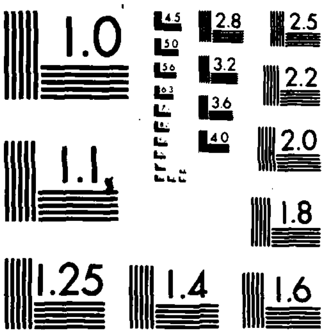
AD-A137 725 STUDY ON EXTREMIZING ADAPTIVE SYSTEMS AND APPLICATIONS 1/1
TO SYNTHETIC APERT..(U) ENVIRONMENTAL RESEARCH INST OF
MICHIGAN ANN ARBOR RADAR DIV D T POLITIS ET AL. OCT 83
UNCLASSIFIED ERIM-163800-2-T AFOSR-TR-84-0024 F/G 17/9 NL

END
FILMED
3-84
DTIC


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

4

163800-2-T

Annual Technical Report

STUDY ON EXTREMIZING ADAPTIVE SYSTEMS AND APPLICATIONS TO SYNTHETIC APERTURE RADARS

DEMETRIOS T. POLITIS
WILLIAM H. LICATA

Radar Division

OCTOBER 1983

Air Force Office of Scientific Research
Bolling Air Force Base
Washington, D.C. 20332
AFOSR Contract Number F49620-82-C-0097

ENVIRONMENTAL
RESEARCH INSTITUTE OF MICHIGAN
BOX 8618 • ANN ARBOR • MICHIGAN 48107

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1 REPORT NUMBER
AFOSR-TR- 84-0024

2 GOVT ACCESSION NO
AD-A137 735

3 RECIPIENT'S CATALOG NUMBER

4 TITLE *(and Subtitle)*
Study on Extremizing Adaptive Systems and
Applications to Synthetic Aperture Radars

5 TYPE OF REPORT & PERIOD COVERED
Annual Technical Report
10 Sept. 1982-9 Sept. 1983

6 PERFORMING ORG REPORT NUMBER

7 AUTHOR*(s)*
Demetrior T. Politis and William H. Licata

8 CONTRACT OR GRANT NUMBER *(s)*
F49620-82-C-0097

9 PERFORMING ORGANIZATION NAME AND ADDRESS
Environmental Research Institute of Michigan
Radar Division
P. O. Box 8618, Ann Arbor, Michigan 48107

10 PROGRAM ELEMENT PROJECT TASK AREA & WORK UNIT NUMBERS
61102F
2312A1

11 CONTROLLING OFFICE NAME AND ADDRESS
U.S. Air Force Office of Scientific Research/NL
Bolling Air Force Base
Washington, D.C. 20332

12 REPORT DATE
October 1983

13 NUMBER OF PAGES
83

14 MONITORING AGENCY NAME AND ADDRESS *(if different from Controlling Office)*

15 SECURITY CLASS *(of this report)*
Unclassified

15a DECLASSIFICATION/DOWNGRADING SCHEDULE

16 DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release;
distribution unlimited.

17 DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18 SUPPLEMENTARY NOTES

19 KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Adaptive control, artificial intelligence, synthetic aperture radar, autofocus, learning networks

20 ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

For this study the application of Artificial Intelligence methods in synthetic aperture radars (SAR) is investigated. It was found that the neuron-like ASE-ACE adaptive algorithm developed by Barts, operating in the extremizing mode suggested by Klopf, can be used in a wide class of engineering problems requiring that some performance function be minimized. One such suggested application is to correct for quadratic phase errors in SAR signal processing.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES
(Concluded)

1
INTRODUCTION

This project was motivated by A.H. Klopf's [1] insightful observation and proposition on the functioning of the neuron cell and the nervous system in general, and the work done by Professor A. Barto and his associates at the University of Massachusetts in an effort to design and computer-simulate networks and systems of networks operating on the principles proposed by Klopf.

Klopf hypothesized that the neuron is an adaptive heterostat element, operating in such a manner as to maximize the frequency of occurrence of certain inputs deemed desirable and minimize the frequency of occurrence of those undesirable. It achieves this by appropriately modifying its transfer characteristic so as to make it easier to respond to desirable inputs. Thus, the neuron learns to exert some control over its output through an input-output associative process and adaptation so as to enhance those conditions that result in desirable output.

Nature is a supreme teacher and observing how it works has always yielded new design ideas. We believe that it would be desirable to build controllers for physical systems that could operate in this manner. Such controllers would learn to develop a control law without requiring to know the dynamics of the controlled system.

Barto and his associates [2] investigated the feasibility of using goal seeking elements operating in the mode theorized by Klopf as components of intelligent machines. Computer simulation models were developed for goal seeking elements, and it was demonstrated that goal seeking nets could be built out of goal seeking components.

Barto's work is an excellent study on adaptation and learning problems and learning rules, with special emphasis given to Klopf's heterostat. In a system operating in accordance with this reinforcement learning rule, the weighting function at the i-th input,

$W_i(t)$, is enhanced if excitation of the i-th input at t leads to excitation at the i-th input τ seconds later, the enhancing function $e(\tau)$ decaying exponentially with τ. Thus τ is a cause-effect measure, a small τ indicating a strong "link" between the i-th input and the output it produces.

Of the several goal-seeking systems of goal-seeking components developed and studied by Barto, those described as "learning with a critic" were judged to be potentially more applicable to our engineering world.

Systems described as "learning with a teacher" require that the controller "knows the answers to a set of questions", i.e., knows what the response to a set of inputs should be and provides the system with appropriate corrective signals. In most engineering systems, this operation requires more information than is usually available. Learning with a critic, on the other hand, requires only that an observation be made as to whether the output is changing in the right direction. One could reasonably expect that such information should be available in many engineering applications.

The most promising learning network Barto developed that demonstrates problem solving/control capability is the ASE-ACE learning loop [3].

In this system, two elements are used to implement a learning strategy as follows. One element, termed the Associative Search Element (ASE) constructs associations between the input and output by searching under the influence of reinforcement feedback. A second element, the Adaptive Critic Element (ACE) constructs a more informative evaluation function than reinforcement feedback can provide, thus improving the performance of the ASE when operating alone. Both of these neuron-like adaptive elements, which constitute the controller of the learning network, were suggested by the work of Klopf.

The example chosen by Barto on which to implement the ASE-ACE learning net was the adaptive learning problem known as "BOXES", developed by Michie and Chambers. BOXES requires that the system learn to balance a pole which is pivoted on top of a cart, by applying a force ±F on the cart, which is free to move along a straight line path. The reason for choosing BOXES was that it provided a good learning control problem with a solution available, hence the improved performance resulting from the use of ASE-ACE learning net could be concreteiy demonstrated.

The specified goal of this project was:

1. Develop an understanding of Klopf's work related to neuron-like adaptive system behavior.

2. Understand Barto's work on realization/simulation of neuron-like adaptive learning networks.

3. Investigate possible implementation of these nets in physical systems, specifically in synthetic aperture radars.

This report documents the progress made toward that goal.

Section 2 of this report discusses the pole-on-cart control problem, which is used for testing the learning algorithms, modified slightly to conform better to engineering concepts. It also describes the implementation of the ASE-ACE controller at ERIM and the studies on ASE-ACE performance.

Section 4 investigates the use of ASE-ACE in minimizing arbitrary functions and Section 5 indicates how the ASE-ACE controller could be applied in the SAR autofocus problem.

Finally, Section 7 describes the possible application of a learning net, like the ASE-ACE, to a robotics problem.

2
IMPLEMENTATION OF THE ASE-ACE NET

2.1 THE POLE-ON-CART CONTROL SYSTEM

Since the pole-on-cart system is used to test the learning algorithms developed, it is useful to have the system and control approach suggested by Michie and Chambers described in some detail.

Consider a system consisting of a cart with a pole pivoted on top of it. The cart is constrained to move along a straight line path, say the x-axis, on the x-y plane. The pole is so pivoted that it can move on a plane perpendicular to the x-y plane, through the x-axis. Let the allowable linear motion of the cart be the interval $(-X, X)$ while the pole's allowable angular displacement is $(-\theta, \theta)$ (see Figure 2.1).

A motor in the cart can move it along the x-axis by applying a constant force F in either direction. The control goal is to keep the pole inside the allowable θ limits while the cart stays within the x-bounds. To do this, sensors measure the cart's linear position and velocity and the pole's angular position and velocity at discrete intervals $t = nT$ and a force F or -F is then applied at these intervals. This could be the result of a +V or -V voltage applied to the motor. If the system reaches the extreme positions $\pm X$ or $\pm\theta$, the experiment ends. The magnitude of X, θ, and F are not important here and could be so chosen as to correspond to appropriate values.

The state of the system at $t = nT$ is described by the vector $\underline{s}(nT) = (x, \dot{x}, \theta, \dot{\theta})$, where each state-variable is evaluated at $t = nT$. Since S is measured at discrete intervals, it is convenient to discetize the state space by arbitrarily allowing a finite number of levels in each state variable. If N_1, N_2, N_3, N_4 are the allowed levels in the variables x, v, θ, and w, respectively, the state space will have exactly $(N_1N_2N_3N_4)$ possible states.

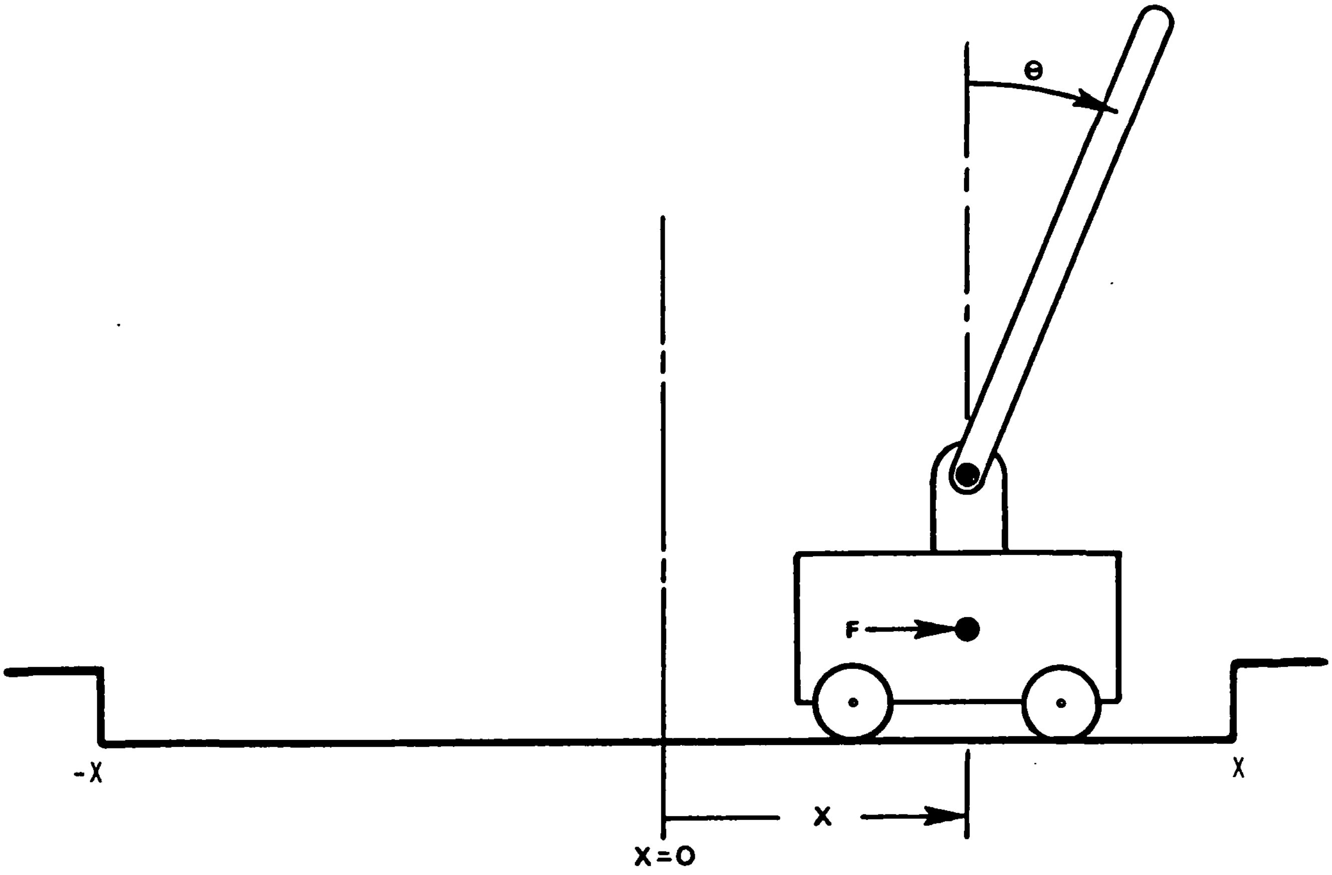


FIGURE 2.1. THE CART-POLE SYSTEM THAT IS TO BE CONTROLLED.

There are two basic assumptions that have to be kept in mind when designing a controller for this system. First, knowledge of system dynamics, i.e., mathematical model or transfer function for the system is not available. Second, a learning system improves its performance through evaluation of its experience. Thus, learning requires long-term memory to allow comparisons of results of actions taken in the past in response to existing conditions. On the basis of this comparison appropriate actions are taken. For such a learning system, then, the controller is the learning network.

In the "BOXES" example, a controller for this system was designed as follows. We choose a controller with at least as many memory cells (boxes) as the number of system states. Each cell is addressed by a state and in it we store the action to be taken by the system, ±F, when the sensed system state corresponds to the cell's address.

Initially, the system does not know what is the correct instruction to give. So the system is initialized by storing values ±F at random in the cells. When control action starts, the system sensor read the state of the system at regular intervals $t = nT$. The system then goes to the memory cell addressed by that state and reads what action is to be taken. At that instant, a clock in the cell starts counting. This process continues until the system fails, i.e., a state variable exceeds the extreme values ±X or $\pm\theta$ and control action stops. The clock readings in each cell are the time until failure (TUF) from the moment the cell was entered. This TUF is now stored in the cell and the instructions in all entered memory cells during the first control action are reversed, +F to -F and vise versa.

The process is repeated but now at the end of the second control action there are two TUF readings for those cells that were entered the second time. We leave in the cell as control instruction whichever instruction leads to longer TUF, together with the value of that TUF.

This control strategy leads to memory settings that favor maximum TUF for the system and after several control actions the system "learns" what to do to stay inside the prescribed bounds.

A system operating with this control law has been computer-simulated by Barto and shown to work well [3]. After about 100 control actions the system can take on the average 4,000 steps before failure occurs.

The described control strategy is not optimal. The system is not learning fast. Actually, since learning takes place when the system fails, the learning process slows down as the system learns. Barto corrected this through the introduction of the ASE-ACE adaptive-learning network.

2.2 THE ASE-ACE ADAPTIVE CONTROLLER

Figure 2.2 shows a system with transfer function G controlled by the ASE-ACE learning net. The state vector $\underline{s}$ of the system is sampled at intervals T sec. and is fed into a decoder which is used to discretize the state space of $\underline{s}$ into a finite number of states and convert $\underline{s}$ into a binary vector $\underline{X}$, whose components are all zero except the one corresponding to the state of the system at the sampling instant t = nT. The dimension of $\underline{X}$ is equal to the number chosen for the discrete states of the space of $\underline{s}$.

The vector $\underline{X}$ is fed into the ASE-ACE. At the Adaptive Critic Element, its adaptive weighting vector $\underline{v}$, the input vector $\underline{X}$ and the external reinforcement function r(t), are used to generate the internal reinforcement function $\hat{r}(t)$ that inputs the ASE, in accordance with the rule:

$$\hat{r}(t) = r(t) + \gamma p(t) - p(t - 1)$$

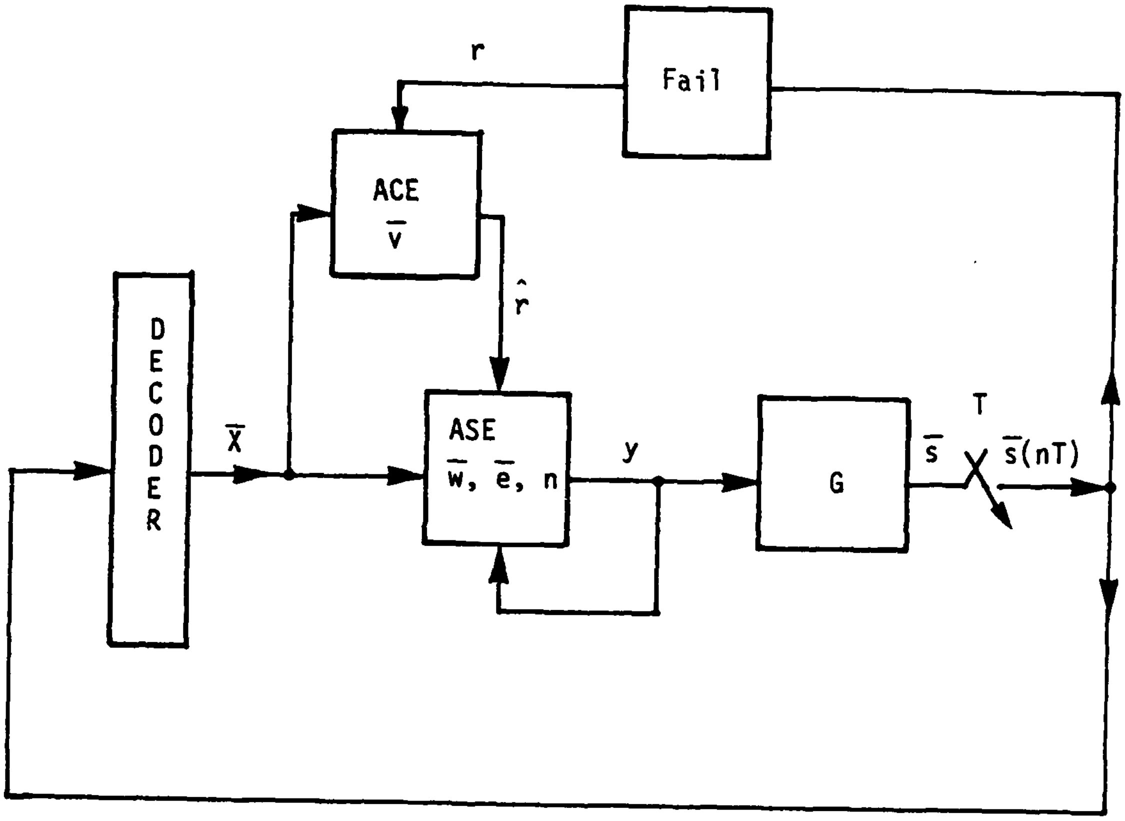


FIGURE 2.2. THE ASE-ACE CONTROLLER.

where

$$p(t) = \sum v_i x_i$$

γ is a non-negative constant less than or equal to one, and the weighting vector $\underline{v}$ updates in accordance with

$$v_i(t + 1) = v_i(t) + \delta\hat{r}(t)\hat{x}_i(t)$$

where $\hat{x}_i(t)$ is the value of a trace of the input variable x_i at t, evaluated from:

$$\hat{x}_i(t + 1) = \beta\hat{x}_i(t) + (1 - \beta)x_i(t)$$

and β and δ are positive constants.

At the Associative Search Element, the input vector $\underline{X}$ generates the output y:

$$y(t) = \pm 1$$

depending on whether $[\sum w_i x_i + n(t)]$ is nonnegative or negative, respectively, n(t) is additive system noise and the weighting vector $\underline{w}$ updates in accordance with the rule:

$$w_i(t + 1) = w_i(t) + \alpha\hat{r}(t)e_i(t)$$

The function $e_i(t)$ is the eligibility at t of path i, adapting in accordance with the rule:

$$e_i(t + 1) = \beta e_i(t) + (1 - \beta)[y(t)x_i(t)]$$

and $\alpha > 0$; $0 \leq \beta < 1$. Figures 2.3 and 2.4 show in block diagram form the implementation of the ASE-ACE algorithms at ERIM.

The way ASE-ACE exercise control over a given system G is as follows. Let us assume that we wish to maintain the values of the state variables s_j and s_k of the system within certain bounds.

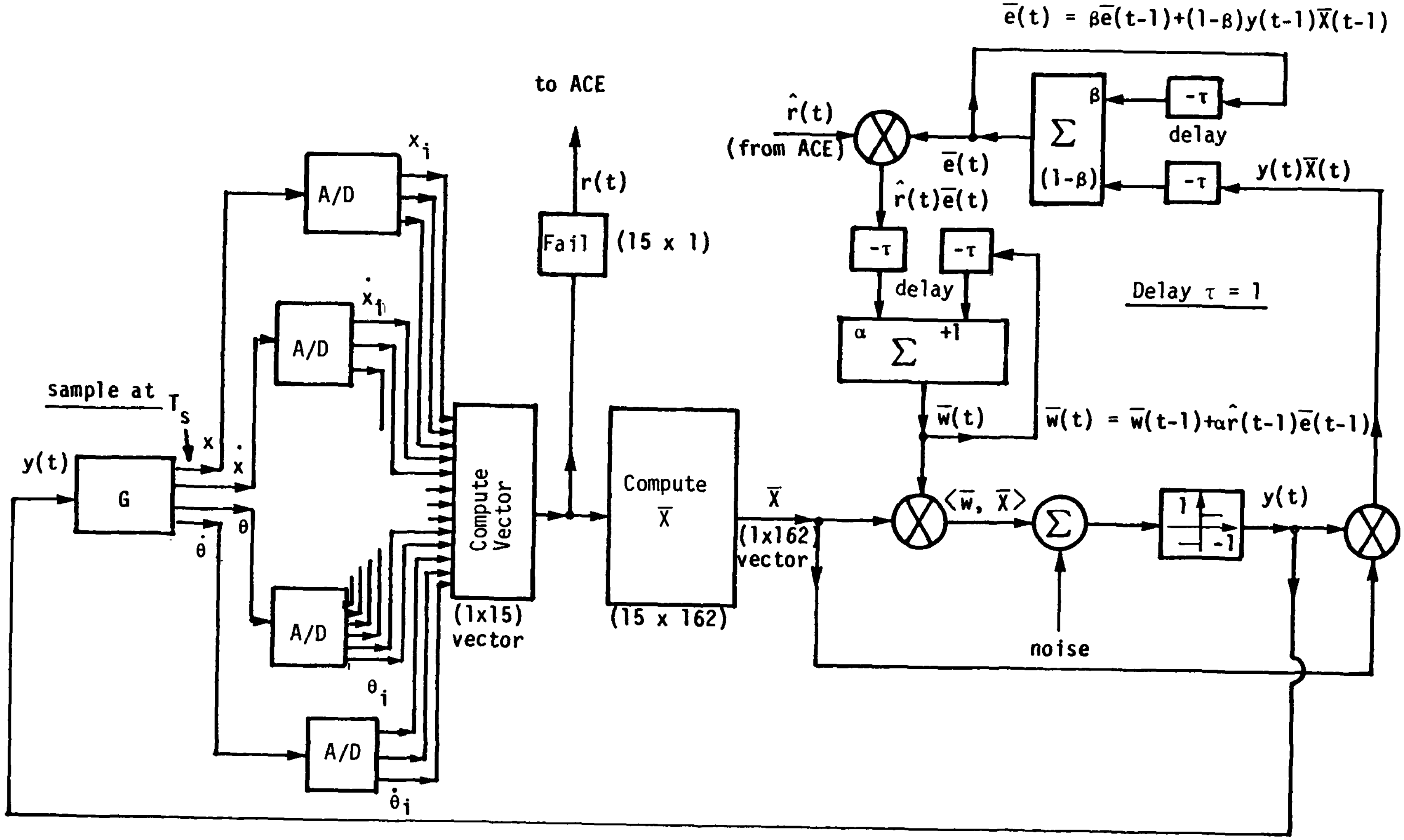


FIGURE 2.3. IMPLEMENTATION OF ASE.

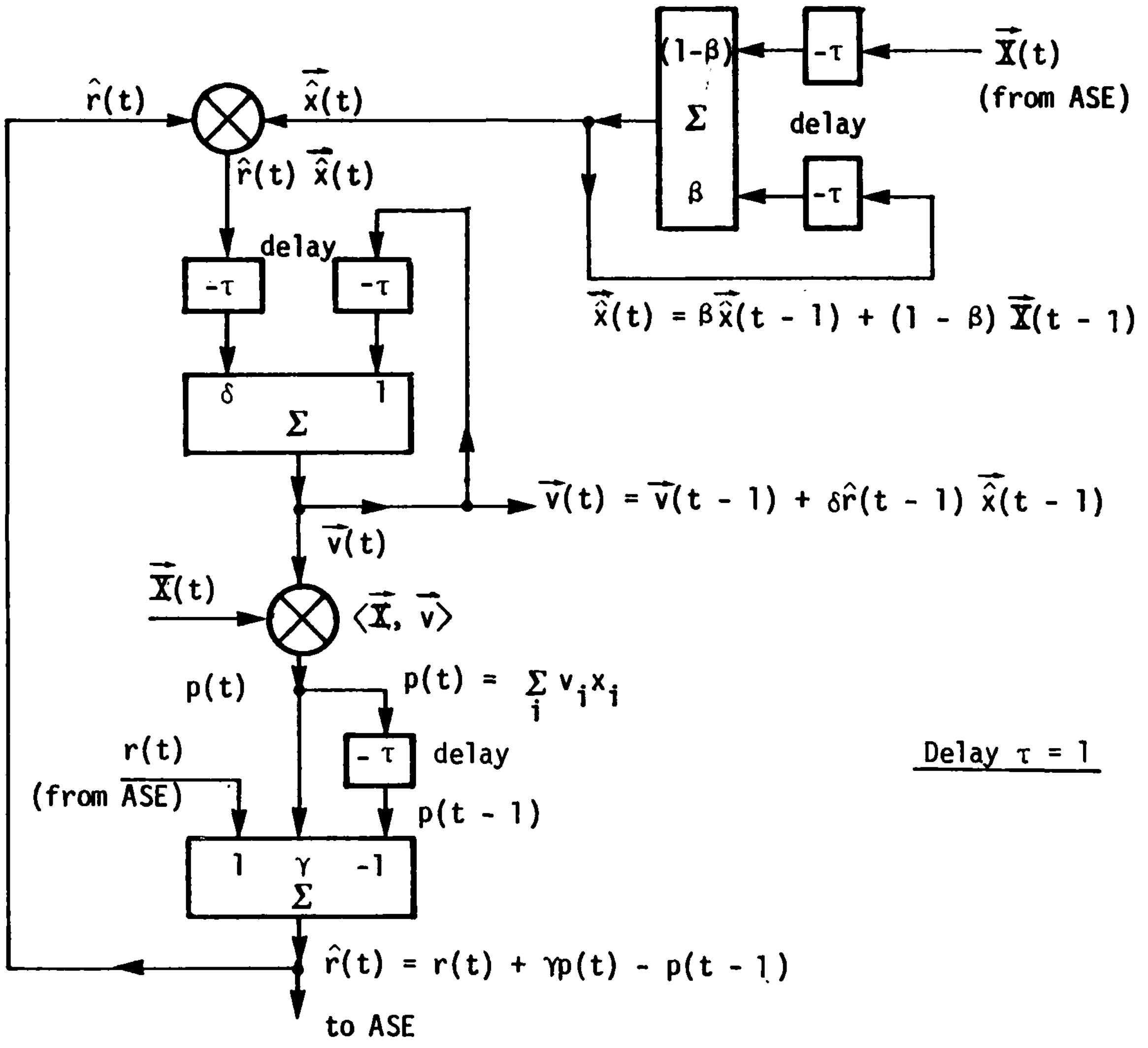


FIGURE 2.4. IMPLEMENTATION OF ACE.

We use the external reinforecement variable r(t) to penalize the system when either s_j or s_k take values outside the desired range. When this happens, we will say that the system has failed and r(t) is set equal to -1. Otherwise, r(t) = 0.

With zero initial values for the system state variables and the ASE-ACE variables $\underline{w}$, $\underline{v}$, $\underline{e}$, $\underline{x}$, the system is activated and goes through a sequence of admissible states, until it finally fails, either in s_j or s_k. At that time, the system state variables and $\hat{x}$ are reset to zero, but $\underline{w}$ and $\underline{v}$ are left untouched. Thus, when the next trial for the system starts, the initial values of $\underline{w}$ and $\underline{v}$ are the final values from the previous trial. Hence, the experience, or learning, of the system at time t is stored in the values $w_i(t)$ and $v_i(t)$. After a few trials, the system learns how to operate without failure, i.e., learns how to operate while maintaining the state variables within the desired bounds.

The previously described ASE-ACE system was independently simulated at ERIM, though the University of Massachusetts program was given to us.

When running the ERIM pole-on-cart simulation, a different set of pole-cart parameters than those used by the University of Massachusetts was selected. This was because an error was found in the University of Massachusetts computer program, which when corrected made it impossible for the ASE-ACE to learn, since the pole hit the boundaries in one sampling interval. This problem was corrected by using a pole length of 10 meters instead of 0.25 meters. All other parameters were the same as used by the University of Massachusetts. The problem could also have been corrected by using a smaller sampling interval so the pole moved less between samples, but the former approach was considered preferable, since it allowed greater flexibility in the choice of sampling period.

Our results substantiate that Barto's ASE-ACE controller after a few trials can indeed learn to keep the pole balanced. Figure 2.5 shows a typical system learning curve. It is a plot of "no. of steps to failure for trial k" versus "trial number." If the system learns, the curve should have a positive slope, the slope being a measure of the rate of learning of the system. In this example, the run was terminated during the 28th trial, after the system exceeded ten thousand steps without failure. When this happens, the computer is instructed to cut off. The system has learned. Several runs were made with different initial seed values in the noise generator program, but with the same noise standard deviation. The resulting curves were very similar, demonstrating a consistent system behavior. The average of these runs is plotted in Figure 2.6.

It is useful at this time to examine briefly the concept of learning. Specifically, how should one measure the performance of a learning system? Figure 2.7 shows the learning curves of two hypothetical systems. System no. 2 is initially learning faster than system no. 1, but after several trials, no. 1 performs better.

Had we decided to declare that a system has "learned" when it exceeded S steps without failure, it would appear that system no. 2 is preferable to no. 1, because it reached and exceeded S steps in fewer trials. Yet, assuming that the sampling period is the same for both systems, it took longer for system no. 2 to reach that level of learning, because each trial it went through lasted longer. The time to learn for each system is proportional to the area under its learning curve and is equal to:

$$T_L = \sum_{j=1}^{N-1} S_{Fj} T_S$$

where S_{FJ} = the number of steps to failure for trial j,

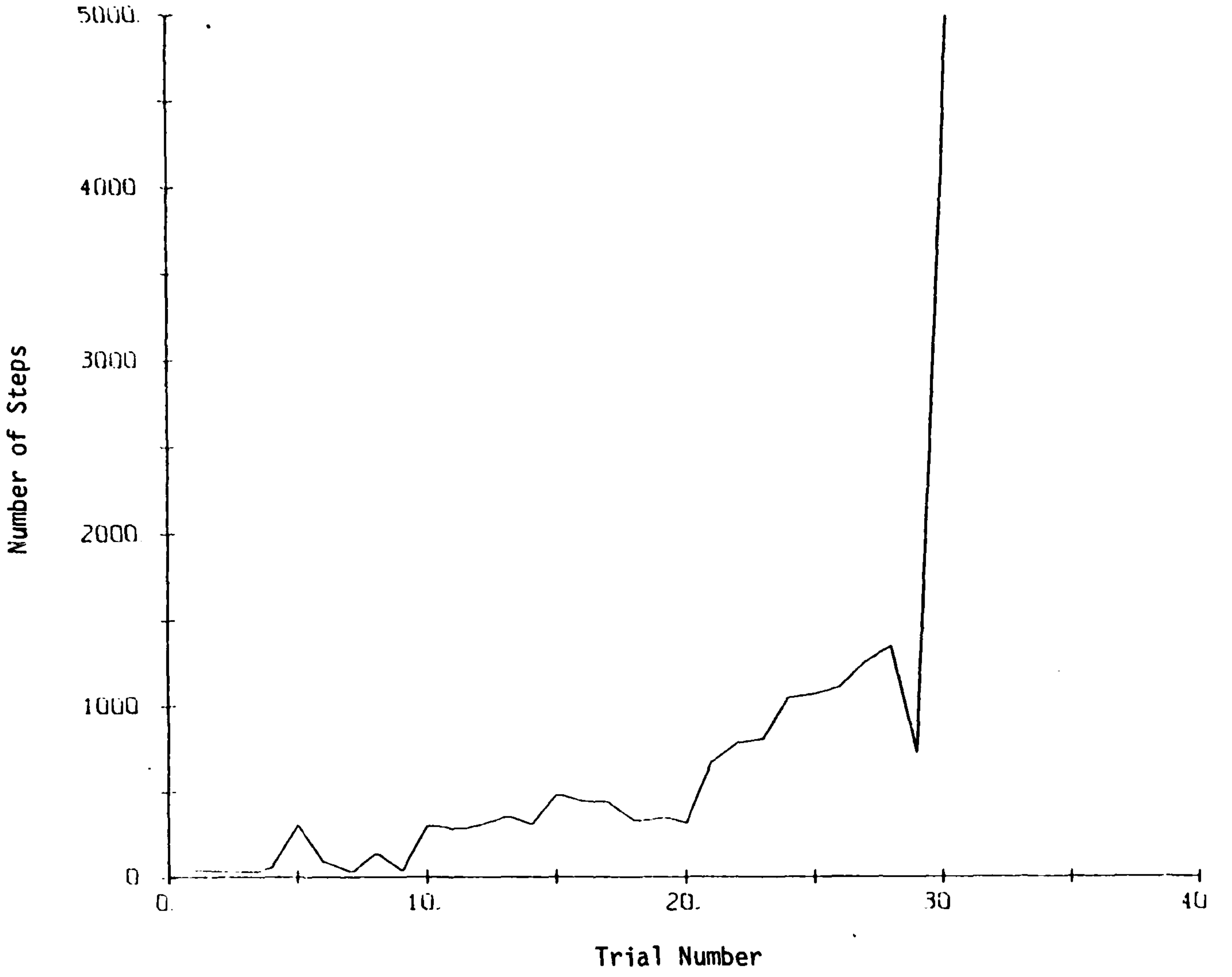


FIGURE 2.5. TYPICAL LEARNING CURVE OF THE POLE-ON-CART SYSTEM WITH ASE-ACE CONTROLLER.

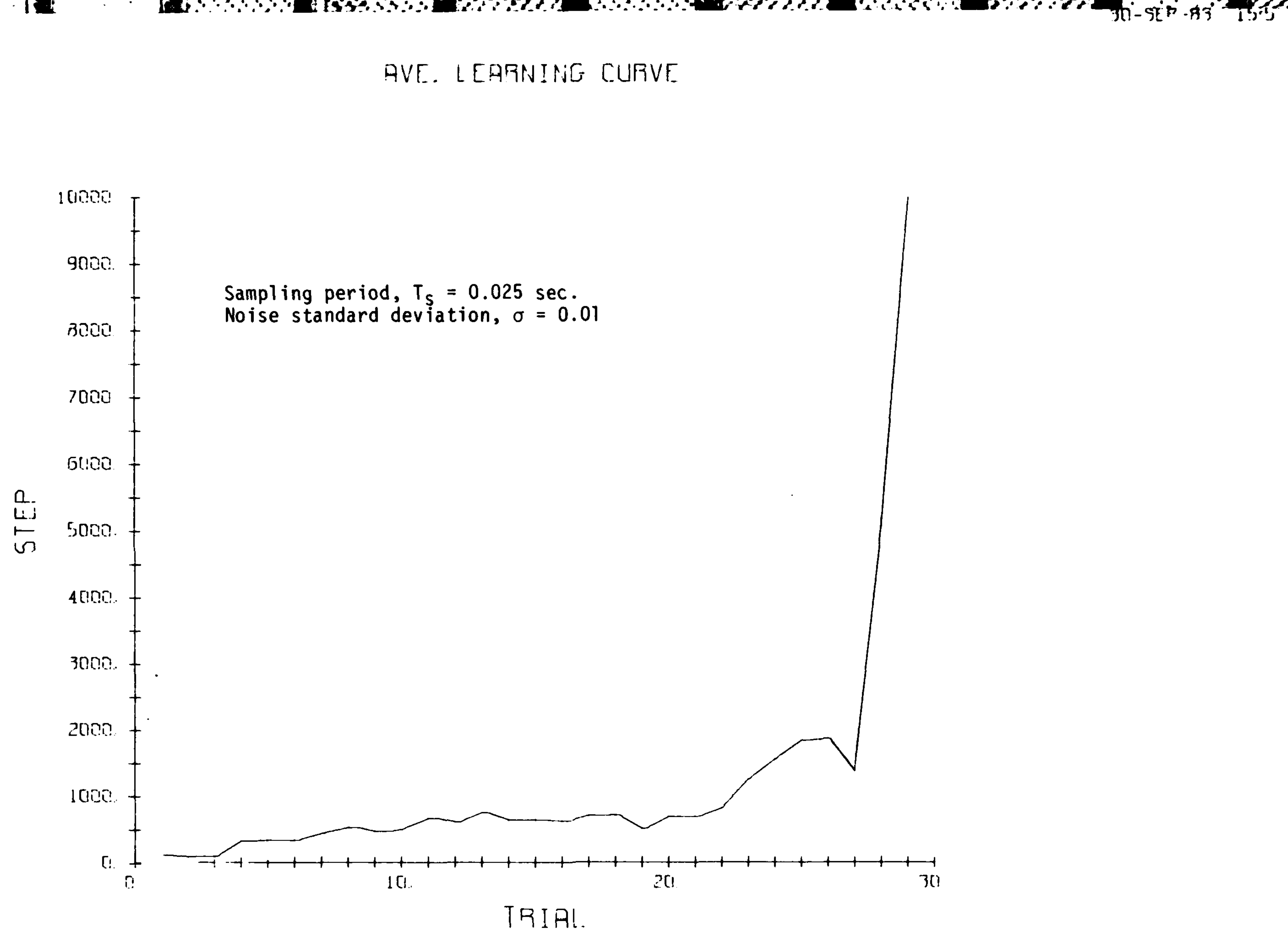


FIGURE 2.6. THE AVERAGE OF FOUR LEARNING CURVES OF THE POLE-ON-CART SYSTEM WITH ASE-ACE CONTROLLER.

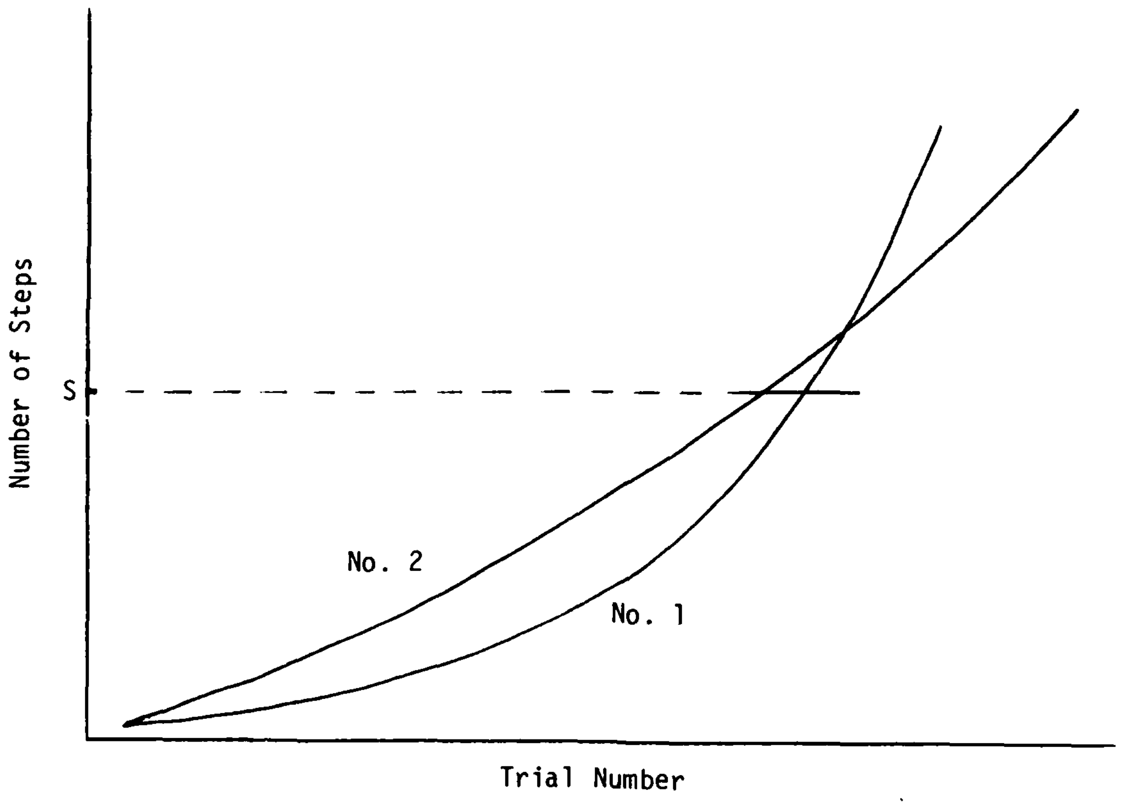


FIGURE 2.7. LEARNING CURVES FOR TWO HYPOTHETICAL SYSTEMS.

T_s = the sampling period, and

N = the trial number at which the system exceeded S steps without failure.

What makes a system better, then, depends on the appiication. It appears, however, that the time-to-learn concept is a more meaningful measure of learning performance.

2.3 PERFORMANCE STUDIES

Understanding how the ASE-ACE controller adapts and learns is vital, if we are to apply it successfully. The following studies therefore were carried out:

a. Observation of the ASE-ACE system variables throughout the learning process.

b. Effect of sampling period on system performance.

c. Effect of noise level on performance.

d. Effect of α, β, γ, δ parameter values on performance; optimal values.

e. Effect of state-space structure on performance.

2.3.1 ASE-ACE SYSTEM VARIABLES BEHAVIOR DURING LEARNING

Close observation of the variation of the system variables, $\underline{w}$, $\underline{v}$, $\underline{e}$, $\underline{x}$ and the input vector $\underline{X}$ through a learning cycle is most instructive.

After several runs were made, it was observed that system states no. 4 and 10 were the most frequented states. The system "walked through" several states, but was coming back to no. 4 and no. 10 regularly. The system does not need to go through all states to learn. With every new trial, it visits a few new states, but most of the time goes through states that were visited previously. By the time the system took 10,000 steps it had gone through seventy-eight different states, out of 162 total.

Plots were made of the input vector $\underline{X}(t)$ and variables $\underline{w}(t)$, $\underline{e}(t)$, $\underline{v}(t)$, $\hat{\underline{x}}(t)$ for states no. 4 and no. 10 throughout ten consecutive trials, for a total of 299 steps (see Figures 2.8 to 2.17). The effect of punishment at each failure on w_i and v_i is clearly evident, as is the effect of the eligibility function e_i on w_i and of the trace x_i function on v_i. Phase plots, i.e., $\dot{x}$ vs. x and $\dot{\theta}$ vs. θ were also plotted for several trials (Figures 2.18, 2.19). These plots illustrate the system behavior during the trial. It is seen that when the sytem has learned, it goes through a cycle of states over and over again, as should be expected.

2.3.2 EFFECT OF SAMPLING PERIOD

The sampling period was certainly expected to have a significant effect on the system's learning behavior for two reasons. First, there must be a minimum sampling rate, which is dictated by the system bandwidth. Second, the nature of the learning system is such that at each sampling instant, a decision is made as to whether a force +F or -F should be applied, and also the system variables are updated. Thus, shorter sampling period implies tighter control and possibly improved learning.

Runs were made with sampling periods, T_S, of 0.025 sec., 0.05 sec., 0.075 sec., 0.1 sec., 0.15 sec. and 0.2 sec. and the system was allowed to go through fifty trials or 10,000 steps. The results are shown in Figures 2.20 to 2.25. The system learning behavior was approximately the same for T_S = 0.025 sec. and T_S = 0.05 sec., in both cases the system exceeding 10,000 steps by trial no. 30. The learning rate decreased as T_S increased, the system finally failing to learn when T_S = 0.2 sec. With T_S = 0.15 sec., the system showed signs of slow learning by the end of the fiftieth trial. Longer runs were made and it was verified that indeed the system is slowly learning (Figure 2.24). The results are summarized in Figure 2.26, where the maximum number of steps in fifty trials is plotted against the sampling period.

2.3.3 EFFECT OF NOISE LEVEL

Additive noise is introduced in the system at the ASE and affects the output, y, when the path weighting values are small. This is certainly true the first time a path is entered. Later, as the eligibility function gets into the picture w_i's assume larger values and the effect of noise diminishes.

Several runs were made with noise standard deviation values of $\sigma = 0.01$, $\sigma = 0.1$, and $\sigma = 1.0$. Figures 2.27, 2.28, and 2.29 show these results, respectively, and it is seen that significant increase in the noise level does inhibit the learning process, though the learning curve for $\sigma = 1$ has a positive slope.

Noise has been considered by Barto, as possibly beneficial to the learning process of the system, because it gets the system to more states quicker, and it was speculated that the sooner a system visits several states the faster it will establish the proper path values. On the other hand, however, once a system has learned it should operate by going continuously through a small number of states in a cyclic fashion. If the weighting path values are small, noise may tend to bounce the system out of the cycle, hence make it less stable and slower in learning.

2.3.4 EFFECT OF α, β, γ, δ PARAMETER VALUES

Use of α, β, γ, δ parameter values chosen by Barto, et al., in their runs were based on logical arguments as to what kind of behavior seemed desirable for the eligibility function, e, trace function, $\hat{x}$, and internal reinforcement function, $\hat{r}$. It was felt, however, that these values had to be tested and verified experimentally. Accordingly, system performance was evaluated over a range of α, β, γ, δ values to determine those values yielding optimal performance, i.e., time to learn.

Several runs were made over a wide range of values for each parameter for two different sampling periods, $T_S = 0.025$ sec. and $T_S = 0.1$ sec. The results are shown in Figures 2.30 to 2.33. These graphs clearly show that the values selected by Barto gave optimal performance when $T_S = 0.025$ sec. For $T_S = 0.1$ sec., however, best performance was obtained for $\alpha = 1{,}000$, $\beta = 0.85$, $\gamma = 0.85$, and $\delta = 0.15$.

2.3.5 EFFECT OF STATE SPACE STRUCTURE

An important question that must be answered is the effect of system state space structure on the learning behavior of the system. Barto has divided in the pole-on-cart example the state space into 162 "boxes", as was done originally by *Michie and Chambers*. This division is arbitrary and was suggested by the need to keep the number of states small, hence the computation time short.

If it were true that a system in order to learn needs to visit a large number of states, then the greater the number of states into which the space is divided the longer it would take for the system to learn. Furthermore, the finer the division the grid is cut into, the finer the control exercised can be. At the same time when the grid gets too fine the system may go through several states between steps, thus without initializing them. This would negate any possible advantages a finer grid could provide. Thus there can be a relationship between sampling period and grid size, if advantage of a fine grid is to be made.

Testing of the effects of grid size have not been completed, but preliminary results seem to substantiate the relationship between grid size and sampling period.

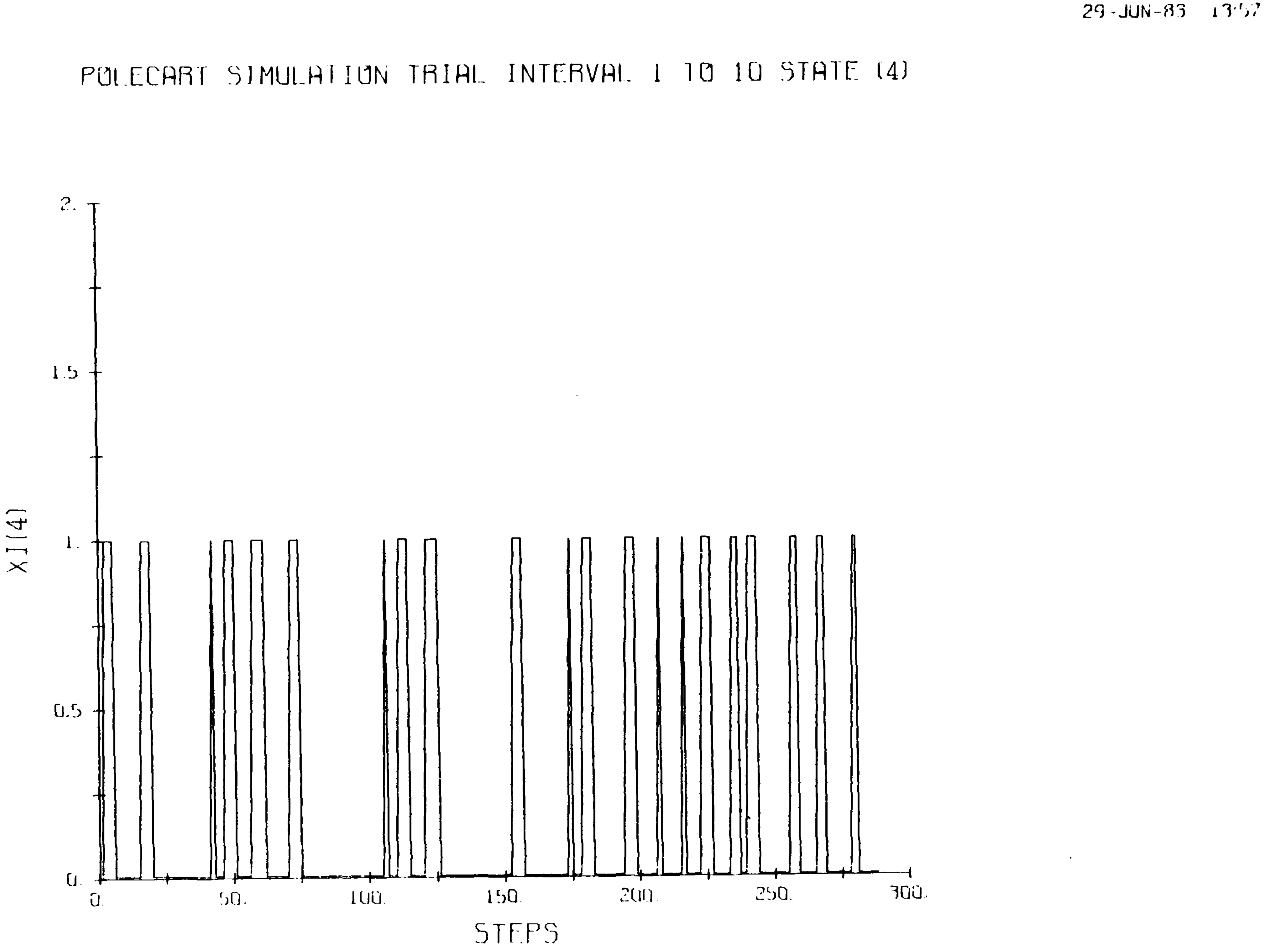


FIGURE 2.8. VARIATION OF INPUT VECTOR X FOR PATH NO. 4 OF THE ASE-ACE OVER TEN TRIALS.

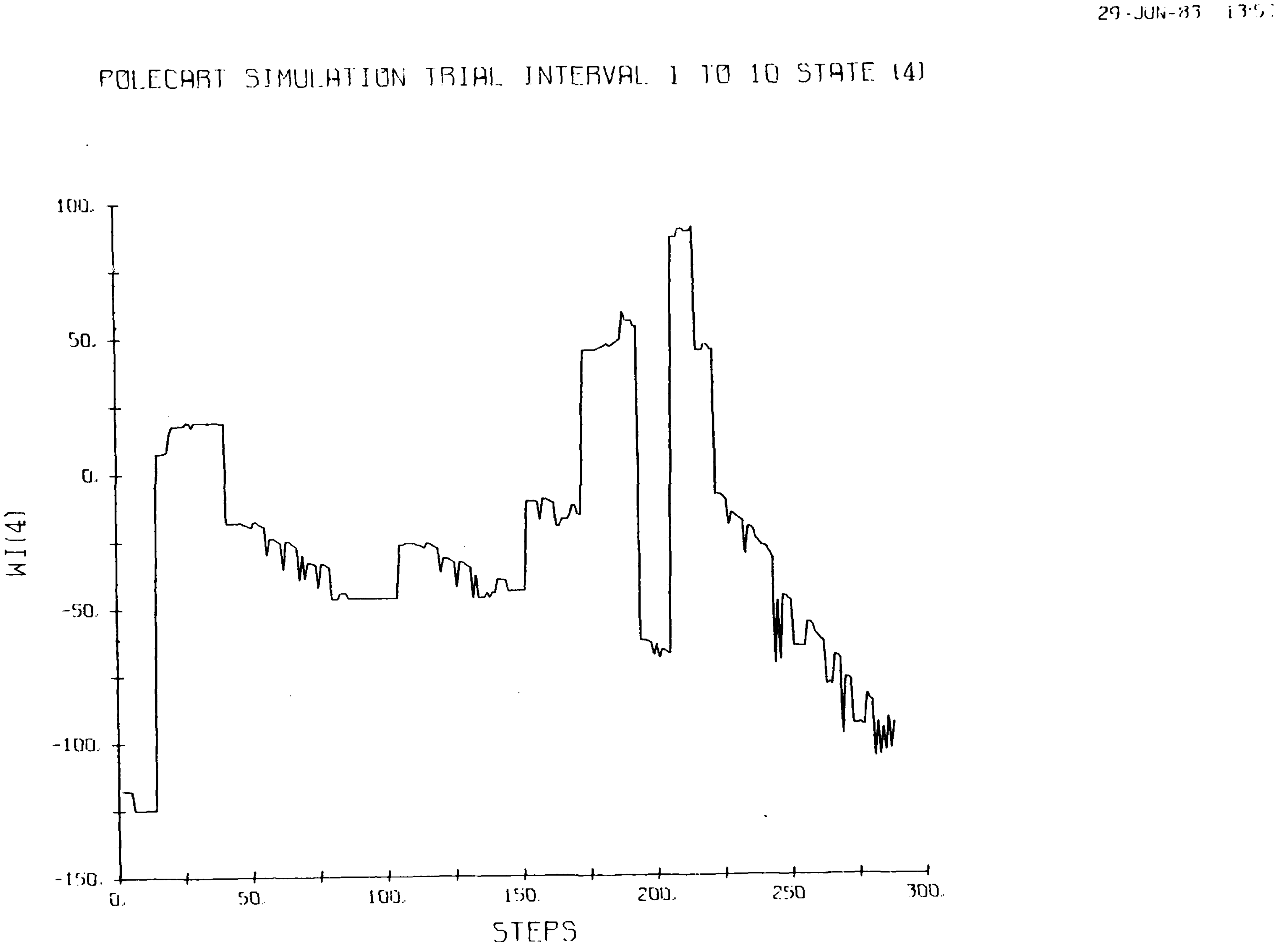


FIGURE 2.9. VARIATION OF W4, THE WEIGHT VALUE IN PATH NO. 4 OF ASE OVER TEN TRIALS.

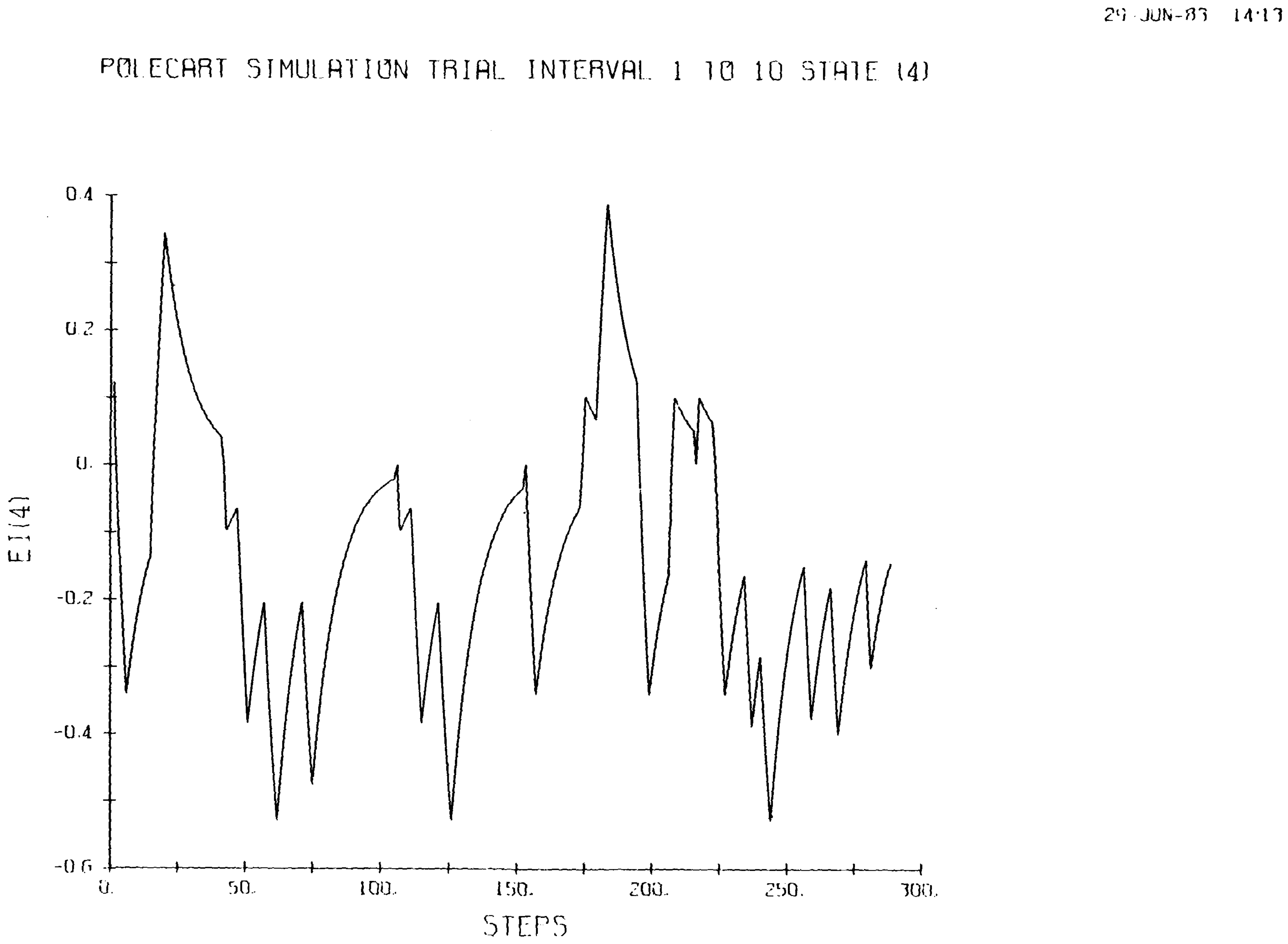


FIGURE 2.10. VARIATION OF E_4, THE ELIGIBILITY FUNCTION FOR PATH NO. 4 OF THE ASE, OVER TEN TRIALS.

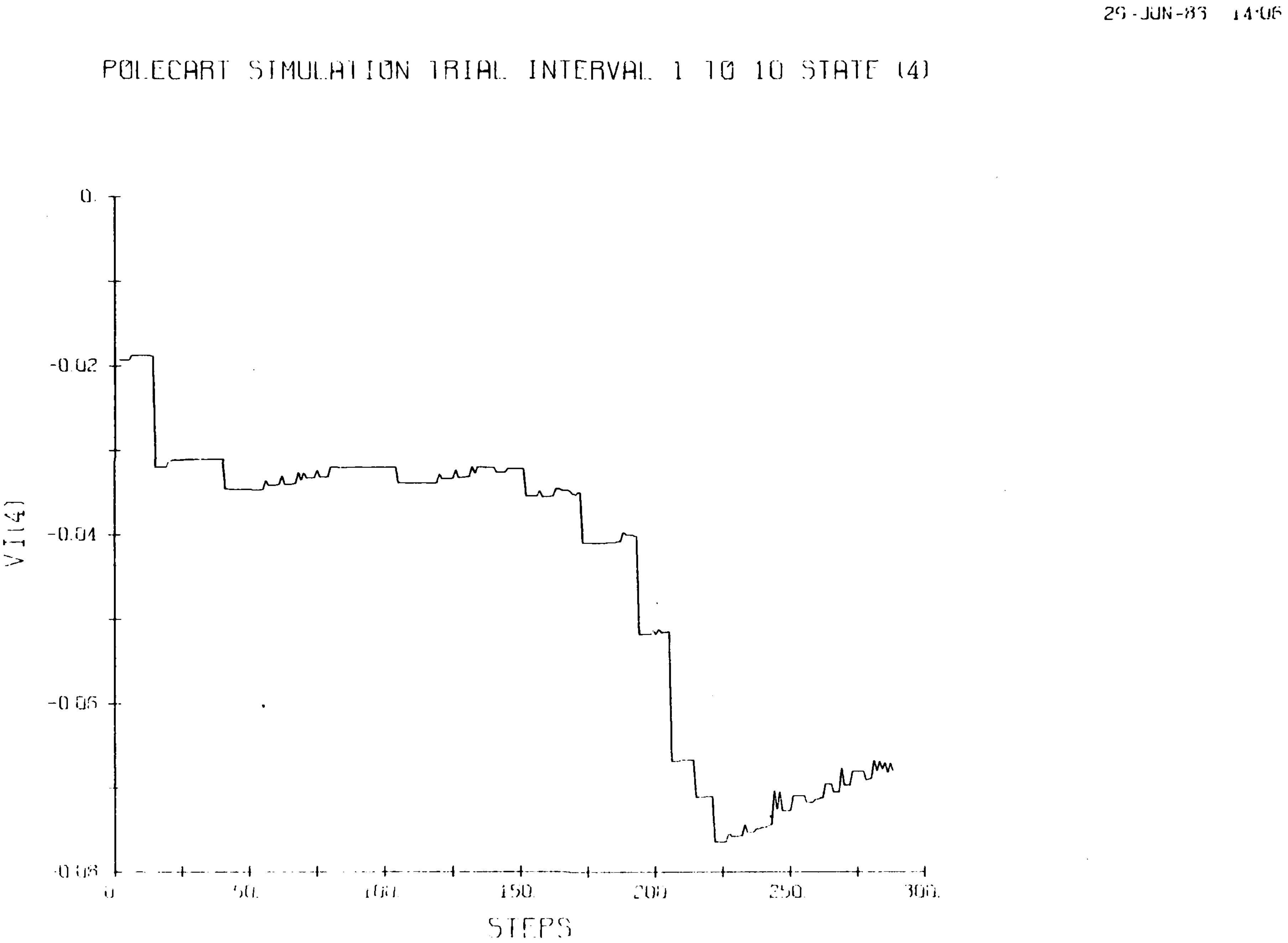


FIGURE 2.11. VARIATION OF V_4, THE WEIGHT VALUE FOR PATH NO. 4 OF THE ACE, OVER TEN TRIALS.

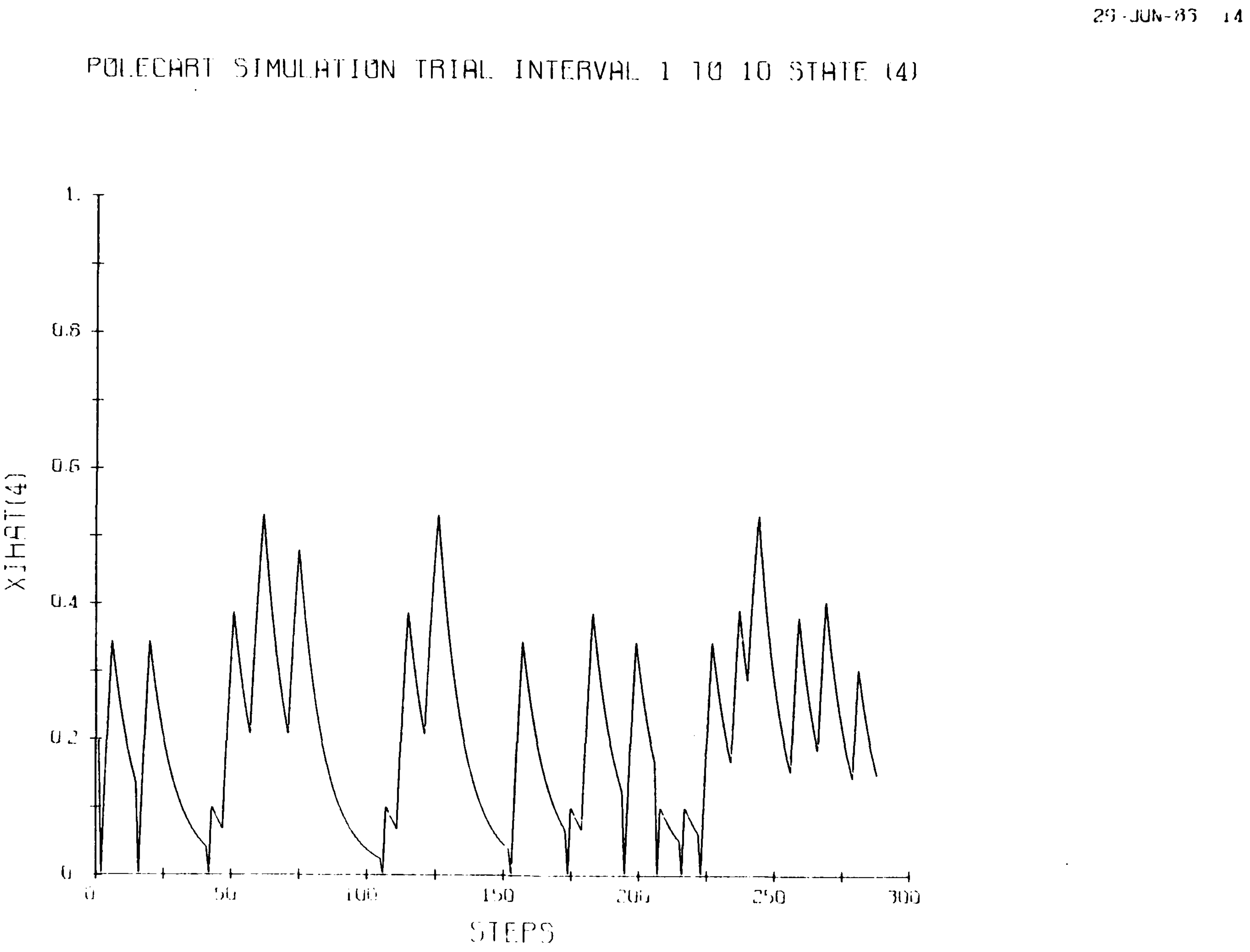


FIGURE 2.12. VARIATION OF $\hat{X}_4$, THE TRACE FUNCTION OF PATH NO. 4 OF THE ASE, OVER TEN TRIALS.

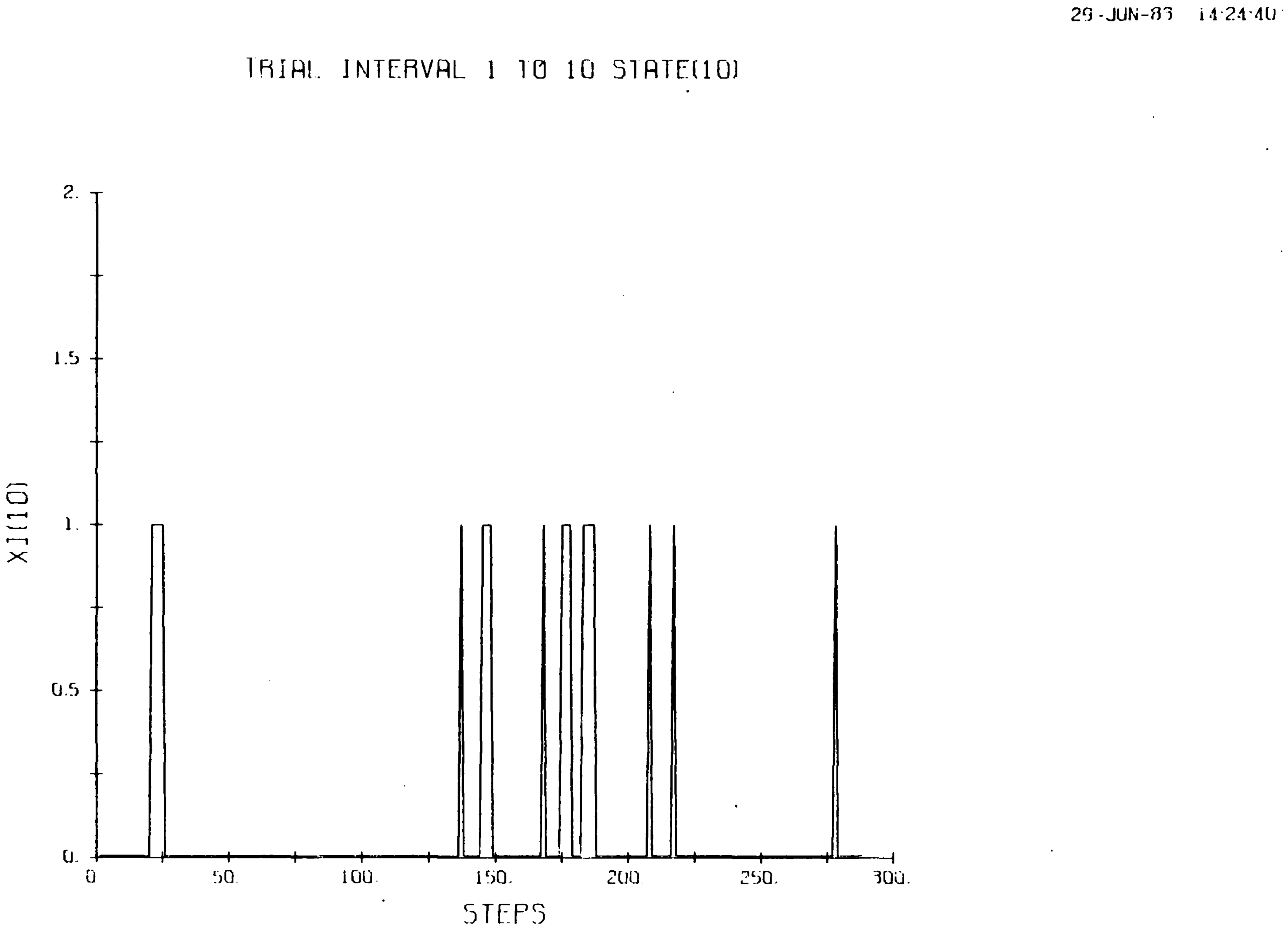


FIGURE 2.13. VARIATION OF THE INPUT VECTOR X FOR PATH NO. 10 OF THE ASE-ACE, OVER TEN TRIALS.

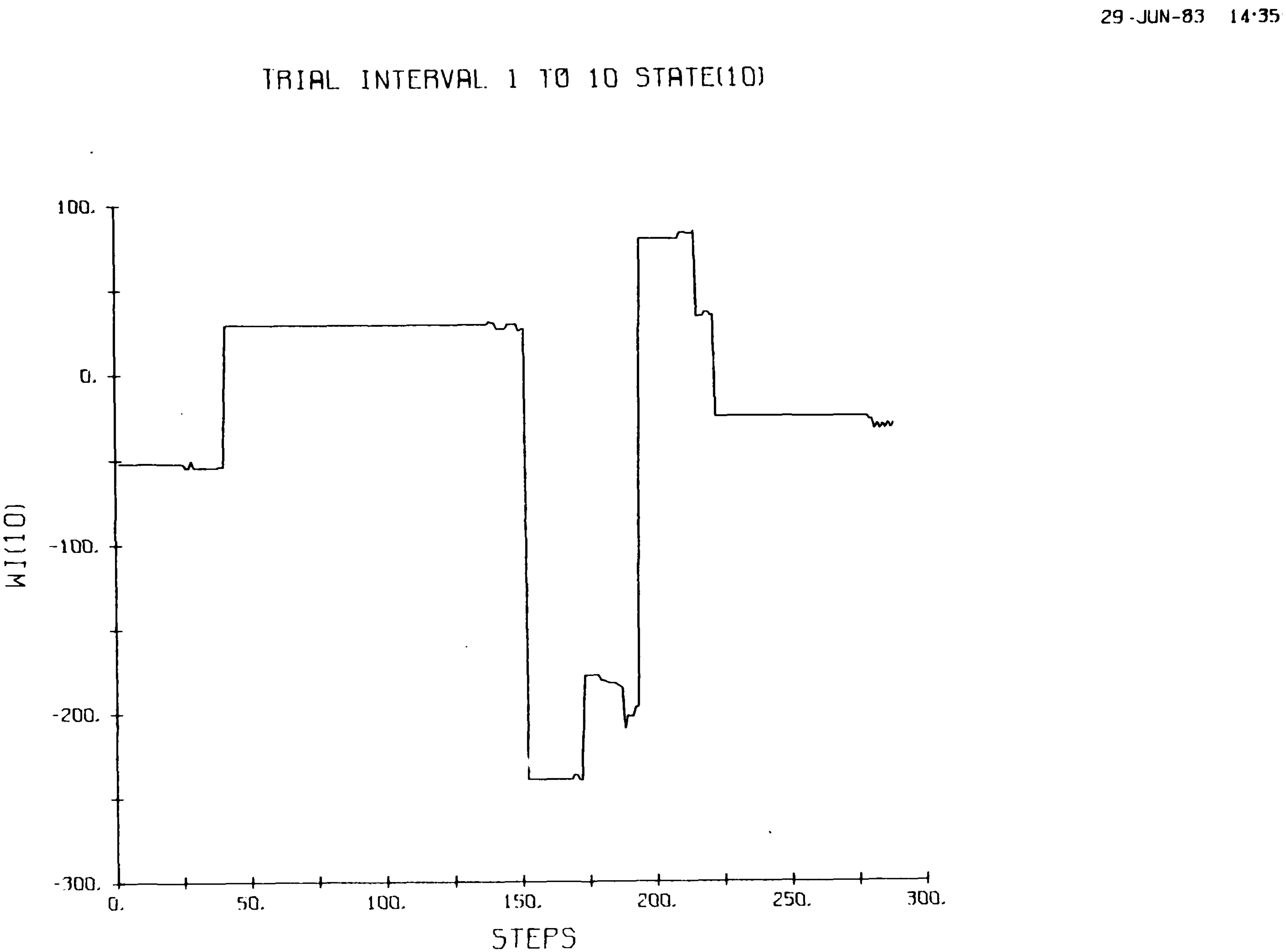


FIGURE 2.14. VARIATION OF W_{10}, THE WEIGHT VALUE IN PATH NO. 10 OF THE ASE, OVER TEN TRIALS.

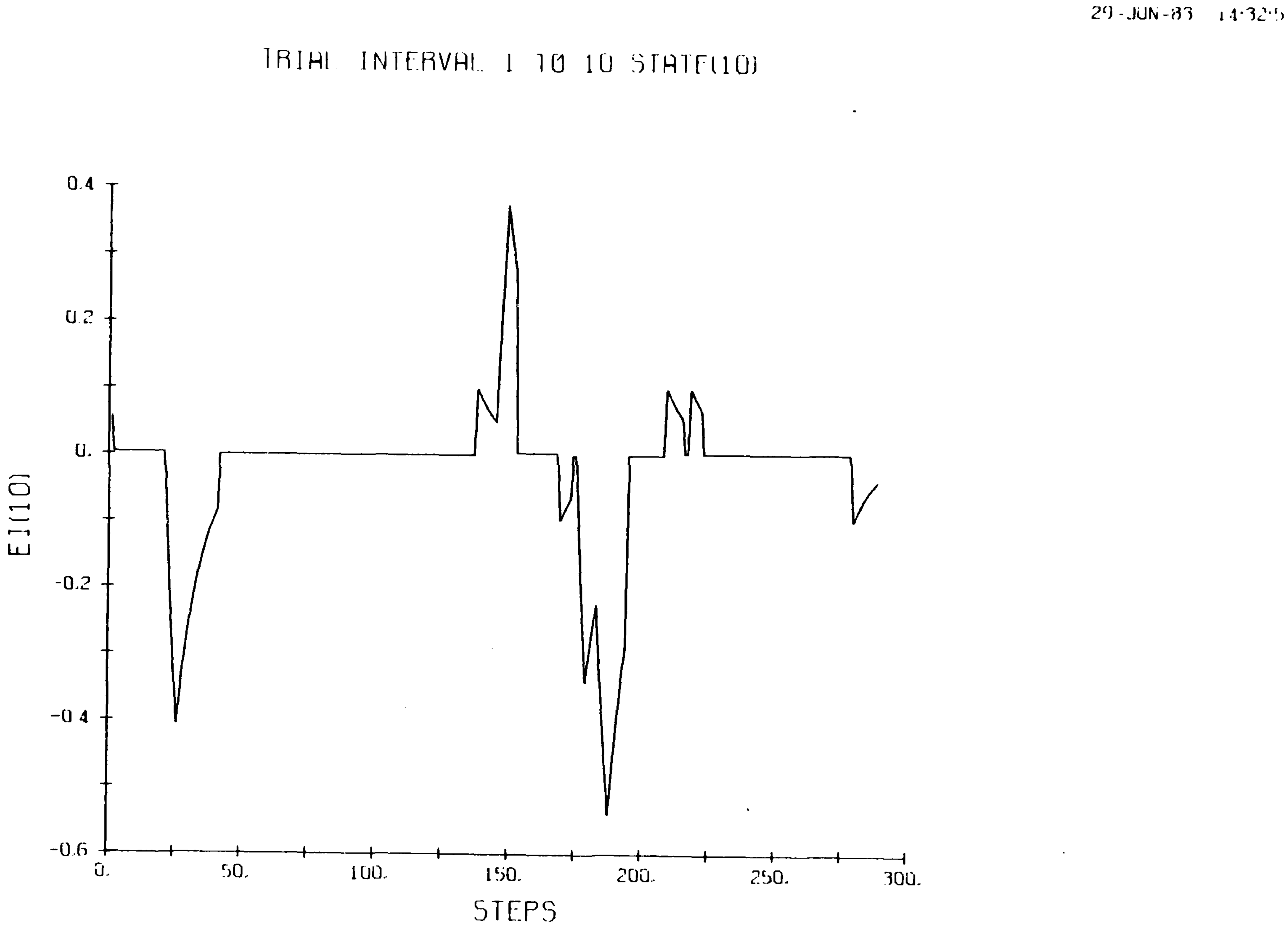


FIGURE 2.15. VARIATION OF E_{10}, THE ELIGIBILITY FUNCTION OF PATH NO. 10 OF THE ASE, OVER TEN TRIALS.

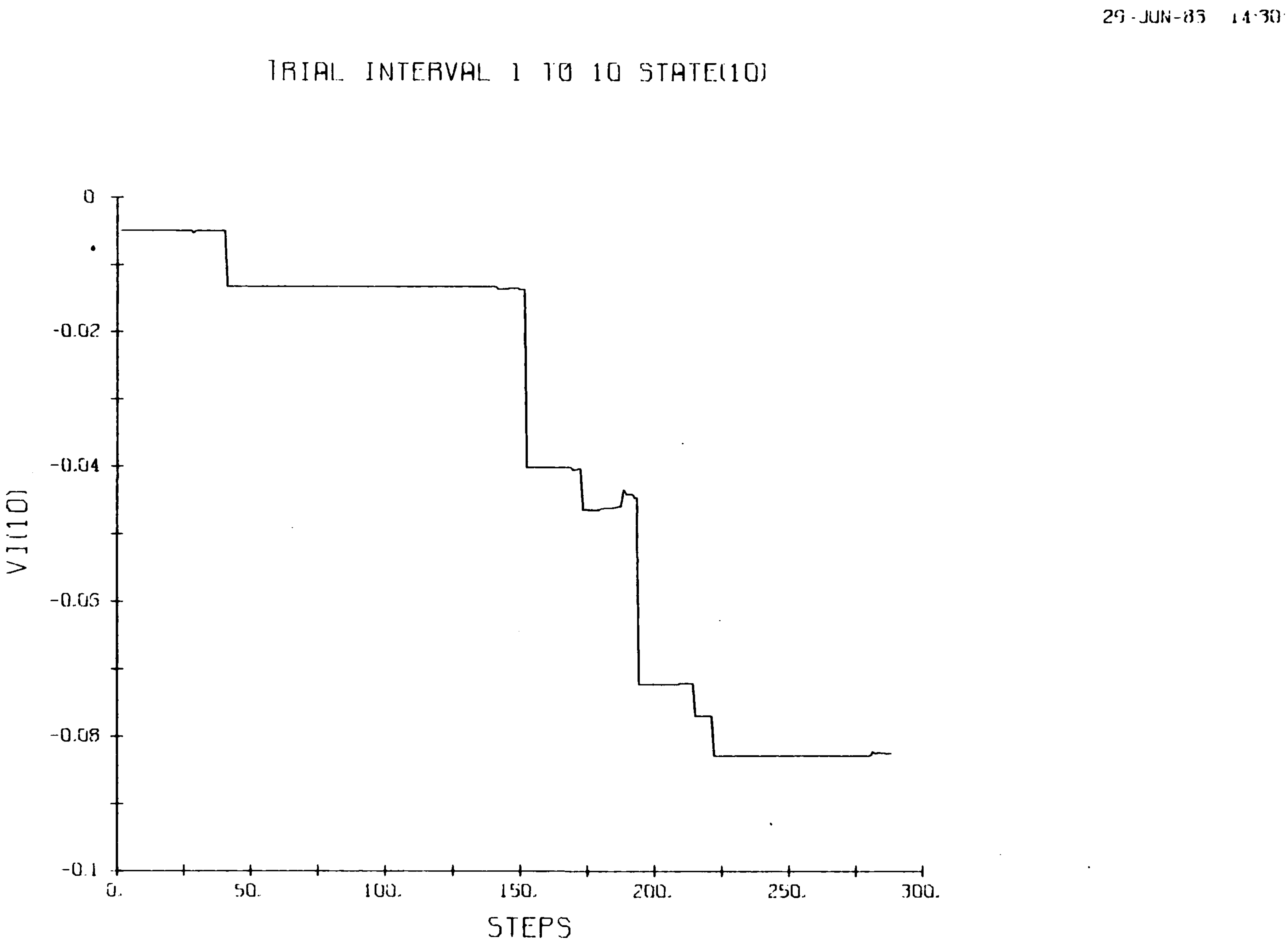


FIGURE 2.16. VARIATION OF V_{10}, THE WEIGHT VALUE FOR PATH NO. 10 OF THE ACE, OVER TEN TRIALS.

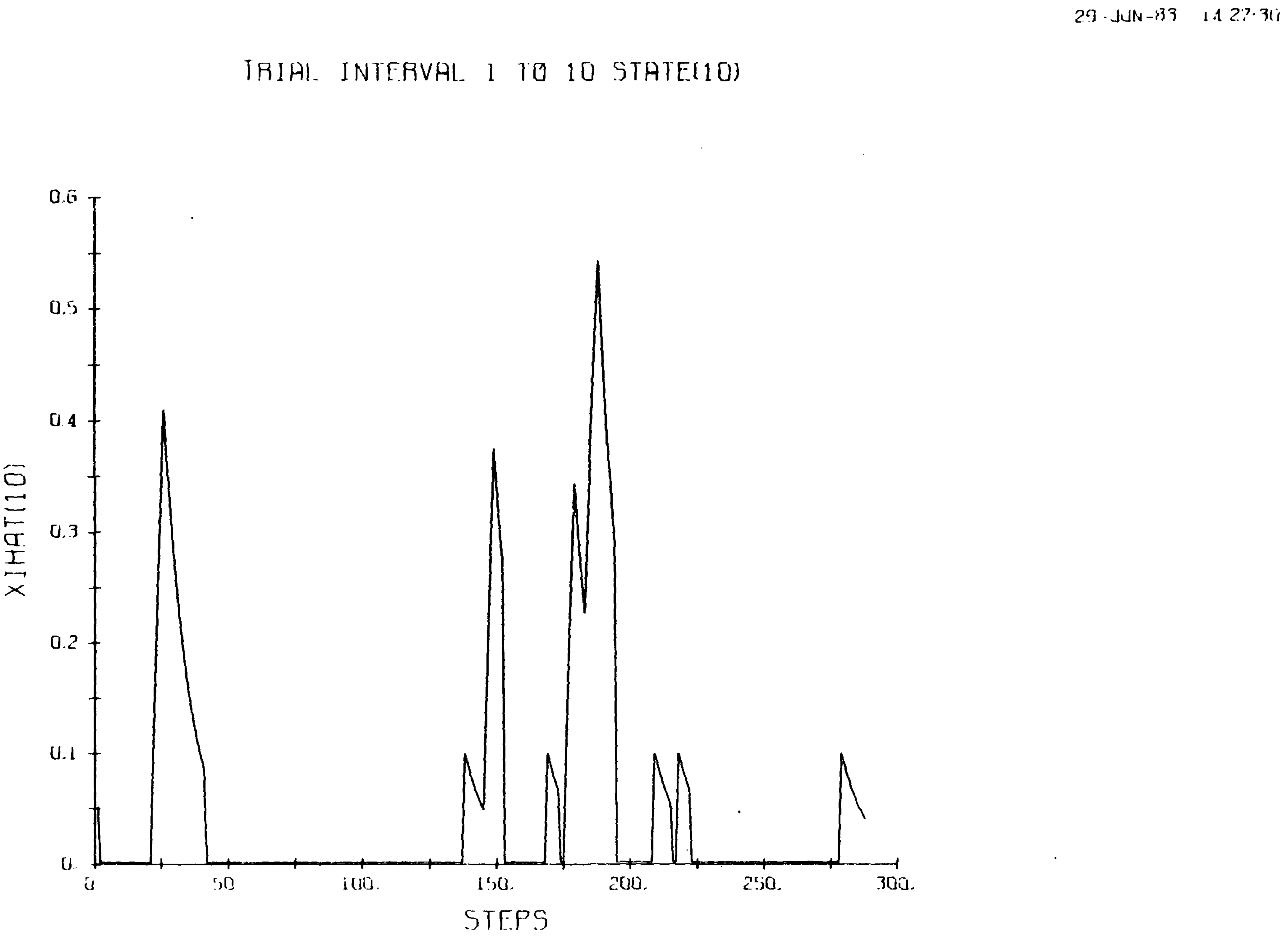


FIGURE 2.17. VARIATION OF $\hat{X}_{10}$, THE TRACE FUNCTION OF PATH NO. 10 OF THE ASE, OVER TEN TRIALS.

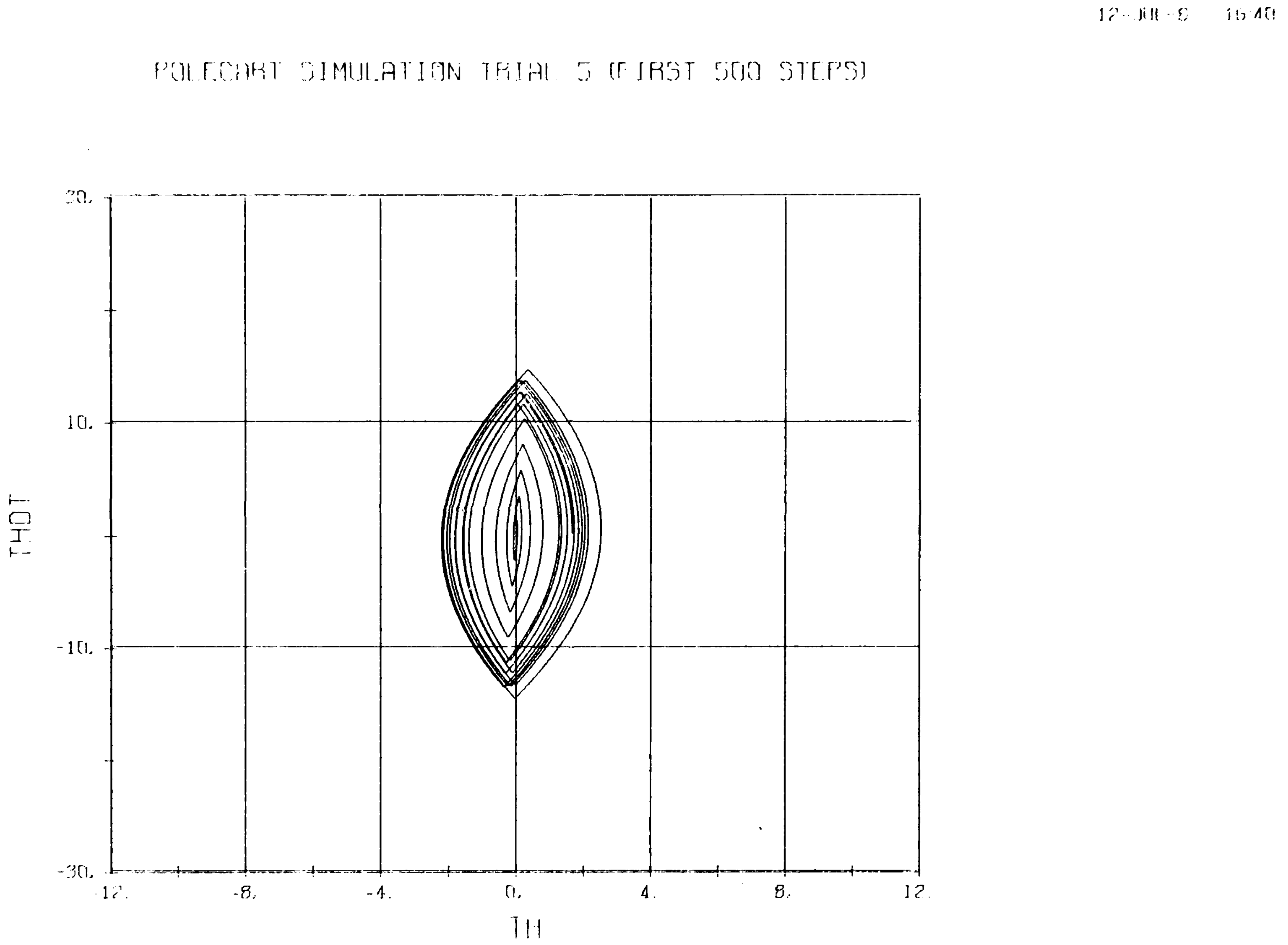


FIGURE 2.18. STATE SPACE PLOT FOR TRIAL NO. 5. FIRST 500 STEPS.

13-JUL-83 17:24:48

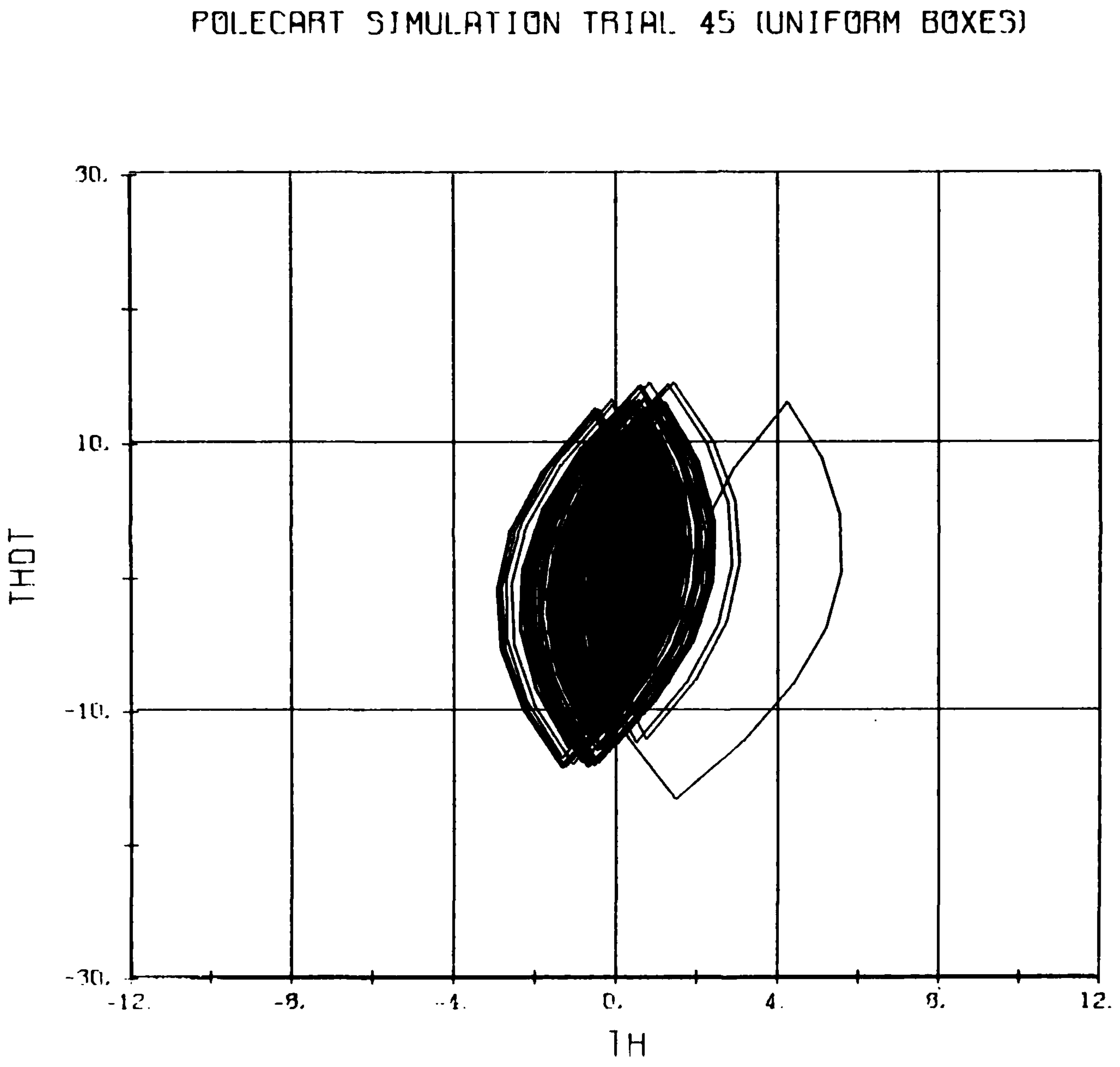


FIGURE 2.19. STATE SPACE PLOT FOR TRIAL NO. 45. FOUR THOUSAND STEPS.

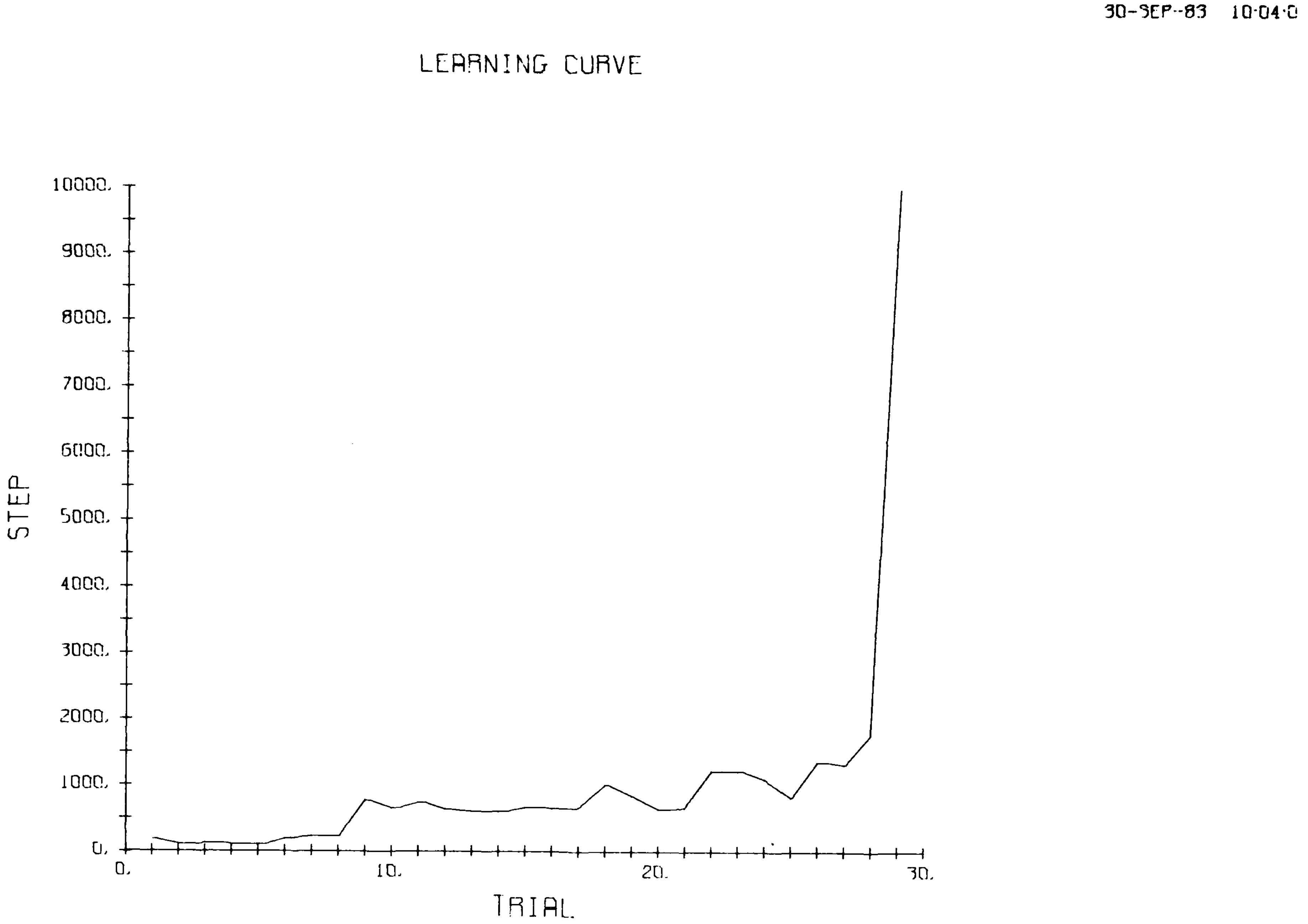


FIGURE 2.20. SYSTEM LEARNING CURVE WITH SAMPLING PERIOD T_S = 0.025 SEC.

7-OCT-83 12:43

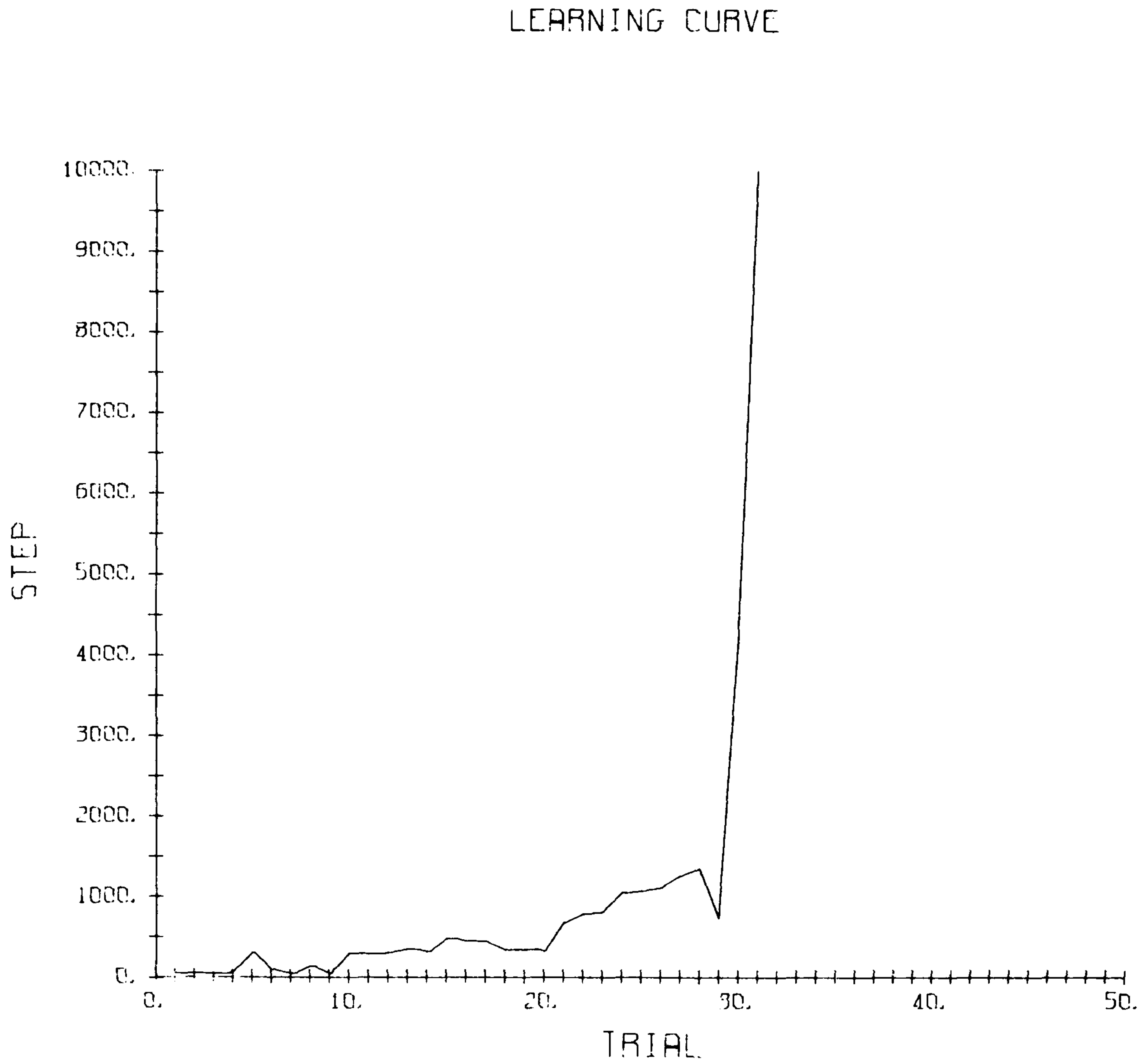


FIGURE 2.21. SYSTEM LEARNING CURVE WITH SAMPLING TIME T_s = 0.05 SEC.

7-OCT-83 12:45

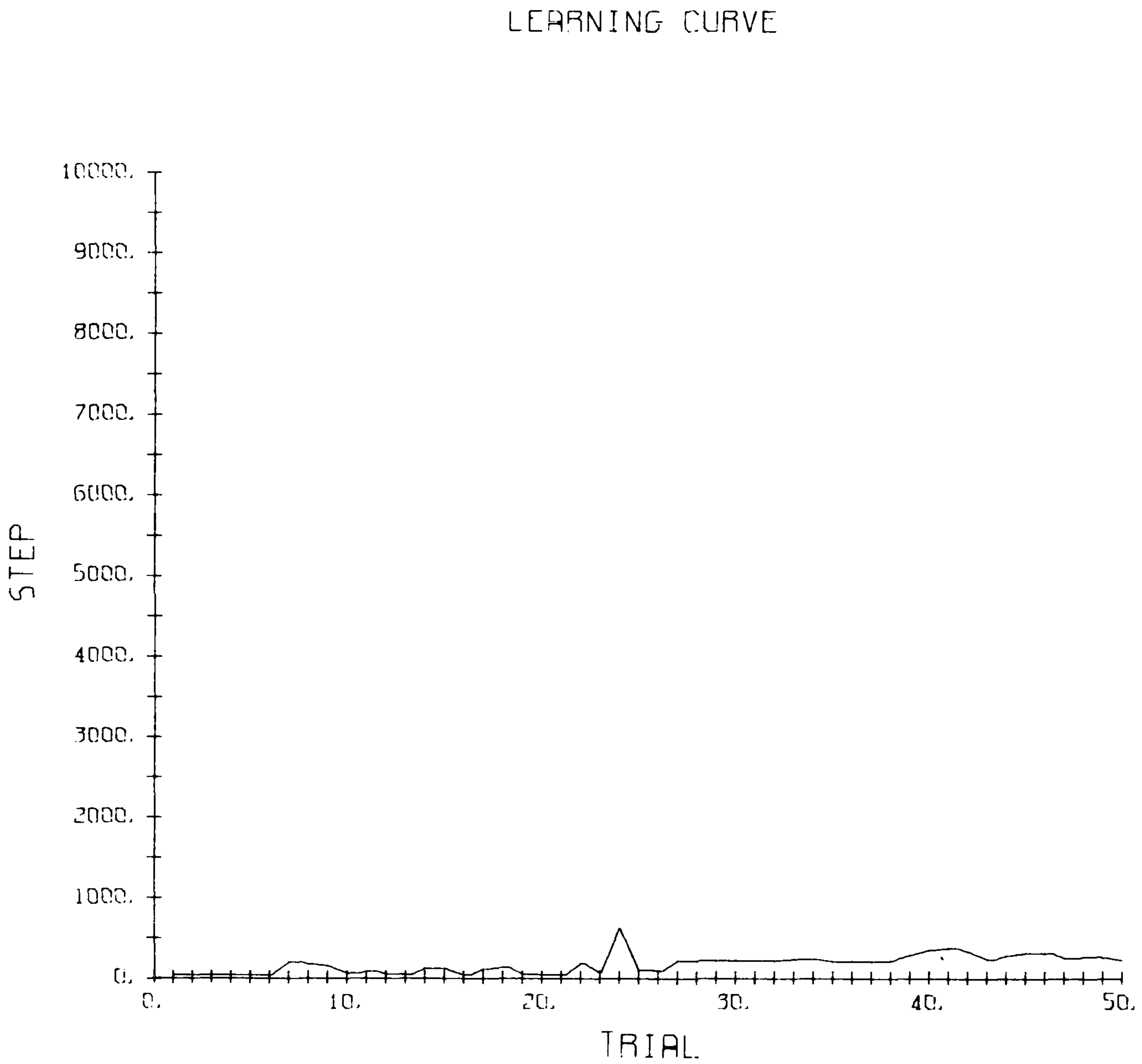


FIGURE 2.22. SYSTEM LEARNING CURVE WITH SAMPLING TIME T_s = 0.075 SEC.

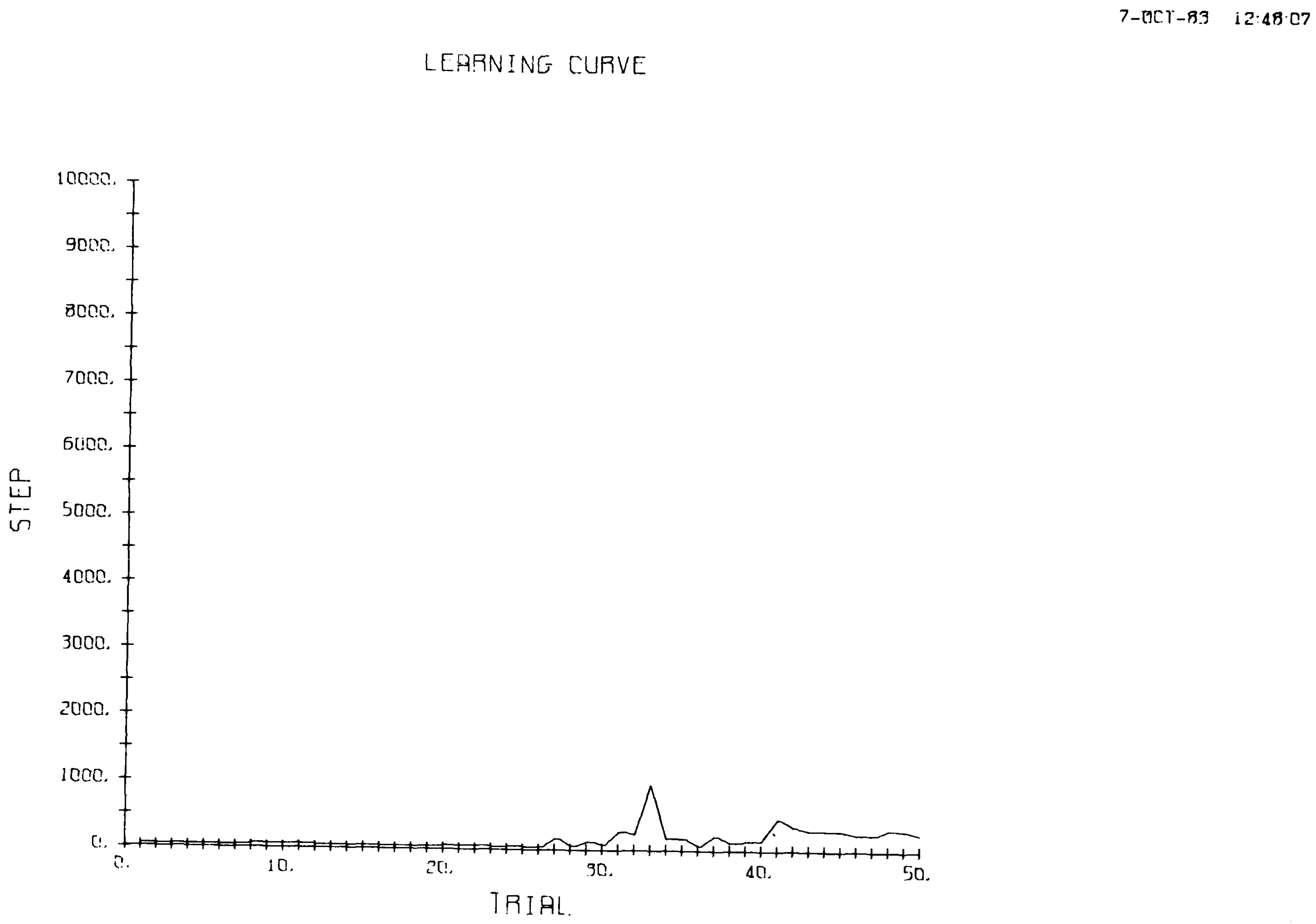


FIGURE 2.23. SYSTEM LEARNING CURVE WITH SAMPLING PERIOD T_s = 0.1 SEC.

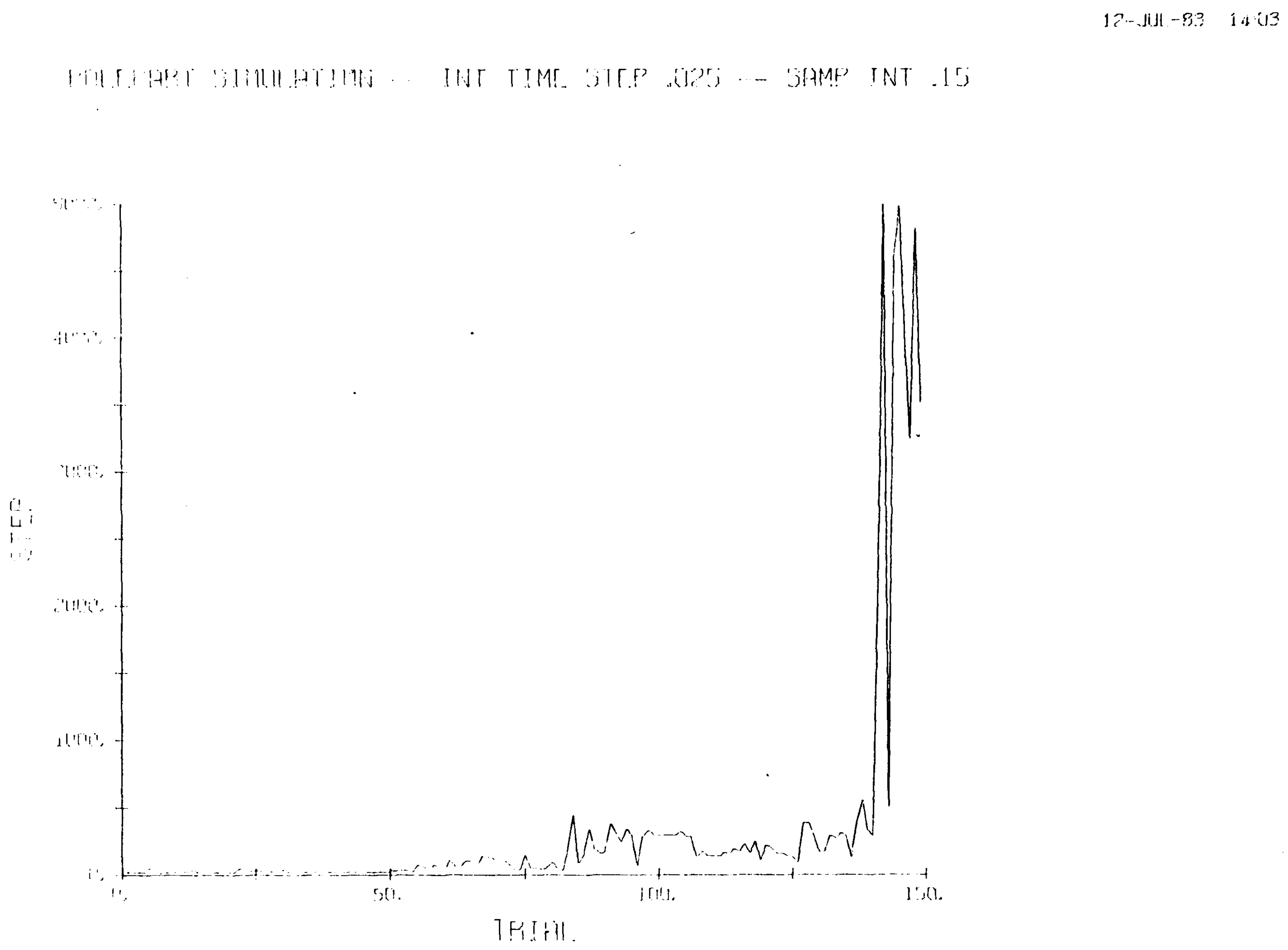


FIGURE 2.24. SYSTEM LEARNING CURVE WITH SAMPLING PERIOD T_S = 0.15 SEC.

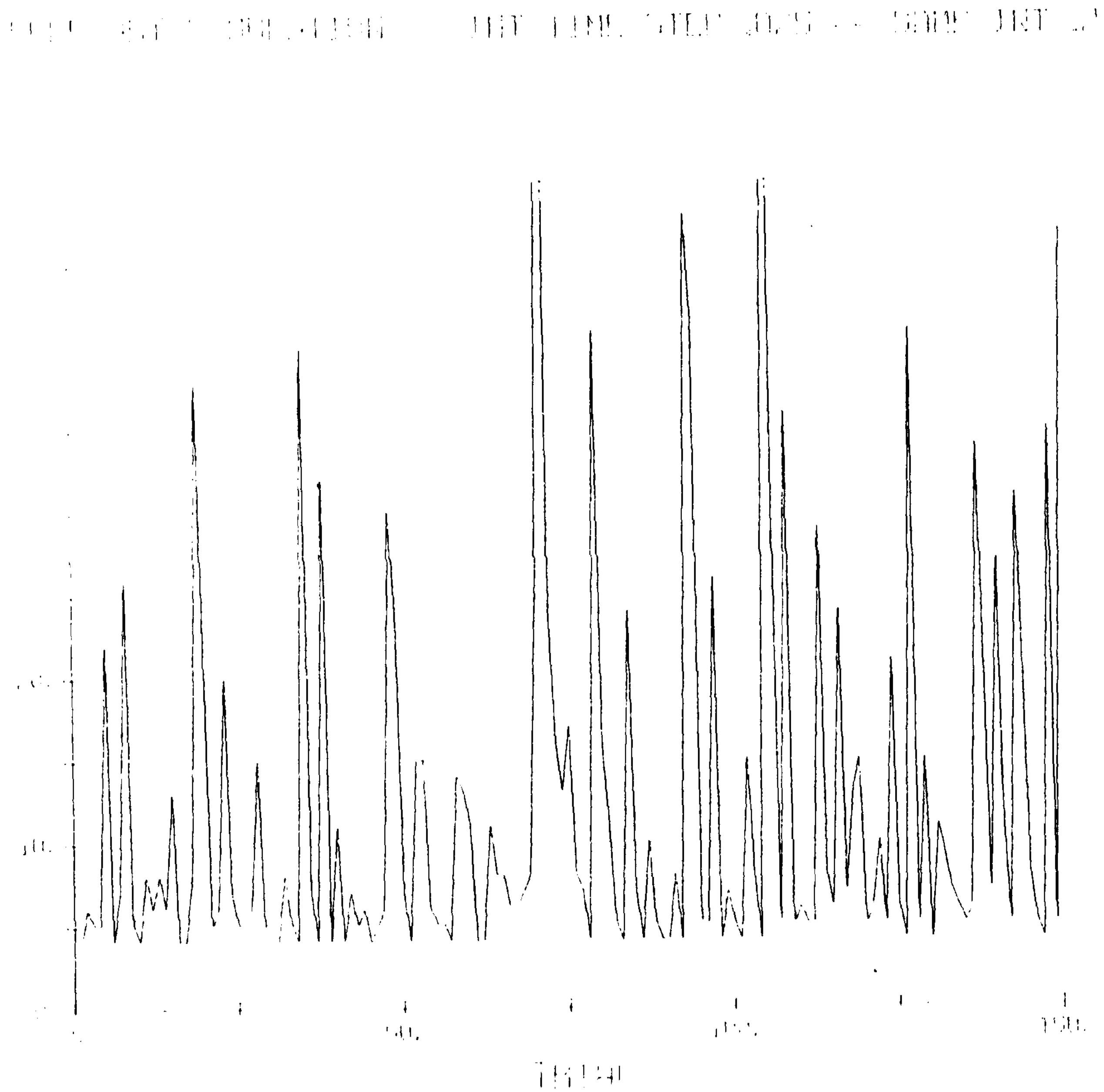


FIGURE 2.25. SYSTEM LEARNING CURVE WITH SAMPLING PERIOD T_s = 0.2 SEC.

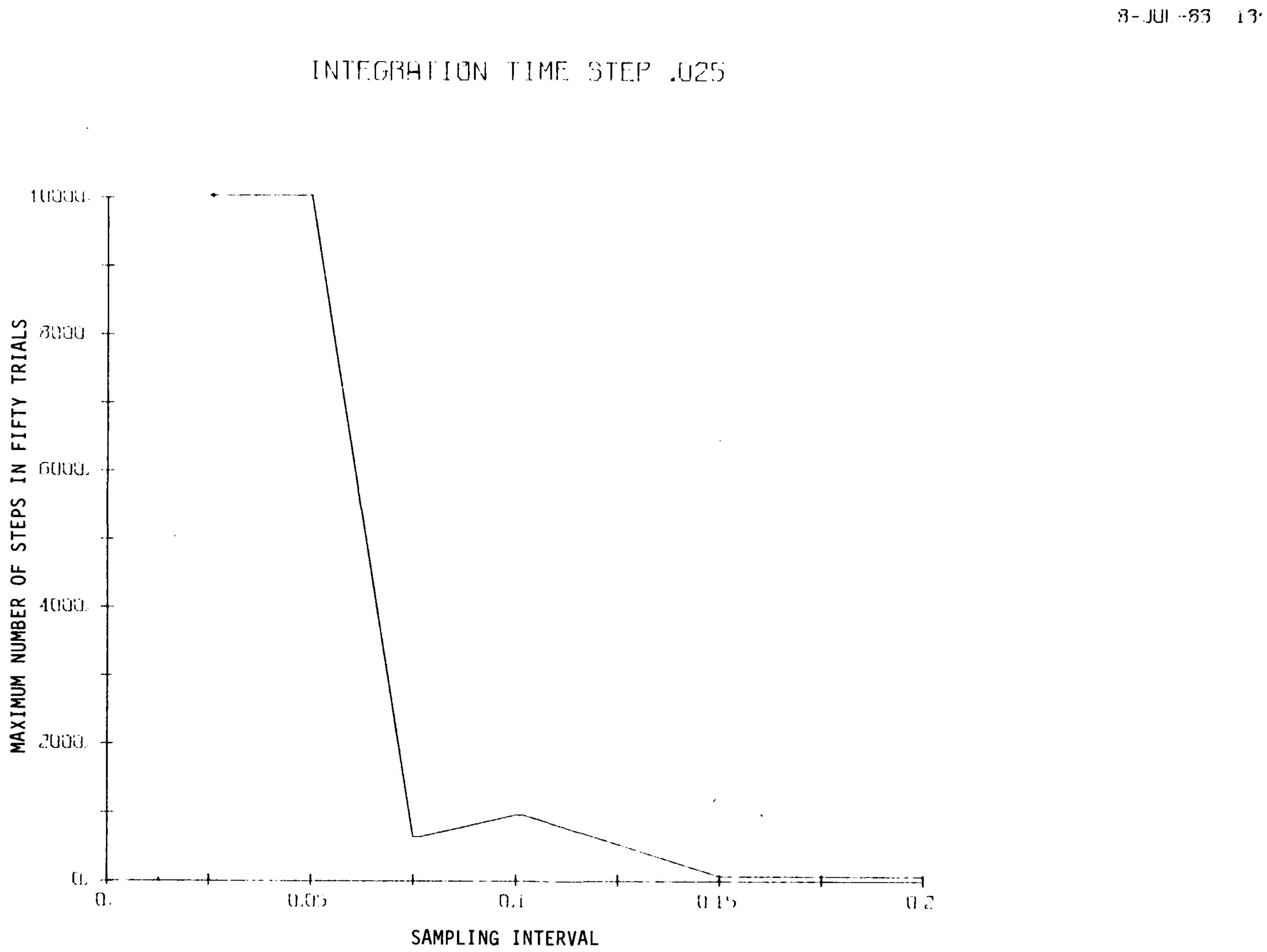


FIGURE 2.26. EFFECT OF SAMPLING PERIOD ON SYSTEM LEARNING.

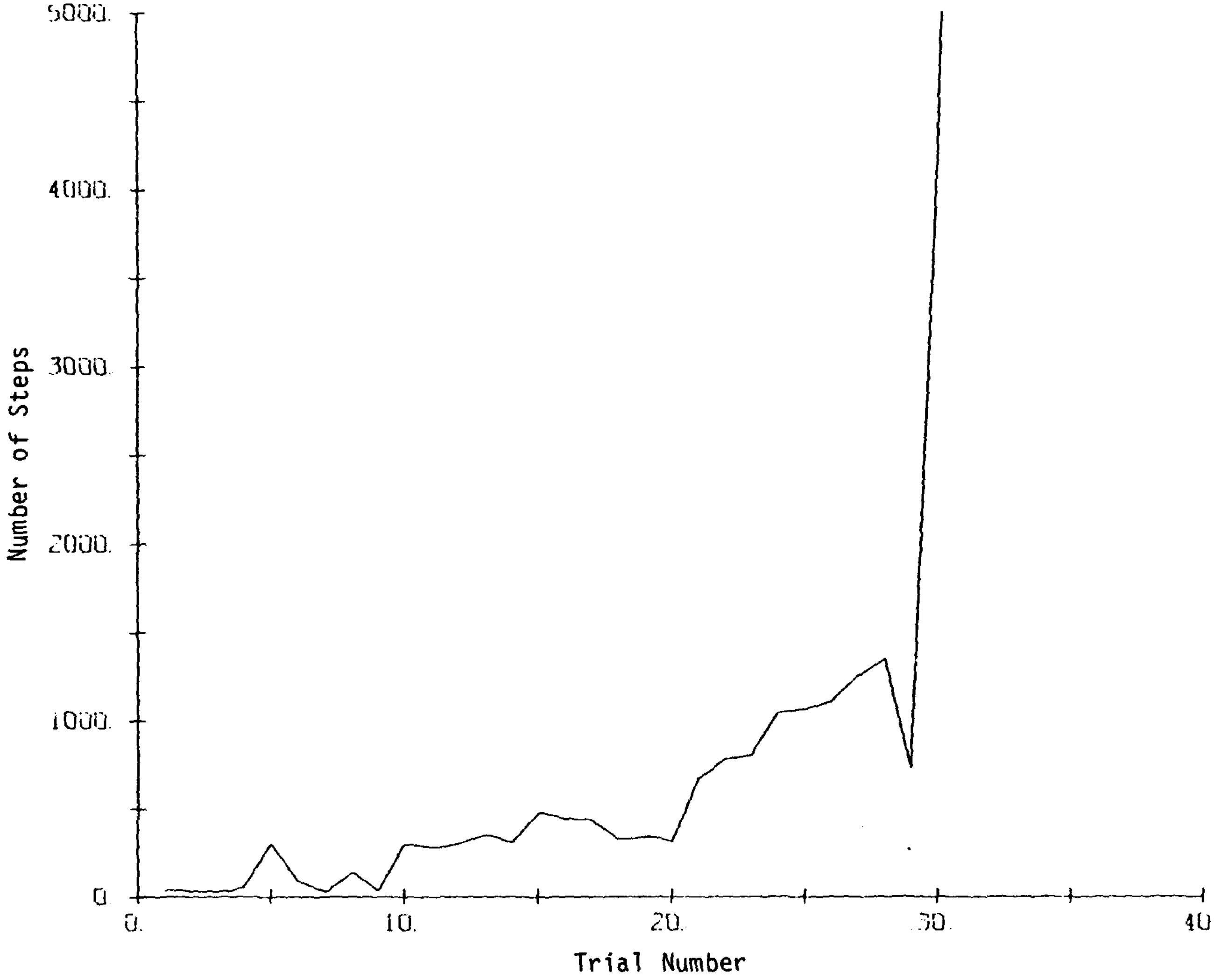


FIGURE 2.27. EFFECT OF NOISE LEVEL ON SYSTEM LEARNING. Noise $\sigma = 0.01$; $T_s = 0.025$ sec.

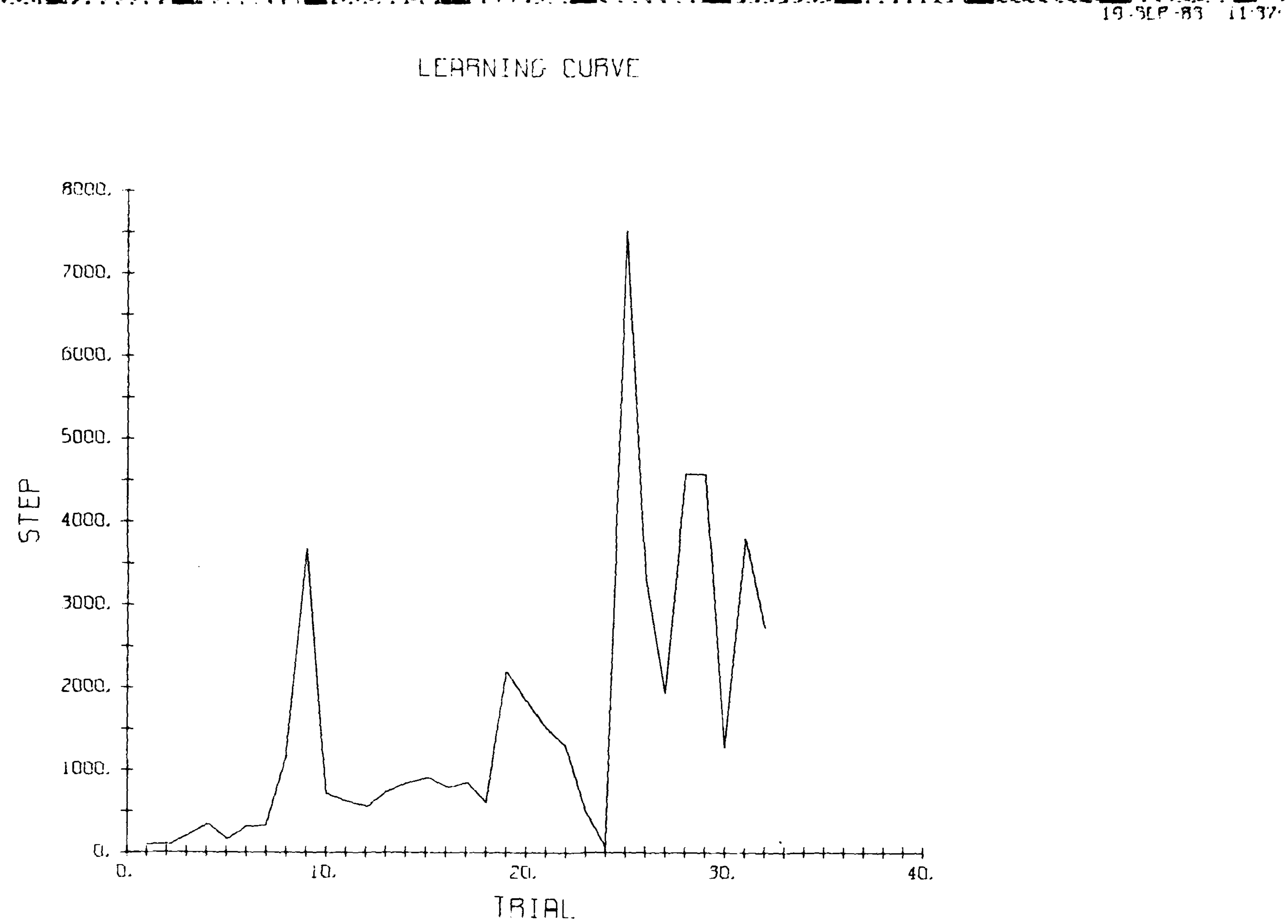


FIGURE 2.28. EFFECT OF NOISE LEVEL ON SYSTEM LEARNING.
Noise $\sigma = 0.1$; $T_s = 0.025$ sec.

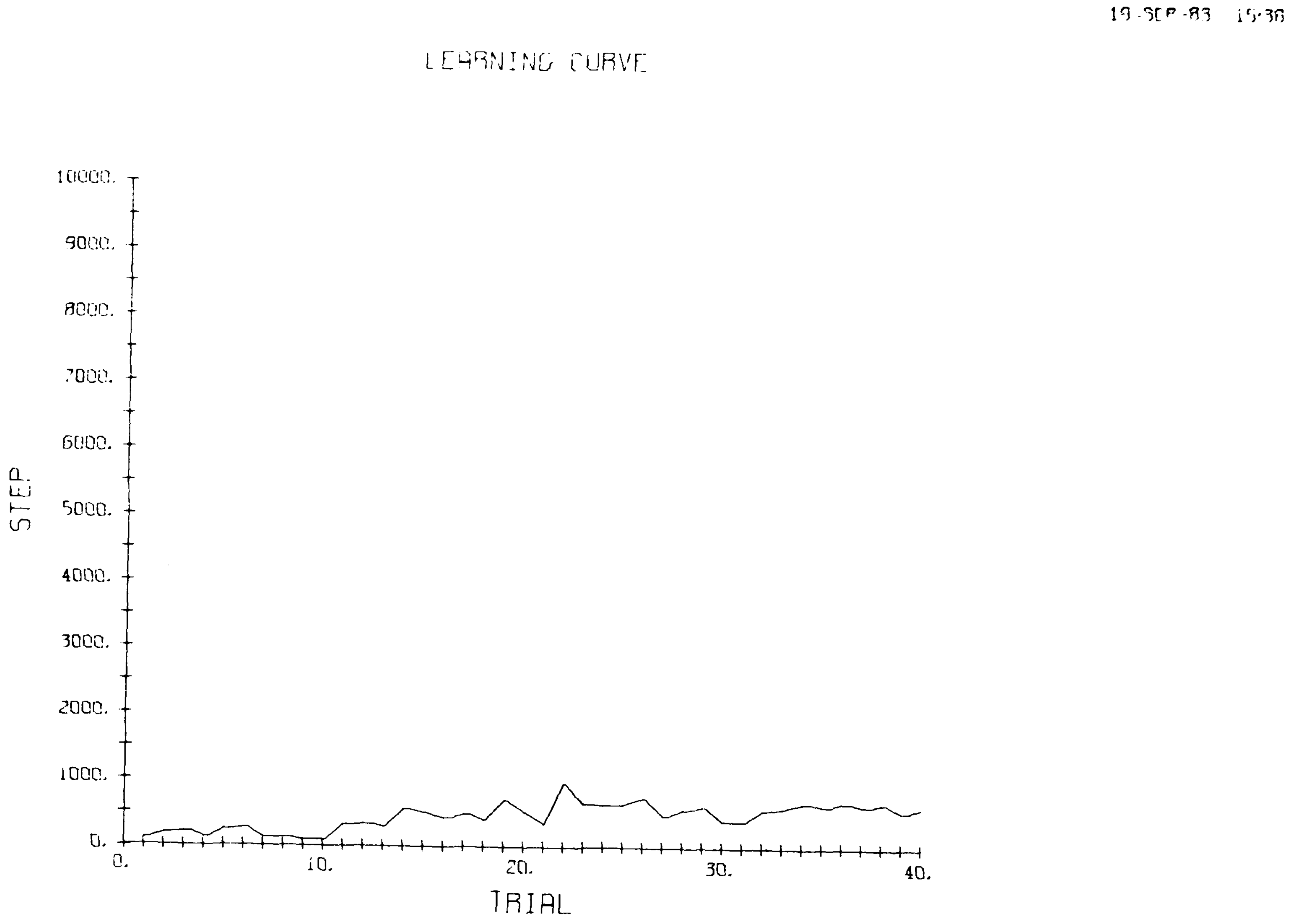


FIGURE 2.29. EFFECT OF NOISE LEVEL ON SYSTEM LEARNING. Noise $\sigma = 1$; $T_s = 0.025$ sec.

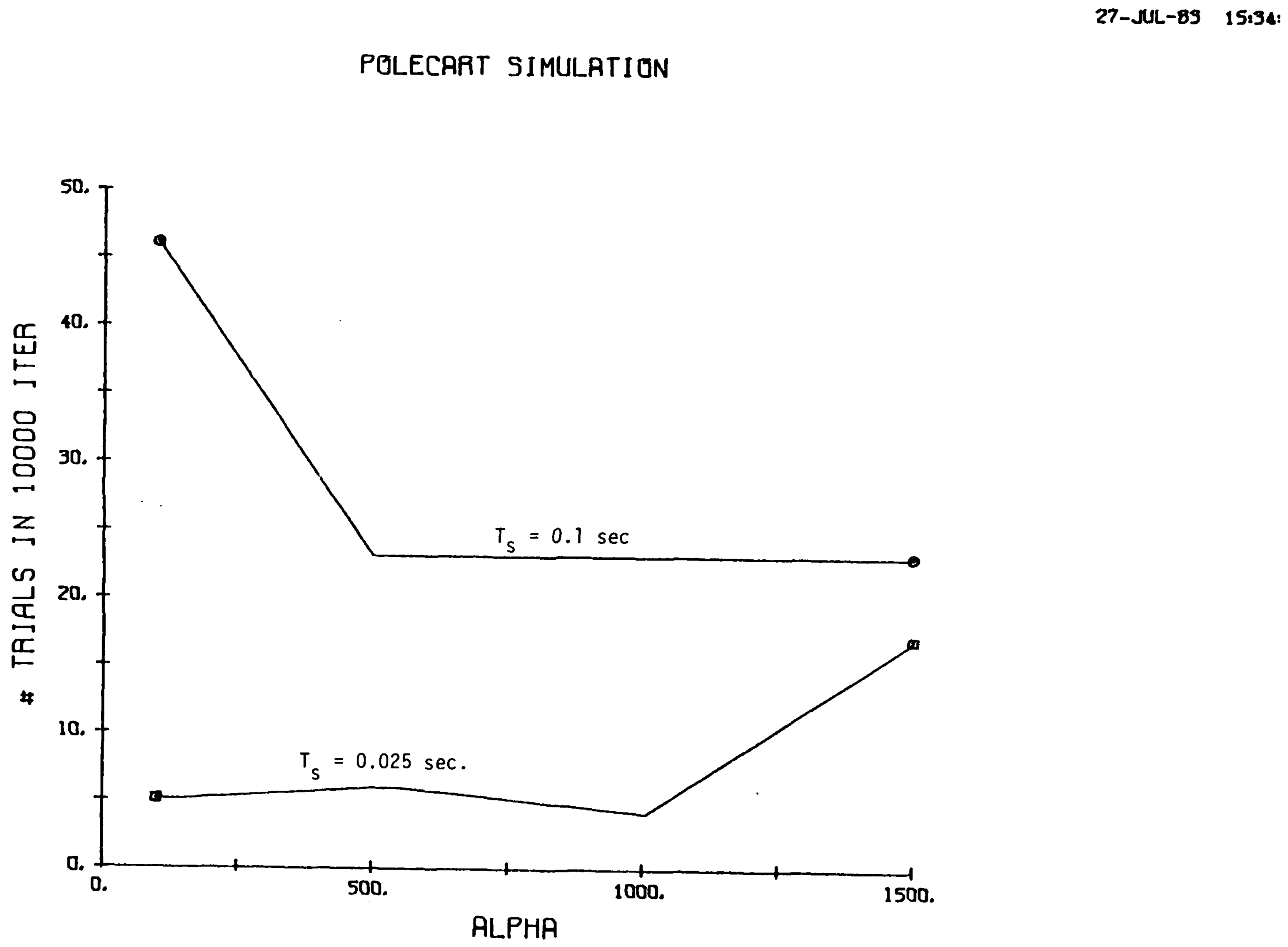


FIGURE 2.30. PERFORMANCE VARIATION WITH ALPHA.

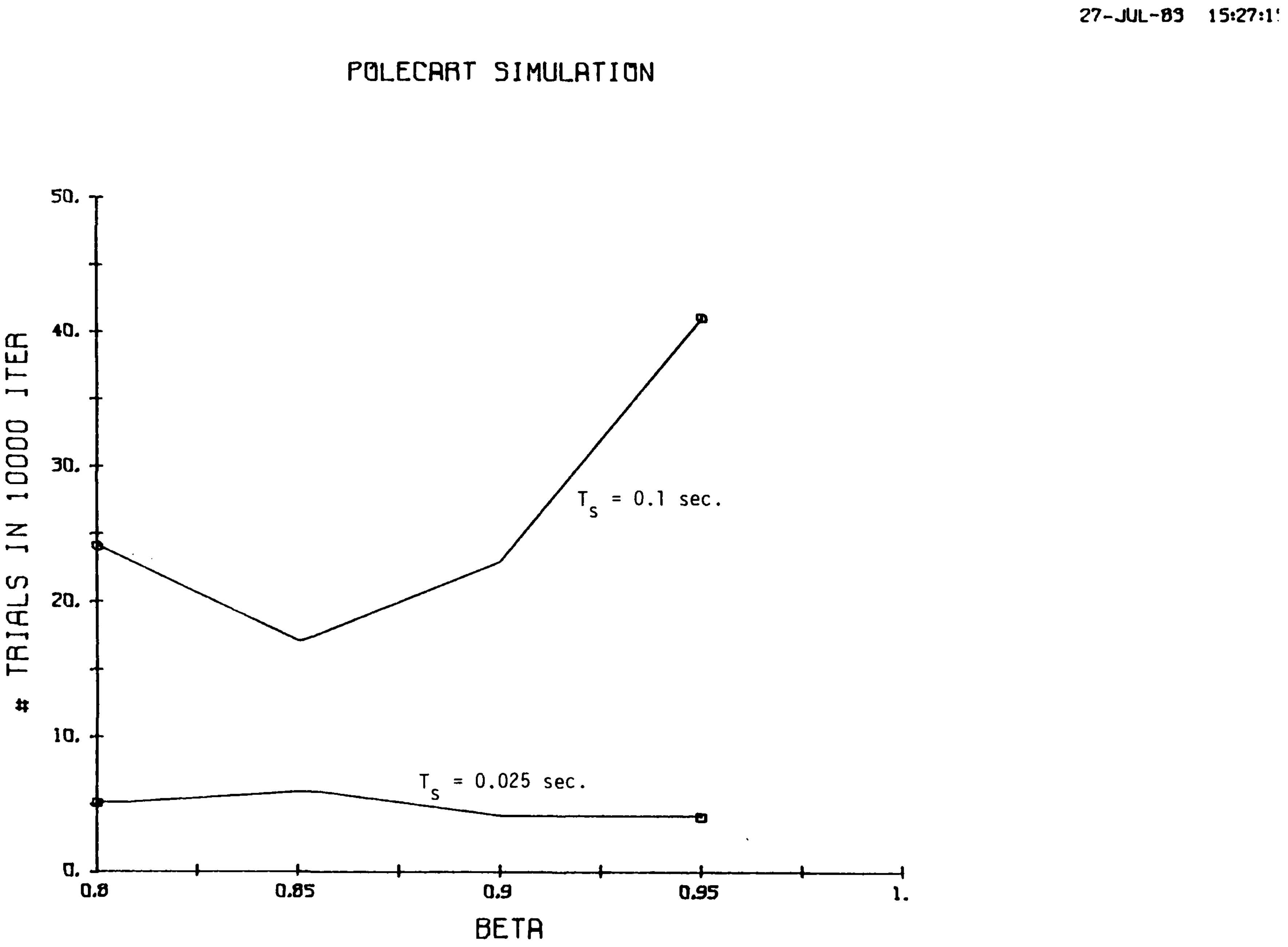


FIGURE 2.31. PERFORMANCE VARIATION WITH BETA.

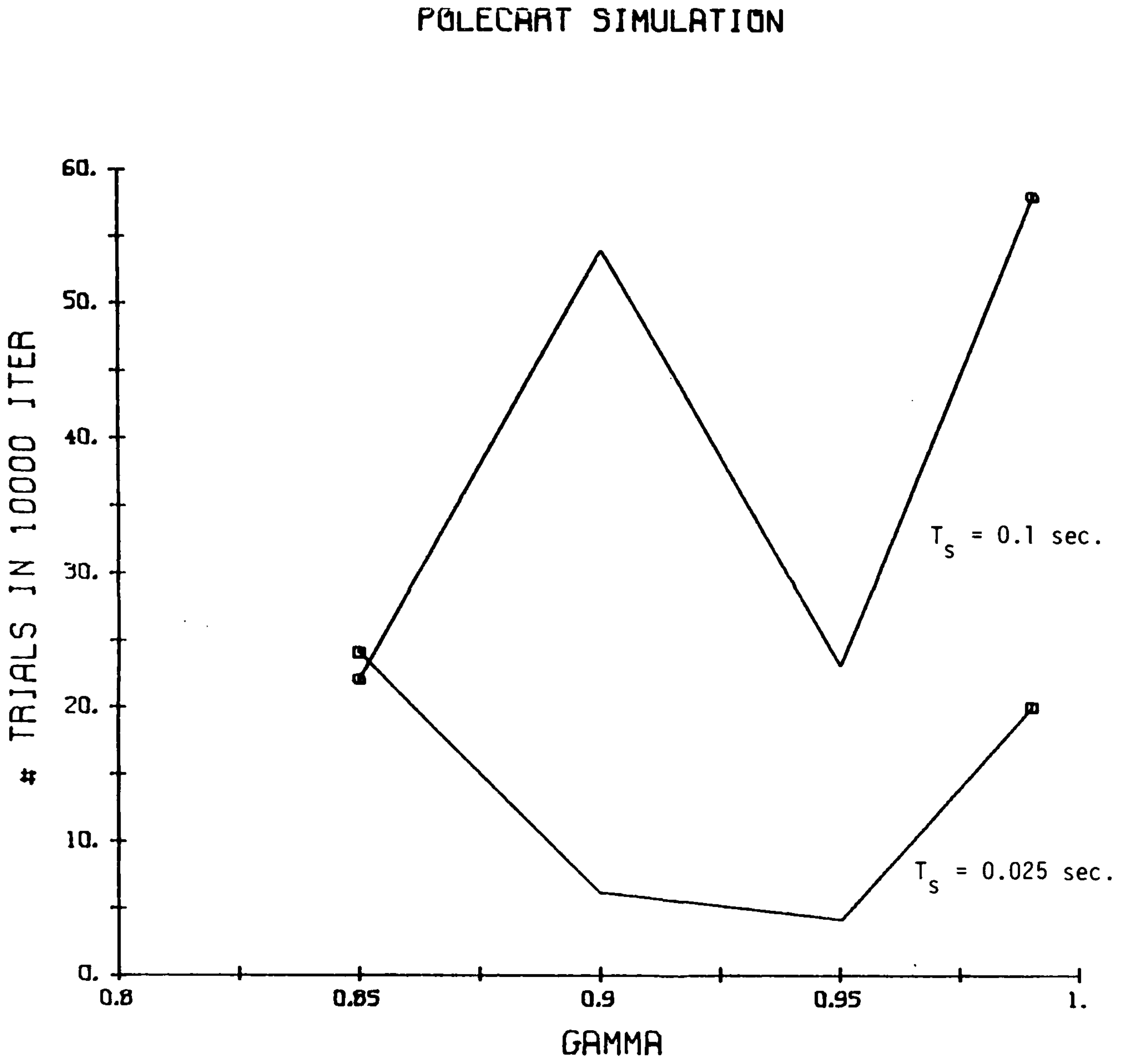


FIGURE 2.32. PERFORMANCE VARIATION WITH GAMMA.

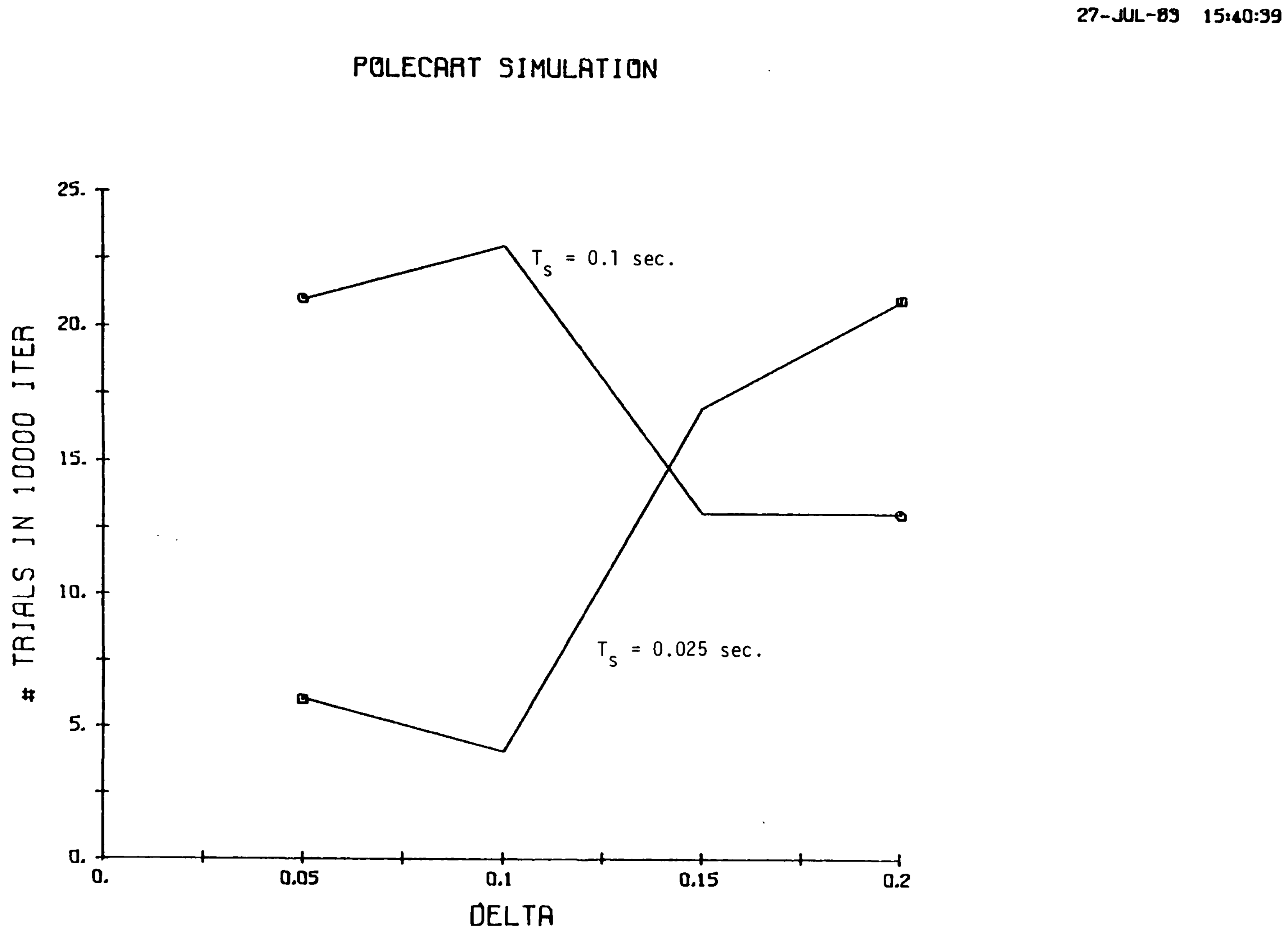


FIGURE 2.33. PERFORMANCE VARIATION WITH DELTA.

3
ASE-ACE COMPARED TO THE TIME OPTIMAL CONTROL LAW

During this study, it was observed that there is a similarity between the ASE-ACE as a controller and the time optimal control law for a double integral plant. Figure 3.1 illustrates the time optimal control problem for a simple double integral plant. Two integrators in series are driven by a control, u(t), and the objective of the control strategy is to drive the outputs of both integrators to zero in the least amount of time.

In Reference 4, it is shown that the time optimal control law for the double integral plant is the bang-bang controller where the input always assumes the value of either +1 or -1. Figure 3.1 is a plot of the state space for the double integral plant with the optimal switching curve drawn in as a solid line. The optimal switching curve is the set of all points (s_1, s_2) which satisfy the relationship $s_1 = - 1/2 \; |s_2| s_2$. The control u takes the value of +1 whenever a point in the state space falls below the switching curve or falls on the portion of the switching curve in the lower right quadrant. The control u takes the value -1 whenever a point in the state space falls above the switching curve or falls on the portion of the switching curve in the upper left quadrant. For any initial conditions, the two integrator outputs are driven to zero after the control has sequenced either from +1 to -1 or -1 to +1.

Ideally, after the double integral plant has been driven to zero, it will stay there until the next set of initial conditions. In practice this will never happen. If there is any noise in the measurements of s_1 and s_2 or any time delay in switching, the control system will go into a limit cycle about the origin. The size of the limit cycle will depend on the amount of noise or the amount of time delay in switching that exists. The time optimal controller does keep the output of the double integral plant bounded with time.

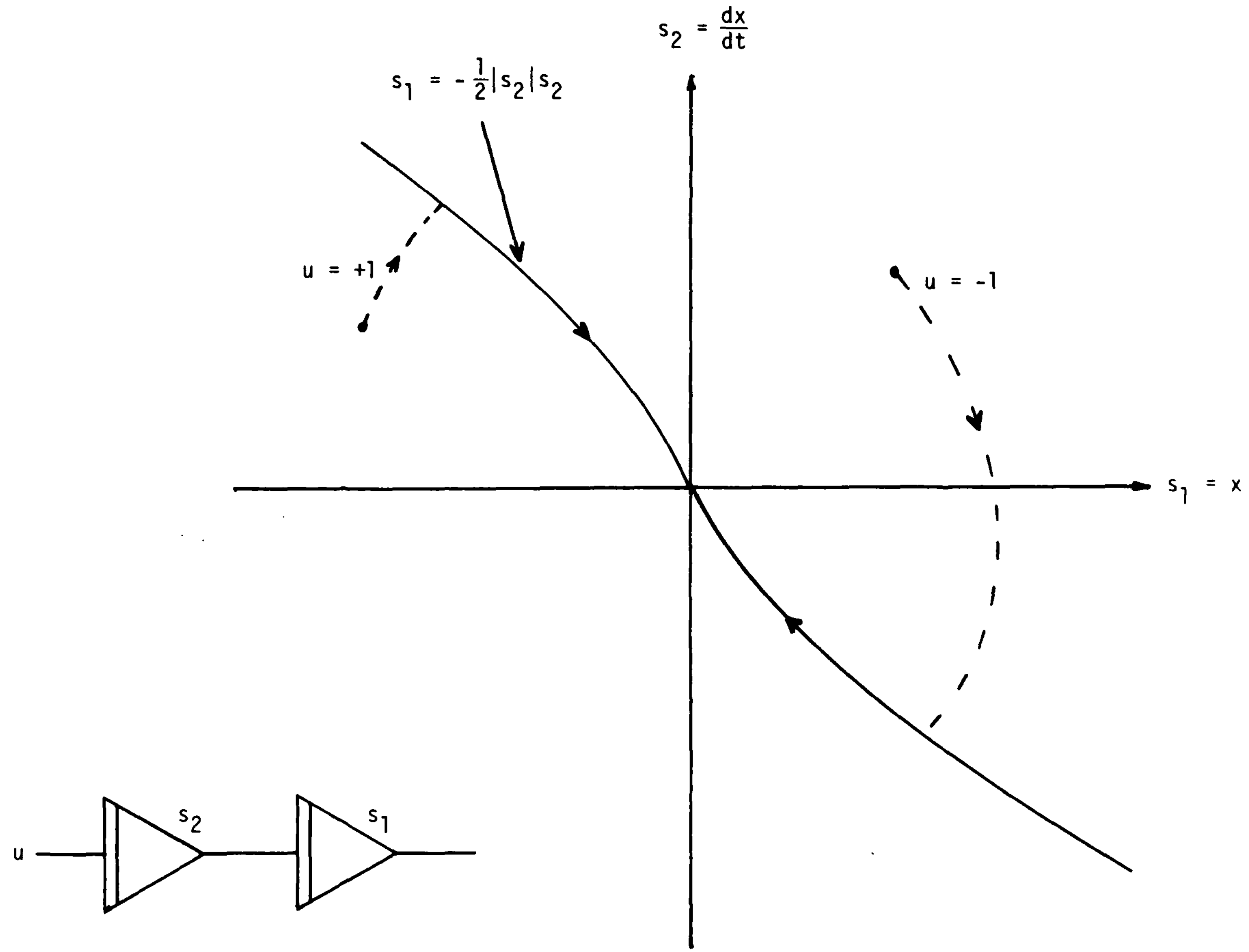


FIGURE 3.1. BANG-BANG TIME OPTIMAL CONTROL.

The pole-on-cart system can be viewed as two double integral plants which are coupled and driven by a common input. The pole hinged on the cart is one double integral plant where the output of the first integrator is the angular velocity of the pole and the output of the second integrator is the angular position of the pole. The cart on the track forms the second double integral plant where the velocity of the cart on the track is the output of the first integrator and the position of the cart on the track is the output of the second integrator. Both double integrator plants are driven by the force applied to the cart.

The ASE-ACE uses a control force to drive the pole-on-cart system which takes only the values of +1 or -1. Therefore, the ASE-ACE is driving two double integral plants with a bang-bang control. The ASE-ACE is performing the function of the optimal control law shown in Figure 3.1 for the single double integral plant. It must learn for every region in the state space the proper direction to drive the system. The ASE-ACE must, however, do this for a four dimensional system.

Consider the optimal control law shown in Figure 3.1. Note that there are closed and open sets of points for which the control action is the same. Therefore, the state space could be divided up into a set of regions referred to as boxes sometimes before, where all the points in any box are associated with the same control action. Also the problem could be restricted to some subspace of the state space such that any point outside this region can never be an initial condition and if the control law drives the system to this point, the controller can be assumed to have failed to properly control the system.

Consider now what would happen if instead of allowing the controller to continuously observe the state of the double integral plant, the controller could only sample the state of the system periodically. This can be viewed as introducing time delay into the system, which it is known will cause the controller to go into a limit cycle about the origin. The size of the limit cycle will depend on how often the controller observes the state of the system. If the controller observes the state of the system too infrequently, the state of the system could exit the allowed region causing a failure.

Comparing the pole-on-cart system controlled by the ASE-ACE to the double integral plant time optimal control problem, it would be expectd that the ASE-ACE can control the system if it learns a control law similar to the one shown in Figure 3.1 and the final state of the system should limit-cycle about the origin. This assumes the existence of a four-dimensional switching curve for the four-dimensional pole-on-cart problem.

A final point of comparison between the time optimal control law and the ASE-ACE controller is that the ASE-ACE uses boxes of non-uniform size. *Figure 3.2 shows one method of defining regions of* the double integral state space where the control action is the same for every point in the region. Figure 3.2 shows a set of switching curves (solid lines) passing through different points on the velocity axis and trajectories starting at the points $\dot{X}_{max}$ and $\dot{X}_{min}$ (dotted lines) for control actions +1 or -1. The regions formed by the intersections of the switching curves, trajectories and x-y axes contain open sets of points which have a common control action. These regions have different sizes and are not rectangular boxes. Forming these regions requires knowledge about the propagation of the system for any control action and the optimal control law. In the absence of such information, it is reasonable to divide the state space up into rectangular regions and regions of unequal size. Note that in using rectangular regions, any given trajectory may exit the subspace defined by the rectangular regions and then re-enter at a later time. This event may not be marked as a failure if the trajectory exits

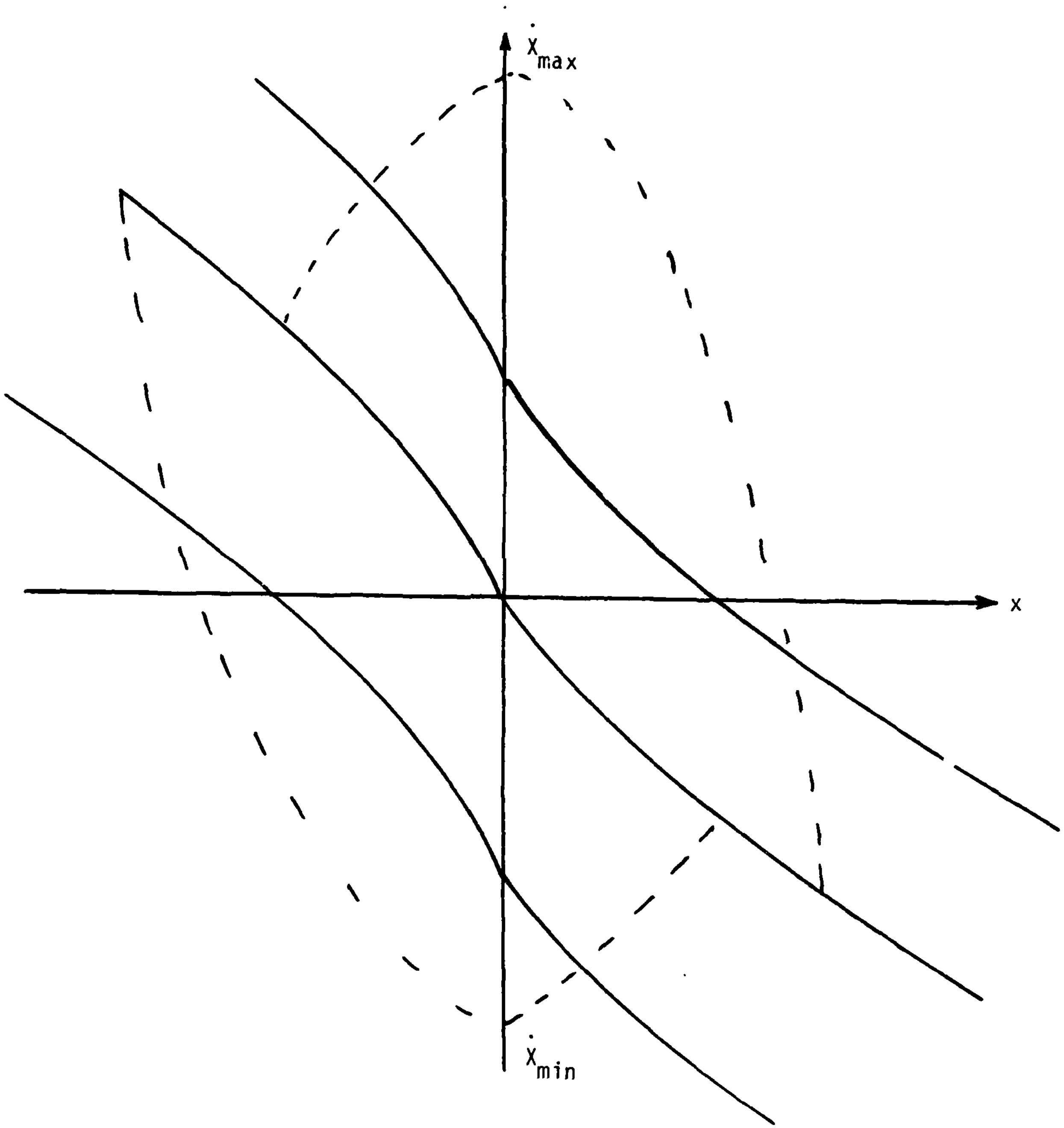


FIGURE 3.2. REGIONS ON COMMON CONTROL ACTION.

the subspace because of velocity and not position. Situations can arise where the state of the system is not in any of the defined regions (boxes) but failure has not occurred. This cannot happen with the regions defined in Figure 3.2.

Figure 3.3 shows how the ASE-ACE can be applied to the double integral plant and Figure 3.4 is a plot of the state space of the double integral plant after the ASE-ACE has learned. The state space plot shows a limit cycle as previously predicted would happen. In the limit cycle, the ASE-ACE only passes through four states so only the weights for those four states must be learned.

It has been shown that the ASE-ACE as a controller for the pole-on-cart system is similar to the bang-bang controller for the double integral plant. The ASE-ACE can be thought of as learning the switching curve for a four-dimensional bang-bang controller. The stability and performance of the ASE-ACE is expected to be very similar to the stability and performance of the bang-bang controller.

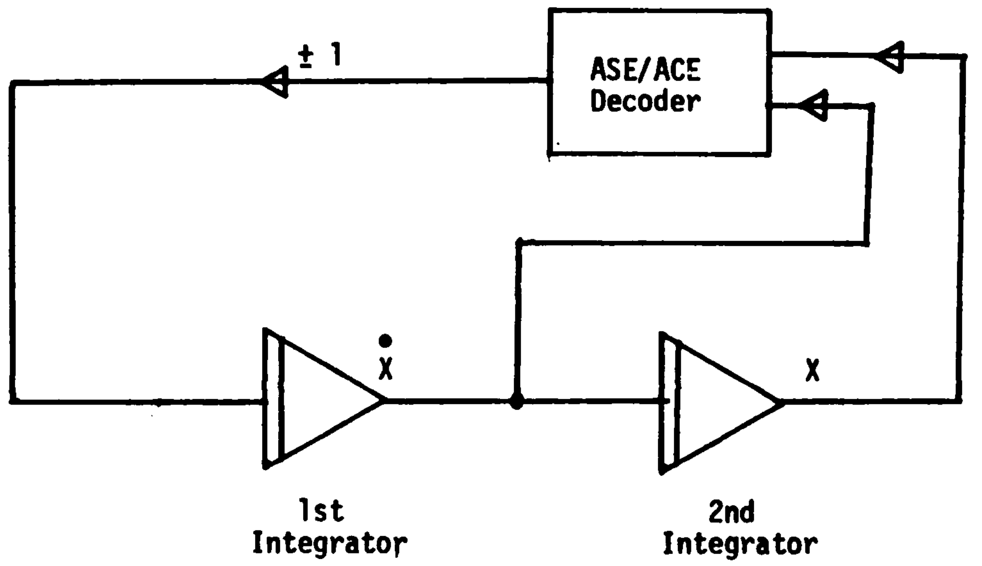


FIGURE 3.3. MODEL OF DOUBLE INTEGRAL PLANT.

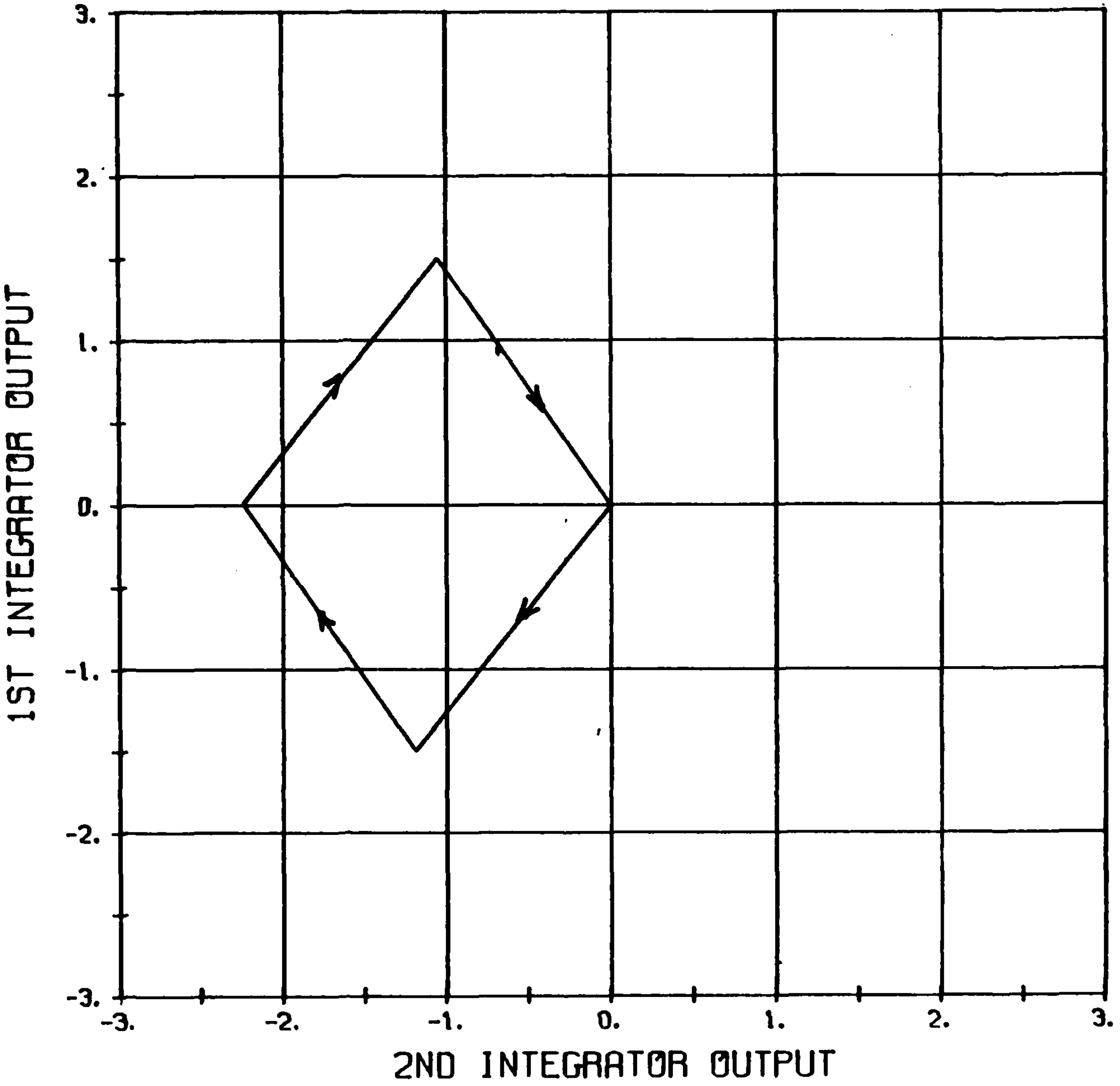


FIGURE 3.4. STATE SPACE FOR DOUBLE INTEGRAL PLANT.

4
MINIMIZING A FUNCTION USING ASE-ACE

Careful study of the ASE-ACE learning controller has led us to believe that there are potential applications of this system to several problems related to synthetic aperture radar. Examples of such problems are:

1. Higher order focus (autofocus)
2. Radar system design optimization
3. SAR moti·.n compensation
4. Target detection and recognition
5. Phase reconstruction algorithm
6. Evaluation and analyzing of multiple error sources
7. Image matching.

A common characteristic of most of these problems is that they can be studied from the perspective of minimizing some performance function. To apply the ASE-ACE to these problems, therefore, the general problem of minimizing a function must be put into the structure that the ASE-ACE was designed to handle.

4.1 FIRST APPROACH TO MINIMIZING A FUNCTION

Two approaches to minimizing a function using the ASE-ACE have been stuaied. Figure 4.1 illustrates the first approach that was used to minimize a function, and shows a feedback system which is stable only when the function $F(X_I)$ is identically zero. The variable X_I represents the present estimate of the value of X that minimizes the function F(X). Each iteration of the control system, the value of X_I is changed by the value of $A_{I+1}F(X_I)$ where A_{I+1} is a positive real number. Unless $F(X_I)$ equals zero at some point, X_I will eventually grow without bound.

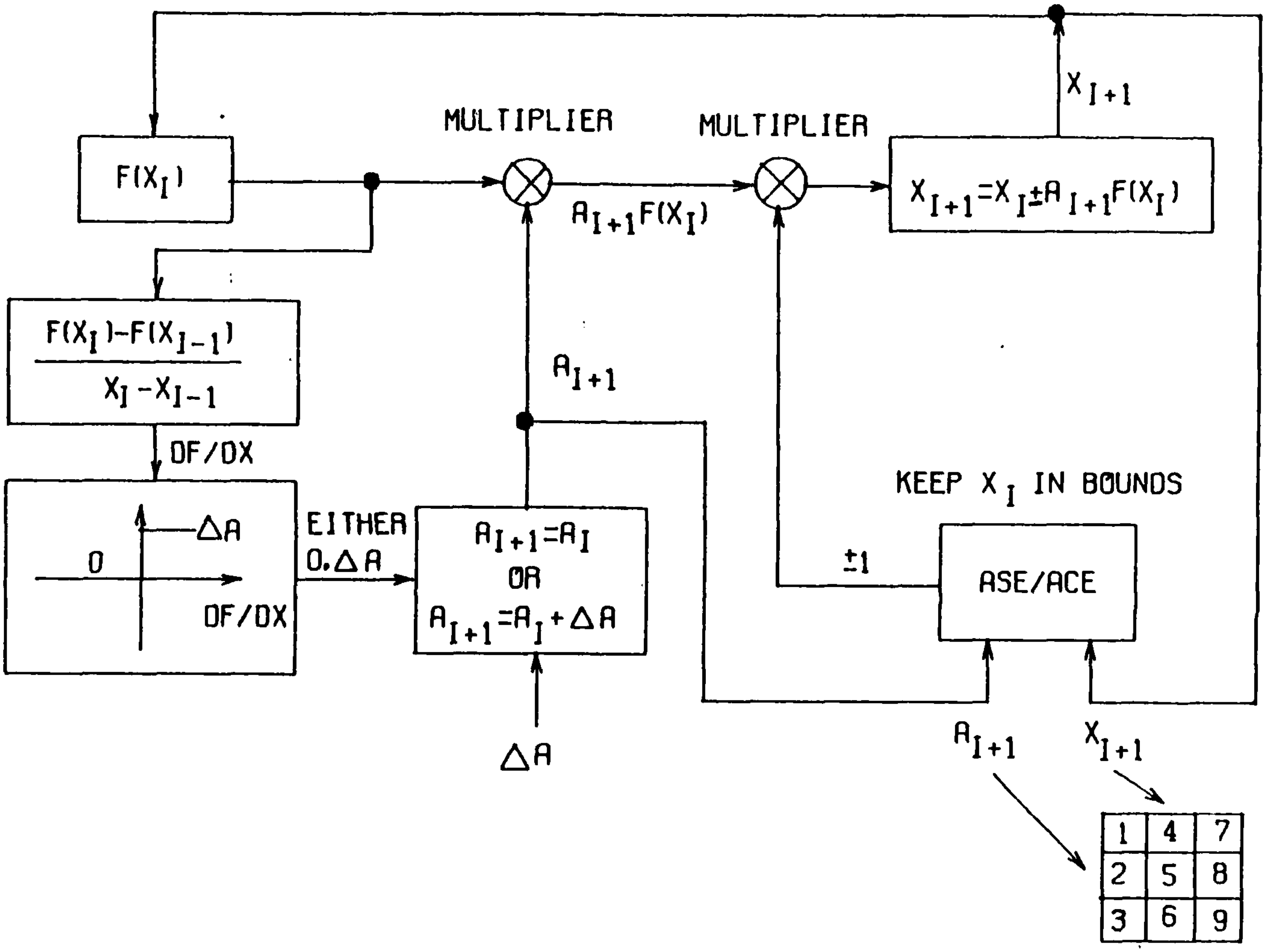


FIGURE 4.1. FIRST APPROACH TO MINIMIZING A FUNCTION.

The control path in the lower lefthand side of Figure 4.1 shows how A_I is computed. Using two consecutive values of $F(X_I)$ and X_I, an estimate is generated for the rate of change of F(X) with respect to X, DF/DX. If DF/DX is positive, A_I is incremented by a constant positive real number ΔA. This is a form of penalty. Unless $F(X_I)$ is decreasing, A_I will grow unbounded and force X_I to grow even faster with time. Initially, A_I is set equal to ΔA.

The ASE-ACE algorithms noted in the lower righthand corner of Figure 4.1 use a two dimensional state space consisting of A_I and X_I. The output of the ASE-ACE algorithms has a value of 1 and can be positive or negative. The ASE-ACE controls whether X_I is increased or decreased from its present value. In order for the ASE-ACE to keep X_I within bounds, it must drive $F(X_I)$ as close to zero and as quickly as possible. If X_I or A_I exceed the selected limits for the problem, the ASE-ACE has failed and it is punished.

The control problem illustrated in Figure 4.1 is very similar to the pole-on-cart problem. Both problems involve a basically unstable system with at most one stable equilibrium point. Both problems also involve two basic control variables. The variable X_I can be compared to the angle of the pole and the variable A_I can be compared to the position of the cart on the track. They differ primarily in that the state space for the minimization problem involves two variables and the pole-on-cart involves four variables.

Figure 4.2 is a state space plot for the control system shown in Figure 4.1 after the ASE-ACE has begun to learn how to control the system. The vertical axis of the plot is A_I and the horizontal axis is X_I. If the ASE-ACE is controlling the system properly, the value of X should move to zero and stay near zero for the function, $X^2 + 1$, that was selected. In this case, the function is never zero so there is no equilibrium point. The boxes shown in

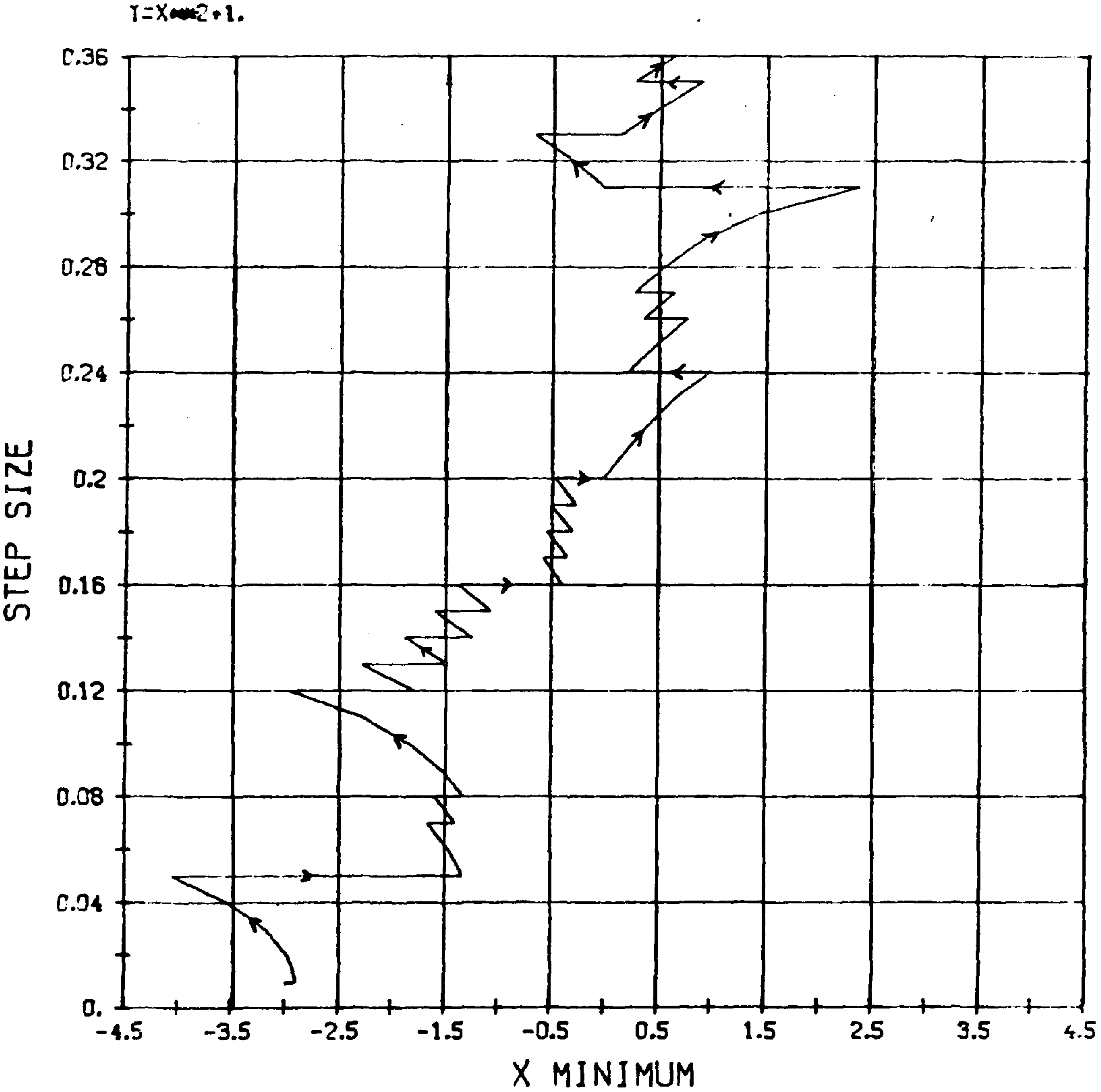


FIGURE 4.2. STATE SPACE PLOT FOR FIRST APPROACH TO MINIMIZING A FUNCTION.

Figure 4.2 are the actual boxes used by the ASE-ACE and failure corresponds to the plot leaving the bounds of the plot region.

The state space plot shown in Figure 4.2 starts with an initial value of X_I equal to -3 and an initial value of A_I equal to 0.01. The ASE-ACE has had several trials to learn prior to the trial shown in Figure 4.2. Although the initial change in X_I is away from zero, X_I does eventually move to zero, and no matter how far it may get away from zero, it always moves back to zero. This trial failed because A_I exceeded the plot limits, but when it failed, X_I was near zero. Eventually, X_I should move more rapidly to zero and stay close to zero for a longer period of time.

4.2 SECOND APPROACH TO MINIMIZING A FUNCTION

Figure 4.3 is a block diagram of the second approach used to minimize a function. In this case, the estimate of X that minimizes F(X) is incremented by plus or minus ΔX where ΔX is a constant. The state space of the ASE-ACE includes X, F(X), dF/dX and d^2F/dX^2. Failure is defined as dF/dX being positive. Failure corresponds to the value of the function increasing after any step. At the minimum point, d^2F/dX^2 should be zero. This second approach is more straightforward than the first.

Figure 4.4 is a plot of the values of X as a function of time, or step number, for trial 97. Note that the ASE-ACE has learned to decrease X until it has reached the value of zero which is the correct minimum for the function $F(X) = X^2$. The value of X was reset to its initial value of 3 after each trial.

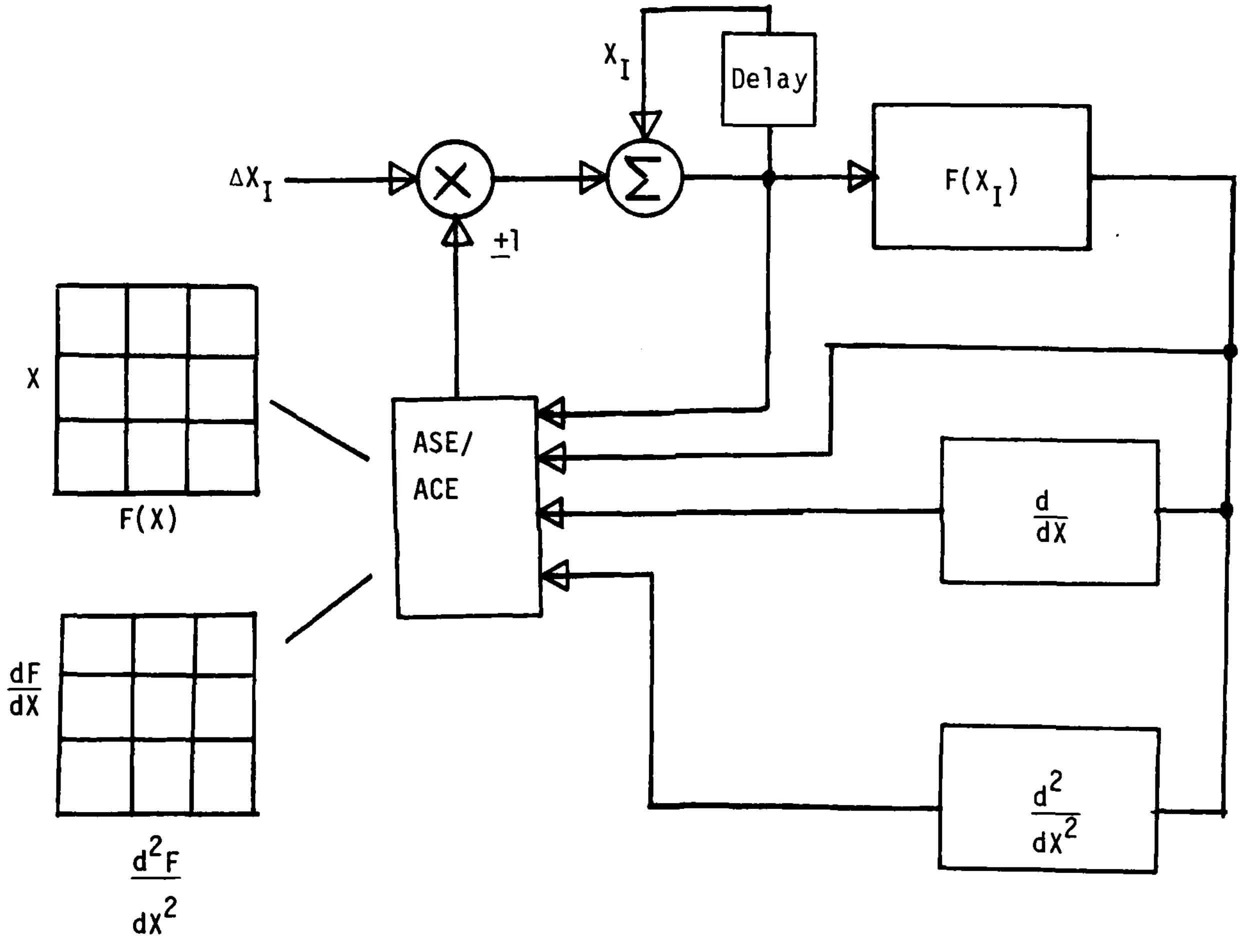


FIGURE 4.3. SECOND APPROACH TO MINIMIZING A FUNCTION.

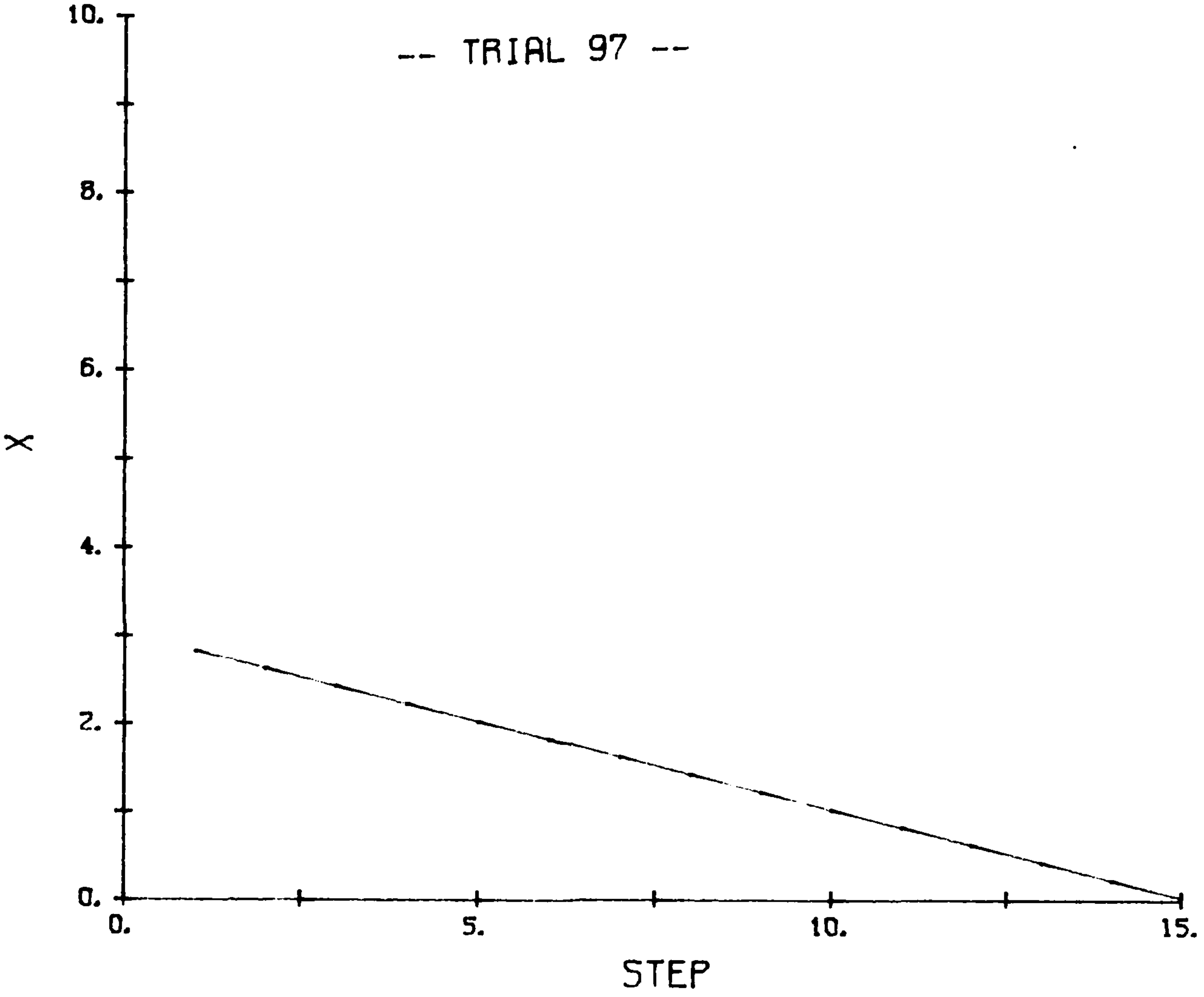


FIGURE 4.4. CONVERGENCE OF ESTIMATE (Δx = .2).

5
APPLICATION OF ASE-ACE TO AUTOFOCUS

One potential application of the ASE-ACE learning algorithms to synthetic aperture radar (SAR) that we would like to discuss in more detail is the autofocus problem.

The objective of a SAR system is to produce a high resolution image of some scene on the ground from an aircraft. A strong point target in the scene will look like a bright spot in the image. The width of the bright spot is a function of the basic resolution capability of the SAR and the amount of quadratic phase error in the system. One source of quadratic phase error is the motion compensation system. Position measurement errors that are a quadratic function of time cause quadratic phase errors.

Figure 5.1 illustrates the application of ASE-ACE to cancel the quadratic motion measurment error. The upper portion of Figure 5.1 represents the motion measurement chain of the motion compensation system, consisting of a motion sensor (inertial navigation unit) followed by two integrators. The motion sensor measures the translational acceleration along the radar line-of-sight. The acceleration measurement is integrated once to give a measure of line-of-sight velocity and then a second time to give a measure of line-of-sight position. Prior to starting the integration process, the integrators must be initialized to the best estimate of line-of-sight velocity and line-of-sight position.

The quadratic motion measurement error is caused by any bias in the acceleration measurement and any error in the velocity initial condition. An error in velocity will cause position to grow linearly with time and an acceleration bias will cause position to grow quadratically with time. The objective of the ASE-ACE is to keep the velocity and position measurements within the expected bounds over the aperture time of the radar. The ASE-ACE output is integrated to give a bias correction to the acceleration measurement out

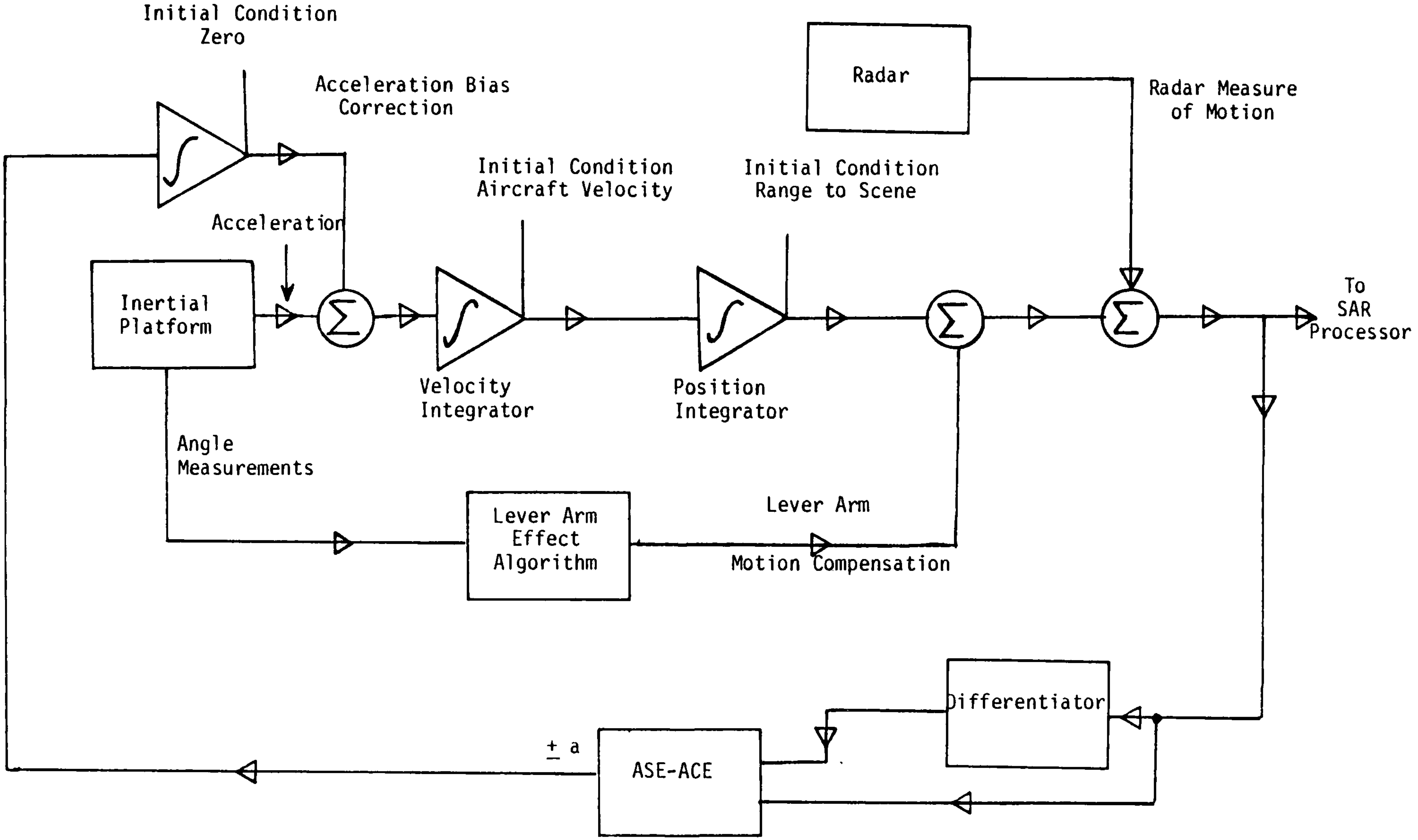


FIGURE 5.1. ASE-ACE APPLIED TO AUTOFOCUS PROBLEM.

of the motion sensor. If the bias integrator output equals the motion measure bias and is opposite in sign, the velocity and position integrators will stay within bounds.

There is an alternate approach that can be used to solve the autofocus problem using the ASE-ACE. Instead of giving the ASE-ACE the velocity and position motion measurements, the 3 dB and 15 dB IPR widths can be measured in the image processor and used as inputs to the ASE-ACE. The SAR aperture time window could be slid along in tim and a sequence made on the same poin target. The ASE-ACE would still drive the bias integrator in the motion measurement chain. This would give better performance but would be more difficult to implement.

6
DEFINING AND CHANGING BOXES

A different approach to defining the boxes used by the ASE-CE than implemented by Barto, et al.[1] has been implemented in this ASE-ACE algorithms studied by ERIM. This approach can be best illustrated by defining the four vectors $\underline{x}_i$, $\underline{\dot{x}}_i$, $\underline{\theta}_i$ and $\underline{\dot{\theta}}_i$ as shown below using $\underline{x}_i$ as an example

$$\underline{x}_i = \begin{bmatrix} \text{1 if cart-position is in X-region 1} \\ \text{1 if cart-position is in X-region 2} \\ \text{1 if cart-position is in X-region 3} \end{bmatrix}$$

The elements of $\underline{x}_i$ take only the values 0 and 1 and each element corresponds to one region. Only one element of $\underline{s}_i$ can be 1 at any given instant of time. A similar definition holds for $\underline{\dot{x}}$, $\underline{\dot{\theta}}_i$ and $\underline{\theta}_i$. With these preliminary definitions out of the way, the following vector $\underline{x}_i'$ can be defined

$$\underline{x}_i' = \begin{bmatrix} \underline{x}_i \\ \underline{\dot{x}}_i \\ \underline{\theta}_i \\ \underline{\dot{\theta}}_i \end{bmatrix}$$

This vector $\underline{x}_i'$ uniquely defines the state of the pole-on-a-cart to within the resolution of the selected quantization levels (region widths).

The vector $\underline{x}_i$ used by Barto can be obtained from the vector $\underline{x}_i$ by multiplying $\underline{x}_i'$ by a matrix $\underline{A}$ and selecting all the elements of the resulting vector to be zero except for the element that exactly equals 4. Each row of $\underline{A}$ has only 4 non-zero elements equal to 1 and all the rows are independent. A typical $\underline{A}$ matrix would have the form:

$$\underline{A} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

The matrix $\underline{A}$ combined with the limiting process can be viewed as a mapping from an m-dimensional space to an n-dimensional space where n is greater than m.

Consider now the problem of changing the sizes of the boxes by redefining the regions used for x, $\dot{x}$, θ and $\dot{\theta}$ during the process of running the ASE-ACE. It will be assumed that the change is always towards smaller boxes (i.e., more boxes). After this change, it is important to retain the information learned which is stored in the weighting vectors $\underline{v}_i$ and $\underline{w}_i$. To do this, the following vectors are defined.

$$\underline{v}_i' = \left[\underline{A}_1^t\underline{A}_1\right]^{-1}\underline{A}_1^t\underline{v}_i$$

$$\underline{w}_i' = \left[\underline{A}_1^t\underline{A}_1\right]^{-1}\underline{A}_1^t\underline{w}_i$$

The matrix $\underline{A}_1^t\underline{A}_1$ will always be invertible and $\left[\underline{A}_1^t\underline{A}_1\right]^{-1}\underline{A}^t$ defines a unique mapping of one vector into another vector.

At this point, it will be assumed that the change in box sizes is accomplished by cutting each region of x, $\dot{x}$, θ, $\dot{\theta}$ in half, thirds, fourths, etc. If they are cut in half, the vectors $\underline{v}_i'$ and $\underline{w}_i'$ can be doubled in size as shown below:

$$\underline{v}_i' \text{ (new)} = \begin{bmatrix} v_1' \\ v_1' \\ v_2' \\ v_2' \\ \cdot \\ \cdot \\ v_m' \\ v_m' \end{bmatrix} \qquad \underline{w}_i' \text{ (new)} = \begin{bmatrix} w_1' \\ w_1' \\ w_2' \\ w_2' \\ \cdot \\ \cdot \\ w_m' \\ w_m' \end{bmatrix}$$

The first element of $\underline{v}_i'$ becomes the first and second element of $\underline{v}_i'$ (new) and the second element of $\underline{v}_i'$ becomes the third and fourth element of $\underline{v}_i'$ (new), etc. In a similar manner, $\underline{w}_i'$ is redefined. Once this is done, then the starting values for v_i and w_i corresponding to the smaller boxes are:

$$\underline{v}_i \text{ (new)} = \underline{A}_2\underline{v}_i' \text{ (new)}$$

$$\underline{w}_i \text{ (new)} = \underline{A}_2 w_i' \text{ (new)}$$

Where $\underline{A}_1$ is the value of the A-matrix for the original boxes and $\underline{A}_2$ is the value of the A-matrix for the smaller boxes. This procedure will not work if the box sizes are changed by arbitrarily redefining the regions.

The objective of changing the size of the boxes during a run is to tighten up the control and keep the pole closer to zero. The larger boxes would be used initially to achieve control of the pole and keep it from exceeding the specified limits. The smaller boxes would be used in conjunction with penalties within the boxes to keep the pole as close to zero as possible and possibly to keep the cart as close to zero position as posible.

7
APPLICATION OF ASE-ACE TO A ROBOTICS TYPE OF PROBLEM

We will now describe a control problem which has some of the characteristics of a robotics problem which is suited to the application of an ASE-ACE type of controller. The problem is illustrated in Figure 7.1 which shows a mass at the end of a rod which rotates in a two-dimensional coordinate system. The length of the rod is variable and the rod is flexible. The objective of the problem is to move the mass M from one point in space to another point in space with minimum bending of the rod. There is very little system damping, so once bending is excited, it continues making it impossible to obtain the desired steady state conditions.

7.1 RIGID DYNAMICS

The controller for the system applies a fixed torque to rotate the rod. The direction of the torque can change, but not the magnitude. The angular acceleration of the rod, θ, is equal to:

$$\ddot{\theta} = \frac{T}{Mr^2}$$

$$r^2 = X_f^2 + Y_f^2$$

where X_f and Y_f are the desired final coordinatees of the mass. It is assumed that the rod is extended before being torqued. The force on the mass is $F = Mr\ddot{\theta}$, which can be rewritten as:

$$F = T/r$$

The force on the mass is directly proportional to the applied torque. The X, Y-coordinates of the mass at any instant of time are:

$$X = r \cos (\theta)$$

$$Y = R \sin (\theta)$$

Therefore, X and Y are nonlinear functions of θ.

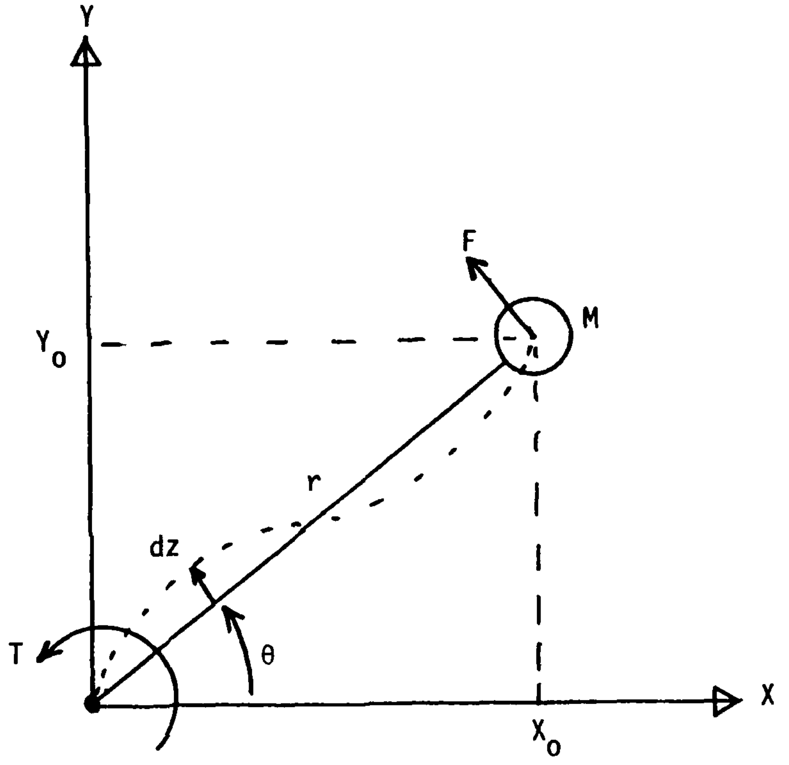


FIGURE 7.1. PROBE POSITIONING WITH BENDING.

7.2 BENDING

It will be assumed that bending can be modeled as a resonant circuit. The bending, dz, is equal to:

$$dz(s) = \frac{F(s)}{S^2 + 2\delta W_b S + W_b^2}$$

in terms of Laplace transforms. The bending model is, therefore, defined by the bending frequency, W_b, and the damping coefficient, δ. Once dz has been computed, the true values of X and Y become:

$$X = r \cos (\theta) - dz \sin (\theta)$$

$$Y = r \sin (\theta) + dz \cos (\theta)$$

Even if θ and r are constants, dz will not be constant so X and Y will not be constant.

7.3 STATE SPACE DEFINITION

For this example problem, it is recommended that the state space consist of θ, $\dot{\theta}$, dz, and r. The variables θ, θ, and r are used since they are the natural quantities expected to be used by the control law. The variable dz is added since it will be used to determine failure. Failure will be defined as dz exceeding positive or negative limits.

The sample time used will have to be a function of W_b so the bending motion will be adequately sampled.

7.4 CONCLUDING REMARKS

The proposed example is a simple learning problem, but it is typical of certain practical design problems such as the cargo boom on the space shuttle. The rod in Figure 7.1 could be the cargo boom and the mass could be a satellite. The example problem would then reflect the problem of placing a satellite into orbit without excessive residual motion. The same problem could also represent a cargo boom on a ship unloading cargo. The problem, however, is simple enough so that it can be easily simulated and programmed into the existing ASE-ACE computer code.

8
CONCLUSIONS AND RECOMMENDATIONS

It has been demonstrated that the ASE-ACE adaptive algorithm of Barto can be used effectively in a learning mode to control a fairly difficult mechanical system.

It was also shown that the ASE-ACE controller can be used to minimize an arbitrary function. Since a large number of engineering problems can be viewed from the perspective of minimizing some performance function, it follows that the ASE-ACE adaptive/learning algorithm may find wide engineering application.

It is suggested that two specific applications be examined in the continuing study: (a) the SAR autofocus problem, and (b) image-matching, which is a more difficult problem as it involves two-dimensional performance functions.

Study of the learning characteristics of the ASE-ACE should continue, however, in order to fully understand the subtleties of the algorithm. Specifically, the effect of the size of the state space on performance and the possibility of using some punishment when the system approaches failure to improve performance should be investigated.

APPENDIX

The dynamic behavior of the cart-pole system is described by the following non-linear differential equations which were used in our simulation:

$$\ddot{\theta} = \frac{g \sin \theta + \cos \theta \left[\dfrac{-F - m\ell\theta^2 \sin \theta + \mu_c \text{ sign } (\dot{x})}{m_c + m} \right] - \dfrac{\mu_p \dot{\theta}}{m\ell}}{\ell \left[\dfrac{4}{3} - \dfrac{m \cos^2 \theta}{m_c + m} \right]}$$

$$x = \frac{F + m\ell \left[\dot{\theta}^2 \sin \theta - \theta \cos \theta \right] - \mu_c \text{ sign } (\dot{x})}{m_c + m}$$

where $g = 9.8$ m/sec^2, acceleration due to gravity,

$m_c = 1.0$ Kg, mass of cart,

$m = 0.1$ Kg, mass of pole,

$\ell = 10$ m, half pole length,

$\mu_c = 0.01$, coefficient of friction of cart on track,

$\mu_p = 0.001$, coefficient of friction of pole on cart, and

$F = \pm 10.0$, newtons, force applied to carts center of mass at time t.

The equations were solved by numerical approximation using Euler's method with a time step equal to or less than the sampling period.

REFERENCES

1. Klopf, A.H., The Hedonistic Neuron, A Theory of Memory, Learning and Intelligence, Hemisphere Publishing Corporation, 1982.

2. Barto, A.G., and R.S. Sutton, Goal Seeking Components for Adaptive Intelligence: An Initial Assessment, Technical Report AFWAL-TR-81-1070, April 1981.

3. Barto, A.G., R.S. Sutton, and C.W. Anderson, Neuron-Like Adaptive Elements that can Solve Difficult Learning Control Problems, Computer and Information Science Department, University of Massachusetts, Report 82-20.

4. Athans, M. and P. Flab, Optimal Control, McGraw-Hill Book Company, 1966.