar) from the "teacher". Using this information, it generates actuator commands ("output").

The reinforcement signal generated by the teacher is a scalar reward, given in accordance with the robot's success measured with respect to the specified goals. This reward may be immediate, or delayed.

Various approaches to robot learning have been proposed. Grey Walter built small mobile robots that learned to approach light sources through conditioning ([Walter 5?]). "Shakey" learned to assemble building blocks of plans to form sequences of actions ([Fikes 72]).

Reinforcement learning, the learning of sensor-motor competences from single-valued scalar rewards, is currently the most popular method to make robots learn.

Sutton's definition of *Reinforcement learning* ([Sutton 91]) reads:

> Reinforcement learning is learning by trial and error from performance feedback, i.e. from feedback that evaluates behaviour, but does not indicate correct behaviour.

The most widely used implementation of reinforcement learning is Q-learning.

[Q-learning as one popular implementation. Definition. Problems (slow to take off, discretization, infinte amount of time needed).]

[Improvements. Take Leslie's example of phototaxis, in contrast of the experiments with the Edinburgh R2. A redescription of the reward function brings immediate improvement with respect to our measurement, i.e. the achievement of a goal (here, phototaxis).]

[Proposal: use an adaptable teacher (see open questions)[5].]

Control learning

We define "control learning" as "change in a control function (function f in figure 3) in response to external stimuli, with the view of achieving specified goals". We evaluate the effectiveness or ineffectiveness of a control learning algorithm in terms of improvement of robot behaviour with respect to goals and environment.

Unsupervised learning

Rewrite and rethink

Whilst the ultimate measure of success of a control learning algorithm is the improvement of robot performance with respect to a specified goal in a particular environment, this objective can only be achieved if other types of learning form part of the controller. For example, the machine will have to be able to cluster and categorize sensory information. This is unsupervised learning (i.e. learning without externally specified goal), not control learning, and forms an important component of a control learning architecture.

'Useless' data

Another example is the mere accumulation of 'useless' data, which can become important to achieve complex tasks. Such data is the basis of concept learning, but again this is not control learning.

'Learning' differs from 'control' in that the latter merely aims to minimize an error signal, whilst the former seeks to detect regularities and build some sort of internal representation of robot, task and environment.

Supervised learning and reinforcement learning

Learning with the purpose of achieving goals, as is the case in the robot control learning problem (see section ??) requires feedback signals. These signals can be mere scalar indications of the degree of success (reinforcement learning, a special case of supervised learning), or complex indications of the correct response in a given situation (the more general case of supervised learning).

In contrast to reinforcement learning, *supervised learning* is learning by any type of feedback, including the desired action, given a particular input stimulus. In supervised learning, input-output mappings must be provided.

[5] In the example of the reinforcement learning implementation given in [Nehmzow 94], such an extension would be the autonomous addition or removal of instinct-rules.

Figure 1: The robot learning scenario

Figure 2: Fundamental principles of robot-world interaction

Figure 3: Diagram of the robot learning problem, ignoring temporal aspects