An investigation of a learning system with a mobile robot

Alan H. Bond [1] and David H. Mott,
Artificial Intelligence Laboratory,
Queen Mary College, University of London
Mile End Road, London E1 4NS, England

[1]Current address: Computer Science Department, 4173C Engineering 1, University of California, Los Angeles, California 90024 email: bond@cs.ucla.edu

Abstract

A learning system is described. It was applied to a concrete learning situation of a mobile robot in a simple environment.

Knowledge was represented as rules called *schemas*, which predicted future sensory and motor events. Each rule had certainty values for its constituents and a confidence for the entire rule.

The system consisted of Prediction, Behavioral and Learning subsystems. Prediction used schemas in a bottom-up manner to generate a list of predicted events. Behavior had three separate mechanisms for generating actions.

- *Goal seeking* behavior used schemas in a top-down manner to seek or to avoid primitive <HIGH> or <LOW> events.
- *Exploratory* behavior generated actions that were predicted, although not necessarily sought.
- *Random* behavior occurred in the absence of any prediction or goal.

The system was primed with a small set of innate schemas.

The Learning subsystem created new schemas from events that occurred, and modified existing schemas according to their success in prediction.

A learning session is described, in which the system autonomously learnt enough schemas for the mobile robot to survive in its environment.

1 Introduction

There are a number of reasons to regard sensory-motor learning as a basis for intelligence, and therefore to be of greater interest than other forms of learning. A few attempts have been made to study learning in sensory-motor environments, for example Jones [1], Friedman [2], Doran [3], Uhr and Kochen [4], however these have in the main been confined to simulated environments. Some have used real robots, for example Fikes, Hart and Nilsson [5], Andreae [6] and Raibert [7]. For an early but insightful review, see Ernst's paper [8].

The work described here used a simple but real mobile robot. The robot was the Queen Mary College Mark IV Experimental Robot and is shown in Figure 1. C. Mark Witkowski developed the robot hardware, both mechanics and electronics. The Mark IV robot is described by Witkowski [9] [10] and Bond [11].

The learning system took Becker's Model of Intermediate Cognition [12] [13] as its starting point, and developed and modified it considerably.

The work was funded by the Science Research Council, Computer Science Committee. The project was entitled "Learning in a Mobile Robot". Alan H. Bond was principal investigator and supervisor. The software was developed by David H. Mott. Further discussion may be found in the Final project report to the Science Research Council [14] and in Mott's Ph.D. thesis [15].

2 The robot and its environment

The robot was a simple vehicle, shown in the photograph and diagram, Figures 1 and 2. Although the robot had many sensors including sonar and also touch bars on all four sides, for the purposes of this study only a few sensors were used, and their ranges were cut down to a small number of values corresponding to intervals of their usual detected values.

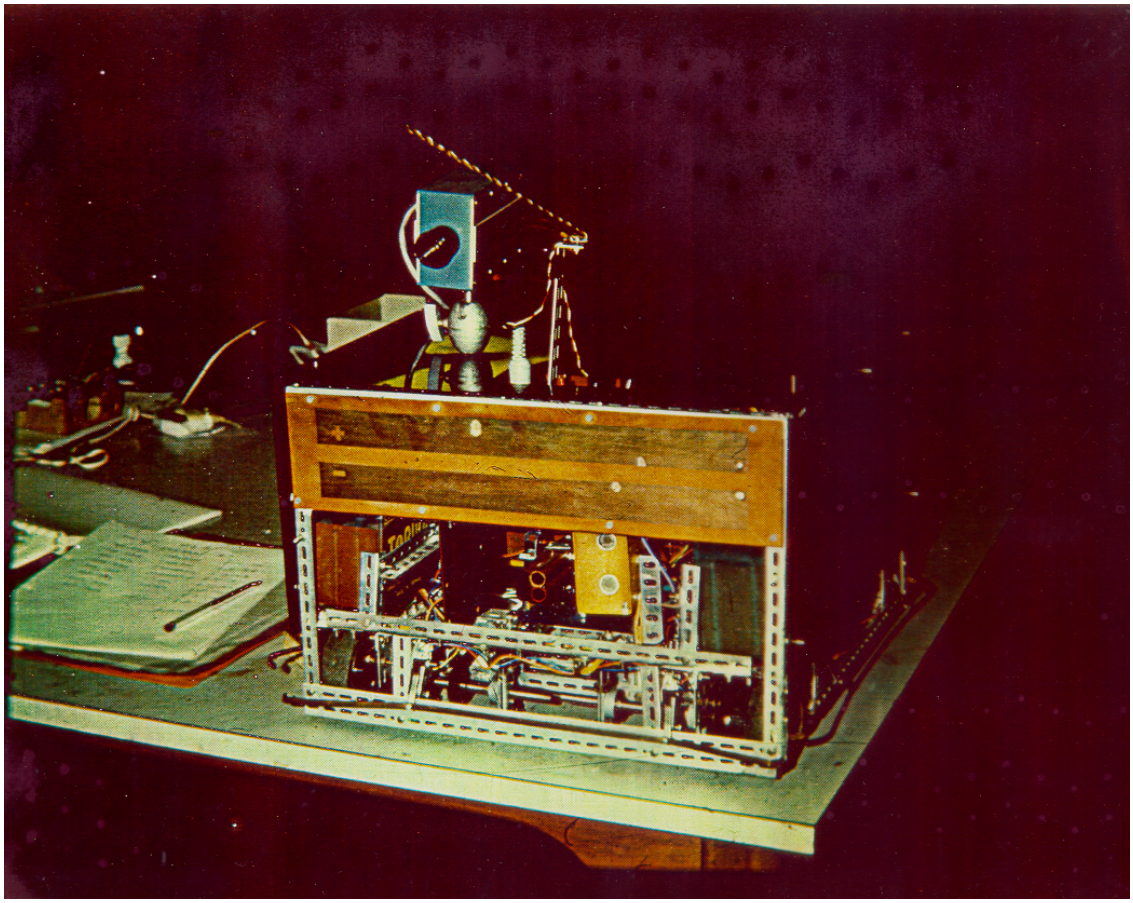


Figure 1: Photograph of the robot

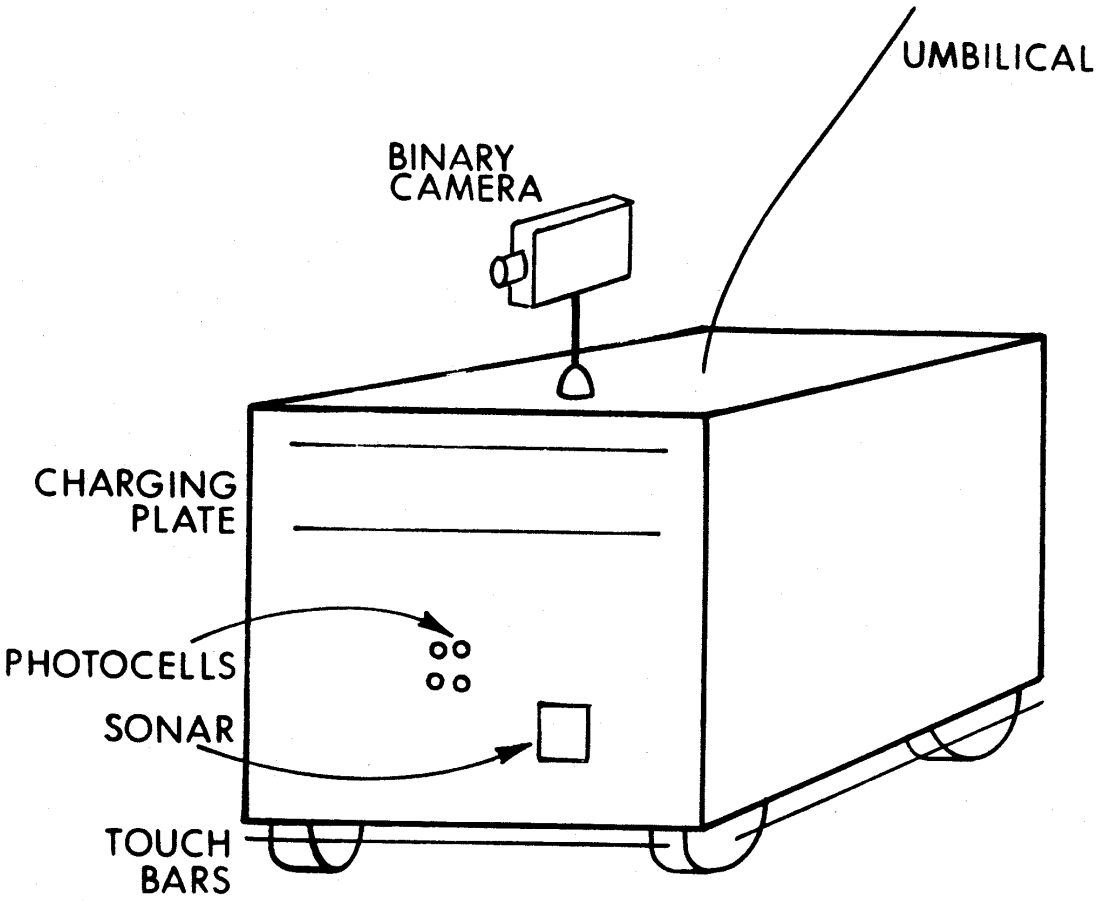


Figure 2: Diagram of the robot

Figure 3 shows a photograph of the robot in its environment and Figures 4 and 5 are diagrams of the external environment and internal environment.

Figure 3: Photograph of the robot in its environment

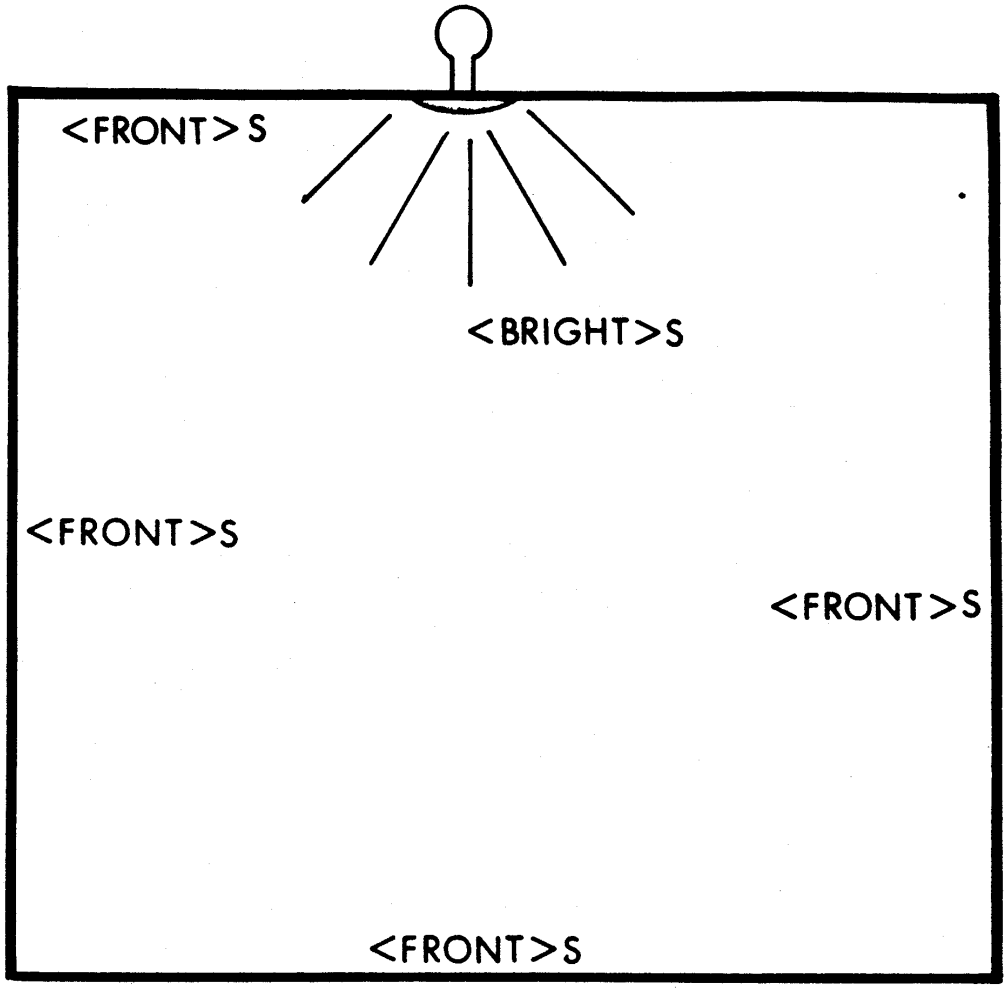


Figure 4: Diagram of the environment - The spatial and external sensor environment

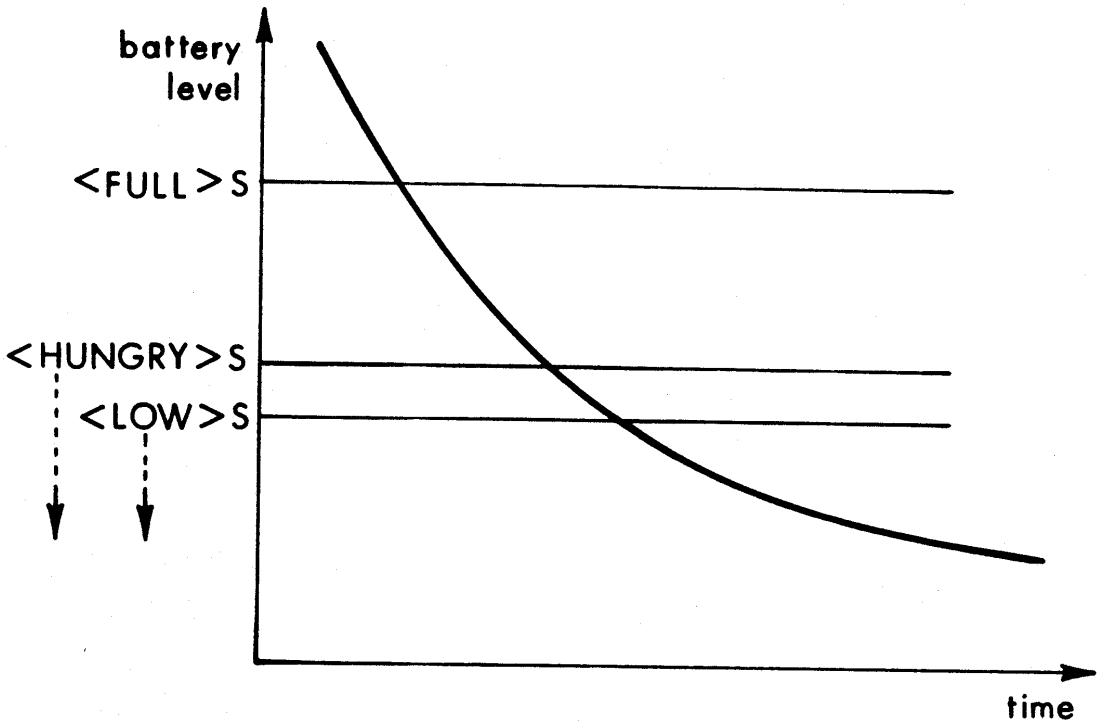


Figure 5: Diagram of the environment - The energy and internal sensor environment

Figure 6 gives what we call the movement environment, or the primitive movements and their effects on the spatial state of the robot.

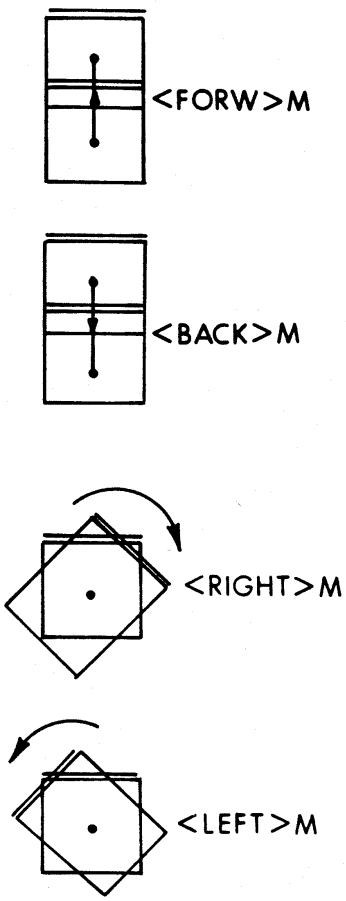


Figure 6: Diagram of the environment - The movement environment

The computer system used is diagrammed in Figure 7.

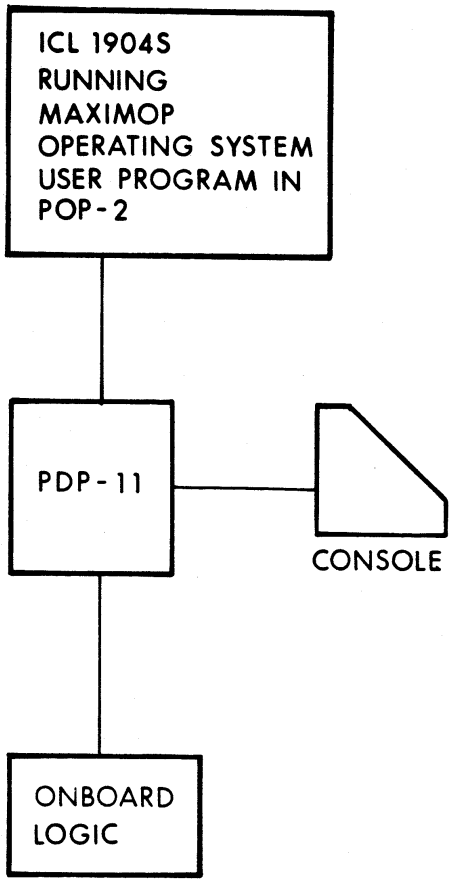


Figure 7: Diagram of the computer system

Sensor or Motor	Kernels used	Significance
Photocells	<Bright>S	Value > threshold
Touch Bars	<Front>S	Front touch bar activated
Battery Level Sensor	<Full>S	Battery fully charged
	<Hunger>S	Battery getting low or dangerously low
	<Low>S	Battery dangerously low
Oncharge	<Charge>S	Battery being charged
Internal software	<High>S	Predicted pleasure
Motors	<Forw>M	Move robot a few inches forward
	<Back>M	Move robot a few inches backward
	<Left>M	Rotate robot about 10 degrees right
	<Right>M	Rotate robot about 10 degrees left
Bleeper	<Cry>M	Bleep Morse code for SOS

Figure 8: I/O vocabulary, kernels, used

3 Elements of the system

Kernels The interface of the system to the robot was by means of a set of symbolic atoms, which following Becker we called "kernels". We use his notation using <> brackets and with a following S or M to denote a sensory kernel or motor kernel. The set of kernels used in this study is shown in Figure 8.

The system received a stream of sensory kernels from the robot, i.e., from a software interface to the robot hardware.

<BRIGHT>S occurred when the photocell gave a value greater than a threshold.

<FRONT>S occurred when the front touch bar was activated.

<FULL>S occurred when the battery was fully charged.

<HUNGER>S occurred when the battery was getting low.

When the battery was getting " dangerously " low, the kernel <LOW>S was produced as well as <HUNGER>S.

If the battery level was normal, i.e., between full and low, then no kernel occurred.

When the battery was being charged, by being connected to a charger, the kernel <CHARGE>S occurred.

The system generated a stream of motor kernels, which were commands to the robot.

<FORW>M moved the robot a few inches forward

<BACK>M moved the robot a few inches backward

<LEFT>M rotated the robot about left 10 degrees about its center

<RIGHT>M rotated the robot about right 10 degrees about its center

<CRY>M bleeped the Morse code for SOS

The original notion of kernel in Becker's work was similar to a literal i.e. it had a function

and value

<RANGE 12>

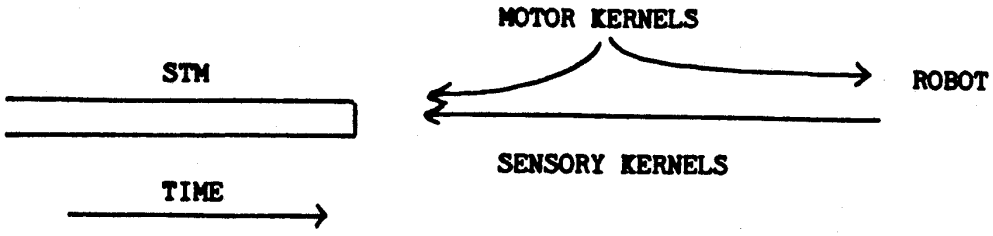
<COLOR RED>

etc.

involving a variable value. Our implementation reduced this to atomic symbols on the grounds that the Becker form gave too much initial structure as a starting point for learning, and also because it was an extra complication in an already complicated system.

3.1 Short term memory - STM

The stream of kernels both input and output flowed into a single infinitely long queue, an item in the queue was the set of kernels occurring in one discrete time interval, together with the (integer) value of the time, see Figure 9.



EVENT is a set of kernels occurring in a given time slot
e.g. { <CHARGE>S <FRONT>S }

EVENT SEQUENCE is a temporal sequence of events in successive time slots

E1 -> E2 -> -> E3

SCHEMA is a knowledge unit which, given the occurrence of a particular event sequence, predicts the future occurrence of an event sequence.

SCHEMA EVENT SEQUENCE => SCHEMA EVENT SEQUENCE

SCHEMA EVENT SEQUENCE is a temporal sequence of event estimates of the form

(<KERNEL> <CERTAINTY FACTOR>) (<KERNEL> <CERTAINTY FACTOR>)

SCHEMA has a confidence measure, more exactly

(SCHEMA EVENT SEQUENCE => SCHEMA EVENT SEQUENCE) <CONFIDENCE>

Figure 9: Short term memory, events and schemas

The system operated on a discrete time scale in cycles lasting one unit of time, and strictly alternated between motor output and sensory input. Thus in any one time unit or slot, either only sensory kernels or only motor kernels were present.

Becker suggested a finite short term memory and he had relevance levels attached to kernels but no times attached. He did not have our alternation of cycles.

We called a set of kernels occurring in one time slot an *event*.

e.g. E1 <CHARGE>S <FRONT>S

An event could of course be empty, the empty event will be denoted by the null string.

The contents of STM is an event sequence and temporal adjacency will be denoted by " $->$ ".

e.g. E1 $->$ E2 $->$ $->$ E3

The current time will be denoted " ct ".

3.2 Schemas and the Model

The central data structure was the "model" which was an unordered set of schema. Schema were rules which made predictions about what was likely to happen in STM and therefore in the environment. A schema had the form

event sequence => event sequence

and its intended meaning was that if the left hand side occured then the right hand side was predicted to happen in the future. More specifically, if the left hand side matched the

contents of STM, this event sequence was said to occur. Then the right hand side was predicted to occur starting in the next time slot.

The schemas were uncertain, both in their construction and in their validity. Each kernel in the schema had a certainty factor, which was a measure of how certain the system was that the kernel should be present in the schema. This certainty factor of a kernel on the left hand side of a schema we called its *relevance*, and the certainty factor of a kernel on the right hand side we called its *certainty of occurrence*. Each schema had an overall certainty factor, which gave a measure of how certain the system was that the schema gave a correct prediction of what happened in STM. We called this its *confidence*.

Thus a schema event sequence, either a left hand side or a right hand side, actually had the form

(kernel, certainty factor) (kernel, certainty factor) . .

and a schema was of the form

(schema event sequence => schema event sequence) confidence

Certainty factors were real numbers in the range [0.0, 1.0]. They were not probabilities, since they did not add up to 1.0 and did not combine like probabilities. We used an ad hoc theory of certainty.

3.3 Negated kernels

Negated kernels were also used, denoted - <kernel>. A negated kernel represented the absence of that kernel in the current time slot, and was placed there by the system if it had

been predicted to occur and didn't.

Negated kernels were used in all the mechanisms of the system with appropriate treatment. They could for example be parts of schemas.

They also gave a natural representation for the goal of avoiding a given kernel or set of kernels.

3.4 Prediction

We regarded prediction and behavior as distinct processes. Prediction was taken to be the bottom-up generation of strings of kernels using the model and producing a lookahead tree. " Behavior " was the top down generation of subgoals using the model, and the stream of motor produced.

The system behaved by trying to avoid <LOW> and trying to achieve <HIGH>. <HIGH> and <LOW> were special experiential kernels corresponding presumably to primitive pleasure and pain. This was our *theory of motivation.*

Prediction affected behavior in several ways. It focussed attention on likely events in the future to seek or to avoid. It enabled the evaluation of a proposed plan of action. It also was used in our system for generating explicit negative kernels representing the absence of an expected or predicted event.

Apart from its role in behavior, prediction was used for learning, that is updating the model in the light of experience. The predictions of the model were compared with the actual events and the learning system updated the model accordingly.

The prediction system and prediction process are depicted in Figures 10 and 11. In a conventional fashion, left hand sides were matched to the contents of time slots in STM, starting from the current time slot, and anchored to it. This produced a *prediction tree* of all lines of prediction from the application of all applicable rules.

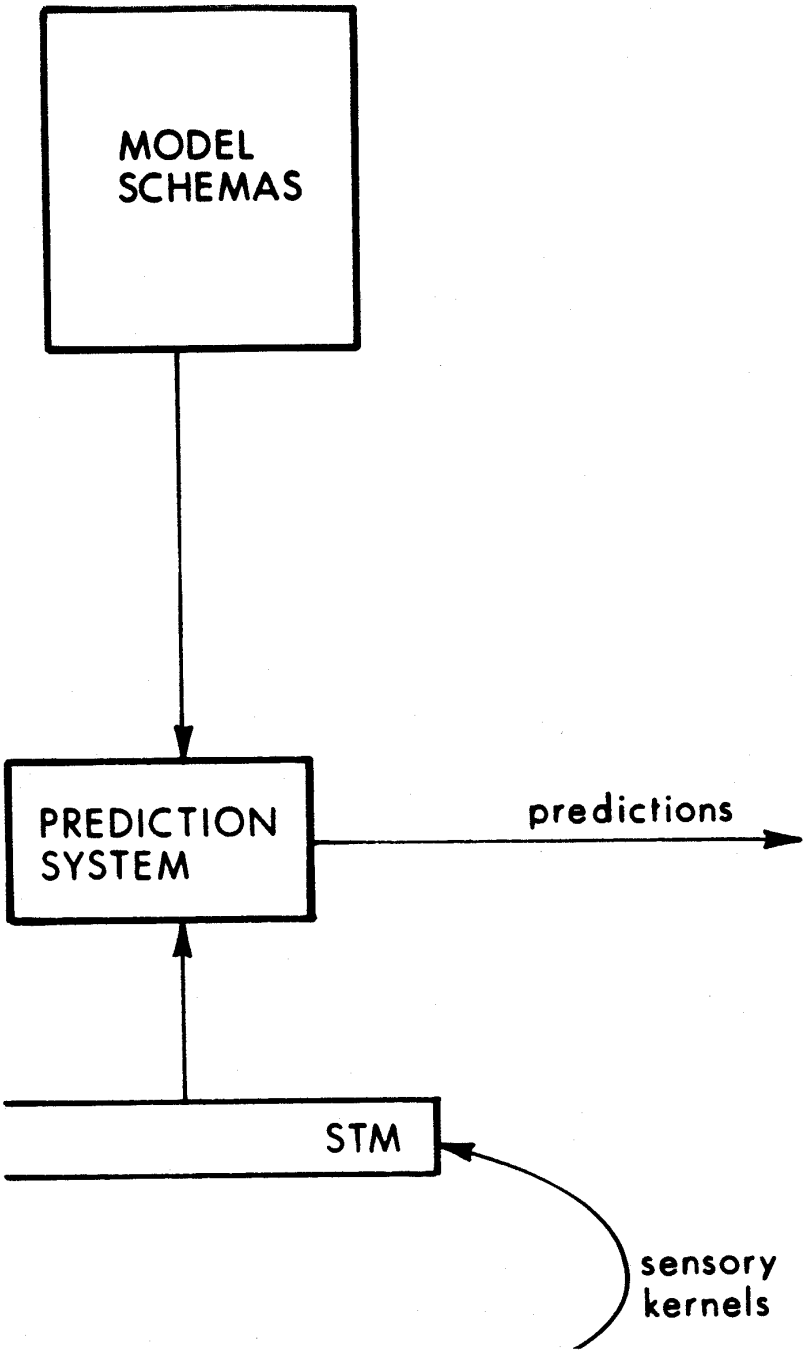


Figure 10: The prediction system

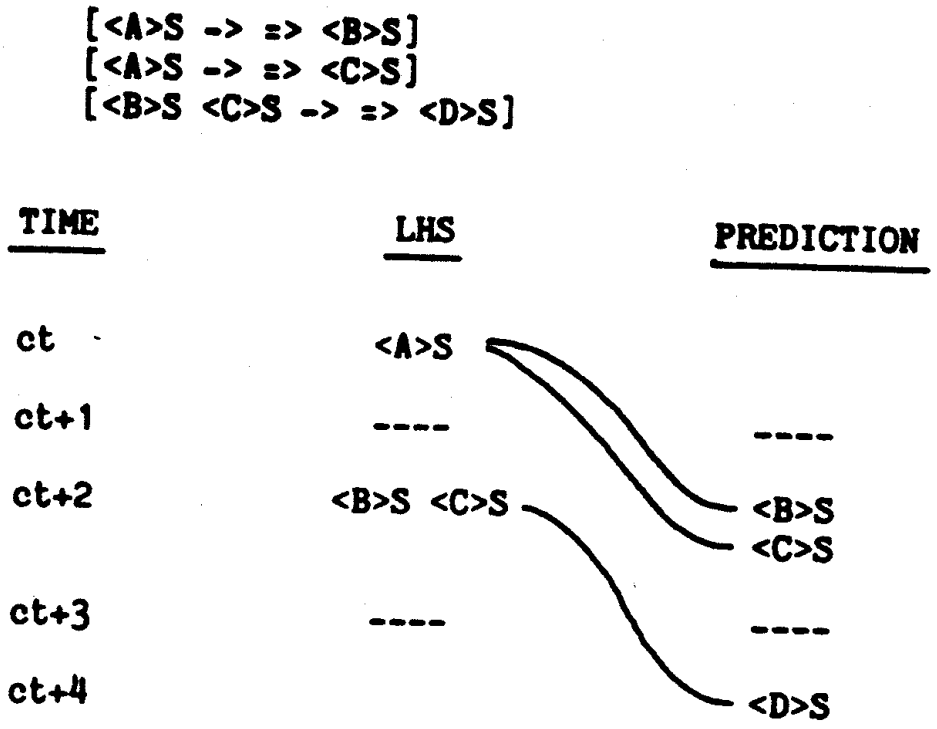


Figure 11: Prediction - The prediction process

For example, as shown in Figure 12, if we have the schema set shown and the kernel <A> occurs, then the prediction tree would be as shown. That is S1 predicts <B> and S2 predicts <C>; from <B> and S3, we predicted <D> if <C> occurred and from <C> and S3, <D> is produced if <B> occurs.

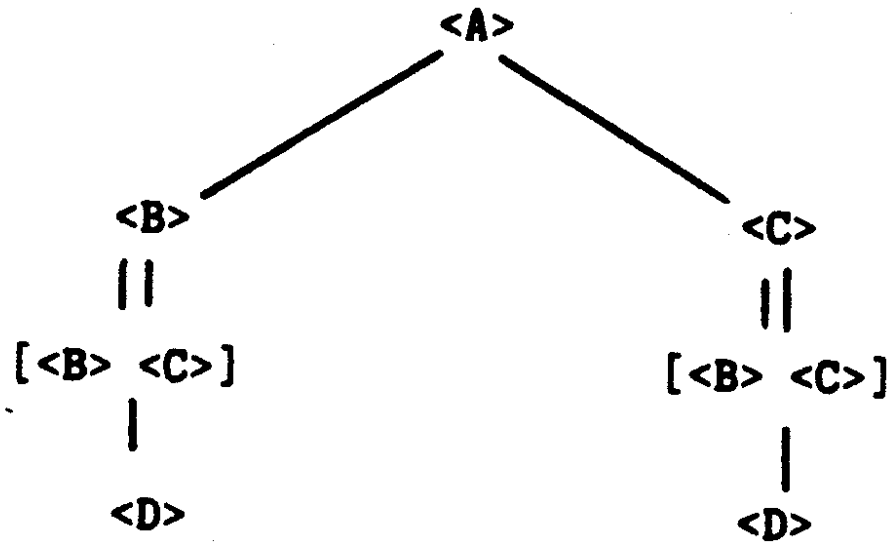


Figure 12: Prediction - A prediction tree

The predictions occurred with calculated certainty factors, and further second order predictions have multiplied certainty factors. Prediction lookahead was produced until a threshold of certainty was reached. A predicted kernel had a certainty and a time of occurrence.

Instead of producing a prediction tree, the prediction system produced a *prediction queue.* This was because of

1. The composite and overlapping nature of predicted events

2. The prediction used being only of the existence of a kernel independent of context.

The prediction queue shown in Figure 13 corresponds to the prediction tree shown in Figure 12.

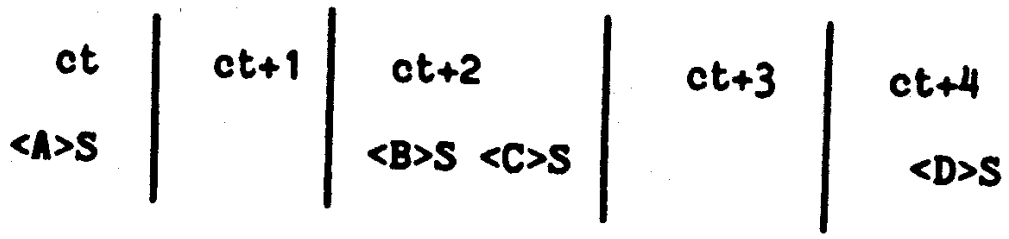


Figure 13: Prediction - A prediction queue

A prediction queue was simply a queue of time slots starting with the current time and going into the future. In each slot were the kernels predicted to occur in that time slot. Also stored in each slot, for the purposes of the learning system, were the left hand sides of the schemas making predictions in that time slot. The predictions made by a schema would be in the next and subsequent time slots. Linking information was held indicating which of the predicted kernels were predicted by which predicting left hand side.

The calculation of certainty factors was as follows. If we had a prediction of kernel <A> with certainty p and a subsequent prediction by a schema of the form

[<A>pa . . => <B>pb] c

then <B> was predicted with certainty

p*c*pb

The relevance pa of <A> in the schema entered the calculation only indirectly. A schema was only used to make a prediction if the relevances of the unmatched kernels were less than a threshold and no more than one kernel was missing in the match of LHS to STM.

In the first time slot, ct, the certainty p of predicted kernels which actually were occurring in STM was 1 and those not occurring was 0.

The prediction queue also held information about the current match of STM to the currently predicting left hand side. Those kernels in the currently predicting left hand side at time ct, which matched a kernel in STM in the corresponding time slot were tagged as matched. The notation we use for the tag is "*". One STM kernel could match more than one predicting left hand side kernel in one cycle. The predicted kernels in the slot were also matched against STM and tagged if they occurred. In this case an STM kernel could only match one predicted kernel. If there was more than one possible matching kernel, the one with highest predicted certainty of occurrence was tagged.

The prediction queue thus stored all the information that the learning system needed about the predictions. The learning system operated upon the tagged schemas, i.e. those containing tagged kernels, and also, as a special case, completely untagged schemas.

The operation of the prediction system in each time unit was as follows:

(i) Shift the entire prediction queue up one place.

(ii) Obtain next STM.

(iii) Place negated kernels into STM in ct slot.

(iv) Tag predicting and predicted kernels.

(v) Match STM against model and put any new predictions into the slots.

(vi) Pass any completed tagged or completely untagged schemas to the learning system.

(vii) Repeat from (i).

4 Behavior

There were four mechanisms:

1. *Random movement*, in the absence of any stimulus or prediction.
2. *Innate reflexes*, which formed the initial state of the model, and could provide such functions as backing off if too close or accomodation of dynamic ranges etc.
3. *Exploratory behavior*, of following predicted actions, if no goals currently existed.
4. *Goal seeking behavior*, in seeking <HIGH> or avoiding <LOW>, if these were predicted.

4.1 Random Movement

Purely random movement turned out to be not very useful, as the robot had little chance of moving very far away from its initial position. Indeed the probability of moving away a distance x is proportional to $1/x^2$.

Simply moving forwards was also tried, but to no avail.

4.2 Innate reflexes

The backoff reflex impeded progress in that the main learning experience of " feeding ", i.e battery charging, required the robot not to back off. We actually used one reflex only with the opposite effect, namely to push forward against any stimulus received from the front touch bar.

Innate reflexes were subject to learning and could be modified as a result of experience. In the context of this system, they were not really innate reflexes so much as *innate predictors*.

4.3 Exploratory Behavior

If there were no current attractions or threats, the system would explore as follows. If any motor kernels were predicted, these would be executed. Thus, if we had

[<A>S − > <M>M => <C>S]

and if <A>S occurred in STM, then <M>M would be predicted for the next time slot and executed.

This mechanism tended to test out schemas by repeating or " exercising " them.

If there were no predicted motor kernels then random movements were made.

4.4 Goal Seeking Behavior

A *goal* was a set of kernels that the system sought to happen. They had to all occur, and as goals currently used by the system had an explicit time of occurrence, all kernels in the set

had to occur in the specified time slot.

Goals were initiated by the prediction system finding a predicted <HIGH>S or <LOW>S at some future time. A goal that the system wished to achieve lead the system to chain backwards from the goal using the schemas of the model. These subgoals had specified time slots, and the chain was pursued until it reached the current time slot.

A predicted <LOW>S was to be avoided. Its backward chain produced the negated kernels of the predicting left hand side.

Thus we had a goal structure derived from the prediction queue and consisting of not a *goal tree*, Figure 14, but a *goal queue*, Figure 15.

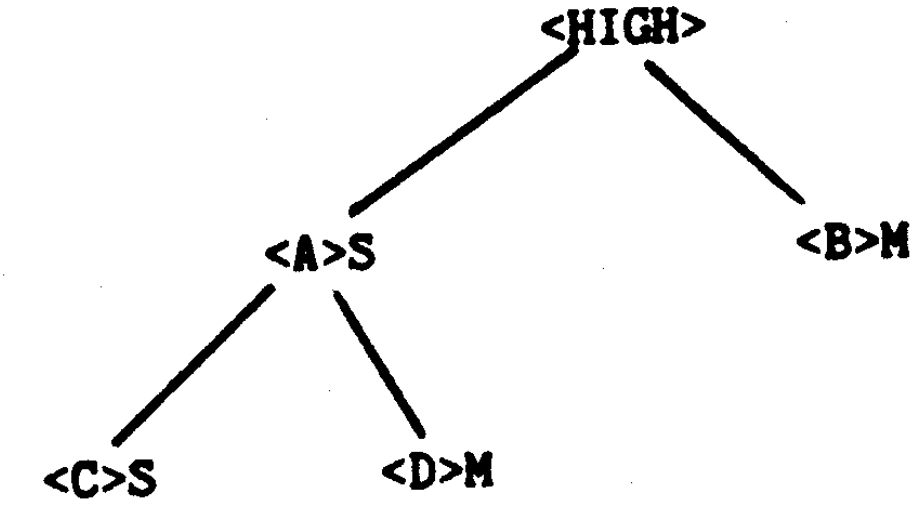


Figure 14: Behavior - A goal tree

This was a queue of time slots, each slot containing the kernels the system wished to achieve at that time. Each goal kernel had a *priority*, determined from the certainty of occurrence

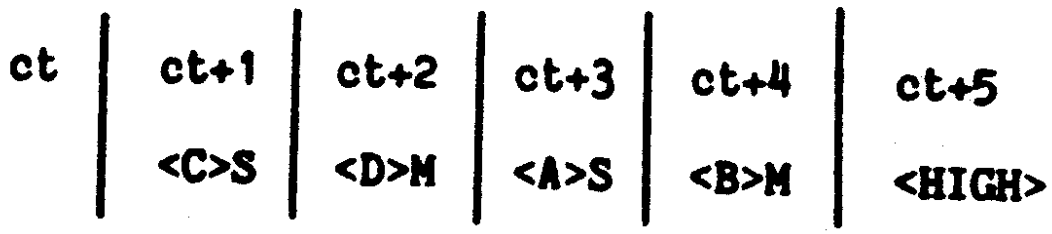


Figure 15: Behavior - A goal queue

of its originating top goal, and its relevance in predicting that top goal.

The goal structure might have needed updating in each time unit, and was thus continuously revised and updated.

The system chose, from amongst the goal motor kernels in the current time slot, that motor kernel with the highest priority and executed it, producing goal seeking behavior.

Figure 16 shows the prediction and behavioral systems together.

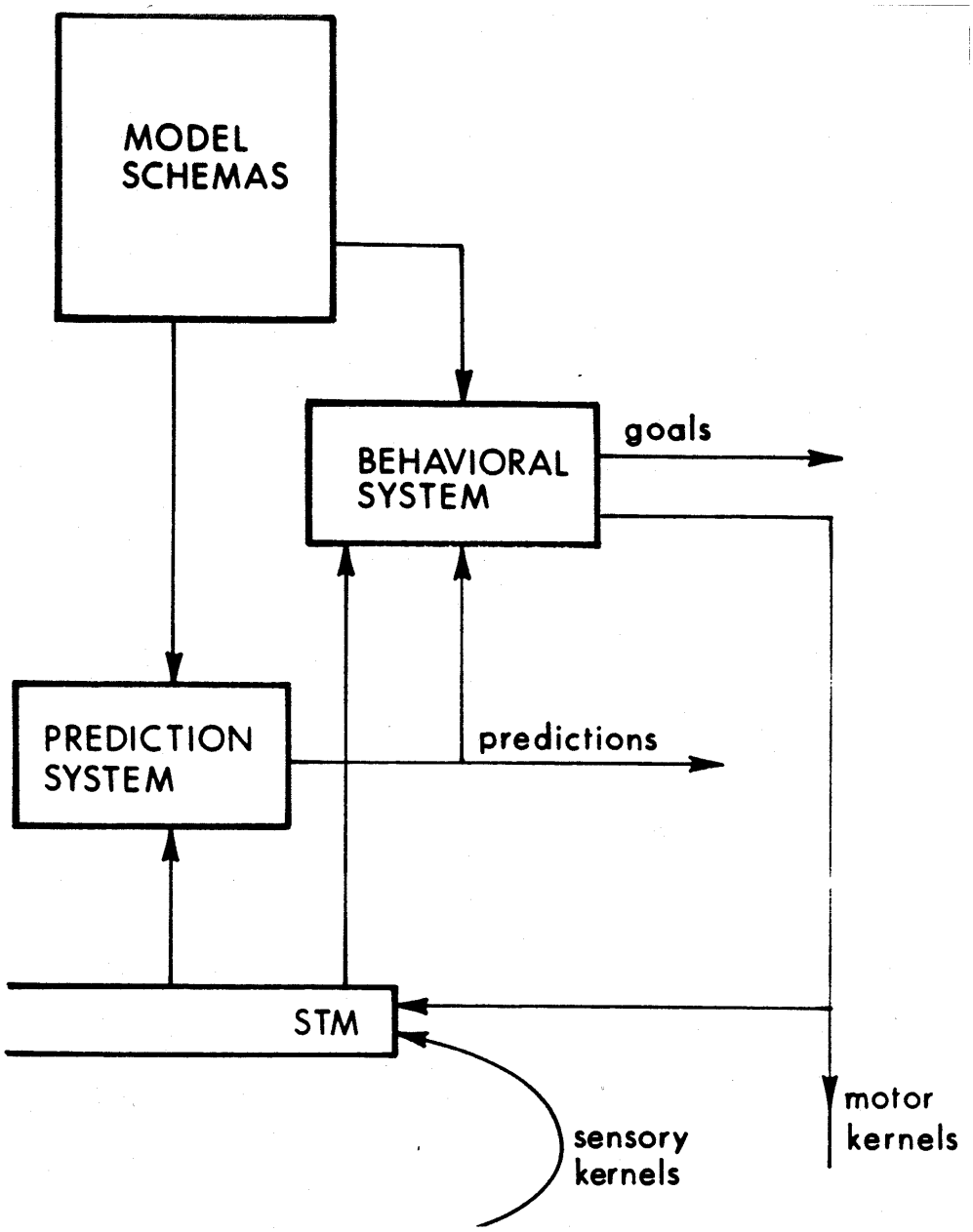


Figure 16: Prediction + Behavioral systems

5 Learning

This occurred in three ways:

1. Schema *Creation*
2. *Adaptation* of certainty factors of existing schemas
3. *Differentiation* of existing schemas

5.1 Schema creation

If unpredicted kernels occurred in STM, then a new schema was created using these kernels as the right hand side and using the kernels from ct-1 and ct-2 as the left hand side. This is shown in Figure 17. The initial values of all the certainty factors were 1.

TIME	ct-2	ct-1	ct
STM	<A>S <B>S	<M>M	<C>S <D>S
NEW SCHEMA	[<A>S <B>S -> <M>M => <C>S <D>S]		

Figure 17: Schema creation

5.2 Adaptation of certainty factors of existing schemas

This used a tagged schema from the prediction system i.e. one that had matched to STM. Adaptation was performed only if the left hand side of a schema partially matched to STM, that is if all but one of the left hand side kernels matched to STM.

We first examined the right hand side to see whether the prediction succeeded or failed. It was taken as succeeding if at least one right hand side kernel matched and if also the average certainty of occurrence of any kernels not matching was less than a threshold, (usually 0.4). Otherwise it failed.

The certainties were only adjusted when one left hand side kernel was missing from STM. They were simply recalculated from the formulae shown in Figure 18.

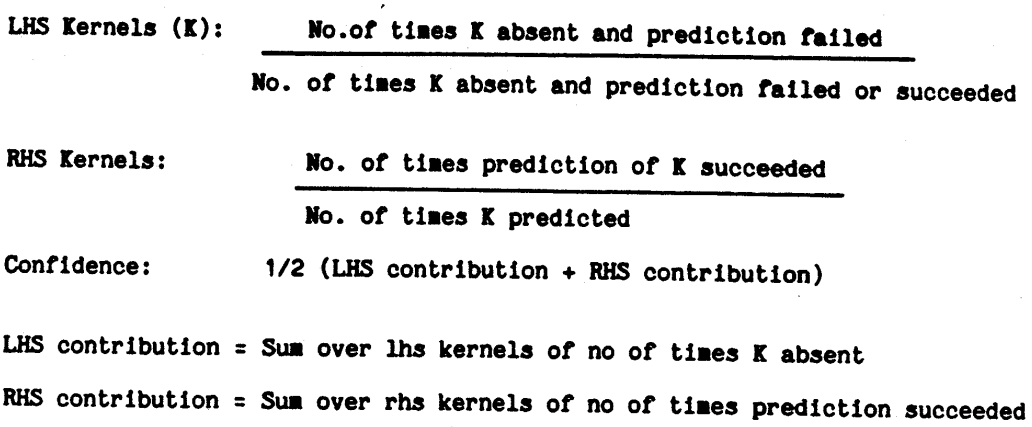


Figure 18: Updating certainty values of an existing schema

When a kernel, or an entire schema, had a very small certainty factor less then a threshold, it was deleted.

Note that we had to store the accumulated values of:

1. Number of times each kernel was absent and the schema's prediction failed
2. Number of times each kernel was absent and the schema's prediction failed or succeeded
3. Number of times the prediction of a each kernel succeeded
4. Number of times each kernel was predicted
5. Number of times each kernel was absent
6. Number of times each prediction of the right hand side of each schema succeeded

5.3 Differentiation of existing schemas

Kernels were never added to the right hand side of a schema, but could be added to the left hand side. This happened in the case that the certainty factor of the right hand side tended to a stable value which was neither 0 nor 1, indicating an insufficiently specified context by the left hand side.

The mechanism of differentiation was to try adding new detail from an event in STM. An example is shown in Figure 19.

The certainties of the new kernels were initially 1.

It was intended that the differentiated kernel was tentative and that if the newly added kernels did not increase the certainty of occurrence of the right hand side then they should be removed. Whether they would automatically "wash out" by adaptation was not clear.

We show, in Figure 20, a diagram of the total system, with all three subsystems - Prediction, Behavior and Learning.

If [<A>S -> <B>M => <C>S] present in model

and if

<X>S -> <Y>M -> <A>S -> <B>M -> <C>S occurs

then

the schema is differentiated to

[<X>S -> <Y>M -> <A>S -> <B>M => <C>S]

Figure 19: Differentiation of an existing schema

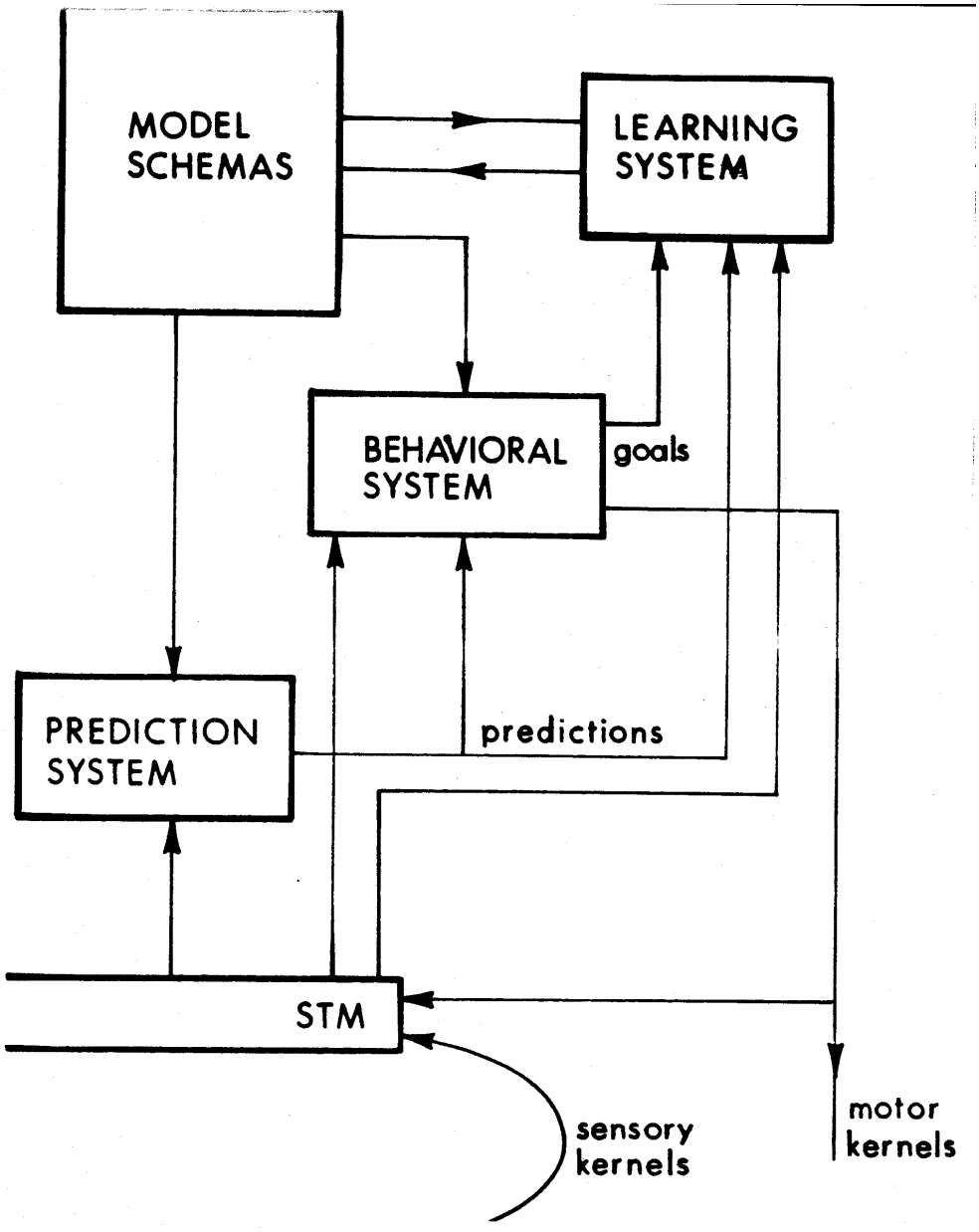


Figure 20: Diagram of total system = Prediction + Behavior + Learning

6 An Example

6.1 What was learnt

Our main example of learnt behavior is diagrammed in Figures 21 and 22. The robot was given one innate schema as shown, which caused it to push forward against a front touch stimulus. In its initial condition, it would exhibit the behavior depicted viz. random movements mainly. If it touched an object it would tend to push forward. As a result of the learning session, the system autonomously acquired a set of schema which enabled it to predict its environment.

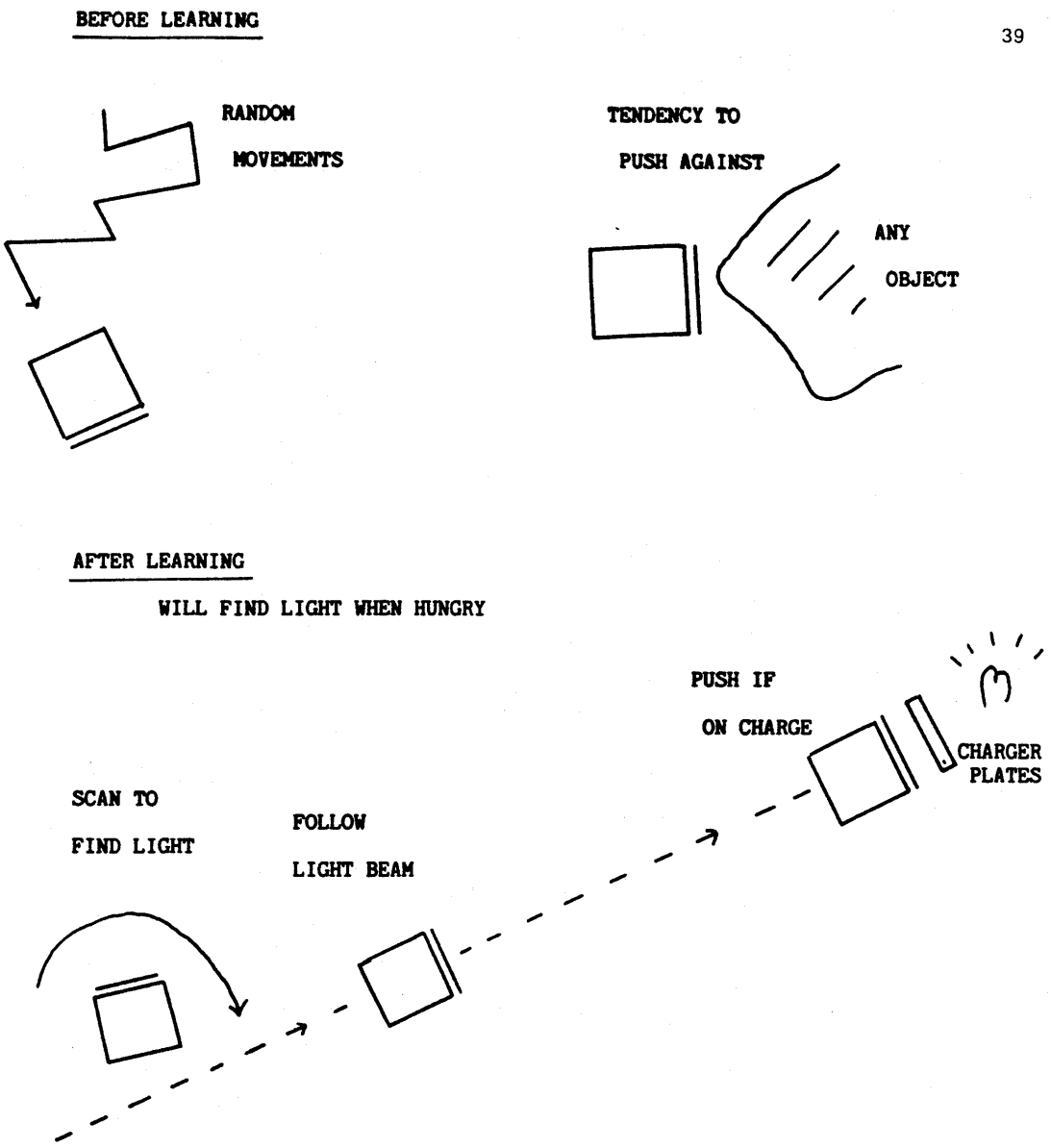


Figure 21: An example of learning - Spatial diagrams

```
Innate schema:
    [ <FRONT>S -> <FORW>M => <HIGH>S ]

Learnt schema:
    [ <HUNGER>S -> => <HUNGER>S ]
    [ <HUNGER>S -> => <LOW>S ]
    [ <RIGHT>M => <BRIGHT>S ]
    [ <BRIGHT>S -> <FORW>M => <BRIGHT>S ]
   [ <CHARGE>S -> <FORW>M => - <HUNGER>S ]
   [<CHARGE>S <FRONT>S <HUNGER>S <BRIGHT>S -> <FORW>M => - <HUNGER>S ]
```

Figure 22: An example of learning - Schemas learnt

The full set of schema learnt consisted of about 15 schema. We show in the figure the main ones that caused behavior. The effect of the learnt schemas was that the system had learnt to find the light and to charge its batteries when it was hungry. This involved learning quite a lot about its environment both internal and external.

The innate schema was

[<FRONT>S − > <FORW>M => <HIGH>S]

The system learnt

1. [<HUNGER>S − > => <HUNGER>S]

i.e. hunger tends to persist.

2. [<HUNGER>S $->$ => <LOW>S]

i.e. hunger tends to (eventually) produce pain.

3. [<RIGHT>M => <BRIGHT>S]

i.e. *scanning* behavior, if you turn or "scan" to the right, you (sometimes) see the light.

4. [<BRIGHT>S $->$ <FORW>M => <BRIGHT>S]

i.e. *light beam following*, if you see the light and move forwards, you (often) still see the light.

5. [<CHARGE>S $->$ <FORW>M => -<HUNGER>S]

i.e. *discriminating pushing* behavior unit, if you are on charge and push forwards, you will (eventually) alleviate hunger. In fact the system also learnt the more complex form

6. [<CHARGE>S <FRONT>S <HUNGER>S <BRIGHT>S $->$ <FORW>M => -<HUNGER>S]

All of these items of information about the environment were needed before the robot could survive.

6.2 The learning session

We give here an annotated trace of the entire learning session.

```
TIME          DESCRIPTION                          SCHEMA NUMBER

1      Memory loaded with the innate schema
```

[<FRONT>S -> <FORW>M => <HIGH>S] (1)
Robot not hungry

(1 13) Random movements made

14 The light was encountered, <BRIGHT>S occurred unpredicted, hence schema created

[<LEFT>M => <BRIGHT>S] (2)

16 <BRIGHT>S again occurred unpredicted, creating

[<BRIGHT>S -> <FORW>M => <BRIGHT>S] (3)

(17 30) Motor kernel <FORW>M now involved in predictions, hence exploratory behavior occured. This caused <FORW>M s to be executed along the light beam. This increased the certainty of occurrence of the right hand side of (3), but didn't affect its left hand side.

31 Robot hit charger obliquely, <FRONT>S occurred, This was unpredicted and so a schema was created

[<BRIGHT>S -> <FORW>M => <FRONT>S] (4)

(1) was also activated, executing another <FORW>M.

34 Another <FORW>M executed, <FRONT>S no longer occurring but <BRIGHT>S still occurring.

36 <FORW>M, due to exploration.

38 <RIGHT>M executed, still no <FRONT>S however.

40 <FORW>M executed, <BRIGHT>S lost.

(40 Random <BACK>S put robot away from the charger.
56)

56 Schema were now

[<BRIGHT>S 1.0 -> <FORW>M 0.75 => <FRONT>S 0.75] 0.27 (4)

[<BRIGHT>S 1.0 -> <FORW>M 0.5 => <BRIGHT>S 1.0] 0.67 (3)

[<FRONT>S 1.0 -> <FORW>M 1.0 => <HIGH>S 0.83] 1.0 (1)

The innate schema was of course subject to adaptation
and in this example would die away completely, eventually.

61 Found light <BRIGHT>S occurred, <FORW>M and <LEFT>M
executed, lost light again.

81 Found light It had just made a random right movement and
created the schema

[<RIGHT>S => <BRIGHT>S] (5)

(81 Executed <FORW>M and <LEFT>M and lost the light again.
94) Schema (3) and (4) were now

[<BRIGHT>S 1.0 -> <FORW>M 0.8 => <FRONT>S 0.63] 0.45 (4)

[<BRIGHT>S 1.0 -> <FORW>M 0.6 => <BRIGHT>S 1.0] 0.75 (3)

```
95      Found light, created

      [ <BACK>M => <BRIGHT>S ]                                    (6)


(95     <FORW>M and <LEFT>M commands executed, lost light again.
 108)

113     <HUNGER>S occurs, creating

        [ <RIGHT>M => <HUNGER>S ]                                 (7)

115     [ <HUNGER>S -> <BACK>M => <HUNGER>S ]                     (8)

        created.

160     <BACK>M in (8) eventually washed out by irrelevance in
        successful predictions, giving

        [ HUNGER>S ->  => <HUNGER>S ]                             (8)

        (4) was now

        [ <BRIGHT>S 1.0 -> <FORW>M 0.86 => <FRONT>S 0.6 ] 0.52    (4)

        (3) as before.

167     <LOW>S occurred, creating

        [ <HUNGER>S -> <LEFT>M => <LOW>S ]                        (9)

204     <LEFT>M in (9) washed out, giving

        [ <HUNGER>S ->  => <LOW>S ]                               (9)
```

Note that without <HUNGER>S being produced together with <LOW>S, this would have had a much lower certainty, and taken much longer to form.

207 Model was now

[<FRONT>S 1.0 -> <FORW>M 1.0 => <HIGH>S 0.83] 1.0 (1)

[<LEFT>M 1.0 => <BRIGHT>S 0.09] 0.15 (2)

[<BRIGHT>S 1.0 -> <FORW>M 0.6 => <BRIGHT>S 1.0] 0.75 (3)

[<BRIGHT>S 1.0 -> <FORW>M 0.86 => <FRONT>S 0.6] 0.52 (4)

[<RIGHT>M 1.0 => <BRIGHT>S 0.05] 0.1 (5)

[<BACK>M 1.0 => <BRIGHT>S 0.06] 0.1 (6)

[<RIGHT>M 1.0 => <HUNGER>S 0.08] 0.1 (7)

[<HUNGER>S 1.0 -> => <HUNGER>S 1.0] 0.55 (8)

[<HUNGER>S 1.0 -> => <LOW>S 1.0] 0.1 (9)

Due to a restart the robot was no longer hungry.

(211 <RIGHT>M executed, found light, did <LFT>M and lost it,
224) <RIGHT>M again found light, <LEFT>M kept it,
<FORW>M executed.

225 <FRONT>S occurred.

228 <PUSH>S, <CHARGE>S occurred, created the schema

[<FRONT>S -> <FORW>M => <FRONT>S <CHARGE>S] (10)

(229 258) Continued executing <FORW>M commands, maintaining <FRONT>S and <CHARGE>S, but not <BRIGHT>S, since at edge of charger. Schema (1) and (10) were altered to

[<FRONT>S 1.0 -> <FORW>M 1.0 => <HIGH>S 0.37] 1.0 (1)

[<FRONT>S 1.0 -> <FORW>M 1.0
=> <FRONT>S 1.0 <CHARGE>S 1.0] 0.55 (10)

259 <FORW>M executed, <CHARGE>S but not <FRONT>S occurred.

260 <FORW>M executed, <FRONT>S occurred.

(261 332) Continued executing <FORW> commands, schema (1) and (10) were now

[<FRONT>S 1.0 -> <FORW>M 1.0 => <HIGH>S 0.16] 1.0 (1)

[<FRONT>S 1.0 -> <FORW>M 1.0
=> <FRONT>S 0.98 <CHARGE>S] 0.55 (10)

The certainties of (8) and (9) were artificially altered by increasing their confidences:

[<HUNGER>S -> => <HUNGER>S] 0.55 (8)

[<HUNGER>S -> => <>S 1.0] 0.35 (9)

This was to allow the prediction of <LOW>s to be sufficiently certain to evoke goal seeking behavior. If the robot had been still hungry (as it should have been) on restart at 207, this wouldn't have been necessary.

333 Restart with robot very hungry, near the charger and in the light beam.

335 <FORW>M executed.

```
(337   <RIGHT>M executed, still in beam, touched charger,
  338) <HUNGER>S, <LOW>S, <FRONT>S and <BRIGHT>S occurred, but
       not <CHARGE>S. <FORW>M executed.


339    <HUNGER>S, <BRIGHT>S, <CHARGE>S and <FRONT>S occurred.
       As <BRIGHT>S not predicted, create the schema

       [ <HUNGER>S <LOW>S <BRIGHT>S <FRONT>S -> <FORW>S
                                          => -<HUNGER>S ]  (12)


347    <BRIGHT>S not predicted, createds

       [ <CHARGE>S <BRIGHT>S <FRONT>S <HUNGER>S -> <FORW>M
                                            => <BRIGHT>S ] (13)


(348   Certainty of (12) rose slowly, certainty of (13) fell.
  361)
```

The model was now sufficient to find the light and to charge the batteries, when hungry.

(12) is discriminating pushing

(5) is scanning

(3) is light beam following

(8) is hunger persistence

(9) is hunger predicts pain.

7 Discussion

Although based upon the original Becker model, substantial modifications and improvements have been introduced in our system.

Some changes were made in order to make the basic mechanisms work:

1. The prediction system was separated out as a separate, and parallel mechanism.
2. Negated kernels were introduced to represent the absence of a kernel. This was needed to compare with the prediction of the kernel in a given time slot. It also allowed the representation of avoidance of a given event, and the derivation of avoidance subgoals.
3. The creation mechanism for new schema was introduced. This was designed in a fairly conservative manner, but was found to be effective for the class of learning tasks investigated.

There were also major modifications and extensions, going beyond Becker's conception:

1. The motivational theory of the system, using primitive pleasure and pain events was introduced.
2. An ad hoc theory of uncertainty was developed.
3. Time ordering of events, predictions and goals was made into a precise discrete scale of time slots. This method of representation is potentially cumbersome, however it was very powerful in controlling the proliferation of predictions, in pruning unwanted and out of date goals, and in evaluating the predictive power of schemas.

This work has allowed the investigation of a very fertile area, namely learning systems in relation to mobile robots. Among the many questions arising, we mention the following:

1. How can this type of representation represent geometric concepts and objects? It was developed as a compromise to allow learning mechanisms to function.
2. Can we understand its representational ability in terms of formal language?
3. Can we extend the theory of grammatical inference to this system?
4. What classes of schema can be learnt using these learning mechanisms? Are there clear counterexamples of learning problems that this type of representation and learning mechanism cannot handle?
5. Does the need to learn in a proliferation of predictions force a linearisation of the basic structures, such as the goal and prediction queue, or can this be controlled in a more general way?

Acknowledgements

We should like to thank Mark Witkowski for designing and building the robot, and Roger S. Brown, for system support to the project, both of the Artificial Intelligence Laboratory, Queen Mary College. This project was supported by the Science and Engineering Research Council.

References

[1] T. L. Jones. A computer model of simple forms of learning. Technical Report MAC-TR-20, AD-720-337, Massachusetts Institute of Technology, 1971.

[2] L. Friedman. Instinctive behavior and its computer synthesis. *Behavioral Science*, 12:85–108, 1967.

[3] J. E. Doran. Experiments with a pleasure-seeking automaton. In *Machine Intelligence*, volume 3, pages 195–216, 1968.

[4] L. Uhr and M. Kochen. Mikrokosms and robots. In *International Joint Conference on Artificial Intelligence*, pages 541–555, 1969.

[5] R. E. Fikes, P. E. Hart, and N. J. Nilsson. Learning and executing generalised robot plans. *Artifical Intelligence Journal*, 3:251–288, 1972.

[6] J. H. Andreae. *Thinking with the teachable machine.* Academic Press, New York, 1977.

[7] M. H. Raibert. Mechanical arm control using a state space memory. Technical Report MS77-750, Soc Mfg. Engrs., 1977.

[8] H. A. Ernst. Computer controlled robots. Technical Report RC2781, IBM, 1970.

[9] C. M. Witkowski. Mark 4 Robot Manual. Technical report, Artificial Intelligence Laboratory, Queen Mary College, London, 1979.

[10] C. M. Witkowski. Introduction to robotics. *Practical Computing*, February - August, 1980.

[11] Alan H. Bond. Fast Vision for a Low Cost Computer Controlled Robot. In *On Theory and Practice of Robots and Manipulators edited by Morecki A. and Kedzior K, Proceedings of 2nd CISM-IFToMM Symposium, P.W.N., Warsaw*, 1976.

[12] J. D. Becker. *An information processing model of intermediate-level cognition.* PhD thesis, Stanford University, 1970. Artificial Intelligence Project, AI-119.

[13] J. D. Becker. A model for the encoding of experiential information. In R. C. Schank and K. M. Colby, editors, *Computer models of thought and language*, pages 396–434. W.H.Freeman, San Francisco, 1973.

[14] A. H. Bond and D. H. Mott. The investigation of learning in a mobile robot. Technical Report Final Report, Science Research Council, 1981.

[15] David H. Mott. *Sensory-Motor Learning in a Mobile Robot.* PhD thesis, London University, 1981.

Contents

1 Introduction **3**

2 The robot and its environment **4**

3 Elements of the system **11**

3.1 Short term memory - STM 13

3.2 Schemas and the Model 15

3.3 Negated kernels 16

3.4 Prediction 17

4 Behavior **24**

4.1 Random Movement 24

4.2 Innate reflexes 25

4.3 Exploratory Behavior 25

4.4 Goal Seeking Behavior 25

5 Learning **29**

5.1 Schema creation 29

5.2 Adaptation of certainty factors of existing schemas 30

5.3 Differentiation of existing schemas 31

6 An Example **34**

6.1 What was learnt 34

6.2 The learning session 37

7 Discussion **44**

List of Figures

1 Photograph of the robot 4

2 Diagram of the robot 5

3 Photograph of the robot in its environment 6

4 Diagram of the environment - The spatial and external sensor environment . 7

5 Diagram of the environment - The energy and internal sensor environment . 8

6 Diagram of the environment - The movement environment 9

7 Diagram of the computer system 10

8 I/O vocabulary, kernels, used 11

9 Short term memory, events and schemas 14

10 The prediction system 19

11 Prediction - The prediction process 20

12 Prediction - A prediction tree 21

13 Prediction - A prediction queue 22

14 Behavior - A goal tree 26

15 Behavior - A goal queue 27

16 Prediction + Behavioral systems 28

17 Schema creation 29

18 Updating certainty values of an existing schema 30

19 Differentiation of an existing schema 32

20 Diagram of total system = Prediction + Behavior + Learning 33

21 An example of learning - Spatial diagrams 35

22 An example of learning - Schemas learnt 36