Comput. & Graphics Vol. 9, No. 4, pp. 383–391, 1985
Printed in Great Britain.

0097-8493/85 $3.00 + .00

Technical Section

SAM-ANIMATION SOFTWARE FOR SIMULATING ARTICULATED MOTION†

ANTHONY A. MACIEJEWSKI and CHARLES A. KLEIN
Department of Electrical Engineering, The Ohio State University, 2015 Neil Avenue, Columbus, OH 43210, U.S.A.

Abstract—A collection of software used for interactive specification, motion control and graphics simulation of articulated objects of arbitrary complexity is described. While used primarily for simulating and evaluating robotic manipulators, it has also been applied to the animation of biological models. One of the key issues discussed involves a flexible and intuitive approach to the motion specification for that large class of objects which possess redundant degrees of freedom.

I. INTRODUCTION

The computer graphic simulation of articulated objects has a wide variety of diverse applications. In robotics, simulations are invaluable for manipulator design, motion planning and control evaluation [19]. The simulation of humans and other biological systems increases our understanding of motion coordination and aids in the design of prosthetics and rehabilitation methods. Finally, simulation techniques for articulated objects have useful applications in the arts, such as interactive aid in the choreography of dance or computer generated animation in the visual arts.

The software described in this paper, collectively referred to by the acronym SAM, has been developed over the past two years for the study of redundant manipulators at the Electrical Engineering Department of The Ohio State University. Redundant manipulators are those which, like human appendages, possess more than the six degrees of freedom necessary to arbitrarily position and orient an object in space. The lack of commercially available redundant manipulators made simulation studies the only viable method of evaluating motion control coordination. While motivated by the robotics application, the computationally efficient and flexible methods of motion coordination for arbitrarily complex articulated objects can be utilized in other applications. Portions of SAM have already been incorporated into an animation system which allows the simulation of various different gaits for multilegged figures [4].

II. DEFINITION OF ARTICULATED OBJECTS

The first step in controlling the motion of articulated objects is to have a general method of defining the location and type of degrees of freedom present in the object. SAM uses the kinematic notation presented by Denavit and Hartenberg in 1955 [3] which defines these relationships for a large class of articulated objects. Each degree of freedom within an articulated object is assigned a unique coordinate system with the relationship between adjacent coordinate systems defined by four parameters. The degrees of freedom within the object, either rotary or prismatic, are referred to as joints while the interconnecting portions are called links.

The four parameters used to specify the relationships between coordinate systems are the length of the link a, the twist of the link α, the distance between links d and the angle between links θ. The definition of these parameters for an arbitrary articulated chain is illustrated in Fig. 1. A simple procedure for defining the origins of the various coordinate systems is given in the literature [17]. This functionality definition easily incorporates physical joints with multiple degrees of freedom located at a single point. It is important to note that these parameters specify the functionality of the degrees of freedom within the object only and in no way restrict its physical appearance. SAM uses a separate point-polygon list file referenced to the respective coordinate frame to model the actual shape of the link.

Given the above specification of coordinate frames, the relationship between adjacent coordinate frames is given by a rotation of θ, followed by translations of d and a, and a final rotation of α. By concatenating these transformations, it can be shown [17] that the relationship between adjacent coordinate frames $i-1$ and i denoted by $A_{i-1,i}$ is given by the homogeneous transformation

$$A_{i-1,i} = \left[\begin{array}{c|c|c|c} \mathbf{n} & \mathbf{o} & \mathbf{a} & \mathbf{p} \\ \hline 0 & 0 & 0 & 1 \end{array}\right], \tag{1}$$

where

$$\begin{array}{llll} n_x = c\theta_i, & o_x = -c\alpha_i s\theta_i, & a_x = s\alpha_i s\theta_i, & p_x = a_i c\theta_i, \\ n_y = s\theta_i, & o_y = c\alpha_i c\theta_i, & a_y = -s\alpha_i c\theta_i, & p_y = a_i s\theta_i, \\ n_z = 0, & o_z = s\alpha_i, & a_z = c\alpha_i, & p_z = d, \end{array}$$

with s and c denoting the sine and cosine functions, respectively. In this derivation of the homogeneous transformation, the independent variable is θ for rotary joints and d for prismatic joints. By multiplying ad-

† This work is the result of research supported by the National Science Foundation under Grant ECS-8121519 and a National Science Foundation graduate fellowship. Work was performed at the Department of Electrical Engineering, The Ohio State University, 2015 Neil Avenue, Columbus, OH 43210.

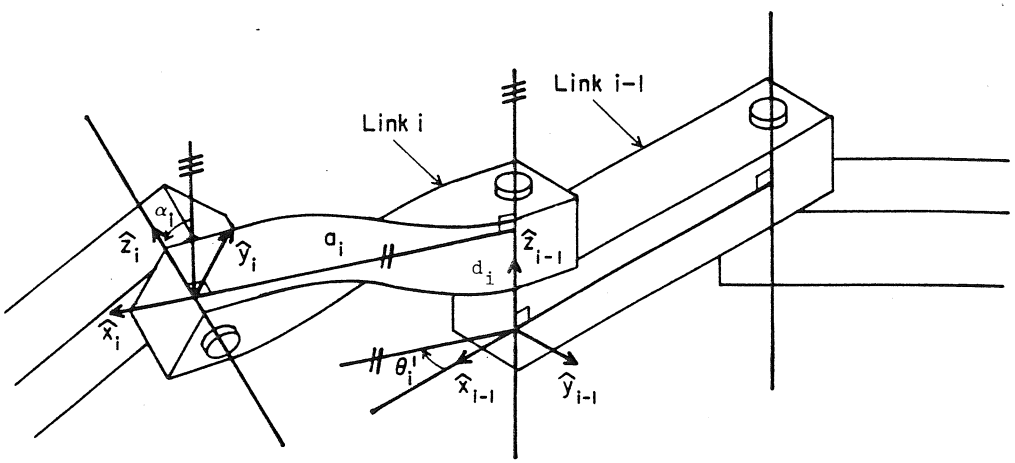

Fig. 1. An arbitrary articulated chain illustrating the definition of the Denavit and Hartenberg parameters for describing the degrees of freedom in articulated objects.

jacent link transformations, the homogeneous transformation between any two coordinate systems i and j is computed by using

$$A_{i,j} = A_{i,i+1}A_{i+1,i+2}\cdots A_{j-1,j}. \tag{2}$$

This formulation, therefore, specifies the configuration of the object in terms of the variables describing the state of its internal degrees of freedom, that is, the value of the joint variables θ and d. The next step in developing coordinated motion control is to relate the rate of change of these joint variables, their velocity, to a set of variables in which the user can conveniently define the desired action of the articulated object.

III. DEFINITION AND GENERATION OF JACOBIAN MATRICES

SAM's motion control philosophy relies on a hierarchical approach to motion descriptions. The user interacts with the system at the most intuitive level of task description. For example, to complete the task of opening a door, the user specifies the location of the door handle and the commands to move to that position, grasp and turn. The system then computes the required joint motions (seven rotations distributed throughout the shoulder, elbow and wrist in a human arm) in order to complete the task. The advantages of this concept of specifying motion in a useful, task-oriented coordinate system, typically Cartesian, with individual joint motions being coordinated by computer control has been recognized in the robotics and prosthetics field. It is typically referred to as resolved motion rate control, a concept presented in the literature by Whitney [25, 26].

Essential to this concept of resolved motion rate control is the Jacobian. The Jacobian matrix J relates the motion of a reference coordinate frame attached to the articulated object to the joint variable velocities through the equation

$$\dot{\mathbf{x}} = J\dot{\theta}, \tag{3}$$

where $\dot{\mathbf{x}}$ is typically a six-dimensional vector describing the desired translational and rotational motion of a point on the object (e.g. motion of a hand) and $\dot{\theta}$ is an n-dimensioal vector representing the joint velocities, n being the number of degrees of freedom in the articulated object.

While a number of techniques for calculating the Jacobian have been studied [15], a particularly elegant and efficient method is available if the desired motion, $\dot{\mathbf{x}}$, is described in terms of the screw axis variables ω and μ [24]. When described in terms of these variables, it can be shown [23] that the Jacobian is given by

$$J = \begin{bmatrix} \mathbf{p}_1 \times \mathbf{a}_1 & \mathbf{p}_2 \times \mathbf{a}_2 & \cdots & \mathbf{p}_n \times \mathbf{a}_n \\ \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_n \end{bmatrix}, \tag{4}$$

where $\mathbf{a}_i$ and $\mathbf{p}_i$ are the third and fourth columns, respectively, of the homogeneous transformation matrix $A_{0,i-1}$. The first column of the Jacobian is given by $\mathbf{p}_1 = [000]^T$ and $\mathbf{a}_1 = [001]^T$.

The desired motion of a reference frame attached to the articulated object, specified in a task-oriented coordinate system by a translational velocity $\mathbf{v}_t$ and a rotational velocity ω_t, can easily be expressed in screw axis variables by applying the equations

$$\omega = R\omega_t, \tag{5}$$

$$\mu = R\mathbf{v}_t - \omega \times \mathbf{p}, \tag{6}$$

where R is the upper 3×3 rotation submatrix of the homogeneous transformation describing the reference frame whose velocity is being specified and $\mathbf{p}$ is the position of this reference frame given by the fourth column of its homogeneous transformation. This formulation of the Jacobian results in a minimum amount of computation since the majority of the work has already been done in generating the homogeneous transformations required to display the object.

The objective of defining a smooth linear transformation between the two sets of variables, the functional set of joint variables describing the degrees of freedom in the object and the convenient task-oriented set of variables which the user specifies, is thus achieved by the Jacobian. In this manner SAM avoids the cumbersome nonlinear relationship between position specification of the two sets of variables. It is clear from eqn (3) that the desired motion specified by the user, $\dot{\mathbf{x}}$, can be achieved by applying the joint velocities specified by

$$\dot{\theta} = J^{-1}\dot{\mathbf{x}} \tag{7}$$

if J is square and nonsingular. For the vast majority of cases, however, the number of degrees of freedom will not match the dimension of the specified velocity. In these cases the Jacobian is rectangular and the inverse is not defined. Although the inverse of the Jacobian in these cases does not exist, there do exist generalized inverses which provide useful solutions to eqn (3). This is the topic of the next section.

IV. APPLICATION OF PSEUDOINVERSES

Classes of generalized inverses are usually defined in terms of the following four properties [1],

$$AGA = A, \tag{8}$$

$$GAG = G, \tag{9}$$

$$(GA)^* = GA, \tag{10}$$

$$(AG)^* = AG, \tag{11}$$

where A is an $m \times n$ matrix, G is its generalized inverse and * denotes the complex conjugate transpose. The properties of these various generalized inverses can be found in the literature [1, 2, 13]. This discussion will be limited to the properties of the generalized inverse which satisfies all four properties (8)–(11), and will be referred to as the pseudoinverse, denoted by A^+. This inverse is sometimes referred to as the Moore–Penrose inverse, due to its independent discovery by E. H. Moore and R. Penrose in 1920 and 1955, respectively.

The Jacobian method of motion control requires the solution of general systems of equations of the form of eqn (3), where J is an $m \times n$ Jacobian matrix and $\dot{\theta}$ and $\dot{\mathbf{x}}$ are n and m dimensional vectors, respectively. There may exist no solution, one solution, or an infinite number of solutions for a given J and $\dot{\mathbf{x}}$ depending on the number of degrees of freedom involved, the dimension of the specified desired velocity and the present configuration of the object. In all cases, the pseudoinverse solution given by

$$\dot{\theta} = J^+\dot{\mathbf{x}} \tag{12}$$

yields a best approximate solution [18], which is defined as follows [16, 18]:

Definition. The vector $\dot{\theta}_0$ is a best approximate solution of eqn (3) if for all vectors $\dot{\theta}$ either

$$\|J\dot{\theta} - \dot{\mathbf{x}}\| > \|J\dot{\theta}_0 - \dot{\mathbf{x}}\| \quad \text{or}$$

$$\|J\dot{\theta} - \dot{\mathbf{x}}\| = \|J\dot{\theta}_0 - \dot{\mathbf{x}}\| \quad \text{and} \quad \|\dot{\theta}\| \geq \|\dot{\theta}_0\|,$$

where $\|\dot{\theta}\|$ denotes the Euclidean norm of $\dot{\theta}$. The vector $\dot{\theta}_0$ is also referred to as the least-squares solution of minimum norm. The existence and uniqueness of the least-squares solution of minimum norm given by eqn (12) has been proven by Penrose [18]. In the case where J is square and nonsingular, the solution given by eqn (12) is identical to that of eqn (7).

The physical interpretation of the pseudoinverse solution as applied to the Jacobian control of articulated objects consists of three distinct cases. The first case is where there do not exist enough independent degrees of freedom in the object in order to achieve the specified motion. In this case, the pseudoinverse will provide the solution which is as close as possible to the desired motion that the available degrees of freedom will allow. It is important to note that the nonexistence of an exact solution is a result of the physical structure of the problem and not due to the mathematical formulation. The second case occurs when the number of independent degrees of freedom exactly matches the dimension of the specified motion. Six degrees of freedom are sufficient to fully specify translational and rotational velocity in three dimensions. In this case, the pseudoinverse will return the unique solution of joint velocities which exactly achieves the desired motion. The third, and most interesting, case occurs when the object contains so-called redundant degrees of freedom with respect to the specified motion. This case includes humans, whose appendages contain redundant degrees of freedom which contribute to their flexibility and dexterity. In this case, there exist an infinite set of joint velocities which will achieve the desired motion. The pseudoinverse solution will be the one which achieves the desired motion with the smallest amount of joint movement, computed in a least-squares sense.

A number of different methods for calculating pseudoinverses have been discussed in the literature [5, 7, 14, 20]. An excellent discussion of the numerical considerations involved in computing the pseudoinverse is presented by Noble [14]. The simplest expressions for a pseudoinverse appear for matrices known to be of full rank. For an $m \times n$ matrix A of rank r, the expression for the pseudoinverse is given by

$$A^+ = (A^*A)^{-1}A^* \quad \text{if} \quad m > n = r \tag{13}$$

and

$$A^+ = A^*(AA^*)^{-1} \quad \text{if} \quad r = m < n \text{ [6].} \tag{14}$$

By making intelligent assumptions about the rank of the Jacobian for a given configuration, SAM is able to apply a computationally efficient Gaussian elimination technique to the above formulation, thereby removing the explicit inverse calculation.

For matrices of unknown rank, decomposition strategies have been developed. If the matrix A is ex-

pressed in the form $A = BC$, where B is $m \times r$ and C is $r \times n$, and B and C are both of rank r, then the pseudoinverse is given by

$$A^+ = C^*(CC^*)^{-1}(B^*B)^{-1}B^*. \tag{15}$$

Although the factorization is not unique [14], the resultant pseudoinverse is. A recursive method for generating the pseudoinverse has been developed by Greville [5]. Other methods utilizing Householder transformations, Gram–Schmidt orthogonalization, or singular value decomposition have also been developed.

Thus the pseudoinverse plays a key role in SAM's ability to allow the user to concentrate on goal-directed [9] motion specification without concern over geometry or configuration constraints which may result in loss of independent degrees of freedom. An efficient pseudoinverse implementation gives SAM the ability to generate the appropriate coordinated joint velocities required to best achieve the desired task under any circumstances.

V. MULTIGOAL DEFINITION OF MOTION CONTROL

It has been shown in the previous section that the pseudoinverse is instrumental in achieving the primary goal in the hierarchy of motion definition. Yet in the case of redundant degrees of freedom, the pseudoinverse solution is only one of an infinite number of solutions. These extra degrees of freedom which are not required to achieve the primary goal are available for achieving secondary goals specified by the user. The secondary goals are typically used to control the configuration of the articulated object while it completes the desired task specified by the primary goal. This section presents formulations which generalize on the pseudoinverse solution, retaining all of the characteristics described above, as well as providing the added versatility of permitting the user to specify additional constraints on how a motion is to occur for situations where redundant degrees of freedom are present.

It can be shown [6] that the general solution to a system of linear equations described by eqn. (3) is given by

$$\dot{\theta} = J^+\dot{\mathbf{x}} + (I - J^+J)\mathbf{z}, \tag{16}$$

where I is an $n \times n$ identity matrix and $\mathbf{z}$ is an arbitrary vector in $\dot{\theta}$ space. That is, the ith element in the vector $\mathbf{z}$ can be described as the desired joint velocity for the ith joint. The homogeneous portion of this solution is described by a projection operator $(I - J^+J)$ that maps the arbitrary vector $\mathbf{z}$ into the null space of the transformation. That is, the projection operator allows the user to choose secondary goals, described by the vector $\mathbf{z}$, and blend them into the desired manner of motion without affecting the primary goal of task completion. In effect what occurs is that the projection operator removes all of the degrees of freedom which would disturb achievement of the primary goal. Any remaining degrees of freedom, in so-called redundant situations, are used to achieve the goal specified by the vector $\mathbf{z}$. In cases where there are no redundant degrees of freedom, the projection operator is the null matrix and the general formulation reduces to the simple pseudoinverse solution.

Some of the many desirable criteria which can be expressed in this form have already been documented in the literature. Liegeois [10] specified a secondary goal of joint availability, that is, keeping the joints as close to their center position as possible. He has shown that if one specifies the vector $\mathbf{z}$ in eqn (16) by

$$\mathbf{z} = -\alpha\nabla H(\theta), \tag{17}$$

where α is a real positive scalar gain value and ∇H is the gradient of a smooth function, the homogeneous solution can be used to minimize H. If the function H is defined as

$$H(\theta) = \sum [(\theta_i - \theta_{ci})/\Delta\theta_i]^2, \tag{18}$$

where θ_i is the ith joint angle, θ_{ci} is the center position of the ith joint angle and $\Delta\theta_i$ is the desired maximum one-sided excursion of the ith joint angle, then the manipulator will achieve the given primary goal in a manner which keeps the joints as close as physically possible to their center position. Alternate optimization criteria have also been applied [8].

Expanding on Liegeois' formulation, Ribble [21] has demonstrated the use of a dynamic secondary goal variation in order to simulate human walking motion. By varying the weighting on joints during different phases of the walking cycle, a realistic human gait can be achieved. A frame from a simulation illustrating the motion of climbing stairs is presented in Fig. 2.

Thus by implementing eqn (16), SAM provides the user with the flexibility of specifying any of the above secondary criteria, or any other secondary goal that can be described in joint space. There are, however, some secondary goals which can be more intuitively described in Cartesian worldspace coordinates. A typical example of such a goal is that of obstacle avoidance. The distance between the articulated object and other objects in the world is much more easily defined in worldspace coordinates rather than joint space ones. Clearly, obstacle avoidance is a useful secondary goal which again should be satisfied under the constraints imposed by the primary goal. The authors have shown that the formulation to achieve this specification of motion is given by

$$\dot{\theta} = J_1^+\dot{\mathbf{x}}_1 + (J_2(I - J_1^+J_1))^+(\dot{\mathbf{x}}_2 - J_2J_1^+\dot{\mathbf{x}}_1), \tag{19}$$

where

$\dot{\mathbf{x}}_1$ = primary goal velocity,
J_1 = Jacobian for primary goal reference frame,
$\dot{\mathbf{x}}_2$ = secondary goal velocity,
J_2 = Jacobian for secondary goal reference frame.

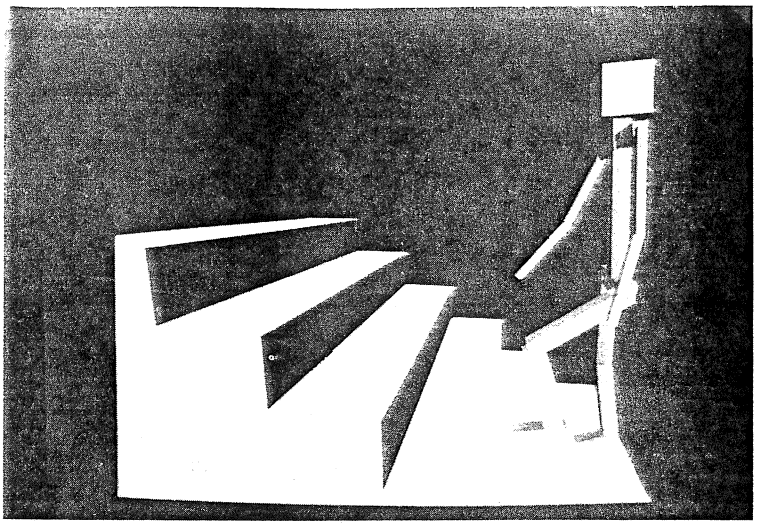

Fig. 2. A frame from a film simulating human walking motion illustrating the use of a motion control formulation using both the pseudoinverse and secondary goals.

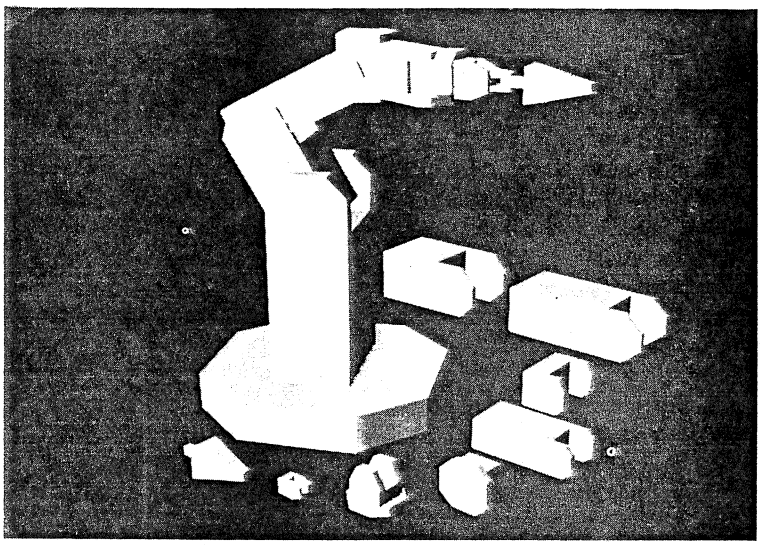

Fig. 4. Independently described by point-polygon list data files, the individual links are assembled with reference to their respective coordinate frames defined in Fig. 3.

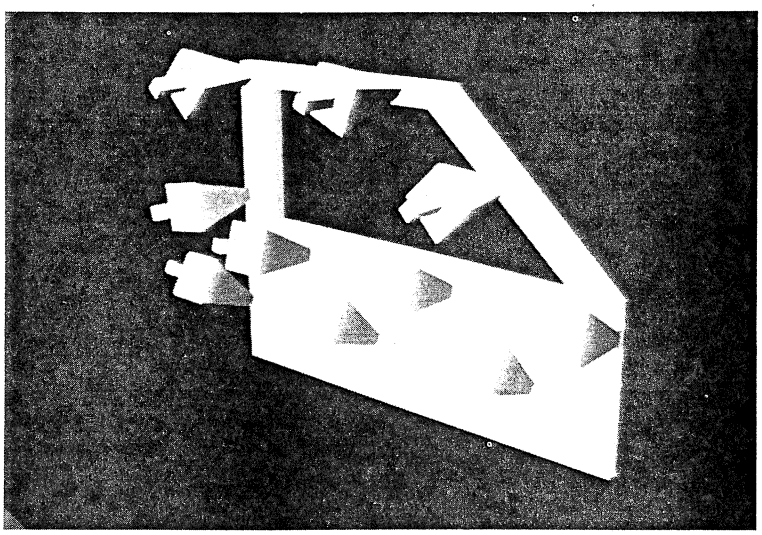

Fig. 5. The disconnected end effector is manipulated by the animator into its desired configurations which define the required trajectory.

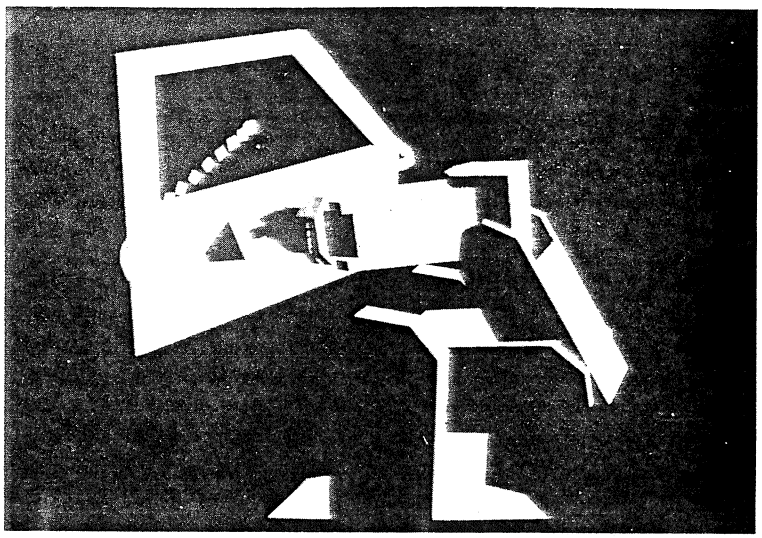

Fig. 6. Using only the selected configurations of the end effector (see Fig. 5) SAM generates the entire trajectory along with the nine sets of joint rotations required to smoothly achieve it.

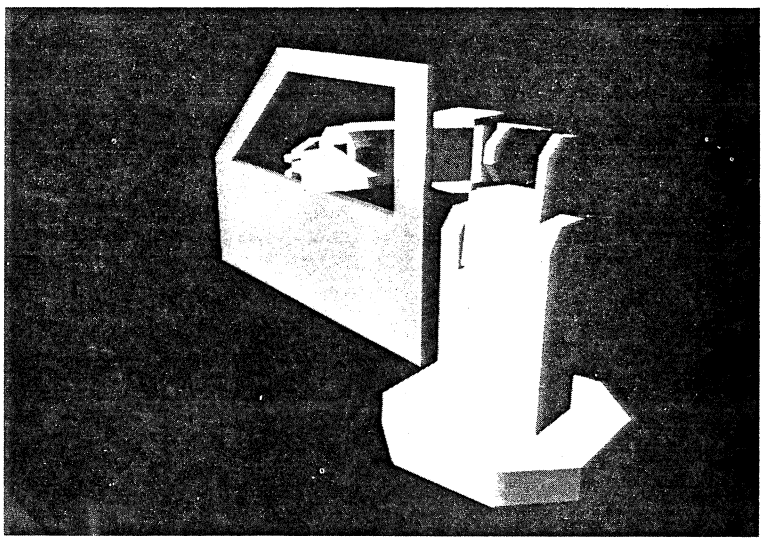

Fig. 8. Given only the position of the door as input SAM generates a trajectory which avoids contact with the obstacle while simultaneously tracking the desired end effector trajectory required to paint the door.

The interpretation of the individual terms and details of the numerical implementation can be found in the literature [12].

Thus eqns (16) and (19) provide SAM with an elegant mathematical formulation for incorporating primary and secondary goals of varying types into the joint motion coordination. SAM can combine multiple secondary goals which have dynamically varying priorities to allow for automatic adaptation to external stimuli. This provides the user with the versatility to determine the priorities of motion with SAM smoothly combining them into a single synergistic set of joint velocities which results in the desired effect.

VI. VELOCITY PROFILES FOR MOTION DEFINITION

The above sections have presented the formulations which SAM uses to generate smooth coordinated joint movements from a velocity-oriented specification of motion. This section describes software which SAM uses to provide an intuitive and flexible approach to describing the desired task-oriented motion. SAM has utilized to date three methods of motion specification, namely, an interactive joystick specified velocity, velocity profile definitions, and a position error formulation.

The joystick specified velocity method is the most primitive of the motion specification techniques, yet it remains an invaluable option to the user. Using a three-axis joystick the user can fully specify a desired linear or rotational velocity for any portion of the articulated object and obtain immediate feedback regarding the response of the articulations. This method of motion specification has been particularly useful in evaluating the effect of different secondary goal functions on the manner in which the primary goal is achieved.

The second method of motion specification which SAM utilizes involves velocity profiles, that is, a function which specifies the velocity command for each point in time for the reference frame under control, which results in a nominal motion. For example, the walking motion of a biped can be decomposed into two six-dimensional velocity profiles, one for each foot. The application of these velocity profiles as the primary goals result in motion which includes the inherent aspects of walking but still allows the application of various different secondary goals to obtain specific unique characteristics. By applying the rules of physics to the generation of such velocity profiles, the dynamics of the articulated object can be simulated for realistic results. An additional method of customizing generic motion plans is the addition of small perturbations to the nominal velocity profiles.

Finally, to specify motion for tasks inherently described in positional terms, SAM uses a position error formulation as the driving primary goal. As an example, consider the task of picking up a glass and filling it with water. Using the position error formulation, SAM defines this task as a specified position and orientation for the glass which the hand must reach at time t_1 and a second position and orientation under a faucet to be obtained at time t_2. Using the initial state of the hand as an additional constraint, a spline [22] of a specified order of continuity is fitted through the required positions and orientations. These splines are then evaluated at intervals of the simulation cycle time in order to determine the desired configuration of the hand at each instant. The incremental position, $\mathbf{e}_p$, and orientation, $\mathbf{e}_o$, between the current actual configuration of the hand, denoted by the subscript a, and the next desired configuration, denoted by the subscript d, given by [11]

$$\mathbf{e}_p(t) = \mathbf{p}_d(t + \Delta t) - \mathbf{p}_a(t) \tag{20}$$

$$\mathbf{e}_o(t) = 0.5[\mathbf{n}_a(t) \times \mathbf{n}_d(t + \Delta t) + \mathbf{o}_a(t) \times \mathbf{o}_d(t + \Delta t) + \mathbf{a}_a(t) \times \mathbf{a}_d(t + \Delta t)] \tag{21}$$

is then used as the primary goal velocity command.

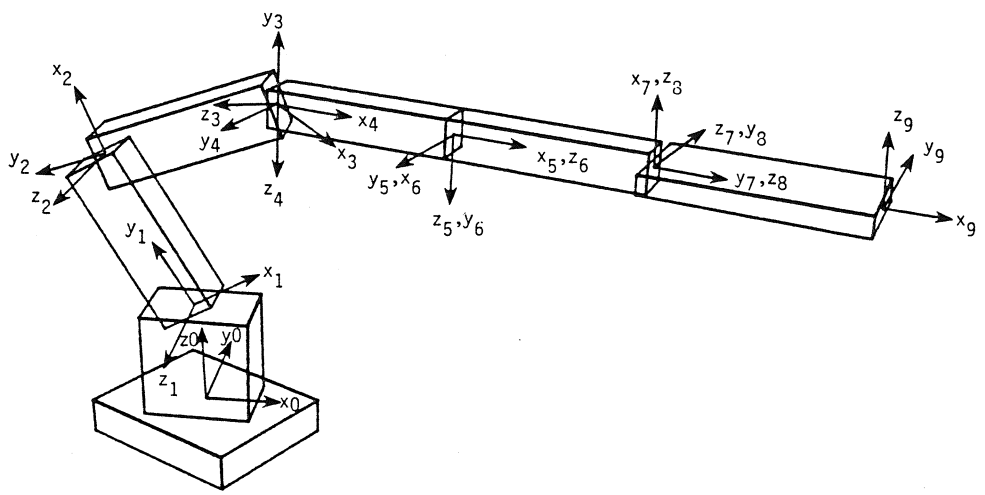

Fig. 3. A block diagram illustrating the relationship between individual degrees of freedom for the Denavit and Hartenberg parameters chosen in Table 1.

Table 1. Denavit and Hartenberg parameters for the robot used in the animation example of Section VII

Link	α (deg)	a (m)	d (m)
1	90	0	1.0
2	0	1.5	0
3	0	1.5	0
4	90	0	0
5	0	1.0	0
6	90	0	0
7	90	0	1.0
8	90	0	0
9	0	1.0	0

This formulation also automatically prevents accumulation of errors introduced by numerical computations.

All three of the above methods of motion definition can be used to specify either primary or secondary goal trajectories. In addition, all three methods can be mixed in a single task without loss of continuity, thus allowing the user the flexibility to describe the task in the most intuitive manner.

VII. SAMPLE ANIMATION SESSION

To illustrate a user's level of interaction with SAM in the development of a simulation, this section presents a simple example. The particular task chosen to be simulated is that of a robot spray painting a car door.

The first step is for the user to decide the functionality and shape of the robot arm to be simulated. Using the parameters of length a, twist α, and offset d as input parameters, SAM produces a block diagram of the arm's functionality. Figure 3 illustrates this diagram including the individual link coordinate axis for the parameters listed in Table 1. Once the user is satisfied with the location of the articulations describing the functionality of the arm, SAM now allows him to specify the physical structure of the individual links using a point list polygon list format. Figure 4 depicts

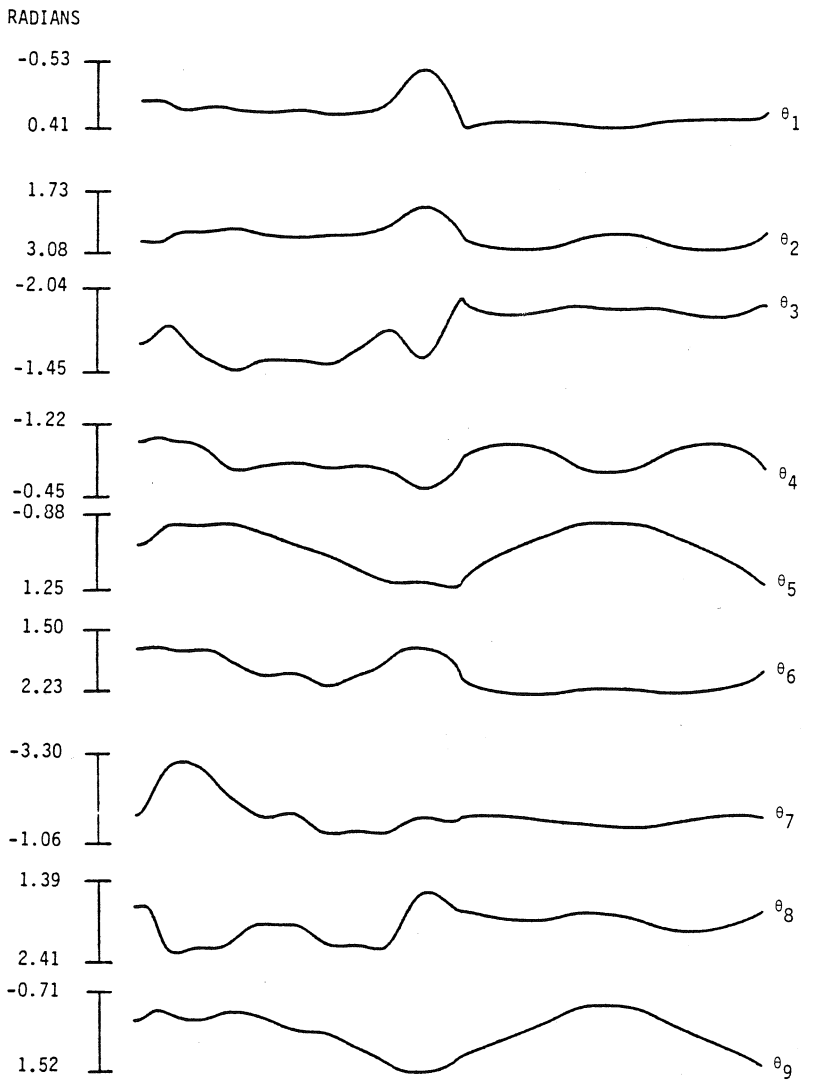

Fig. 7. Joint angle positions as a function of time required to complete the specified trajectory.

the individual stylized link structures along with the assembled arm.

Having completed designing the "actor," the user turns to control of the action. Since the desired task involves painting a car door located at a specific position, the user chooses the position error formulation for describing the desired motion. The user then defines selected positions and orientations for the end effector of the robot relative to the car door which are required to complete the task. The output from this procedure is shown in Fig. 5. Given this information, SAM fits a spline through these points and uses the pseudoinverse of the Jacobian for each increment to compute the required incremental joint rotations.

At this point SAM is ready to play back the currently defined action. Figure 6 shows a single frame of the simulation with the resultant end effector trajectory superimposed. The continuity of the joint coordination is illustrated by the graphs of Fig. 7. The user can modify the trajectory of the end effector by simply adding or deleting spline knots. The process is repeated until the action is as desired.

Having completed the specification of the primary goal, the user can now try different secondary goals to modify the arm's configuration while painting the car door. For this particular example the desired posture of the arm was specified to maximize the distance to objects in the workspace. This goal was easily implemented through eqn (19) by updating $\dot{x}_2$ each cycle time using the current state of the environment. The computation cycle time for this implementation was only 102 ms on a PDP 11/70 [12]. This allows the arm to act with additional intelligence in avoiding the door while painting. The effectiveness of this secondary goal is illustrated in Fig. 8. Having completed the specification of the desired action, the user can now have SAM produce a 16 mm film of the resultant simulation.

VIII. CONCLUSION

The main advantage of SAM's approach to motion coordination is the ability of the user to interact with the motion specification at different levels of complexity. The hierarchical approach to motion coordination, from primary goals to secondary goals to individual joint control, allows concentration on particular aspects of the motion without disturbing previously defined requirements. The multiple methods for specifying motions allow the user the flexibility to deal with the desired task in the most intuitive set of variables. Finally, by employing computationally efficient formulations of the control equations, SAM obtains reasonable computation cycle times.

Acknowledgement—The authors wish to acknowledge the assistance of Eric Ribble, who performed the work on simulating walking motion.

REFERENCES

1. A. Ben-Israel and T. N. E. Greville, *Generalized Inverses: Theory and Applications.* Wiley–Interscience, New York (1974).
2. T. L. Boullion and P. L. Odell, *Generalized Inverse Matrices.* Wiley–Interscience, New York (1971).
3. J. Denavit and R. S. Hartenberg, A kinematic notation for lower-pair mechanisms based on matrices. *ASME J. Appl. Mech.* **23,** 215–221, (1955).
4. M. Girard and A. A. Maciejewski, Computational modeling for the computer animation of legged figures. *Comput. Graphics* **19**(3), (1985).
5. T. N. E. Greville, Some applications of the pseudoinverse of a matrix. *SIAM Rev.* **2**(1) (1960).
6. T. N. E. Greville, The pseudoinverse of a rectangular or singular matrix and its applications to the solutions of systems of linear equations. *SIAM Rev.* **1**(1) (1959).
7. R. J. Hanson and C. L. Lawson, Extensions and applications of the Householder algorithm for solving linear least squares problems. *Math. Comput.* **23,** 787–812 (1969).
8. C. A. Klein and C. H. Huang, Review of pseudoinverse control for use with kinematically redundant manipulators. *IEEE Trans. Systems, Man, Cybern.* **SMC-13**(2), 245–250 (1983).
9. J. U. Korein and N. I. Badler, Techniques for generating the goal-directed motion of articulated structures. *IEEE Comput. Graphics Appl.* **2**(9), 71–81 (1982).
10. A. Liegeois, Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Trans. Systems, Man and Cybern.* **SMC-7**(12) (1977).
11. J. Y. S. Luh, M. W. Walker and R. P. C. Paul, Resolved-acceleration control of mechanical manipulators. *IEEE Trans. Autom. Control* **AC-25**(3) (1980).
12. A. A. Maciejewski and C. A. Klein, Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *Int. J. Robotics Res.* **4**(3) (1985).
13. B. Noble, *Applied Linear Algebra.* Prentice–Hall, Englewood Cliffs, NJ (1969).
14. B. Noble, Methods for computing the Moore–Penrose generalized inverses, and related matters. In *Generalized Inverses and Applications* (Edited by M. Z. Nashed), pp. 245–301. Academic Press, New York (1975).
15. D. E. Orin and W. W. Schrader, Efficient Jacobian determination for robot manipulators. In *Robotics Research: The First International Symposium* (Edited by M. Brady and R. Paul), pp. 727–734 (1984).
16. E. E. Osborne, Smallest least squares solutions of linear equations. *SIAM J. Numer. Anal. Ser. B,* **2**(2), 300–307 (1965).
17. R. Paul, *Robot Manipulators.* MIT Press, Cambridge, MA (1981).
18. R. Penrose, On best approximate solutions of linear matrix equations. *Proc. Cambridge Philos. Soc.* **52,** 17–19 (1956).
19. R. M. V. Perez, Computer graphics as an aid to a robot dynamical simulation analysis. *Comput. Graphics* **8**(3), 265–268 (1984).
20. G. Peters and J. H. Wilkinson, The least squares problem and pseudo-inverses. *Comput. J.* **13**(3) (1970).
21. E. A. Ribble, Synthesis of human skeletal motion and the design of a special-purpose processor for real-time animation of human and animal figure motion. Masters thesis, The Ohio State University, (June 1982).
22. D. F. Rogers and J. A. Adams, *Mathematical Elements for Computer Graphics.* McGraw–Hill, New York (1976).
23. K. J. Waldron, Geometrically based manipulator rate control algorithms. *Mech. Machine Theory* **17**(6), 379–385 (1982).
24. K. J. Waldron, The use of motors in spatial kinematics. *Proc. of the IFToMM Conf. on Linkages and Computer Design Methods,* Bucharest, B, pp. 535–545 (June 1973).
25. D. E. Whitney, Resolved motion rate control of manipulators and human prostheses. *IEEE Trans. Man-Machine Systems* **MM2-10**(2), 47–53 (1969).
26. D. E. Whitney, The mathematics of coordinated control of prostheses and manipulators. *J. Dynamic Systems, Measurement, and Control, Trans. ASME* **94,** Ser. G, 303–309 (1972).