**Stephen Derby**

Rensselaer Polytechnic Institute
Troy, New York 12181

# Simulating Motion Elements of General-Purpose Robot Arms

## Abstract

*Robot arm displacement curves are constructed, assuming that all links in the arm are rigid. This can be accomplished by using familiar, cam-motion profile techniques whenever possible. The benefits of these considerations can help achieve smoother motion while minimizing wear on and effort by the robot actuators. A set of motion primitives within the General Robot Arm Simulation Program (GRASP) are presented as examples.*

## 1. Introduction

Basically, an operator-taught robot will move in a path that is only as intelligent as the person who programmed it. Such a path, which may be repeated a vast number of times, will probably not be traversed in the shortest possible time and may well produce stresses on the machinery that are far from desirable. Ideally, the robot should move in a path that takes the least total time and makes as few actuator demands as possible. Previous work has resulted in straight-line solutions and the fitting of polynomials to world hand-path constraints (Gill, Paul, and Scheinman 1973; Paul 1975*a*, 1975*b*, 1977, 1981; Taylor 1977; Lewis 1979).

Polynomials have also been fitted to intermediate points as well as end points (Lewis 1979). Makimo and Furuya (1981) have very recently used cam theory on a microcomputer to control a robot for point-to-point motion. Other efforts in controlling the trajectory have to do with the equations of motion together with various feedback and feedforward options (Beckett and Mergler 1970; Kahn 1970; Blanchard 1976; Liegeois 1977; Vukobratović, Stokić, and Hristić 1977; Mian 1978). Attempts have also been made to optimize the path with respect to avoiding obstacles, which requires some kind of solid modeling (Pfister 1973; Loeffand Soni 1975; Waldron 1976; Udupa 1977; Lozano-Perez and Wesley 1979; Meagher 1980). Most of these methods work on some variation of the inverse square law and are not optimized for the motion's continuity.

Unfortunately, there is no spatial equivalent to planar mechanism synthesis of desired paths or other motion characteristics. Any "optimal" path is achieved by trial and error.

The material presented here was developed during the writing of GRASP (Derby 1981; 1982*a*). This program allows users to design new robots, modify existing robots, and evaluate existing robots in their working environment. These arms must be kinematically modeled as in Fig. 1. GRASP associates desired points, with various motion primitives as options. It does no obstacle avoidance or optimal path search. The user has to guide the arm around obstacles by entering appropriate locations. The user defines the desired speed, and the program computes the time required. If there is no actuator information, GRASP uses default velocities and accelerations to help determine what size actuators would be needed.

## 2. Present Methods

Two basic types of motion are used in industrial robots. The first consists of straight-line hand movement, in world coordinates, from point A to point B (Fig. 2). This is accomplished by solving the backward-solution equations for every small delta position or delta time and results in whatever joint-

The International Journal of Robotics Research
Vol. 2, No. 1, Spring 1983
0278-3649/83/010003-10 $05.00/0

Fig. 1. Vector definitions.

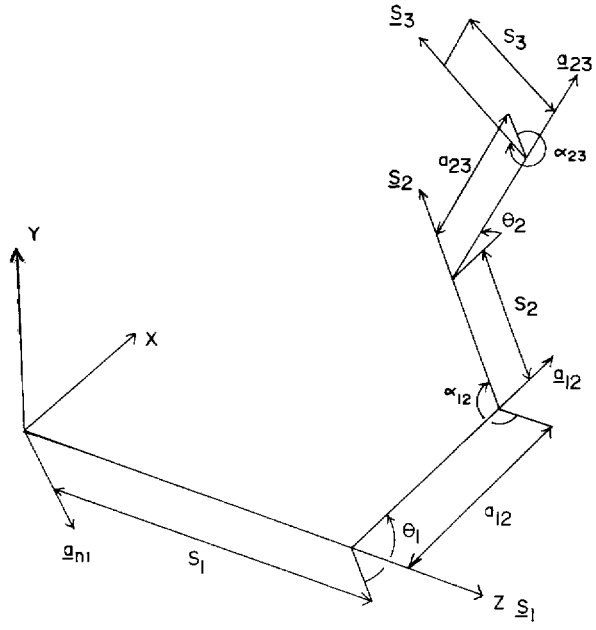
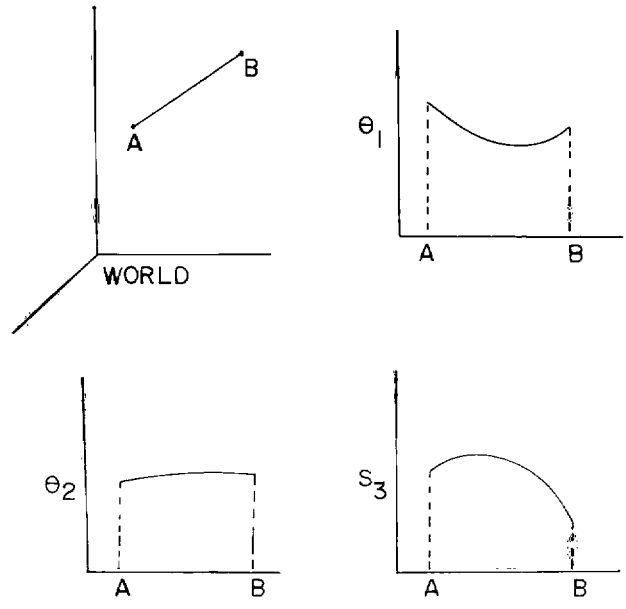



Fig. 2. Straight-line movement.




variable values are needed to achieve this (Fig. 2). Some arms move only in this fashion, using some starting and finishing ramping; most arms use short, straight-line segments to approximate spatial arcs.

The second type of motion is joint-interpolated motion. This procedure takes the existing joint-variable values at a point A and their computed values at a second point B and linearly traverses the difference in time (Fig. 3). The hand's trajectory is not an easily defined path. The shortest ideal time is achieved, however. This is accomplished by having at least one of the joints running at its maximum speed. Realistically, some kind of starting and finishing consideration is needed to avoid instantaneous, high accelerations and jerks.

## 3. Cam Motion Considerations

The path of the hand is important. The time derivatives of the path and the joint variables are also important, however. Confining the hand's trajectory may cause high joint velocities and torques. An area that has been well researched is the displacement

**Table 1. Displacement Boundary Conditions**

| Variables | Start | End |
|---|---|---|
| Position | 0 | L |
| Velocity | 0 | 0 |
| Acceleration | 0 | 0 |

equations of a cam. Motion can be cycloidic, harmonic, or defined by polynomials that are calculated from boundary conditions. The motion described by a 3-4-5 polynomial (Shigley and Uicker 1980) with the boundary conditions listed in Table 1 is shown in Fig. 4. This polynomial can be written in the form

$$\text{Pos} = L(10t^3 - 15t^4 + 6t^5), \quad (1)$$

where $L$ is the total displacement and $t$ is a unit of time. (With respect to the interval, it is assumed that it takes 1 s to travel along this polynomial.) The vast majority of motions will not occur in a unit second, and (Eq. 1) can be modified by replacing $t$ by $\frac{t}{T}$, where $t$ is the real time and $T$ is the total path time.

*Fig. 3. Joint interpolation.*

*Fig. 4. Polynomial position and velocity acceleration.*

*Fig. 5. Matching derivatives.*

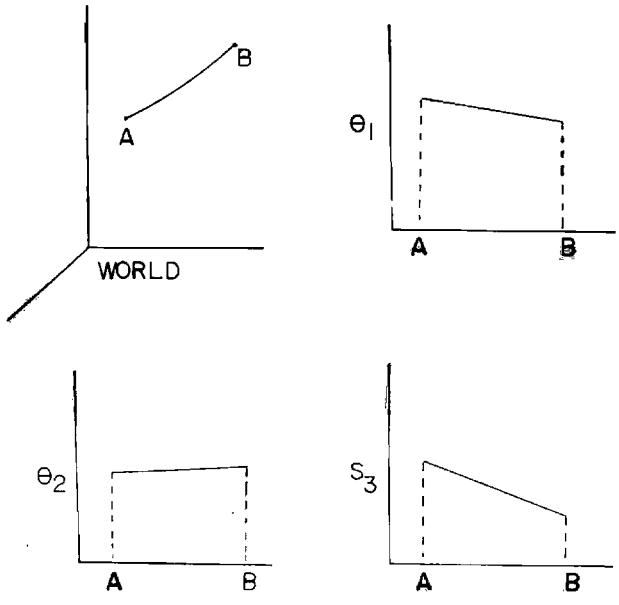
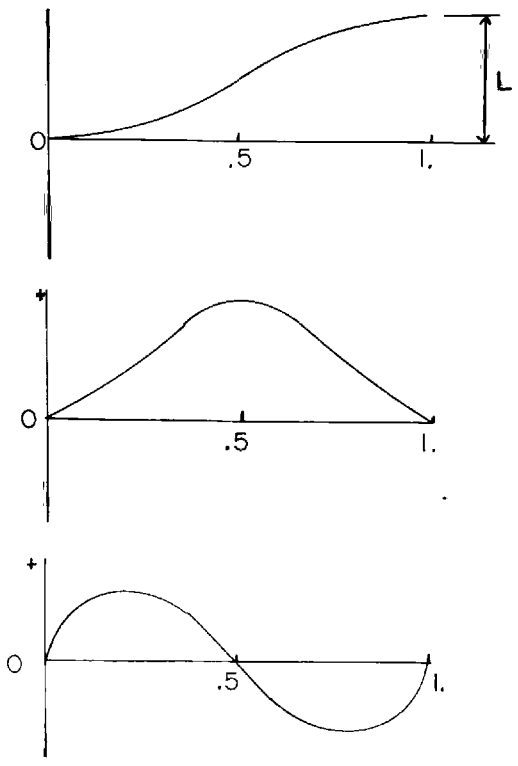
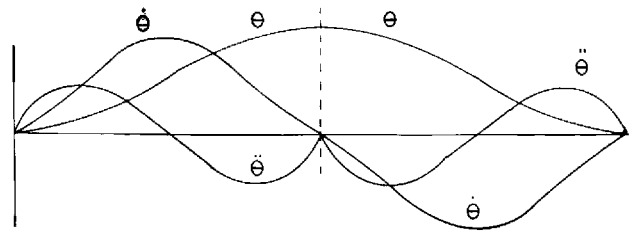



Fig. 4




Hence, (Eq. 1) can be expressed in the form

$$\text{Pos} = L\left[10\left(\frac{t}{T}\right)^3 - 15\left(\frac{t}{T}\right)^4 + 6\left(\frac{t}{T}\right)^5\right]. \quad (2)$$

The first time derivative (Fig. 4) is given by

$$\text{Vel} = \frac{L}{T}\left[30\left(\frac{t}{T}\right)^2 - 60\left(\frac{t}{T}\right)^3 + 30\left(\frac{t}{T}\right)^4\right], \quad (3)$$

and the second time derivative (Fig. 4) is given by

$$\text{Acc} = \frac{L}{T^2}\left[60\left(\frac{t}{T}\right) - 180\left(\frac{t}{T}\right)^2 + 120\left(\frac{t}{T}\right)^3\right]. \quad (4)$$

One advantage in cam design is the ability to match derivatives in order to ensure against high or infinite jerk when changing from one defined curve to another. Figure 5 shows how the start of the return portion can match the end of the rise portion.

Whenever the hand is not required to travel in a confined path, the program takes advantage of the results of cam design. Having a choice between these types of motion leads to generic results that may not match the characteristics of all existing robots but that will be of value to the designer.

## 4. GRASP Motion Primitives

There are seven motion primitives and three different blends in the program. All motion is made up of a composite of these primitives, given in Table 2. Each primitive will now be explained in more detail.

*Fig. 6. Pick up (PIC primitive).*

*Fig. 7. Place (PLA primitive).*

**Table 2. Motion Primitives**

| Notation | Definition |
|---|---|
| PIC | Pick up, whether hand has grasped something or not |
| PLA | Place, whether hand has anything to place or not |
| TWIST | Twist the hand |
| SLIDE | Slide the hand |
| STLINE | Move hand in straight line |
| PT TO PT | Joint interpolated from point to point |
| TEACH PT | Joint interpolated from taught point to point |

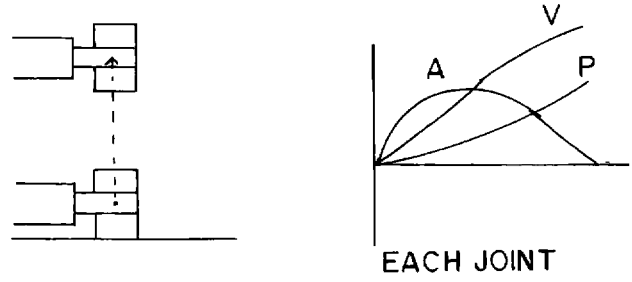
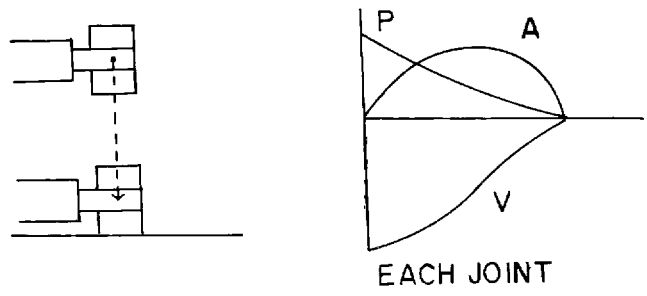



Fig. 7




### 4.1. PIC Primitive

The PIC primitive is used whenever the hand is to accelerate from rest to a desired velocity. It is shown in Fig. 6 (left) as a straight vertical line, but will deviate depending on the programmed displacement. The PIC primitive can be used to lift in any direction. If the arm were in the lower position, the user would program the location and orientation of the upper position, and the program would solve for the joint variables. It is assumed that the hand is to reach the target velocity in one direction, and this velocity is used to scale the first half of a 3-4-5 rise polynomial (the maximum velocity point). For an arm with six degrees of freedom, there would be six 3-4-5 polynomials created. Several points are chosen from the joint-position curve, and the joint velocities and output torques are tested. Hence, no other backward arm solutions are needed. These curves are used from rest up to the 90% point of the path, whereupon the blending polynomial has control transferred to it. The PIC primitive produces a different type of point-to-point motion, allowing for a smoother acceleration.

### 4.2. PLA Primitive

The PLA primitive serves the reverse function of the PIC primitive. It uses the latter half of the return 3-4-5 polynomial (Fig. 7), and its characteristics are similar. It guides an empty or full hand to a resting position. The blend function will intersect 10% into the path.

### 4.3. ST Line Primitive

Straight-line motion does not allow for any shortcuts. The motion (Fig. 8) ideally has constant velocity (i.e., zero hand acceleration). At each practical finite location the arm joint variables must be solved for using the backward solution, and realistically need to ramp up to and down from the constant velocity. The program does not calculate points with as much resolution as would usually be needed, but this saves time because information that is not helpful in the design and application phase is not calculated and stored. Also, since blending polynomials are used to connect primitives, the motion is at a constant velocity for the entire distance $d$.

### 4.4. TWIST Primitive

The TWIST primitive can be used in two ways. Figure 9 shows how an object can be rotated about a specified axis by an angle $\phi$ in such a manner that it goes from rest to having a specified twist velocity.

*Fig. 8. Straight line (STLINE primitive).*

*Fig. 9. Twist (TWIST primitive).*

*Fig. 10. Slide (SLIDE primitive).*

*Fig. 11. Point to point (PT TO PT primitive).*

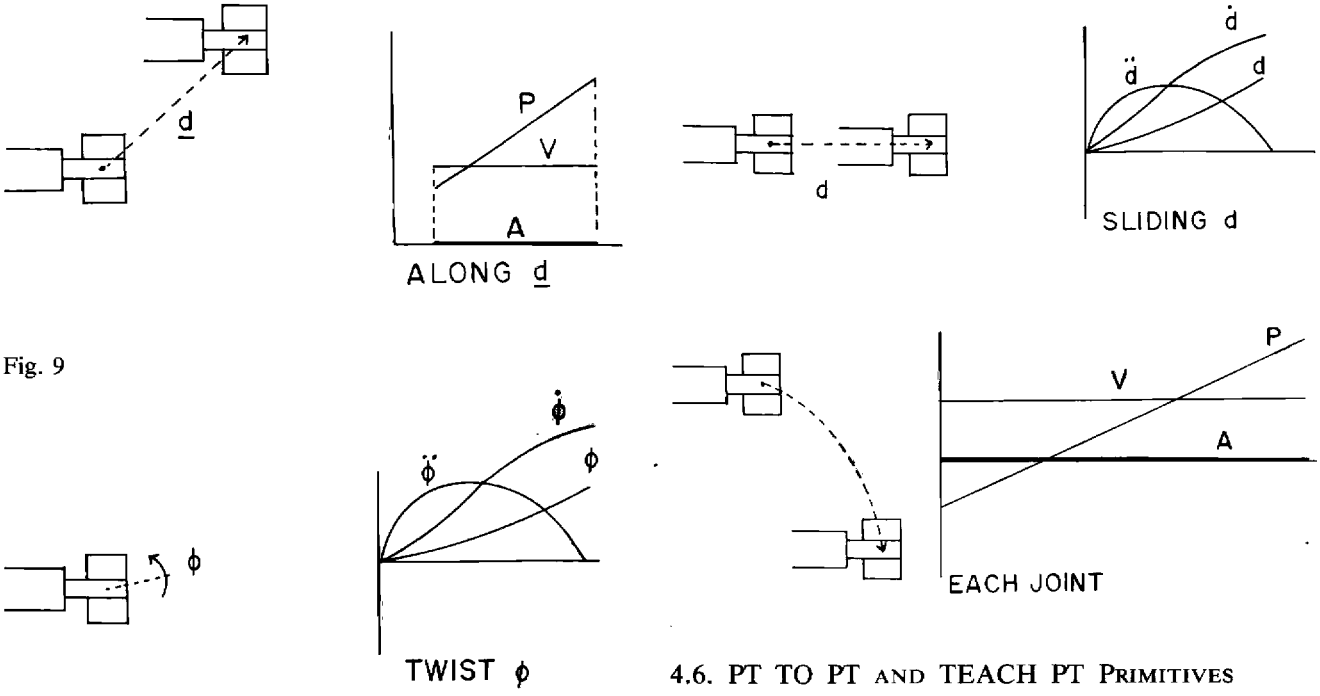



Fig. 9

This is used to obtain a motion that can be blended into a second motion, which moves the object to the next desired position. It is intended to eliminate the kinds of motion that have an intermediate rest or stationary point between motion segments. Several points are taken from the $\phi$ curve, and the backward solution is used. It is blended at the motion end.

The second method is to use the TWIST option after one of the displacement options in order to twist to a resting position. TWIST is a special case of the STLINE primitive.

### 4.5. SLIDE Primitive

The SLIDE primitive moves the hand from the current position to the next position only if the orientation is to remain the same. It is similar to the TWIST option in that it has the ability to move the hand from rest to motion (Fig. 10) or from motion to rest. It also relies on the backward solution and is blended at the motion end. SLIDE is a special case of the STLINE primitive.

### 4.6. PT TO PT and TEACH PT Primitives

If the arm has five degrees of freedom or less or has special geometry and is not one of the seven special cases as given in (Eq. 1), then the program can only use TEACH PT. The PT TO PT motion and the TEACH PT motion are identical except for the method by which the user enters the next position. Figure 11 shows a curved path. The point-to-point motion shown is for a small displacement judged to be too small to have both a blending polynomial and joint-interpolated motion. In this case, the joint-interpolated motion is not used, and only the motion-to-motion blending is used. This will be discussed more in the next section.

If the displacement is not small, then the motion is joint interpolated as shown in Fig. 3. Blending occurs 10% from each end of the hand path.

### 4.7. Blending Polynomials

Three types of blending are used: (1) from rest to motion, (2) from motion to rest, and (3) from motion to motion. They are used when the motion primitive does not start or end at the right boundary condi-

tions with respect to the previous and following motions. The proper type of blending is selected by the program. GRASP requires that the gripper be at rest in order to grasp or release an object.

The rest-to-motion blend is achieved by using the first half of the 3-4-5 rise polynomial (see Section 4.1) and is also in the joint-variable world. The motion-to-rest blend is achieved using the latter half of the 3-4-5 return polynomial (see Section 4.2). The motion-to-motion blend is achieved by different means. Attempts to use a 3-4-5–type polynomial and have the position, velocity, and acceleration match at both ends proved to be unsatisfactory. When a totally smooth acceleration curve was required, the position's derived path became unruly and consequently did not perform well as a blend. It wasted much time and would very often cancel out much of the progress made with the preceding motion primitive during its intermediate points. A second attempt used a third-degree polynomial and matched only position and velocity, but a better method was chosen.

A class of third-degree blending polynomials called *Bezier curves* has been used in computer graphics. This is a form of a binomial distribution and, for points A, B, and C as shown in Fig. 12, is given as

$$\text{Pos} = (1 - t)^2A + 2t(1 - t)B + t^2C. \qquad (5)$$

This allows for matching of velocities, as the tangents to $\overline{AB}$ and $\overline{BC}$ are given when points A and C are selected 10% into the motion primitives, and point B is the unblended midpoint where the two primitives

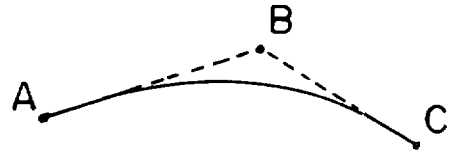



*Fig. 12. Three points in a Bezier curve.*

meet. The acceleration is kept at a reasonable, constant rate.

The midpoint of all blends is used for computing velocities and torques.

## 5. Motion Program Philosophy

With a fair number of primitives, many combinations can be achieved. Not all combinations are permissible (e.g., two PIC primitives in a row). Table 3 shows the combinations that are permissible; permissibility is checked after each primitive is selected.

The positions, orientations, and motion primitives of the user-programmed path are stored for calculating in the GRASP RUN function. The entire path could be programmed in one operation (Lewis 1979) if all the primitives were polynomials. The matrix inversion needed for the solution of simultaneous boundary conditions can get very large. However, GRASP avoids this problem by taking one primitive at a time and doing a running check on velocities and torques.

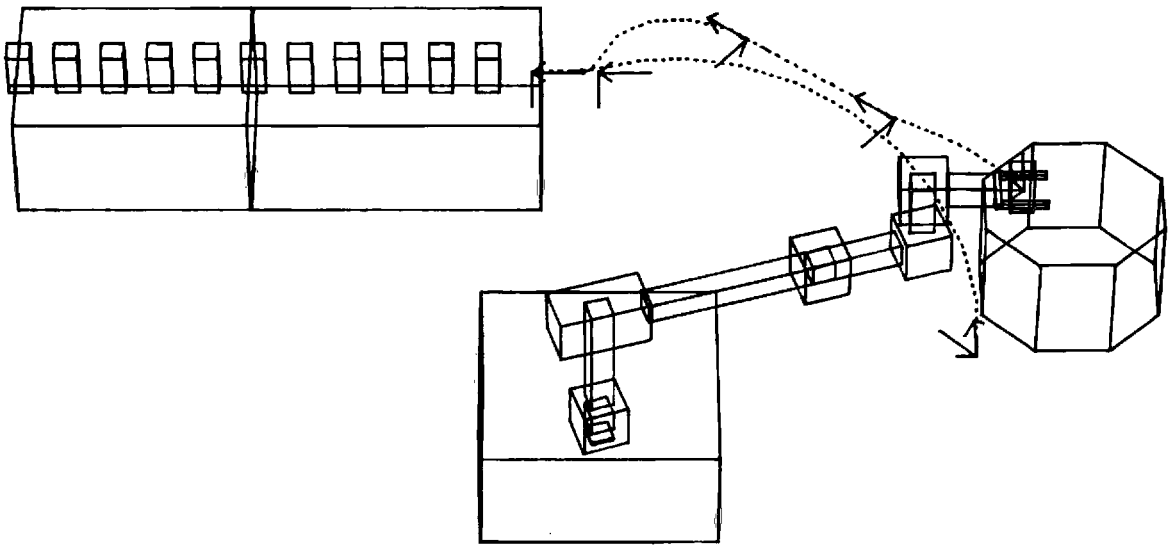
The end of the present motion state, whether or not it is rest, is used as the starting position for the next motion. After calculating that primitive's mo-

**Table 3. Valid Primitive Orders**

| From | To | | | | | | |
|---|---|---|---|---|---|---|---|
| | PIC | PLA | PT TO PT | STLINE | SLIDE | TWIST | TEACH PT |
| PIC | I | I | V | I | I | I | V |
| PLA | V | I | V | I | V | V | V |
| PT TO PT | V | V | V | V | V | V | V |
| STLINE | I | I | V | V | I | I | V |
| SLIDE | V | I | V | I | V | I | V |
| TWIST | V | I | V | I | I | V | V |
| TEACH PT | V | V | V | V | V | V | V |

Note: I = invalid choice; V = valid choice.

*Fig. 13. GRASP motion.*



| | USING H0001 A0001 W0001 | STORED P0002 | | | |
|---|---|---|---|---|---|
| MOTION NO LIMITS | READIN | RUN | POSITION | RETURN | NO DEFORMATION HAND TRACE |
| NO MINMAX | STATUS | PL OUTPUT | HARDCOPY | NEWPOSFIL | NO INTERFER CHK NOT SHOW ATTEMPTS |

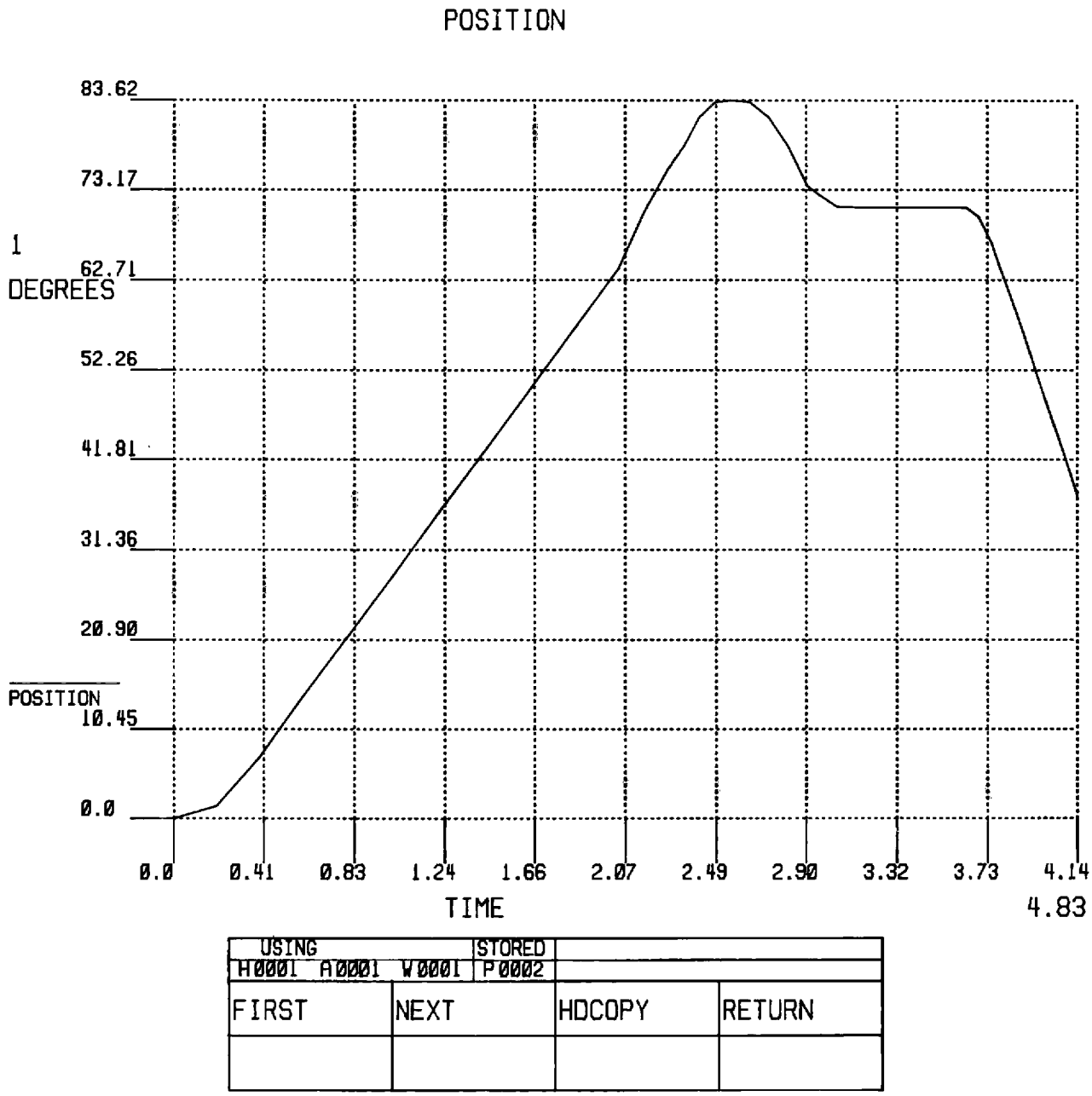
tion equations, GRASP blends the present motion primitive and the next primitive, if one is needed. An initial run through is performed with this blend and primitive in order to compare the velocity and test the actuator limits, if given. Most likely, this will result in a time revision. The potential drawback to this procedure of checking one motion segment at a time is that the following primitive may make greater demands on the actuators, and the next blend may not be able to accomplish its task. A preliminary blend is made in order to prevent this from happening. Safety is incorporated by use of 110% of the revised time, since the sampled points may not occur at the maximum positions on the motion curves. A finish blend is also needed when the hand must be at rest so that it can grab or release an object. A finish blend is also used to update a pulse conveyor.

The only time this segment-checking procedure may produce different outputs is when the RUN command is executed and the last primitive leaves the arm in motion. The next blend function will not have been used, and the calculations may differ. Using the revised time, the path is retraced and the calculated results recorded. These results may be plotted for evaluation.

A sample model of GRASP is given in Fig. 13, which shows a generic, six-degrees-of-freedom robot, its hand, and the working environment. The dotted line represents the center point of the hand. Figures 14 and 15 are samples of calculated outputs from GRASP that can aid the user in designing a robot or evaluating its performance along a possible path. It should be noted that the blending does not eliminate the higher-order discontinuities in accelerations and jerk. As with cam theory, one cannot achieve all the desired characteristics at one time. This is particularly true in a motion-to-motion change. The GRASP program is explained in great detail elsewhere (Derby 1981; 1982*a*; 1982*b*).

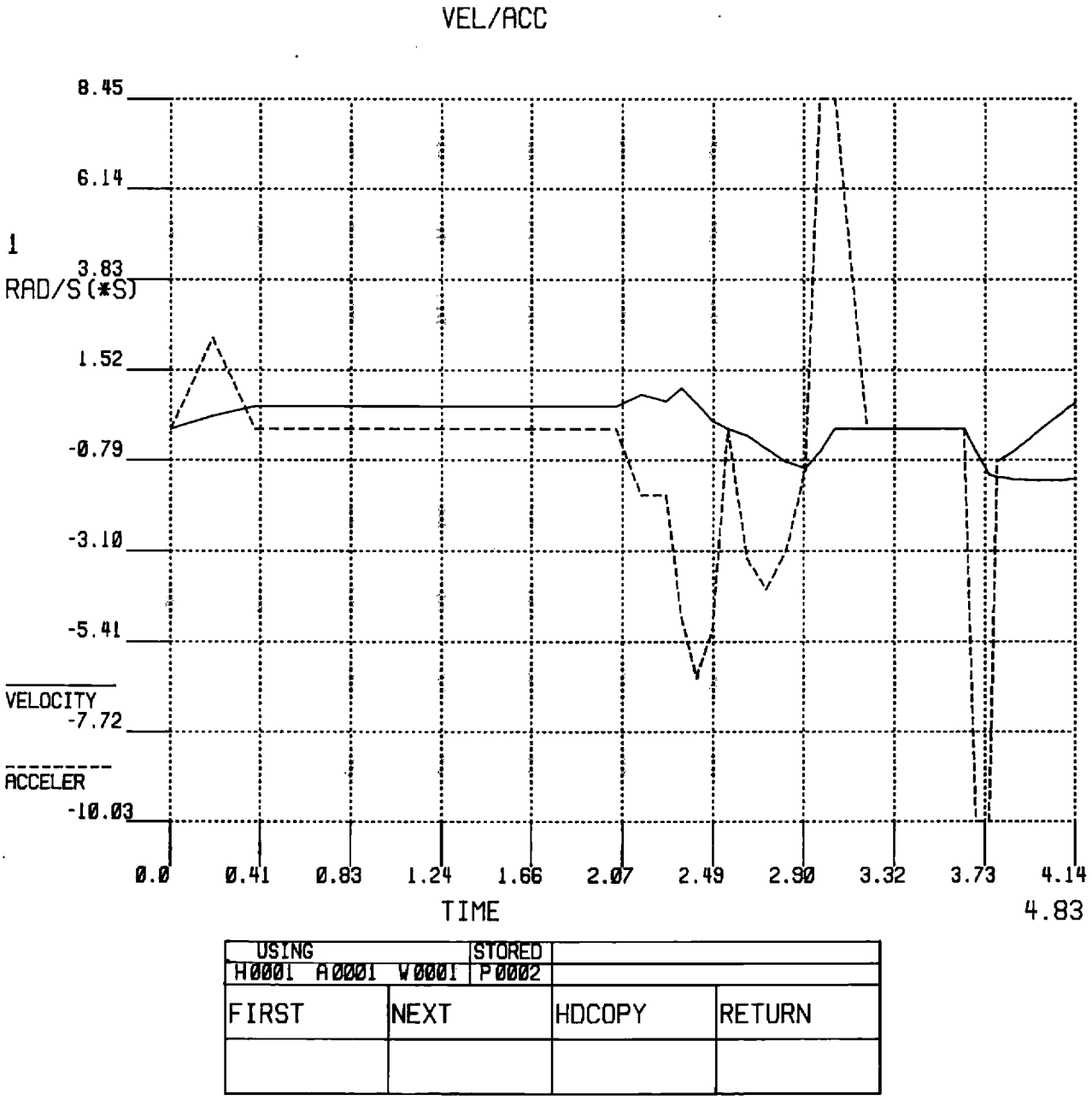
GRASP is a novel, general-purpose computer program developed to enable a designer to evaluate the performance of robot arms in potential working environments. The designer can base evaluations on time

*Fig. 14. Position versus time.*




and motion studies of task performance. A vector-refresh graphics terminal displays the "wire-frame" image of the manipulator, its hand, and the primitives describing the workspace. The resulting animation in simulated time will provide a proper base for designing new manipulators, for modifying existing designs, and for the logical layout of any machines with which the robot may interact.

*Fig. 15. Velocity and acceleration versus time.*




## 6. Summary

A planned set of motion primitives can logically move a robot arm so as to minimize overall motion and stress. Consideration of motion in the joint worlds as well as the hand world can aid in the development of dependable robots. These motions can help one understand requirements for robotic

mechanical design and control strategy. GRASP is a great help in utilizing this approach.

## References


Beckett, J. T., and Mergler, H. W. 1970. Controlling a remote manipulator with the aid of a small computer. ASME paper no. 70-DE-18.

Blanchard, D. C. 1976 (Aug.). Digital control of a six-axis manipulator. MIT Working Paper 129. Cambridge, Mass.: Massachusetts Institute of Technology Artificial Intelligence Laboratory.

Derby, S. 1981. Kinematic elasto-dynamic analysis and computer graphics simulation of general purpose robot manipulators. Ph.D. thesis, Rensselaer Polytechnic Institute.

Derby, S. 1982*a* (Mar.). Computer graphics simulation of robot arms using the GRASP program. Paper delivered at MIT CAD/CAM Conf., Cambridge, Mass.

Derby, S. 1982*b* (Aug.). General robot arm simulation program (GRASP): Part 1, a program to evaluate the performance of industrial robots in their working environment. Paper delivered at ASME Comput. Engineering Conf., San Diego, Calif.

Derby, S. 1982*c* (Aug.). General robot arm simulation program (GRASP): Part 2, methods of joint solutions and the reachable volume. Paper delivered at ASME Comput. Engineering Conf., San Diego, Calif.

Gill, A., Paul, R., and Scheinman, V. 1973. Computer manipulator control, visual feedback and related problems. Paper delivered at 1st CISM IFToMM Symp. Robots, Udine, Italy.

Kahn, M. E. 1970. The near-minimum-time control of open-loop articulated kinematic chains. Ph.D. thesis, Stanford University.

Liegeois, A. 1977. Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Trans. Syst. Man Cybern.* SME-7:(12):868–871.

Lewis, R. A. 1979. Autonomous manipulation on a robot: Summary of manipulation software. NASA Tech. Memo 33-679. Pasadena, Calif.: Jet Propulsion Laboratory.

Loeff, L. A., and Soni, A. H. 1975 (Aug.). An algorithm for computer guidance of a manipulator in between obstacles. *J. Engineering for Industry.* 79B(3):836–842.

Lozano-Perez, T., and Wesley, M. A. 1979. An algorithm for planning collision-free paths among polyhedral obstacles. *Commun. ACM* 22(10):560–570.

Makimo, H., and Furuya, N. 1981 (Oct.). Motion control of a jointed arm utilizing a microcomputer. Paper delivered at 11th Int. Symp. Industrial Robots, Tokyo, Japan.

Meagher, D. 1980. Octree encoding: A new technique for the representation manipulation. RPI Tech. Rept. IPL-TR-80-111. Troy, N.Y.: Rensselaer Polytechnic Institute Image Processing Laboratory.

Mian, M. I. A. 1978. Trajectory control of a robot joint with microprocessor. Master's thesis, North Carolina State University.

Paul, R. 1975*a*. Manipulator path control. *IEEE Trans. Syst. Man Cybern.* SMC-9(11):702–711.

Paul, R. 1975*b*. Manipulator path control. Fifth Stanford Research Institute Report. Menlo Park, Calif.: SRI International.

Paul, R. 1977. WAVE—a model based language for manipulator control. *Industrial Robot* 4(1):10–17.

Paul, R. 1981. *Robot Manipulators.* Cambridge, Mass.: MIT Press.

Pfister, G. F. 1973 (Mar.). On solving the findspace problem, or how to find out where things aren't. MIT Working Paper 113. Cambridge, Mass.: Massachusetts Institute of Technology.

Shigley, J. E., and Uicker, J. J., Jr. 1980. *Theory of machines and mechanisms.* New York: McGraw-Hill.

Taylor, R. 1977. Planning and execution of straight-line manipulator trajectories. IBM Res. Rept. RC 6657. Yorktown Heights, N.Y.: IBM Thomas J. Watson Research Center.

Udupa, S. M. 1977. Collision detection and avoidance in computer controlled manipulators. Ph.D. thesis, California Institute of Technology.

Vukobratović, M., Stokić, D., and Hristić, D. 1977. New control concept of anthropomorphic manipulators. *Mechanism Mach. Theory* 12(5):515–530.

Waldron, K. J. 1976. The manipulator interference problem. ASME paper no. 76-DET-68.