01 May 1984

The Optimal Trajectory Modelling of Robot Manipulators

Mary Claire Miller

Chung You Ho
Missouri University of Science and Technology

Arlan R. Dekock
Missouri University of Science and Technology, adekock@mst.edu

THE OPTIMAL TRAJECTORY MODELLING OF ROBOT MANIPULATORS

Mary Claire Miller*, Chung Y. Ho,
and Arlan R. DeKock

CSc-84-4

Department of Computer Science

University of Missouri-Rolla

Rolla, MO 65401 (314)341-4491

*This report is substantially the M.S. thesis of the first author, completed, May, 1984.

ABSTRACT

Greater robot capability can be achieved through the use of robot manipulator control systems. Crucial to the success of these control systems is the optimal trajectory modelling of the path traced by the end-effector. To create this optimal path the utilization of B-Spline curve functions will be investigated, and compared to Cubic Spline curve functions.

ACKNOWLEDGEMENTS

The author wishes to express her appreciation to Dr. C. Y. Ho for his guidance and support throughout the writing of this thesis. She would like to acknowledge Dr. Arlan DeKock and Dr. Ray Kluczny for their efforts as members of the thesis committee. The author also wishes to thank her fiance, Mark Bolten, for his support and encouragement.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

I. INTRODUCTION

Robots have advanced from manually controlled teleoperators to the computer-controlled manipulators seen in today's factories. This level of control sophistication permits robots to be used in such applications as die casting, spot welding, arc welding, investment casting, forging, press work, spray painting, plastic molding, foundry practice, machine tool loading, heat treatment, deburring metal parts, palletizing, brick manufacture, glass manufacture, and many more applications. The use of robots in these applications has been made possible through advancements in the areas of kinematics, dynamics, servoing, computer controls, modelling, and trajectory calculation.

Industrial robots have two major components: the moving parts (the "arm", the "wrist", and the "hand") and the control system.

The arm is considered to be the first three links of a robot and determines the location of the robot within the working area. The wrist is made up of the next few links and describes the orientation of the robots end effector. The hand, or end effector, is a device designed according to the task to be accomplished.

The control system for a robot may range from simple mechanical stops to computerized controls. The methodology employed allows the robot manipulator to trace a limited path, follow a heuristic path defined only at discrete points, or complete a continuous (smooth) path defined by the operator.

Conventional analog servos, sometimes called hardware servos, may be used to provide complete control of the robot manipulator. However, difficulties arise due to this approach. Analog servos are susceptible to variations in kinematic configurations and to payload changes which may result in structural resonances. The limited sequence robots controlled by hardware servos are the least sophisticated. They are often referred to as "bang-bang" machines. The end positions of the robot path are specified and the robot follows a non-controlled trajectory between points A and B. Their drive systems may be pneumatic, electrical, or hydraulic. These robots are usually small and useful for die-casting, press loading, and plastics molding. These robots are also tedious to program and modify.

An alternative to the use of hardware servos in robot manipulators is to design control systems using software servos. The software servo permits a large degree of flexibility in designing control systems, and may reduce the overall cost of designing control strategies. Two possible strategies are shown in robots with point-to-point path control and continuous path control.

Point-to-point robots are usually led through a task by an operator using a teach pendant. As the task is being simulated or rehearsed, the programmer specifies discrete points along the path to be followed. This method of programming the robot is relatively fast and easy. Additional capabilities of this method permits random access to multiple control programs and subroutines stored in a computer. An obstacle to the use of this methodology concerns the definition of discrete points along the manipulator trajectory. As modifications are made to the path of the robot, the operator must interact with the control system. It is often difficult for the operator to modify the programmed positions.

Robots with continuous path control are led through the task as with point-to-point robots. However the data points are collected by the controller on a time basis, perhaps in the range of 60 to 80 Hz. This requires much more memory, many times using tape or disk storage. There is no noticeable change in speed as with the point-to-point method, thus the name continuous path control. The speed of the manipulator is varied by the rate at which the data points are fed to the manipulator. This capability is useful in areas such as seam welding, spray painting, polishing, and grinding.

Each of these methods requires taking the robot off the production line to teach it the next task via a teach pendant or limit switches. This factor alone can lengthen the pay-back time of the robot equipment. To alleviate this problem extensive work is being performed on the development of off-line programming languages. These languages will allow changes to trajectory control programs while permitting the robot to continue work on the production line. These new or modified programs can be executed by the robot controller when production loads dictate.

Off-line robot programming languages commercially available include: AL, developed by Stanford University and Robot Technology, Inc.; AML, developed by IBM Corporation; HELP, developed by General Electric Corporation; MCL, developed by McDonnell Douglas Corporation for the United States Air Force; RAIL, developed by Automatix, Inc.; and VAL, developed by Westinghouse-Unimation, Inc. These languages still have difficulties that prevent widespread use at the present time, but are promising alternatives for the future.

Three areas to be addressed in developing software servos for any computer controlled robot manipulator are:

• Kinematics. Given the position and orientation of the end effector in cartesian coordinates and orientational vectors, the robot controller will determine the corresponding manipulator joint positions. This information enables a program to transform an end effector trajectory defined in cartesian space to a trajectory in joint space. Analytic solutions for this problem are discussed in several research papers.[1,2]

• Statics. Given the payload or force exerted by the end effector of the manipulator, the controller will determine the corresponding force developed at the manipulator joint actuators. The transformation from forces in cartesian space to forces in joint space is linear. The joint forces are the products of the force resolution matrix and the vector force in cartesian space[3] where the force resolution matrix is the transpose of the Jacobian matrix. The determination of the Jacobian matrix of a manipulator can be found using techniques in the literature.[4,5]

• Dynamics. Given the position, velocity, and acceleration of the end effector, the controller will determine the corresponding forces or torques developed at the manipulator actuators, based on Lagrangian mechanics. The Hollerbach research[6] discusses a solution to the dynamics problem.

Robot manipulator control systems may utilize feedforward control to prevent a deviation of the end effectors' path from becoming so large as to negate effective corrective action. Using solutions from the three areas outlined above, a feedforward control system can be constructed. Feedforward systems can provide the robot controller with a method to determine the optimal trajectory path to be traced by the end effector. Optimization of these events lies in the optimal trajectory planning of the end effector.

In this paper, we develop a path for the end effector to follow, with the knowledge of the fixture geometry to avoid possible collision and the knowledge of the operational motion. This leads us to establish a sequence of crucial control knot points in terms of three-dimensional cartesian parametric coordinates $(x(t), y(t), z(t))$. From these knots, a trajectory for the robot may be generated. This trajectory yields information about position, velocity, and acceleration. It is necessary to supply instantaneous information about position, velocity, and acceleration of the end effector along the path for the feedforward control system, because the controller must know the exact position of all robot joints at all times.

Mathematically, the trajectory should be continuous tnrough the second derivative providing continuous position, velocity, and acceleration functions for the end effector path. We define the boundary conditions of the trajectory, specifying velocity and acceleration at the trajectory endpoints to be zero. The trajectory is generated by using a B-spline function shaped by the sequence of control knot points.

II. REVIEW OF LITERATURE

This thesis is particularly indebted to the research of Ho and Cook Cook[7] in the application of spline functions to robot trajectory generation. Their work describes how cubic spline functions may represent both the trajectory of the tool-tip of the manipulator hand in three-space and the trajectories of the manipulator joints in time. Mathematically, the cubic spline curve is continuous through the second derivative providing continuous velocity and acceleration functions for the end-effector traveling along its trajectory path. The boundary conditions for liftoff and setdown of the end-effector are accommodated. Their approach offers flexibility, computational efficiency, and a compact representation of the path.

Schoenberg introduced B-Splines in his paper On Spline Distributions and their Limits: The Polya Distributions.[8] Substantial research in the area of B-Splines has been conducted by DeBoor, Mansfield, Cox, and Riesenfield. DeBoor and others have proposed a recursive definition of the B-Spline basis[9] rather than the divided difference formulation formulation of the B-Spline basis suggested by Schoenberg. The recursive evaluation of B-Splines is well conditioned, efficient and needs no adjustments in the case of coincident knots. The condition of the B-Spline basis will increase exponentially with the order. Riesenfield contributed a comparison of B-Spline and Bezier curves for graphics applications in his doctoral dissertation.[10]

III. RESULTS

A B-Spline curve is a parametric B-Spline approximation to the polygon defined by the A_i vertices. On the following pages the differentiation of the B-Spline function definition will be presented. This is significant in the fact that the velocity and acceleration vectors must provide 'take-off' and 'set-down' values of zero. As the derivation suggests, B-Splines can be used effectively to control the robot manipulator in its movements throughout three-space.

Figures 1 through 18 show a sample B-Spline generated trajectory path. They show a profile in three-space of position, velocity, and acceleration. Also shown is the position of the tool-tip along each axis at any point in the path.

POSITION VECTOR

$$P(t) = \sum_{i=0}^{n} A_i \, N_{i,k}(t)$$

where

n is the number of defining polygon vertices minus one

k is the order of the curve

t is the parameter, varying from 0 to t_{max}

A_i are the n+1 defining polygon vertices

$$N_{i,k}(t) = ((t-x_i)N_{i,k-1}(t)) \,/\, (x_{i+k-1}-x_i)$$

$$+ ((x_{i+k}-t)N_{i+1,k-1}(t)) \,/\, (x_{i+k}-x_{i+1})$$

$$N_{i,1}(t) = \{ \; 1 \text{ if } x_i <= t < x_{i+1}$$

$$\{ \; 0 \text{ otherwise}$$

x_i are the elements of a knot vector

POSITION VECTOR - The j^{th} Derivative:

$$P^{(j)}(t) = (k-1)\ldots(k-j)\Sigma A_i^{(j)} N_{i,k-j}(t)$$

where

$$A_i^{(0)} = A_i$$

$$A_i^{(j)} = (A_i^{(j-1)} - A_{i-1}^{(j-1)}) / (x_{i+k-j} - x_i) \quad ; j > 0$$

$$P^{(1)}(t) = (k-1)\Sigma A_i^{(1)} N_{i,k-1}(t)$$

where

$$A_i^{(1)} = (A_i - A_{i-1}) / (x_{i+k-1} - x_i)$$

$$P^{(2)}(t) = (k-1)(k-2)\Sigma A_i^{(2)} N_{i,k-2}(t)$$

where

$$A_i^{(2)} = (A_i^{(1)} - A_{i-1}^{(1)}) / (x_{i+k-2} - x_i)$$

VELOCITY VECTOR - At t = 0:

$$P^{(1)}(t) = (k-1)\Sigma A_i^{(1)} N_{i,k-1}(t)$$

where

$$A_i^{(1)} = (A_i - A_{i-1}) / (x_{i+k-j} - x_i)$$

$$\lim_{t\to 0} P^{(1)}(t) = \lim_{t\to 0} (k-1)\Sigma A_i^{(1)} N_{i,k-1}(t)$$

$$\lim_{t\to 0} P^{(1)}(t) = (k-1)\Sigma A_i^{(1)} \lim_{t\to 0} N_{i,k-1}(t)$$

where

$$\lim_{t\to 0} N_{i,k-1}(t) = (\lim_{t\to 0} t - x_i)\lim_{t\to 0} N_{i,k-1}(t)/(x_{i+k-1} - x_i) + (x_{i+k} - \lim_{t\to 0} t)\lim_{t\to 0} N_{i,k-1}(t)/(x_{i+k} - x_{i+1})$$

Note:

$$N_{i,1}(t) = \{ \ 1 \text{ if } x_i <= t < x_{i+1}$$

$$\{ \ 0 \text{ otherwise}$$

Therefore, if knots of multiplicity k are chosen at the beginning of the knot set, the velocity vector will equal zero.

$$\lim_{t\to 0} P^{(1)}(t) = (k-1)\Sigma A_i^{(1)}((0/(x_{i,k-1} - x_i) + (0/(x_{i+k} x_{i+1}))$$

$$\lim_{t\to 0} P^{(1)}(t) = 0$$

VELOCITY VECTOR - At $t = t_{max}$:

$$P^{(1)}(t) = (k-1)\Sigma A_i^{(1)} N_{i,k-1}(t)$$

where

$$A_i^{(1)} = (A_i - A_{i-1}) / (x_{i+k-j} - x_i)$$

$$\lim_{t \to T} P^{(1)}(t) = \lim_{t \to T} (k-1)\Sigma A_i^{(1)} N_{i,k-1}(t) \text{ ; where } T = t_{max}$$

$$\lim_{t \to T} P^{(1)}(t) = (k-1)\Sigma A_i^{(1)} \lim_{t \to T} N_{i,k-1}(t)$$

where

$$\lim_{t \to T} N_{i,k-1}(t) = (\lim_{t \to T} t - x_i)\lim_{t \to T} N_{i,k-1}(t)/(x_{i+k-1} - x_i) + (x_{i+k} - \lim_{t \to T} t)\lim_{t \to T} N_{i,k-1}(t)/(x_{i+k} - x_{i+1})$$

Note:

$$N_{i,1}(t) = \{ \ 1 \text{ if } x_i <= t < x_{i+1}$$

$$\{ \ 0 \text{ otherwise}$$

Therefore, if knots of multiplicity k are chosen at the end of the knot set, the velocity vector will equal zero.

$$\lim_{t \to T} P^{(1)}(t) = (k-1)\Sigma A_i^{(1)}((0/(x_{i,k-1} - x_i) + (0/(x_{i+k} x_{i+1}))$$

$$\lim_{t \to T} P^{(1)}(t) = 0$$

<u>ACCELERATION VECTOR</u> - At t = 0:

$$P^{(2)}(t) = (k-1)(k-2)\Sigma A_i^{(2)} N_{i,k-2}(t)$$

where

$$A_i^{(2)} = (A_i^{(1)} - A_{i-1}^{(1)}) / (x_{i+k-2} - x_i)$$

$$\lim_{t\to 0} P^{(2)}(t) = \lim_{t\to 0} (k-1)(k-2)\Sigma A_i^{(2)} N_{i,k-2}(t)$$

$$\lim_{t\to 0} P^{(2)}(t) = (k-1)(k-2)\Sigma A_i^{(2)} \lim_{t\to 0} N_{i,k-2}(t)$$

where

$$\lim_{t\to 0} N_{i,k-2}(t) = (\lim_{t\to 0} t - x_i)\lim_{t\to 0} N_{i,k-2}(t)/(x_{i+k-2} - x_i) + (x_{i+k} - \lim_{t\to 0} t)\lim_{t\to 0} N_{i,k-2}(t)/(x_{i+k} - x_{i+1})$$

Note:

$$N_{i,1}(t) = \begin{cases} 1 \text{ if } x_i <= t < x_{i+1} \\ 0 \text{ otherwise} \end{cases}$$

Therefore, if knots of multiplicity k are chosen at the beginning of the knot set, the acceleration vector will equal zero.

$$\lim_{t\to 0} P^{(2)}(t) = (k-1)\Sigma A_i^{(2)}((0/(x_{i,k-1} - x_i) + (0/(x_{i+k} x_{i+1}))$$

$$\lim_{t\to 0} P^{(2)}(t) = 0$$

ACCELERATION VECTOR - At $t = t_{max}$:

$$P^{(2)}(t) = (k-1)(k-2)\Sigma A_i^{(2)} N_{i,k-2}(t)$$

where

$$A_i^{(2)} = (A_i^{(1)} - A_{i-1}^{(1)}) / (x_{i+k-2} - x_i)$$

$$\lim_{t \to T} P^{(2)}(t) = \lim_{t \to T} (k-1)(k-2)\Sigma A_i^{(2)} N_{i,k-2}(t) \ ; \ T = t_{max}$$

$$\lim_{t \to T} P^{(2)}(t) = (k-1)(k-2)\Sigma A_i^{(2)} \lim_{t \to T} N_{i,k-2}(t)$$

where

$$\lim_{t \to T} N_{i,k-2}(t) = (\lim_{t \to T} t - x_i) \lim_{t \to T} N_{i,k-2}(t)/(x_{i+k-2} - x_i) + (x_{i+k} - \lim_{t \to T} t) \lim_{t \to T} N_{i,k-2}(t)/(x_{i+k} - x_{i+1})$$

Note:

$$N_{i,1}(t) = \begin{cases} 1 \text{ if } x_i <= t < x_{i+1} \\ 0 \text{ otherwise} \end{cases}$$

Therefore, if knots of multiplicity k are chosen at the end of the knot set, the acceleration vector will equal zero.

$$\lim_{t \to T} P^{(2)}(t) = (k-1)\Sigma A_i^{(2)}((0/(x_{i,k-1} - x_i) + (0/(x_{i+k} x_{i+1}))$$

$$\lim_{t \to T} P^{(2)}(t) = 0$$

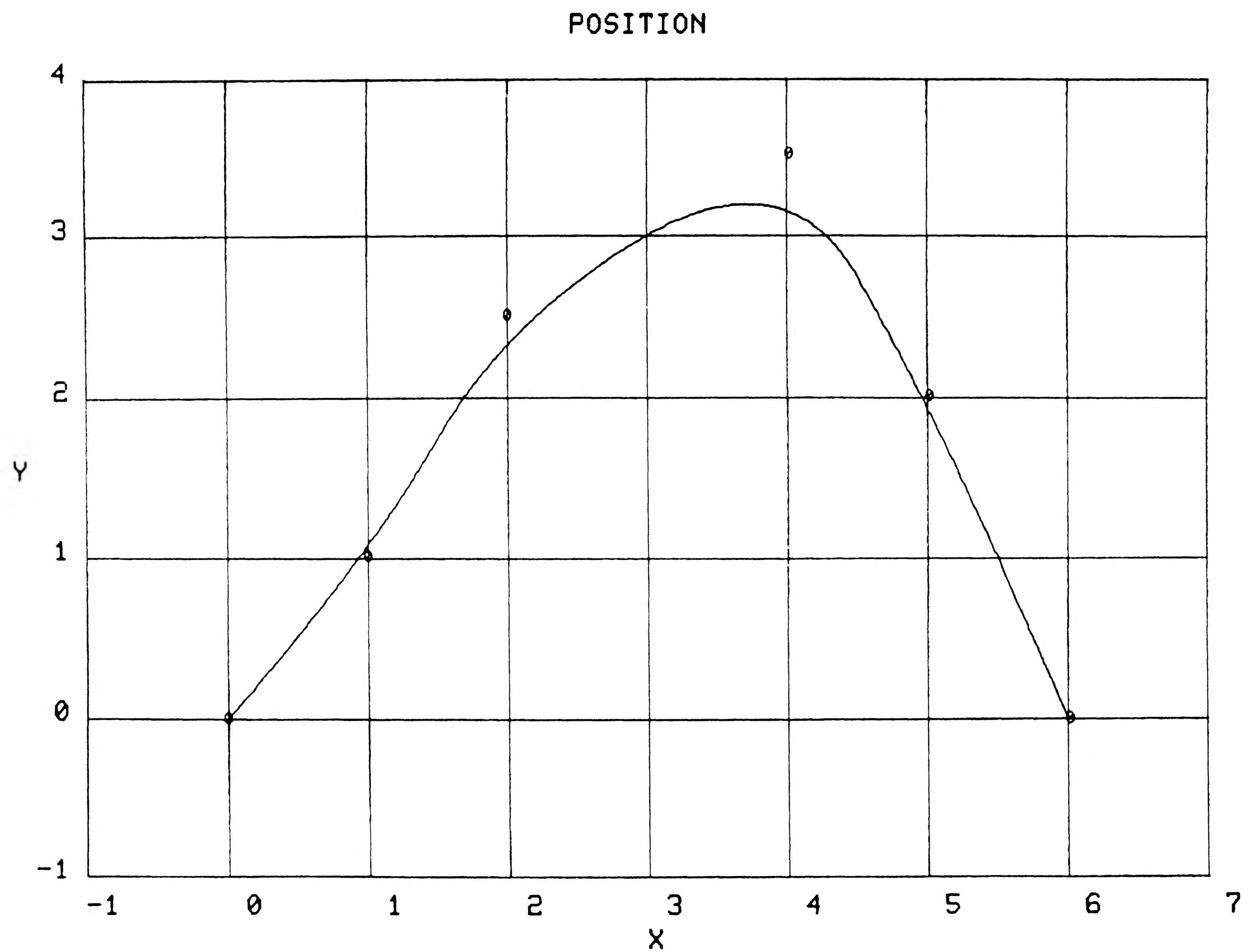

FIGURE 1. POSITION: Y vs X

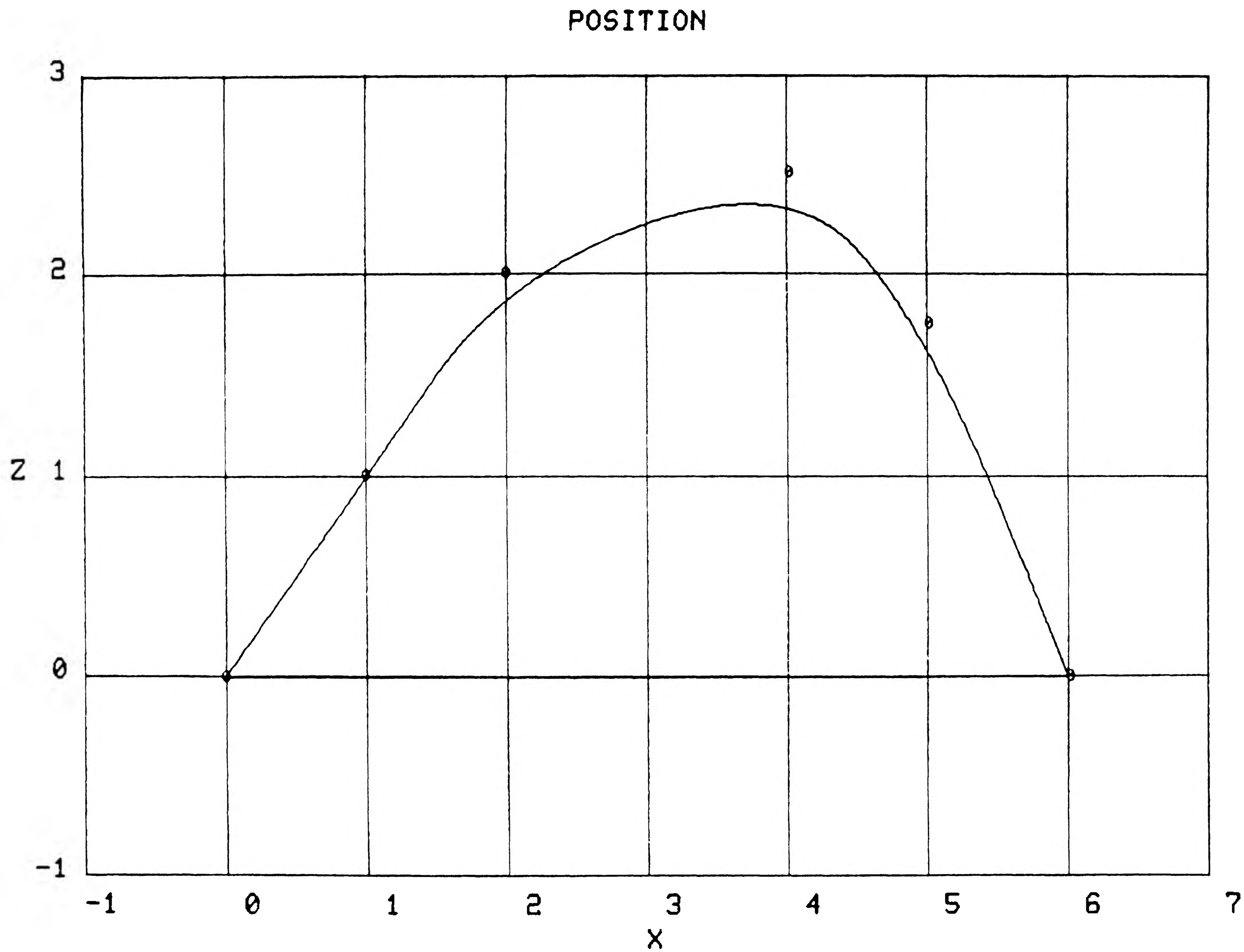

FIGURE 2. POSITION: Z vs X

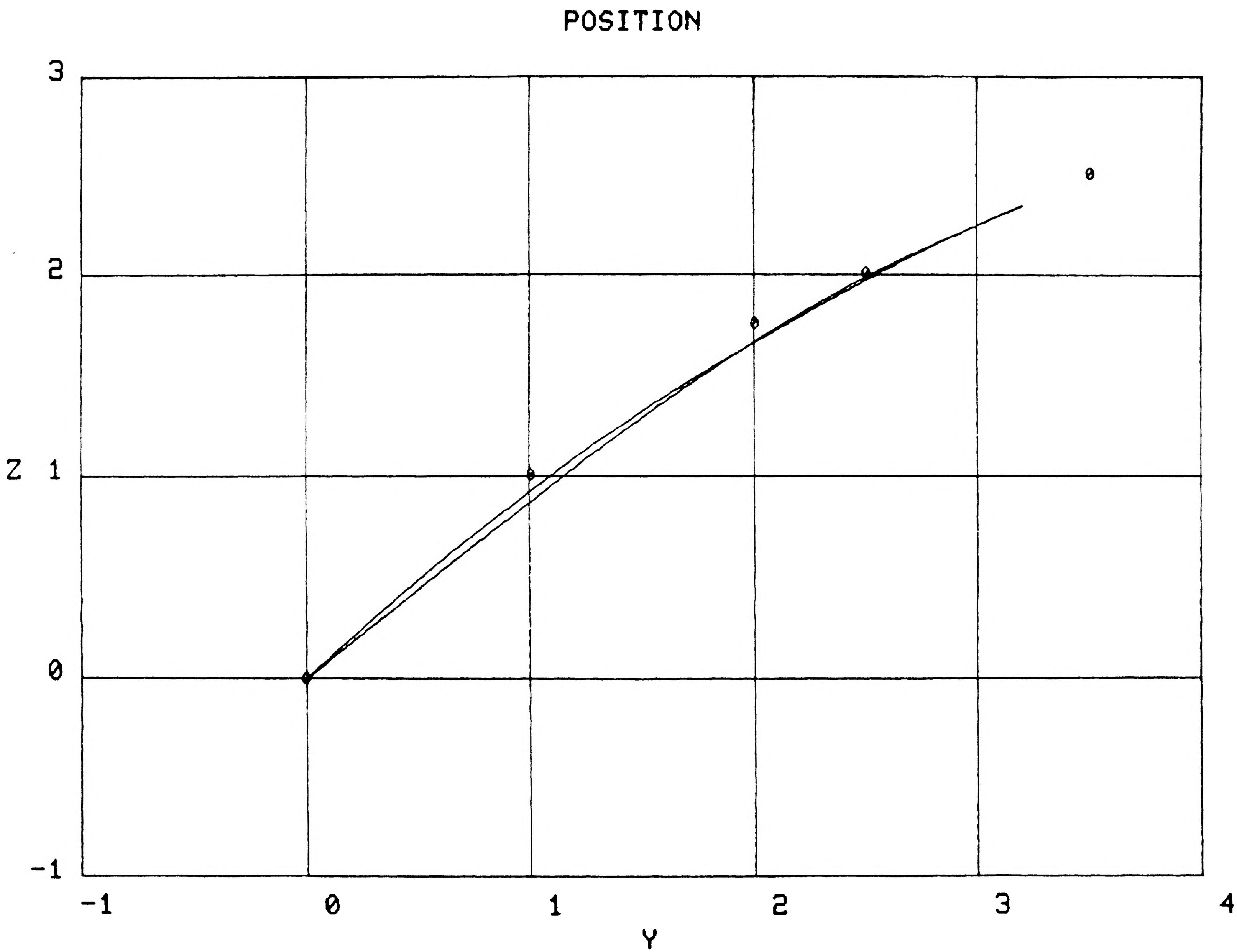

FIGURE 3. POSITION: Z vs Y

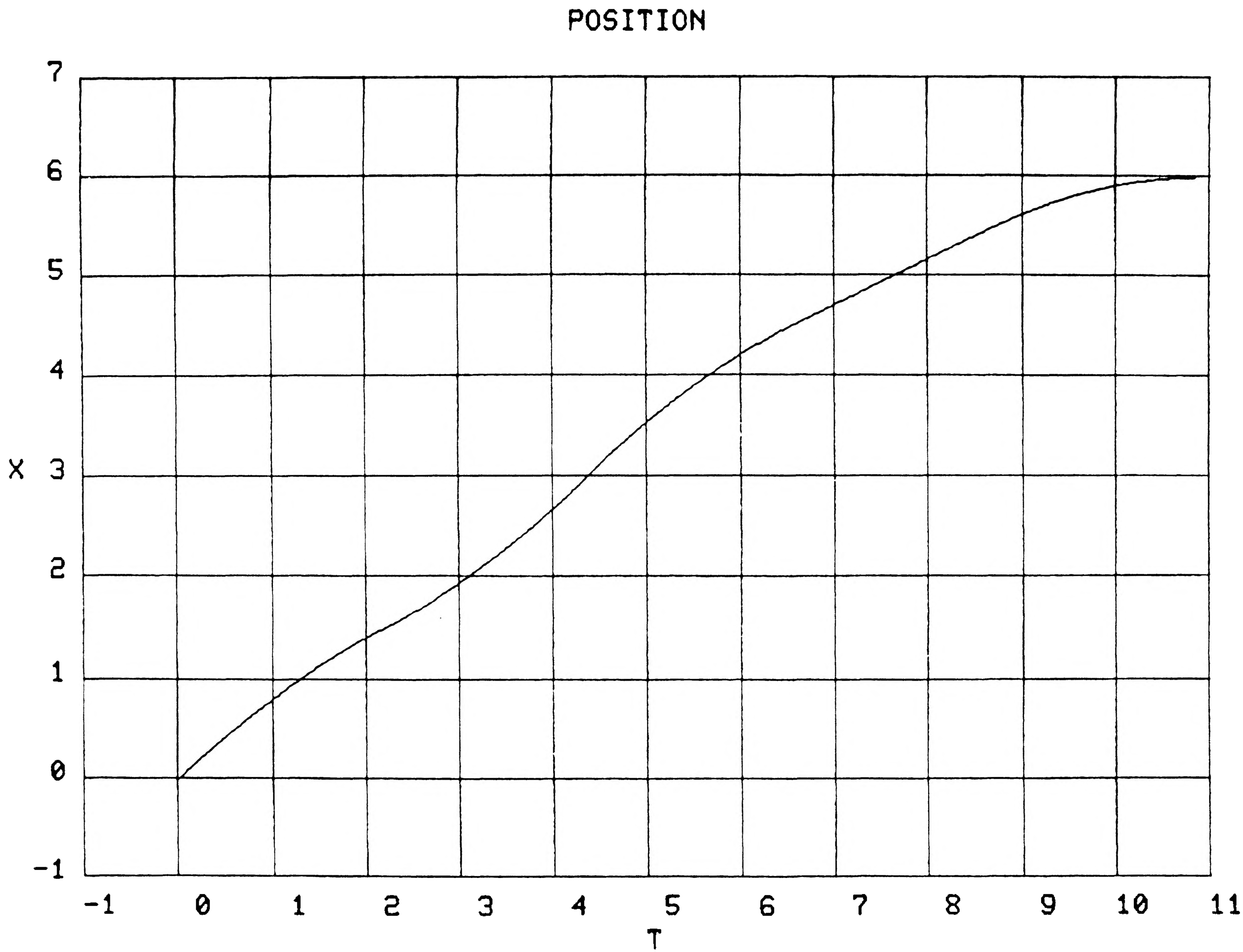

FIGURE 4. POSITION: X vs T

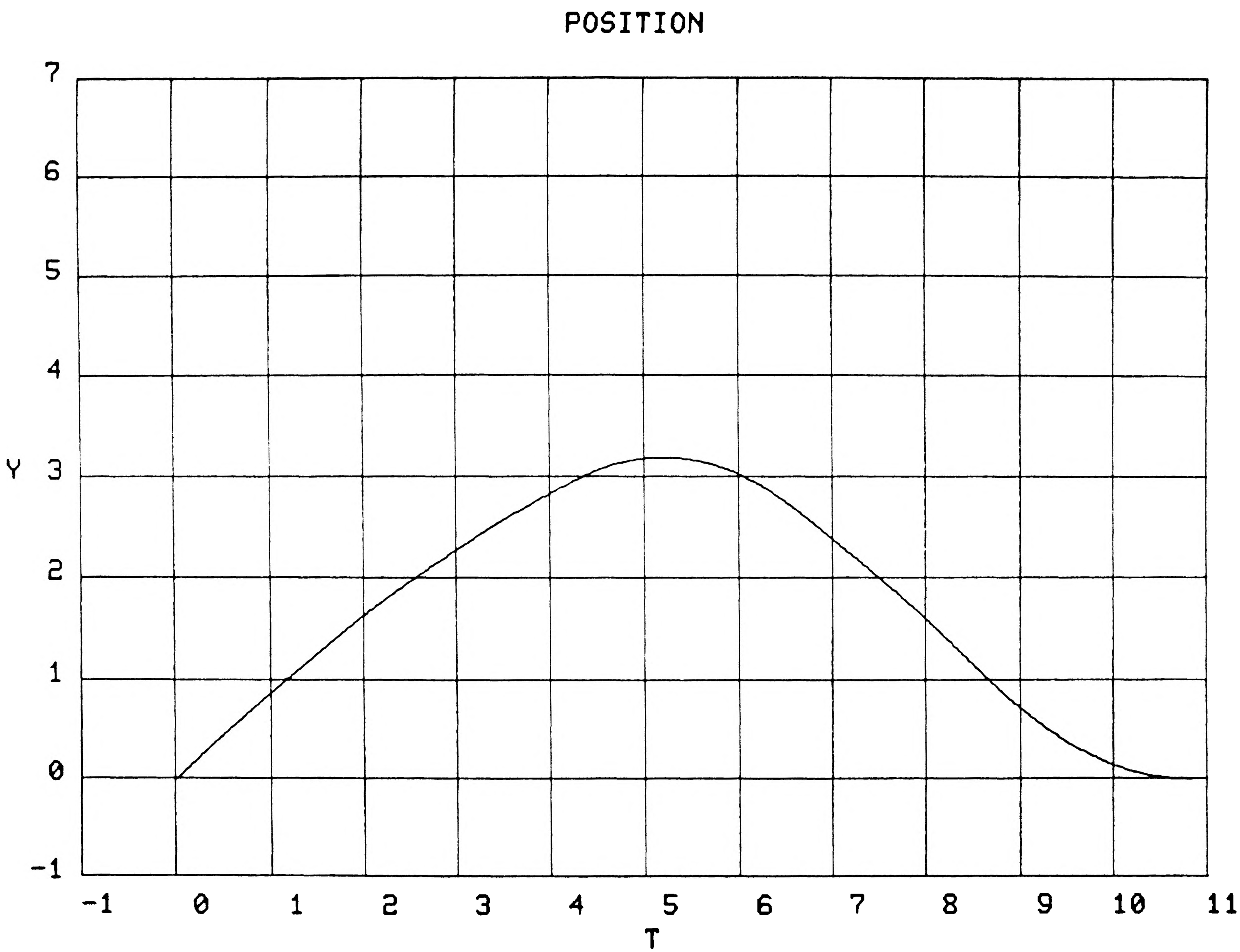

FIGURE 5. POSITION: Yvs T

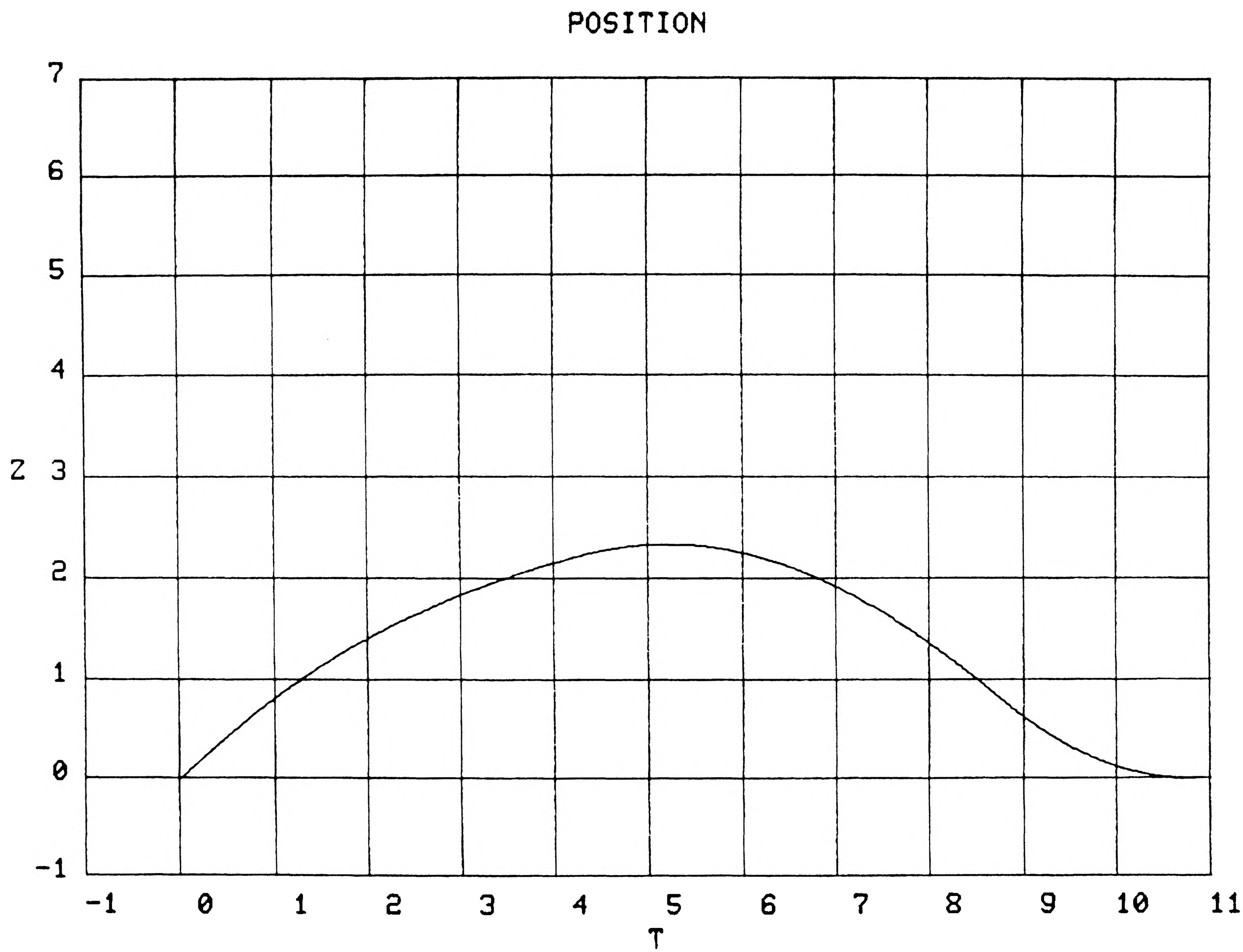

FIGURE 6. POSITION: Z vs T

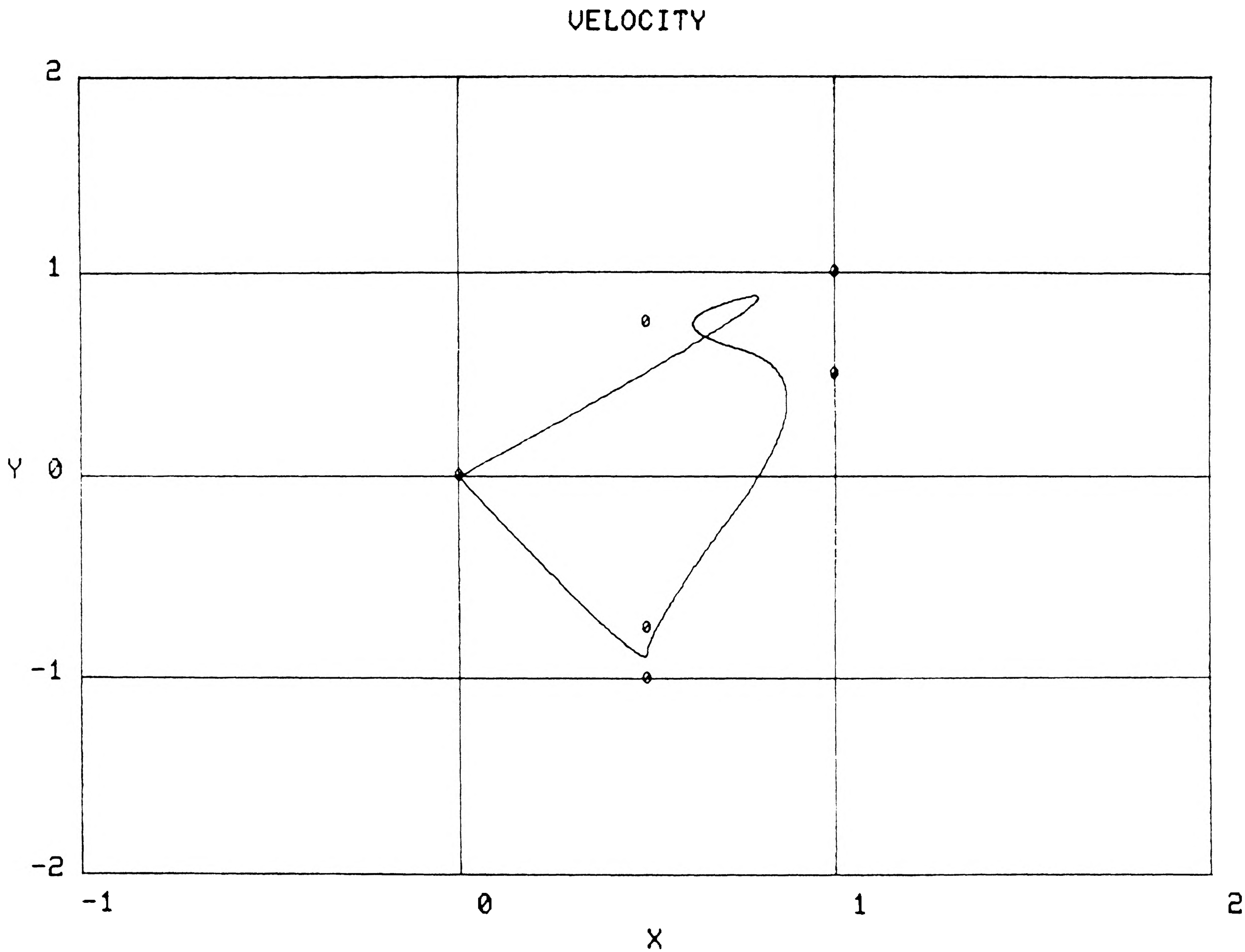

FIGURE 7. VELOCITY: Y vs X

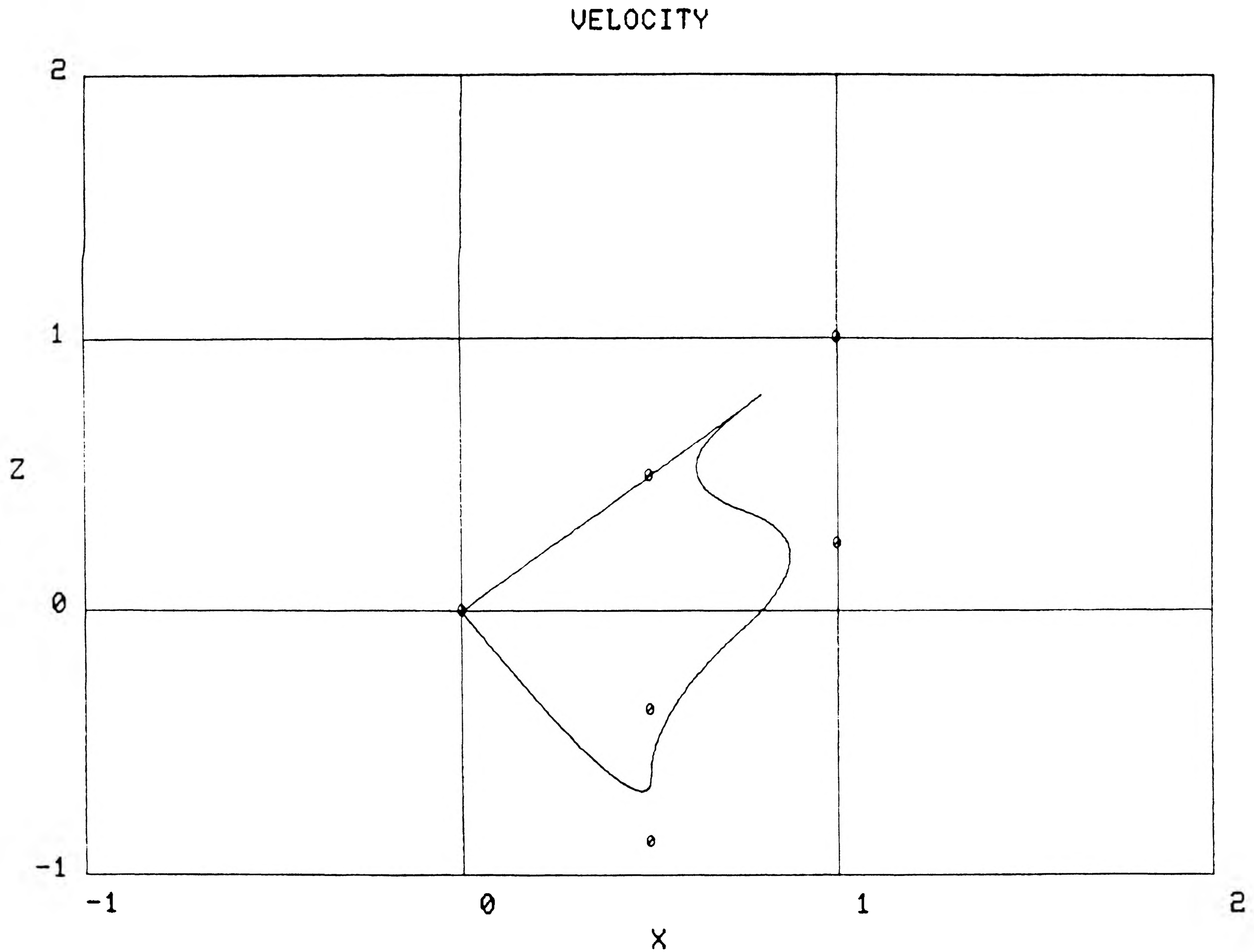

FIGURE 8. VELOCITY: Z vs X

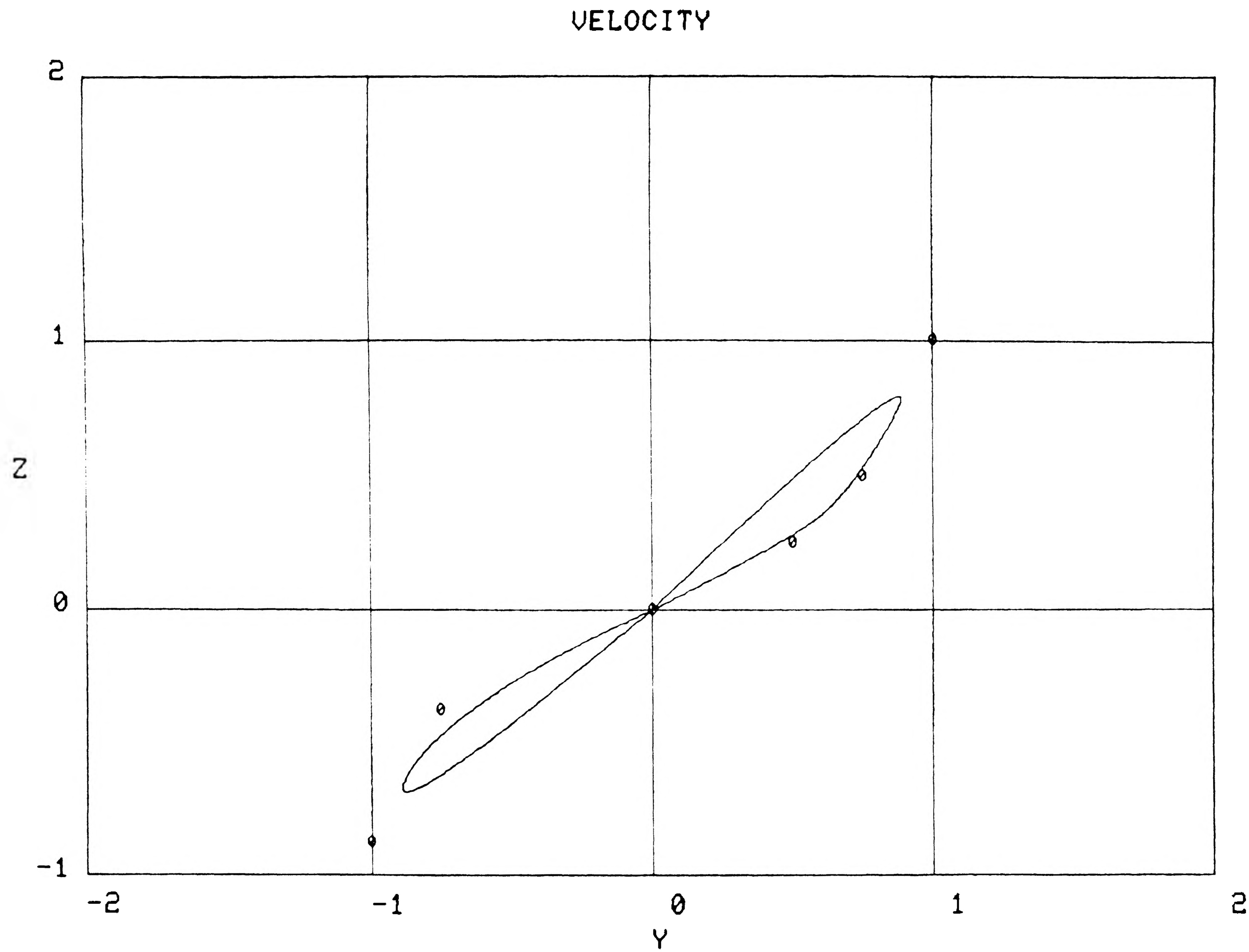

FIGURE 9. VELOCITY: Z vs Y

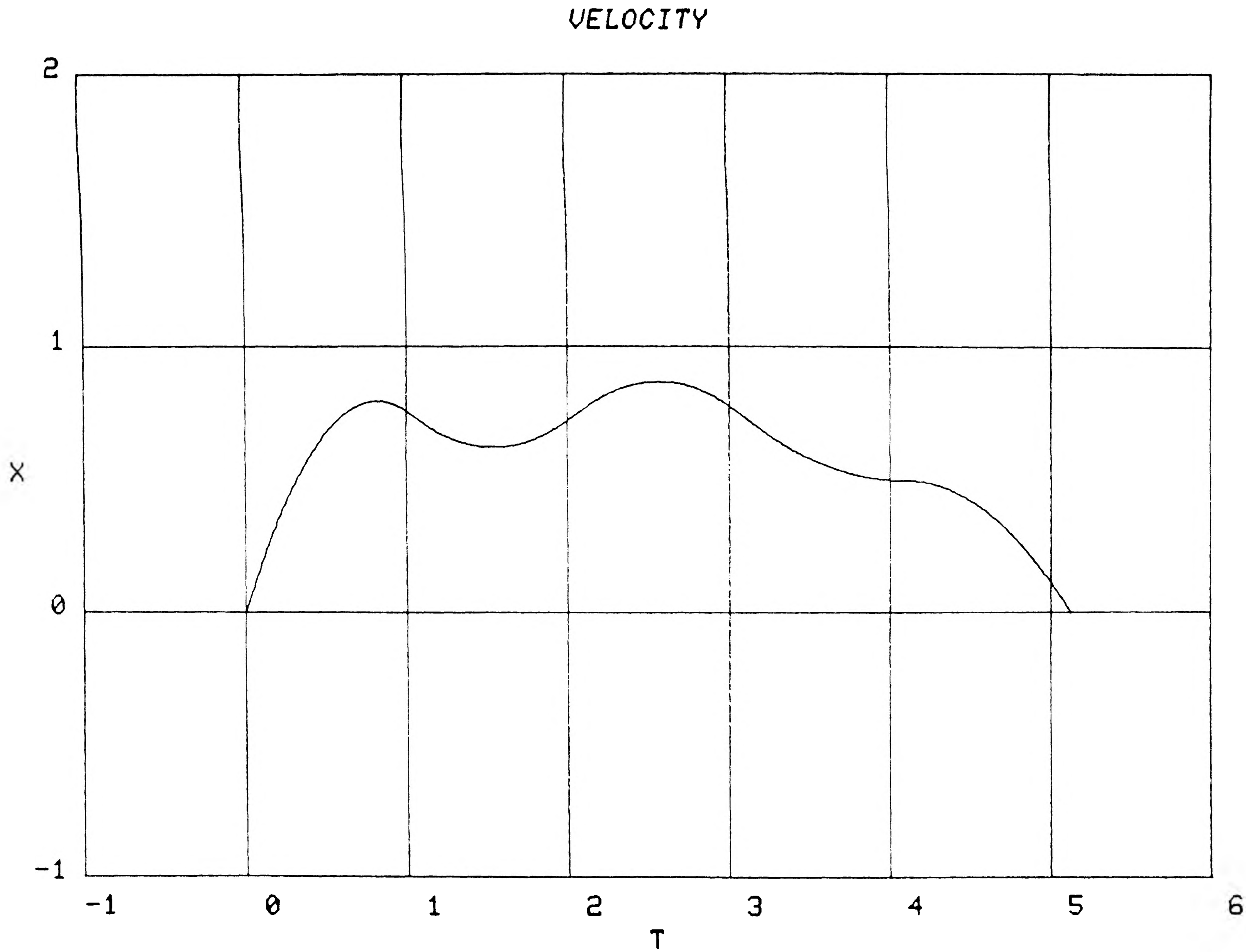

FIGURE 10. VELOCITY: X vs T

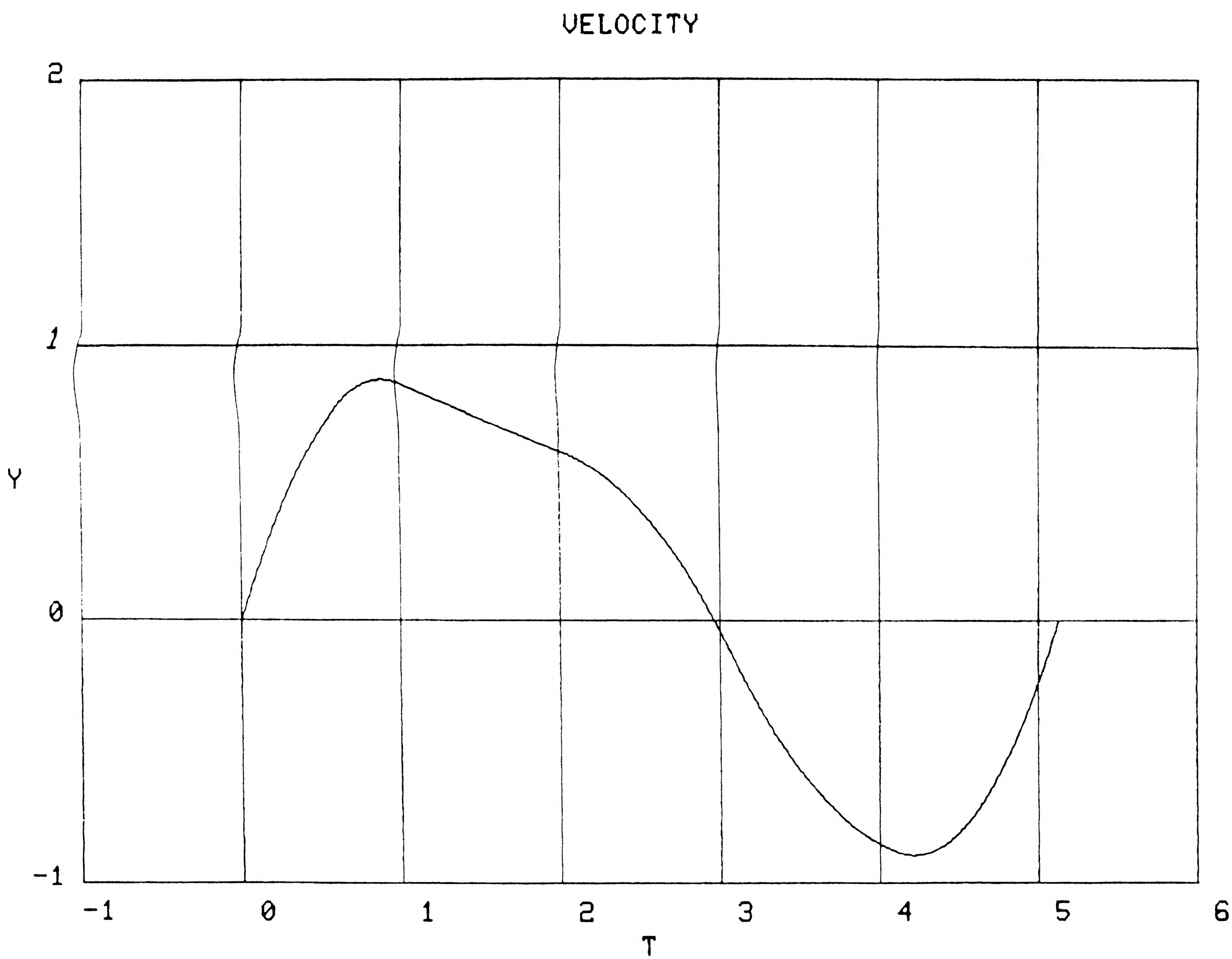

FIGURE 11. VELOCITY: Y vs T

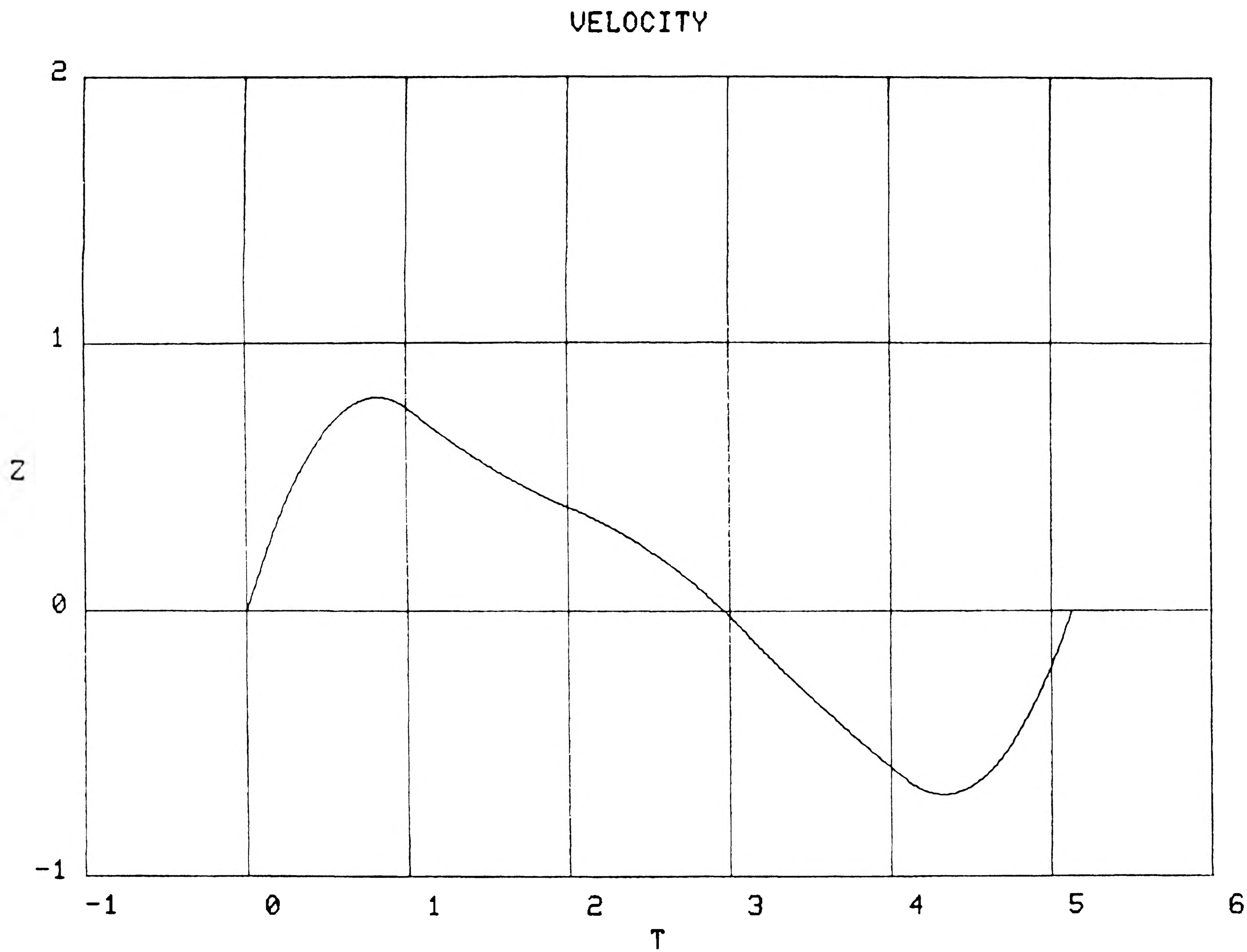

FIGURE 12. VELOCITY: Z vs T

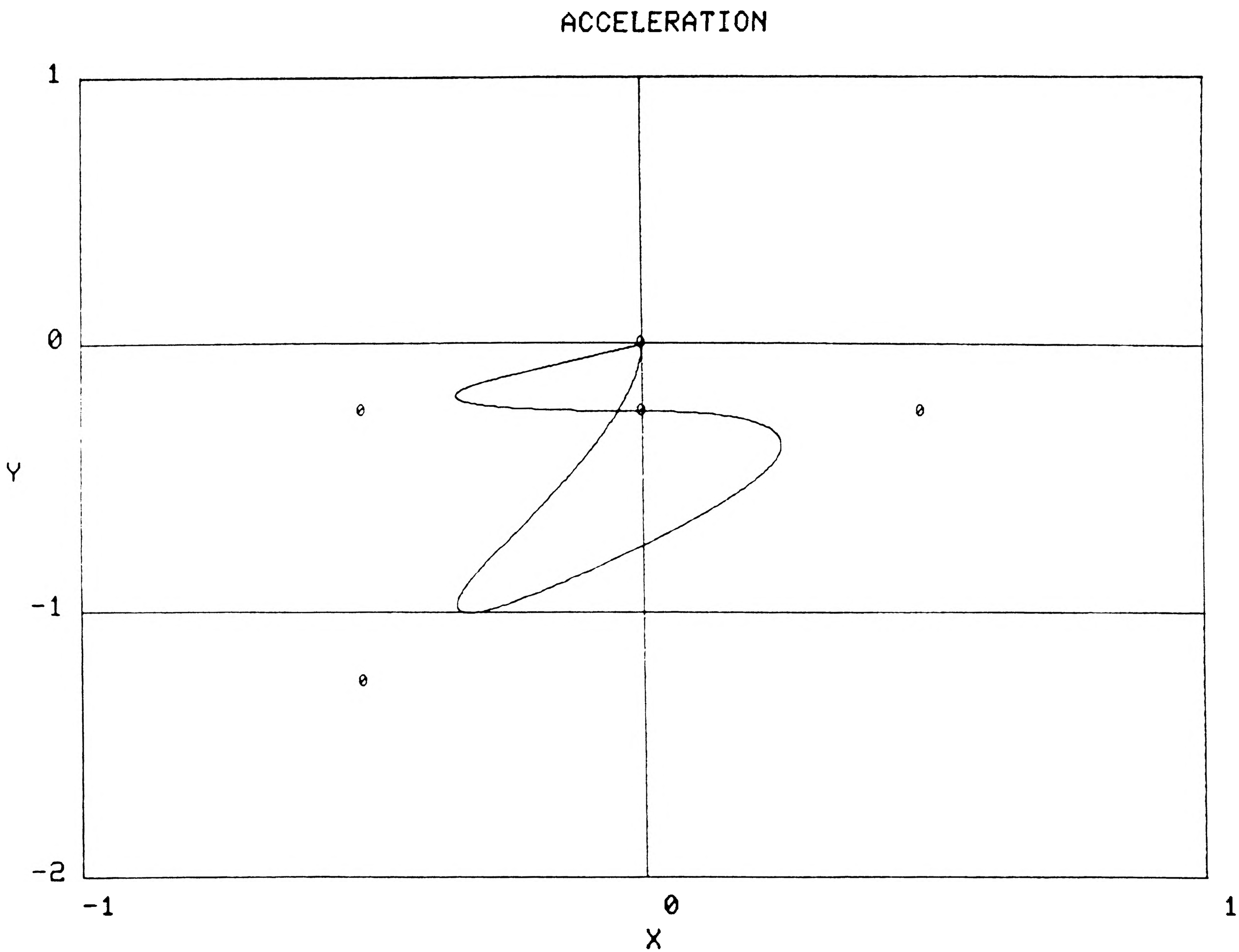

FIGURE 13. ACCELERATION: Y vs X

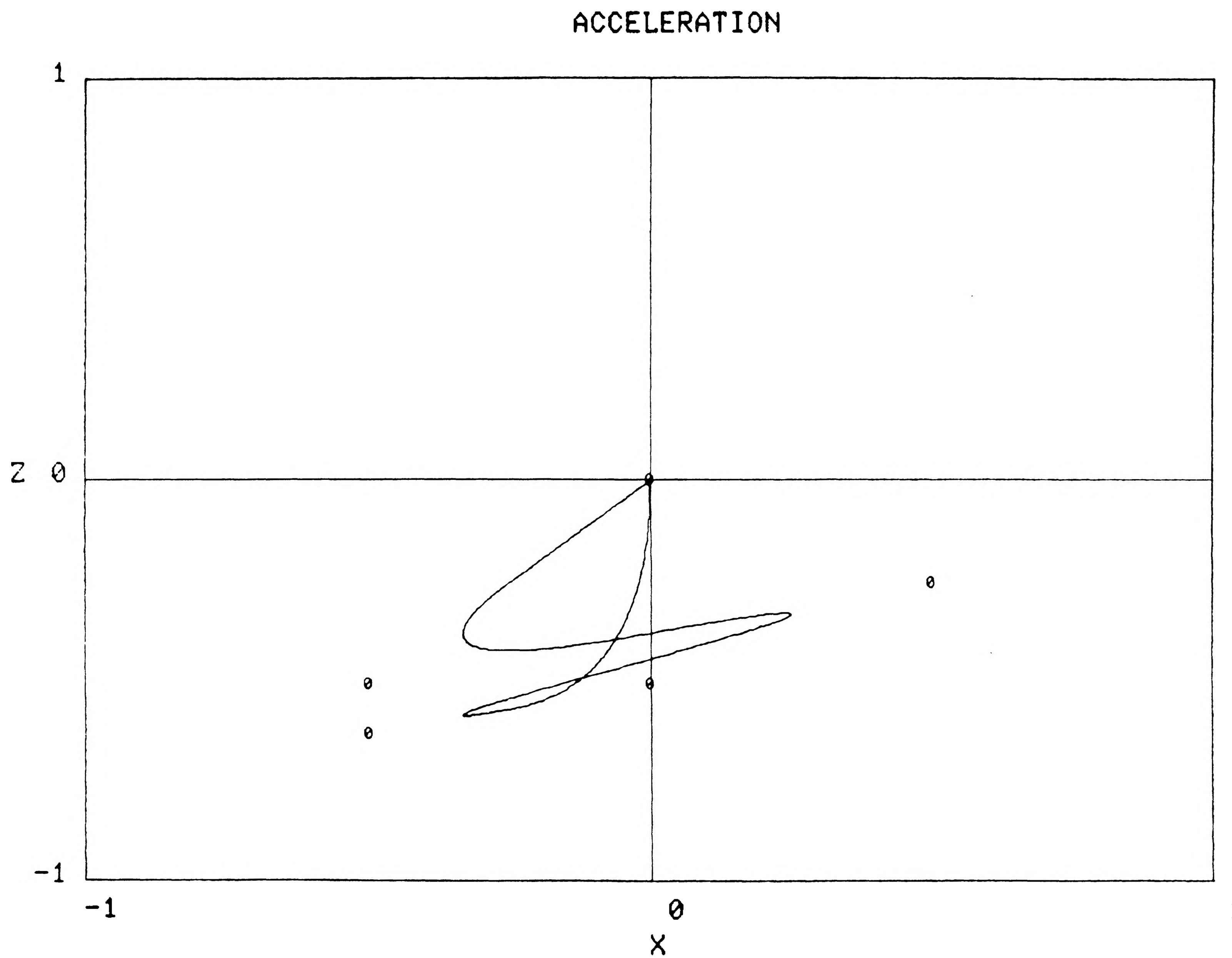

FIGURE 14. ACCELERATION: Z vs X

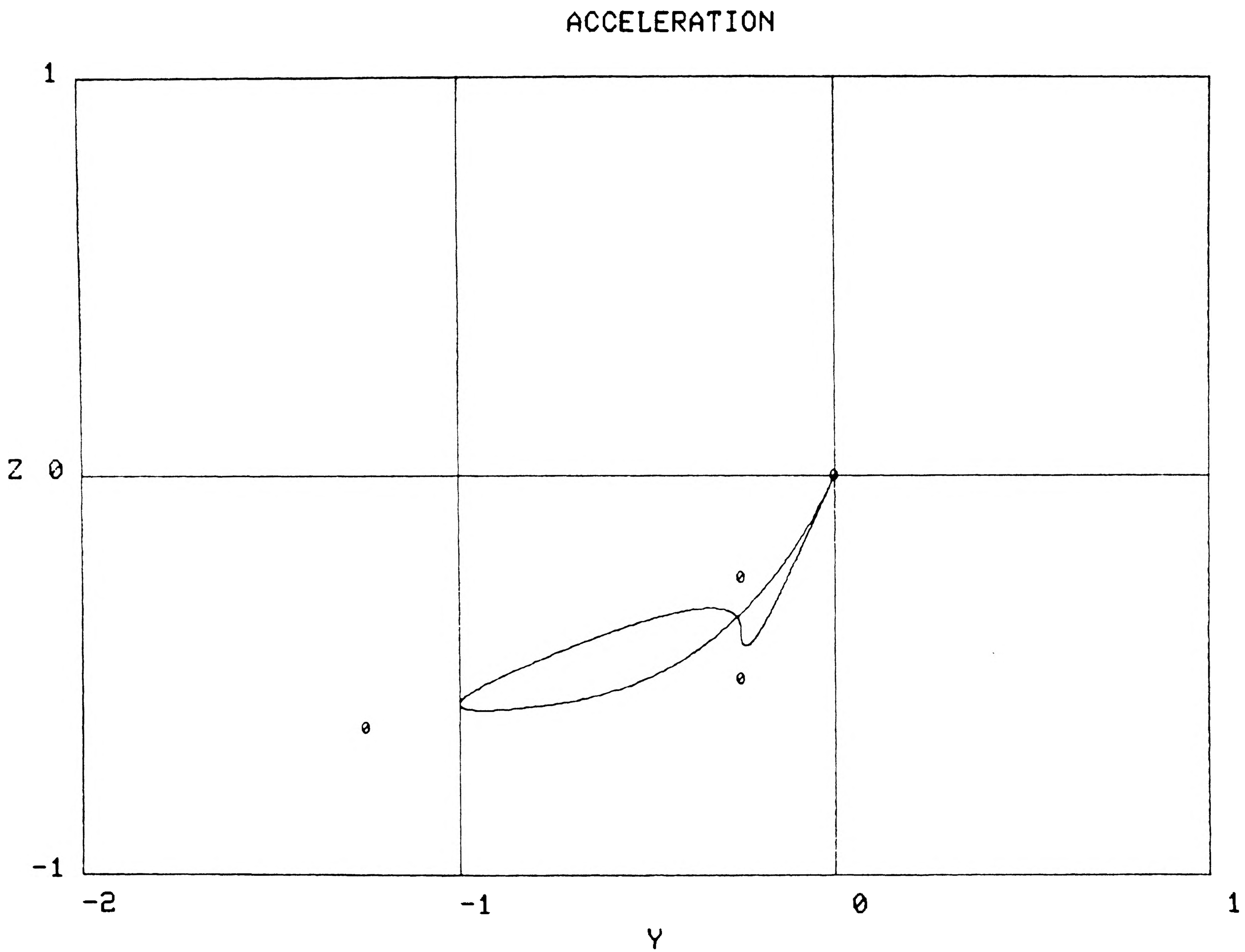

FIGURE 15. ACCELERATION: Z vs Y

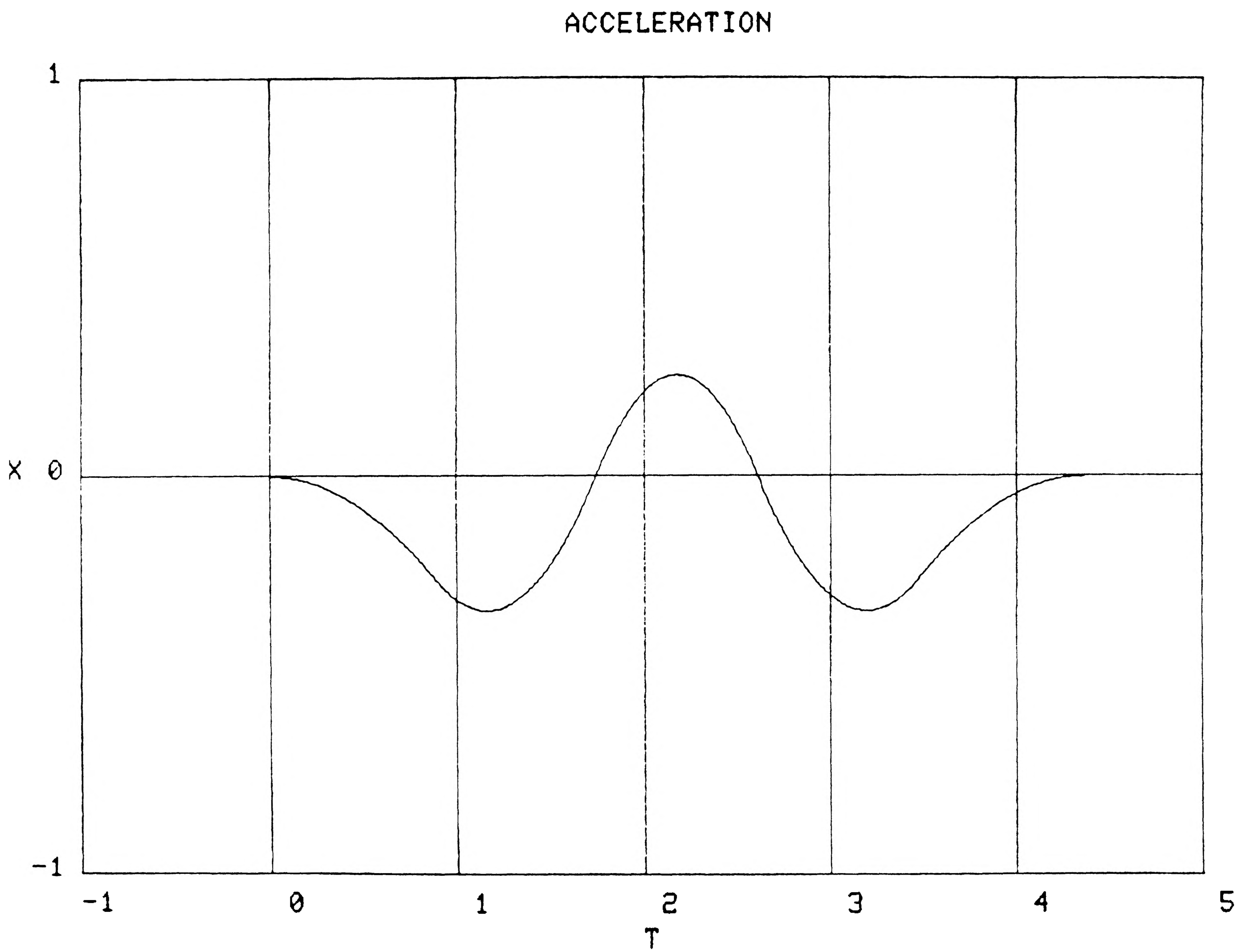

FIGURE 16. ACCELERATION: X vs T

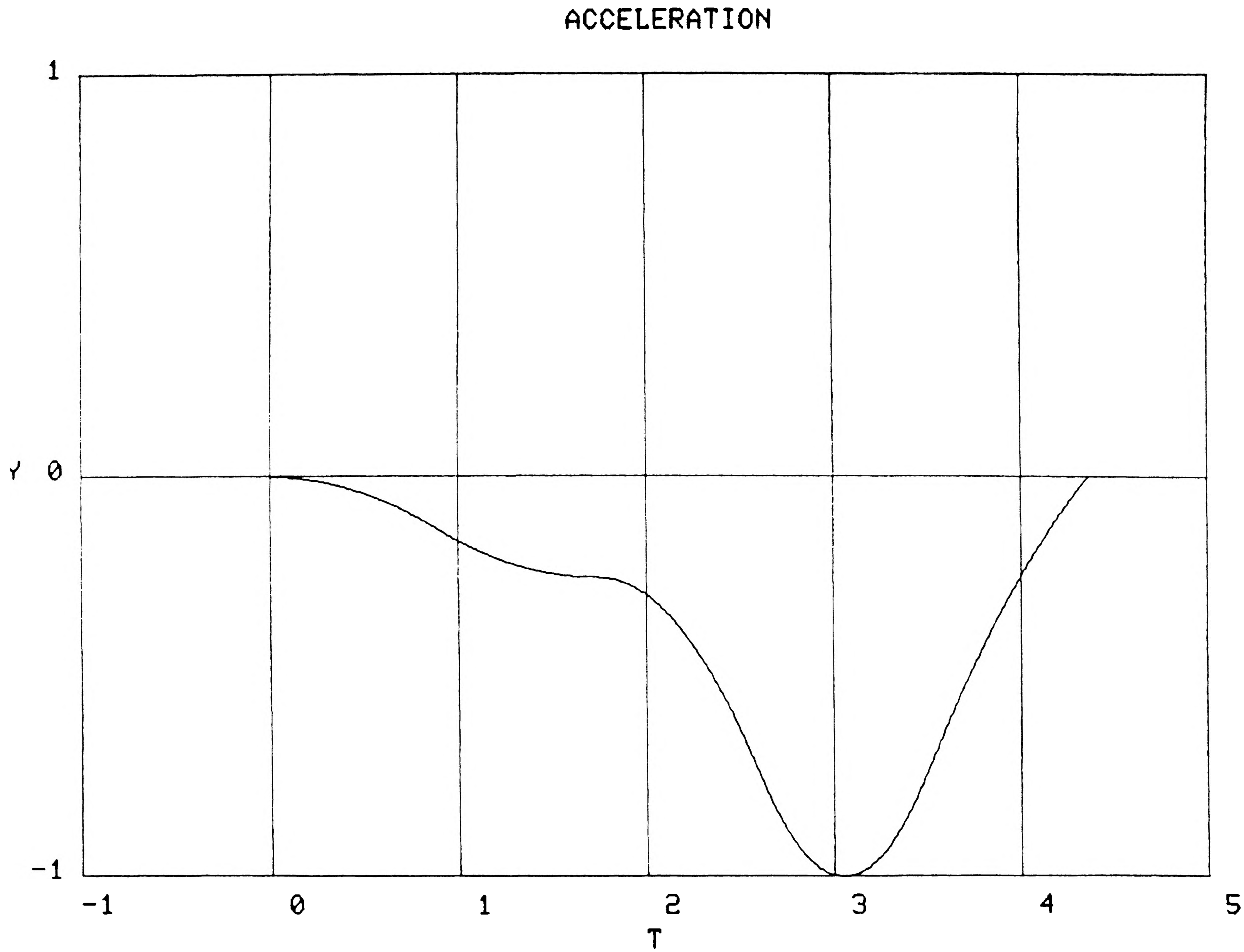

FIGURE 17. ACCELERATION: Y vs T

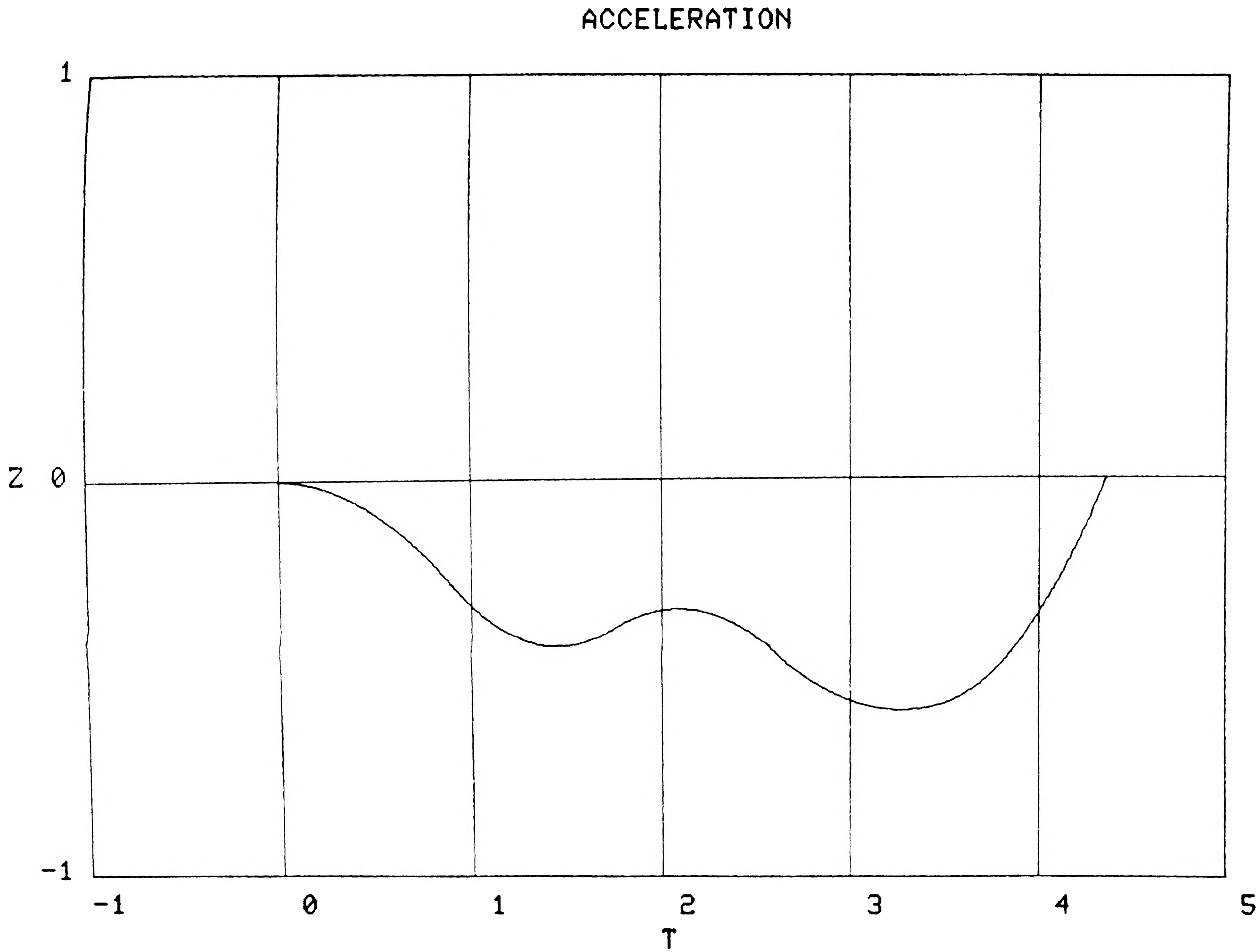

FIGURE 18. ACCELERATION: Z vs T

IV. DISCUSSION

The term spline originated during the days before the advent of computer graphics. When draftsmen were required to draw non-conventional curves of future workpieces they utilized a flexible wooden ruler. This wooden ruler, called a spline, would be weighted at points to trace the outline of the curve. The weights had a protrusion that fitted into a slot located on the spline. This allowed the spline to rotate around the fixed weights while holding it in place. Using the theory of mechanical elasticity one can prove that the resulting curve is a piecewise cubic polynomial. This polynomial is considered continuous in the first and second derivatives. These conditions assure that the curve has a continuous curvature and any discontinuities occur only in the third derivative. If one directs a mechanical motion along a spline, a continuous second derivative implies continuous accelerations and therefore no abrupt changes in force. These two properties make such curves desirable in many applications.

A. CUBIC SPLINES

In general a mathematical spline is a piecewise polynomial of degree K. The continuity of derivatives of order K - 1 of the spline occur at the common joints between the segments. Thus, for example, the cubic spline has second-order continuity at the joints of the segments. Piecewise splines of low degree are usually more useful for forming a curve through a series of points. The use of low-degree polynomials reduces the computational requirements and reduces numerical instabilities that arise with higher order curves. Since low-degree polynomials cannot span an arbitrary series of points, adjacent polynomial segments are needed. Based on these considerations and the analogy of the physical spline, a frequent technique is to use a series of cubic splines with each segment spanning only two points. Further the cubic spline is advantageous since it is the lowest degree space curve which allows a point of inflection and has the ability to twist through space.

A summary of the positive aspects regarding cubic splines include: The take-off and set-down positions of the velocity and and acceleration curves can be set to zero. As one can plainly see, this is useful in the generation of velocity and acceleration curves in robot trajectory paths. The user of such an algorithm can control the generated path to a much higher degree than some of the other path trajectory schemes. Also, the use of cubic splines allows an optimality in describing the shortest curve that passes through the control knots (as seen in Ho and Cook).

A negative aspect: Due to the global nature of the curve generated by the cubic spline method, the entire curve must be recalculated even when one only wishes to change a small portion of the robot path trajectory. Another negative aspect: adjacent polynomial segments are required as points are added to the curve.

As noted in Ho and Cook, cubic splines have been utilized in the creation of robot trajectory paths. In that research it was determined that continuous trajectories could be calculated with respect to position, velocity and acceleration.

B. B-SPLINES

B-Splines are a choice alternative to the use of cubic splines in the generation of robot trajectory paths. The B-Spline approach to trajectory path generation is easy to implement and computationally efficient. The shape of the curve is easily modified by the user who may add or delete knots. B-Splines allow the user to generate a trajectory path that does not have to be recalculated in totem. As one wishes to alter a small portion of the curve, only that portion of the curve that is to be changed must be recalculated. This is possible because B-Spline curves are iocally defined, thus the computation depends on the degree of the curve and not the number of knots. Multiple vertices allow interpolation and cusps and B-Spline curves are variation-diminishing. The differentiability is determined by the degree of the curve. This allows one to specify the velocity and acceleration at the ends of the path.

Thus B-Spline curves provide the user with the advantages of the cubic spline without the problems created by changing a portion of the curve. B-Spline curves also allow the user to create a trajectory by defining the vertices of an open polygon. To change the trajectory so that the curve will bend away from an object, the user may add an additional vertice to the open polygon. The generated curve will then bend towards the new vertice while avoiding a particular 'object'. This same objective (of 'bending away') can be achieved by adding weight to a particular vertice (giving the knot associated with the vertice a multiplicity of two or more).

Robot trajectory paths generated utilizing the B-Spline function definition are a useful way to provide control motion for the robot manipulator. Such curves can be recalculated locally and efficiently. Also they provide a methodology for collision avoidance when an object may be found in the manipulator's work envelope. Curves generated by the B-Spline function definition will provide the control system of the robot with optimal control motions with a minimum of effort.

V. CONCLUSION

Spline function curve fitting techniques to model trajectory paths precisely through controlling knot points have been demonstrated by Cook and Ho[7]. In this paper, a method of generating optimum trajectory models for for robot manipulators was developed through the use of B-spline curve functions. These functions provide the robot controller with a computationally efficient (and easy to implement) method of generating paths for the end effector.

B-spline curves are locally defined which greatly reduces the computational cost since any changes to the curve will only have to be calculated for that part of the curve and not the entire curve. The local definition enhances the ability to quickly modify a curve because adjusting the number and location of knots has a local effect on the curve, not a global effect on the entire curve. Thus, B-spline curve generation techniques are very useful in the creation of optimum paths for the end effector of a robot manipulator.

BIBLIOGRAPHY

1. Ho, C. Y. and K. W. Copeland. "A Simplified Solution of Kinematic Equations for Robot Manipulators", Mechanism and Machine Theory (1983).

2. Paul, R. P., B. Shimano and G. E. Mayer. "Kinematic Control Equations for Simple Manipulators", IEEE System, and Cybernetics, SMC-11 No 6 (June 1981) pp 449-455.

3. Paul, R. P. Robot Manipulators, MIT Press (1981).

4. Ho, C. Y. "Jacobian Determination of Robot Manipulators", Conference On Advanced Software In Robotics, Belgium (1983).

5. Paul, R. P., B. Shimano and G. E. Mayer. "Differential Kinematic Control Equations for Manipulators", IEEE System, Man, and Cybernetics, SMC-11 No 6 (June 1981) pp 445-460.

6. Hollerback, J. M. "A Recursive Lagrangian Formulation of Manipulator Dynamics and A Comparative Study of Dynamics Formulator", IEEE System, Man, and Cybernetics, SMC-11 No6 (Nov 1980) pp 164-172.

7. Cook, C. C. and C. Y. Ho. "The Application of Spline Functions to Trajectory Generation for Computer-Controlled Manipulators", J. of Digital Systems for Industrial Automation, (1982).

8. Curry, H. B. and I. J. Schoenberg. "On Spline Distributions and Their Limits: the Polya Distributions," Abstr. Bull. Amer. Math Soc., 53: 1114 (1947).

9. De Boor, C. "On Calculating with B-Splines", J. Approx. Theory,

6: 50-62 (July 1972).

10. Riesenfeld, R. F. "Application of B-Spline Approximation to Geometric Problems of Computer Aided Design", Available at U. of Utah, UTEC-CSc-73-126.

11. Curry, H. B. and I. J. Schoenberg. "On Polya Frequency Functions IV: The Fundamental Spline Functions and Their Limits", J. Analyse Math., 17: 71-107 (1966).

12. Schweikert, Daniel G. "An Interpolation Curve Using a Spline In Tension," J. Math. Phys., 45: 312-317 (September 1966).

13. Cline, A. K. "Curve Fitting Using Splines Under Tension," Atmos. Tech., No 3: 60-65 (1973).

14. Denman, H. H. "Smooth Cubic Spline Interpolation Functions," Industrial Mathematics, J. Ind. Math. Soc., 21: Part 2, 55-75 (1971).

15. De Boor, C. "On Uniform Approximation by Splines," J. Approx. Theory, 1: 219-235 (1968).

16. Manning, J. R. "Continuity Conditions for Spline Curves," The Computer Journal, 17: 181-186 (May 1974).

17. Marsden, Martin J. "An Identity for Spline Functions with Applications to Variation-Diminishing Spline Approximation," J. Approx. Theory, 3: 7-49 (March 1970).

18. Schoenberg, I. J. "Cardinal Interpolation and Spline Functions," J. Approx. Theory, 2: 167-206 (June 1969).

19. Karlin, Samuel and John M. Karon. "A Variation-Diminishing Generalized Spline Approximation Method," J. Approx. Theory, 1: 255-268 (November 1968).

20. Adams, J. Alan. "A Comparison of Methods for Cubic Spline Curve Fitting," Comput. Aided Des., 6: 1-9 (1974).

21. Nutbourne, A. W. "A Cubic Spline Package Part 2 - The Mathematics," Comput. Aided Des., 5: 7-13 (January 1974).

22. Schoenberg, I. J. "Contributions to the Problem of Approximationof Equidistant Data by Analytic Functions," Quart. Appl. Math., 4: 45-99, 113-141 (1946).

23. Gordon, W. J. and R. F. Riesenfeld. "Bernstein-Bezier Methods for the Computer-Aided Design of Free-Form Curves and Surfaces," J. ACM, 21: 293-310 (April 1974).

24. Forrest, A. R. "Interactive Interpolation and Approximation by Bezier Polynomials," The Computer Journal, 71-79 (April 1974).

25. Bezier, P. E. "Example of an Existing System in the Motor Industry: the Unisurf System," Proc. Roy. Soc. (London), A321: 207-218 (1971).

26. Barnhill, R. E. and R. F. Riesenfeld. Computer Aided Geometric Design. New York: Academic Press, Inc., 1974.

27. South, N. E. and J. P. Kelly. "Analytic Surface Methods," Ford Motor Company N/C Development Unit, Product Engineering Office, (December 1965).

VITA

Mary Claire Miller was born on September 15, 1958 in Trenton, Missouri. She received her primary and secondary education in Kansas City, Missouri. She has received her college education from William Jewell College, in Liberty, Missouri; the University of Missouri-Columbia, in Columbia, Missouri; and the University of Missouri-Rolla, in Rolla, Missouri. She also attended Oxford University, Oxford, England as a William Jewell Scholar. She received a Bachelor of Arts degree in Statistics from the University of Missouri-Columbia, in Columbia, Missouri in May 1980. She has been enrolled in the Graduate School of the University of Missouri-Rolla during the period January 1981 to May 1981 and since January 1982.

While at the University of Missouri-Rolla she was a CAD/CAM Graduate Research Assistant for the Computer Center. As an engineering analyst for a major company, she provided the design for an automated robot work cell. Miss Miller has also worked as a systems analyst for a North American oil company.

Miss Miller is a member of Upsilon Pi Epsilon, the Association for Computing Machinery, the Institute of Electrical and Electronics Engineers, the Society for Manufacturing Engineers, and the American Production and Inventory Control Society.