

*Proceedings of the 3<sup>rd</sup> Congress of the International Federation of Automatic Control (IFAC), London, (1966) pp.342-9. [Paper 14.B. Session 14. Tuesday 21st June 1966]*

## **A Learning Machine in the Context of the General Control Problem**

B. R. Gaines\*† and J. H. Andreae\*

\* Standard Telecommunication Laboratories Ltd., Harlow, Essex.

† Psychological Laboratory, Cambridge.

### **Introduction**

Advances in automatic control theory may be seen to have two main objectives: the synthesis of improved controllers for a given plant and the extension of the range of plants considered for control. Optimal control theory and the application to linear systems is an example of the former, whilst advances in the latter have been largely concerned with describing-function and quasi-linearization techniques for the linear approximation of non-linear but reasonably continuous plant.

It has become conventional to introduce the abstract concept of a plant through its state-transitions<sup>1</sup> rather than through the differential or difference equations which describe them in particular instances. However, the powerful and restrictive simplification that the plant should be linear is made immediately, so that the operational and matrix calculi developed for linear systems may be used to synthesise a controller. Essentially non-linear systems<sup>2</sup> are approximated in the time or frequency domain by quasi-linearization or describing functions, and those whose transitions are indeterminate through inadequate state or input description are treated as noisy or time-varying.

Application of the maximum principle to autonomous, completely identified plant leads to equations for the optimal, non-linear control policy, but the computational problem of solution under a variety of performance criteria are certainly not trivial<sup>3</sup>. For linear plant with unknown parameters, on-line identification and synthesis by adaptive control theory has been extended to take into account uncertainty about state variables or transitions<sup>18,4</sup>. Thus the emphasis has been upon the synthesis of optimal, non-linear controllers for linear or linearized plant, rather than upon control by any means of plant which may not satisfy the conditions of linearization.

Alternative approaches to control problems have been made under the guise of learning machines or artificial intelligence<sup>5</sup>, where the controller is often ridiculously sub-optimal for a given plant but is capable of controlling a wide range of logical as well as numerical environments. As the survey paper at the Second IFAC Congress<sup>6</sup> demonstrated, this work has been extensive and the games-playing and pattern-recognition environments taken as plant for control have raised problems of a different nature from those of the linear or quasi-linear plant considered in automatic control theory. However, at the state-transition level of the general control problem the differences are much less apparent and the schemes proposed for learning machines may be seen to be closely related to the adaptive controllers of automatic control. Comparison

is generally difficult not only because of disparity between plant considered for control, but also through lack of coherence in work on artificial intelligence which has ramified without general direction.

In this paper the STeLLA learning scheme, which was described at the Second IFAC Congress<sup>7</sup> in the context of games-playing and pattern-manipulation environments, is examined both for the nature of its alternative simplification of the general control problem and for the relationship of its control strategies to those proposed in adaptive control. These strategies are exemplified by its behaviour in controlling a second-order, sampled-data, stochastic plant with bounded phase-space and transport-lag; this is a common type of plant which simulates the situation faced by one of the few fully-commissioned adaptive controllers, the automobile driver!

### **Vehicle-Steering as an Environment**

STeLLA, the Standard Telecommunication Laboratories Learning Automaton, is a storage-limited, optimizing controller which has been extensively simulated in a variety of plants or environments. For purposes of comparison with other controllers this paper will take the minimal-path aspect of a simple control task to illustrate the various strategies of adaptive policy-formation used by STeLLA.

To make the plant itself meaningful we have simulated fairly closely the parameters and dynamics of an automobile being driven at constant speed, although the output in the form of a binary pattern and the on-off input to the steering wheel would not be relished by a human operator! STeLLA has to learn to steer a car travelling at 30 mph down the centre of a 40 foot wide straight road, using one of three possible control actions, wheel right, left or centre. Feedback information is presented as a binary pattern of 10 features, present or absent, relating to the quantized position and angle of the vehicle relative to the road; these are sampled at 200 msec intervals 50 msec before each steering action takes effect. Both position and angle are bounded in that the vehicle can never run back along or off the road and, should it hit the side of the road, it 'rebounds' at an angle of  $9^\circ$ . Random noise in the control output in the form of indeterminacy and 'skids' plus a camber effect which turns the wheels away from centre add to the difficulty of control.

Since the plant is second-order, the phase plane portrait is particularly convenient for representation of the state-space and state-transitions. Data sampling and on-off control imply that trajectories are piecewise continuous and that out of each point of discontinuity can arise one of three arcs subject to slight random variation. In figure 2(b) a typical trajectory is shown as a position against time plot along the road and in 2(a) as a position against angle plot in the phase plane.

The binary representations of position and angle together form a 10-bit feedback pattern for the machine and the distinguishable states are shown by a grid over the phase plane. This quantization makes the state-transitions not merely stochastic but also indeterminate, in that transition probabilities are not state-determined, although individual transitions cannot be distinguished from those of a time-varying stochastic system.

The demanded region marked in the centre of the phase plane corresponds to the vehicle being within 5 feet of the centre of the road and within  $11^\circ$  of straight. When the

actual trajectory is within this region a binary signal given to the controller changes from 0 to 1, informing it that demand has been attained. This 'reward' signal is the only goal-oriented information given to STeLLA in order to effect control.

## **STeLLA's Strategies**

The purpose of STeLLA's strategies is to synthesise a control policy which maximizes the expected rate of receipt of the reward signal. This criterion is sufficient to define optimal solutions of both the minimal-path problem of constructing a trajectory utilizing the smallest number of control actions to get from the actual state to a demanded state and also the stability problem of remaining within the demanded region. In the vehicle-steering environment both problems arise but we shall emphasize the minimal-path aspect for purposes of comparison with other controllers.

Given an on-off input to the 'plant' and sampled feedback, one form of optimal policy<sup>8</sup> will consist of a division of phase space into cells ordered from demand, to which are attached control actions implemented when the feedback sample falls within the cell. A trajectory commencing in a cell  $n$  steps away from demand will have its next point of discontinuity in a cell  $(n-1)$  steps away so that the policy elements consisting of cell and control action form natural sequences to the demanded region. Quantization of the state space for purposes of feedback causes transitions to become indeterminate but these may be regarded as generated by time-varying stochastic processes<sup>9</sup>, and the ordering of policy elements must then be based on the expected number of steps to demand.

STeLLA's strategy of sequential-policy formation is an algorithm for synthesising such policies on-line, one step at a time from demand. Her strategy of state-generalization is an algorithm for varying the size of the cells. Her strategies of random and predictive search are means of manipulating the control actions in a random or model-directed manner when the feedback does not fall within an established cell so that it eventually comes to do so. Figure 1(b) shows schematically how the strategies appear as reaching forwards from the actual state in predictive search and working backwards from the demanded state in sequential-policy paths. In the following section these strategies are discussed in detail.

### **1. Search**

Since nothing is known initially of the demanded state or the plant-transitions, random manipulation of control actions is the only policy which guarantees that demand will be reached given that the state-transition graph is strongly-connected. Attempts to improve this strategy by restricting actions are apt to lead to trapping cycles which make the demanded state unobtainable<sup>10</sup> and those which utilize feedback before the goal is reached may yield an even more inefficient procedure than random search<sup>11</sup>. It is possible to bias the random selection of actions in such a way as to improve efficiency, for example by decreasing the probability of the same control action occurring again for given feedback<sup>7</sup>, but this requires storage which might be better used elsewhere.

One trajectory which enters the demanded region under random search is shown in figure 2. In the vehicle-steering plant it takes on average 8 seconds or 40 control actions for demand to be reached under this strategy, but variations about these figures is

large. The use of any prior information to improve the search procedure would be advantageous, but in general such information may be false and a safety routine to take the controller back to random search is essential. This is therefore the basic strategy to which the controller reverts when none other is available.

## **2. Sequential-Policy Formation.**

The minimum-time problem in a synchronous sampled-data, deterministic environment reduces to finding shortest paths between initial and final states on the graph of state-transition. In general only one optimal path may leave any state and, given the complete graph, a dynamic-programming technique for finding the optimal policy could be used. This would determine all states which have a one-step transition to demand, and then all those states which have a one-step transition to these and so on<sup>8</sup>. Eventually optimal paths would have been mapped through all controllable initial states and, after rejecting redundant paths, the control policy could be formulated.

This computation of optimal policies back from demand has the disadvantage of generating a high proportion of irrelevant paths which cannot be discarded until the final stage<sup>12</sup>. Also the technique is not directly applicable to stochastic or indeterminate environments where an optimal solution, if one exists, must be obtained by approximation and iteration. The strategy of sequential-policy formation used by STeLLA is similar to dynamic programming but utilizes the information available through on-line policy-synthesis to generate paths which are relevant to the particular control task.

Under random search the plant will eventually be controlled to the demanded region and STeLLA will receive a reward signal informing her that demand has been met. The last feedback pattern received together with the control action performed is then stored as a policy element connected to reward, i.e. leading to demand. When this pattern is received again the attached action, having led directly to reward, is optimal if the environment is deterministic or indeterminate. Hence STeLLA implements the policy element, performing the attached action expecting to attain demand. At the same time the previous pattern and action are stored as a policy element connected to the first which is already connected to reward. Storage of policy elements one step at a time favours shorter paths and, by on-line iteration of this procedure, STeLLA builds up near-optimal paths terminating in reward.

Path formation not only synthesises a control policy but also identifies the plant; the latter process may be refined to deal with stochastic or indeterminate plant by estimating the conditional probability of each stored transition both occurring and causing an increased expectation of reward. With this additional information STeLLA is able not only to choose between alternative paths by comparing expectations of leading to reward but also to adapt the policy to environmental changes by erasing those transitions which become of low probability. The probability estimates are decremented with lack of use of the policy element concerned so that the strategy is adaptive not only to actual changes in the plant but also to changing relevance of individual policy elements with improvement of the overall strategy. The expectation of reward gives an ordering of patterns along paths which enables the machine to determine whether implementation of

a policy element has caused an advance towards reward, and hence ensures the convergence of trajectories generated by the policy. This is illustrated in figure 1(c).

### **3. State-Generalization**

The strategy of sequential-policy formation may be seen as a practical solution to the problem of identifying a plant whilst improving control over it, since it utilizes limited storage to identify that part of the environment relevant to the control problem and in so doing generates a control policy. Given the necessary storage, this strategy would be sufficient for control and, if there is no structure in the state description, some technique of path-formation is all that can be used. In plant represented by linear systems, however, one of the most powerful properties is the implied metric of similarity upon the state space and one would expect a similar but weaker property in all but the most pathological state spaces of non-linear plant.

In a linear system a small change in state-vector for a given control action causes a small change in the succeeding state-vector so that an action which is optimal in one state will be nearly optimal in the neighbourhood of that state. In linear systems this property has the unwanted further implication of complete extension from local to global so that identification of any small region of the plant is sufficient to identify the entire plant. However, the concept of a metric neighbourhood, although developed in linear vector spaces, may be extended to non-linear systems without the implication from local to global; this forms the basis of STeLLA's strategy of state-generalization.

At the interface (figure 1(a)) feedback from the plant is coded for the controller and this code is assumed to carry a simple metric structure for measurement of similarity between feedback patterns. STeLLA stores with each pattern forming part of a policy element the parameters of its local metric initially set at some given value and uses a policy element not only when the feedback pattern is the same as the stored pattern but also when it is a sufficiently small distance away. The success or failure of this generalization, determined by whether the machine continues to advance along a path to reward, determines whether the stored parameters of the local metric are adjusted so as to increase or decrease the generalized region. The policy is thus adaptive and, although a good coding of the feedback will benefit the machine, a bad one will be rejected by it.

In the vehicle-steering environment the feedback pattern is a binary  $n$ -tuple ( $n = 10$ ) representing the presence or absence of features of the position and angle of the vehicle across the road. A weighting  $n$ -tuple is stored with each pattern forming part of a policy element and is used to define the 'deviation' of any feedback pattern from the stored pattern: each feature which differs between the patterns contributes its appropriate weight to the deviation. If the deviation is small enough the feedback pattern is regarded as the same as the stored pattern and the appropriate control action is implemented. If this generalization succeeds and the expectation of reward rises, the weights of the neglected features are decreased making similar generalizations more likely, but otherwise they are increased making them less likely. Should several policy elements have patterns with small enough deviations, one of these is selected for implementation by a random process weighted according to their expectation of reward.

### **4. Prediction by Model-Reference.**

In the minimal-path problem a trajectory is to be constructed from an actual state of the plant to a demanded state. So far the strategies considered have been largely concerned with the terminal boundary at demand and no strategy has been proposed to enable the machine to formulate a forward path meeting those from demand. However, some form of predictive control, in which estimates are made of the changes in state under various control actions, would enable the controller to 'explore' the region around the plant's actual state for states which are on stored paths to demand. Thus the search strategy could be directed so as to increase the probability of entering a region where the generalized sequential policy might be used.

Identification of state-transition would enable predictions to be made, but the storage required would be vast and unwieldy and better used for sequential policies. Instead the technique of adaptive model reference<sup>13</sup> may be used and some assumed structure fitted to the state-transitions by statistical estimation. Even if this structure is not very realistic its local predictions may be usefully accurate. The use of a simple linear model for a complex plant with more poles or non-linearities is a basic technique of adaptive control<sup>14</sup> and, in a non-linear plant which cannot be thus approximated, some form of Bayesian predictor may be employed. The exact form again depends upon the assumed structure of state-description. In the vehicle-steering environment, where feedback is presented as a binary pattern, STeLLA uses logical functions of the feedback-pattern features as evidence from which to predict features of the succeeding pattern under a given control action.

In the simulation illustrated by the figures (1-3) the predictor is used to direct search and to control the extent of generalization by checking the anticipated transitions of the stored sequential policy. In directing search, the predictor projects a path forward from the present state step-by-step until its predictions become unreliable or until the projected path meets one stored as a sequential policy. In the former event STeLLA resorts to random search, but in the latter the appropriate control actions are implemented in an attempt to follow the predicted path.

## **Discussion.**

STeLLA may be seen as an attempt to synthesise a widely applicable and readily fabricated control scheme. The goal differs from that of synthesis techniques such as dynamic programming which emphasise the computation of control policies rather than their implementation or the gathering of knowledge for computation. Adaptive controllers which identify, compute and implement have generally been specific to a particular plant, whereas viability in a wide range of applications requires a non-specific control scheme which makes few assumptions about the plant and is readily fabricated. For STeLLA to assume that the actual plant transitions and demanded states are initially unknown helps to realize both aims since the controller has initially a simple homogeneous structure and the plant has only to satisfy the assumption that it is a strongly-connected stochastic automaton.

In real control problems prior information about the plant must be used, where possible, to improve both the rate of adaption and the ultimate control policies of the adaptive controller. For example, the feedback of position and angle from the vehicle steering plant was put into a Gray code for STeLLA since this most nearly matched the

generalization metric to the topology of phase space. On one occasion the learning machine was 'primed' with an initial control policy to reduce the duration of search. Initial control policies may also be given to the controller by 'training' it along restricted trajectories. Increasing attention about the plant<sup>15,16,17</sup> and techniques such as coding, priming and training will undoubtedly be necessary for the practical application of adaptive control.

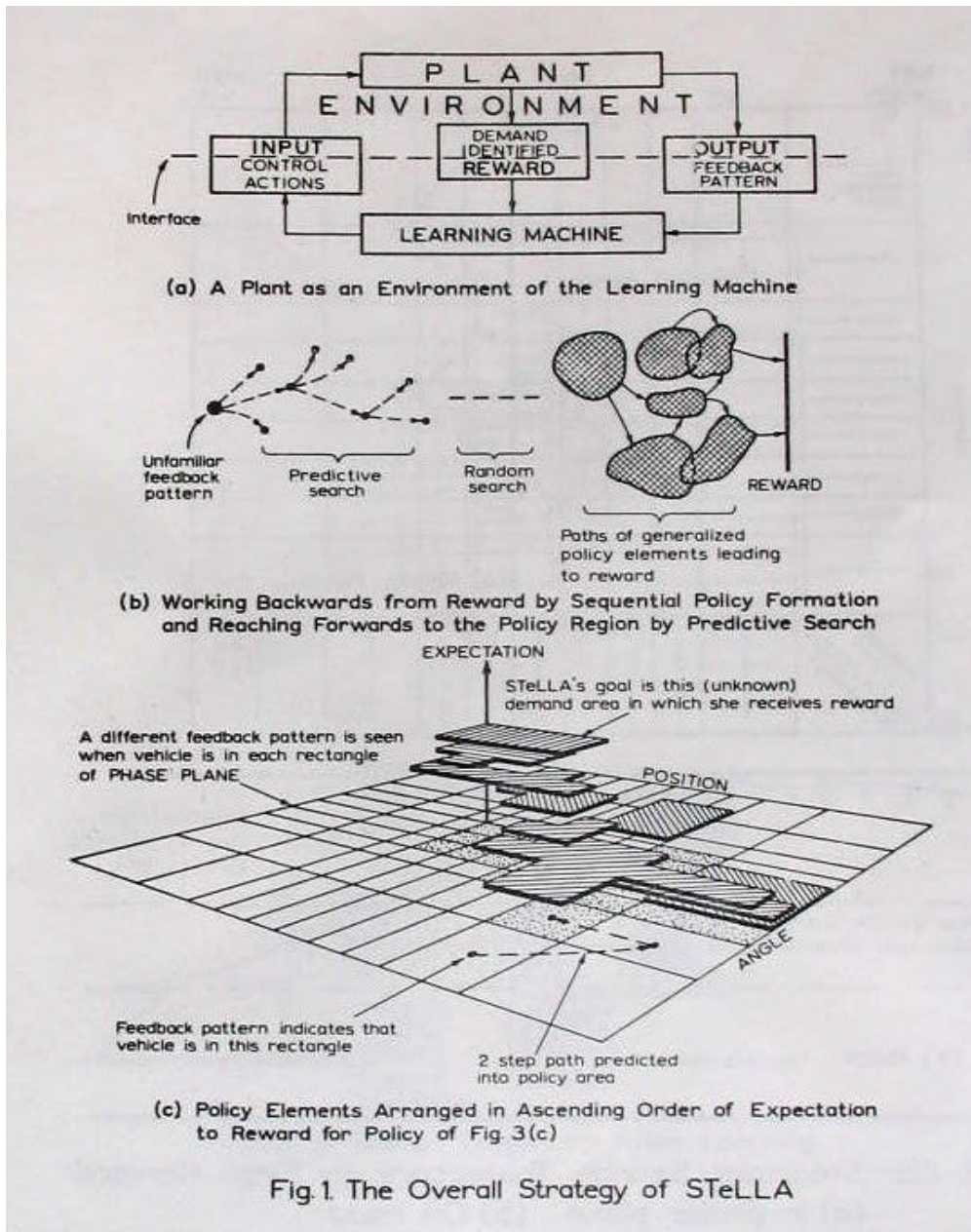
It is customary to think of controllers in terms of optimality but, when the plant is indeterminate or time-varying and the controller itself is required to be widely applicable, such a concept loses much of its force. When fabrication and storage costs have also to be taken into account, one can only ask whether satisfactory control is possible and, if so, how much it will cost.

### **Acknowledgements.**

We should like to thank Messrs D. R. Hill and O. E. Morgan for criticism, and Imperial Chemical Industries Ltd. for computer time on their KDF9.

### **References**

1. Zadeh L.A. and Desoer, C.A.; Linear System Theory, McGraw Hill, 1963.
2. Gibson, J.E.; Non-linear Automatic Control, McGraw-Hill, 1963.
3. Boyadjieff, G., Eggleston, D., Jacques, M., Sutabutra, H. and Takahashi, Y.; J. Basic Engineering **86D** 11 (1964).
4. Feldbaum, A.A.; Proc. 2<sup>nd</sup> IFAC Congress, Basle, 1963.
5. Feigenbaum, E.A. and Feldman, J.; Computers and Thought, McGraw-Hill, 1963.
6. Pask, G; Proc. 2<sup>nd</sup> IFAC Congress, Basle, 1963.
7. Andreae, J.H.; Proc. 2<sup>nd</sup> IFAC Congress, Basle, 1963.
8. Desoer, C.A. and Wing, J.; IRE Trans. **AC-6** 5, 111 (1961)
9. Kalman, R.E.; Proc. Symp. Non-linear Cct Analysis, p.273, Interscience, 1957.
10. Bremermann, H.J. and Salaff, S.; Tech. Rept., Univ. of California, Nov. 1963.
11. Friedberg, R.M., Dunham, B. and North J.H.; IBM J. **3** 282 (1959)
12. Westcott, J.H.; Paper S.3, Advances in Automatic Control, Nottingham, 1965.
13. Blandhol, E. and Balchen, J.; Proc. 2<sup>nd</sup> IFAC Congress, Basle, 1963.
14. Mishkin, E, and Braun, L.; Adaptive Control, McGraw-Hill, 1961.
15. Clark, J.M.C.; Paper 13. Advances in Automatic Control, Nottingham, 1965.
16. Kalaba, R.; J. Proc. Symp. Appl. Maths. **14** 173 (1962).
17. Aström, K.J.; J. Math. Anal. Applications **10** 174 (1965).
18. Rosenbrock, H.H.; Automatica **1** 263 (1963).



The STeLLA path memory stores a policy comprising adaptive policy elements (PE). Each PE stores a control action, a feedback pattern with 'pattern digit weights' to determine a set of patterns equivalent by generalization, and a number of sequence connections which anticipate on the basis of experience successful consequences of performing the prescribed action given one of the equivalent patterns. A 'successful consequence' is either the use of a PE with high expectation, or reward. The expectation, which is the estimated probability of reaching reward via the policy, sets the PEs in an ascending order of 'closeness to reward' and thereby ensures a convergent policy.

With a feedback pattern not covered by the policy, STeLLA attempts to project a path forwards into the policy region by predictive search using adaptive model reference as suggested in (c). In this diagram the policy of figure 3(c) for the vehicle-steering environment of figure 2 is shown as a hill of PEs rising to a summit at reward.

If this fails, STeLLA reverts to random search.

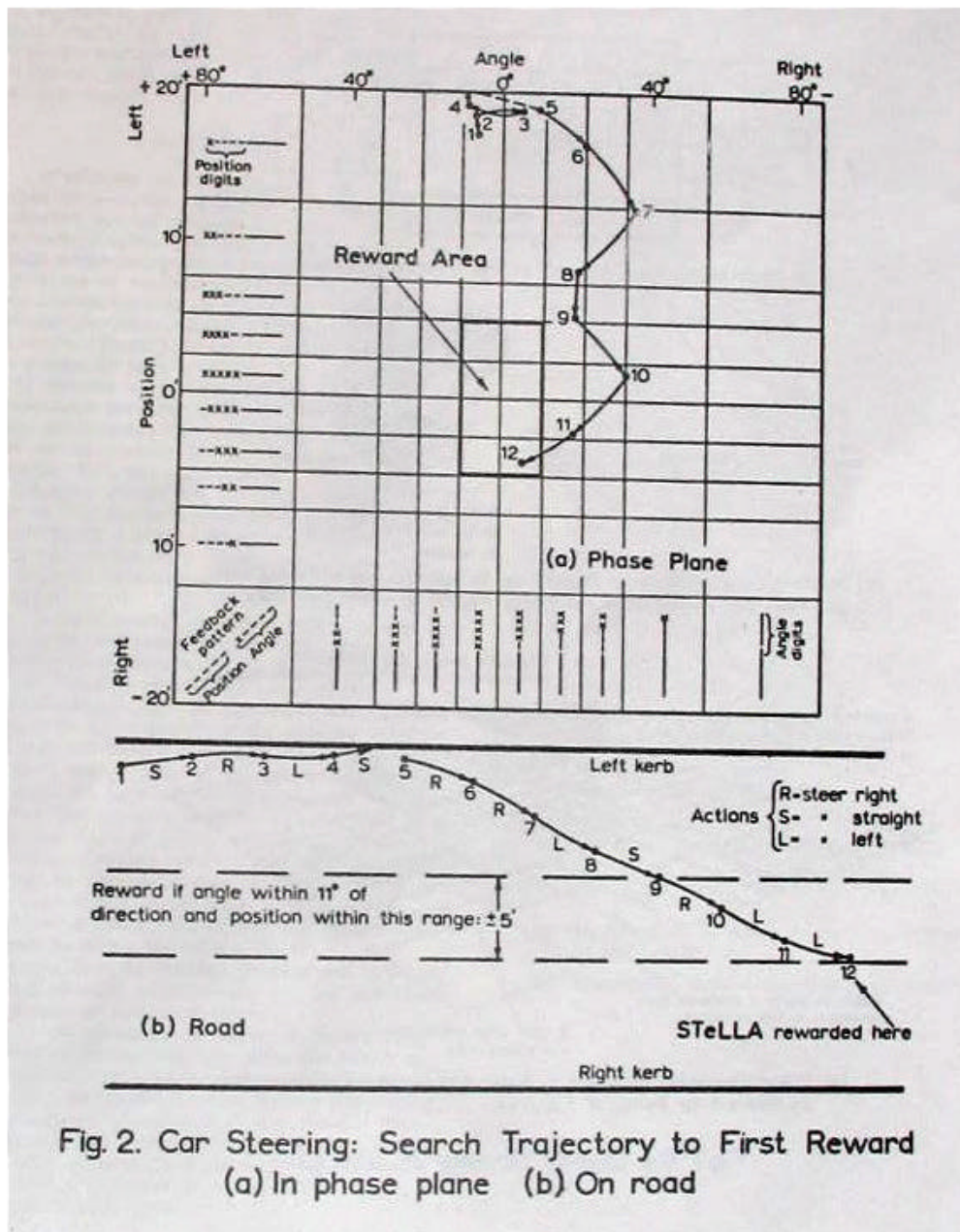
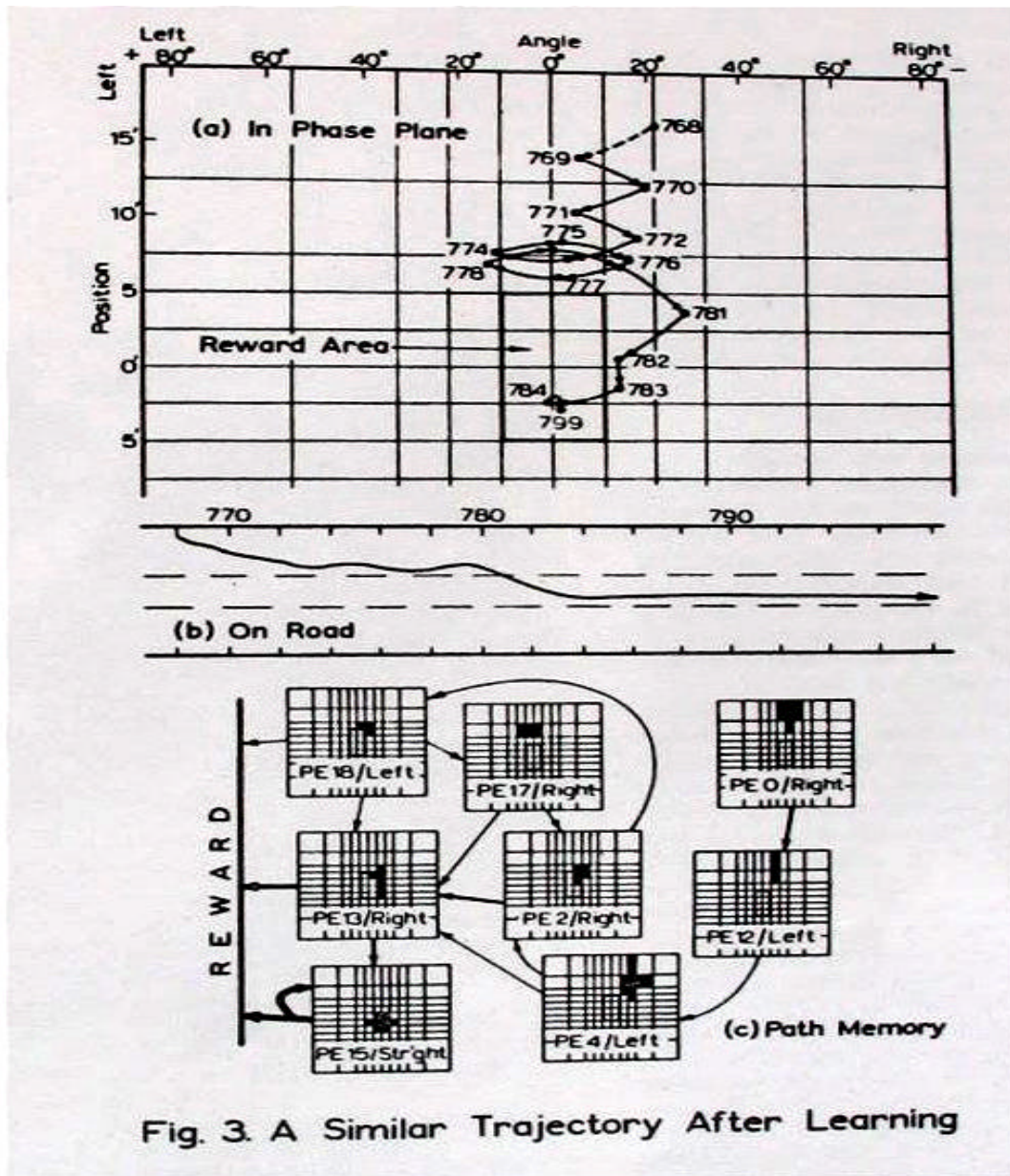


Fig. 2. Car Steering: Search Trajectory to First Reward  
 (a) In phase plane (b) On road

The three actions are selected randomly in this search strategy. Reward on reaching position 12, which is within the reward area, causes STeLLA to store the last action to position 12 (Left), the feedback patten (-XXXX--XX) sampled at position 11 and a sequence connection 'to reward'. The pattern is derived from the actual position  $\frac{3}{4}$  of the way from positions 10 to 11 because of sampling delay. The pattern is composed of 5 position digits (-XXXX) and 5 angle digits (---XX) which prescribe one of the rectangular areas in the phase plane. From position 4 a straight line took the vehicle into the kerb whence it was 'rebounded' on to the road in position 5.

Computer simulation: Carstella 11-2-1965 ICI/C/1043F/24.1 40' wide road. 30mph. Sampling interval: 0.2msec. Sampling delay: 0.05sec. Turning radius: 35'. Turning angle:  $14.3^\circ + 0.07^\circ/\text{ft}$  camber  $\pm 0.14^\circ$  indeterminacy  $\pm 7.1^\circ$  random skids with probability of 1 in 50 per interval.



**Fig. 3. A Similar Trajectory After Learning**

(a) In phase plane. The decision to steer left from positions 768 to 769 was taken with the help of the predictor. All actions from positions 769 on were decided by policy elements (PEs) of the policy in the path memory: 769 (PE 0); 770 (PE 12); 771,774,775,778,779 (PE 17); 772,773,777 (PE 18); 776,781 (PE 4); 780 (PE 2); 783 (PE 13); 782,784...798 (PE 15).

(b) On road.

(c) Part of STeLLA's path memory at the start of the trajectory. Each PE stores an action, a pattern and one or more sequence connections. The pattern (including pattern digit weights) indicates conditions under which the action has been found successful. The conditions are equivalent here to areas in the phase plane and, instead of the pattern, we show for each PE the relevant area of the phase plane. See also figure 1(c). The successful consequences of performing the action of a PE under the conditions imposed by the patterns are stored as sequence connections from a PE to other PEs or to reward. Strongly confirmed connections are indicated here by thick arrows.

Computer simulation: Carstella 8-3-1965 ICI/C/1043F/24.5.